NOKIA

7450 ETHERNET SERVICE SWITCH
7750 SERVICE ROUTER
7950 EXTENSIBLE ROUTING SYSTEM
VIRTUALIZED SERVICE ROUTER

SYSTEM MANAGEMENT GUIDE
RELEASE 20.10.R1

3HE 15829 AAAD TQZZA 01

Issue: 01

October 2020

Nokia is committed to diversity and inclusion. We are continuously reviewing our customer documentation and consulting with standards bodies to ensure that terminology is inclusive and aligned with the industry. Our future customer documentation will be updated accordingly.

Table of Contents

1 Getting Started

1.1 About This Guide

This guide describes system concepts and provides configuration explanations and examples to configure SR OS boot option file (BOF), file system and system management functions.7450 ESS, 7750 SR, 7950 XRS, and VSR Classic CLI Command Reference Guide.

This guide is organized into functional chapters and provides concepts and descriptions of the implementation flow, as well as Command Line Interface (CLI) syntax and command usage.

The topics and commands described in this document apply to the:

- 7450 ESS
- 7750 SR
- 7950 XRS
- VSR

Table 1 lists the available chassis types for each SR OS router.

Table 1 **Supported SR OS Router Chassis Types**

7450 ESS	7750 SR	7950 XRS
• 7450 ESS-7/12	• 7750 SR-a4/a8 • 7750 SR-1e/2e/3e • 7750 SR-12e • 7750 SR-1 • 7750 SR-7/12 • 7750 SR-1s/2s • 7750 SR-7s/14s	• 7950 XRS-16c • 7950 XRS-20/40 • 7950 XRS-20e

For a list of unsupported features by platform and chassis, refer to the *SR OS 20.x*.R*x* Software Release Notes, part number 3HE 16194 000*x* TQZZA.

Command outputs shown in this guide are examples only; actual displays may differ depending on supported functionality and user configuration.

Note: The SR OS CLI trees and command descriptions have been removed from this guide and can now be found in the following guides:

- *7450 ESS, 7750 SR, 7950 XRS, and VSR Classic CLI Command Reference Guide*
- *7450 ESS, 7750 SR, 7950 XRS, and VSR Clear, Show, and Tools Command Reference Guide* (for both MD-CLI and Classic CLI)
- *7450 ESS, 7750 SR, 7950 XRS, and VSR MD-CLI Command Reference Guide*

Note: This guide generically covers Release 20.*x*.R*x* content and may contain some content that will be released in later maintenance loads. Refer to the *SR OS 20.x*.R*x* Software Release Notes, part number 3HE 16194 000*x* TQZZA, for information about features supported in each load of the Release 20.*x*.R*x* software.

1.2 Router Configuration Process

Table 2 lists the tasks necessary to configure system security and access functions and logging features on the 7450 ESS, 7750 SR, and 7950 XRS platforms. Each chapter in this book is presented in an overall logical configuration flow. Each section describes a software area and provides CLI syntax and command usage to configure parameters for a functional area.

Table 2 **Configuration Process**

Area	Task	Section
System security	Configure system security	Security Configurations
	Configure RADIUS	RADIUS Configurations
	Configure TACACS+	TACACS+ Configurations
	Configure LDAP	LDAP Configurations
	Configure login controls	Configuring Login Controls
Network management	Configure SNMP elements.	Configuring SNMP with CLI
Secure network management	Configure NETCONF elements	NETCONF
Operational functions	Configure event and accounting logs	Configuring Logging with CLI
Data management	Configure sFlow elements	sFlow
Network monitoring	Configure telemetry	gRPC
Network security	Configure TLS server and client	Common Configuration Tasks

Table 2 **Configuration Process (Continued)**

Area	Task	Section (Continued)
Equipment monitoring	Configure facility alarms	Configuring Logging with CLI

Note: All features are supported on all SR OS platforms (7750 SR, 7450 ESS, and 7950 XRS) unless indicated otherwise.

1.3 Node Management Using VPRN

While customarily node management is operated either via the out-of-band interface or in-band via the Base routing instance, it is also possible to manage the node using a VPRN. Both IPv4 and IPv6 are supported.

The following management plane clients are supported using VPRN:

- DNS
- gRPC (dial-out telemetry)
- RADIUS
- SNMP (traps)
- SSH
- Syslog
- TACACS+
- Telnet

The following servers are supported using VPRN:

- FTP
- gRPC
- NETCONF (including notifications)
- SNMP
- SSH
- Telnet

Refer to section 3.2.16 in the Layer 3 Services Guide for further details.

2 Security

2.1 Authentication, Authorization, and Accounting

This chapter describes authentication, authorization, and accounting (AAA) used to monitor and control network access on routers. Network security is based on a multi-step process. The first step, authentication, validates a user's credentials. The second step is authorization, which allows the user to access and execute commands at various command levels based on profiles assigned to the user.

Another step, accounting, keeps track of the activity of a user who has accessed the network. The type of accounting information recorded can include a history of the commands executed, the amount of time spent in the session, the services accessed, and the data transfer size during the session. The accounting data can then be used to analyze trends, and also for billing and auditing purposes.

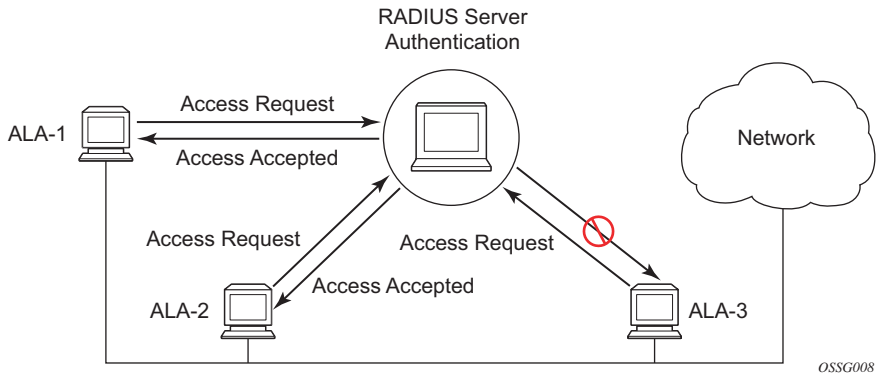
Configure routers to use local, Remote Authentication Dial In User Service (RADIUS), Lightweight Directory Access Protocol (LDAP), or Terminal Access Controller Access Control System Plus (TACACS+) security to validate users who attempt to access the router by console, telnet, SSH, NETCONF, FTP, and more. Select the authentication order, which determines the authentication method to try first, second, third, or fourth.

The router supports the following security features:

- Local security can be implemented for authentication and authorization.
- LDAP can be implemented for authentication.
- RADIUS can be used for authentication, authorization, and accounting.
- TACACS+ can be used for authentication, authorization, and accounting.

Figure 1 depicts end user access-requests sent to a RADIUS server. After validating the user names and passwords, the RADIUS server returns an access-accept message to the users on ALA-1 and ALA-2. The user name and password from ALA-3 could not be authenticated, thus access was denied.

Figure 1 **RADIUS Requests and Responses**



2.1.1 Authentication

Authentication validates a user's credentials when a user attempts to log in.

When a user attempts to log in through the console, FTP, or other methods, the client sends credentials to the router. Based on the received credentials, the router creates and sends an authentication request to a RADIUS, TACACS+, LDAP, or local database. The order in which the router tries different types of AAA servers and local databases is defined by the configured authentication order.

Transactions between the router and a RADIUS or TACACS+ server are authenticated through the use of a shared secret. The secret is never transmitted over the network. TLS can be used for the connection between the router and the LDAP server. User passwords are sent encrypted between the client and the AAA (RADIUS, TACACS+, or LDAP) server which prevents someone snooping on an insecure network to learn password information.

If the AAA server (of the chosen authentication method) does not respond within a specified time, the router issues the access request to the next configured servers of the same authentication method. Each AAA server must be configured identically to guarantee consistent results.

If any AAA server rejects the authentication request, it sends an access reject message to the router. In this case, no access request is issued to any other AAA servers of the chosen authentication method. However, if other authentication methods, such as TACACS+ and/or local, are configured and the option exit-on-reject is not set, then these methods are attempted. If no other authentication methods are configured, or all methods reject the authentication request, then access is denied.

For the AAA server selection, round-robin is used if multiple AAA servers for one particular authentication method are configured. Although, if the first alive server in the list cannot find a user-name, the router does not re-query the next server in the AAA server list for that authentication method and denies the access request. It may get authenticated on the next login attempt if the next selected AAA server has the appropriate user-name. It is recommended that the same user databases are maintained for AAA servers in order to avoid inconsistent behavior.

The user login is successful when the AAA server accepts the authentication request and responds to the router with an access accept message.

Implementing authentication without authorization for the routers does not require the configuration of VSAs (Vendor Specific Attributes) on the RADIUS server. However, users, user access permissions, and command authorization profiles must be configured on each router.

Any combination of these authentication methods can be configured to control network access from a router:

- Local Authentication
- RADIUS Authentication
- TACACS+ Authentication
- LDAP Authentication

2.1.1.1 Local Authentication

Local authentication uses PKI or user names and passwords as authentication credentials to authenticate login attempts. The authentication credentials are local to each router, not to user profiles.

By default, local authentication is enabled. When one or more of the other security methods are enabled, local authentication is used in case it is configured as first method in the authentication order, or if other authentication methods are configured before local in the authentication order and fail.

Locally, user names, public keys, and password management information can be configured. This is referred to as local authentication.

2.1.1.2 RADIUS Authentication

Remote Authentication Dial-In User Service (RADIUS) is a client/server security protocol and software that enables remote access servers to communicate with a central server to authenticate dial-in users and authorize access to the requested system or service.

RADIUS allows administrators to maintain user profiles in a shared central database and provides better security, allowing a company to set up a policy that can be applied at a single administered network point.

2.1.1.2.1 RADIUS Server Selection

The RADIUS server selection algorithm is used by different applications:

- RADIUS operator management
- RADIUS authentication for Enhanced Subscriber Management
- RADIUS accounting for Enhanced Subscriber Management
- RADIUS PE-discovery

In all these applications, up to 5 RADIUS servers pools (per RADIUS policy, if used) can be configured.

The RADIUS server selection algorithm can work in 2 modes, either Direct mode or Round-robin mode.

Direct Mode

The first server is used as the primary server. If this server is unreachable, the next server, based on the server index, of the server pool is used. This continues until either all servers in the pool have been tried or an answer is received.

If a server is unreachable, it will not be used again by the RADIUS application for the next 30 seconds to allow the server to recover from its unreachable state. After 30 seconds the unreachable server is available again for the RADIUS application. If in these 30 seconds the RADIUS application receives a valid response for a previously sent RADIUS packet on that unreachable server, the server will be available for the RADIUS application again, immediately after reception of that response.

Round-Robin Mode

The RADIUS application sends the next RADIUS packet to the next server in the server pool. The same server non-reachability behavior is valid as in the Direct mode.

Server Reachability Detection

A server is reachable, when the operational state UP, when a valid response is received within a timeout period which is configurable by the retry parameter on the RADIUS policy level.

A server is treated as not-reachable, when the operational state down, when the following occurs:

- A timeout — If a number of consecutive timeouts are encountered for a specific server. This number is configurable by the retry parameter on RADIUS policy level.
- A send failed — If a packet cannot be sent to the RADIUS server because the forwarding path towards the RADIUS server is broken (for example, the route is not available, the is interface shutdown, etc.), then, no retry mechanism is invoked and immediately, the next server in line is used.

A server that is down can only be used again by the RADIUS algorithm after 30 seconds, unless, during these 30 seconds a valid RADIUS reply is received for that server. Then, the server is immediately marked UP again.

The operational state of a server can also be “unknown” if the RADIUS application is not aware of the state of the RADIUS server (for example, if the server was previously down but no requests had been sent to the server, thus, it is not certain yet whether the server is actually reachable).

Application Specific Behavior

Operator Management

The server access mode is fixed to Round-Robin (Direct cannot be configured for operator management). A health-check function is available for operator management, which can optionally be disabled. The health-check polls the server once every 10 seconds with an improbable user name. If the server does not respond to this health-check, it will be marked down.

If the first server in the list cannot find a user, the next server in the RADIUS server list is not queried and access is denied. If multiple RADIUS servers are used, it is assumed they all have the same user database.

RADIUS Authentication

If the first server in the list cannot find a user, the next server in the RADIUS server list is not queried and access is denied. If multiple RADIUS servers are used, it is assumed they all have the same user database.

RADIUS Challenge/Response Interactive Authentication

Challenge-response interactive authentication is used for key authentication where the RADIUS server is asking for the valid response to a displayed challenge. The challenge packet includes a challenge to be displayed to the user, such as a unique generated numeric value unlikely ever to be repeated. Typically this is obtained from an external server that knows what type of authenticator is in the possession of the authorized user and can therefore choose a random or non-repeating pseudorandom number of appropriate length.

The user then enters the challenge into his device (or software) and it calculates a response, which the user enters into the client which forwards it to the RADIUS server within an access request. If the response matches the expected response, the RADIUS server allows the user access, otherwise it rejects the response.

RADIUS challenge/response mode is enabled using the CLI interactive-authentication command in the config>system>security>radius context. RADIUS interactive authentication is disabled by default. The option needs to be enabled using CLI.

Enabling interactive authentication under CLI does not mean that the system uses RADIUS challenge/response mode by default. The configured password authentication-order parameter is used. If the authentication-order parameter is local RADIUS, the system will first attempt to login the user using local authentication. If this fails, the system will revert to RADIUS and challenge/response mode. The authentication-order will precede the RADIUS interactive-authentication mode.

Even if the authentication-order is RADIUS local, the standard password prompt is always displayed. The user enters a username and password at this prompt. If RADIUS interactive-authentication is enabled the password does not have to be the correct password since authentication is accomplished using the RADIUS challenge/response method. The user can enter any password. The username and password are sent to the RADIUS server, which responds with a challenge request that is transmitted back to the node by the RADIUS server. Once the user enters the challenge response, the response is authenticated by the RADIUS server to allow node access to the user.

For example, if the system is configured with system security authentication-order set to local RADIUS, at the login prompt the user can enter the username “admin” and the corresponding password. If the password for local authentication does not match, the system falls into RADIUS authentication mode. The system checks the interactive-authentication configuration and if it is enabled it enters into challenge/response mode. It sends the username and password to the RADIUS server, and the server sends the challenge request back to the node and to the user where it appears as a challenge prompt on screen. A challenge received from the RADIUS server typically contains a string and a hardware token that can be used to generate a password on the users’ local personal token generator. For example, the RADIUS server might send the challenge prompt “Enter response for challenge 12345:” to the SR OS. The string “12345” can be entered in the local token generator which generates the appropriate challenge response for the entered string. This challenge response can then be entered on the SR OS prompt for authorization.

Once the user enters the correct challenge response it is authenticated using the RADIUS server. The server authenticates the user and the user gains access to the node.

If session timeout and Idle timeout values are configured on the RADIUS server, these are used to govern the length of time before the SR OS cancels the challenge prompt. If the user is idle longer than the received idle-timeout (seconds) from the RADIUS server, and/or if the user does not press ENTER before the received session-timeout (seconds).

Note: For SSH only the session-timeout value is used. The SSH stack cannot track character input into the login prompt until the enter key is pressed.

If the idle/session attribute is not available or if the value is set to a very large number, the SR OS uses the smallest value set in “configure system login-control idle-timeout” and the idle/session timeout attribute value to terminate the prompt. If the “login-control idle-timeout” is disabled, the maximum idle-timeout (24-hours) is used for the calculation.

The SR OS displays the log-in attempts/failure per user in the “show system security user user-name” screen. If the RADIUS rejects a challenge response, it counts as a failed login attempt and a new prompt is displayed. The number of failed attempts is limited by the value set for “configure system security password attempt.” An incorrect challenge response results in a failure count against the password attempts.

RADIUS Accounting

RADIUS accounting can be used for two purposes:

- CLI command accounting
- Enhanced Subscriber Management subscriber host accounting

The RADIUS accounting application will try to send all the accounting records of a subscriber host to the same RADIUS server. If that server is down, then the records are sent to the next server, and from that moment on, the RADIUS application uses that server as the destination for accounting records for that subscriber host. Enhanced Subscriber Management applies to the 7750 SR platform.

RADIUS PE-Discovery

If the first server in the list cannot find a user, the next server in the RADIUS server list is not queried and access is denied. If multiple RADIUS servers are used, it is assumed they all have the same user database.

The RADIUS PE-discovery application makes use of a 10 second time period instead of the generic 30 seconds and uses a fixed consecutive timeout value of 2 (see Server Reachability Detection).

As long as the Session-Timeout (attribute in the RADIUS user file) is specified, it is used for the polling interval. Otherwise, the configured polling interval will be used (60 seconds by default).

2.1.1.3 TACACS+ Authentication

Terminal Access Controller Access Control System (TACACS) is an authentication protocol that allows a remote access server to forward a user's login password to an authentication server to determine whether access can be allowed to a given system. TACACS is an encryption protocol and therefore less secure than the later Terminal Access Controller Access Control System Plus (TACACS+) and RADIUS protocols.

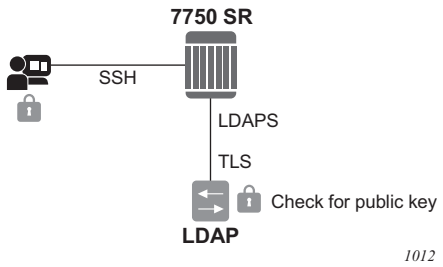
TACACS+ and RADIUS have largely replaced earlier protocols in the newer or recently updated networks. TACACS+ uses Transmission Control Protocol (TCP) and RADIUS uses the User Datagram Protocol (UDP). TACACS+ is popular as TCP is thought to be a more reliable protocol. RADIUS combines authentication and authorization. TACACS+ separates these operations.

2.1.1.4 LDAP Authentication

Lightweight Directory Access Protocol (LDAP) can provide authentication, authorization, and accounting (AAA) functionality using in-band-management, and can allow users to access the full virtualized data center and networking devices. SR OS currently supports LDAP provision of a centralized authentication method with public key management. The authentication method is based on SSH public keys or keyboard authentication (username, password).

Administrators can access networking devices with one private key; public keys are usually saved locally on the SSH server. Proper key management is not feasible with locally-saved public keys on network devices or on virtual machines, as this would result in hundreds of public keys distributed on all devices. LDAPv3 provides a centralized key management system that allows for secure creation and distribution of public keys in the network. Public keys can be remotely saved on the LDAP server, which makes key management much easier, as shown in Figure 2.

Figure 2 **Key Management**



The administrator starts an SSH session through an SSH client using their private key. The SSH client for the authentication method sends a signature created with the user's private key to the router. The router authenticates the signature using the user's public key and gives access to the user. To access the public key, the router looks up the public key stored on the LDAP server and the public key stored locally.

The order in which the public keys are looked up is defined by the authentication order. Communication between the router and the LDAP server should be secured with LDAP over SSL/TLS (LDAPS). After opening successfully a secured connection, LDAP returns a set of public keys that can be used by the router to verify the signature.

LDAP is integrated into the SR OS as an AAA protocol alongside existing AAA protocols, such as RADIUS and TACACS+. The AAA framework provides tools and mechanisms (such as method lists, server groups, and generic attribute lists) that enable an abstract and uniform interface to AAA clients, irrespective of the actual protocol used for communication with the AAA server.

The authentication functions are:

- Public key authentication — The client tries to SSH to the SR OS using public keys.

  Public keys can be stored locally or on the LDAP server and retrieved as needed to authenticate the user.
- Password authentication — Keyboard interactive

  The LDAP server can be used for user authentication using keyboard interactive, as with simple user name and password authentication.

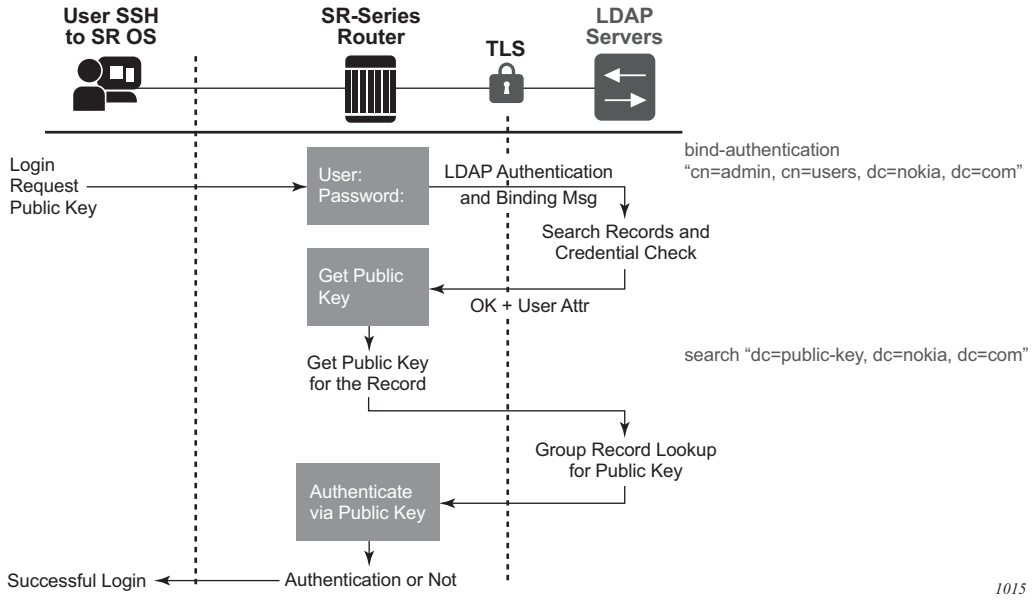
2.1.1.4.1 LDAP Authentication Process

A client starts an LDAP session by connecting to an LDAP server, called a Directory System Agent (DSA), which–by default–are on TCP port 389 and UDP port 636 for LDAP. The SR OS then sends an operation request to the server, and the server sends responses in return, as shown in Figure 3. With some exceptions, the client does not need to wait for a response before sending the next request, and the server may send the responses in any order. All information is transmitted using Basic Encoding Rules (BER).

In the SR OS, the client can request the following operations:

- StartTLS — Uses the LDAPv3 Transport Layer Security (TLS) extension for a secure connection.
- Bind — Authenticates and specify the LDAP protocol version.
- Search — Searches for and retrieve directory entries.
- Unbind — Closes the connection (not the inverse of Bind).

Figure 3 **LDAP Server and SR OS Interaction for Retrieving the Public Key**



The connection between the router as the LDAP client and the LDAP server should be encrypted using TLS, as all credentials between the router and LDAP are transmitted in clear text.

2.1.1.4.2 Authentication Order

SR OS supports local and LDAP public key storage, the order of which is configured using the **config>system>security>password>authentication-order** command.

Note: The SR OS sends available authentication methods to the client and supports public key and password authentication. If the client is configured using **public-key-authentication** then it will use the public key authentication method.

If the client chooses the public key and LDAP is first in authentication order, then the SR OS will try to authenticate using public key retrieval from the LDAP server. If the public key retrieval from LDAP server fails and **exit-on-reject** was not configured, the SR OS will try the next method (**local**) in authentication order for the public key. If the next method also fails, a user authentication fail message will be sent to the client.

If the public key retrieval from the LDAP server fails and **exit-on-reject** is configured, the SR OS will not try the next method in the authentication order. A user authentication fail message will be sent to the client. At this point, the client can be configured to only use public key authentication, or use both public key authentication followed by password authentication. If the client is configured to use password authentication, it will go through the authentication order again, (for example, it will try all the configured methods in the configured **authentication-order**) as long as **exit-on-reject** is not configured.

Authentication Order Public Key Detail

There are two keys for public key authentication: a private key stored on the client and a public key stored on the server (local) or AAA server (LDAP). The client uses the private key to create a signature, which only the public key can authenticate. If the signature is authenticated using the public key, then the user is also authenticated and is granted access. SR OS can locally store, using CLI, as many as 32 RSA keys and 32 ECDHA keys for a single user. In total, the SR OS can load a maximum of 128 public keys in a single authentication attempt.

Note: The client creates a signature using a single private key, but this signature can be authenticated on the SR OS with maximum of 128 public keys in a single try. If all these public keys fail to authenticate, then a failure message will be sent to the client and the number of failed attempts will be incremented.

If the client has another private key, it can create a new signature with this new private key and attempt the authentication one more time, or switch to password authentication.

The following steps outline the procedure where the client attempts to authenticate using a public key and the authentication order is configured as **ldap**, then **local**.

Note: With each increment of failed attempts, the SR OS also checks the limit for lock-out. If the limit is reached, the user is locked out.

1. The SSH client opens a session and tries to authenticate the user with private-key-1 (creating signature-1 from private-key-1).
2. The SR OS checks the authentication order.
3. The SR OS loads public keys for the user, as follows.
    a. If **exit-on-reject** is not configured, the SR OS loads all public keys from the LDAP server and all public keys from the locally-saved location.

b. If **exit-on-reject** is configured, the SR OS only loads all public keys from the LDAP server and not from the locally-saved location.

4. The SR OS compares received client signature-1 with signature calculated from loaded public keys and attempts to find a match.
   a. If a match is found, the user is authenticated. The procedure ends.
   b. If no match is found, authentication fails and the SSH client is informed. The LDAP server waits for the SSH client's reaction.
5. The SSH client reacts in one of several ways.
   a. The connection is closed.
   b. The password authentication method is continued. In this case, on the SR OS, the number of failed authentication attempts is not incremented.
   c. The next public key is continued, as follows.
      i. If it is not 21st received public key, return to step 3.
      ii. If it is the 21st received public key, the number of failed authentication attempts is incremented and the connection is closed.

2.1.1.4.3 LDAP Authentication Using a Password

In addition to public key authentication, the SR OS supports password (keyboard) authentication using the LDAP server.

Note: TLS provides the encryption for password authentication.

In the following example, the client attempts to authenticate using a password and only **ldap** is configured in the authentication order.

1. The client uses telnet or SSH to reach the SR OS.
2. The SR OS retrieves the user name and password (in plain text).
3. The SR OS performs a bind operation to the LDAP server using the **config>system>security>ldap>server>blind-operation** command to set the *root-dn* and *password* variables.
4. The SR OS performs a search operation for the username on LDAP server.
   a. If the user name is found, LDAP sends user_distinguished_name to the router.
   b. If the user name is not found, the authentication fails. The attempt and failed attempt counters will be incremented.

5. The SR OS performs a bind operation to LDAP with user_distinguished_name and the password from step 2.
6. The LDAP server checks the password.
   a. If the password is correct, the bind operation succeeds. The failed attempt and successful attempt counters are incremented.
   b. If the password is incorrect, bind is unsuccessful and authentication fails. The attempt and failed attempt counters are incremented.
7. The SR OS sends a message to unbind from the LDAP server.

2.1.1.4.4 Timeout and Retry Configuration for the LDAP Server

The **retry** value is the maximum number of connection attempts that the SR OS can make to reach the current LDAP server before attempting the next server. For example, if the value is set to the default of 3, the SR OS will try to establish the connection to current server three times before attempting to establish a connection to the next server.

The **timeout** value is the number of seconds that the SR OS will wait for a response from the server with which it is attempting to establish a connection. If the server does not reply within the specified timeout value, the SR OS increments the **retry** counter by one. The SR OS attempts to establish the connection to the current server up to the configured **retry** value before moving to the next configured server.

2.1.1.4.5 TLS Behavior and LDAP

RFC 4511 section 4.14.1 states, “A client requests TLS establishment by transmitting a StartTLS request message to the server” and “The client MUST NOT send any LDAP PDUs at this LDAP message layer following this request until it receives a StartTLS Extended response”. As such, if an LDAP has a TLS profile configured and the TLS is in an operationally down state, no LDAP packets will be transmitted if TLS negotiation has not been completed, including when the TLS profile is shut down.

2.1.1.4.6 LDAP Health Check

The health check for LDAP is configured under **config**>**system**>**security**>**password**.

The **health-check** function, which can be disabled, is available for operator management. The health check polls the server at a specified interval (the default is 30 seconds). The SR OS health check attempts to establish a TCP connection to the LDAP server. The TCP connection is closed by an LDAP unbind message.

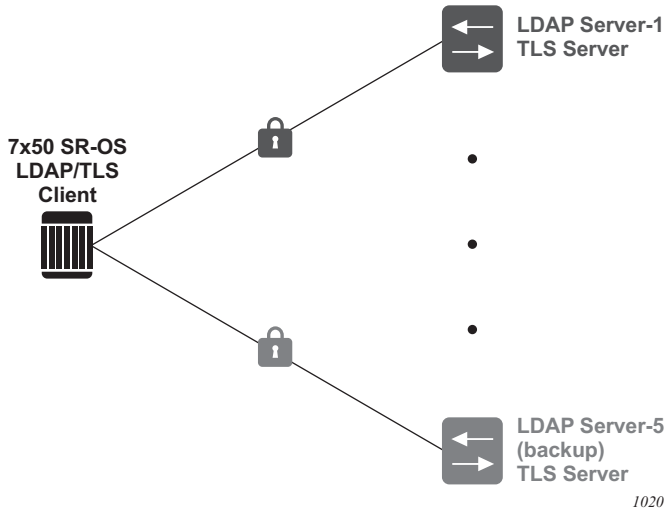
2.1.1.4.7 LDAP Redundancy and TLS

LDAP supports up to five redundant (backup) servers. Depending on the configuration of **timeout** and **retry** values, if an LDAP server is found to be out of service or operationally down, the SR OS will switch to the redundant servers. The SR OS will try the next LDAP server in the server list by choosing the next largest configured server index.

LDAP servers can use the same TLS profile or can have their own TLS profile. Each TLS profile can have a different configuration of **trust-anchor**, **cipher-list** and **cert-profile**. For security reasons, the LDAP server could be in different geographical areas and, as such, each will be assigned its own server certificate and trust anchor. The TLS profile design allows users to mix and match all components.

Redundant LDAP servers are shown in Figure 4.

Figure 4 **LDAP and TLS Redundancy**



2.1.1.5 Password Hashing

SR OS supports two algorithms for user password hashing: bcrypt, which is the default algorithm, and PBKDF2. The PBKDF2 algorithm can use SHA2 (SHA-256) or SHA3 (SHA-512) for hashing.

The algorithm can be configured using the **hashing** command from the **configure**>**system**>**security**>**password** context. The configured algorithm hashes all user passwords.

When password hashing is configured, the following sequence of steps occurs at login:

1. The node checks the stored password and notes its hash algorithm.
2. The password entered by the user is hashed with the noted algorithm, and the node compares the hash with the stored user password hash.
3. If the entered and the stored passwords are the same, and if the hash algorithm of the stored user password is different than the hash algorithm of the system password, the user is prompted to enter a new password 2 times to ensure password match. The node stores this new password in the RAM (not in the system configuration file).

   To store the new password in the configuration file, an admin user must perform an **admin save** command. If the **admin save** command is not executed, then on the next reboot the hash algorithm of the stored user password might be different than the system hash and the user must go through this process again from step 2.

After an upgrade to a software load that supports PBKDF2, the default password continues to be stored using the bcrypt algorithm. The following example describes the procedure to change the algorithm. In the example, the algorithm is changed to PBKDF2 and “User_name” can be any user.

1. User_name logs in and runs the **hashing** command to change the algorithm.
2. To save the algorithm change, an admin user performs an **admin save** command.
3. To store User_name’s password using PBKDF2, the admin user changes User_name’s password.
4. From this point onward, any new user passwords or changes to existing user passwords are stored using PBKDF2.

2.1.2 Authorization

The SR OS supports local, RADIUS, and TACACS+ authorization to control the actions of specific users. Any combination of these authorization methods can be configured to control actions of specific users:

- Local Authorization
- RADIUS Authorization
- TACACS+ Authorization

Local authorization and RADIUS authorization operate by applying a command authorization profile that is associated in configuration with the user. The profiles are configured locally on the router or downloaded using VSAs from a RADIUS server. See Vendor-Specific Attributes (VSAs).

Authorization applies to CLI access as well as NETCONF or gRPC access. See Authorization Profiles for Different Interfaces for more details.

2.1.2.1 Local Authorization

Local authorization uses user profiles and user access information after a user is authenticated. The profiles and user access information specifies the actions the user can and cannot perform.

By default, local authorization is enabled. Local authorization is disabled only when a different remote authorization method is configured, such as TACACS+ or RADIUS authorization and **local** is removed from the authorization order.

2.1.2.2 RADIUS Authorization

RADIUS authorization grants or denies access permissions for a router. Permissions include the use of FTP, Telnet, SSH (SCP), and console access. When granting Telnet, SSH (SCP) and console access to the router, authorization can be used to limit what CLI commands the user is allowed to issue and which file systems the user is allowed or denied access.

Once a user has been authenticated using RADIUS (or another method), the router can be configured to perform authorization. The RADIUS server can be used to:

- Download the user profile to the router.
- Send the profile name that the node should apply to the router.

Profiles consist of a suite of commands that the user is allowed or not allowed to execute. When a user issues a command, the authorization server looks at the command and the user information and compares it with the commands in the profile. If the user is authorized to issue the command, the command is executed. If the user is not authorized to issue the command, then the command is not executed.

Profiles must be created on each router and should be identical for consistent results. If the profile is not present, then access is denied.

Table 3 displays the following scenarios:

- Remote (RADIUS) authorization cannot be performed if authentication is done locally (on the router).
- The reverse scenario is supported if RADIUS authentication is successful and no authorization is configured for the user on the RADIUS server, then local (router) authorization is attempted, if configured in the authorization order.

When authorization is configured and profiles are downloaded to the router from the RADIUS server, the profiles are considered temporary configurations and are not saved when the user session terminates.

Table 3 **Supported Authorization Configurations**

	Router	RADIUS Supplied Profile
Router configured user	✓	
RADIUS server configured user	✓	✓
TACACS+ server configured user	✓	

When using authorization, maintaining a user database on the router is not required. User names can be configured on the RADIUS server. User names are temporary and are not saved in the configuration when the user session terminates. Temporary user login names and their associated passwords are not saved as part of the configuration.

2.1.2.3 TACACS+ Authorization

TACACS+ command authorization operates in one of three ways:

- All users who authenticate via TACACS+ can use a single common default command authorization profile that is configured on the SR OS

- Each command attempted by a user is sent to the TACACS+ server for authorization
- The operator can configure local profiles and map tacplus priv-lvl based authorization to those profiles (the **use-priv-lvl** option)

To use a single common default command authorization profile to control command authorization for TACACS+ users, the operator must enable the **tacplus use-default-template** option and configure the parameters in the **user-template tacplus_default** to point to a valid local profile. The **tacplus authorization** command must also be disabled.

If the default template is not being used for TACACS+ authorization and the **tacplus authorization** command is enabled without the **use-priv-lvl**, then each CLI command issued by an operator is sent to the TACACS+ server for authorization. The authorization request sent by the SR OS contains the first word of the CLI command as the value for the TACACS+ cmd and all following words become a cmd-arg. Quoted values are expanded so that the quotation marks are stripped off and the enclosed value are seen as one cmd or cmd-arg.

When the **use-priv-lvl** option is used, the router will map the priv-lvl returned by the TACACS+ server to a local profile as configured under the **priv-lvl-map**. Command authorization will then use the local profile. If the TACACS+ server does not return a priv-lvl, and the **tacplus use-default-template** setting is enabled, then the router will use the local profile in the **user-template tacplus_default** for command authorization.

2.1.2.3.1 Examples

Here is a set of examples, where the following commands are typed in the CLI:

```
- "show"
- "show router"
- "show port 1/1/1"
- "configure port 1/1/1 description "my port"
```

This results in the following AVPairs:

```
cmd=show

cmd=show
cmd-arg=router

cmd=show
cmd-arg=port
cmd-arg=1/1/1

cmd=configure
cmd-arg=port
```

```
cmd-arg=1/1/1
cmd-arg=description
cmd-arg=my port
```

For TACACS+ authorization, the SR OS sends the entire CLI context in the **cmd** and **cmd-arg** values. Here is a set of examples where the CLI context is different:

```
- *A:dut-c# configure service
- *A:dut-c>config>service# vprn 555 customer 1 create
- *A:dut-c>config>service>vprn$ shutdown
```

This results in the following AVPairs:

```
cmd =configure
cmd-arg=service

cmd=configure
cmd-arg=service
cmd-arg=vprn
cmd-arg="555"
cmd-arg=customer
cmd-arg=1
cmd-arg=create

cmd=configure
cmd-arg=service
cmd-arg=vprn
cmd-arg="555"
cmd-arg=customer
cmd-arg=1
cmd-arg=create
cmd-arg=shutdown
```

2.1.2.4 Authorization Profiles for Different Interfaces

Authorization profiles can be configured in any format including classic the CLI and the MD-CLI. Depending on the configuration, a match might be hit.

Each entry in a profile can be formatted for the classic CLI or the MD-CLI. Nokia recommends creating separate profiles for each interface type. For example, a profile for the classic CLI and a different profile for the MD-CLI.

Authorization checks are not performed by default for telemetry data. All configuration and state elements are available to authenticated telemetry subscriptions, with the exception of LI (Lawful Intercept) configuration and state elements, which are authorized separately based on the LI authorization configuration. To control telemetry data authorization, use the classic CLI **configure>system>security>managment-interface>output-authorization>telemetry-data** command or the MD-CLI **configure system security aaa management-interface output-authorization telemetry-data** command.

Table 4 shows authorization and match hit based on the entry format configuration. This is true whether authorization is done using local user profiles or using an AAA server like TACACS+ or RADIUS.

Table 4 **Authorization and Match Hit Based on Entry Format**

Profile Entry Format	Classic CLI	MD-CLI	NETCONF	gNMI Set & Get (gRPC)
Classic CLI	Yes	Maybe	Maybe	Maybe
MD-CLI	Maybe	Yes	Yes	Yes

2.1.2.5 Authorization Support

Table 5 shows authorization support using a local profile or an AAA server.

Table 5 **Authorization Support**

	Classic CLI	MD-CLI	NETCONF	gNMI Set & Get (gRPC)
LDAP	—	—	—	—
TACACS+	Yes	Yes	Yes	Yes
RADIUS	Yes	Yes	Yes	Yes
Local	Yes	Yes	Yes	Yes

2.1.2.5.1 System-Provisioned AAA Command Authorization Profiles

SR OS provides two built-in (system-provisioned) AAA command authorization profiles. These profiles can be removed or modified.

- default

- administrative

The built-in profiles are applicable to users using classic CLI or MD-CLI, and contain rules that apply to classic CLI and rules that apply to MD-CLI interfaces in the same profile.

By default, in SR OS, the administrative profile is associated with the built-in user called 'admin'.

In classic CLI, the default profile is automatically assigned to any newly-created user, but the operator can remove the profile from any user and replace it with another profile.

In MD-CLI, a newly-created user is not associated with any profile. The operator can manually associate a user with the default profile if required.

2.1.2.5.2 Authorization Support for Configuration Groups

Authorization for MD-CLI configuration groups is done explicitly by creating an entry for that group configuration in the user’s profile.

For example, to deny access to router interfaces in both the main configuration branch and in the group configuration branch, create an entry for each one:

```
entry 10
    match "configure router interface"
    action deny
exit
entry 20
    match "configure groups group router interface"
    action deny
exit
```

Entry 10 prevents the user from viewing, creating, and editing router interfaces in the main configuration branch and from inheriting router interface configurations via configuration groups.

Entry 20 prevents the user from viewing, creating, and editing router interfaces in the groups configuration branch.

2.1.3 Accounting

When enabled, RADIUS accounting sends command line accounting from the router to the RADIUS server. The router sends spars using UDP packets at port 1813 (decimal).

The router issues an accounting request packet for each event requiring the activity to be recorded by the RADIUS server. The RADIUS server acknowledges each accounting request by sending an accounting response after it has processed the accounting request. If no response is received in the time defined in the timeout parameter, the accounting request must be retransmitted until the configured retry count is exhausted. A trap is issued to alert the NMS (or trap receiver) that the server is unresponsive. The router issues the accounting request to the next configured RADIUS server (up to 5).

User passwords and authentication keys of any type are never transmitted as part of the accounting request.

2.1.3.1 RADIUS Accounting

Accounting tracks user activity to a specified host. When RADIUS accounting is enabled, the server is responsible for receiving accounting requests and returning a response to the client indicating that it has successfully received the request. Each command issued on the router generates a record sent to the RADIUS server. The record identifies the user who issued the command and the timestamp.

Accounting can be configured independently from RADIUS authorization and RADIUS authentication.

2.1.3.2 TACACS+ Accounting

The OS allows the administrator to configure the type of accounting record packet that is to be sent to the TACACS+ server when specified events occur on the device. The **accounting record-type** parameter indicates whether TACACS+ accounting start and stop packets be sent or just stop packets be sent. Start/stop messages are only sent for individual commands, not for the session.

When a user logs in to request access to the network using Telnet or SSH, or a user enters a command for which accounting parameters are configured, or a system event occurs, such as a reboot or a configuration file reload, the router checks the configuration to see if TACACS+ accounting is required for the particular event.

If TACACS+ accounting is required, then, depending on the accounting record type specified, sends a start packet to the TACACS+ accounting server which contains information about the event.

The TACACS+ accounting server acknowledges the start packet and records information about the event. When the event ends, the device sends a stop packet. The stop packet is acknowledged by the TACACS+ accounting server.

2.1.3.3 Command Accounting Log Events

In addition to RADIUS and TACACS+ accounting, SR OS supports a set of log events dedicated to command accounting.

Refer to “Service Router Log Events” in the *Log Events Guide* for the following log events related to command accounting:

- cli_user_io
- snmp_user_set
- cli_config_io
- cli_unauth_user_io
- cli_unauth_config_io
- md_cli_io
- md_cli_unauth_io
- netconf_auth
- netconf_unauth
- grpc_auth
- grpc_unauth

2.2 Security Controls

Configure routers to use RADIUS, TACACS+, LDAP, and local authentication to validate users requesting access to the network. The order in which authentication is processed among RADIUS, TACACS+, LDAP, and local can be specifically configured. In other words, the authentication order can be configured to process authorization through TACACS+ first, then RADIUS for authentication and accounting. Local access can be specified next in the authentication order in the event that the RADIUS and TACACS+ servers are not operational. The security methods capabilities are listed in Table 6.

Table 6 **Security Methods Capabilities**

Method	Authentication	Authorization	Accounting*
Local	✓	✓	Not supported
TACACS+	✓	✓	✓
RADIUS	✓	✓	✓
LDAP	✓	Not supported	Not supported
* Local commands always perform account logging using the **config log** command.			

2.2.1 When a Server Does Not Respond

A trap is issued if a RADIUS + server is unresponsive. An alarm is raised if RADIUS is enabled with at least one RADIUS server and no response is received to either accounting or user access requests from any server.

Periodic checks to determine if the primary server is responsive again are not performed. If a server is down, it will not be contacted for 5 minutes. If a login is attempted after 5 minutes, then the server is contacted again. When a server does not respond with the health check feature enabled, the server's status is checked every 30 seconds. Health check is enabled by default. When a service response is restored from at least one server, the alarm condition is cleared. Alarms are raised and cleared on Nokia's Fault Manager or other third party fault management servers.

The servers are accessed in order from lowest to highest specified index (from 1 to 5) for authentication requests until a response from a server is received. A higher indexed server is only queried if no response is received, implying a lower indexed server is not available. If a response from the server is received, no other server is queried.

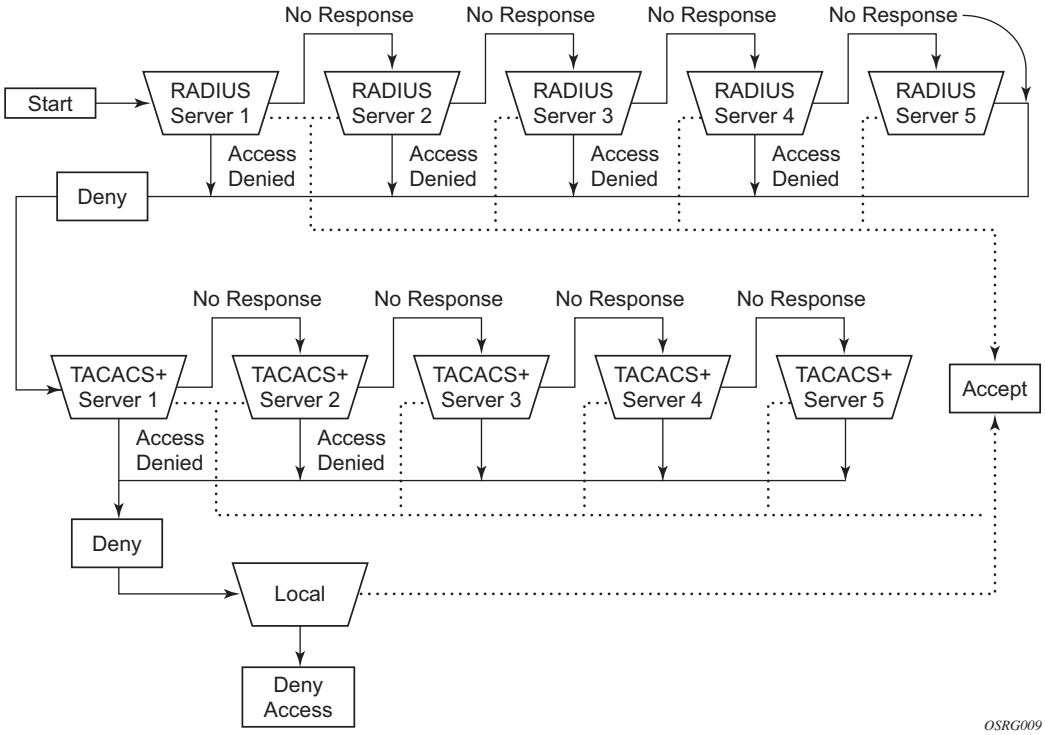
2.2.2 Access Request Flow

In Figure 5, the authentication process is defined in the config>system>security> password context. The authentication order is determined by specifying the sequence in which authentication is attempted among RADIUS, TACACS+, LDAP, and local. This example uses the authentication order of RADIUS, then TACACS+, and finally, local. An access request is sent to RADIUS server 1. One of two scenarios can occur. If there is no response from the server, the request is passed

to the next RADIUS server with the next lowest index (RADIUS server 2) and so on, until the last RADIUS server is attempted (RADIUS server 5). If server 5 does not respond, the request is passed to the TACACS+ server 1. If there is no response from that server, the request is passed to the next TACACS+ server with the next lowest index (TACACS+ server 2) and so on.

If a request is sent to an active RADIUS server and the user name and password is not recognized, access is denied and passed on to the next authentication option, in this case, the TACACS+ server. The process continues until the request is either accepted, denied, or each server is queried. Finally, if the request is denied by the active TACACS+ server, the local parameters are checked for user name and password verification. This is the last chance for the access request to be accepted.

Figure 5 **Security Flow**



2.3 Control and Management Traffic Protection

SR OS routers support an extensive set of configurable mechanisms to protect the CPU from being flooded with control or management traffic.

These protection mechanisms are a set of configurable hardware-based filters, classification, queuing, and rate-limiting functions that drop unwanted traffic before it reaches the control processor.

- In-band traffic extracted from the line cards to the CPM:
  - Line card features:
    - ACLs filters: IPv4, IPv6, and MAC
    - Anti-spoofing, uRPF
    - Distributed CPU protection
  - CPM features:
    - CPM Filters: IPv4, IPv6, and MAC
    - Centralized CPU Protection
    - Per-peer queues, protocol queues, CPM queues
- Out-band and in-band traffic: Management access filters

2.3.1 CPM Filters

CPM filters are hardware-based filters used to restrict traffic from the line cards directed to the control processor. This filtering is performed by the Fast Path (FP) network processor and it uses no resources on the main CPU.

CPM filters filter traffic is extracted from the data plane and sent to the CPM for processing. Packets from all network and access ports are filtered. Packets originating from a management Ethernet port can be filtered using management access filters. See Management Access Filter for more information.

2.3.1.1 CPM Filter Packet Match

Three different CPM filter policies can be configured: **ip-filter**, **ipv6-filter**, and **mac-filter**.

CPM filter packet match rules:

- Each CPM filter policy is an ordered list of entries. Entries must be sequenced correctly from the most explicit to the least explicit.
- If multiple match criteria are specified in a single CPM filter policy entry, all criteria must be met for the packet to be considered a match against that policy entry (logical AND).

- Any match criteria not explicitly defined is ignored during a match.
- A CPM filter policy entry defined without any match criteria is inactive.
- A CPM filter policy entry with match criteria defined, but no action configured, inherits the default action defined at the **cpm-filter** level.
- The **cpm-filter default-action** applies to IPv4, IPv6, or MAC CPM filters that are in a **no shutdown** state.
- When **mac-filter** and **ip-filter**/**ipv6-filter** are applied to a specific packet, the **mac-filter** is applied first.

2.3.1.2 CPM IPv4 and IPv6 Filter Entry Match Criteria

The supported IPv4 and IPv6 match criteria are shown in the following tables.

Table 7 lists the basic Layer 3 match criteria.

Table 7 **Basic Layer 3 Match Criteria**

Criteria	Description
dscp	Matches the specified DSCP value against the DSCP/Traffic Class field in the IPv4 or IPv6 packet header.
src-ip/**dst-ip**	Matches the specified source/destination IPv4/IPv6 address prefix/mask against the source/destination IPv4/IPv6 address field in the IP packet header. Optionally, operators can match a list of IP addresses defined in **filter match-list ip-prefix-list** or **match-list ipv6-prefix-list**. The **prefix-list** can be defined statically or using the **apply-path** command to automatically populate using configured BGP peers defined in the base router or VPRN services. Refer to the “Match List for Filter Policies” section in the *Router Configuration Guide* for more details on filter **match-list** configuration and capabilities.
fragment	For IPv4, match against the MF bit or Fragment Offset field to determine if the packet is a fragment. For IPv6 match against the next-header field or Fragment Extension Header value to determine whether the packet is a fragment. Up to six extension headers are matched against to find the Fragmentation Extension Header.

Table 8 lists the IPv4 options match criteria.

Table 8 **IPv4 Options Match Criteria**

Criteria	Description
ip-option	Matches the specified option value in the first option of the IPv4 packet. Optionally, operators can configure a mask to be used in a match.
option-present	Matches the presence of IP options in the IPv4 packet. Padding and EOOL are also considered as IP options. Up to six IP options are matched against.
multiple-option	Matches the presence of multiple IP options in the IPv4 packet.

Table 9 lists the IPv6 next-header match criteria.

Table 9 **IPv6 Next-Header Match Criteria**

Criteria	Description
hop-by-hop-opt	Matches for the presence of hop-by-hop options extension header in the IPv6 packet. This match criterion is supported on ingress only. Up to six extension headers are matched against.

Table 10 lists the upper-layer protocol match criteria.

Table 10 **Upper-Layer Protocol Match Criteria**

Criteria	Description
next-header	Matches the specified upper-layer protocol (such as TCP or UDP) against the next-header field of the IPv6 packet header. “*” can be used to specify TCP or UDP upper-layer protocol match (logical OR). Next-header matching also allows matching on the presence of a subset of IPv6 extension headers. See the CLI section for information on which extension header match is supported.
protocol	Matches the specified protocol against the Protocol field in the IPv4 packet header (for example, TCP, UDP, or IGMP) of the outer IPv4. “*” can be used to specify TCP or UDP upper-layer protocol match (logical OR).
icmp-code	Matches the specified value against the Code field of the ICMP/ ICMPv6 header of the packet. This match is supported only for entries that also define protocol/next-header match for ICMP/ ICMPv6 protocol.

Table 10 **Upper-Layer Protocol Match Criteria (Continued)**

Criteria	Description
icmp-type	Matches the specified value against the Type field of the ICMP or ICMPv6 header of the packet. This match is supported only for entries that also define protocol/next-header match for “ICMP” or “ICMPv6” protocol.
src-port/dst-port/ port	Matches the specified port value (with or without mask), port list, or port range against the Source Port Number/Destination Port Number of the UDP/TCP packet header. An option to match either source or destination port or both (logical OR) using a single filter policy entry is supported by using a directionless **port** command. Source/ destination match is supported only for entries that also define protocol/next-header match for “TCP”, “UDP” or “TCP or UDP” protocols. A non-initial fragment will not match an entry with non-zero port criteria specified.
tcp-ack/tcp-syn	Matches the presence or absence of the TCP flags in the TCP header of the packet. This match criteria also requires defining the protocol/next-header match as “TCP”.

Table 11 lists the router instance match criteria.

Table 11 **Router Instance Match Criteria**

Criteria	Description
router	Matches the router instance packets that are ingressing from for this filter entry.

2.3.1.3 CPM MAC Filter Entry Match Criteria

The MAC match criteria are evaluated against the Ethernet header of the Ethernet frame.

Table 12 lists the router instance match criteria.

Table 12 **Router Instance Match Criteria**

Criteria	Description
frame-type	The filter matches a specific type of frame format. For example, configuring frame-type ethernet_II matches only Ethernet-II frames.

Table 12 **Router Instance Match Criteria (Continued)**

Criteria	Description
src-mac	Matches the specified source MAC address frames. Optionally, operators can configure a mask to be used in a match.
dst-mac	Matches the specified destination MAC address frames. Optionally, operators can configure a mask to be used in a match.
etype	Matches the specified Ethernet II frames. The Ethernet type field is a two-byte field used to identify the protocol carried by the Ethernet frame.
ssap	Matches the specified frames with a source access point on the network node designated in the source field of the packet. Optionally, operators can configure a mask to be used in a match.
dsap	Matches the specified frames with a destination access point on the network node designated in the destination field of the packet. Optionally, operators can configure a mask to be used in a match.
cfm-opcode	Matches the specified packet with the specified **cfm-opcode**.

2.3.1.4 CPM Filter Policy Action

The two main CPM filter actions allow the option to accept or drop traffic.

Optionally, traffic can be sent to a user-configured hardware queue using a CPM filter. Nokia recommends this primarily for temporary debug or attack investigation activities.

2.3.1.5 CPM Filter Policy Statistics and Logging

Refer to the "Filter Policy Logging" and "Filter Policy" sections in the *7450 ESS, 7750 SR, 7950 XRS, and VSR Router Configuration Guide Guide*.

2.3.1.6 CPM Filter: Protocols and Ports

Nokia recommends using a strict CPM filter policy allowing traffic from trusted IP subnets for protocols and ports actively used in the router and to explicitly drop other traffic.

Table 13 identifies which ports are used by which applications in the SR OS. The source port and destination port reflect the CPM filter entry configuration for traffic ingressing the router and sent to the CPM.

Table 13 **Protocols and Ports**

Src Port Number	Dst Port Number	IP Protocol	Application	Description	Accessible Out of Band	Accessible in Band
	20	TCP	FTP	FTP Server Data. Active FTP Client.	Yes	Yes
	21	TCP	FTP	FTP Server Control	Yes	Yes
20		TCP	FTP	FTP Client Data	Yes	Yes
21		TCP	FTP	FTP Client Control	Yes	Yes
	22	TCP	SSH, NETCONF	SSH Server, NETCONF Server	Yes	Yes
22		TCP	SSH	SSH Client. Responses for initiated TCP sessions.	Yes	Yes
	23	TCP	TELNET	TELNET server	Yes	Yes
49		TCP	TACACS	TACACS client. Responses for initiated sessions.	Yes	Yes
53		UDP	DNS	DNS client	—	Yes
67	67	UDP	DHCPv4	DHCPv4: Relay agent to server, server to relay agent, and relay agent to relay agent	—	Yes
68	67	UDP	DHCPv4	DHCPv4: Client to relay agent/server	—	Yes
67	68	UDP	DHCPv4	DHCPv4: relay agent/server to client	—	Yes
	123	UDP	NTP	NTP server	Yes	Yes
123		UDP	NTP	NTP client	Yes	Yes
	161	UDP	SNMP	SNMP server: SET and GET commands	Yes	Yes
	179	TCP	BGP	BGP: server terminated TCP sessions	—	Yes
179			BGP	BGP: client responses for initiated TCP session	—	Yes

Table 13 **Protocols and Ports (Continued)**

Src Port Number	Dst Port Number	IP Protocol	Application	Description	Accessible Out of Band	Accessible in Band
	319	UDP	PTP	1588 PTP event	—	Yes
	320	UDP	PTP	1588 PTP general	—	Yes
389		TCP	LDAP	LDAP client (non TLS)	Yes	Yes
	520	UDP	RIP	RIP	—	Yes
546	547	UDP	DHCPv6	DHCPv6 - Client to Server/ Relay Agent	—	Yes
547	547	UDP	DHCPv6	DHCPv6 - server to relay agent, relay agent to server, and relay agent to relay agent	—	Yes
	639	UDP	PIM	MSDP: multicast source discovery protocol	—	Yes
636		TCP	LDAPS	LDAP client over TLS	—	Yes
	646	UDP	LDP	LDP Hello adjacency	—	Yes
	646	TCP	LDP	LDP/T-LDP: terminated TCP sessions	—	Yes
646		TCP	LDP	LDP/T-LDP: responses for initiated TCP sessions	—	Yes
	701	UDP	LMP	Link management protocol	—	Yes
	830	TCP	NETCONF	NETCONF Server	Yes	Yes
	ANY	UDP	TWAMP	TWAMP test	—	Yes
	862	TCP	TWAMP	TWAMP control: terminated TCP session	—	Yes
	862, 64364-64373	UDP	TWAMP	TWAMP Light (Reflector)	—	Yes
862, 64364-64373		UDP	TWAMP	Nokia TWAMP Light Initiator. Non Nokia initiator may use the entire range.	—	Yes
	1025	UDP	MC-LAG-APS-EP-IPsec	Multi Chassis: LAG, APS (Automation Protection Switching), End Point, IPsec (MIMP), AARP	—	Yes

Table 13 **Protocols and Ports (Continued)**

Src Port Number	Dst Port Number	IP Protocol	Application	Description	Accessible Out of Band	Accessible in Band
	1491	TCP	SNMP Streaming	SNMP streaming server	Yes	Yes
	1645	UDP	Radius Proxy	RADIUS proxy authentication	—	Yes
	1646	UDP	Radius Proxy	RADIUS proxy accounting	—	Yes
	1647	UDP	Radius CoA	RADIUS Dynamic authorization (CoA/DM)	Yes	Yes
	1700	UDP	Radius CoA	RADIUS Dynamic authorization (CoA/DM)	Yes	Yes
	1701	UDP	L2TP	L2TP server	—	Yes
	1812	UDP	Radius CoA	RADIUS Dynamic authorization (CoA/DM)	Yes	Yes
1812		UDP	RADIUS	RADIUS authentication	Yes	Yes
1813		UDP	RADIUS	RADIUS accounting	Yes	Yes
	2000	UDP	WPP	Web portal authentication protocol	—	Yes
	2123	UDP	GTP	GTP control plane	—	Yes
2123		UDP	GTP	GTP control plane	—	Yes
	2152	UDP	GTP	GTP user plane	—	Yes
2152		UDP	GTP	GTP user plane	—	Yes
	3232	UDP	PIM	PIM MDT	—	Yes
	3503	UDP	OAM	LSP Ping, LSP Trace, VPRN Trace, VPRN Ping	—	Yes
3868		UDP	DIAMETER	Diameter	Yes	Yes
	3784	UDP	BFD	BFD Control 1 hop BFD and BFD over MPLS LSP	—	Yes
	3785	UDP	BFD	BFD echo	—	Yes
	3799	UDP	RADIUS	RADIUS Dynamic Authorization (CoA/DM)	Yes	Yes

Table 13 **Protocols and Ports (Continued)**

Src Port Number	Dst Port Number	IP Protocol	Application	Description	Accessible Out of Band	Accessible in Band
	4189	TCP	PCEP	Path Computation Element Protocol	Yes	Yes
	4739	UDP	NAT	NAT debug	—	Yes
	4784	UDP	BFD	BFD control multi-hop	—	Yes
	5000	UDP	Mtrace2	IP Multicast Mtrace2	—	Yes
	5351	UDP	NAT	PCP NAT port mapping protocol	—	Yes
	6068	TCP	ANCP	ANCP - terminated TCP session	—	Yes
	6653	TCP	OpenFlow	OpenFlow - terminated TCP sessions	—	Yes
	6784	UDP	BFD	uBFD	—	Yes
	8805	UDP	PFCP	Packet and forwarding control protocol - Used to install dynamic forwarding state	—	Yes
	33408-33535	UDP	OAM	OAM Traceroute	—	Yes
	45067	TCP	MCS	Multi-chassis synchronization - Terminated TCP Session (**mcs**, **mc-ring**, **mc-ipsec**)	—	Yes
45067		TCP	MCS	Multi-chassis synchronization - Responses for initiated TCP session (**mcs**, **mc-ring**, **mc-ipsec**)	—	Yes
	49151	UDP	L2TP	L2TP	—	Yes
	57400	TCP	gRPC	gRPC	—	Yes
	64353	UDP	MPLS DM	MPLS Delay Measurement using UDP return object	—	Yes
N/A	N/A	GRE	GRE	GRE	—	Yes
N/A	N/A	ICMP	ICMP	ICMP	Yes	Yes
N/A	N/A	IGMP	IGMP	IGMP	—	Yes

Table 13 **Protocols and Ports (Continued)**

Src Port Number	Dst Port Number	IP Protocol	Application	Description	Accessible Out of Band	Accessible in Band
N/A	N/A	OSPF	OSPF	OSPF	—	Yes
N/A	N/A	PIM	PIM	PIM	—	Yes
N/A	N/A	RSVP	RSVP	RSVP	—	Yes
N/A	N/A	VRRP	VRRP, SRRP	VRRP, SRRP	—	Yes
pki-server-port or 80/8080	any	TCP	PKI	CMPv2 (Certificate Management Protocol v2) client - Responses for initiated TCP session	—	Yes
pki-server-port	any	TCP	PKI	OCSP (Online Certificate Status Protocol) client - Responses for initiated TCP session	—	Yes
pki-server-port or 80/8080	any	TCP	PKI	Auto CRL (Certificate Revocation List) update (client) - Responses for initiated TCP session	Yes	Yes

2.3.2 CPM Per-Peer Queuing

Per-peer queuing provides isolation between peers by allocating hardware queues on a per-peer basis for the following TCP-based protocols: BGP, T-LDP, LDP, MSDP, Telnet, and SSH.

This mechanism guarantees fair and non-blocking access to shared CPU resources across all peers. For example, this ensures that an LDP-based DoS attack from a specific peer is mitigated and compartmentalized and not all CPU resources are dedicated to the overwhelming control traffic sent by that specific peer.

The **per-peer-queuing** command ensures that service levels would not be (or only partially be) impacted in case of an attack towards BGP, T-LDP, LDP, MSDP, Telnet, or SSH. SSH and Telnet supports per-peer queuing when the **login-control ttl-security** command is enabled.

2.3.3 CPM Queues

CPM queues provides the operator with a tool that is primarily useful for debugging or investigations during an attack. When using the CPM queues, the following recommendations should be considered.

- CPM queues can be used for temporary debug or attack investigation activities, in this case packets can be filtered and directed into the queue using the CPM filter.
- CPM queues are not recommended for normal operation where the system default handling and isolation of protocols into protocol queues is already carefully balanced. If additional protection is desired, then the use of the Centralized CPU Protection and Distributed CPU Protection (DCP) features is recommended.

2.3.4 Centralized CPU Protection

SR OS CPU protection is a centralized rate-limiting function that operates on the CPM to limit traffic destined to the CPU. The term “centralized CPU protection” is referred to as “CPU protection” in this guide and in the CLI to differentiate it from “Distributed CPU Protection”.

CPU protection provides interface isolation by rate limiting the total amount of traffic extracted to the CPM per port, interface, or SAP in hardware using a combination of limits configurable at the CPU protection system level or as CPU protection policies assigned to access or network interfaces.

The following limits are configurable at the CPU protection system level:

- **link-specific rate** — Applies to the link-specific protocols LACP (Ethernet LAG control) and LMI (ATM, Ethernet and Frame Relay). The rate is a per-link limit (each link in the system will have LACP/LMI packets limited to this rate).
- **port-overall-rate** – Applies to all control traffic, the rate is a per-port limit, each port in the system will have control traffic destined to the CPM limited to this rate.
- **protocol-protection** — Blocks network control traffic for unconfigured protocols.

The following limits are configurable independently for access or network interfaces using a dedicated CPU protection policy:

- **overall-rate** — Applies to all control traffic destined to the CPM (all sources) received on an interface where the policy is applied. This is a per-interface limit. Control traffic received above this rate will be discarded.
- **per-source-rate** — Used to limit the control traffic destined to the CPM from each individual source. This per-source rate is only applied when an object (SAP) is configured with a **cpu-protection** policy and also with the optional **mac-monitoring** or **ip-src-monitoring** keywords. A source is defined as a *SAP, Source MAC Address* tuple for MAC monitoring and as a *SAP, Source IP Address* tuple for IP source monitoring. Only certain protocols (as configured under *included-protocols* in the CPU protection policy) are limited (per source) when the **ip-src-monitoring** keyword is used.
- **out-profile-rate** – Applies to all control traffic destined to the CPM (all sources) received on an interface where the policy is applied. This is a per-interface limit. Control traffic received above this rate will be marked as discard eligible (such as, out-profile/low-priority/yellow) and is more likely to be discarded if there is contention for CPU resources.

There are two default CPU protection policies for access and network interfaces.

Policy 254:

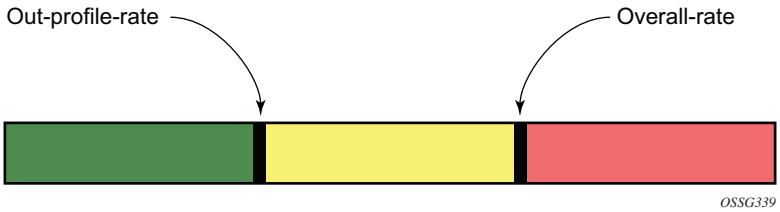
- This is the default policy that is automatically applied to access interfaces
- Traffic above 6000 pps is discarded
- overall-rate = 6000
- per-source-rate = max
- out-profile-rate = 6000

Policy 255:

- This is the default policy that is automatically applied to network interfaces
- Traffic above 3000 pps is marked as discard eligible, but is not discarded unless there is congestion in the queuing towards the CPU
- overall-rate = max
- per-source-rate = max
- out-profile-rate = 3000

A three-color marking mechanism uses a green, yellow, and red marking function. This allows greater flexibility in how traffic limits are implemented. A CLI command within the CPU protection policy called **out-profile-rate** maps to the boundary between the green (accept) and yellow (mark as discard eligible/low priority) regions. The **overall-rate** command marks the boundary between the yellow and red (drop) regions point for the associated policy (Figure 6).

Figure 6 **Profile Marking**



If the overall rate is set to 1000 pps and as long as the total traffic that is destined to the CPM and intended to be processed by the CPU is less than or equal to 1000 pps, all traffic will be processed. If the rate exceeds 1000 pps, then protocol traffic is discarded (or marked as discard eligible/low priority in the case of the **out-profile-rate**) and traffic on the interface is affected.

This rate limit protects all the other interfaces and ensures that a violation from one interface does not affect the rest of the system.

CPU protection is not supported on 7750 SR-1, 7750 SR-1s, 7750 SR-2s, 7750 SR-e, 7750 SR-a, and 7750 VSR.

2.3.4.1 Protocol Protection

Protocol protection allows traffic to be discarded for protocols not configured on the router. This helps mitigate DoS attacks by filtering invalid control traffic before it reaches the CPU. This is a feature of CPU Protection and can be enabled or disabled for the entire system.

When using **protocol-protection**, the system automatically maintains a per-interface list of configured protocols. For example, if an interface does not have IS-IS configured, then protocol protection will discard any IS-IS packets received on that interface. Other protocols, such as L2TP, are controlled by **protocol-protection** at the VPRN service level.

Protocols controlled by the **protocol-protection** mechanism include:

- GTP
- IGMP
- IS-IS
- MLD
- L2TP control
- OSPFv2

- OSPFv3
- PPPoE
- PIM
- RIP
- PFCP

The following protocols are protected independently from Protocol Protection:

- **per-peer-queuing** protects BGP, LDP, T-LDP, MSDP, Telnet and SSH
- BFD control packets are dropped if BFD is not configured on a specific interface

2.3.4.2 CPU Protection Extensions for ETH-CFM

CPU protection supports the ability to explicitly limit the amount of ETH-CFM traffic that arrives at the CPU for processing. ETH-CFM packets that are redirected to the CPU by either a Management Endpoint (MEP) or a Management Intermediate Point (MIP) will be subject to the configured limit of the associated policy. Up to four CPU protection policies may include up to ten individual ETH-CFM-specific entries. The ETH-CFM entries allow the operator to apply a packet-per-second rate limit to the matching combination of level and opcode for ETH-CFM packet that are redirected to the CPU. Any ETH-CFM traffic that is redirected to the CPU by a Management Point (MP) that does not match any entries of the applied policy is still subject to the overall rate limit of the policy itself. Any ETH-CFM packets that are not redirected to the CPU are not subject to this function and are treated as transit data, subject to the applicable QoS policy.

The operator first creates a CPU policy and includes the required ETH-CFM entries. Overlap is allowed for the entries within a policy, first match logic is applied. This means ordering the entries in the proper sequence is important to ensure the proper behavior is achieved. Even though the number of ETH-CFM entries is limited to ten, the entry numbers have a valid range from 1 to 100 to allow for ample space to insert policies between one and other.

Ranges are allowed when configuring the level and the OpCode. Ranges provide the operator a simplified method for configuring multiple combinations. When more than one level or OpCode is configured in this manner the configured rate limit is applied separately to each combination of level and OpCode match criteria. For example, if the levels are configured as listed in Table 14, with a range of five (5) to seven (7) and the OpCode is configured for 3,5 with a rate of 1. That restricts all possible combinations on that single entry to a rate of 1 packet-per-second. In this example, six different match conditions are created.

Table 14 **Ranges versus Levels and OpCodes**

Level	OpCode	Rate
5	3	1
5	5	1
6	3	1
6	5	1
7	3	1
7	5	1

Once the policy is created, it must be applied to a SAP or binding within a service for these rates to take effect. This means the rate is on a per-SAP or per-binding basis. Only one policy may be applied to each SAP or binding. The **eth-cfm-monitoring** option must be configured in order for the ETH-CFM entries to be applied when the policy is applied to the SAP or binding. If this option is not configured, ETH-CFM entries in the policy will be ignored. It is also possible to apply a policy to a SAP or binding by configuring **eth-cfm-monitoring** which does not have an MP. In this case, although these entries are enforced, no packets are redirect to the CPU.

By default, rates are applied on a per-peer basis. This means each individual peer is subject to the rate. Use the **aggregate** option to apply the rate to all peers. MIPs, for example, only respond to loopback messages and linktrace messages. These are typically on-demand functions and per-peer rate limiting is not required, making the aggregate function more appealing.

The **eth-cfm-monitoring** and **mac-monitoring** commands are mutually exclusive and cannot be configured on the same SAP or binding. The **mac-monitoring** command is used in combination with the traditional CPU protection and is not specific to ETH-CFM rate limiting feature described here.

When an MP is configured on a SAP or binding within a service which allows an external source to communicate with that MP, for example a User to Network Interface (UNI), **eth-cfm-monitoring** with the **aggregate** option should be configured on all SAPs or bindings to provide the highest level of rate control.

The example below shows a sample configuration for a policy and the application of that policy to a SAP in a VPLS service configured with an MP.

Policy 1 entry 10 limits all ETH-CFM traffic redirected to the CPU for all possible combinations to 1 packet-per-second. Policy 1 entry 20 limits all possible combinations to a rate of zero, dropping all request which match any combination. If entry 20 did not exist then only rate limiting of the entry 10 matches would occur and any other ETH-CFM packets redirected to the CPU would not be bound by a CPU protection rate.

```
config>sys>security>cpu-protection#
  policy 1
    eth-cfm
      entry 10 level 5-7 opcode 3,5 rate 1
      entry 20 level 0-7 opcode 0-255 rate 0

config>service>vpls#
  sap 1/1/4:100
    cpu-protection 1 eth-cfm-monitoring aggregate
    eth-cfm
      mip
    no shutdown
```

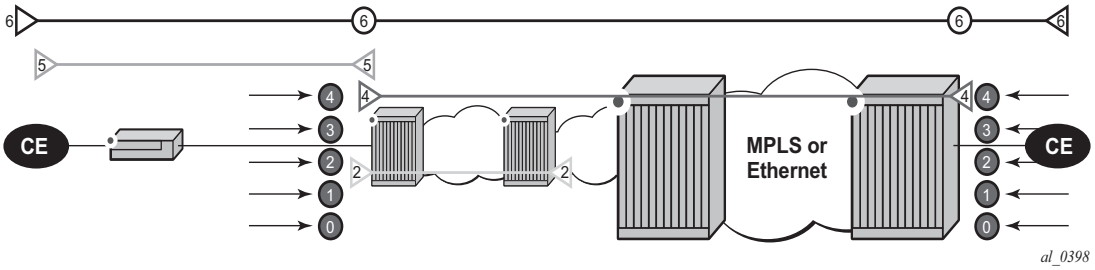
2.3.4.3 ETH-CFM Ingress Squelching

CPU protection provides a granular method to control which ETH-CFM packets are processed. As indicated in CPU Protection Extensions for ETH-CFM, a unique rate can be applied to ETH-CFM packets classifying on specific MD-level and a specific OpCode and applied to both ingress (down MEP and ingress MIP) and egress (up MEP and egress MIP) extraction. This function is to protect the CPU on extraction when a Management Point (MP) is configured.

It is also important to protect the ETH-CFM architecture deployed in the service provider network. This protection scheme varies from CPU protection. This model is used to prevent ETH-CFM frames at the service provider MD-levels from gaining access to the network even when extraction is not in place. ETH-CFM squelching drops all ETH-CFM packets at or below the configured MD-level. The ETH-CFM squelch feature is supported at ingress only.

Figure 7 shows a typical ETH-CFM hierarchical model with a subscriber ME (6), test ME (5), EVC ME (4) and an operator ME (2). This model provides the necessary transparency at each level of the architecture. For security reasons, it may be necessary to prevent errant levels from entering the service provider network at the UNI, ENNI, or other untrusted interconnection points. Configuring squelching at level four on both UNI-N interconnection ensures that ETH-CFM packets matching the SAP or binding delimited configuration will silently discard ETH-CFM packets at ingress.

Figure 7 **ETH-CFM Hierarchical Model**



Squelching configuration uses a single MD-level (0 to 7) to silently drop all ETH-CFM packets matching the SAP or binding delimited configuration at or below the specified MD-level. In Figure 7, a squelch level is configured at MD-level 4. This means the configuration will silently discard MD-levels 0,1,2,3 and 4, assuming there is a SAP or binding match.

Note: Extreme caution must be used when deploying this feature.

The operator is able to configure down MEPs and ingress MIPs that conflict with the squelched levels. This means that any existing MEP or MIP processing ingress CFM packets on a SAP or binding where a squelching policy is configured will be interrupted as soon as this command is entered into the configuration. These MPs will not be able to receive any ingress ETH-CFM frames because squelching is processed before ETH-CFM extraction.

CPU protection extensions for ETH-CFM are still required in the model above because the subscriber ME (6) and the test ME (5) are entering the network across an untrusted connection, the UNI. ETH-CFM squelching and CPU protection for ETH-CFM can be configured on the same SAP or binding. Squelching is processed followed by CPU protection for ETH-CFM.

MPs configured to support primary VLANs are not subjected to the squelch function. Primary VLAN-based MPs, supported only on Ethernet SAPs, are extractions that take into consideration an additional VLAN beyond the SAP configuration.

The difference in the two protection mechanisms is shown in the Table 15. CPU protection is used to control access to the CPU resources when processing is required. Squelching is required when the operator is protecting the ETH-CFM architecture from external sources.

Table 15 **CPU Protection and Squelching**

Description	CPU Protection Extension for ETH-CFM	ETH-CFM Squelching
Ingress Filtering	Yes	Yes
Egress Filtering	Yes	—
Granularity	Specified level and OpCode	Level (at and below)
Rate	Configurable rate (includes 0=drop all)	Silent drop
Primary VLAN Support	Rate shared with SAP delineation	Not exposed to squelch
Extraction	Requires MEP or MIP to extract	No MEP or MIP required

As well as including the squelching information under the **show service** *service-id* **all**, display output the **squelch-ingress-level** key also appears in the output of the **sap-using** and **sdp-using** show commands.

```
show service sap-using squelch-ingress-levels
===============================================================================
ETH-CFM Squelching
===============================================================================
PortId              SvcId      Squelch Level
-------------------------------------------------------------------------------
6/1/1:100.*         1          0 1 2 3 4 5 6 7
lag-1:100.*         1          0 1 2 3 4
6/1/1:200.*         2          0 1 2
lag-1:200.*         2          0 1 2 3 4 5
-------------------------------------------------------------------------------
Number of SAPs: 4
-------------------------------------------------------------------------------
===============================================================================
show service sdp-using squelch-ingress-levels
===============================================================================
ETH-CFM Squelching
===============================================================================
SdpId               SvcId       Type Far End                Squelch Level
-------------------------------------------------------------------------------
12345:4000000000    2147483650   Spok 10.1.1.1               0 1 2 3 4
===============================================================================
```

2.3.5 Distributed CPU Protection (DCP)

Distributed CPU Protection (DCP) is a rate-limiting function distributed to the line cards to rate-limit traffic extracted from the data path and sent to the CPM. DCP is performed in hardware and provides per-interface, access or network, and per-protocol granular rate-limit control.

DCP rate limiting is configured using policies that are applied to objects (for example, SAPs or network interfaces).

Per protocol rate-limit policers are configured in the DCP policy. The basic types of policers in DCP are:

- Enforcement policers — An instance of a policer that is policing a flow of packets comprised of a single (or small set of) protocols arriving on a single object (for example, SAP). Enforcement policers perform a configurable action (for example, discard) on packets that exceed configured rate parameters. There are two basic sub-types of enforcement policers:
  - Static policers — Always instantiate.
  - Dynamic policers — Only instantiated (allocated from a free pool of dynamic policers) when a local monitor detects nonconformance for a set of protocols on a specific object.
- Local monitors — A policer that is primarily used to measure the conformance of a flow comprised of multiple protocols arriving on a single object. Local monitors are used as a trigger to instantiate dynamic policers.

The use of dynamic policers reduces the number of policers required to effectively monitor and control a set of protocols across a large set of objects since the per-protocol-per-object dynamic policers are only instantiated when an attack or misconfiguration occurs, and they are only instantiated for the affected objects.

Figure 8 shows per SAP and protocol static rate limiting with DCP and Figure 9 shows per network interface and protocol static rate limiting with DCP.

Figure 8 **Per SAP Per-protocol Static Rate Limiting with DCP**

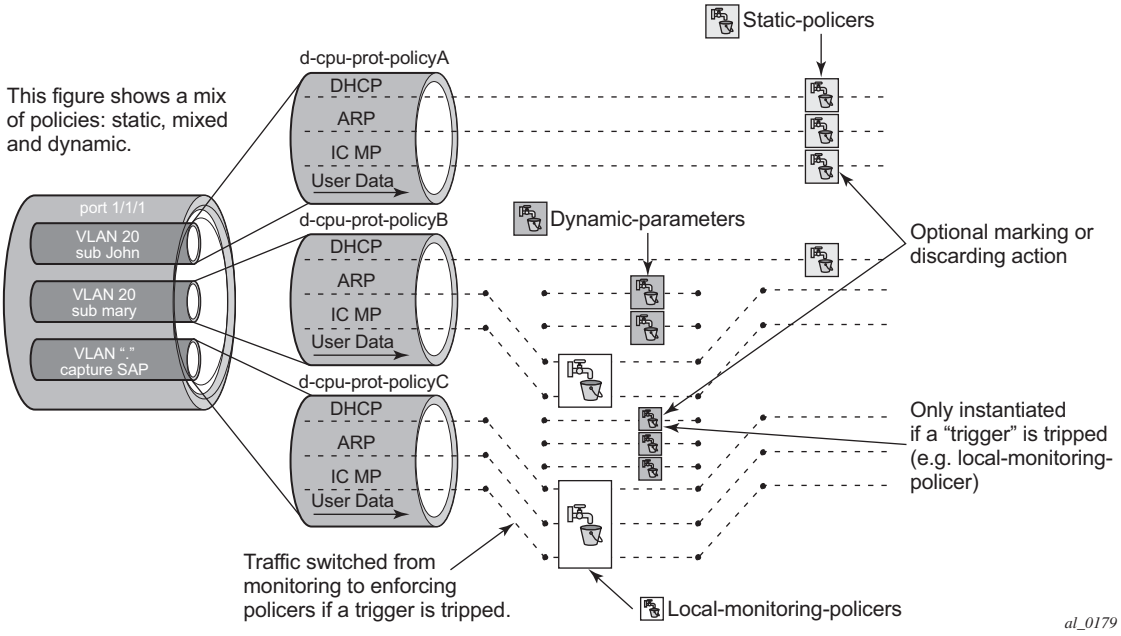
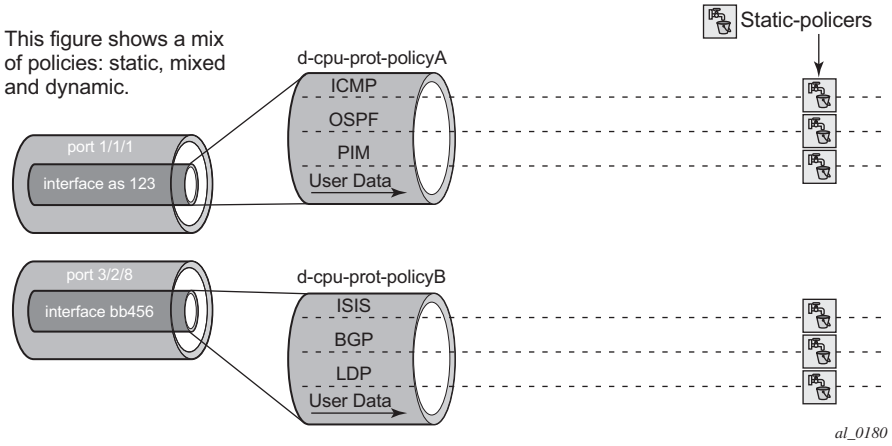


Figure 9 **Per Network Interface Per-protocol Static Rate Limiting with DCP**



2.3.5.1 Applicability of Distributed CPU Protection

The system assigns a default Distributed CPU Protection (DCP) policy to newly-created access and network interfaces. These policies, "_default-access-policy" and "_default-network-policy", are created empty by default and can be modified by the operator.

Additional DCP policies can be created for interfaces requiring a dedicated configuration. If DCP functionality is not required on a given access or network interface, then an empty DCP policy can be created and explicitly assigned to the interface.

DCP policies can be applied to the following types of objects:

- most types of SAPs on Layer 2 and Layer 3 services, including capture SAPs, SAPs on pseudowires, B-VPLS SAPs and VPLS template SAPs, but are not applicable to Epipe template SAPs and video ISA SAPs
- network interfaces

Control traffic that arrives on a network interface, but inside a tunnel (for example, SDP, LSP, PW) and logically terminates on a service (that is, traffic that is logically extracted by the service rather than the network interface layer itself) will bypass the DCP function. The control packets in this case will not be subject to the DCP policy that is assigned to the network interface on which the packets arrived. This helps to avoid customer traffic in a service from impacting other services or the operator's infrastructure.

Control packets that are extracted in a VPRN service, where the packets arrived into the node through a VPLS SAP (that is, R-VPLS scenario), will use the DCP policy and policer instances associated with the VPLS SAP. In this case, the DCP policy that an operator creates for use on VPLS SAPs, for VPLSs that have a Layer 3-interface bound to them (R-VPLS), may have protocols such as OSPF, ARP, configured in the policy.

2.3.5.2 Log Events, Statistics, Status, and SNMP support

A comprehensive set of log events are supported for DCP in order to alert the operator to potential attacks or misconfigurations and to allow tuning of the DCP settings. Refer to the NOTIFICATION-TYPE objects with "Dcp" in the names in the following MIBs for details:

- TIMETRA-CHASSIS-MIB
- TIMETRA-SAP-MIB
- TIMETRA-VRTR-MIB

The log events can also be seen in the CLI using the following **show log event-control | match Dcp** command.

DCP throttles the rate of DCP events to avoid event floods when multiple parallel attacks or problems are occurring.

Many of the DCP log events can be individually enabled or disabled at the DCP policy level (in the DCP policy config) as well as globally in the system (in log event-control).

If needed, when a DCP log event indicates a SAP, and that SAP is an MSAP, the operator can determine which subscribes are on a specific MSAP by using the **show service active-subs** command and then filtering ("| match") on the MSAP string.

Statistics and status related to DCP are available both through:

- CLI
- SNMP — See various tables and objects with "Dcp" or "DCpuProt" in their name in the TIMETRA-CHASSIS-MIB¸ TIMETRA-SECURITY-MIB, TIMETRA-SAP-MIB and TIMETRA-VRTR-MIB

2.3.5.3 DCP Policer Resource Management

The policer instances are a limited hardware resource on a given forwarding plane. DCP policers (static, dynamic, local-monitor) are consumed from the overall forwarding plane policer resources (from the ingress resources if ingress and egress are partitioned). Each per-protocol policer instantiated reduces the number of FP child policers available for other purposes.

When DCP is configured with dynamic enforcement, then the operator must set aside a pool of policers that can be instantiated as dynamic enforcement policers. The number of policers reserved for this function are configurable per card or FP. The policers in this pool are not available for other purposes (normal SLA enforcement).

Static enforcement policers and local monitoring policers use policers from the normal or global policer pool on the card or FP. Once a static policer is configured in a DCP policy and it is referenced by a protocol in the policy, then this policer will be instantiated for each object (SAP or network interface) that is created and references the policy. If there is no policer free on the associated card or FP, then the object will not be created. Similarly, for local monitors, once a local monitoring policer is configured and referenced by a protocol, then this policer will be instantiated for each object that is created and references the policy. If there is no policer free, then the object will not be created.

Dynamic enforcement policers are allocated as needed (when the local monitor detects nonconformance) from the reserved dynamic enforcement policer pool.

When a DCP policy is applied to an object on a LAG, then a set of policers is allocated on each FP (on each line card that contains a member of the LAG). The LAG mode is ignored and the policers are always shared by all ports in the LAG on that forwarding plane on the SAP or interface. In other words, with link-mode lag a set of DCP policers are not allocated per-port in the LAG on the SAP.

In order to support large scale operation of DCP, and also to avoid overload conditions, a polling process is used to monitor state changes in the policers. This means there can be a delay between when an event occurs in the data plane and when the relevant state change or event notification occurs towards an operator, but in the meantime the policers are still operating and protecting the control plane.

2.3.5.4 Operational Guidelines and Tips

The following points offer various optional guidelines that may help an operator decide how to leverage Distributed CPU Protection.

- The rates in a policy assigned to a capture SAP should be higher than those assigned to MSAPs that will contain a single subscriber. The rates for the capture sap policy should allow for a burst of MSAP setups.
- To completely block a set of specific protocols on a given SAP, create a single static policer with a rate of 0 and map the protocols to that policer. Dynamic policers and local monitors can not be used to simultaneously allow some protocols but block others (the non-zero rates in the monitor would let all protocols slip through at a low rate).
- During normal operation it is recommended to configure “log-events” (no verbose keyword) for all static policers, in the dynamic parameters of all protocols and for all local monitoring policers. The verbose keyword can be used selectively during debug, testing, tuning, and investigations.
- Packet-based rate limiting is generally recommended for low-rate subscriber-based protocols whereas kb/s rate limiting is recommended for higher rate infrastructure protocols (such as BGP).
- It is recommended to configure an **exceed-action** of low-priority for routing and infrastructure protocols. Marked packets are more likely to be discarded if there is congestion in the control plane of the router, but will get processed if there is no contention for CPU resources allowing for a work-conserving behavior in the CPM.
- In order to assign a different **dist-cpu-protection** policy to a specific MSAP instance or to all MSAPs for a specific MSAP policy, the operator can assign a new **dist-cpu-protection** policy to the MSAP policy and then use the **eval-msap** tool:

A:nodeA>tools>perform# subscriber-mgmt eval-msap
- **eval-msap** {**policy** *msap-policy-name* | **msap** *sap-id*}

Note: Any new MSAPs will also be assigned the new **dist-cpu-protection** policy.

- If needed, an operator can determine which subscriber is on a specific MSAP by using the **show service active-subs** command and then filtering ("| match") on the MSAP string.
- If protocol is trusted, and using the "all-unspecified" protocol is not required, then avoid referencing this protocol in the policy configuration.
- If a protocol is trusted, but the all-unspecified bucket is required, then there are two options:
  - avoid creating a protocol so that it is treated as part of the all-unspecified bucket (but account for the packets from X in the all-unspecified rate and local-mon rate)
  - create this protocol and configure it to bypass

2.3.6 Classification-Based Priority for Extracted Protocol Traffic

The SR OS supports a set of mechanisms to protect the router control and management planes from various types of attacks, floods, and misconfigurations. Many of the mechanisms operate by default with no need for operator configuration or intervention.

One class of mechanisms employed on the router to protect against floods of control traffic involves identifying potentially harmful or malicious traffic through the use of rate measurements. Centralized CPU protection protects and isolates interfaces from each other by default by treating unexpectedly high rate control traffic on an interface as lower priority (to be discarded if the control plane experiences congestion). Distributed CPU protection can protect and isolate at a per-protocol, per-interface granularity through configured rate profiles. These rate-based protection mechanisms make no assumptions about the contents of the packets and can be used when nothing about the packets can be trusted (for example, DSCP or source IP address, which can be spoofed).

The SR OS also supports an alternative to rate-based mechanisms for cases where the packet headers can be trusted to differentiate between good and bad control traffic. A configurable prioritization scheme can be enabled (using the **init-extract-prio-mode l3-classify** command) on a per-FP basis to initialize the drop priority of all Layer 3 extracted control traffic based on the QoS classification of the packets. This is useful, for example, in networks where the DSCP and EXP markings can be trusted as the primary method to distinguish, protect, and isolate good terminating protocol traffic from unknown or potentially harmful protocol traffic instead of using the rate-based distributed CPU protection and centralized CPU protection traffic marking/coloring mechanisms (for example, **out-profile-rate** and **exceed-action low-priority**).

The operational guidelines for deploying classification-based priority for extracted control traffic are as follows.

- Centralized CPU protection should be effectively disabled for all interfaces/SAPs on FPs configured in **l3-classify** mode by changing some CPU protection policy parameters from their default values. This is required so that centralized CPU protection does not re-mark good control traffic (traffic that was initially classified as high priority) as low priority if a flood attack occurs on the same interface. Effectively disabling centralized CPU protection can be done by ensuring that:
  - a rate value of **max** is configured for **port-overall-rate** (**max** is the default value for **port-overall-rate**)
  - all objects (interfaces, MSAP policies, and SAPs) that can be assigned a CPU protection policy are referencing a policy that sets the **out-profile-rate** to **max** and the **overall-rate** to **max** (this can be done in the two default CPU protection policies if all FPs in the system are in **l3-classify** mode)
- DCP can be used in conjunction with **l3-classify** mode, but care must be taken to prevent DCP from acting on protocols where the operator wants to use QoS classification (such as DSCP or EXP) to differentiate between good and bad Layer 3 packets. On an FP with **l3-classify** mode, DCP should be configured so that BGP, LDP, and other protocols do not have their initial drop priority (color) overwritten by DCP if the QoS classification of these protocols is trusted. This can be achieved by using **exceed-action none** for those protocols in a DCP policy. For other protocols where QoS classification cannot be used to distinguish between good and bad extracted packets, DCP can be used to color the packets with a drop priority based on a configured rate.
- If any LAG member is on an FP in **l3-classify** mode, all FPs that host the other members of that LAG should also be in **l3-classify** mode.
- The QoS classification rules that are used on interfaces/SAPs on FPs in **l3-classify** mode should be configured to differentiate between good and bad control traffic. The default network ingress QoS policies do differentiate (for example, based on DSCP), but the default access ingress QoS policies do not.

The **l3-classify** mode for extracted control traffic is supported on the 7750 SR and 7950 XRS.

2.3.7 TTL Security

The SR OSTTL security evaluates the value of the incoming packets against a maximum TTL value configured in the system. This capability, also known as Generalized TTL Security Mechanism (GTSM) defined in RFC 5082, is supported for BGP, LDP, SSH and Telnet. If the incoming TTL value is less than the configured TTL value, the packets are discarded and a log is generated preventing attackers generating spoof traffic with larger number of hops than expected.

The TTL value is configurable on a per-peer basis for BGP and LDP and configurable at the system level for SSH and Telnet.

The TTL security mechanism was originally designed to protect the BGP infrastructure where the vast majority of ISP External Border Gateway Protocol (EBGP) peerings are established between adjacent routers. Since TTL spoofing cannot be performed, a mechanism based on an expected TTL value provides a simple and robust defense from infrastructure attacks based on forged BGP packets.

While TTL security is most effective in protecting directly-connected BGP or LDP peers, it can also provide protection to multi-hop sessions. For multi-hop sessions the expected TTL value can be set to 255 minus the configured range of hops.

2.3.8 Management Access Filter

Management Access Filters (MAF) are software-based filters used to restrict both traffic extracted from the data plane and traffic from the management port to the CPU.

2.3.8.1 MAF Filter Packet Match

Three different **management-access-filter** policies can be configured: **ip-filter**, **ipv6-filter**, and **mac-filter**. Each policy is an ordered list of entries. For this reason, entries must be sequenced correctly from the most to the least explicit.

Management Access filter (MAF) packet match rules:

- Each MAF policy is an ordered list of entries, therefore entries must be sequenced correctly from the most to the least explicit.
- If multiple match criteria are specified in a single MAF filter policy entry, all criteria must be met for the packet to be considered a match against that policy entry (logical AND).
- Any match criteria not explicitly defined is ignored during a match.
- A MAF filter policy entry with match criteria defined, but no action configured, inherits the default action.
- The **management-access-filter default-action** applies individually per IPv4, IPv6, or MAC CPM filter policies that are in a **no shutdown** state.
- When both **mac-filter** and **ip-filter** or **ipv6-filter** are applied to a specific packet, **mac-filter** is applied first.

2.3.8.2 MAF IPv4/IPv6 Filter Entry Match Criteria

Table 16 lists the supported IPv4 and IPv6 match criteria.

Table 16 **IPv4 and IPv6 Match Criteria**

Criteria	Description
src-ip	Matches the specified source IPv4 or IPv6 address prefix and mask against the source IPv4 or IPv6 address field in the IP packet header. IPv4 and IPv6 matching prefix-lists can be used to enhance matching capabilities.
next-header	Matches the specified upper-layer protocol (such as TCP, UDP, or IGMPv6) against the next-header field of the IPv6 packet header. "*" can be used to specify a TCP or UDP upper-layer protocol match (Logical OR). Next-header matching allows also matching on presence of a subset of IPv6 extension headers. See the CLI section for details on which extension header match is supported.
protocol	Matches the specified protocol against the Protocol field in the IPv4 packet header (for example, TCP, UDP, or IGMP) of the outer IPv4. "*" can be used to specify TCP or UDP upper-layer protocol match (Logical OR).
dst-port	Matches the specified port value against the destination port number of the UDP or TCP packet header.
flow-label	Matches the IPv6 flow label.
router	Matches the router instance packets that are ingressing from for this filter entry.

Table 16 **IPv4 and IPv6 Match Criteria (Continued)**

Criteria	Description
src-port	Matches the port packets that are ingressing from for this filter entry.

2.3.8.3 MAF MAC Filter Entry Match Criteria

Table 17 describes the supported MAC match criteria. The criteria are evaluated against the Ethernet header of the Ethernet frame.

Table 17 **Router Instance Match Criteria**

Criteria	Description
frame-type	Matches a specific type of frame format.
src-mac	Matches the specified source MAC address frames. Optionally, operators can configure a mask to be used in a match.
dst-mac	Matches the specified destination MAC address frames. Optionally, operators can configure a mask to be used in a match.
dot1p	Matches 802.1p frames. Optionally, operators can configure a mask to be used in a match.
etype	Matches the specified Ethernet II frames. The Ethernet type field is a two-byte field used to identify the protocol carried by the Ethernet frame.
snap-oui	Matches frames with the specified three-byte OUI field.
snap-pid	Matches frames with the specified two-byte protocol ID that follows the three-byte OUI field.
ssap	Matches the specified frames with a source access point on the network node designated in the source field of the packet. Optionally, operators can configure a mask to be used in a match.
dsap	Matches the specified frames with a destination access point on the network node designated in the destination field of the packet. Optionally, operators can configure a mask to be used in a match.
cfm-opcode	Matches the specified packet with the specified **cfm-opcode**.
svc-id	Matches the service ID packets are ingressing from.
svc-name	Matches the service name packets are ingressing from.

2.3.8.4 MAF Filter Policy Action

A management access filters allow to **permit** or **deny** (or use deny-host-unreachable response for IP filters) traffic.

2.3.8.5 MAF Filter Policy Statistics and Logging

Management access filter match count can be displayed using **show** commands. Logging is recorded in the system security logs.

2.4 Vendor-Specific Attributes (VSAs)

The software supports the configuration of Nokia-specific RADIUS attributes. These attributes are known as vendor-specific attributes (VSAs) and are discussed in RFC 2138. VSAs must be configured when RADIUS authorization is enabled. It is up to the vendor to specify the format of their VSA. The attribute-specific field is dependent on the vendor's definition of that attribute. The Nokia-defined attributes are encapsulated in a RADIUS vendor-specific attribute with the vendor ID field set to 6527, the vendor ID number.

Note: The PE-record entry is required to support the RADIUS Discovery for Layer 2 VPN feature. A PE-record is only relevant if the RADIUS Discovery feature is used, not for the standard RADIUS setup.

The following RADIUS vendor-specific attributes (VSAs) are supported by Nokia.

- timetra-access <ftp> <console> <both> — This is a mandatory command that must be configured. This command specifies if the user has FTP and /or console (serial port, Telnet, and SSH) access.
- timetra-profile <profile-name> — When configuring this VSA for a user, it is assumed that the user profiles are configured on the local router and the following applies for local and remote authentication:
  1. The authentication-order parameters configured on the router must include the local keyword.
  2. The user name may or may not be configured on the router.
  3. The user must be authenticated by the RADIUS server.

4. Up to 8 valid profiles can exist on the router for a user. The sequence in which the profiles are specified is relevant. The most explicit matching criteria must be ordered first. The process stops when the first complete match is found.

   If all the above mentioned conditions are not met, then access to the router is denied and a failed login event/trap is written to the security log.

- timetra-default-action <permit-all | deny-all | none> — This is a mandatory command that must be configured even if the timetra-cmd VSA is not used. This command specifies the default action when the user has entered a command and no entry configured in the timetra-cmd VSA for the user resulted in a match condition.
- timetra-cmd <match-string> — Configures a command or command subtree as the scope for the match condition.

  The command and all subordinate commands in subordinate command levels are specified.

2.5 Other Security Features

This section describes the other security features supported by the SR OS.

2.5.1 SSH

Secure Shell (SSH) is a protocol that provides a secure, encrypted Telnet-like connection to a router. A connection is always initiated by the client (the user). Authentication takes places by one of the configured authentication methods (local, RADIUS, TACACS+, and LDAP). With authentication and encryption, SSH allows for a secure connection over an insecure network.

The OS allows the administrator to configure Secure Shell version 1 (SSHv1) and version 2 (SSHv2). SSHv1 and SSHv2 are different protocols and encrypt at different parts of the packets. SSHv2 does not use the same networking implementation that SSHv1 does and is considered a more secure, efficient, and portable version of SSH.

SSH runs on top of a transport layer (like TCP or IP), and provides authentication and encryption capabilities.

The OS has a global SSH server process to support inbound SSH, sFTP, NETCONF, and SCP sessions initiated by external client applications. This server process is separate from the SSH and SCP client commands on the routers which initiate outbound SSH and SCP sessions.

Inbound SSH, Telnet, and FTP sessions are counted separately and it is possible to set the limit for each type separately in the **config>system>login-control** submenu, However there is a maximum total of 50 sessions for SSH and Telnet together. SCP, sFTP, and NETCONF sessions are counted as SSH sessions.

When the SSH server is enabled, an SSH security key is generated. The key is only valid until the node is restarted or the SSH server is stopped and restarted (unless the **preserve-key** option is configured for SSH). The key size is non-configurable and set at 1024 bits or 2048 bits in FIPS mode. When the server is enabled, all inbound SSH, SCP, sFTP, and NETCONF sessions will be accepted provided the session is properly authenticated.

When the global SSH server process is disabled, no inbound SSH, SCP, sFTP, or NETCONF sessions will be accepted.

When using SCP to copy files from an external device to the file system, the SCP server will accept either forward slash (“/”) or backslash (“\”) characters to delimit directory and/or filenames. Similarly, the SCP client application can use either slash or backslash characters, but not all SCP clients treat backslash characters as equivalent to slash characters. In particular, UNIX systems will often times interpret the backslash character as an “escape” character which does not get transmitted to the SCP server. For example, a destination directory specified as “cf1:\dir1\file1” will be transmitted to the SCP server as “cf1:dir1file1” where the backslash escape characters are stripped by the SCP client system before transmission. On systems where the client treats the backslash like an “escape” character, a double backslash “\\” or the forward slash “/” can typically be used to properly delimit directories and the filename.

There are three pairs of configurable lists: cipher lists, MAC lists, and KEX lists. In each pair one list is dedicated to the SSH server and second to the SSH/SCP client. These can be configured for negotiation of the best compatible cipher, MAC, and KEX algorithm between the client and server. The lists can be created and managed under the **security ssh** submenu. The client lists are used when the SR OS is acting as the SSH client and the server lists are used when the SR OS is acting as a server. The first algorithm matched on the lists between the client and server is the preferred algorithm for the session.

SSHv2 authentication methods supported by the SR OS are password, keyboard-interactive, and public key.

Note: SSHv1 is not supported when the node is running in FIPS-140-2 mode.

2.5.1.1 SSH PKI Authentication

The SSH server also supports a public key authentication as long as the server has been previously configured to know the client's public key.

Using Public Key authentication (also known as Public Key Infrastructure - PKI) can be more secure than the existing username and password method because:

- A user will typically re-use the same password with multiple servers. If the password is compromised, the user must reconfigure the password on all affected servers.
- A password is not transmitted between the client and server using PKI. Instead the sensitive information (the private key) is kept on the client. Therefore the password is less likely to be compromised.

SR OS supports server-sider SSHv2 public key authentication but does not include a key-generation utility.

Support for PKI should be configured in the system-level configuration where one or more public keys may be bound to a username. This configuration will not affect any other system security or login functions.

PKI has preference over password or keyboard authentication. PKI is supported using local authentication and using an AAA server with LDAP only. PKI authentication is not supported on TACACS+ or RADIUS.

2.5.1.1.1 User Public Key Generation

Before SSH can be used with PKI, someone must generate a public/private key pair. This is typically supported by the SSH client software. For example, PuTTY supports a utility called PuTTYGen that will generate key pairs.

SR OS currently supports only RSA and ECDSA user public keys.

If the client is using PuTTY, they will first generate a key pair using PuTTYGen. The user sets the key type to SSH-2 RSA and sets the number of bits to be used for the key. The user can also configure a passphrase that is used to store the key locally in encrypted form. If the passphrase is configured, the user must enter the passphrase in order to use the private key, acting as a password for the private key. If a passphrase is not used, the key is stored in plain text locally.

Next, the public key must be configured for the user on SR OS using the **config**>**system**>**security**>**user**>**public-keys** command. On the SR OS, the user can program the public key using Telnet/SSH or SNMP.

2.5.1.2 MAC Client and Server List

SR OS supports a configurable server and client MAC list for SSHv2. This allows the user to add or remove MAC algorithms from the list. The user can program the strong HMAC algorithms on top of the configurable MAC list (for example, lowest index in the list) in the order to be negotiated first between the client and server. The first algorithm in the list that is supported by both the client and the server is the one that is agreed upon.

There are two configurable MAC lists:

- server list
- client list

The default MAC list includes all supported algorithms with the following preference:

- mac 200 name hmac-sha2-512
- mac 210 name hmac-sha2-256
- mac 215 name hmac-sha1
- mac 220 name hmac-sha1-96
- mac 225 name hmac-md5
- mac 230 name hmac-ripemd160
- mac 235 name hmac-ripemd160-openssh-com
- mac 240 name hmac-md5-96

Note: Configurable MAC list is only supported for SSHv2 and not supported for SSHv1. SSv1 only supports 32-bit CRC.

2.5.1.3 KEX Client and Server List

SR OS supports KEX client and server lists. The user can remove or add the desired KEX client/server algorithms to be negotiated using an SSHv2 phase one handshake. The list is an index list with the lower index having higher preference in the SSH negotiation. The lowest index algorithm in the list will be negotiated first in SSH and will be on top of the negotiation list to the peer.

By default the KEX list is empty and this hard-coded list with all supported algorithms and the following preference is used:

- kex 200 name diffie-hellman-group16-sha512

- kex 210 name diffie-hellman-group14-sha256
- kex 215 name diffie-hellman-group14-sha1
- kex 220 name diffie-hellman-group-exchange-sha1
- kex 225 name diffie-hellman-group1-sha1

As soon as any algorithm is configured in the KEX list, the SR OS will start using the user-defined KEX list instead of the hard-coded list. To go back to the hard-coded list, the user must remove all configured KEX indexes until the list is empty.

The CLI used is inline with cipher/mac server/client list and is as follow:

```
configure system security ssh server-kex-list kex
   kex <index> name <kex-name>
   no kex <index>

configure system security ssh client-kex-list kex
   kex <index> name <kex-name>
   no kex <index>

<index>              : [1..255]
<kex-name>           : diffie-hellman-group14-sha1| diffie-hellman-group14-sha256|
                       diffie-hellman-group16-sha512|
     diffie-hellman-group-exchange-sha1| diffie-hellman-group1-sha1
```

2.5.1.4 Regenerate the SSH key without disabling SSH

Two releases ago, SR OS did not periodically rollover the SSH symmetric key. SR OS now supports periodic rollover of the SSH symmetric key. Symmetric key rollover is important in long SSH sessions. Symmetric key rollover ensures that the encryption channel between the client and server is not jeopardized by an external hacker that is trying to break the encryption via a brute force attack.

This feature introduces symmetric key rollover on SSH client or server. The following are triggers for symmetric key rollover and negotiation:

- the negotiation of the key base on a configured time period
- the negotiation of the key base on a configured data transmission size

For extra security, by default, the key re-exchange is enabled under SR OS. The default values are as follow:

```
client
    bytes 1000000000
    minutes 60
    no shutdown
 exit
 server
    bytes 1000000000
```

```
            minutes 60
            no shutdown
        exit
```

2.5.1.4.1 Key re-exchange procedure

Key re-exchange is started by sending an SSH_MSG_KEXINIT packet while not already doing a key exchange. When this message is received, a party must respond with its own SSH_MSG_KEXINIT message, except in cases where the received SSH_MSG_KEXINIT already was a reply. Either party may initiate the re-exchange, but roles must not be changed (for example, the server remains the server, and the client remains the client).

Key re-exchange is performed using whatever encryption was in effect when the exchange was started. Encryption, compression, and MAC methods are not changed before a new SSH_MSG_NEWKEYS is sent after the key exchange (as in the initial key exchange). Re-exchange is processed identically to the initial key exchange, except that the session identifier will remain unchanged. Some or all of the algorithms can be changed during the re-exchange. Host keys can also change. All keys and initialization vectors are recomputed after the exchange. Compression and encryption contexts are reset.

RFC 4253 recommends key exchange after every hour or 1Gbytes of transmitted data, which is met by SR OS default implementation.

SR OS can roll over keys via two mechanisms:

- bytes (default is 1 Gbyte and the keys will be negotiated)
- minutes (default is 1 minute)

Note:

- If both the bytes and minutes key rollover mechanisms are configured, the key rollover happens based on whichever occurs first.
- If these parameters change, only new SSH connections inherit them. The existing SSH connections continue to use the previously configured parameters.

2.5.1.5 Cipher Client and Server List

SR OS supports cipher client and server lists. The user can add or remove the desired SSH cipher client/server algorithms to be negotiated. The list is an index list with the lower index having higher preference in the SSH negotiation. The lowest index algorithm in the list will be negotiated first in SSH and will be on top of the negotiation list to the peer.

There is separate cipher list for SSHv1 and SSHv2 for both client and server.

The default client cipher list for SSHv1 includes all supported algorithms with the following preference:

- cipher 200 name 3des
- cipher 205 name blowfish
- cipher 210 name des

The default Server cipher list for SSHv1 includes algorithms in the following preference order:

- cipher 200 name 3des
- cipher 205 name blowfish

The default server and client lists for SSHv2 include all supported algorithms with the following preference:

- cipher 190 name aes256-ctr
- cipher 192 name aes192-ctr
- cipher 194 name aes128-ctr
- cipher 200 name aes128-cbc
- cipher 205 name 3des-cbc
- cipher 210 name blowfish-cbc
- cipher 215 name cast128-cbc
- cipher 220 name arcfour
- cipher 225 name aes192-cbc
- cipher 230 name aes256-cbc
- cipher 235 name rijndael-cbc

The CLI used to configure client/server cipher list is as follows:

```
configure system security ssh server-cipher-list
    server-cipher-list protocol-version <version>
 <version>            : [1..2]
```

```
configure system security ssh server-cipher-list protocol-version 2 ciper
    no cipher <index>
    cipher <index> name <cipher-name>

<index>              : [1..255]
 <cipher-name> : aes128-ctr | aes192-ctr | aes256-ctr | 3des-cbc| blowfish-
cbc|cast128-cbc | arcfour | aes128-cbc | aes192-cbc | aes256-cbc | rijndael-cbc
```

2.5.2 Exponential Login Backoff

A malicious user may attempt to gain CLI access by means of a dictionary attack using a script to automatically attempt to login as an “admin” user and using a dictionary list to test all possible passwords. Using the exponential-back off feature in the **config**>**system**>**login-control** context the OS increases the delay between login attempts exponentially to mitigate attacks.

A malicious user may attempt to gain CLI access by means of a dictionary attack using a script to automatically attempt to login as an “admin” user and using a dictionary list to test all possible passwords. Using the exponential-back off feature in the **config**>**system**>**login-control** context the OS increases the delay between login attempts exponentially to mitigate attacks.

When a user tries to login to a router using a Telnet or an SSH session, there are a limited number of attempts allowed to authenticate a user. The interval between the unsuccessful attempts change after each try (1, 2 and 4 seconds). If the system is configured for user lockout, then the user will be locked out when the number of attempts is exceeded.

However, if lockout is not configured, there are three password entry attempts allowed after the first failure, at fixed 1, 2 and 4 second intervals, in the first session, and then the session terminates. Users do not have an unlimited number of login attempts per session. After each failed authentication attempt, the wait period becomes longer until the maximum number of attempts is reached.

The OS terminates after four unsuccessful tries. A wait period will never be longer than 4 seconds. The periods are fixed and will restart in subsequent sessions.

The **config**>**system**>**login-control**>[**no**] **exponential-backoff** command works in conjunction with the **config**>**system**>**security**>**password**>**attempts** command, which is also a system wide configuration.

For example:

```
*A:ALA-48>config>system# security password attempts
  - attempts <count> [time <minutes1>] [lockout <minutes2>]
  - no attempts
```

```
<count>              : [1..64]
<minutes1>           : [0..60]
<minutes2>           : [0..1440]
```

Exponential backoff applies to any user and by any login method such as console, SSH and Telnet.

Refer to Configuring Login Controls.

2.5.3 User Lockout

When a user exceeds the maximum number of attempts allowed (the default is 3 attempts) during a certain period of time (the default is 5 minutes), the account used during those attempts will be locked out for a pre-configured lock-out period (the default is 10 minutes).

A security or LI event log will be generated as soon as a user account has exceeded the number of allowed attempts, and the **show**>**system**>**security**>**user** command can be used to display the total number of failed attempts per user.

In addition to the security or LI event log, an SNMP trap is also generated so that any SNMP server (including the NSP NFM-P) can use the trap for an action.

The account will be automatically re-enabled as soon as the lock-out period has expired. The list of users who are currently locked out can be displayed with the **show**>**system**>**security**>**lockout** command.

A lock-out for a specific user can be administratively cleared using the **admin**>**user** *user-name*>**clear-lockout** command.

2.5.4 CLI Login Scripts

The SR OS supports automatic execution of CLI scripts when a user successfully logs into the router and starts a CLI session.

Users who authenticate via the local user database can use the configurable **config**>**system**>**security**>**user** *user-name*>**console**>**login-exec** *file-url* login exec script.

A global login-script can be configured to execute a common script when any user logs into CLI. A per user login-script can also be configured to execute when a specific user logs into CLI. These login-scripts execute whether the user was authenticated via the local user database, TACACS+ or RADIUS. The scripts can be used, for example, to define a common set of CLI aliases that are made available on the router for all users.

To configure a global login exec script, use the **config**>**system**>**login-control**>l**ogin-scripts**> **global** *file-url* script.

To configure a user-specific login exec script, use the **config**>**system**>**login-control**>**login-scripts**>**per-user**>**user-directory**>*file-url* **file-name** *file-name* script.

2.5.5 802.1x Network Access Control

The SR OS supports network access control of client devices (PCs, STBs, etc.) on an Ethernet network using the IEEE. 802.1x standard. 802.1x is known as Extensible Authentication Protocol (EAP) over a LAN network or EAPOL.

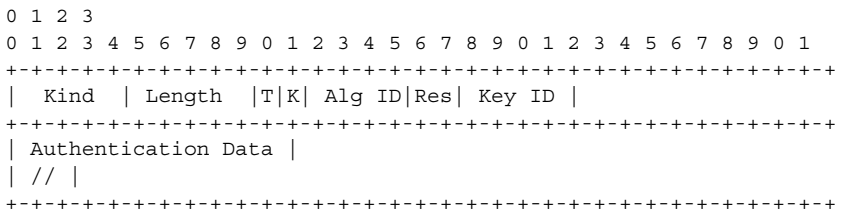
2.5.6 TCP Enhanced Authentication Option

The TCP Enhanced Authentication Option, currently covered in RFC 5925, *The TCP Authentication Option*, extends the previous MD5 authentication option to include the ability to change keys without tearing down the session, and allows for stronger authentication algorithms to be used.

The TCP Enhanced Authentication Option is a TCP extension that enhances security for BGP, LDP and other TCP-based protocols. This includes the ability to change keys in a BGP or LDP session seamlessly without tearing down the session. It is intended for applications where secure administrative access to both the end-points of the TCP connection is normally available.

TCP peers can use this extension to authenticate messages passed between one another. This strategy improves upon current practice, which is described in RFC 2385, *Protection of BGP Sessions via the TCP MD5 Signature Option*. Using this new strategy, TCP peers can update authentication keys during the lifetime of a TCP connection. TCP peers can also use stronger authentication algorithms to authenticate routing messages.

2.5.6.1 Packet Formats



Option Syntax

- Kind: 8 bits

  The Kind field identifies the TCP Enhanced Authentication Option. This value will be assigned by IANA.

- Length: 8 bits

  The Length field specifies the length of the TCP Enhanced Authentication Option, in octets. This count includes two octets representing the Kind and Length fields.

  The valid range for this field is from 4 to 40 octets, inclusive.

  For all algorithms specified in this memo the value will be 16 octets.

- T-Bit: 1 bit

  The T-bit specifies whether TCP Options were omitted from the TCP header for the purpose of MAC calculation. A value of 1 indicates that all TCP options other than the Extended Authentication Option were omitted. A value of 0 indicates that TCP options were included.

  The default value is 0.

- K-Bit: 1 bit

  This bit is reserved for future enhancement. Its value must be equal to zero.

- Alg ID: 6 bits

  The Alg ID field identifies the MAC algorithm.

- Res: 2 bits

  These bits are reserved. They must be set to zero.

  Key ID: 6 bits

  The Key ID field identifies the key that was used to generate the message digest.

- Authentication Data: Variable length

- The Authentication Data field contains data that is used to authenticate the TCP segment. This data includes, but need not be restricted to, a MAC. The length and format of the Authentication Data Field can be derived from the Alg ID.

- The Authentication for TCP-based Routing and Management Protocols draft provides and overview of the TCP Enhanced Authentication Option. The details of this feature are described in draft-bonica-tcp-auth-04.txt.

2.5.6.2 Keychain

The keychain mechanism allows for the creation of keys used to authenticate protocol communications. Each keychain entry defines the authentication attributes to be used in authenticating protocol messages from remote peers or neighbors, and it must include at least one key entry to be valid. Through the use of the keychain mechanism, authentication keys can be changed without affecting the state of the associated protocol adjacencies for OSPF, IS-IS, BGP, LDP, and RSVP-TE.

Each key within a keychain must include the following attributes for the authentication of protocol messages:

- key identifier
- authentication algorithm
- authentication key
- direction
- start time

In addition, additional attributes can be optionally specified, including:

- end time
- tolerance

Table 18 shows the mapping between these attributes and the CLI command to set them.

Table 18 **Keychain Mapping**

Definition	CLI
The key identifier expressed as an integer (0...63)	config>system>security>keychain>direction>bi>entry config>system>security>keychain>direction>uni>receive>entry config>system>security>keychain>direction>uni>send>entry

Table 18 **Keychain Mapping (Continued)**

Definition	CLI
Authentication algorithm to use with key[i]	config>system>security>keychain>direction>bi>entry with algorithm *algorithm* parameter. config>system>security>keychain>direction>uni>receive>entry with algorithm *algorithm* parameter. config>system>security>keychain>direction>uni>send>entry with algorithm *algorithm* parameter.
Shared secret to use with key[i].	config>system>security>keychain>direction>uni>receive>entry with shared secret parameter config>system>security>keychain>direction>uni>send>entry with shared secret parameter config>system>security>keychain>direction>bi>entry with shared secret parameter
A vector that determines whether the key[i] is to be used to generate MACs for inbound segments, outbound segments, or both.	config>system>security>keychain>direction
Start time from which key[i] can be used.	config>system>security>keychain>direction>bi>entry>begin-time config>system>security>keychain>direction>uni>send>entry >begin-time
End time after which key[i] cannot be used by sending TCPs.	Inferred by the begin-time of the next key (youngest key rule).
Start time from which key[i] can be used.	config>system>security>keychain>direction>bi>entry>begin-time config>system>security>keychain>direction>bi>entry>tolerance config>system>security>keychain>direction>uni>receive>entry >begin-time config>system>security>keychain>direction>uni>receive>entry >tolerance
End time after which key[i] cannot be used	config>system>security>keychain>direction>uni>receive>entry>end-time

Table 19 lists the authentication algorithms that can be used in association with specific routing protocols.

Table 19 **Security Algorithm Support Per Protocol**

Protocol	Clear Text	MD5	HMAC-MD5	HMAC-SHA-1-96	HMAC-SHA-1	HMAC-SHA-256	AES-128-CMAC-96
OSPF	Yes	Yes	—	Yes	Yes	Yes	—

Table 19 **Security Algorithm Support Per Protocol (Continued)**

Protocol	Clear Text	MD5	HMAC-MD5	HMAC-SHA-1-96	HMAC-SHA-1	HMAC-SHA-256	AES-128-CMAC-96
IS-IS	Yes	—	Yes	—	Yes	Yes	—
RSVP	Yes	—	Yes	—	Yes	—	—
BGP	—	Yes	—	Yes	—	—	Yes
LDP	—	Yes	—	Yes	—	—	Yes

2.5.7 gRPC Authentication

gRPC communication between the client and server must be authenticated and encrypted. There are two types of authentication:

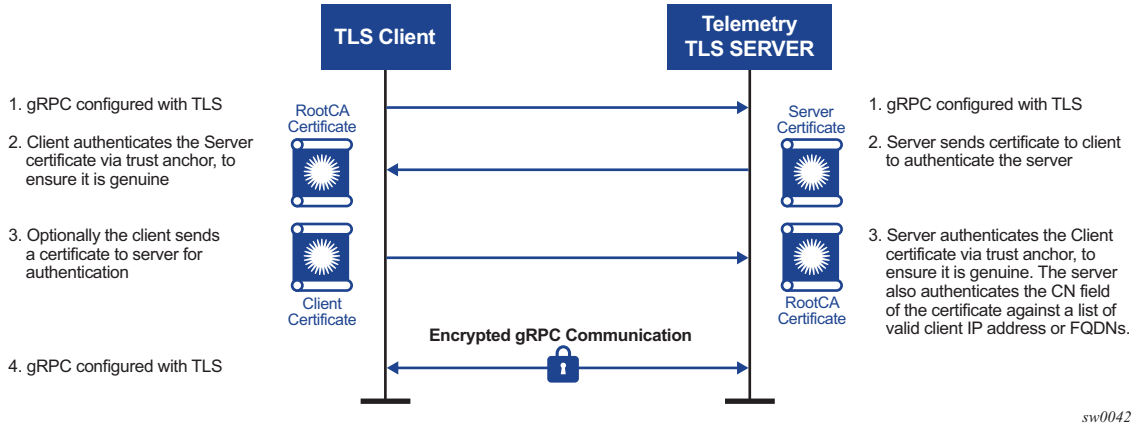
- Authentication via session credentials — Session credentials operate similarly to device authentication, ensuring that the device is allowed in the network and is authorized by the provider. This type of authentication is performed using PKI and X.509.3 certificates. gRPC uses TLS for session authentication.

  SR OS supports TLS servers for gRPC.
- Authentication using channel credentials — Channel credentials use a user name and password that are entered at the gRPC client terminal to authenticate gRPC packets using an AAA method.

Session authentication provides proof that the client and server are authorized devices and that they belong to the provider. After authentication, the session becomes encrypted using TLS, and gRPC PDUs are transmitted between the client and server.

Figure 10 shows a basic session authentication using TLS.

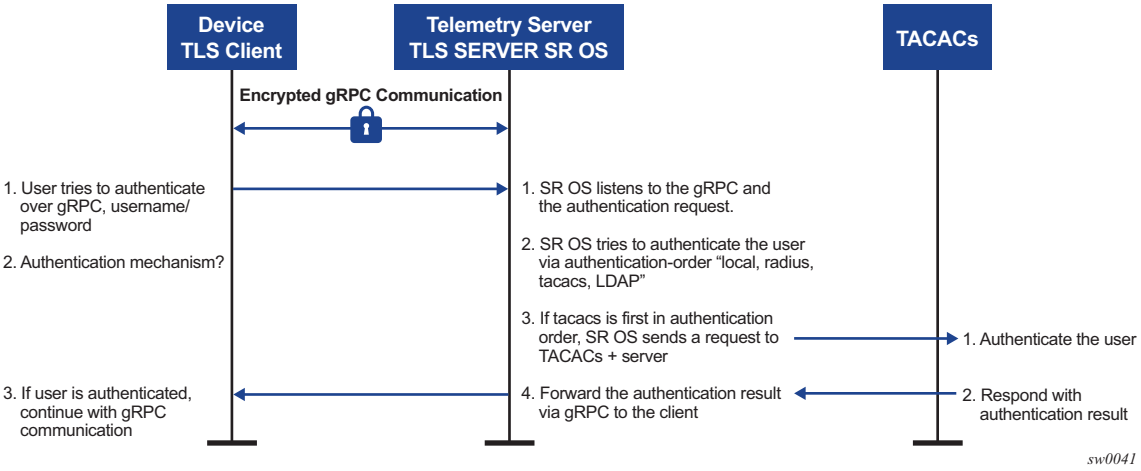
Figure 10 **Session Authentication Using TLS**



Channel credentials use username and password authentication. Each gRPC channel packet can contain a username and a password. Authentication is done through standard SR OS authentication order and mechanisms. All current authentication methods, including local and AAA servers, are applicable to gRPC channels. In addition, all authentication orders currently used by Telnet or SSH are compatible with gNMI Call authentication.

Figure 11 shows a basic gNMI Call authentication using SR OS.

Figure 11 **gNMI Call Authentication Using SR OS**



The gRPC channel packets contain the username and password in clear text, and are only encrypted using TLS. If a TLS server profile is assigned to the gRPC session, all PDUs between the server and client are encrypted. If TLS becomes operationally down, no gRPC PDUs are transmitted in clear text.

SR OS relies on existing authentication mechanisms for gRPC channels, including:

- AAA servers and local authentication orders configured using the **config**>**system**>**security**>**password**>**authentication-order** command
- password complexity rules
- requiring the user to be configured as part of gRPC access by using the **config**>**system**>**security**>**user**>**access**>**grpc** command
- disconnecting the gRPC session by using the **admin**>**disconnect gNMI** command

Note: gRPC is not affected by password aging.

Security profiles can authorize bulk **get**, **set**, and **subscribe gRPC** commands that are received by the server. Profiles can be configured to permit or deny specific gRPC commands; for example, a profile for one user can authorize **get** and **set** commands, while a profile for another user can authorize **get** commands only.

2.5.8 Hash Management per Management Interface Configuration

Hash management is configurable per management interface, for example, the classic CLI, the MD-CLI, NETCONF, or gRPC. Each management interface will have its own write-hash algorithm. Depending on which management interface the user logs into, the write hash of that interface should be checked and used for displaying the critical phrases.

In the classic CLI interface, the read and write hash algorithms can be different, for example, hash for write and hash2 for read.

For the MD-CLI, NETCONF, and gRPC interfaces, when a hash is configured, only write will be implemented using that hash algorithm. For example, if hash2 is configured, SR OS will display the phrase in hash2 format and read the phrase in all formats. The read algorithm is not affected by hash algorithm configuration and SR OS reads in all hash formats.

2.5.8.1 Hash encryption Using AES 256

Hash and hash2 use the AES 256 algorithm for all interfaces. However, hash2 uses module-specific text to make the hash unique per module or protocol. For example, BGP will use a different pre-pending text than IGP. This pre-pending text is appended to the key and then hashed using hash2.

Classic CLI hash has been changed to AES-256.

Upgrade from DES to AES-256 is allowed and loading a config file in classic CLI with DES to a new software that supports AES-256 is also allowed.

The DES and the DES key should only be used for decryption of the old password to obtain clear text and the password should then be rehashed using AES-256. The few characters of the old hashed phrase are used to determine that the phrase is hashed using DES.

2.5.8.2 Clear Text

The cleartext option for the write algorithm displays the hash in clear text in the config file, info, info detail, and so on.

2.6 Configuring Security with CLI

This section provides information to configure security using the command line interface.

2.6.1 Security Configurations

This section provides configuration examples for the following security capabilities:

- User profiles
- User access parameters
- Password management parameters
- Authentication, authorization and accounting using local, RADIUS, TACACS+, and/or LDAP
- Filtering using CPM filters and management access filters

Table 20 list the capabilities of authentication, authorization, and accounting configurations. For example, authentication can be enabled locally and on RADIUS, TACACS+, and LDAP servers. Authorization can be executed locally, on a RADIUS server, or on a TACACS+ server. Accounting can be performed on a RADIUS or TACACS+ server.

Table 20 **Security Configuration Requirements**

Authentication	Authorization	Accounting
Local	Local	None
RADIUS	Local and RADIUS	RADIUS
TACACS+	Local	TACACS+
LDAP	None	None

2.6.2 Configuring Management Access Filters

The following is an example of a management access filter configuration that accepts packets matching the criteria specified in IP, IPv6 and MAC entries. Non-matching packets are denied.

```
*A:Dut-C>config>system>security>mgmt-access-filter# info
```

```
-----------------------------------------------
                ip-filter
                    default-action deny
                    entry 10
                        description "Accept SSH from mgmnt subnet"
                        src-ip 192.168.5.0/26
                        protocol tcp
                        dst-port 22 65535
                        action permit
                    exit
                exit
                ipv6-filter
                    default-action permit
                    entry 10
                        src-ip 2001:db8:1000::/64
                        next-header rsvp
                        log
                        action deny
                    exit
                exit
                mac-filter
                    default-action permit
                    entry 12
                        match frame-type ethernet_II
                            svc-id 1
                            src-mac 00:01:01:01:01:01 ff:ff:ff:ff:ff:ff
                        exit
                        action permit
                    exit
                exit
-----------------------------------------------
*A:Dut-C>config>system>security>mgmt-access-filter#
```

2.6.3 Configuring IP CPM Filters

Nokia recommends using a strict CPM filter policy allowing traffic from trusted IP subnets for protocols and ports actively used in the router and to explicitly drop other traffic.

The configuration below is an example that follows the recommendations for SSH and BGP:

- Allow SSH from trusted subnet only
- Allow BGP from trusted subnet only
- Explicitly deny all other traffic and operationally log unexpected packets

```
A:Dut-A>config>sys>security>cpm-filter# info
-----------------------------------------------
             default-action drop
             ip-filter
                 entry 100 create
                     action accept
```

```
                        description "SSH: server terminated TCP sessions from trusted
subnets"
                        match protocol tcp
                            dst-port 22 65535
                            src-ip ip-prefix-list "trusted-mgmt-subnet"
                        exit
                    exit
                    entry 200 create
                        action accept
                        description "BGP: server terminated TCP Sessions"
                        match protocol tcp
                            dst-port 179 65535
                            src-ip ip-prefix-list "trusted-bgp-subnet"
                        exit
                    exit
                    entry 300 create
                        action accept
                        description "BGP: client responses for initiated TCP sessions"
                        match protocol tcp
                            src-ip ip-prefix-list "trusted-bgp-subnet"
                            src-port 179 65535
                        exit
                    exit
                    entry 6000 create
                        action drop
                        description "Drop all other UDP"
                        log 102
                        match protocol udp
                        exit
                    exit
                    entry 6010 create
                        action drop
                        description "drop all other TCP"
                        log 103
                        match protocol tcp
                        exit
                    exit
                    no shutdown
                exit
-------------------------------------------
```

2.6.4 Configuring IPv6 CPM Filters

Nokia recommends using a strict CPM filter policy allowing traffic from trusted IP subnets for protocols and ports actively used in the router and to explicitly drop other traffic.

The configuration below is an example that follows the recommendations for SSH and BGP:

- Allow SSH from trusted subnet only
- Allow BGP from trusted subnet only

- Explicitly deny all other traffic and operationally log unexpected packets

```
A:Dut-A>config>sys>security>cpm-filter# info
----------------------------------------------
              default-action drop
              ip-filter
                  entry 100 create
                      action accept
                      description "SSH: server terminated TCP sessions from trusted
subnets"
                      match protocol tcp
                          dst-port 22 65535
                          src-ip ip-prefix-list "trusted-mgmt-subnet"
                      exit
                  exit
                  entry 200 create
                      action accept
                      description "BGP: server terminated TCP Sessions"
                      match protocol tcp
                          dst-port 179 65535
                          src-ip ip-prefix-list "trusted-bgp-subnet"
                      exit
                  exit
                  entry 300 create
                      action accept
                      description "BGP: client responses for initiated TCP sessions"
                      match protocol tcp
                          src-ip ip-prefix-list "trusted-bgp-subnet"
                          src-port 179 65535
                      exit
                  exit
                  entry 6000 create
                      action drop
                      description "Drop all other UDP"
                      log 102
                      match protocol udp
                      exit
                  exit
                  entry 6010 create
                      action drop
                      description "drop all other TCP"
                      log 103
                      match protocol tcp
                      exit
                  exit
                  no shutdown
              exit
              ipv6-filter
                  entry 100 create
                      action accept
                      description "SSH: server terminated TCP sessions from trusted
subnets"
                      match next-header tcp
                          dst-port 22 65535
                          src-ip ipv6-prefix-list "trusted-mgmt-subnet"
                      exit
                  exit
                  entry 200 create
                      action accept
```

```
                        description "BGP: server terminated TCP Sessions"
                        match next-header tcp
                            dst-port 179 65535
                            src-ip ipv6-prefix-list "trusted-bgp-subnet"
                        exit
                    exit
                    entry 300 create
                        action accept
                        description "BGP: client responses for initiated TCP sessions"
                        match next-header tcp
                            src-ip ipv6-prefix-list "trusted-bgp-subnet"
                            src-port 179 65535
                        exit
                    exit
                    entry 6000 create
                        action drop
                        description "Drop all other UDP"
                        log 102
                        match next-header udp
                        exit
                    exit
                    entry 6010 create
                        action drop
                        description "drop all other TCP"
                        log 103
                        match next-header tcp
                        exit
                    exit
                    no shutdown
                exit
-------------------------------------------------
```

2.6.5 Configuring MAC CPM Filters

The following displays a MAC CPM filter configuration example:

```
*A:ALA-49>config>sys>sec>cpm>mac-filter# info
-------------------------------------------------
                    entry 10 create
                        description "MAC-CPM-Filter 10.10.10.100 #007"
                        match
                        exit
                        log 101
                        action drop
                    exit
                    entry 20 create
                        description "MAC-CPM-Filter 10.10.10.100 #008"
                        match
                        exit
                        log 101
                        action drop
                    exit
                    no shutdown
-------------------------------------------------
*A:ALA-49>config>sys>sec>cpm>mac-filter#
```

2.6.6 Configuring CPM Queues

CPM queues can be used for troubleshooting purposes to provide rate limit capabilities for traffic destined to CPM as described in an earlier section of this document.

The following example displays a CPM queue configuration:

```
A:ALA-987>config>sys>security>cpm-queue# info
----------------------------------------------
                queue 101 create
                    rate 100
                exit
----------------------------------------------
A:ALA-987>config>sys>security>cpm-queue#
```

2.6.7 IPsec Certificates Parameters

The following is an example to importing a certificate from a PEM format:

```
*A:SR-7/Dut-A# admin certificate import type cert input cf3:/pre-import/R1-0cert.pem
 output R1-0cert.der format pem
```

The following is an example for exporting a certificate to PEM format:

```
*A:SR-7/Dut-A#  admin certificate export type cert input R1-0cert.der output cf3:/
R1-0cert.pem format pem
```

The following displays an example of profile output:

```
*A:SR-7/Dut-A>config>system>security>pki# info
----------------------------------------------
                ca-profile "Root" create
                    description "Root CA"
                    cert-file "R1-0cert.der"
                    crl-file "R1-0crl.der"
                    no shutdown
                exit
----------------------------------------------
*A:SR-7/Dut-A>config>system>security>pki#
```

The following displays an example of an ike-policy with cert-auth output:

```
*A:SR-7/Dut-A>config>ipsec>ike-policy# info
----------------------------------------------
            ike-version 2
```

```
            auth-method cert-auth
            own-auth-method psk
-----------------------------------------------
```

The following displays an example of a static LAN-to-LAN configuration using cert-auth:

```
...
    interface "VPRN1" tunnel create
        sap tunnel-1.private:1 create
            ipsec-tunnel "Sanity-1" create
                security-policy 1
                local-gateway-address 10.1.1.13 peer 10.1.1.15 delivery-service 300
                dynamic-keying
                    ike-policy 1
                    pre-shared-key "Sanity-1"
                    transform 1
                    cert
                        trust-anchor "R1-0"
                        cert "M2cert.der"
                        key "M2key.der"
                    exit
                exit
                no shutdown
            exit
        exit
    exit
```

2.6.8 Configuring Local Command Authorization Profiles

Profiles are used to deny or permit access to a hierarchical branch or specific commands.

The following example displays a local command authorization profile called “ghost” that is associated with a user named “userA”:

```
A:ALA-1>config>system>security# info
-----------------------------------------------
...
            profile "ghost"
                default-action permit-all
                entry 1
                    match "configure"
                    action permit
                exit
                entry 2
                    match "configure service vprn <22>"
                    action read-only
                exit
                entry 3
                    match "show"
                exit
```

```
                entry 4
                    match "exit"
                exit
            exit
...
-------------------------------------------
A:ALA-1>config>system>security#
A:ALA-1>config>system>security# info
-------------------------------------------
...
    user "userA"
        ...
        console
            member "ghost"
        exit
...
```

2.6.8.1 Parameters

Matching in authorization profiles allows the use of parameters and optional parameters. A set of angle brackets <...> indicates matching on a parameter and/or optional parameter.

The following rules govern parameter matching in the CLI:

Rule 1

Any parameter and/or optional parameter can be present in the match string.

Rule 2

When a parameter and an optional parameter is present in the user-profile match string, all parameters or optional parameters to its left must also be stated/present.

Rule 3

The user can either specifically state or completely omit unnamed parameters in the match string, as required. However, all unnamed parameter in the CLI command must be present in the match string when matching on an unnamed parameter is used.

For example, consider the **OSPF** command:

```
*A:SwSim14# configure router ospf
  - no ospf [<ospf-instance>]
  - ospf [<ospf-instance>] [<router-id>]

<ospf-instance>      : [0..31]
<router-id>          : <ip-address>
```

In this case, the user can match on OSPF to allow or deny the command per user-profile, as follows:

```
Match "configure router ospf" action deny
```

Or the user can decide to only allow a certain OSPF instance for a user, as follows:

```
Match "configure router ospf <ospf-instance-value> <router-id-value>"
```

Note: Although the user's matching is based on <ospf-instance-value> that is "an unnamed value", all other unnamed values in the **OSPF** command (such as the <router-id-value>) must also be present in the match string.

Rule 4

When multiple unnamed parameters are present in the match string, the parameters must be provided in the correct order as described in the command **help** to generate the correct match behavior. For example, using the order of parameters described in the **OSPF** command usage in Rule 3 above, use the following statement for a user-profile match:

```
match "configure router ospf <ospf-instance-value> <router-id-value>
```

The desired match behavior might not be achieved if the unnamed parameters <ospf-instance-value> and <router-id-value> are out of order with respect to the help screen.

The following displays a parameter matching output:

```
config>system>security>profile# info
                entry 10
                    match "show router <22> route-table "
                    action permit
                exit
                entry 20
                    match "configure service vprn <22>"
                    action read-only
                exit
                entry 30
                    match "show service id <22>"
                    action permit
                exit
                entry 40
                    match "configure router interface <system>"
                    action deny
                exit
```

2.6.8.2 Wildcards

In addition, parameter configuration is facilitated by the availability of wildcards (.*) in the OAM subtree and for commands such as **ping**, **trace route** and **mtrace**. For example, consider the following command:

```
ping <ip-address> router 10
```

Instead of listing all the permitted IP addresses in the policy, as shown in the following example,

```
Match ping <10.0.0.1> router <10>
Action permit
Match ping <10.0.0.2> router <10>
Action permit
```

The wildcard<ip-address> parameter allows a simpler search criterion. In the following example, the use of <.*> wildcard enables the ability to ping any address in the router 10 context, that is, any address in VRF 10:

```
Match ping <.*> router <10>
Action permit
```

Note: While wildcards are available and allowed for all parameters in the OAM subtree, Nokia recommends that caution is exercised when using wildcards and limit their use to commands such as **ping**, **trace route**, and **mtrace**. The use of wildcards in certain formats may be a security concern and result in making the IP addresses in the VRF, including the base routing table, unreachable. Or it could allow the customer to ping any IP address in the VRF, including the base routing table. This may be a potential security concern and should be avoided.

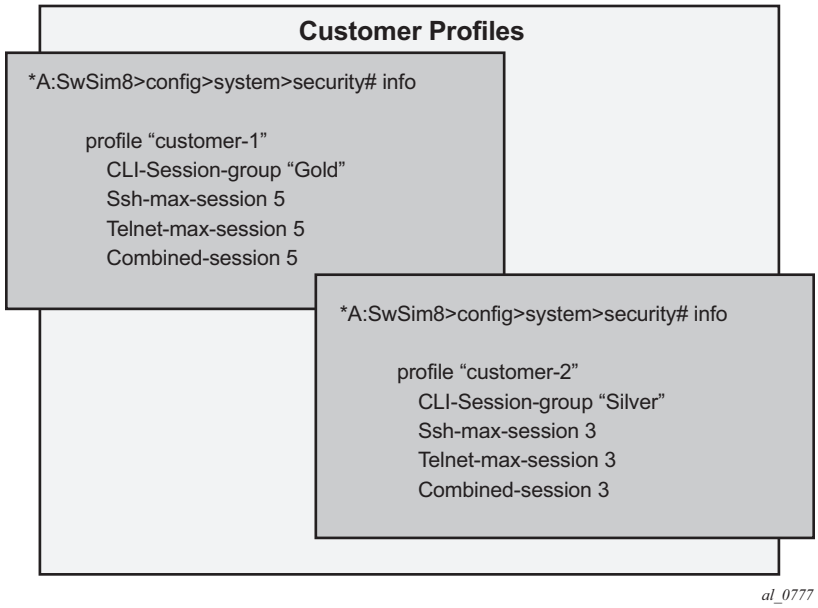
For example, the following usage is not advised:

```
Match ping <.*> router <.*>
Action permit
```

2.6.8.3 CLI Session Resource Management

SR OS has the capability to manage telnet/ssh sessions per user and at a higher level per system. At the system level, the user can configure a **cli-session-group** for different customer priorities. The **cli-session-group** is a container that sets the maximum number of CLI sessions for a class of customers, with a unique session limit for each customer. For example, as depicted in Figure 12, “Gold” category customers can have a **cli-session-group** that allows them more telnet/ssh sessions compared to “Silver” category customers.

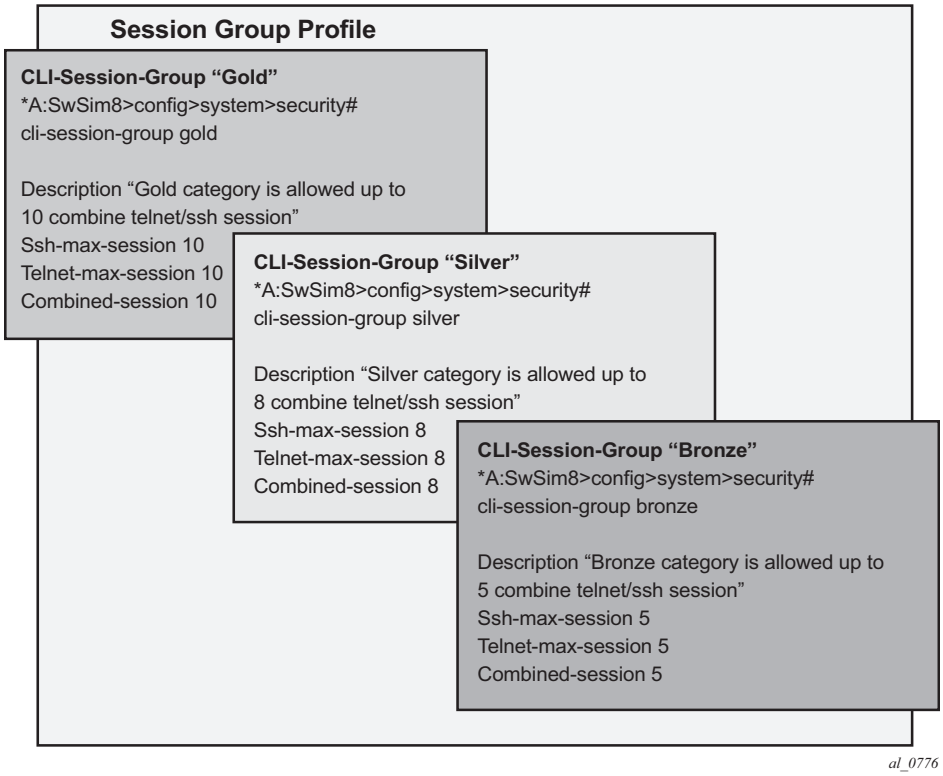
Figure 12 **cli-session-group for Customer Classes**



The configured **cli-session-group** can be assigned to user-profiles. At the user profile level, each profile can be configured with its own max ssh/telnet session and it will be policed/restricted by the higher order **cli-session-group** that is assigned to it.

As depicted in Figure 13, the final picture is a hierarchical configuration with top-level cli-session-groups that control each customer’s total number of SSH or telnet sessions and the user-profile for each user for that customer.

Figure 13 **Hierarchy of cli-session-group Profiles**



Every profile will subtract one from it's corresponding **max-session** when a TELNET or SSH session is established in the following cases:

- where multiple profiles are configured under a user
- where multiple profiles arrive from different AAA servers (Local Profile, RADIUS Profile or TACACS Profile)

The first profile to run out of corresponding **max-session** will limit future TELNET or SSH sessions. In other words, while each profile for the user can have its independent **max-session**, only the lowest one will be honored. If the profile with the lowest **max-session** is removed, the next lower profile **max-session** will be honored and so on. All profiles for a user are updated when a TELNET or SSH session is established.

For information about login control, see Configuring Login Controls.

Use the following CLI commands to configure CLI session resources:

CLI Syntax:
```
config>system>security>profile <name>
     [no] ssh-max-sessions session-limit
```

```
[no] telnet-max-sessions session-limit
[no] combined-max-session session-limit
[no] cli-session-group session-group-name
```

2.6.9 Configuring Users

Configure access parameters for individual users. For user, define the login name for the user and, optionally, information that identifies the user.

The following displays a user configuration example:

```
A:ALA-1>config>system>security# info
----------------------------------------------
...
            user "userA"
                password "$2y$10$pFoehOg/tCbBMPDJ/
kqpu.8af0AoVGY2xsR7WFqyn5fVTnwRzGmOK"
                access console ftp snmp
                restricted-to-home
                console
                    member "default"
                    member "ghost"
                exit
            exit
...
----------------------------------------------
A:ALA-1>config>system>security#
```

2.6.10 Configuring Keychains

The following displays a keychain configuration.

```
A:ALA-1>config>system>security# info
----------------------------------------------
...
            keychain "abc"
                direction
                    bi
                        entry 1 key "ZcvSElJzJx/wBZ9biCtOVQJ9YZQvVU.S" hash2 alg
orithm aes-128-cmac-96
                            begin-time 2006/12/18 22:55:20
                        exit
                    exit
                exit
            exit
            keychain "basasd"
                direction
                    uni
                        receive
```

```
                            entry 1 key "Ee7xdKlYO2DOm7v3IJv/84LIu96R2fZh" hash2
 algorithm aes-128-cmac-96
                                tolerance forever
                            exit
                        exit
                    exit
                exit
            exit
...
-------------------------------------------------
A:ALA-1>config>system>security#
```

2.6.11 Copying and Overwriting Users and Profiles

You can copy a profile or user. You can copy a profile or user or overwrite an existing profile or user. The **overwrite** option must be specified or an error occurs if the destination profile or user name already exists.

2.6.11.1 User

CLI Syntax: config>system>security# copy {user *source-user* | profile *source-profile*} to *destination* [overwrite]

Example:
```
config>system>security# copy user testuser to testuserA
MINOR: CLI User "testuserA" already exists - use
 overwrite flag.
config>system>security#
config>system>security# copy user testuser to testuserA
 overwrite
config>system>security#
```

The following output displays the copied user configurations:

```
A:ALA-12>config>system>security# info
-------------------------------------------------
...
            user "testuser"
                password "$2y$10$pFoehOg/tCbBMPDJ/
kqpu.8af0AoVGY2xsR7WFqyn5fVTnwRzGmOK"
                access snmp
                snmp
                    authentication hash md5 e14672e71d3e96e7a1e19472527ee969 privacy
 none
                    group "testgroup"
                exit
            exit
            user "testuserA"
                password ""
```

```
                access snmp
                console
                    new-password-at-login
                exit
                snmp
                    authentication hash md5 e14672e71d3e96e7a1e19472527ee969 privacy
 none
                    group "testgroup"
                exit
            exit
...
-----------------------------------------------
A:ALA-12>config>system>security# info
```

Note: The cannot-change-password flag is not replicated when a **copy user** command is performed. A new-password-at-login flag is created instead.

```
A:ALA-12>config>system>security>user# info
-----------------------------------------------
password "$2y$10$pFoehOg/tCbBMPDJ/kqpu.8af0AoVGY2xsR7WFqyn5fVTnwRzGmOK"
access snmp
console
cannot-change-password
exit
snmp
authentication hash md5 e14672e71d3e96e7a1e19472527ee969 privacy none
group "testgroup"
exit
-----------------------------------------------
A:ALA-12>config>system>security>user# exit
A:ALA-12>config>system>security# user testuserA
A:ALA-12>config>system>security>user# info
-----------------------------------------------
password ""
access snmp
console
new-password-at-login
exit
snmp
authentication hash md5 e14672e71d3e96e7a1e19472527ee969 privacy none
group "testgroup"
exit
-----------------------------------------------
A:ALA-12>config>system>security>user#
```

2.6.11.2 Profile

CLI Syntax: `config>system>security# copy {user` *`source-user`* `| profile` *`source-profile`*`} to` *`destination`* `[overwrite]`

Example: `config>system>security# copy profile default to testuser`

The following output displays the copied profiles:

```
A:ALA-49>config>system>security# info
----------------------------------------------
...
A:ALA-49>config>system>security# info detail
----------------------------------------------
...
            profile "default"
                default-action none
                entry 10
                    no description
                    match "exec"
                    action permit
                exit
                entry 20
                    no description
                    match "exit"
                    action permit
                exit
                entry 30
                    no description
                    match "help"
                    action permit
                exit
                entry 40
                    no description
                    match "logout"
                    action permit
                exit
                entry 50
                    no description
                    match "password"
                    action permit
                exit
                entry 60
                    no description
                    match "show config"
                    action deny
                exit
                entry 70
                    no description
                    match "show"
                    action permit
                exit
                entry 80
                    no description
                    match "enable-admin"
                    action permit
                exit
            exit
            profile "testuser"
                default-action none
                entry 10
                    no description
                    match "exec"
                    action permit
                exit
                entry 20
```

```
                        no description
                        match "exit"
                        action permit
                    exit
                    entry 30
                        no description
                        match "help"
                        action permit
                    exit
                    entry 40
                        no description
                        match "logout"
                        action permit
                    exit
                    entry 50
                        no description
                        match "password"
                        action permit
                    exit
                    entry 60
                        no description
                        match "show config"
                        action deny
                    exit
                    entry 70
                        no description
                        match "show"
                        action permit
                    exit
                    entry 80
                        no description
                        match "enable-admin"
                        action permit
                    exit
                exit
                profile "administrative"
                    default-action permit-all exit
...
-----------------------------------------------
A:ALA-12>config>system>security#
```

2.6.12 RADIUS Configurations

2.6.12.1 Configuring RADIUS Authentication

RADIUS is disabled by default and must be explicitly enabled. The mandatory commands to enable RADIUS on the local router are **radius** and server *server-index* address *ip-address* secret *key*.

Also, the system IP address must be configured in order for the RADIUS client to work. See “Configuring a System Interface” of the *7450 ESS, 7750 SR, 7950 XRS, and VSR Router Configuration Guide*.

The other commands are optional. The server command adds a RADIUS server and configures the RADIUS server’s IP address, index, and key values. The index determines the sequence in which the servers are queried for authentication requests.

On the local router, use the following CLI commands to configure RADIUS authentication:

CLI Syntax:
```
config>system>security
radius
   port port
   retry count
   server server-index address ip-address secret key
   timeout seconds
   no shutdown
```

The following displays a RADIUS authentication configuration example:

```
A:ALA-1>config>system>security# info
----------------------------------------------
                retry 5
                timeout 5
                server 1 address 10.10.10.103 secret "test1"
                server 2 address 10.10.0.1 secret "test2"
                server 3 address 10.10.0.2 secret "test3"
                server 4 address 10.10.0.3 secret "test4"
...
----------------------------------------
A:ALA-1>config>system>security#
```

2.6.12.2 Configuring RADIUS Authorization

In order for RADIUS authorization to function, RADIUS authentication must be enabled first. See Configuring RADIUS Authentication.

In addition to the local configuration requirements, VSAs must be configured on the RADIUS server. See Vendor-Specific Attributes (VSAs).

On the local router, use the following CLI commands to configure RADIUS authorization:

CLI Syntax:
```
config>system>security
radius
```

```
                authorization
```

The following displays a RADIUS authorization configuration example:

```
A:ALA-1>config>system>security# info
----------------------------------------------
...
            radius
                authorization
                retry 5
                timeout 5
                server 1 address 10.10.10.103 secret "test1"
                server 2 address 10.10.0.1 secret "test2"
                server 3 address 10.10.0.2 secret "test3"
                server 4 address 10.10.0.3 secret "test4"
            exit
...
----------------------------------------------
A:ALA-1>config>system>security#
```

2.6.12.3 Configuring RADIUS Accounting

On the local router, use the following CLI commands to configure RADIUS accounting:

CLI Syntax:

```
config>system>security
radius
   accounting
```

The following displays RADIUS accounting configuration example:

```
A:ALA-1>config>system>security# info
----------------------------------------------
...
           radius
               shutdown
               authorization
               accounting
               retry 5
               timeout 5
               server 1 address 10.10.10.103 secret "test1"
               server 2 address 10.10.0.1 secret "test2"
               server 3 address 10.10.0.2 secret "test3"
               server 4 address 10.10.0.3 secret "test4"
           exit
...
----------------------------------------------
A:ALA-1>config>system>security#
```

2.6.13 Configuring 802.1x RADIUS Policies

Use the following CLI commands to configure generic authentication parameters for clients using 802.1x EAPOL. Additional parameters are configured per Ethernet port. Refer to the *7450 ESS, 7750 SR, 7950 XRS, and VSR Interface Configuration Guide*.

To configure generic parameters for 802.1x authentication, enter the following CLI syntax.

CLI Syntax:
```
config>system>security
dot1x
   radius-plcy policy-name
      server server-index address ip-address secret key
         [port port]
      source-address ip-address
      no shutdown
```

The following displays a 802.1x configuration example:

```
A:ALA-1>config>system>security# info
----------------------------------------------
            dot1x
                radius-plcy "dot1x_plcy" create
                   server 1 address 10.1.1.1 port 65535 secret "a"
                   server 2 address 10.1.1.2 port 6555 secret "a"
                   source-address 10.1.1.255
                no shutdown
...
----------------------------------------------
A:ALA-1>config>system#
```

2.6.14 TACACS+ Configurations

2.6.14.1 Enabling TACACS+ Authentication

To use TACACS+ authentication on the router, configure one or more TACACS+ servers on the network.

Use the following CLI commands to configure profiles:

CLI Syntax:
```
config>system>security
   tacplus
      server server-index address ip-address secret key
```

```
timeout seconds
no shutdown
```

The following displays a TACACS+ authentication configuration example:

```
A:ALA-1>config>system>security>tacplus# info
----------------------------------------------
                timeout 5
                server 1 address 10.10.0.5 secret "test1"
                server 2 address 10.10.0.6 secret "test2"
                server 3 address 10.10.0.7 secret "test3"
                server 4 address 10.10.0.8 secret "test4"
                server 5 address 10.10.0.9 secret "test5"
----------------------------------------------
A:ALA-1>config>system>security>tacplus#
```

2.6.14.2 Configuring TACACS+ Authorization

In order for TACACS+ authorization to function, TACACS+ authentication *must* be enabled first. See Enabling TACACS+ Authentication.

On the local router, use the following CLI commands to configure RADIUS authorization:

CLI Syntax:

```
config>system>security
tacplus
  authorization
  no shutdown
```

The following displays a TACACS+ authorization configuration example:

```
A:ALA-1>config>system>security>tacplus# info
----------------------------------------------
                authorization
                timeout 5
                server 1 address 10.10.0.5 secret "test1"
                server 2 address 10.10.0.6 secret "test2"
                server 3 address 10.10.0.7 secret "test3"
                server 4 address 10.10.0.8 secret "test4"
                server 5 address 10.10.0.9 secret "test5"
----------------------------------------------
A:ALA-1>config>system>security>tacplus#
```

2.6.14.3 Configuring TACACS+ Accounting

On the local router, use the following CLI commands to configure TACACS+ accounting:

CLI Syntax:
```
config>system>security
tacplus
   accounting
```

The following displays a TACACS+ accounting configuration example:

```
A:ALA-1>config>system>security>tacplus# info
----------------------------------------------
               accounting
               authorization
               timeout 5
               server 1 address 10.10.0.5 secret "test1"
               server 2 address 10.10.0.6 secret "test2"
               server 3 address 10.10.0.7 secret "test3"
               server 4 address 10.10.0.8 secret "test4"
               server 5 address 10.10.0.9 secret "test5"
----------------------------------------------
A:ALA-1>config>system>security>tacplus#
```

2.6.14.4 Enabling SSH

Use the SSH command to configure the SSH server as SSH1, SSH2 or both. The default is SSH2 (SSH version 2). This command should only be enabled or disabled when the SSH server is disabled. This setting should not be changed while the SSH server is running since the actual change only takes place after SSH is disabled or enabled.

CLI Syntax:
```
config>system>security
ssh
   preserve-key
   no server-shutdown
   version ssh-version
```

The following displays a SSH server configuration as both SSH and SSH2 using a host-key:

```
A:sim1>config>system>security>ssh# info
----------------------------------------------
               preserve-key
               version 1-2
----------------------------------------------
A:sim1>config>system>security>ssh#
```

2.6.15 LDAP Configurations

2.6.15.1 Configuring LDAP Authentication

LDAP is disabled by default and must be explicitly enabled. To use LDAP authentication on the router, configure one or more LDAP servers on the network.

TLS certificates and clients must also be configured. Refer to the “TLS” section of the *7450 ESS, 7750 SR, 7950 XRS, and VSR System Management Guide* for more information about configuring TLS.

Use the following CLI commands to configure LDAP:

CLI Syntax:

```
config>system>security>ldap
   [no] public-key-authentication
   [no] retry
   [no] server
   [no] shutdown
   [no] timeout
   [no] use-default-template

config>system>security>password
   authentication-order [method] exit-on-reject

config>system>security>ldap
   public-key-authentication
   server server-index create
      address ip-address port port
      bind-authentication root-dn [password password]
         [hash | hash2 | custom]
      ldap-server server-name
      search base-dn
      tls-profile tls-profile-name
      no shutdown
   exit
   no shutdown
```

The following displays an LDAP authentication configuration example:

```
A:SwSim14>config>system>security>ldap#
-------------------------------------------
    [no] public-key-authentication
    [no] retry
    [no] server
    [no] shutdown
    [no] timeout
    [no] use-default-template
```

```
-----------------------------------------------
*A:SwSim14>config>system>security>password#
-----------------------------------------------
    authentication-order [local | radius | tacplus | ldap] exit-on-reject
-----------------------------------------------
*A:SwSim14>config>system>security>ldap# info
-----------------------------------------------
    public-key-authentication
    server 1 create
        address 10.1.1.1
        bind-authentication "cn=administrator,cn=users,dc=nacblr2,dc=example,dc=com
          password"
        ldap-server "active-server"
        search "dc=sns,dc=example,dc=com"
        tls-profile "server-1-profile"
        no shutdown
    exit
    no shutdown
-----------------------------------------------
*A:SwSim8>config>system>security>tls# info
-----------------------------------------------
    client-tls-profile "server-1-profile" create
        cipher-list "to-active-server"
        trust-anchor-profile "server-1-ca"
    no shutdown
    exit
```

2.6.15.2 Configuring Redundant Servers

Up to five redundant LDAP servers can be configured. The following examples show configuration of two servers, Server-1 and Server-5.

Configuration of Server-1:

```
A*:SwSim14>config>system>security>ldap# info
    public-key-authentication
    server 1 create
        address 10.1.1.1
        ldap-server "active-server"
        tls-profile "server-1-profile"

A*:SwSim14>config>system>security>tls# info
    client-tls-profile "server-1-profile" create
        cert-profile "client-cert-profile"
        cipher-list "to-active-server"
        trust-anchor-profile "server-1-ca"
        no shutdown
    exit
```

Configuration of Server-5 (backup):

```
A*:SwSim14>config>system>security>ldap# info
    public-key-authentication
    server 5 create
```

```
        address 10.5.5.1
        ldap-server "backup-server-5"
        tls-profile "server-5-profile"

A*:SwSim14>config>system>security>tls# info
    client-tls-profile "server-5-profile" create
        cert-profile "client-cert-profile"
        cipher-list "to-backup-server-5"
        trust-anchor-profile "server-5-ca"
        no shutdown
    exit
```

2.6.15.3 Enabling SSH

SSH must be enabled to use LDAP authentication. See Enabling SSH for more information.

2.6.16 Configuring Login Controls

Configure login control parameters for console, Telnet, and FTP sessions.

The following displays a login control configuration example:

```
A:ALA-1>config>system# info
----------------------------------------------
...
       login-control
           ftp
               inbound-max-sessions 5
           exit
           telnet
               inbound-max-sessions 7
               outbound-max-sessions 2
           exit
           idle-timeout 1440
           pre-login-message "Property of Service Routing Inc. Unauthorized access
                              prohibited."
           motd text "Notice to all users: Software upgrade scheduled 3/2 1:00 AM"
       exit
 no exponential-backoff
...
----------------------------------------------
A:ALA-1>config>system#
```

3 Classic and Model-Driven Management Interfaces

SR OS supports two basic classes of management interfaces:

- classic management interfaces
- model-driven management interfaces

Classic management interfaces include:

- SNMP
- the classic CLI

Model-driven management interfaces include:

- the MD-CLI
- NETCONF using the Nokia SR OS YANG modules
- the gRPC Network Management Interface (gNMI)

References to the term CLI in the SR OS user documentation are generally referring to the classic CLI. The classic CLI is the CLI that has been supported in SR OS from the initial introduction of SR OS.

The MD-CLI is a model-driven CLI introduced in SR OS Release 16.0.R1. Refer to *The MD-CLI User Guide* and *The MD-CLI Command Reference Guide* for details on using configuration commands in the MD-CLI.

3.1 Model-Driven Management Interfaces

Model-driven management interfaces are based on a common infrastructure that uses YANG models as the core definition for configuration, state, and operational actions. All model-driven interfaces take the same common underlying YANG modules and render them for the particular management interface.

The Nokia SR OS YANG modules of the model-driven infrastructure are similar to the classic CLI tree with the following notable differences:

- the classic and model-driven configuration formats are incompatible; the system automatically converts the classic configuration to the model-driven format when the management interface configuration mode is changed to **model-driven**

- some classic CLI branches have been moved, renamed, or re-organized in the SR OS YANG modules
- many elements use string names as keys in model-driven interfaces instead of the numerical identifiers used in the classic CLI and SNMP. The name can only be assigned or modified for these elements in releases prior to Release 15.1.R1. Elements without names are automatically assigned a name (the identifier converted to a string) during an upgrade to Release 15.1.R1 or later, and cannot be changed without manually deleting and recreating the element. It is recommended that the following elements are assigned names prior to an upgrade to Release 15.1 or later:
  - all services (**configure service vprn**, **vpls**, **epipe**, and so on)
  - **configure mirror mirror-dest**
  - **configure service pw-templates**
  - **configure service customer**
  - **configure filter ip-filter**, **ipv6-filter**, and **mac-filter**
  - **configure qos network**, **sap-ingress**, and **sap-egress**
  - **configure eth-cfm domain** and **association**
- the classic CLI **shutdown** command has been replaced with **admin-state** in model-driven interfaces
- the classic CLI commands with multiple parameters have been separated into individual leafs in model-driven interfaces
- the model-driven interfaces make extensive use of Boolean values (true and false) for configuration settings. A newly created routing instance, group, or EBGP neighbor in a model-driven interface applies the secure default behavior to reject all routes. It is compliant with RFC 8212 using the **ebgp-default-reject-policy** command. However, Nokia recommends configuring import and export policies that express the intended routing instead of using the insecure default behavior.

In model-driven configuration mode, SR OS operates with 'explicit' default handling. Users can set a leaf to the same value as the default and SR OS remembers that it was explicitly set and displays it as part of the configuration. This is similar to what RFC 6243 refers to as 'explicit' mode.

In the classic configuration mode, the default handling is similar to RFC 6243 'trim' mode. Configuration values are not reported if they are equal to the default value, even if the user explicitly configured the value.

In mixed configuration mode, the system uses 'explicit' default handling but it is not persistent. Explicitly configured default values are lost or forgotten at a High-availability CPM switchover or a reboot. Nokia does not recommend setting any leaf explicitly to its default value in mixed configuration mode (the leaf should be deleted instead).

3.1.1 Prerequisites for Using Model-Driven Management Interfaces

Before configuration editing is permitted in model-driven interfaces, the management interface configuration mode must be set to **model-driven** or **mixed**. See Management Interface Configuration Mode for details.

All loose references using IDs to certain elements (elements which use IDs as keys in classic interfaces but string names in model-driven interfaces) must be replaced with references using string names. See Loose References to IDs for details.

Strict routing policy validation is used for model-driven interfaces. The routing policy must exist for the management interface configuration mode to be changed. References to non-existent routing policies must be removed before attempting to switch modes. Strict policy validation is applied to the following routing policy references:

- ARP and ND: in the Base router and VPRN instances
- BGP: in the Base router and VPRN instances
- Global and local variables: in main policies and sub-policies
- IGMP, MLD, and PIM: in the Base router and VPRN instances
- IS-IS: in the Base router and VPRN instances
- LDP
- OSPF and OSPFv3: the Base router and VPRN instances
- Policy-option: **from**, **to**, **action**, and **default-action** statements
- Policy-option: sub-policies, **prefix-list**, **as-path**, **as-path-group**, **damping**, and **community** policies
- RIP and RIPng: in the Base router and VPRN instances
- RSVP
- Single policy-statement or logical policy expressions
- Static routes: in the Base router and VPRN instances
- Subscriber management: except for in **mld-policy** configuration for a local user database (LUDB) host

- VPLS: for BGP VSI
- VPRN: for GRT, MVPN, and VRF

Use the **tools perform system management-interface configuration-mode check** command to check if the configuration meets the preceding prerequisite reference requirements to change the management interface configuration mode. Incompatible configuration commands are displayed with an error reason if the prerequisite is not met. The following example shows several incompatible configuration commands.

```
A:node-2# tools perform system management-interface configuration-mode model-
driven check
===============================================================================
Mode Switch Validation Check
===============================================================================
Current Mode       : classic            Desired Mode       : model-driven
Configure          : Errors Detected    LI                 : No Errors
-------------------------------------------------------------------------------
Configuration Validation Errors
-------------------------------------------------------------------------------
1   : MINOR: MGMT_CORE #2004 Incompatible configuration - dynsvc-password
      configured in system security password
2   : MINOR: MGMT_CORE #2004 Incompatible configuration - 'eth-cfm association
      bridge-identifier' reference to service-id exists
3   : MINOR: MGMT_CORE #2004 Incompatible configuration - ca-profile cmpv2 url
      service-id references exist
-------------------------------------------------------------------------------
Action required: configuration requires updating before mode switch
===============================================================================
```

Note: The command does not check if the configuration contains commands that are unsupported in model-driven interfaces. For more information, refer to section “Unsupported Configuration in MD Interfaces” in the *SR OS 20.x*.R*x* Software Release Notes, part number 3HE 16194 000*x* TQZZA.

3.2 YANG Data Models

Model-driven management interfaces are based on a common infrastructure that uses YANG models as the core definition for configuration, state, and operational actions. All model-driven interfaces (NETCONF, gRPC, gNMI, and MD-CLI) take the same common underlying YANG modules and render them for the particular management interface. These YANG models are also used for telemetry.

SR OS supports:

- SR OS YANG data models

- OpenConfig YANG data models

3.2.1 SR OS YANG Data Models

The Nokia SR OS YANG modules are the base for the model-driven architecture.

SR OS configuration is divided into several top level configuration regions (see Datastores and Regions for details). The data models for each configuration region are separated into different YANG modules.

The primary configuration region (configure) is modeled in the YANG module in a single file called nokia-conf-combined.yang.

An alternative packaging of the primary configuration region is also available as a set of submodules (for example, nokia-conf-system) that all belong to a single complete module called nokia-conf. The submodules have independent revision dates and can be used to identify which parts of the configuration model have changed.

Some YANG tools may show errors about circular dependencies in the submodules. For instance, Pyang gives an error about circular dependencies but does complete the processing to build complete tree or jstree output. If circular dependencies are preventing any necessary tools from correctly processing the YANG, then use the nokia-conf-combined.yang packaging instead of the submodules. For details about enabling various sets of YANG modules, see the **yang-modules** commands.

The lawful intercept (LI) configuration region is modeled in the nokia-li-conf YANG module specified in a single file called nokia-li-conf.yang.

SR OS state information is modeled in the nokia-state YANG module specified in a single file called nokia-state-combined.yang.

The LI state information is modeled in nokia-li-state.yang which augments the primary nokia-state module.

There are also a series of nokia-types-* modules that are included by various configuration and state modules.

The SR OS YANG modules have the following attributes.

- The modules can be used with NETCONF, telemetry, or with the Set/Get RPCs of the gRPC-based gNMI service.

- The modules and submodules indicate the SR OS major release stream using a YANG extension (for example, sros-ext:sros-major-release "rel16";). Module and submodule revisions form a contiguous series of revisions inside a major release stream. There may be two files for the same module with the same revision date but with different contents because they are from two different major release streams. Each active major release stream has revisions ongoing in parallel.

All configuration modules, state modules, and types modules are advertised in the SR OS NETCONF server <hello>. Submodules are not advertised in the <hello> message.

The classic CLI **bof**, **admin**, **tools**, **debug**, **clear**, and **monitor** branches do not have equivalent YANG data models.

3.2.2 OpenConfig YANG Data Models

OpenConfig presents a vendor-independent set of YANG models. OpenConfig YANG model elements are mapped to application-specific SR OS configuration and state.

3.2.2.1 Basic Configuration

OpenConfig YANG models are available in model-driven interfaces, including the MD-CLI, gNMI, and NETCONF when enabled with the **configure system management-interface yang-modules openconfig-modules** command. Access to the OpenConfig models is different depending on the model-driven interface.

- MD-CLI
  - OpenConfig configuration statements are located in the **configure openconfig** context.
  - OpenConfig state information is located in the **state openconfig** context.
  - When a configuration is validated or committed, the system verifies that **openconfig-modules** is set to **true**. If **openconfig-modules** is set to **false** and there are OpenConfig configuration statements in the candidate, the action fails with an error indicating that the OpenConfig module cannot be disabled when OpenConfig configuration elements exist.
  - The operator must set **openconfig-modules** to **true** and perform the **validate** or **commit** action again. Assuming the configuration is complete and there are no other errors, the transaction succeeds.

- The system checks **openconfig-modules** to determine whether OpenConfig state elements can be accessed.

• gNMI and NETCONF

- The system checks **openconfig-modules** to determine whether OpenConfig models can be advertised and whether the system can accept or send OpenConfig configuration or state elements.
- If **openconfig-modules** is set to **false**, the system blocks OpenConfig edits, requests, and responses from being sent or accepted at the gNMI or NETCONF level. A <get> operation from the root without a declared namespace or branch succeeds but does not include any OpenConfig data. However, a <get> operation that explicitly requests data from the OpenConfig namespace generates an error.

• AAA rules for OpenConfig are different in the MD-CLI, NETCONF and gNMI

- A **configure openconfig** AAA profile entry applies to **configure openconfig** commands in the MD-CLI, and to config and state elements in NETCONF and gNMI.
- A **state openconfig** AAA profile entry only applies to **state openconfig** information in the MD-CLI. AAA entries for NETCONF and gNMI state elements are not supported.

3.2.2.2 Shared Model Management Support

3.2.2.2.1 Introduction

Nokia provides a suite of vendor-specific YANG models to configure the network element. OpenConfig is an informal working group which provides vendor-neutral YANG models based on the desired usage of a technology by the community. The Nokia vendor-specific model is a more complete representation of the capabilities of the network element, which includes vendor specific features and functions not described by the OpenConfig YANG models. The two YANG configuration models, Nokia’s vendor-specific and OpenConfig’s vendor-neutral, may be used together to configure the network element. Support for OpenConfig models can be established by examining the OpenConfig model with the vendor-specific deviations and augments.

3.2.2.2.2 Merging Configuration Statements

In order to ensure complete traceability and the origin of the configuration (that is, which data model configured the feature), the Nokia and OpenConfig configuration statements are maintained separately in the configuration tree. This allows for the greatest flexibility when accommodating configuration differences between the Nokia and OpenConfig models. The configuration statements are merged, giving precedence to the Nokia model configuration statements when there is a collision (that is, when the same function is configured in both the OpenConfig and Nokia models).

In order to merge configuration for objects, the keys for an object must be equal and deterministic for both the Nokia and OpenConfig models. This provides an anchor for the object and allows the configuration to be rationalized and merged. For example, augments may have been made to OpenConfig models to allow for a deterministic key where a key function is not supported. One example is the use of the **configure openconfig interfaces interface** *interface* **subinterfaces subinterface** *number* **ipv4 config primary-address** option. In this case, the OpenConfig model does not allow which of the specified interfaces should be the primary. The control of the primary interface is very important.

When configuration statements are completed using one configuration model, tab completion for a name or reference identifier is not available in the other model. For example, the name or identifier of a list entry must be equally and explicitly entered in both data models in order to share the configuration elements across the different models.

There are two different approaches taken for shared model management, on a per Nokia application basis: leaf level and list level management.

An application that supports shared model management at the leaf level allows both configuration models access to the leaf and merge operations can occur at the leaf level. If both Nokia and OpenConfig models include configuration for a leaf, the Nokia configuration takes precedence. The OpenConfig configuration statements remain in its configuration database but are not applied as part of the operational configuration

An application that supports granularity at the list level allows individual list entries for an application to be managed by one model only. The configuration model that creates the list entry is the only model that can modify or delete the list entry. An attempt to modify the list entry using the configuration access method that does not manage the list entry returns an error message identifying the managing owner of the list entry.

```
Cannot access or modify element - managed by <managing owner> module
```

Unless configured explicitly using the Nokia configuration model, a configuration element that does not have a static default value is managed by OpenConfig.

In some situations, partial or incomplete OpenConfig configurations may be allowed. For example, where the OpenConfig structure is accepted but the triggering mapping has not been configured under OpenConfig, the information is not pushed to the application. These partial configurations remain in the OpenConfig configuration tree as they are syntactically correct, however, without an application mapping event, they remain outside of the operating configuration. When a partial configuration is stored in the OpenConfig configuration tree, it does not show as an active element under the SR OS specific application, that is, via **show** commands or in the /state tree.

3.2.2.2.3 Application Support

Applications may allow for the configuration to be delivered from either the Nokia YANG model or the OpenConfig YANG models. In many cases, applications allow some level of cooperative configuration such that the configuration statements can be received from both Nokia YANG models and OpenConfig YANG models. In order to determine the level of cooperative configuration allowed by an application, the application-specific Nokia or the nokia-conf-combined.yang YANG models can be checked for the following extension statement.

```
sros-ext:shared-model-management {
                    sros-ext:openconfig false;
                }
```

If the above statement is found, the cooperative shared model management configuration is not allowed for that element and all descendants of the element.

The level of shared model management support can be viewed via the MD CLI help. The models that prevent shared model management at a specific level of the hierarchy include the following statement in the help. For example, the commands in the **configure log log-id** context display the following:

```
[ex:configure log log-id 100]
A:admin@node-2# source ?

 source

Note: 'configure log log-id 100' and all other elements in this context support
single-model management only.
```

3.2.2.2.4 Validating and Committing

Validation ensures the structure and completeness of the configuration against the OpenConfig model. It does not deliver the configuration to application. It is possible that a validation succeeds when the structure and requirements of the OpenConfig model are met.

The commit function performs the validation as above, with the additional step of delivering the converted OpenConfig statements to the application. A successful validation can be followed by a failure to commit the transaction. For example, the following scenarios result in a failed commit action:

- the Nokia application requirements are not met
- the list entry is managed by Nokia
- a resource limit enforced by the application is exceeded by merging the OpenConfig configuration

Nokia applications that include conditional “when” statements using the Nokia YANG model must have the statements satisfied by the Nokia configuration. The OpenConfig configuration cannot verify or satisfy Nokia conditional “when” statements. This approach prevents “when” statements from changing from one state to another by updating the OpenConfig statements and affecting a non-child leaf in the Nokia configuration. For example, the following message is displayed when the OpenConfig configuration sets the port ethernet mode to hybrid but the conditional “when” statement requires the Nokia configuration to satisfy the condition.

```
configure port 1/1/4 ethernet access - OpenConfig and Nokia condition mismatch -
 failed condition
```

3.2.2.2.5 Error Reporting

Errors can occur in situations such as the following:

- the OpenConfig model attempts to deliver an incomplete configuration as required by the Nokia application
- conflicts exist where an OpenConfig model attempts to access a list entry managed by Nokia
- other delivery errors from the commit operation

Failed transactions display an error message indicating the reason for the failure. A failure maintains the complete set of YANG parameters, as if the commit function had not been issued. This allows the administrator to correct the source of the error.

In the event of a delivery error, the OpenConfig path and the Nokia path are included in the error message. A sample error message is shown below.

```
<severity>:<module> #<code>: <context in which the error occurred> <related context>
 - <error message>
```

3.2.2.2.6 Using the info Command

Several variations of the **info** command are available in order to collect the required operational data required to view the configuration. The **info** command can be issued under the Nokia configuration context in the MD-CLI. These include:

info - Show the configuration as explicitly entered for the current context.

info converted - Include converted third party model configuration from the running datastore. When an object is management by OpenConfig, meaning the running configuration has an entry delivered by an OpenConfig configuration statement, the leaf or container be precedes with the statement "## 'xxx' managed by OpenConfig", where 'xxx' is the name of the element.

info converted model openconfig - Include converted third party model configuration from the running datastore with the "managed by" indicator stripped from the output.

info inheritance - Include configuration inherited from configuration groups.

The **converted** and **inheritance** options can be combined into a single command

For more information about the **info** command, see the *7450 ESS, 7750 SR, 7950 XRS, and VSR MD-CLI User Guide*.

3.2.2.2.7 Deviating and Augmenting

Deviation files are created for the OpenConfig model when the model deviates from the application requirements of the network elements, such as implementations that are not supported, added, or replaced, granularity mismatches, and different ranges. These deviations are included in an OpenConfig YANG file which contains text descriptions when different units or ranges are in place. Deviations are not raised for OpenConfig "must" statements, as the "must" statement in OpenConfig models is not supported in SR OS. The deviation file follows the naming format nokia-sr-<OpenConfigModule>-deviations.yang, for example, nokia-sr-openconfig-network-instance-deviations.yang.

It is not always necessary to use a deviation file where a specific function is not supported. For example, in the case of enumerations, when an enumerated OpenConfig value is not supported, the validation or commit function fails with an indication that the entry is not valid.

When a mapping exists for an attribute and the configuration is out of range, an error is generated. For example, the Nokia application configuration for leaf B has a range of 1 to 100, where the OpenConfig leaf B specifies a range of 1 to 300. When the OpenConfig value is set above 100, an unsupported value error message is returned.

As an example of a granularity mismatch, Nokia application leaf C supports centiseconds and OpenConfig leaf C supports milliseconds. If the OpenConfig value in milliseconds can be converted to a valid application value, the OpenConfig value is accepted. For example, OpenConfig leaf C 100 ms is converted to application leaf C 1 centisecond. However, if the OpenConfig value cannot be converted to a valid application value, an error is generated. For example, OpenConfig leaf C 125ms cannot be mapped into centiseconds.

Augments files are also included to add configuration for OpenConfig that is required by the Nokia application in order to function as expected. The augments file follows the naming format nokia-sr-<OpenConfigModule>-augments.yang.

3.3 Datastores and Regions

As per RFC 8342 a datastore is a conceptual place to store and access information. A datastore maps to an instantiated YANG data tree. See RFC 8342 for more information about datastores.

SR OS supports conventional configuration datastores (for example, running and candidate) as well as some proprietary datastores (for example, li-running).

SR OS also has a proprietary concept called a region (or configuration region). The set of branches and elements in the **configure** branch of the CLI are all in the primary configuration region simply called **configure**. The majority of SR OS configuration is in the configuration region including ports, interfaces, services and filters. Examples of other regions are:

- **bof** (boot options file)
- **li** (lawful intercept)

Each region has its own configuration datastores (running, candidate, and so on). The saved configuration for each region is stored in a separate file on compact flash or remotely (for example, bof.cfg, config.cfg, li.cfg). Regions are independently locked for configuration changes. See the output of **show system management-interface datastore-locks** for an example of per-region per-datastore information.

3.4 System-Provisioned Configuration (SPC) Objects

There is a set of configuration objects (configuration list elements and their descendants) that are provisioned (added to the <running> datastore) automatically by SR OS; for example, log-id 99. Some SPC objects are created at bootup time, and some are created or removed dynamically based on configuration. The dynamically created SPC objects are typically children objects created as a side effect of the creation of their parent object.

Some of these objects can be deleted (removed) by a user (deletable SPC objects).

- In the classic CLI these are removed by specifying the keyword **no**, which is then visible in an **info** command or in a saved config (**admin save**); for example, **no log-id 99**.
- The deletable SPC objects are not visible in a <get-config> response in the “urn:nokia.com:sros:ns:yang:sr:conf” namespace (the Nokia SR OS YANG modules) if the child leaves are all at default values.
- Some example deletable SPC objects (shown in classic CLI format) are:

```
configure system security profile default
configure system security profile default entry 10-100
configure system security profile administrative
configure system security profile administrative entry 10-112
configure system security user "admin"
configure system security user console member "default"
configure system security ssh client-cipher-list protocol-version 1 cipher 200-210
configure system security ssh client-cipher-list protocol-version 2 cipher 190-235
configure system security ssh server-cipher-list protocol-version 1 cipher 200-205
configure system security ssh server-cipher-list protocol-version 2 cipher 190-235
configure log filter 1001
configure log filter 1001 entry 10
configure log log-id 99 & 100
```

Some SPC objects cannot be deleted (non-deletable SPC objects).

- In the classic CLI, an attempt to delete these objects (for example, **no sap-ingress 1**) returns an error.

- In the MD-CLI, deleting one of these objects simply resets all descendant elements as unconfigured.
- Some non-deletable SPC objects contain leafs (or other descendants) that can be modified. Some objects cannot be modified.
- Non-deletable SPC objects do not appear in the configuration (the output of “info” in the classic CLI or the MD-CLI) unless some of the child leafs or descendants have been configured.
- The non-deletable SPC objects are not visible in a <get-config> response in the “urn:nokia.com:sros:ns:yang:sr:conf” namespace (the Nokia SR OS YANG modules) if the child leaves are all unconfigured.
- Some example non-deletable SPC objects (shown in classic CLI format) are:

```
configure system security user-template {tacplus_default|radius_default}
configure system security snmp view iso …
configure system security snmp view li-view …
configure system security snmp view mgmt-view …
configure system security snmp view vprn-view …
configure system security snmp view no-security-view …
configure system security snmp access group  xyz (a set of access groups)
configure log event-control …
configure filter log 101
configure qos … various default policies can’t be deleted
configure qos queue-group-templates … these can’t be deleted
configure card <x>
configure router network-domains network-domain “default”
configure oam-pm bin-group 1
configure call-trace trace-profile “default”
configure eth-cfm default-domain bridge-identifier <x>
```

3.5 Management Interface Configuration Mode

The management interface configuration mode controls how classic management interfaces, such as SNMP and the classic CLI, and model-driven (MD) interfaces, such as the MD-CLI, NETCONF, and gRPC/gNMI, are used to modify the configuration of the router. The **configure system management-interface configuration-mode** command must be used to enable configuration editing by MD interfaces.

Table 21 **Management Interface Configuration Mode**

		Configuration Mode		
		Classic	**Mixed**	**Model-driven**
Classic Interfaces	Classic CLI: configuration write/edit	✓	✓	
	Classic CLI: configuration read	✓	✓	✓
	Classic CLI: non-configuration commands	✓	✓	✓
	SNMP: configuration write/edit	✓		
	SNMP: non-configuration writes (such as admin reboot)	✓		
	SNMP: configuration read	✓	✓	✓
	SNMP: state read	✓	✓	✓
	SNMP: notifications (traps)	✓	✓	✓
	NETCONF (Base-R13 model): configuration write/edit	✓	✓	
	NETCONF (Base-R13 model): configuration read	✓	✓	✓
Model-driven Interfaces	MD-CLI (Nokia model): configuration write and read		✓	✓
	MD-CLI (Nokia model): state read	✓	✓	✓
	NETCONF (Nokia model): configuration write and read		✓	✓
	NETCONF (Nokia model): state read	✓	✓	✓
	Telemetry: configuration nodes		✓	✓
	Telemetry: state nodes	✓	✓	✓
	gNMI Set/Get: configuration write and read		✓	✓
	gNMI Get: state read	✓	✓	✓

Table 21 **Management Interface Configuration Mode (Continued)**

		Configuration Mode		
		Classic	**Mixed**	**Model-driven**
Features	OpenConfig YANG models			✓
	Configuration groups			✓
	MD-CLI **rollback** command			✓
	Classic CLI **admin rollback revert** command	✓	✓	
	Explicit defaults [1]			✓
	Configuration changes accepted immediately after a CPM high-availability switchover [2]	✓		✓
	Named route policy entries			✓

Note:

1. In **model-driven** mode, users can set a parameter to the same value as the default, and SR OS remembers that it was explicitly set and displays it as part of the configuration. In mixed mode these values are not persistent and they are lost or forgotten at a CPM high-availability switchover or a reboot.
2. In **mixed** mode, changes to the configuration are blocked for a few minutes after a CPM high-availability switchover event while the model-driven database is synchronized with the SR OS application layer. There is no impact to running services.

3.5.1 Mixed Configuration Mode

Mixed configuration mode is useful for operators to migrate from classic management interfaces to operating in a full model-driven mode. It allows the use of previous classic CLI scripts or other OSS integration (for configuration) although with some pre-requisites (see Prerequisites for Using Model-Driven Management Interfaces) and some limitations (see Table 21).

3.5.2 Loose References to IDs

A loose reference is a reference where the target of the reference does not have to exist. For example, **configure service pw-template 23 egress filter ip 37** can be configured (when the management interface configuration mode is **classic**) even if **ip-filter 37** does not yet exist in the configuration.

Before switching the management interface configuration mode to **model-driven** or **mixed**, all loose references using IDs must be replaced with references using string names (or removed from the configuration) for the following elements:

- all services (**configure service vprn**, **vpls**, **epipe**, and so on)
- **configure mirror mirror-dest**
- **configure service pw-templates**
- **configure service customer**
- **configure filter ip-filter**, **ipv6-filter**, and **mac-filter**
- **configure qos network**, **sap-ingress**, and **sap-egress**
- **configure eth-cfm domain** and **association**

In the following configuration example,

```
configure service pw-template 23 egress filter ip 37
```

can be changed to

```
configure service pw-template 23 egress filter-name ip ops-sec-filter-a33
```

Because **ip-filter 37** is a loose reference, it does not require a name for the configuration to be valid. However, it may be desirable to assign a name as follows, to make the binding operational.

```
configure filter ip-filter 37 name ops-sec-filter-a33
```

Note: A name can only be assigned to a filter or any element in the above list of elements which use IDs as keys in classic interfaces but string names in model-driven interfaces. It is recommended to assign names to the elements prior to an upgrade to Release 15.1.R1.

A name can also be changed in releases prior to Release 15.1.R1. Elements without names are automatically assigned a name (the ID converted to a string) during an upgrade to Release 15.1.R1 or later, and cannot be changed without manually deleting and recreating the element.

Loose references to IDs for the objects listed above cannot be created while in **mixed** or **model-driven** configuration mode. Any classic CLI scripts must also be updated to avoid the use of any of the commands below.

The following lists the set of affected loose references. Some items take a service name as an input. SR OS converts these service names to IDs, and stores the IDs in the configuration. In these cases, the service name becomes an alias at configuration edit time and is not stored as a reference.

IPsec related configuration:

```
configure service vprn interface sap ipsec-tunnel local-gateway-address
configure service vprn interface sap ip-tunnel delivery-service
configure service vprn interface sap l2tpv3-session router
configure service epipe sap l2tpv3-session router
configure service vpls sap l2tpv3-session router
configure service vprn interface sap ipsec-gw default-secure-service
configure service ies interface sap ipsec-gw default-secure-service
configure service vprn interface sap ipsec-gw dhcp server
configure service ies interface sap ipsec-gw dhcp server
configure service vprn interface sap ipsec-gw dhcp6 server
configure service ies interface sap ipsec-gw dhcp6 server
configure service vprn interface sap ipsec-gw local-address-assignment ipv4 address-
source
configure service vprn interface sap ipsec-gw local-address-assignment ipv6 address-
source
configure service ies interface sap ipsec-gw local-address-assignment ipv4 address-
source
configure service ies interface sap ipsec-gw local-address-assignment ipv6 address-
source
configure service vprn interface sap ipsec-tunnel bfd-enable
configure ipsec client-db client private-service
configure system file-transmission-profile router
```

Eth-cfm, oam-pm, and saa:

```
configure eth-cfm default-domain bridge-identifier
configure eth-cfm domain association bridge-identifier
configure oam-pm session ip router
configure oam-pm session ip router service-name
configure saa test type cpe-ping service
configure saa test type icmp-ping router
configure saa test type icmp-ping service-name
configure saa test type icmp-trace router
configure saa test type icmp-trace service-name
configure saa test type mac-ping service
configure saa test type mac-trace service
configure saa test type vprn-ping
configure saa test type vprn-ping service
configure saa test type vprn-trace
configure saa test type vprn-trace service
```

Filters:

```
configure service pw-template egress filter ipv6
```

```
configure service pw-template egress filter ip
configure service pw-template egress filter mac
configure service pw-template ingress filter ipv6
configure service pw-template ingress filter ip
configure service pw-template ingress filter mac

configure service template epipe-sap-template egress filter ip
configure service template epipe-sap-template egress filter ipv6
configure service template epipe-sap-template egress filter mac
configure service template epipe-sap-template ingress filter ip
configure service template epipe-sap-template ingress filter ipv6
configure service template epipe-sap-template ingress filter mac

configure service template vpls-sap-template egress filter ip
configure service template vpls-sap-template egress filter ipv6
configure service template vpls-sap-template egress filter mac
configure service template vpls-sap-template ingress filter ip
configure service template vpls-sap-template ingress filter ipv6
configure service template vpls-sap-template ingress filter mac

configure li li-filter-block-reservation li-reserved-block ip-filter
configure li li-filter-block-reservation li-reserved-block ipv6-filter
configure li li-filter-block-reservation li-reserved-block mac-filter
```

PKI:

```
configure system security pki ca-profile cmpv2 url
configure system security pki ca-profile ocsp service
```

QoS:

```
configure service template epipe-sap-template ingress qos
configure service template epipe-sap-template egress qos

configure service template vpls-sap-template ingress qos
configure service template vpls-sap-template egress qos

configure service pw-template ingress qos
configure service pw-template egress qos
```

Subscriber management:

```
configure service ies subscriber-interface group-interface srrp bfd-enable
configure service vprn subscriber-interface group-interface srrp bfd-enable


configure subscriber-mgmt local-user-db ipoe host host-identification service-id
configure subscriber-mgmt local-user-db ipoe host interface service-id
configure subscriber-mgmt local-user-db ipoe host match-radius-proxy-cache server
configure subscriber-mgmt local-user-db ipoe host msap-defaults service
configure subscriber-mgmt local-user-db ipoe host retail-service-id

configure subscriber-mgmt local-user-db ppp host interface service-id
configure subscriber-mgmt local-user-db ppp host l2tp group service-id
configure subscriber-mgmt local-user-db ppp host msap-defaults service
configure subscriber-mgmt local-user-db ppp host retail-service-id
```

```
configure subscriber-mgmt msap-policy vpls-only-sap-parameters igmp-
snooping mvr from-vpls

configure service vpls sap msap-defaults service
```

Miscellaneous:

```
configure vrrp policy
configure service vprn interface vrrp bfd-enable

configure service vprn interface ipv6 vrrp bfd-enable
configure router l2tp group ppp default-group-interface service-id
configure router l2tp group tunnel ppp default-group-interface service-id
configure service vprn l2tp group ppp default-group-interface service-id
configure service vprn l2tp group tunnel ppp default-group-interface service-id

configure redundancy multi-chassis peer mc-ring l3-ring in-band-control-
path service-id
configure redundancy multi-chassis peer mc-ring l3-ring ring-node connectivity-
verify service-id
configure redundancy multi-chassis peer mc-ring ring in-band-control-path service-id
configure redundancy multi-chassis peer mc-ring ring ring-node connectivity-
verify service-id

configure open-flow of-switch of-controller vprn
```

3.5.3 Transitioning Between Modes

Depending on the size of the system configuration, transitioning away from classic mode may take several seconds to several minutes while the model-driven database is populated and synchronized to the current configuration. During the transition period, configuration changes are not allowed and service is not affected.

Transitioning to classic mode is immediate with no impact to services on the router.

3.6 Configuring the CLI Engine

The CLI engine refers to the CLI environment that is being used in a user session (for example, console, Telnet, or SSH) to configure and operate the router. The CLI engine is either the classic CLI engine or the MD-CLI engine. The following terms are also used:

- preferred CLI engine — the CLI engine that is started at user login

- authorized CLI engine — a CLI engine that a user can switch to (using the CLI engine switch command (“//”)) or where a user can execute commands
- active CLI engine — the CLI engine that is currently in use for a user session

The default preferred CLI engine and authorized CLI engines for a session are determined by the management interface configuration mode, which eliminates the need to explicitly configure the CLI engine. With the use of these dynamic defaults, it is possible to transition between the different configuration modes. Table 22 summarizes the CLI engines for the management interface configuration modes.

Table 22 **Management Interface Configuration Modes and CLI Engines**

Management Interface Configuration Mode	Default Preferred CLI Engine	Default Authorized CLI Engines
classic	classic-cli	classic-cli
mixed	classic-cli	md-cli, classic-cli
model-driven	md-cli	md-cli, classic-cli (read-only)

The preferred CLI engine, and the authorized CLI engines for a session can be changed to use either the classic CLI or the MD-CLI engine.

In the classic CLI, the first engine configured is the preferred CLI engine. The default is **no cli-engine**.

```
A:node-2>config>system>management-interface>cli# cli-engine ?
  - cli-engine <engine-type> [<engine-type>...(upto 2 max)]
  - no cli-engine

 <engine-type>        : classic-cli|md-cli
```

In the MD-CLI, the **cli-engine** parameter is a user-ordered list, and the first engine from that list is configured as the preferred CLI engine. Leaving the **cli-engine** parameter unconfigured (or deleting the **cli-engine** values) maintains or reverts to the dynamic default. Table 23 summarizes the possible actions available with the MD-CLI **cli-engine** configuration.

Note: In order for the changes to the **cli-engine** parameter to take effect, log out of the CLI session and start a new session.

Table 23 **MD-CLI cli-engine Configurations**

cli-engine Configuration	Preferred CLI engine	Authorized CLI engines	Description
[classic-cli]	classic-cli	classic-cli	User is restricted to the classic CLI engine
[classic-cli md-cli]	classic-cli	classic-cli, md-cli	User can switch between classic CLI and MD-CLI engines in a session
[md-cli classic-cli]	md-cli	md-cli, classic-cli	User can switch between MD-CLI and classic CLI engines in a session
[md-cli]	md-cli	md-cli	User is restricted to the MD-CLI engine

Refer to the *MD-CLI User Guide* for information about using the MD-CLI to manage to router.

3.7 Legacy Alcatel-Lucent Base-R13 NETCONF/ YANG

SR OS supports an older variant of NETCONF and YANG called Base-R13 that is not part of the model-driven architecture.

Base-R13 NETCONF/YANG is not recommended for new deployments.

The NETCONF behavior when using Base-R13 YANG modules aligns closely to the structure and behavior of the classic CLI. The Base-R13 implementation is tightly linked to the classic CLI infrastructure. That linkage results in NETCONF behavior that follows many of the classic CLI behaviors and constraints.

In order to use the Base-R13 NETCONF/YANG, the **base-r13-modules** must be enabled under **configure system management-interface yang-modules**. The **writable-running** command must also be enabled under **configure system netconf capabilities**.

A NETCONF client selects the Base-R13 NETCONF/YANG by using the urn:alcatel-lucent.com:sros:ns:yang:conf-r13 XML namespace in the NETCONF <edit-config> requests.

3.7.1 Alcatel-Lucent Base-R13 SR OS YANG Modules

In addition to the model-driven Nokia SR OS YANG modules, SR OS also supports a second set of YANG configuration data models called the Alcatel-Lucent Base-R13 SR OS YANG modules (sometimes simply referred to as Base-R13).

The two sets of configuration models have a similar overall tree structure (just as the classic CLI and the MD-CLI have a similar overall tree structure). But the behavior of Base-R13 NETCONF/YANG is quite different from the model-driven NETCONF/YANG.

The YANG modules for the Alcatel-Lucent Base-R13 SR OS YANG modules) have the following attributes.

- The names of the modules are alu-conf-*-r13 (for example, alu-conf-log-r13). Note the -r13 suffix at the end of the names.
- The Alcatel-Lucent Base-R13 model consists of a set of modules with groupings that are all used by a single top-level configuration module called alu-conf-r13. All configuration data in the Alcatel-Lucent Base-R13 models sits in the urn:alcatel-lucent.com:sros:ns:yang:conf-r13 XML namespace.
- The Base-R13 modules can only be used in the NETCONF interface and only with the <running> datastore. They can not be used with the NETCONF <candidate> datastore or with any other management interface.
- Although the Base-R13 modules were first introduced in SR OS Release 13.0, they do not only contain objects from Release 13.0. For example, features from any later release are also configurable using versions of the Base-R13 modules that are distributed with that release.
- The Base-R13 modules align closely to the structure and behavior of the classic CLI.

The Base-R13 configuration data models are not interchangeable with the model-driven Nokia models. An XML request based on the Alcatel-Lucent Base-R13 YANG modules does not work if applied to a router using the urn:nokia.com:sros:ns:yang:sr:conf namespace and vice versa.

There are no state models for Base-R13 NETCONF/YANG.

3.7.2 SPC Objects in Base-R13 NETCONF/YANG

See the System-Provisioned Configuration (SPC) Objects section for general background information about SPC Objects in SR OS. This section only describes SPC object specifics for Base-R13 NETCONF/YANG.

The following apply to deletable SPC objects with Base-R13 NETCONF/YANG:

- The deletable SPC objects can be removed or recreated using NETCONF <edit-config> requests, but they are not visible in a <get-config> response in the urn:alcatel-lucent.com:sros:ns:yang:conf-*-r13 namespace (the Alcatel-Lucent Base-R13 SR OS YANG modules) when they are:
  - set to their default values (including all child leafs and objects)
  - removed or deleted
- The deletable SPC objects are visible in a <get-config> response in the urn:alcatel-lucent.com:sros:ns:yang:conf-*-r13 namespace (the Alcatel-Lucent Base-R13 SR OS YANG modules), if a child leaf or object is changed away from the default value; for example, changing **log-99** to **time-format local**.

The following apply to non-deletable SPC objects with Base-R13 NETCONF/YANG:

- The non-deletable SPC objects are not visible in a <get-config> response in the urn:alcatel-lucent.com:sros:ns:yang:conf-*-r13 namespace (the Alcatel-Lucent Base-R13 SR OS YANG modules) when they are set to their default values (including all child leafs and objects).
- The non-deletable SPC objects are visible in a <get-config> response in the urn:alcatel-lucent.com:sros:ns:yang:conf-*-r13 namespace (the Alcatel-Lucent Base-R13 SR OS YANG modules), if a child leaf or object is changed away from the default value; for example, setting the card type.

Some non-deletable SPC objects are visible in a <get-config> request in the urn:alcatel-lucent.com:sros:ns:yang:conf-*-r13 namespace (the Alcatel-Lucent Base-R13 SR OS YANG modules), even if they are set to default values:

```
configure system security cpu-protection policy 254 and 255
configure router interface "system"
configure service customer 1
```

4 SNMP

4.1 SNMP Overview

This section provides an overview of the Simple Network Management Protocol (SNMP).

4.1.1 SNMP Architecture

The Service Assurance Manager (SAM) is comprised of two elements: managers and agents. The manager is the entity through which network management tasks are facilitated. Agents interface managed objects. Managed devices, such as bridges, hubs, routers, and network servers can contain managed objects. A managed object can be a configuration attribute, performance statistic, or control action that is directly related to the operation of a device.

Managed devices collect and store management information and use Simple Network Management Protocol (SNMP). SNMP is an application-layer protocol that provides a message format to facilitate communication between SNMP managers and agents. SNMP provides a standard framework to monitor and manage devices in a network from a central location.

An SNMP manager controls and monitors the activities of network hosts which use SNMP. An SNMP manager can obtain (get) a value from an SNMP agent or store (set) a value in the agent. The manager uses definitions in the management information base (MIB) to perform operations on the managed device such as retrieving values from variables or blocks of data, replying to requests, and processing traps.

Between the SNMP agent and the SNMP manager the following actions can occur:

- The manager can get information from the agent.
- The manager can set the value of a MIB object that is controlled by an agent.
- The agent can send traps to notify the manager of significant events that occur on the router.

4.1.2 Management Information Base

A MIB is a formal specifications document with definitions of management information used to remotely monitor, configure, and control a managed device or network system. The agent's management information consists of a set of network objects that can be managed with SNMP. Object identifiers are unique object names that are organized in a hierarchical tree structure. The main branches are defined by the Internet Engineering Task Force (IETF). When requested, the Internet Assigned Numbers Authority (IANA) assigns a unique branch for use by a private organization or company. The branch assigned to Nokia (TiMetra) is 1.3.6.1.4.1.6527.

The SNMP agent provides management information to support a collection of IETF specified MIBs and a number of MIBs defined to manage device parameters and network data unique to Nokia's router.

4.1.3 SNMP Protocol Operations

Between the SNMP agent and the SNMP manager the following actions can occur:

- The manager can get information from the agent.
- The manager can set the value of a MIB object that is controlled by an agent.
- The agent notifies the manager of significant events that occur on the router.

4.1.4 SNMP Versions

The agent supports multiple versions of the SNMP protocol.

- SNMP Version 1 (SNMPv1) is the original Internet-standard network management framework.

  SNMPv1 uses a community string match for authentication.
- The OS implementation uses SNMPv2c, the community-based administrative framework for SNMPv2. SNMPv2c uses a community string match for authentication.
- In SNMP Version 3 (SNMPv3), USM defines the user authentication and encryption features. View Access Control MIB (VACM) defines the user access control features. The SNMP-COMMUNITY-MIB is used to associate SNMPv1/SNMPv2c community strings with SNMPv3 VACM access control.

  SNMPv3 uses a username match for authentication.

4.1.5 Management Information Access Control

By default, the OS implementation of SNMP uses SNMPv3. SNMPv3 incorporates security model and security level features. A security model is the authentication type for the group and the security level is the permitted level of security within a security model. The combination of the security level and security model determines which security mechanism handles an SNMP packet.

To implement SNMPv1 and SNMPv2c configurations, several access groups are predefined. These access groups provide standard read-only, read-write, and read-write-all access groups and views that can simply be assigned community strings. In order to implement SNMP with security features, security models, security levels, and USM communities must be explicitly configured. Optionally, additional views which specify more specific OIDs (MIB objects in the subtree) can be configured.

Access to the management information in as SNMPv1/SNMPv2c agent is controlled by the inclusion of a community name string in the SNMP request. The community defines the sub-set of the agent's managed objects can be accessed by the requester. It also defines what type of access is allowed: read-only or read-write.

The use of community strings provide minimal security and context checking for both agents and managers that receive requests and initiate trap operations. A community string is a text string that acts like a password to permit access to the agent on the router.

Nokia's implementation of SNMP has defined three levels of community-named access:

- Read-Only permission — Grants only read access to objects in the MIB, except security objects.
- Read-Write permission — Grants read and write access to all objects in the MIB, except security objects.
- Read-Write-All permission — Grants read and write access to all objects in the MIB, including security objects.

4.1.6 User-Based Security Model Community Strings

User-based security model (USM) community strings associates a community string with an SNMPv3 access group and its view. The access granted with a community string is restricted to the scope of the configured group.

4.1.7 Views

Views control the access to a managed object. The total MIB of a router can be viewed as a hierarchical tree. When a view is created, either the entire tree or a portion of the tree can be specified and made available to a user to manage the objects contained in the subtree. Object identifiers (OIDs) uniquely identify managed objects. A view defines the type of operations for the view such as read, write, or notify.

OIDs are organized in a hierarchical tree with specific values assigned to different organizations. A view defines a subset of the agent’s managed objects controlled by the access rules associated with that view.

The Nokia SNMP agent associates SNMPv1 and SNMPv2c community strings with a SNMPv3 view.

The following system-provisioned views are available through the **config**>**system**>**security**>**snmp# view** context, which are particularly useful when configuring SNMPv1 and SNMPv2c:

- “iso” view—intended for administrative-type access to the entire supported object tree (except Lawful Interception). The "iso" view is automatically associated with any SNMP community configured in the **config**>**system**>**security**>**snmp** context that has an access-permission of “r”, “rw”, or “rwa”.
- “no-security” view—similar to “iso” view, but removes access to several security areas of the object tree (such as SNMP communities, user and profile configuration, SNMP engine ID, and so on). The “no-security” view is generally recommended over the “iso” view to reduce access to security objects.
- “li-view” view—provides access to a small set of Lawful Interception related objects
- “mgmt-view” view—provides access to IF-MIB and a few other basics. The "mgmt-view" view is automatically associated with any SNMP community configured in the **config**>**system**>**security**>**snmp** context that has an access-permission of “mgmt”.
- “vprn-view” view—used to limit access to objects associated with a specific VPRN (for example, the Per-VPRN Logs and SNMP Access feature). The "vprn-view" view is automatically associated with any SNMP community configured in the **config**>**service**>**vprn**>**snmp** context.

4.1.8 Access Groups

Access groups associate a user group and a security model to the views the group can access. An access group is defined by a unique combination of a group name, security model (SNMPv1, SNMPv2c, or SNMPv3), and security level (no-authorization-no privacy, authorization-no-privacy, or privacy).

An access group, in essence, is a template which defines a combination of access privileges and views. A group can be associated to one or more network users to control their access privileges and views.

When configuring access groups, the “no-security” view is generally recommended over the “iso” view in order to restrict access to security objects.

A set of system-provisioned access groups and system-created communities are available in SR OS. The system-provisioned groups and communities that begin with “cli-” are only used for internal CLI management purposes and are not exposed to external SNMP access.

Additional access parameters must be explicitly configured if the preconfigured access groups and views for SNMPv1 and SNMPv2c do not meet your security requirements.

4.1.9 Users

By default, authentication and encryption parameters are not configured. Authentication parameters which a user must use in order to be validated by the router can be modified. SNMP authentication allows the device to validate the managing node that issued the SNMP message and determine if the message has been tampered with.

User access and authentication privileges must be explicitly configured. In a user configuration, a user is associated with an access group, which is a collection of users who have common access privileges and views (see Access Groups).

4.1.10 Per-VPRN Logs and SNMP Access

Configuration of VPRN-specific logs (with VPRN-specific syslog destinations, SNMP trap, notification groups, and so on) is supported in addition to the global logs configured under **config**>**log**. By default, the event streams for VPRN logs contain only events that are associated with the particular VPRN. Access to the entire system-wide set of events (VPRN and non-VPRN) can be enabled using the **config**>**log**>**services-all-events** command.

Each VPRN service can be configured with a set of SNMP v1/v2c community strings. These communities are associated with the system provisioned SNMP view called "vprn-view", which limits SNMP access to objects associated with a specific VPRN (along with a few basic system level OIDs).

SNMP communities configured under a VPRN are also associated with the SNMP context "vprn". For example, walking the ifTable (IF-MIB) using the community configured for VPRN 5 returns counters and status for interfaces in VPRN 5 only.

4.1.11 Per-SNMP Community Source IP Address Validation

SNMPv1 and SNMPv2c requests can be validated against per-snmp-community whitelists (**src-access-list**) of configured source IPv4 and IPv6 addresses. Source IP address lists can be configured and then associated with an SNMP community.

SNMPv1 and SNMPv2c requests that fail the source IP address and community validation checks are discarded and are logged as SNMP event 2003 authenticationFailure (suppressed by default under “event-control”).

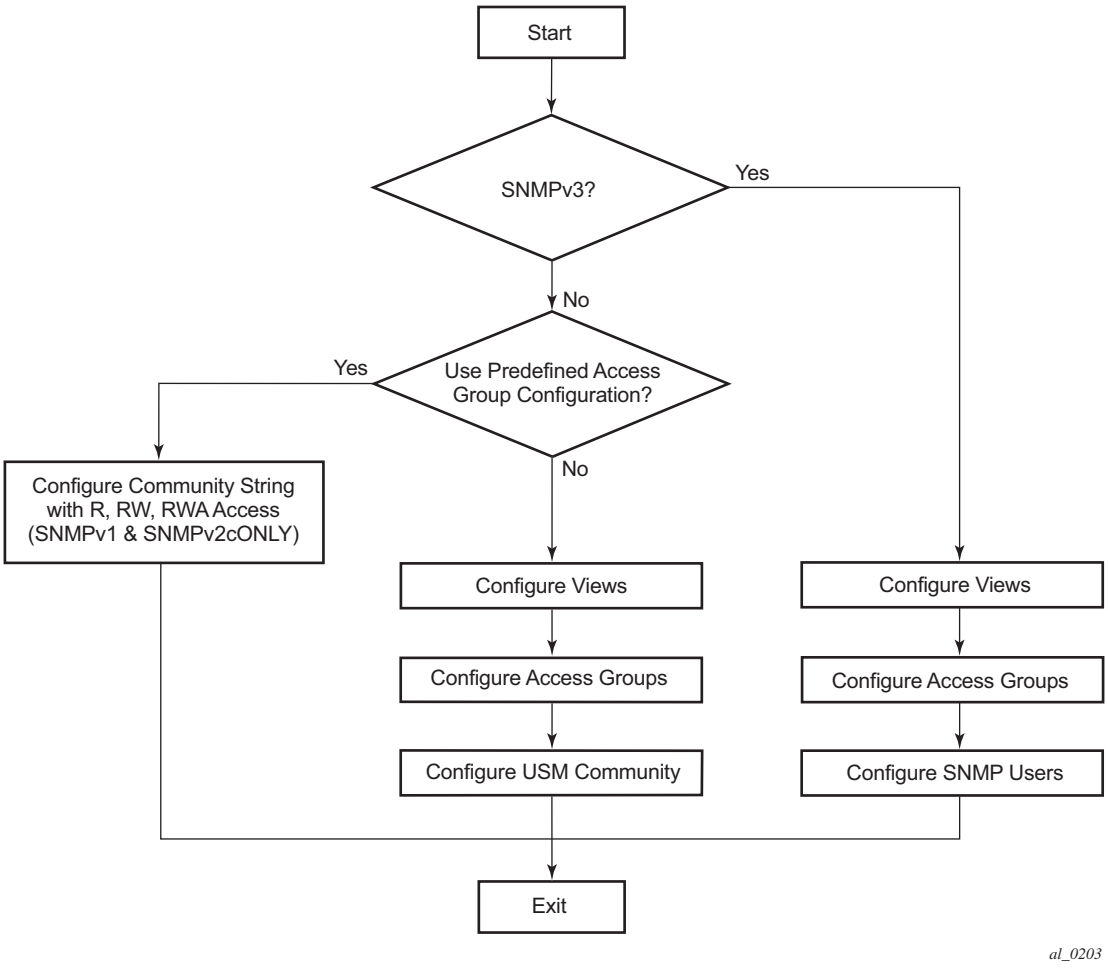
4.2 SNMP Versions

SNMPv1 and SNMPv2c do not provide security, authentication, or encryption. Without authentication, a non-authorized user could perform SNMP network management functions and eavesdrop on management information as it passes from system to system. Many SNMPv1 and SNMPv2c implementations are restricted read-only access, which, in turn, reduces the effectiveness of a network monitor in which network control applications cannot be supported.

To implement SNMPv3, an authentication and encryption method must be assigned to a user in order to be validated by the router. SNMP authentication allows the router to validate the managing node that issued the SNMP message and determine if the message was tampered with.

Figure 14 depicts the configuration requirements to implement SNMPv1/SNMPv2c, and SNMPv3.

Figure 14 **SNMPv1 and SNMPv2c Configuration and Implementation Flow**



4.3 Best Practices for SNMP Information Retrieval

This section describes best practices for achieving optimal performance when retrieving high volumes of data from SR OS using SNMP.

4.3.1 SNMP GetBulkRequest

The SNMP GetBulkRequest method should be used instead of GetRequest or GetNextRequest.

During GetBulkRequest processing, the SR OS SNMP layer uses all the objects from the application data that it can from the returned table row to continue filling in the SNMP reply.

To maximize the advantage of SR OS pre-fetching and caching optimizations, construct GetBulkRequests with a sequential list of OIDs that represent sequential columns from the same SNMP table row. For example, enter the following objects and OIDs in the GetBulkRequest request to perform a row-by-row retrieval:

```
interface A, counter 1
interface A, counter 2
...
interface A, counter N
```

Do not perform column-by-column retrievals for GetBulkRequest requests, as in the following example:

```
interface A, counter 1
interface B, counter 1
...
interface X, counter 1
```

To align all responses at the start of a row, avoid performing GetBulkRequests that result in more data than can fit in a single response. This can be accomplished by limiting the max-repetitions depending on the number of repeaters and OIDs, and the size of data returned for each repeater and OID.

4.3.2 Queueing, RTT, and Collection Performance

The best collection performance is achieved if the SNMP manager keeps the SNMP input queue of the SR OS router "wet" (that is, non-empty), but without overflowing it. If maximum performance is required, then the SNMP manager should always have at least two outstanding requests toward the SR OS router: one request that the SR OS router is currently processing (but to which it has not replied yet), and one request that is waiting in the SNMP input queue of the SR OS router.

When the 7750 SR replies to the request, it immediately processes the next request if one is waiting in the input queue.

When the SNMP manager receives the reply, it immediately sends another request to the 7750 SR SNMP input queue.

If the round trip time (RTT) between the SNMP manager and the 7750 SR is significant, the SNMP manager may need to have more than two outstanding requests to maximize collection performance.

The SNMP manager must also avoid sending too many requests at a high rate without waiting for responses. A large number of outstanding requests can cause a backup in the SNMP input queue in SR OS. A backup can cause a long delay in response to the last item in the queue and a timeout on the SNMP manager. It can also cause discards at the SNMP input queue in SR OS.

4.4 Configuration Notes

This section describes SNMP configuration restrictions.

4.4.1 General

- To avoid management systems attempting to manage a partially booted system, SNMP will remain in a shut down state if the configuration file fails to complete during system startup. While shutdown, SNMP gets and sets are not processed. However, notifications are issued if an SNMP trap group has been configured.

  In order to enable SNMP, the portions of the configuration that failed to load must be initialized properly. Start SNMP with the **config>system>snmp>no shutdown** CLI command.
- Use caution when changing the SNMP engine ID. If the SNMP engine ID is changed in the **config>system>snmp> engineID** *engine-id* context, the current configuration must be saved and a reboot must be executed. If not, the previously configured SNMP communities and logger trap-target notify communities will not be valid for the new engine ID.

4.5 Configuring SNMP with CLI

This section provides information about configuring SNMP with CLI.

4.5.1 SNMP Configuration Overview

This section describes how to configure SNMP components which apply to SNMPv1 and SNMPv2c, and SNMPv3 on the router.

4.5.1.1 Configuring SNMPv1 and SNMPv2c

Nokia routers are based on SNMPv3. To use the routers with SNMPv1 and/or SNMPv2c, SNMP community strings must be configured. Three pre-defined access methods are available when SNMPv1 or SNMPv2c access is required. Each access method (**r**, **rw**, or **rwa**) is associated with an SNMPv3 access group that determines the access privileges and the scope of managed objects available. The **community** command is used to associate a community string with a specific access method and the required SNMP version (SNMPv1 or SNMPv2c). The access methods are:

- Read-Only — Grants read only access to the entire management structure with the exception of the security area.
- Read-Write — Grants read and write access to the entire management structure with the exception of the security area.
- Read-Write-All — Grants read and write access to the entire management structure, including security.

If the predefined access groups do not meet your access requirements, then additional access groups and views can be configured. The **usm-community** command is used to associate an access group with an SNMPv1 or SNMPv2c community string. Nokia does not recommend associating a **usm-community** with an SNMP access group that is configured with the **li** (lawful intercept) context.

SNMP trap destinations are configured in the **config**>**log**>**snmp-trap-group** context.

4.5.1.2 Configuring SNMPv3

The OS implements SNMPv3. If security features other than the default views are required, then the following parameters must be configured:

- Configure views
- Configure access groups
- Configure SNMP users

4.5.2 Basic SNMP Security Configuration

This section provides information to configure SNMP parameters and provides examples of common configuration tasks. The minimal SNMP parameters are:

For SNMPv1 and SNMPv2c:

- Configure community string parameters.

For SNMPv3:

- Configure view parameters
- Configure SNMP group
- Configure access parameters
- Configure user with SNMP parameters

The following displays SNMP default views, access groups, and attempts parameters.

```
A:ALA-1>config>system>security>snmp# info detail
----------------------------------------------
             view iso subtree 1
                 mask ff type included
             exit
             view no-security subtree 1
                 mask ff type included
             exit
             view no-security subtree 1.3.6.1.6.3
                 mask ff type excluded
             exit
             view no-security subtree 1.3.6.1.6.3.10.2.1
                 mask ff type included
             exit
             view no-security subtree 1.3.6.1.6.3.11.2.1
                 mask ff type included
             exit
             view no-security subtree 1.3.6.1.6.3.15.1.1
                 mask ff type included
```

```
            exit
            access group snmp-ro security-model snmpv1 security-level no-auth-no-
privacy read no-security notify no-security
            access group snmp-ro security-model snmpv2c security-level no-auth-no-
privacy read no-security notify no-security
            access group snmp-rw security-model snmpv1 security-level no-auth-no-
privacy read no-security write no-security notify no-security
            access group snmp-rw security-model snmpv2c security-level no-auth-no-
privacy read no-security write no-security notify no-security
            access group snmp-rwa security-model snmpv1 security-level no-auth-no-
privacy read iso write iso notify iso
           access group snmp-rwa security-model snmpv2c security-level no-auth-no-
privacy read iso write iso notify iso
            access group snmp-trap security-model snmpv1 security-level no-auth-
no-
privacy notify iso
            access group snmp-trap security-model snmpv2c security-level no-auth-
no-privacy notify iso
            attempts 20 time 5 lockout 10
```

4.5.3 Configuring SNMP Components

4.5.3.1 Configuring a Community String

SNMPv1 and SNMPv2c community strings are used to define the relationship between an SNMP manager and agent. The community string acts like a password to permit access to the agent. The access granted with a community string is restricted to the scope of the configured group.

One or more of these characteristics associated with the string can be specified:

- Read-only, read-write, and read-write-all permission for the MIB objects accessible to the community.
- The SNMP version, SNMPv1 or SNMPv2c.

Default access features are pre-configured by the agent for SNMPv1/SNMPv2c.

Use the following CLI syntax to configure community options:

```
config>system>security>snmp
community community-string access-permissions [version
 SNMP version]
```

The following displays an SNMP community configuration example:

```
*A:cses-A13>config>system>security>snmp# info
----------------------------------------------
```

```
                community "uTdc9j48PBRkxn5DcSjchk" hash2 rwa version both
                community "Lla.RtAyRW2" hash2 r version v2c
                community "r0a159kIOfg" hash2 r version both
-----------------------------------------------
*A:cses-A13>config>system>security>snmp#
```

4.5.3.2 Configuring View Options

Use the following CLI syntax to configure view options:

CLI Syntax:
```
config>system>security>snmp
view view-name subtree oid-value
   mask mask-value [type {included | excluded}]
```

The following displays a view configuration example:

```
*A:cses-A13>config>system>security>snmp# info
-----------------------------------------------
                view "testview" subtree "1"
                    mask ff
                exit
                view "testview" subtree "1.3.6.1.2"
                    mask ff type excluded
                exit
                community "uTdc9j48PBRkxn5DcSjchk" hash2 rwa version both
                community "Lla.RtAyRW2" hash2 r version v2c
                community "r0a159kIOfg" hash2 r version both
-----------------------------------------------
*A:cses-A13>config>system>security>snmp#
```

4.5.3.3 Configuring Access Options

The **access** command creates an association between a user group, a security model and the views that the user group can access. Access must be configured unless security is limited to the preconfigured access groups and views for SNMPv1 and SNMPv2. An access group is defined by a unique combination of the group name, security model and security level.

Use the following CLI syntax to configure access features:

CLI Syntax:
```
config>system>security>snmp
access group group-name security-model security-model
 security-level security-level [context context-name
 [prefix-match]] [read view-name-1] [write view-name-2]
 [notify view-name-3]
```

The following displays an access configuration with the view configurations.

```
*A:cses-A13>config>system>security>snmp# info
----------------------------------------------
                view "testview" subtree "1"
                    mask ff
                exit
                view "testview" subtree "1.3.6.1.2"
                    mask ff type excluded
                exit
                access group "test" security-model usm security-level auth-no-pr
ivacy read "testview" write "testview" notify "testview"
                community "uTdc9j48PBRkxn5DcSjchk" hash2 rwa version both
                community "Lla.RtAyRW2" hash2 r version v2c
                community "r0a159kIOfg" hash2 r version both
----------------------------------------------
*A:cses-A13>config>system>security>snmp#
```

Use the following CLI syntax to configure user group and authentication parameters:

CLI Syntax:

```
config>system>security# user user-name
access [ftp] [snmp] [console]
snmp
   authentication [none]|[[hash]{md5 key | sha key}
      privacy {none | des-key | aes-128-cfb-key key}]
   group group-name
```

The following displays a user's SNMP configuration example.

```
A:ALA-1>config>system>security# info
----------------------------------------------
user "testuser"
access snmp
snmp
authentication hash md5 e14672e71d3e96e7a1e19472527ee969 privacy none
group testgroup
exit
exit
...
----------------------------------------------
A:ALA-1>config>system>security#
```

4.5.3.4 Configuring USM Community Options

User-based security model (USM) community strings associate a community string with an SNMPv3 access group and its view. The access granted with a community string is restricted to the scope of the configured group.

By default, the OS implementation of SNMP uses SNMPv3. However, to implement SNMPv1 and SNMPv2c, USM community strings must be explicitly configured.

Nokia does not recommend associating a **usm-community** with an SNMP access group that is configured with the **li** (lawful intercept) context.

Use the following CLI syntax to configure USM community options:

CLI Syntax: `config>system>security>snmp`
`usm-community community-string group group-name`

The following displays a SNMP community configuration example:

```
A:ALA-1>config>system>security>snmp# info
----------------------------------------------
view "testview" subtree "1"
                    mask ff
                exit
                view "testview" subtree "1.3.6.1.2"
                    mask ff type excluded
                exit
                access group "test" security-model usm security-level auth-no-pr
ivacy read "testview" write "testview" notify "testview"
                community "uTdc9j48PBRkxn5DcSjchk" hash2 rwa version both
                community "Lla.RtAyRW2" hash2 r version v2c
                community "r0a159kIOfg" hash2 r version both
----------------------------------------------
A:ALA-1>config>system>security>snmp#
```

The group **grouptest** was configured in the **config>system>security>snmp>access** CLI context.

4.5.3.5 Configuring Other SNMP Parameters

Use the following CLI syntax to modify the system SNMP options:

CLI Syntax: `config>system>snmp`
`engineID` *`engine-id`*
`general-port` *`port`*
`packet-size` *`bytes`*
`no shutdown`

The following example displays the system SNMP default values:

```
A:ALA-104>config>system>snmp# info detail
----------------------------------------------
            shutdown
            engineID "0000xxxx000000000xxxxx00"
            packet-size 1500
            general-port 161
----------------------------------------------
A:ALA-104>config>system>snmp#
```

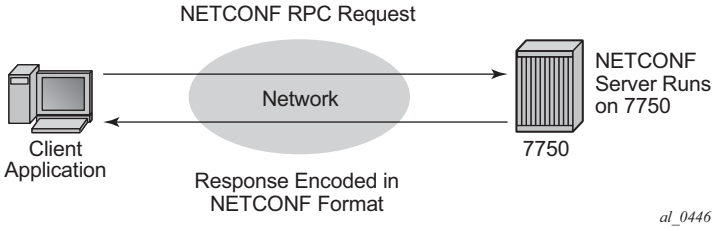
5 NETCONF

5.1 NETCONF Overview

NETCONF is a standardized IETF configuration management protocol specified in RFC 6241, *Network Configuration Protocol (NETCONF)*. It is secure, connection-oriented, and runs on top of the SSHv2 transport protocol as specified in RFC 6242, *Using the NETCONF Configuration Protocol over Secure Shell (SSH)*. NETCONF is an XML-based protocol that can be used as an alternative to CLI or SNMP for managing an SR OS router.

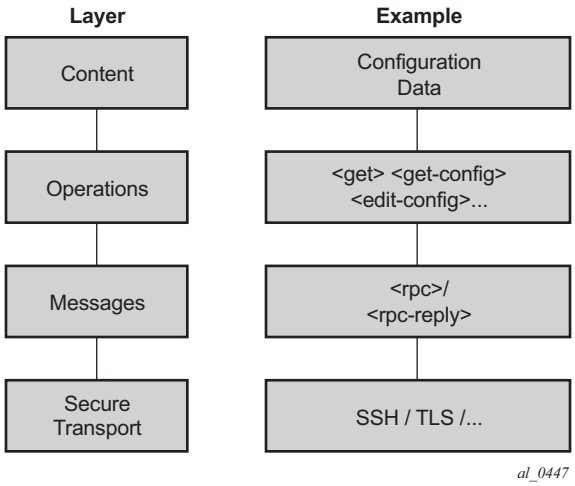
NETCONF uses RPC messaging for communication between a NETCONF client and the NETCONF server running on SR OS. An RPC message and configuration data is encapsulated within an XML document. These XML documents are exchanged between a NETCONF client and a NETCONF server in a request/response type of interaction. The SR OS NETCONF interface supports both configuration and retrieval of operational information. Figure 15 shows a NETCONF RPC request.

Figure 15 **NETCONF RPC Request**



NETCONF can be conceptually partitioned into four layers as described in RFC 6241. Figure 16 shows the NETCONF layers.

Figure 16 **NETCONF Layers (RFC 6241)**



5.2 NETCONF in SR OS

NETCONF can be used on an SR OS router to perform router management operations including:

- changing the configuration of the router (using the <edit-config> operation)
- reading the configuration of the router (using the <get-config> operation, equivalent to the **info** command while in the **configure** branch of the SR OS CLI)
- reading operational status and data using the <get> operation (equivalent to the **show** commands in the SR OS CLI, or executing the **info** command while in the state branch of MD-CLI)
- notifications on an SR OS router (equivalent to the SR OS log events)
- operations equivalent to MD-CLI commands, such as **admin**, **file**, **clear**, **oam**, and **ping**, using md-cli-raw-command (See NETCONF Operations Using the md-cli-raw-command Request.)

The SR OS NETCONF server supports both the base:1.1 capability and the base:1.0 capability.

SR OS NETCONF supports an XML-based content layer and a CLI content layer.

5.2.1 Transport and Sessions

SSH transport for NETCONF is supported on TCP port 830 (default) or port 22 with IPv4 or IPv6 in-band in the “Base” routing instance or in a VPRN, or out-of-band in the “Management” routing instance on the CPM Ethernet ports.

NETCONF SSH sessions (the same as CLI, SCP, and sFTP sessions) are subject to any configurable and non-configurable session limits; for example, inbound-max-sessions.

Both the SSH server and NETCONF protocol must be enabled in the router configuration in order to use NETCONF.

NETCONF sessions do not time out automatically and are not subject to the CLI session timeout. Operators can disconnect sessions manually using the **admin disconnect** command.

A client establishing a NETCONF session must log in to the router so user accounts must exist for NETCONF on SR OS. An access type **netconf** is provided. For access to the Nokia SR OS YANG data models, only **netconf** access is necessary.

Authentication using the local user database is supported for NETCONF users. The **access netconf** statement must be configured in the local user record. Also, NETCONF runs over SSH, and SSH supports RADIUS/TACACS+ user authentication.

For RADIUS, the **access netconf** statement must be configured in **user-template radius_default** (and **radius use-default-template** must be enabled), or the RADIUS server must send the Timetra-Access VSA with a value that includes **netconf** access.

For TACACS+, the **access netconf** statement must be configured in **user-template tacplus_default** (and **tacplus use-default-template** must be enabled).

Authorization is supported for configuration and state elements in NETCONF. The local, RADIUS, or TACACS+ authorization rules are translated and applied to NETCONF requests to modify or see configuration or state data.

5.2.2 Datastores and URLs

SR OS supports the following datastores:

- <running>

- <candidate>
- <startup>
- <intended>

Some NETCONF functions use data from <url> locations.

The supported datastores can be obtained through the “/yang-library” state data model that contains a list of supported datastores, as defined in RFC 8525.

The :candidate capability is advertised in the SR OS NETCONF server <hello> as:

```
<capability>urn:ietf:params:netconf:capability:candidat
 e:1.0</capability>
```

Configuration changes (using the Nokia SR OS YANG data models) made to the <candidate> datastore take effect after a successful <commit> operation.

The <intended> datastore is a read-only datastore that represents the configuration after configuration transformations (such as configuration group expansion) to <running> are performed.

The <startup> datastore and <url> can only be used with <copy-config> and <delete-config> and are not supported with any other operations (including <edit-config>, <get-config>, <get>, <validate>, and so on).

The :startup capability is advertised in the SR OS NETCONF server <hello> as:

```
<capability>urn:ietf:params:netconf:capability:startup:
 1.0</capability>
```

The <url> supports the same options as CLI <file-url>: local urls (CF) and remote urls (ftp and tftp).

The :url capability is advertised in the SR OS NETCONF server <hello> as:

```
<capability>urn:ietf:params:netconf:capability:url:1.0?
 scheme=ftp,tftp,file</capability>
```

The following examples show the format of each URL scheme:

- <url>ftp://name:passwd@*IP_ADDRESS*/myfiles/myfile.cfg</url>
- <url>tftp://name:passwd@*IP_ADDRESS*/myfiles/myfile.cfg</url>
- <url>file:///cf3:/myfiles/myfile.cfg</url>
- <url>cf3:/myfiles/myfile.cfg</url>

Note: The examples use “///” for the file URL. Also, the file://localhost/... format is not supported.

The BOF is not considered part of any configuration datastore.

Debug configuration (such as debug mirrors, or anything saved with **admin debug-save**) is not considered part of any configuration datastore.

Lawful Intercept (LI) configurations can be performed using the NETCONF Nokia SR OS YANG modules (including configuring any LI log-ids needed to subscribe/receive LI NETCONF notifications). The same user permissions apply using NETCONF as with MD-CLI (in other words, only LI users can access LI data).

The <candidate> datastore supports the XML content layer only. Requests/replies to/from the <candidate> datastore cannot contain the CLI content layer.

5.2.3 NETCONF Operations and Capabilities

Each RPC request can only contain one operation. Table 24 summarizes the protocol operations and capabilities supported on the 7450 ESS, 7750 SR, and 7950 XRS.

Table 24 **Summary of Operations and Capabilities**

Support	Capabilities	Operations
Base Protocol Operations	—	• <get> • <get-config> • <edit-config> • <copy-config> • <delete-config> • <lock> • <unlock> • <close-session> • <kill-session>

Table 24 **Summary of Operations and Capabilities (Continued)**

Support	Capabilities	Operations
RFC 6241 [1]	writable-running capability	—
	candidate configuration capability	• <commit> • <discard-changes>
	confirmed commit	<cancel-commit>
	validate	<validate>
	startup	—
	URL	—
	rollback-on-error	—
RFC 6243	with-defaults	—
RFC 5277	notification	<create-subscription>
	interleave	—
RFC 6022	ietf-network-monitoring	<get-schema>
RFC 8525	ietf-yang-library	—
RFC 8526	—	<get data>

Note:

1. Optional capabilities defined in RFC 6241 are supported

Table 25 shows supported NETCONF operations.

Table 25 **Supported Standard NETCONF Operations and Arguments**

Operation	Arguments
get-config	source/[configuration-region] [filter]
edit-config	target/[configuration-region] [default-operation] [test-option] [error-option] config
copy-config	target/[configuration-region] source/[configuration-region]

Table 25 **Supported Standard NETCONF Operations and Arguments**

Operation	Arguments
delete-config	target
lock	target/[configuration-region]
unlock	target/[configuration-region]
get	[filter] [configuration-region]
close-session	—
kill-session	session-id
discard-changes	[configuration-region]
validate	source/[configuration-region]
commit	[confirmed] [confirm-timeout] [persist] [persist-id] [configuration-region]
cancel-commit	[persist-id]
create-subscription	[stream] [startTime] [stopTime]
get-schema	identifier [version] [format]
get-data	datastore [subtree-filter] [max-depth] [with-defaults]

Note: Bracketed arguments are optional.

X/[configuration-group] means that the [configuration-group] can only be used as a child of *X*.

Table 26 lists protocol operations and level of support in SR OS NETCONF servers, and limitations, if any, in the current implementation.

Table 26 **Protocol Operations and Level of Support in Nokia SR OS NETCONF Servers**

Protocol Operation	Example	Supported	Notes
get-config	<?xml version="1.0" encoding="UTF-8"?> <rpc message-id="101" xmlns="urn:ietf:params:xml:ns:netconf:base:1.0"> <get-config> <source> **<running/>** </source> </get-config> </rpc>]]>]]>	Yes	—
	<source> **<startup/>** </source>	No	—
	<source> **<candidate/>** </source>	Yes	—
	<source> **<config/>** </source>	No	—
	<source> **<url/>** </source>	No	—
	<?xml version="1.0" encoding="UTF-8"?> <rpc message-id="101"xmlns="urn:ietf:params:xml:ns:netconf:base:1.0"> <get-config> <source><running/></source> **<filter type="subtree">** **</filter>** </get-config> </rpc>]]>]]>	Yes	A <filter> is an optional argument. All subtree filters are supported in SR OS except for “attribute match expressions”.
	<filter type="xpath"> **</filter>**	No	—

Table 26 **Protocol Operations and Level of Support in Nokia SR OS NETCONF Servers**

Protocol Operation	Example	Supported	Notes
	<source> **<configuration-region>...</ configuration-region** </source>	Yes	Optional. Can be "li" or "configure". The default if not specified is "configure". Can only be used with the Nokia SR OS modules. A datastore must be specified inside the <source> if the <configuration-region> is used.
get	<?xml version="1.0" encoding="UTF-8"?> <rpc message-id="101" xmlns="urn:ietf:params:xml:ns:netconf:base:1.0"> <get/> </rpc>]]>]]>	Yes	Retrieves both configuration and state data if in XML content layer. Retrieves state data but no configuration data if in CLI content layer.
	<?xml version="1.0" encoding="UTF-8"?> <rpc message-id="101" xmlns="urn:ietf:params:xml:ns:netconf:base:1.0"> <get> **<filter type="subtree">** **</filter>** </get> </rpc>]]>]]>	Yes	A <filter> is an optional argument. Subtree filters are supported except for "attribute match expressions".
	<filter type="xpath"> **</filter>**	No	—
	<configuration-region>...</ configuration-region>	Yes	Optional. Can be "li" or "configure". The default if not specified is "configure". Can only be used with the Nokia SR OS modules.

Table 26 **Protocol Operations and Level of Support in Nokia SR OS NETCONF Servers**

Protocol Operation	Example	Supported	Notes
edit-config	<?xml version="1.0" encoding="UTF-8"?> <rpc message-id="101" xmlns="urn:ietf:params:xml:ns:netconf:base:1.0"> <edit-config> <target/> <default-operation/> <test-option/> <error-option/> <config/> </edit-config> </rpc>]]>]]>	Yes	<default-operation>, <test-option>, and <error-option> are optional arguments.
	<target> **<url/>** </target>	No	—
	<target> **<running/>** </target>	Yes	Can be used when all of the following are true: • CLI content layer is used • writeable-running capability is set to "true" • configuration mode is set to "mixed-mode"
	<target> **<startup/>** </target>	No	—
	<target> **<candidate/>** </target>	Yes	—
	<default-operation>**merge**</default-operation>	Yes	Default

Table 26 **Protocol Operations and Level of Support in Nokia SR OS NETCONF Servers**

Protocol Operation	Example	Supported	Notes
	<default-operation>**none**</default-operation>	Yes	With the Nokia SR OS YANG modules, an operation of "none" (inherited or direct) on a leaf node that does not exist in the data model causes SR OS to return an error with an <error-tag> value of “data-missing”.
	<default-operation>**replace**</default-operation>	Yes	—
	<test-option>**test-then-set**</test-option>	Yes	—
	<test-option>**set**</test-option>	Yes	—
	<test-option>**test-only**</test-option>	Yes	—
	<error-option>**continue-on-error**</error-option>	No	—
	<error-option>**rollback-on-error**</error-option>	Yes	—
	<error-option>**stop-on-error**</error-option>	Yes	Default. This can be specified but, starting in Release 20.2.R1, behaves the same as a **rollback-on-error**.
	<target> **<configuration-region>...</configuration-region>** </target>	Yes	Optional. Default if not specified is "configure". Can be "li" or "configure". Can only be used with the Nokia SR OS modules.
close-session	<?xml version="1.0" encoding="UTF-8"?> <rpc message-id="101" xmlns="urn:ietf:params:xml:ns:netconf:base:1.0"> <close-session/> </rpc>]]>]]>	Yes	When a session is closed, any locks held are released and the session is terminated. Any pending RPC requests are discarded.

Table 26 **Protocol Operations and Level of Support in Nokia SR OS NETCONF Servers**

Protocol Operation	Example	Supported	Notes
commit	<?xml version="1.0" encoding="UTF-8"?> <rpc message-id="101" xmlns="urn:ietf:params:xml:ns:netconf:base:1.0"> <commit/> </rpc>]]>]]>	Yes	When commit is issued, any configuration stored in the <candidate> datastore is written to the running configuration unless the device is locked by a NETCONF client or <running> datastore is locked by any other NETCONF session. The startup configuration is also written if **system management-interface netconf auto-config-save** is configured.
	<commit>**<confirmed/>**</commit>	Yes	Optional. Can only be used with non-li configurations.
	<commit> <confirmed/> **<confirm-timeout/>** </commit>	Yes	Optional. Can only be used with non-li configurations.
	<commit> <confirmed/> <confirm-timeout/> **<persist/>** </commit>	Yes	Optional. Can only be used with non-li configurations.
	<commit> <confirmed/> <confirm-timeout/> <persist/> **<persist-id/>** </commit>	Yes	Optional. Can only be used with non-li configurations.
	<commit> **<configuration-region>...</configuration-region>** </commit>	Yes	Optional. Default if not specified is "configure". Can be "li" or "configure". Can only be used with the Nokia SR OS modules.

Table 26 **Protocol Operations and Level of Support in Nokia SR OS NETCONF Servers**

Protocol Operation	Example	Supported	Notes
cancel-commit	<?xml version="1.0" encoding="UTF-8"?> <rpc message-id="101" xmlns="urn:ietf:params:xml:ns:netconf:base:1.0"> <cancel-commit/> </rpc>	Yes	Can only be used with non-li configurations.
	<cancel-commit> **<persist-id/>**</cancel-commit>	Yes	Optional. When a <commit> <persist> is used, the value of a <cancel-commit> <persist-id> must be equal to the value used in the <commit> <persist>.
copy-config	<?xml version="1.0" encoding="UTF-8"?> <rpc message-id="101" xmlns:nc="urn:ietf:params:xml:ns:netconf:base:1.0"> <copy-config> <target/> <source/> </copy-config> </rpc>]]>]]>	Yes	The <copy-config> operation is supported for specific combinations of source and target datastores. When the specified <configuration-region> is "li", the <source><running/> to <target><startup/> becomes the only valid <copy-config> combination.
	<target><running/></target>	No	The running datastore cannot be a <target> for a <copy-config>.
	<target><candidate/></target> <source><running/></source>	No	Use <discard-changes>.
	<target><startup/></target> <source><running/></source>	Yes	Equivalent to **admin save.** An index file is also saved in classic management-interface configuration-mode if **persist on** is configured in the BOF.

Table 26 **Protocol Operations and Level of Support in Nokia SR OS NETCONF Servers**

Protocol Operation	Example	Supported	Notes
	<target><url/></target> <source><running/></source>	Yes	Equivalent to **admin save** *file-url*. An index file is also saved in classic management-interface configuration-mode if **persist on** is configured in the BOF.
	<target><candidate/></target> <source><startup/></source>	Yes	—
	<target><url/></target> <source><startup/></source>	Yes	Supported if both source and target are not remote URLs. Only configuration changes are saved; for example, an index file is not saved in classic management-interface configuration-mode even if **persist on** is configured in the BOF.
	<target><running/></target> <source><startup/></source>	No	—
	<target><startup/></target> <source><candidate/></source>	Yes	—
	<target><url/></target> <source><candidate/></source>	Yes	—
	<target><running/></target> <source><candidate/></source	No	Use <commit> instead.
	<target><running/></target> <source><url/></source>	No	—
	<target><candidate/></target> <source><url/></source>	Yes	—

Table 26 **Protocol Operations and Level of Support in Nokia SR OS NETCONF Servers**

Protocol Operation	Example	Supported	Notes
	<target><startup/></target> <source><url/></source>	Yes	Supported if both source and target are not remote URLs. Only configuration changes are saved; for example, an index file is not saved in classic management-interface configuration-mode even if **persist on** is configured in the BOF.
	<target><url/></target> <source><url/></source>	Yes	Supported if both source and target are not remote URLs.
	<target><config/></target>	No	—
	<target><candidate/></target> <source><config/></source>	Yes	—
	<target><startup/></target> <source><config/></source>	Yes	—
	<target><url/></target> <source><config/></source>	Yes	—
	<target> **<configuration-region>...</configuration-region>** </target>	Yes	Optional. Default if not specified is "configure”. Can be "li" or "configure". Can only be used with the Nokia SR OS modules.
	<source> **<configuration-region>...</configuration-region>** </source>	Yes	Optional. Default if not specified is "configure". Can be "li" or "configure". Can only be used with the Nokia SR OS modules.

Table 26 **Protocol Operations and Level of Support in Nokia SR OS NETCONF Servers**

Protocol Operation	Example	Supported	Notes
kill-session	<?xml version="1.0" encoding="UTF-8"?> <rpc message-id="101" xmlns="urn:ietf:params:xml:ns:netconf:base:1.0"> <kill-session> **<session-id/>** </kill-session> </rpc>]]>]]>	Yes	A NETCONF session cannot kill itself. A NETCONF session cannot kill a non-NETCONF session. When a session is killed, any operations pending in that session are discarded. Any locks held by that session are released. Only a NETCONF user that belongs to a profile with base-op-authorization/kill-session enabled, can kill a NETCONF session.
lock	<?xml version="1.0" encoding="UTF-8"?> <rpc message-id="101" xmlns="urn:ietf:params:xml:ns:netconf:base:1.0"> <lock> <target/> </lock> </rpc>]]>]]>	Yes	Only a NETCONF user that belongs to a profile with base-op-authorization/lock enabled, can lock a datastore.
	<target>**<candidate/>**</target>	Yes	Locking the <candidate> datastore implicitly locks both the <running> and <candidate> datastores.
	<target>**<running/>**</target>	Yes	Locking the <running> datastore locks both the <running> and <candidate> datastores.
	<target>**<startup/>**</target>	No	—
	<target>**<url/>**</target>	No	—

Table 26 **Protocol Operations and Level of Support in Nokia SR OS NETCONF Servers**

Protocol Operation	Example	Supported	Notes
	<target> **<configuration-region>...</configuration-region>** </target>	Yes	Optional. Default if not specified is "configure". Can be "li" or "configure". Can only be used with the Nokia SR OS modules.
unlock	<?xml version="1.0" encoding="UTF-8"?> <rpc message-id="101" xmlns="urn:ietf:params:xml:ns:netconf:base:1.0"> <unlock> **<target/>** </unlock> </rpc>]]>]]>	Yes	Only a NETCONF user that belongs to a profile with base-op-authorization/lock enabled, can unlock a datastore. A datastore lock is unlocked when: - using <unlock> - disconnecting a NETCONF session (from CLI using the **admin disconnect** command, by using Ctrl-c, by performing <kill-session>, or by performing <close-session>) Upon unlocking/ disconnecting a NETCONF session that had acquired a datastore lock, SR OS: - releases the lock - discards any “uncommitted” changes
	<target>**<candidate/>**</target>	Yes	—
	<target>**<running/>**</target>	Yes	—
	<target>**<startup/>**</target>	No	—
	<target>**<url/>**</target>	No	—
	<target> **<configuration-region>...</configuration-region>** </target>	Yes	Optional. Default if not specified is "configure". Can be "li" or "configure". Can only be used with the Nokia SR OS modules.

Table 26 **Protocol Operations and Level of Support in Nokia SR OS NETCONF Servers**

Protocol Operation	Example	Supported	Notes
validate	<?xml version="1.0" encoding="UTF-8"?> <rpc message-id="101" xmlns:nc="urn:ietf:params:xml:ns:netconf:base:1.0"> <validate> <source/> </validate> </rpc>]]>]]>	Yes	XML content layer only. Only syntax validation is performed. If more than one error exists, SR OS returns multiple errors. No semantic validation is performed.
	<source>**<candidate/>**</source>	Yes	—
	<source>**<running/>**</source>	Yes	—
	<source>**<startup/>**</source>	No	—
	<source>**<url/>**</source>	No	—
	<source>**<config/>**</source>	Yes	—
	<source>**<configuration-region>...</configuration-region>**</source>	Yes	Optional. Default if not specified is "configure". Can be "li" or "configure". Can only be used with the Nokia SR OS modules.
delete-config	<?xml version="1.0" encoding="UTF-8"?> <rpc message-id="101" xmlns:nc="urn:ietf:params:xml:ns:netconf:base:1.0"> <delete-config> <target/> </delete-config> </rpc>]]>]]>	Yes	—
	<target>**<startup/>**</target>	Yes	—
	<target>**<url/>**</target>	Yes	—
	<target>**<running/>**</target>	No	—
	<target>**<candidate/>**</target>	No	—

Table 26 **Protocol Operations and Level of Support in Nokia SR OS NETCONF Servers**

Protocol Operation	Example	Supported	Notes
discard-changes	<?xml version="1.0" encoding="UTF-8"?> <rpc message-id="101" xmlns:nc="urn:ietf:params:xml:ns:netconf:base:1.0"> <discard-changes/> </rpc>]]>]]>	Yes	—
	<discard-changes> **<configuration-region>...</configuration-region>** </discard-changes>	Yes	Optional. Default if not specified is "configure". Can be "li" or "configure". Can only be used with the Nokia SR OS modules.
create-subscription	<?xml version="1.0" encoding="UTF-8"?> <rpc message-id="101" xmlns:nc="urn:ietf:params:xml:ns:netconf:base:1.0"> <create-subscription/> </rpc>]]>]]>	Yes	—
	<create-subscription>**<stream/>**</create-subscription>	Yes	Optional
	<create-subscription>**<startTime/>**</create-subscription>	Yes	Optional
	<create-subscription>**<stopTime/>**</create-subscription>	Yes	Optional
	<create-subscription>**<filter/>**</create-subscription>	No	Optional
get-schema	<?xml version="1.0" encoding="UTF-8"?> <rpc message-id="101" xmlns="urn:ietf:params:xml:ns:netconf:base:1.0"> <get-schema xmlns="urn:ietf:params:xml:ns:yang:ietf-netconf-monitoring"> </get-schema> </rpc>]]>]]>	Yes	—

Table 26 **Protocol Operations and Level of Support in Nokia SR OS NETCONF Servers**

Protocol Operation	Example	Supported	Notes
	<get-schema><**identifier/**></get-schema>	Yes	Mandatory
	<get-schema><**version/**></get-schema>	Yes	Optional
	<get-schema><**format/**></get-schema>	Yes	Optional
get-data	<?xml version="1.0" encoding="UTF-8"?> <rpc message-id="101" xmlns="urn:ietf:params:xml:ns:netconf:base:1.0"> <get-data> </get-data> </rpc>]]>]]>	Yes	—
	<get-data><**datastore/**></get-data>	Yes	Mandatory
	<get-data><**subtree-filter/**></get-data>	Yes	Optional
	<get-data><**max-depth/**></get-data>	Yes	Optional
	<get-data><**config-filter/**></get-data>	No	Optional
	<get-data><**origin-filters/**></get-data>	No	Optional
	<get-data><**with-origin/**></get-data>	No	Optional
	<get-data><**with-defaults/**></get-data>	Yes	Optional
	<get-data><**xpath-filter/**></get-data>	No	Optional

5.2.3.1 <get>

A <get> request can retrieve both the configuration and state data from the "urn:nokia.com:sros:ns:yang:sr:conf" namespace (the Nokia SR OS YANG modules).

If any nodes from the configure tree are included in a <get> request filter then, at minimum, the <configure> tag must contain a namespace. If the namespace is not specified, SR OS returns an error.

A <get> request is first analyzed for syntax errors before any execution starts. If a syntax error is found, a single global <rpc-error> for the entire request is sent in the reply.

Responses are provided for each item in the request until the first item with an error is found. The item with an error has a <response> tag containing some error information, followed by an <rpc-error> tag (and sub-tags). The reply is then returned, and subsequent items are not executed.

The <rpc-error> for an individual item (that is, for a non-syntax error) is after the </response> information and not inside the <response>.

See the <get-config> section for details about subtree filtering support.

To retrieve LI configurations, the “li” <configuration-region> must be specified within the <get> RPC. For example:

```
<get>
   <configuration-region>li</configuration-region>
    <filter>
        <li xmlns="urn:nokia.com:sros:ns:yang:sr:li-conf">
       </li>
    </filter>
</get>
```

When a <configuration-region> is not specified, by default the <configuration-region> is considered to be “configure” (that is, the main non-LI configuration region).

When a mismatched namespace or <configuration-region> combination is specified, SR OS returns an empty <data>.

See Table 26 for more details.

See the following sections for examples of <get> request and response messages:

- Example: Namespace Specified in <configure> Tag
- Example: Namespace Error in <configure> Tag
- Example: Namespace Specified in <state> Tag
- Example: Namespace Error in <state> Tag

5.2.3.1.1 Example: Namespace Specified in <configure> Tag

The following shows a <configure> tag that contains a namespace.

```
<?xml version="1.0" encoding="UTF-8"?>
<rpc message-id="101" xmlns="urn:ietf:params:xml:ns:netconf:base:1.0">
<get>
    <filter>
        <configure xmlns="urn:nokia.com:sros:ns:yang:sr:conf">
        </configure>
    </filter>
</get>
```

```
</rpc>
]]>]]>
```

The following example shows the reply, which returns no errors.

```
<?xml version="1.0" encoding="UTF-8"?>
<rpc-reply message-id="101" xmlns="urn:ietf:params:xml:ns:netconf:base:1.0">
    <data>
        <configure xmlns="urn:nokia.com:sros:ns:yang:sr:conf">
        </configure>
    </data>
</rpc-reply>
]]>]]>
```

5.2.3.1.2 Example: Namespace Error in <configure> Tag

The following example shows a <configure> tag that does not contain a namespace.

```
<?xml version="1.0" encoding="UTF-8"?>
<rpc message-id="101" xmlns="urn:ietf:params:xml:ns:netconf:base:1.0">
<get>
    <filter>
        <configure>
            <python xmlns="urn:nokia.com:sros:ns:yang:sr:conf-python">
            </python>
        </configure>
    </filter>
</get>
</rpc>
]]>]]>
```

The following example shows the reply, which returns SR OS errors.

```
<?xml version="1.0" encoding="UTF-8"?>
<rpc-reply message-id="101" xmlns="urn:ietf:params:xml:ns:netconf:base:1.0">
    <rpc-error>
        <error-type>protocol</error-type>
        <error-tag>bad-element</error-tag>
        <error-severity>error</error-severity>
        <error-message>
            Element is not valid in the specified context.
        </error-message>
        <error-info>
            <bad-element>configure</bad-element>
        </error-info>
    </rpc-error>
</rpc-reply>
]]>]]>
```

5.2.3.1.3 Example: Namespace Specified in <state> Tag

The following example shows a <state> tag that contains a namespace.

```
<?xml version="1.0" encoding="UTF-8"?>
<rpc message-id="101" xmlns="urn:ietf:params:xml:ns:netconf:base:1.0">
<get>
    <filter>
        <state xmlns="urn:nokia.com:sros:ns:yang:sr:state">
        </state>
    </filter>
</get>
</rpc>
]]>]]>
```

The following example shows the reply, which returns no errors.

```
<?xml version="1.0" encoding="UTF-8"?>
<rpc-reply message-id="101" xmlns="urn:ietf:params:xml:ns:netconf:base:1.0">
    <data>
        <state xmlns="urn:nokia.com:sros:ns:yang:sr:state">
...
...
        </state>
    </data>

</rpc-reply>
]]>]]>
```

5.2.3.1.4 Example: Namespace Error in <state> Tag

The following example shows a <state> tag that does not contain a namespace.

```
<?xml version="1.0" encoding="UTF-8"?>
<rpc message-id="101" xmlns="urn:ietf:params:xml:ns:netconf:base:1.0">
<get>
    <filter>
        <state>
        </state>
    </filter>
</get>
</rpc>
]]>]]>
```

The following example shows the reply, which returns errors.

```
<?xml version="1.0" encoding="UTF-8"?>
<rpc-reply message-id="101" xmlns="urn:ietf:params:xml:ns:netconf:base:1.0">
    <rpc-error>
        <error-type>protocol</error-type>
        <error-tag>bad-element</error-tag>
        <error-severity>error</error-severity>
        <error-message>
```

```
            Element is not valid in the specified context.
        </error-message>
        <error-info>
            <bad-element>state</bad-element>
        </error-info>
    </rpc-error>
</rpc-reply>
]]>]]>
```

5.2.3.2 <get-config>

A <get-config> operation is supported on the <candidate> datastore.

The <get-config> requests on the <candidate> datastore return only XML-formatted content. Using the <candidate> datastore, if no filter is specified, SR OS returns the Nokia SR OS configurations only.

Using the Nokia SR OS YANG modules, <get-config> requests that specify a nonexistent list node or presence container result in an <rpc-error> response.

Using the “report-all” value with the <with-defaults> tag (RFC 6243) in an XML-content layer <get-config> returns the equivalent of the CLI command info detail (the returned data includes attributes that are set to their default values).

Subtree filtering is supported for <get-config> (and <get> requests). The subtree filtering behavior is as follows:

- Containment nodes are supported (section 6.2.3 of RFC 6241). Nodes that contain children nodes (containers) can be used for subtree filtering. See Example: Containment Node for more information.
- Attribute match expressions (section 6.2.2 of RFC 6241) are not supported.
- Selection nodes are supported (section 6.2.4 of RFC 6241). Empty leaf nodes and list name nodes can be used as selection nodes. A selection node that is a list and does not have a key specified is supported. See Example: List with Non-key Leaf Specified as Selection Node for more information.
- Content match nodes are supported (section 6.2.5 of RFC 6241). Content match nodes that are leafs but not keys are also supported. See Example: Non-key Leaf Specified as a Content Match Node for more information.

The <get-config> operation returns data nodes that were set by a client to their default values for the Nokia SR OS models (the “explicit” mode as per RFC 6243).

To retrieve LI configurations, the “li” <configuration-region> must be specified within the <get-config> <source>. For example:

```
<get-config>
```

```
  <source>
    <configuration-region>li</configuration-region>
    <candidate/>
  </source>
    <filter>
       …
    </filter>
</get-config>
```

Alternatively, the <source> can be specified in the format of “configuration-region”-”datastore”. For example:

```
<get-config>
  <source>
    <li-candidate/>
  </source>
    <filter>
       …
    </filter>
</get-config>
```

When both the <configuration-region> and the “configuration-region”-”datastore” format are used, SR OS applies the last tag used in the XML request. For example:

```
<get-config>
<source>
<configuration-region>configure</configuration-region>
<li-candidate/>
</source>
<filter>
…
</filter>
</get-config>
```

In the preceding example, the <get-config> is used to retrieve the “li” configuration data from the “li” candidate datastore.

When a mismatched namespace or <configuration-region> combination is specified, SR OS returns an empty <data>.

See Table 26 for more details.

The following sections contain examples of <get-config> request and response messages.

- Example: Reply with Defaults
- Example: Reply Without Default Values
- Example: Containment Node
- Example: List Without a Key Specified
- Example: List with Non-key Leaf Specified as Selection Node

- Example: Non-key Leaf Specified as a Content Match Node
- Example: Content Match Node on a List Key
- Example: Content Match Node on a Leaf-list

5.2.3.2.1 Example: Reply with Defaults

The following example shows the use of <with-defaults> with a value of "report-all".

```
<?xml version="1.0" encoding="UTF-8"?>
<rpc message-id="101" xmlns="urn:ietf:params:xml:ns:netconf:base:1.0">
<get-config>
    <source>
        <candidate/>
    </source>
    <filter>
        <configure xmlns="urn:nokia.com:sros:ns:yang:sr:conf">
            <system>
                <security>
                    <cpm-filter>
                        <ipv6-filter>
                        </ipv6-filter>
                    </cpm-filter>
                </security>
            </system>
        </configure>
    </filter>
<with-defaults xmlns="urn:ietf:params:xml:ns:yang:ietf-netconf-with-defaults">
    report-all
</with-defaults>
</get-config>
</rpc>
]]>]]>
```

The following example shows the reply, which returns all attributes, even those with with default values.

```
<?xml version="1.0" encoding="UTF-8"?>
<rpc-reply message-id="101" xmlns="urn:ietf:params:xml:ns:netconf:base:1.0">
    <data>
        <configure xmlns="urn:nokia.com:sros:ns:yang:sr:conf">
            <system>
                <security>
                    <cpm-filter>
                        <ipv6-filter>
                            <admin-state>disable</admin-state>
                        </ipv6-filter>
                    </cpm-filter>
                </security>
            </system>
        </configure>
    </data>
</rpc-reply>
]]>]]>
```

5.2.3.2.2 Example: Reply Without Default Values

The following example shows a <get-config> request using <with-defaults>.

```
<?xml version="1.0" encoding="UTF-8"?>
<rpc message-id="101" xmlns="urn:ietf:params:xml:ns:netconf:base:1.0">
<get-config>
    <source>
        <candidate/>
    </source>
    <filter>
        <configure xmlns="urn:nokia.com:sros:ns:yang:sr:conf">
            <system>
                <security>
                <cpm-filter>
                    <ipv6-filter>
                    </ipv6-filter>
                    </cpm-filter>
                </security>
            </system>
        </configure>
    </filter>
</get-config>
</rpc>
]]>]]>
```

The following output shows the reply, which does not return attributes with default values.

```
<?xml version="1.0" encoding="UTF-8"?>
<rpc message-id="101" xmlns="urn:ietf:params:xml:ns:netconf:base:1.0">
<get-config>
    <source>
        <candidate/>
    </source>
    <filter>
        <configure xmlns="urn:nokia.com:sros:ns:yang:sr:conf">
            <system>
                <security>
                <cpm-filter>
                    <ipv6-filter>
                    </ipv6-filter>
                    </cpm-filter>
                </security>
            </system>
        </configure>
    </filter>
</get-config>
</rpc>
]]>]]>
```

5.2.3.2.3 Example: Containment Node

The following example shows a containment node.

```
<?xml version="1.0" encoding="UTF-8"?>
<rpc message-id="101" xmlns="urn:ietf:params:xml:ns:netconf:base:1.0">
<get-config>
    <source><running/></source>
         <filter>
             <configure xmlns="urn:nokia.com:sros:ns:yang:sr:conf">
                 <router/>
             </configure>
         </filter>
</get-config>
</rpc>
]]>]]>
```

5.2.3.2.4 Example: List Without a Key Specified

The following example shows a selection node that is a list and does not have a key specified.

```
<?xml version="1.0" encoding="UTF-8"?>
<rpc message-
id="101" xmlns="urn:ietf:params:xml:ns:netconf:base:1.0" xmlns:alu="urn:nokia.c
om:sros:ns:yang:sr:conf">
    <get-config>
        <source><running/></source>
        <filter>
            <configure xmlns="urn:nokia.com:sros:ns:yang:sr:conf">
                <router>
                    <interface>
                    </interface>
                </router>
            </configure>
        </filter>
    </get-config>
</rpc>
]]>]]>
```

5.2.3.2.5 Example: List with Non-key Leaf Specified as Selection Node

The following example shows a list with a non-key leaf specified as a selection node. Keys are returned as well.

```
<?xml version="1.0" encoding="UTF-8"?>
<rpc message-
id="101" xmlns="urn:ietf:params:xml:ns:netconf:base:1.0" xmlns:alu="urn:nokia.c
om:sros:ns:yang:sr:conf">
    <get-config>
        <source><running/></source>
        <filter>
            <configure xmlns="urn:nokia.com:sros:ns:yang:sr:conf">
```

```
                <router>
                    <interface>
                        <admin-state/>
                    </interface>
                </router>
            </configure>
        </filter>
    </get-config>
</rpc>
]]>]]>
```

5.2.3.2.6 Example: Non-key Leaf Specified as a Content Match Node

The following example shows a non-key leaf specified as a content match node.

```
<?xml version="1.0" encoding="UTF-8"?>
<rpc message-
id="101" xmlns="urn:ietf:params:xml:ns:netconf:base:1.0" xmlns:alu="urn:nokia.c
om:sros:ns:yang:sr:conf">
    <get-config>
        <source><running/></source>
        <filter>
            <configure xmlns="urn:nokia.com:sros:ns:yang:sr:conf">
                <router>
                    <interface>
                        <admin-state>disable</admin-state>
                    </interface>
                </router>
            </configure>
        </filter>
    </get-config>
</rpc>
]]>]]>
```

5.2.3.2.7 Example: Content Match Node on a List Key

Multiple key leafs for the same key cannot be requested inside the same instance of the list name node. Instead, each key value must be inside its own instance of the list name node; for example, `<interface><interface-name>abc</interface-name></interface><interface> <interface-name>def</interface-name></interface>`.

The following example shows a content match node on a list key.

```
<?xml version="1.0" encoding="UTF-8"?>
<rpc message-
id="101" xmlns="urn:ietf:params:xml:ns:netconf:base:1.0" xmlns:alu="urn:nokia.c
om:sros:ns:yang:sr:conf">
    <get-config>
        <source><running/></source>
        <filter>
```

```
            <configure xmlns="urn:nokia.com:sros:ns:yang:sr:conf">
                <router>
                    <interface>
                        <interface-name>Test</interface-name>
                    </interface>
                </router>
            </configure>
        </filter>
    </get-config>
</rpc>
]]>]]>
```

5.2.3.2.8 Example: Content Match Node on a Leaf-list

A content match node can be performed on a leaf-list but SR OS requires that all of the leaf-list elements and nodes must be specified. The full configuration (equivalent to the classic CLI command **admin display-config** or the MD-CLI command **admin show configuration**) can be obtained using a <get-config> request both when a <filter> tag is not present and when the <configure> tag is present inside a <filter> tag.

The following example shows a content match node on a lead-list when the <filter> tag is not present.

```
<?xml version="1.0" encoding="UTF-8"?>
<rpc message-id="101" xmlns="urn:ietf:params:xml:ns:netconf:base:1.0">
    <get-config>
        <source><running/></source>
    </get-config>
</rpc>
]]>]]>

<?xml version="1.0" encoding="UTF-8"?>
<rpc message-id="101" xmlns="urn:ietf:params:xml:ns:netconf:base:1.0">
    <get-config>
        <source><candidate/></source>
    </get-config>
</rpc>
]]>]]>
```

The following example shows a content match node on a lead-list when only the <configure> tag is present inside the <filter> tag.

```
<?xml version="1.0" encoding="UTF-8"?>
<rpc message-id="101" xmlns="urn:ietf:params:xml:ns:netconf:base:1.0">
    <get-config>
        <source><running/></source>
        <filter>
            <configure xmlns="urn:nokia.com:sros:ns:yang:sr:conf"/>
        </filter>
    </get-config>
</rpc>
```

```
]]>]]>

<?xml version="1.0" encoding="UTF-8"?>
<rpc message-id="101" xmlns="urn:ietf:params:xml:ns:netconf:base:1.0">
    <get-config>
        <source><candidate/></source>
        <filter>
            <configure xmlns="urn:nokia.com:sros:ns:yang:sr:conf"/>
        </filter>
    </get-config>
</rpc>
]]>]]>
```

5.2.3.3 <edit-config>

An <edit-config> operation is supported on the <candidate> datastore. The <edit-config> requests that specify the <running> datastore as a target while using the "urn:nokia.com:sros:ns:yang:sr:conf" namespace (the Nokia SR OS YANG modules) result in an error response.

The <edit-config> requests to the <candidate> datastore only result in XML-formatted content.

There is an internal "implicit" lock on the <running> datastore that has a scope of all configuration commands in SR OS (not just the "urn:nokia.com:sros:ns:yang:sr:conf" namespace). The following actions affect the "implicit" lock:

- The first NETCONF <edit-config> on a global <candidate> datastore triggers the "implicit" lock.
- The completion of a NETCONF <commit> releases the "implicit" lock.
- The NETCONF <discard-changes> operation releases the "implicit" lock.

The following scenarios are impacted when the "implicit" lock is in place:

- A classic CLI command is blocked while in model-driven configuration mode, and SR OS returns an error.
- An SNMP set request is blocked while in model-driven configuration mode, and SR OS returns an error.
- A NETCONF session attempting an <edit-config> using the Alcatel-Lucent Base-R13 SR OS data models on the <running> datastore is blocked, and SR OS replies with an error. The <error-info> element includes the <session-id> of the lock owner. For more details, see NETCONF Using the Legacy Alcatel-Lucent Base-R13 SR OS YANG Modules.

One or more <edit-config> requests can be performed on the <candidate> datastore before the changes are committed or discarded.

The supported <edit-config> operation attribute values are listed in Table 27.

Table 27 **<edit-config> Operation Attribute Values**

Command	Notes
urn:nokia.com:sros:ns:yang:sr:conf namespace Nokia SR OS YANG modules	
merge (Nokia SR OS modules)	Supported
remove (Nokia SR OS modules)	A <remove> operation removes the deleted configuration and returns it to the default value. A <remove> operation automatically removes all child objects of a deleted object (leaves, lists, containers, and so on). Explicit shutdown of the object being removed (or any child) is not required and results in an error if a merge operation is specified on a tag that inherits a <remove> operation. A <remove> operation is allowed on non-presence containers. The non-presence container and all of its children are removed (for example, a non-presence container with no child nodes is not displayed in a <get> or <get-config> reply). A <remove> operation is allowed on an object where all child branches and dependencies are automatically removed (but the <remove> operation fails if any outside objects refer to the object being removed). A <remove> operation is allowed on a <shutdown/> leaf (which returns it to its default value). A <remove> operation is allowed on a non-Boolean leaf. Upon specifying a <remove> operation on a node where none of its children belong to the urn:nokia.com:sros:ns:yang:sr:conf namespace (the Nokia SR OS YANG modules), SR OS does not return an error and completes the node removal. A <remove> operation for a leaf, where the request also specifies a value for the leaf, results in an error.

Table 27 **<edit-config> Operation Attribute Values (Continued)**

Command	Notes
delete (Nokia SR OS modules)	SR OS returns an error if a <delete> operation is performed on a list that does not specify a key (that is, an attempt to delete all members of a list). SR OS returns an error if a <delete> operation is performed on a leaf or presence container that is already deleted (or has the default value and the default-handling is trim). SR OS may return an error and may not complete the deletion operation when a <delete> operation is performed on a node where any of its children do not belong to the urn:nokia.com:sros:ns:yang:sr:conf namespace (the Nokia SR OS YANG modules). A <delete> operation removes the deleted configuration and returns it to the default value. A <delete> operation automatically deletes all child objects of a deleted object (leaves, lists, containers, and so on). Explicit shutdown of the object being deleted (or any of its children) is not required and results in an error if a merge operation is specified on a tag that inherits a <delete> operation. A <delete> operation is allowed on non-presence containers. The non-presence container and all of its children are deleted (for example, a non-presence container with no child nodes is not displayed in a <get> or <get-config> reply). A <delete> operation is allowed on an object where all child branches and dependencies are automatically deleted (but the <delete> operation fails if any outside objects refer to the object being deleted). A <delete> operation is allowed on a <shutdown/> leaf (which returns it to its default value). A <delete> operation is allowed on a non-boolean leaf. Upon specifying a <delete> operation on a node where none of its children belong to the urn:nokia.com:sros:ns:yang:sr:conf namespace (the Nokia SR OS YANG modules), SR OS does not return an error and completes the node deletion. A <delete> operation for a leaf, where the request also specifies a value for the leaf, will result in an error.
create (Nokia SR OS modules)	When a <create> operation for a leaf or presence container is performed, SR OS returns an error if the leaf or presence container is being set to the same value (unless the default-handling is trim and the value being set is the default value).
replace (Nokia SR OS modules)	Supported

The <edit-config> operation <default-operation> parameter is supported with the following values:

- replace
- merge
- none
  - In the urn:nokia.com:sros:ns:yang:sr:conf namespace (the Nokia SR OS YANG modules), an operation of "none" (inherited or direct) on a leaf node that does not exist in the data model causes SR OS to return an error with an <error-tag> value of data-missing.

For <delete> and <remove> operations in the Nokia SR OS namespace, the SR OS NETCONF server will recursively unwind any children of the node being deleted or removed first before removing the node. The deepest child branch of the request is examined first, and any leaves are processed, after which the server works backwards out of the deepest branches back up to the object where the delete operation was specified.

The following applies to the urn:nokia.com:sros:ns:yang:sr:conf namespace (the Nokia SR OS YANG modules):

- SR OS returns an error if an explicitly defined <edit-config> operation (such as "delete") is specified on a "key" leaf.
- The "operation" attribute is inherited from the parent node if not explicitly specified (similar to namespaces). If no parent node is available, the "default-operation" value is used. This means that the "operation" attribute has a "scope" that it applies to the nested nodes until it is redefined.

See Example: Application of Default Operation Value for Parent and Child Nodes and Example: Exceptions to the Default Operation Handling for more information.

The following scenarios simplify “operation" inheritance, where the first line in each scenario represents the operation value of the parent node and the following lines represent the possible operation values for the child nodes and the SR OS behavior in each case:

- Create

  Create/Merge: SR OS processes the request, which succeeds or fails based on the behavior of this operation.

  Delete/Remove: SR OS returns an error.
- Merge

  Create/Merge/Delete/Remove: SR OS processes the request, which succeeds or fails based on the behavior of this operation.
- Delete/Remove

  Create/Merge: SR OS returns an error.

Delete/Remove: SR OS processes the request, which succeeds or fails based on the behavior of this operation.

The <error-option> is supported. SR OS implements the rollback-on-error behavior at all times, when:

- the error-option is not specified
- the error-option is specified and set to either stop-on-error or rollback-on-error

As per RFC 6020, *YANG - A Data Modeling Language for the Network Configuration Protocol (NETCONF)*, the “insert” and “value” attributes are supported with user-ordered leaf-lists to insert or move a user-ordered leaf-list entry in the candidate datastore.

As per RFC 6020, the “insert” and “key” attributes are supported with user-ordered lists to insert or move a user-ordered list entry in the candidate datastore.

With a NETCONF <edit-config> RPC, SR OS authorizes all configuration changes in the <candidate> datastore; that is, it checks the YANG tree and authorizes every changed managed object (MO).

The deletion of a container results in the deletion of any children containers that are authorized for deletion, as well as their contents. Children containers that are not authorized for deletion, as well as their contents, are retained. For example, upon deletion of **configure system, configure system security** is not deleted because the deletion of that child container is not authorized.

Note: A “no change” for a value does not require authorization. Therefore, it is possible to execute a non-authorized command if there is no change in value.

For example, when a user is not authorized to change **access li,** but attempts to change it for another a user who already has **access li**, SR OS allows that action because there is no change in value.

To edit LI configurations, the “li” <configuration-region> must be specified within the <edit-config> <target>. For example:

```
<edit-config>
  <target>
    <configuration-region>li</configuration-region>
    <candidate/>
  </target>
  <config>
    <!-- place LI configuration changes here -->
  </config>
</edit-config>
```

Alternatively, the <target> can be specified in the format of “configuration-region”-”datastore”. For example:

```
<edit-config>
  <target>
    <li-candidate/>
  </target>
  <config>
    <!-- place LI configuration changes here -->
  </config>
</edit-config>
```

When both the <configuration-region> and the “configuration-region”-”datastore” format are used, SR OS applies the last tag used in the XML request. For example:

```
<edit-config>
<target>
    <configuration-region>configure</configuration-region>
    <li-candidate/>
</target>
<config>
    <!-- place LI configuration changes here -->
</config>
</edit-config>
```

When a mismatched namespace or <configuration-region> combination is specified, SR OS returns an error.

The <edit-config> RPC can only be used to push LI configuration changes if all of the following conditions are true:

- The NETCONF user is a LI user.
- The NETCONF session has an exclusive lock on the LI configuration region and <candidate> datastore.
- The specified <configuration-region> is “li”.
- The YANG modules that are used are the Nokia SR OS YANG modules.

If any of the preceding conditions are false, SR OS returns an error.

See Table 26 for more details.

See the following sections for examples of <edit-config> request and response messages:

- Example: <running> Datastore with the “urn:nokia.com:sros:ns:yang:sr:conf” Namespace
- Example: Application of Default Operation Value for Parent and Child Nodes
- Example: Exceptions to the Default Operation Handling

5.2.3.3.1 Example: <running> Datastore with the “urn:nokia.com:sros:ns:yang:sr:conf” Namespace

The following example shows the use of the <running> datastore with the “urn:nokia.com/sros:ns:yang:sr:conf” namespace.

```
<?xml version="1.0" encoding="UTF-8"?>
<rpc message-id="101" xmlns="urn:ietf:params:xml:ns:netconf:base:1.0">
    <edit-config>
        <target><running/></target>
        <config>
            <configure xmlns="urn:nokia.com:sros:ns:yang:sr:conf">
                <python>
                    <python-script>
                        <script-name>testing</script-name>
                    </python-script>
                </python>
            </configure>
        </config>
    </edit-config>
</rpc>
]]>]]>
```

The following example shows the reply, which returns SR OS errors.

```
<?xml version="1.0" encoding="UTF-8"?>
<rpc-reply message-id="101" xmlns="urn:ietf:params:xml:ns:netconf:base:1.0">
    <rpc-error>
        <error-type>protocol</error-type>
        <error-tag>operation-not-supported</error-tag>
        <error-severity>error</error-severity>
        <error-message>
            writable-running capability is not supported
        </error-message>
        <error-info>
            <bad-element>running</bad-element>
        </error-info>
    </rpc-error>
</rpc-reply>
]]>]]>
```

5.2.3.3.2 Example: Application of Default Operation Value for Parent and Child Nodes

The following example shows that the default (operation=”merge”) applies to all parent and child nodes.

```
<?xml version="1.0" encoding="UTF-8"?>
<rpc message-id="101" xmlns="urn:ietf:params:xml:ns:netconf:base:1.0">
<edit-config>
    <target><candidate/></target>
    <config>
        <configure xmlns="urn:nokia.com:sros:ns:yang:sr:conf">
```

```
            <service>
                <epipe>
                    <service-name>CustDoc</service-name>
                    <customer>1</customer>
                    <description>Local epipe</description>
                </epipe>
            </service>
           </configure>
    </config>
</edit-config>
</rpc>
]]>]]>
```

The following example shows the reply.

```
<?xml version="1.0" encoding="UTF-8"?>
<rpc-reply message-id="101" xmlns="urn:ietf:params:xml:ns:netconf:base:1.0">
  <ok/>
</rpc-reply>
]]>]]>
```

5.2.3.3.3 Example: Exceptions to the Default Operation Handling

The following example shows that the default (operation=”merge”) applies to all parent and child nodes except for <description>, which has a (operation=”delete”).

```
 <?xml version="1.0" encoding="UTF-8"?>
<rpc message-id="101" xmlns="urn:ietf:params:xml:ns:netconf:base:1.0">
<edit-config>
    <target><candidate/></target>
    <config>
        <configure xmlns="urn:nokia.com:sros:ns:yang:sr:conf">
            <service>
                <epipe>
                    <service-name>CustDoc</service-name>
                    <customer>1</customer>
                    <description operation="remove">Local epipe</description>
                </epipe>
            </service>
        </configure>
     </config>
</edit-config>
</rpc>
]]>]]>
```

The following example shows the reply.

```
<?xml version="1.0" encoding="UTF-8"?>
<rpc-reply message-id="101" xmlns="urn:ietf:params:xml:ns:netconf:base:1.0">
  <ok/>
</rpc-reply>
]]>]]>
```

5.2.3.4 <copy-config>

To <copy-config> LI configurations, the “li” <configuration-region> must be specified within the <copy-config> RPC <source> and <target>. When the <configuration-region> is “li”, SR OS can only <copy-config> from the <running> datastore to the <startup> datastore. For example:

```
<copy-config>
  <target>
    <configuration-region>li</configuration-region>
    <startup/>
  </target>
  <source>
    <configuration-region>li</configuration-region>
    <running/>
  </source>
</copy-config>
```

<copy-config> between datastores from different <configuration-region> is not allowed. Mismatching the source or target <configuration-region> causes SR OS to return an error.

Alternatively, the <target>/<source> can be specified in the format of “configuration-region”-”datastore”. For example:

```
<copy-config>
  <target><li-startup/></target>
  <source><li-running/></source>
</copy-config>
```

When both the <configuration-region> and the “configuration-region”-”datastore” format are used, SR OS applies the last tag used in the XML request. For example:

```
<copy-config>
  <target>
    <configuration-region>configure</configuration-region>
    <li-startup/>
  </target>
  <source>
    <configuration-region>configure</configuration-region>
    <li-running/>
  </source>
</copy-config>
```

In the preceding example, the <copy-config> is used to copy the configuration data from the “li” <running> datastore to the “li” <startup> datastore.

See Table 26 for more details.

5.2.3.5 <delete-config>

See Table 26 for more details.

5.2.3.6 <lock>

Taking the <candidate> datastore lock is equivalent to starting a CLI exclusive session. A NETCONF session cannot take the <candidate> datastore lock if there were any uncommitted configuration changes in the <candidate> datastore.

It is recommended that a NETCONF session should always take the <candidate> datastore lock before reading or writing configurations to ensure the <candidate> datastore is not changed by other model-driven sessions. Release the <candidate> datastore lock after all configurations are successfully read or committed.

When either the <running> datastore lock or the <candidate> datastore lock is taken by a NETCONF session:

- no NETCONF session can take the <running> datastore lock
- no NETCONF session can take the <candidate> datastore lock
- no other NETCONF session can do an <edit-config> on the <running> datastore
- no other NETCONF session can do an <edit-config> on the <candidate> datastore
- no other NETCONF session can do a <commit> on the <candidate> datastore
- no other NETCONF session can do a <discard-changes> on the <candidate> datastore
- CLI becomes read-only
- classic CLI **rollback revert** is blocked

A datastore lock is unlocked when disconnecting a NETCONF session (either from the CLI using the **admin disconnect** command, using Ctrl-c, or by performing a <kill-session> / <close-session> operation). Upon disconnecting a NETCONF session that had acquired a datastore lock, SR OS:

- releases the lock
- discards any “uncommitted” changes

Note: The behavior is different if the disconnected NETCONF session was using the global <candidate> datastore and had uncommitted configuration changes. In that case, SR OS keeps the “uncommitted” changes in the global <candidate> datastore.

Timeouts for locks are not supported. No specific admin/tools commands are provided to release the lock without disconnecting the session that holds it, but the session that holds the lock can be administratively disconnected through a CLI command to release the lock.

Using a CLI **show** command, the operator can determine if the <running> datastore is locked, the <candidate> datastore is locked, or both are locked, and the session ID of the session that holds the lock.

From CLI, the operator can configure whether users that belong to a specific profile have permission to lock NETCONF sessions.

An active NETCONF session can be disconnected from the CLI using the session ID. The user can use the show command to find the NETCONF session ID, then use the admin command to disconnect the NETCONF session using the session ID obtained from the show command.

To lock an LI datastore, the “li” <configuration-region> must be specified within the <lock> <target>. For example:

```
<lock>
  <target>
    <configuration-region>li</configuration-region>
    <candidate/>
  </target>
</lock>
```

Alternatively, the <target> can be specified in the format of “configuration-region”-”datastore”. For example:

```
<lock>
  <target>
    <li-candidate/>
  </target>
</lock>
```

When both the <configuration-region> and the “configuration-region”-”datastore” format are used, SR OS applies the last tag used in the XML request. For example:

```
<lock>
  <target>
    <configuration-region>configure</configuration-region>
    <li-candidate/>
  </target>
</lock>
```

In the preceding example, the <lock> is used to lock the “li” <candidate> datastore.

The LI datastores have independent locks from the main configuration datastores.

See Table 26 for more details.

5.2.3.7 <unlock>

Because there is a single lock per datastore regardless of what the scope of that lock is:

- The <running> datastore lock is unlocked by using the <unlock> command only on the <running> datastore. An error results and the lock stays if a different datastore is used with the <unlock> operation.
- The <candidate> datastore lock is unlocked by using the <unlock> command only on the <candidate> datastore. An error results and the lock stays if a different datastore is used with the <unlock> operation.

Performing an <unlock> operation on the <candidate> datastore discards all pending (not committed) <candidate> datastore changes.

To unlock an LI datastore, the “li” <configuration-region> must be specified within the <unlock> <target>. For example:

```
<unlock>
  <target>
    <configuration-region>li</configuration-region>
    <candidate/>
  </target>
</unlock>
```

Alternatively, the <target> can be specified in the format of “configuration-region”-”datastore”. For example:

```
<unlock>
  <target>
    <li-candidate/>
  </target>
</unlock>
```

When both the <configuration-region> and the “configuration-region”-”datastore” format are used, SR OS applies the last tag used in the XML request. For example:

```
<unlock>
  <target>
    <configuration-region>configure</configuration-region>
    <li-candidate/>
  </target>
</unlock>
```

In the preceding example, the <unlock> is used to unlock the “li” <candidate> datastore.

See Table 26 for more details.

5.2.3.8 <commit>

The <commit> command has the following characteristics:

- It represents the equivalent of the CLI command **commit**.
- When a <commit> operation fails and more than one error exists, SR OS returns multiple errors.
- When SR OS is not able to commit all the changes in the <candidate> datastore, SR OS keeps the <running> datastore unchanged.
- When a NETCONF session is disconnected (using the CLI command, Ctrl-c, or <kill-session>) in the middle of a <commit> operation, SR OS keeps the <running> datastore unchanged.
- The persistency of changes made using a <commit> operation is operator-controlled. A copy of the <running> datastore to the <startup> datastore can be automatically performed after each successful <commit> operation. This behavior can be enabled or disabled through a CLI command.
- When some changes exist in the <candidate> datastore (before being committed to the <running> datastore), there are impacts to:
  - a CLI user trying to make immediate changes, as SR OS may block some CLI immediate configurations
  - an SNMP set request, as SR OS may block the request and returns an error
  - an <edit-config> to the <running> datastore using the Alcatel-Lucent Base-R13 SR OS YANG models, as SR OS blocks all <edit-config> requests to the running datastore and returns an error. For more details, see NETCONF Using the Legacy Alcatel-Lucent Base-R13 SR OS YANG Modules.

To commit LI configurations, the “li” <configuration-region> must be specified within the <commit> RPC. For example:

```
<commit>
  <configuration-region>li</configuration-region>
</commit>
```

The <commit> RPC can only be used with LI configuration changes if all of the following conditions are true:

- The NETCONF user is a LI user.

- The NETCONF session has an exclusive lock on the LI configuration region and <candidate> datastore.
- The specified <configuration-region> is “li”.
- The YANG modules used are the Nokia SR OS YANG modules.

If any of the preceding conditions are false, SR OS returns an error.

The :confirmed-commit capability cannot be used with LI configuration changes.

The :confirmed-commit capability is advertised in the SR OS NETCONF server <hello> as:

<capability>urn:ietf:params:netconf:capability:confirmed-commit:1.1</capability>

The :confirmed-commit capability has the following characteristics:

- The capability is not advertised if the operator disables the <candidate> datastore capability using the available SR OS CLI command.
- The parameters listed in Table 28 are supported for the <commit> operation.

Table 28 **Parameters for a <commit> Operation**

Parameter	Description
<confirmed>	Indicates a confirmed <commit> operation.
<confirm-timeout>	Specifies the timeout period for confirmed commit (in seconds). If unspecified, the confirmed commit timeout defaults to 600 seconds (10 minutes).
<persist>	Configures the confirmed commit changes to survive a session termination. It sets a token on the ongoing confirmed commit. If <persist> is not in the confirmed commit operation, any follow-up commit and the confirming commit must be issued on the same session that issued the confirmed commit. If <persist> is in the confirmed commit operation, a follow-up commit and the confirming commit can be on any session. However, they must include a <persist-id> element with a value equal to the value of the <persist> element in the confirmed commit. The <persist> element cannot be changed through a follow-up confirmed commit.
<persist-id>	Issues a follow-up confirmed commit or the confirming commit from any session, using the same token from the <persist> element of the confirmed commit. The <persist-id> element cannot be changed through a follow-up confirmed commit.

- If <persist> was specified in the confirmed commit, the configuration changes are rolled back only if the timeout expires before receiving a confirming commit. The confirming commit must include a <persist-id> tag with a value equal to the value of the <persist> tag that was in the confirmed commit.
- If the NETCONF session that initiated the confirmed commit is closed while waiting for the confirming commit (for example, disconnected), SR OS restores the configuration to its state before the confirmed commit was issued. This is valid only if <persist> was not defined in the confirmed commit.
- If a follow-up confirmed commit is issued before the timer expires, the timer is reset to the new value.
- The confirming commit and the follow-up confirmed commit cannot introduce additional changes to the configuration.
- The <cancel-commit> operation is supported. It can cancel an ongoing confirmed commit (that is, cancel the timer and rollback the changes introduced with the confirmed commit).
- Without the <persist> parameter, the <cancel-commit> operation must be issued on the same session that issued the confirmed commit.
- If the configuration changes involve changing the configuration-mode to classic, a confirmed commit should not be used to commit those configuration changes, as SR OS would switch to classic mode before sending the second commit.

See Table 26 for more details.

5.2.3.9 <discard-changes>

The <discard-changes> operation causes the <candidate> datastore to revert back to match the <running> datastore and discard any uncommitted configuration changes.

To discard LI configuration changes, the “li” <configuration-region> must be specified within the <discard-changes> RPC. For example:

```
<discard-changes>
  <configuration-region>li</configuration-region>
</discard-changes>
```

The <discard-changes> RPC can only be used with LI configuration changes if all of the following conditions are true:

- The NETCONF user is a LI user.
- The NETCONF session has an exclusive lock on the LI configuration region and <candidate> datastore.

- The specified <configuration-region> is “li”.
- The YANG modules used are the Nokia SR OS YANG modules.

If any of the preceding conditions are false, SR OS returns an error.

See Table 26 for more details.

5.2.3.10 <validate>

The :validate capability is supported in the following ways:

- The validate:1.1 and :validate:1.0 capabilities are advertised in the NETCONF server <hello> as:

  <capability>urn:ietf:params:netconf:capability:validate:1.0</capability>

  <capability>urn:ietf:params:netconf:capability:validate:1.1</capability>
- The <validate> operation is not supported for a CLI content layer request. Detection of a <config-format-cli-block> or <oper-data-format-cli-block> tag in a <validate> request will result in an “operation not supported” error response.

To validate LI configurations, the “li” <configuration-region> must be specified within the <validate> <source>. For example:

```
<validate>
  <source>
    <configuration-region>li</configuration-region>
    <candidate/>
  </source>
</validate>
```

Alternatively, the <source> can be specified in the format of “configuration-region”-”datastore”. For example:

```
<validate>
  <source>
    <li-candidate/>
  </source>
</validate>
```

When both the <configuration-region> and the “configuration-region”-”datastore” format are used, SR OS applies the last tag used in the XML request. For example:

```
<validate>
  <source>
    <configuration-region>configure</configuration-region>
    <li-candidate/>
  </source>
</validate>
```

In the preceding example, the <validate> is used on the “li” <candidate> datastore.

See Table 26 for more details.

5.2.3.11 <get-schema>

A <get-schema> operation is supported for explicit schema retrieval using NETCONF. See NETCONF Monitoring for more information.

See Table 26 for more details.

5.2.3.12 <get-data>

A <get-data> operation is similar to the <get-config> operation. When applied to the <running/> or <candidate/> datastores, it returns the same data as the <get-config> operation. A <get-data> operation can be applied to the <intended/> datastore, whereas a <get-config> operation cannot be used on the <intended/> datastore.

See Table 26 for more details.

5.2.4 Datastore and Operation Combinations

Table 29 shows which operations are supported by the Nokia modules and datastore combination.

Table 29 **Datastore and Nokia Modules Combinations**

Operation	Nokia Modules		
	<running>	**<candidate>**	**<intended>**
<edit-config>		✓	
<get-config>	✓	✓	
<get-data>	✓	✓	✓
<get>*	Retrieves both configuration and state data (XML format only)		

Notes:

1. The <running> or <candidate> datastores are not applicable for a <get> operation.
2. Retrieves both configuration and state data (XML format only)

5.2.5 General NETCONF Behavior

Using Ctrl-c in a NETCONF session will immediately terminate the session.

The SR OS NETCONF implementation supports XML namespaces (xmlns).

If an invalid namespace is specified within the client hello message, no error will be returned because the NETCONF server is still waiting for the client to send a valid <hello/>. For further NETCONF requests (without sending a proper hello message), even though correct, SR OS returns an error indicating that “Common base capability not found.”

In the <rpc> element, the allowed XML namespaces are:

- standard NETCONF “urn:ietf:params:xml:ns:netconf:base:1.0” namespace
- NOKIA SR OS namespaces; for example, “urn:nokia.com:sros:ns:yang:sr:conf” namespace or “urn:nokia.com:sros:ns:yang:sr:state” namespace
- SR OS “urn:alcatel-lucent.com:sros:ns:yang:conf-r13” namespace. For more information, see NETCONF Using the Legacy Alcatel-Lucent Base-R13 SR OS YANG Modules.

In the <rpc> element, prefixes are accepted and have to be specified with a valid URI. If an incorrect URI is declared with a prefix, SR OS detects the invalid URI and sends an <rpc-error> response.

If any other XML namespace is declared (or assigned to a prefix) in the <rpc> element, SR OS returns an error.

Any prefix declarations in the rest of the request are ignored and unused. The SR OS NETCONF server puts the correct NETCONF namespace declaration (“urn:ietf:params:xml:ns:netconf:base:1.0”) in all replies.

An <edit-config> request must specify which data model (for example, Nokia SR OS YANG modules) is being used in the top-level <configure> element.

- SR OS accepts a single namespace at the top-level <configure> element. For example:

```
<configure xmlns="urn:nokia.com:sros:ns:yang:sr:conf">
   <system>
        ....
```

- The NETCONF client can declare the namespaces with prefixes at the <rpc> element and use the corresponding prefixes later in the request message <configure/> block.
- SR OS returns an error if the request contains one or more incorrect namespaces.

The chunked framing mechanism is supported in addition to the EOM mechanism. As per RFC 6242, Section 4.1 - Framing Protocol, “[...] If the :base:1.1 capability is advertised by both peers, the chunked framing mechanism (see Section 4.2) is used for the remainder of the NETCONF session. Otherwise, the end-of-message-based mechanism is used.” See Example: Chunked Frame Mechanism for more information.

Handling of default data (for example, **info** vs **info detail**) uses the mechanisms described in RFC 6243. The SR OS NETCONF server supports the “explicit” method as the default for the Nokia SR OS YANG modules. It also supports the “report-all” method.

The advertised capability changes depending on which YANG modules are enabled or disabled in SR OS. For example, when Nokia modules are enabled and all other modules are disabled, the following capability is advertised:

<capability>urn:ietf:params:netconf:capability:with-defaults:1.0?basic-mode=explicit&also-supported=report-all</capability>

A **debug system netconf info** command can be used to dump NETCONF debug message streams.

5.2.5.1 Example: Multiple Use of Standard NETCONF Namespace

The following example shows the standard NETCONF namespace “urn:ietf:params:xml:ns:netconf:base:1.0” is used more than once in the <rpc> element:

```
<?xml version="1.0" encoding="UTF-8"?>
<rpc message-id="101"
xmlns="urn:ietf:params:xml:ns:netconf:base:1.0"
xmlns:alu="urn:ietf:params:xml:ns:netconf:base:1.0">
     <get-config>
       <source> <running/> </source>
```

```
       <filter>
             <configure xmlns="urn:nokia.com:sros:ns:yang:sr:conf">
                    <router>
                       <router-name>Base</router-name>
                       <interface>
                             <interface-name>system</interface-name>
                       </interface>
                    </router>
             </configure>
       </filter>
     </get-config>
</rpc>
]]>]]>
```

In the following reply, the namespace is accepted and no error message is returned.

```
<?xml version="1.0" encoding="UTF-8"?>
<rpc-reply message-id="101" xmlns="urn:ietf:params:xml:ns:netconf:base:1.0"
 xmlns:alu="urn:ietf:params:xml:ns:netconf:base:1.0">
    <data>
        <configure xmlns="urn:nokia.com:sros:ns:yang:sr:conf">
            <router>
                <router-instance>Base</router-instance>
                <interface>
                    <interface-name>system</interface-name>
                    <admin-state>disable</admin-state>
                </interface>
            </router>
        </configure>
    </data>
</rpc-reply>
]]>]]>
```

5.2.5.2 Example: Non-default NETCONF Base Namespace

The following example shows an allowed non-default NETCONF base namespace used in the <rpc> element.

```
<?xml version="1.0" encoding="UTF-8"?>
<rpc message-id="101"
xmlns="urn:ietf:params:xml:ns:netconf:base:1.0"
xmlns:alu="urn:nokia.com:sros:ns:yang:sr:conf">
     <get-config>
          <source> <running/> </source>
          <filter>
               <configure xmlns="urn:nokia.com:sros:ns:yang:sr:conf">
                  <router>
                       <router-name>Base</router-name>
                       <interface>
                             <interface-name>system</interface-name>
                       </interface>
                  </router>
               </configure>
          </filter>
```

```
     </get-config>
</rpc>
]]>]]>
```

In the following reply, a non-NETCONF base namespace is allowed and no error is returned.

```
<?xml version="1.0" encoding="UTF-8"?>
<rpc-reply message-id="101" xmlns="urn:ietf:params:xml:ns:netconf:base:1.0"
xmlns:alu=="urn:nokia.com:sros:ns:yang:sr:conf">
    <data>
        <configure xmlns="urn:nokia.com:sros:ns:yang:sr:conf">
            <router>
                <router-name>Base</router-name>
                <interface>
                    <interface-name>system</interface-name>
                    <admin-state>disable</admin-state>
                </interface>
            </router>
        </configure>
    </data>
</rpc-reply>
]]>]]>
```

5.2.5.3 Example: Invalid NETCONF Namespace Declaration

The following example shows an invalid NETCONF namespace declared in the <rpc> element.

```
<?xml version="1.0" encoding="UTF-8"?>
<rpc message-id="101"
xmlns="urn:ietf:params:xml:ns:netconf:base:1.0"
xmlns:alu="urn:alcatel-lucent.com:sros:ns:yang:sr:conf">
        <get-config>
                <source><running/></source>
                <filter>
                    <configure xmlns="urn:nokia.com:sros:ns:yang:sr:conf">
                            <router>
                                    <router-name>Base</router-name>
                                    <interface>
                                         <interface-name>system</interface-name>
                                    </interface>
                            </router>
                    </configure>
                </filter>
        </get-config>
</rpc>
]]>]]>
```

In the following reply, SR OS returns an error.

```
<?xml version="1.0" encoding="UTF-8"?>
<rpc-reply message-id="101" xmlns="urn:ietf:params:xml:ns:netconf:base:1.0"
```

```
 xmlns:alu="urn:alcatel-lucent.com:sros:ns:yang:sr:conf">
    <rpc-error>
        <error-type>protocol</error-type>
        <error-tag>unknown-namespace</error-tag>
        <error-severity>error</error-severity>
        <error-message>
          An unexpected namespace is present.
        </error-message>
        <error-info>
          <bad-element>rpc</bad-element>
          <bad-namespace>urn:alcatel-lucent.com:sros:ns:yang:sr:conf</bad-namespace>
        </error-info>
    </rpc-error>
</rpc-reply>
]]>]]>
```

5.2.5.4 Example: Non-default NETCONF Namespace or Prefix Declaration in a Child Tag

The following example shows a non-default NETCONF namespace or prefix declared in any child tag overriding the one declared under the <rpc> tag.

```
<?xml version="1.0" encoding="UTF-8"?>
<rpc message-id="101"
xmlns="urn:ietf:params:xml:ns:netconf:base:1.0"
xmlns:alu="urn:ietf:params:xml:ns:netconf:base:1.0">
     <get-config>
          <source> <running/> </source>
          <filter>
             <configure xmlns="urn:nokia.com:sros:ns:yang:sr:conf">
                <router>
                    <router-name>Base</router-name>
                    <interface xmlns:alu="urn:nokia.com:sros:ns:yang:sr:conf">
                          <alu:interface-name>system</alu:interface-name>
                    </interface>
                </router>
             </configure>
          </filter>
     </get-config>
</rpc>
]]>]]>
```

In the following reply, the non-standard NETCONF namspace or prefix used in the tag is ignored.

```
<?xml version="1.0" encoding="UTF-8"?>
<rpc-reply message-id="101" xmlns="urn:ietf:params:xml:ns:netconf:base:1.0"
 xmlns:alu="urn:ietf:params:xml:ns:netconf:base:1.0">
    <data>
        <configure xmlns="urn:nokia.com:sros:ns:yang:sr:conf">
            <router>
                <router-name>Base</router-name>
                <interface>
```

```
                    <interface-name>system</interface-name>
                    <admin-state>disable</admin-state>
                </interface>
            </router>
        </configure>
    </data>
</rpc-reply>
]]>]]>
```

5.2.5.5 Example: Chunked Frame Mechanism

The following example shows a chunked message.

```
#359
<?xml version="1.0" encoding="UTF-8"?><rpc message-id="101"
xmlns="urn:ietf:params:xml:ns:netconf:base:1.0"><get-config><source><running/></
source><filter>
<configure xmlns="urn:nokia.com:sros:ns:yang:sr:conf"><router><router-name>Base</
router-name>
<interface><interface-name>system</interface-name></interface></router></
configure></filter></get-
config></rpc>
##
```

The following example shows the reply.

```
#38
<?xml version="1.0" encoding="UTF-8"?>
#1
#10
<rpc-reply
#17
 message-id="101"
#48
 xmlns="urn:ietf:params:xml:ns:netconf:base:1.0"
#1
>
#1
#9
    <data
#1
>
#1
#63
        <configure xmlns="urn:nokia.com:sros:ns:yang:sr:conf">
#21
            <router>
#48
                <router-name>Base</router-name>
#28
                <interface>
#60
                    <interface-name>system</interface-name>
#55
```

```
                    <admin-state>disable</admin-state>
#29
                </interface>
#22
            </router>
#21
        </configure>
#11
    </data>
#1
#12
</rpc-reply>
##
```

5.2.6 Establishing a NETCONF Session

The following example shows a client on a Linux PC initiating a connection to an SR OS NETCONF server. The SSH session must be invoked using an SSH subsystem (as recommended in RFC 6242).

```
ssh user_name@netconf_server_ip -p port_number -s netconf
```

The following example shows an exchange of hello messages, which include advertisement of capabilities.

From the SR OS server:

```
<?xml version="1.0" encoding="UTF-8"?>
<hello xmlns="urn:ietf:params:xml:ns:netconf:base:1.0">
    <capabilities>
        <capability>urn:ietf:params:netconf:base:1.0</capability>
        <capability>urn:ietf:params:netconf:base:1.1</capability>
        ...
        ...
    </capabilities>
    <session-id>20</session-id>
</hello>
]]>]]>
```

A NETCONF client can reply with a hello message as shown in either of the following:

```
<?xml version="1.0" encoding="UTF-8"?>
<hello xmlns="urn:ietf:params:xml:ns:netconf:base:1.0">
    <capabilities>
        <capability>urn:ietf:params:netconf:base:1.0</capability>
    </capabilities>
</hello>
]]>]]>
```

```
<?xml version="1.0" encoding="UTF-8"?>
<hello xmlns="urn:ietf:params:xml:ns:netconf:base:1.0">
```

```
  <capabilities>
    <capability>urn:ietf:params:netconf:base:1.1</capability>
  </capabilities>
</hello>
]]>]]>
```

5.2.6.1 Example: Checking NETCONF Status

The following example shows a <get-config> request on the <running> datastore that checks on whether NETCONF is shut down or not on the router.

```
<?xml version="1.0" encoding="UTF-8"?>
<rpc message-id="101" xmlns="urn:ietf:params:xml:ns:netconf:base:1.0">
    <get-config>
          <source><running/></source>
          <filter>
               <configure xmlns="urn:nokia.com:sros:ns:yang:sr:conf">
                       <system>
                            <management-interface>
                                   <netconf/>
                            </management-interface>
                       </system>
               </configure>
          </filter>
</get-config>
</rpc>
]]>]]>
```

The following example shows the reply.

```
<?xml version="1.0" encoding="UTF-8"?>
<rpc-reply message-id="101" xmlns="urn:ietf:params:xml:ns:netconf:base:1.0">
    <data>
        <configure xmlns="urn:nokia.com:sros:ns:yang:sr:conf">
            <system>
                <management-interface>
                   <netconf>
                      <admin-state>enable</admin-state>
                      <auto-config-save>true</auto-config-save>
                   </netconf>
                <management-interface>
            </system>
        </configure>
    </data>
</rpc-reply>
]]>]]>
```

5.2.6.2 Example: Retrieving System Configurations, QoS, and Log Branches

The following example shows a <get-config> request on the <candidate> datastore to get the full configurations of the system, QoS, and log branches.

```
<?xml version="1.0" encoding="UTF-8"?>
<rpc message-id="101" xmlns="urn:ietf:params:xml:ns:netconf:base:1.0">
<get-config>
    <source><candidate/></source>
    <filter>
        <configure xmlns="urn:nokia.com:sros:ns:yang:sr:conf">
           <system>
           </system>
        </configure>
        <configure xmlns="urn:nokia.com:sros:ns:yang:sr:conf">
            <log/>
        </configure>
    </filter>
</get-config>
</rpc>
]]>]]>
```

The following example shows the reply.

```
<?xml version="1.0" encoding="UTF-8"?>
<rpc-reply message-id="101" xmlns="urn:ietf:params:xml:ns:netconf:base:1.0">
    <data>
        <configure xmlns="urn:nokia.com:sros:ns:yang:sr:conf">
            <log>
                <filter>
                    <filter-id>1001</filter-id>
                    <entry>
                       <entry-id>10</entry-id>
                       <description>events of major severity or higher</description>
                       <action>forward</action>
                       <match>
                           <severity>
                               <gte>major</gte>
                           </severity>
                       </match>
                    </entry>
                </filter>
...
...
                <log-id>
                    <id>101</id>
                    <destination>
                        <netconf>
                        </netconf>
                    </destination>
                </log-id>
            </log>
            <system>
                <name>Test</name>
                <dns>
```

```
                    <address-pref>ipv4-only</address-pref>
                </dns>
...
...
            </system>
        </configure>
    </data>
</rpc-reply>
]]>]]>
```

5.2.6.3 Example: Creating an Epipe Service

The following example shows an <edit-config> request on the <candidate> datastore to create a basic Epipe service.

```
<?xml version="1.0" encoding="UTF-8"?>
<rpc message-id="101" xmlns="urn:ietf:params:xml:ns:netconf:base:1.0">
<edit-config>
    <target><candidate/></target>
    <config>
        <configure xmlns="urn:nokia.com:sros:ns:yang:sr:conf">
            <service>
                <epipe>
                    <service-name>CustDoc</service-name>
                    <customer>1</customer>
                    <service-mtu>1514</service-mtu>
                </epipe>
            </service>
        </configure>
    </config>
</edit-config>
</rpc>
]]>]]>
```

The following example shows the reply.

```
<?xml version="1.0" encoding="UTF-8"?>
<rpc-reply message-id="101" xmlns="urn:ietf:params:xml:ns:netconf:base:1.0">
    <ok/>
</rpc-reply>
]]>]]>
```

5.2.6.4 Example: Returning Multiple Errors

The following example shows SR OS returning multiple errors with the <commit>.

```
<?xml version="1.0" encoding="UTF-8"?>
<rpc message-id="101" xmlns="urn:ietf:params:xml:ns:netconf:base:1.0">
<edit-config>
    <target><candidate/></target>
```

```
    <config>
        <configure xmlns="urn:nokia.com:sros:ns:yang:sr:conf">
            <router>
                <router-name>Base</router-name>
                <ldp>
                    <interface-parameters>
                        <interface>
                            <ip-int-name>xe-1/1/1</ip-int-name>
                            <ipv4>
                            </ipv4>
                        </interface>
                        <interface>
                            <ip-int-name>xe-1/2/1</ip-int-name>
                            <ipv4>
                            </ipv4>
                        </interface>
                    </interface-parameters>
                    <targeted-session>
                        <peer>
                        <ip-address>172.22.1.34</ip-address>
                        </peer>
                    </targeted-session>
                    <tcp-session-parameters>
                        <peer-transport>
                        <ip-address>172.22.1.34</ip-address>
                        <authentication-key>Ru4bf!n</authentication-key>
                        </peer-transport>
                   </tcp-session-parameters>
                </ldp>
           </router>
         </configure>
      </config>
</edit-config>
</rpc>
]]>]]>


<?xml version="1.0" encoding="UTF-8"?>
<rpc-reply message-id="101" xmlns="urn:ietf:params:xml:ns:netconf:base:1.0">
    <ok/>
</rpc-reply>
]]>]]>


<?xml version="1.0" encoding="UTF-8"?>
<rpc message-id="101" xmlns="urn:ietf:params:xml:ns:netconf:base:1.0">
   <commit/>
 </rpc>
]]>]]>


<?xml version="1.0" encoding="UTF-8"?>
<rpc-reply message-id="101" xmlns="urn:ietf:params:xml:ns:netconf:base:1.0">
    <rpc-error>
        <error-type>application</error-type>
        <error-tag>operation-failed</error-tag>
        <error-severity>error</error-severity>
        <error-path xmlns:a="urn:nokia.com:sros:ns:yang:sr:conf">
/a:configure/a:router[a:router-name=&quot;Base&quot;]/a:ldp/a:interface-parameters/
a:interface[a:ip-int-name=&quot;xe-1/1/1&quot;]
        </error-path>
```

```
        <error-message>
            MINOR: MGMT_CORE #224: Entry does not exist - configure router router-
name &quot;Base&quot; interface interface-name
&quot;xe-1/1/1&quot;
        </error-message>
        <error-info>
            <err-element>interface</err-element>
        </error-info>
    </rpc-error>
    <rpc-error>
        <error-type>application</error-type>
        <error-tag>operation-failed</error-tag>
        <error-severity>error</error-severity>
        <error-path xmlns:a="urn:nokia.com:sros:ns:yang:sr:conf">
/a:configure/a:router[a:router-name=&quot;Base&quot;]/a:ldp/a:interface-parameters/
a:interface[a:ip-int-name=&quot;xe-1/2/1&quot;]
        </error-path>
        <error-message>
            MINOR: MGMT_CORE #224: Entry does not exist - configure router router-
name &quot;Base&quot; interface interface-name
&quot;xe-1/2/1&quot;
        </error-message>
        <error-info>
            <err-element>interface</err-element>
        </error-info>
    </rpc-error>
</rpc-reply>
]]>]]>
```

5.3 NETCONF Notifications

NETCONF notifications support is a standard IETF asynchronous notification delivery service for the NETCONF that is specified in RFC 5277.

The :notification capability and the :interleave capability are advertised in the SR OS NETCONF server <hello> as:

<capability>urn:ietf:params:netconf:capability:notification:1.0</capability>
<capability>urn:ietf:params:netconf:capability:interleave:1.0</capability>

The following are characteristics of the NETCONF notifications capabilities supported in SR OS:

- The :notification capability allows the SR OS NETCONF server can process a subscription and send event notifications to the NETCONF client.
- The :interleave capability allows the SR OS NETCONF server supports receiving, processing, and responding to NETCONF requests on the same NETCONF session that has an active notification subscription.

- A NETCONF client needs to maintain an open NETCONF session with the NETCONF server in order to receive NETCONF notifications.
- A NETCONF client can send a <create-subscription> RPC to the SR OS NETCONF server to start receiving notification messages.
- If the SR OS NETCONF server can satisfy the request, SR OS sends an <OK> element within the <rpc-reply>.
- If the SR OS NETCONF server cannot satisfy the request, SR OS sends an <rpc-error> element within the <rpc-reply>.
- Subscriptions are nonpersistent and their lifetime is defined by their NETCONF session (not maintained with a router reboot).
- An optional parameter that can be defined for a <create-subscription> RPC is [stream]. The following are characteristics of the [stream] parameter:
  - An event stream is a set of event notifications matching a specified forwarding criteria and available to the NETCONF clients for subscription.
  - A NETCONF session can subscribe to only one stream at a time.
  - One stream can be subscribed-to by many NETCONF sessions.
  - The SR OS NETCONF server maintains one or more event streams.
  - SR OS uses the SR OS event reporting framework for NETCONF notifications.
  - A log-id can be configured to be a NETCONF stream. A “netconf-stream” exists for each log-id to assign a NETCONF "stream" name to the log-id. A netconf-stream is unique per SR OS device. The netconf-stream must be configured with “to netconf” for subscriptions to be accepted. If a netconf-stream is changed, active subscriptions to the changed NETCONF stream name are terminated by SR OS.
  - There is one preconfigured stream with the netconf-stream set to “to netconf”, that is, log-id 101. It is used by default if the [stream] parameter is not specified. The preconfigured stream is modifiable but not deletable.
  - Other streams can be configured using NETCONF or CLI. These streams are user-configured, which means that they are modifiable and can be deleted. A user-configured stream netconf-stream cannot be set to “to netconf” because “to netconf” is reserved for the preconfigured stream (that is, log-id 101).
  - When a NETCONF client tries to subscribe to the SNMP log-id or a non-configured log-id, SR OS returns an error.
  - SR OS supports a maximum number of 64 concurrent subscriptions to all streams.
  - Notifications can be filtered out using a log-id “filter” or using base-op for create-subscriptions RPC.

- After the NETCONF server receives an SR OS event through a stream, a <notification> element is ready to be sent to all NETCONF sessions subscribed to that stream as per their filters.

- SR OS supports the following NETCONF notifications (see NETCONF Notification Examples for more information):
  - **sros-config-change-event:** sent with every configuration change; that is, any new, deleted, or modified configuration
  - **sros-state-change-event:** sent with every state change
  - **sros-command-accounting-event:** sent to keep track of which user did what activity on the SR OS device
  - **sros-log-generic-event:** contains the rest of the SR OS log events (except for the LI ones)
  - **netconf-config-change**: A notification based on the model-driven configuration change log events, “mdConfigChange”, “mdOcConfigChange”, “mdBofConfigChange”, and “mdDebugConfigChange”. The notification is sent upon any configuration change that occurs in the running datastore by a model-driven management interface, using either the Nokia SR OS or OpenConfig data models, and in any configuration region except li (such as configure, bof, and debug). By default, the notification is disabled because all corresponding log events are also disabled by default. The notification uses the standard notification: netconf-config-change (as per RFC 6470) augmented with a value leaf.

    A single configuration change may involve editing more than one object (target). Each log event contains only a single object edit. As a result, only one object (target) edit can exist per **netconf-config-change** notification. Bundling of edits in a single **netconf-config-change** notification is not allowed.
  - **sros-md-rpc-accounting-event**: A notification based on the NETCONF/gRPC local command accounting log events (the netconf_auth, netconf_unauth, grpc_auth, and grpc_unauth log events). This notification is sent upon receiving any RPC from a NETCONF/gRPC client. The NETCONF/gRPC local command accounting log events and NETCONF notification do not show the details of the configuration changes sent using the NETCONF/gRPC RPCs.
- SR OS supports the following LI NETCONF notifications:
  - **sros-li-config-change-event:** Sent with every LI configuration change; that is, any new, deleted, or modified configuration
  - **sros-li-state-change-event:** Sent with every LI state change
  - **sros-li-command-accounting-event:** Sent to keep track of each LI user’s activities on the SR OS device
  - **sros-li-log-generic-event:** Contains the remaining SR OS LI log events

- **netconf-li-config-change**: A notification based on the model-driven configuration change log event (the “mdLiConfigChange” log event). It is sent to the running datastore in the system for all configuration changes to a model-driven management interface. By default, this notification is enabled, because the log event is also enabled by default.
- **sros-md-li-rpc-accounting-event**: A notification based on the NETCONF/gRPC local command accounting LI log events (the netconf_auth, netconf_unauth, grpc_auth, and grpc_unauth log events). This notification is sent upon receiving any RPC from a NETCONF/gRPC client. The NETCONF/gRPC local command accounting LI log events and LI NETCONF notification do not show the details of the LI configuration changes sent using the RPCs.

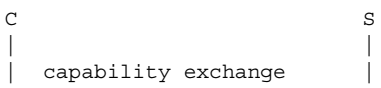
In a <create-subscription>, a <filter> is an optional argument that is not supported by SR OS.

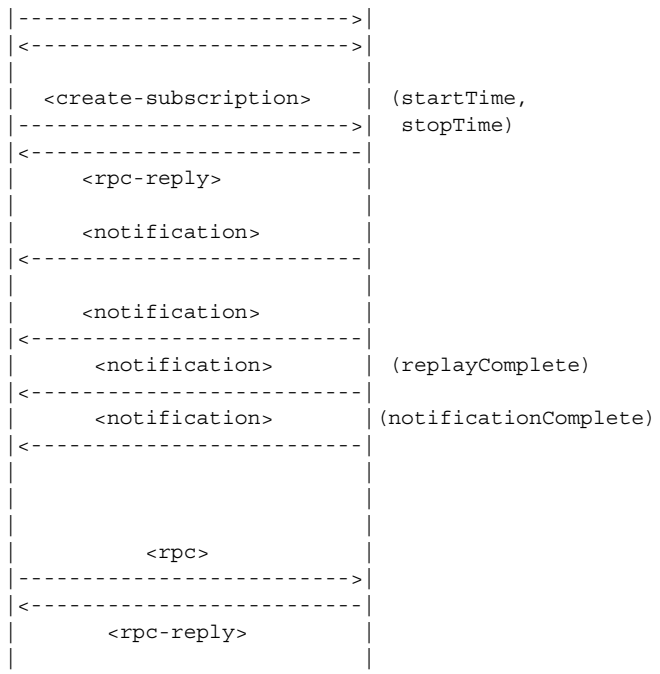
In a <create-subscription>, a <startTime> is an optional argument. This argument triggers the starting time of a replay. If it is not present, the subscription cannot be used to replay. A <startTime> cannot specify a time that is later than the current time (that is, in the future). SR OS supports timezones.

In a <create-subscription>, a <stopTime> is another optional argument. If this argument is not present, notifications continue to be sent until the subscription is terminated. A <stopTime> can specify a time that is later than the current time (that is, in the future). SR OS supports timezones.

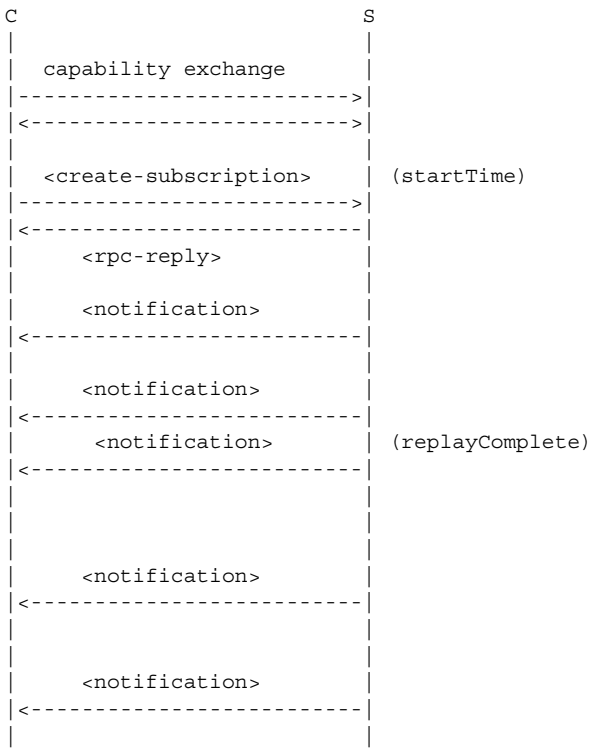
A replay buffer is maintained by the SR OS server (per stream) and sorted by the order they were initially sent out (that is, by sequence-id, and not by timestamps).

- A replay request from the client causes stored events to be sent to the client for the specified time interval.
- A stream that supports replay is not expected to have an unlimited supply of saved notifications available to accommodate any replay request.
- The <startTime> and <stopTime> arguments specify when collections begin and end, respectively.
- A <replayComplete> notification is sent to indicate that all the replay notifications have been sent.
  - If a <stopTime> was specified, the session then becomes a normal NETCONF session, and the NETCONF server accepts <rpc> operations. A <notificationComplete> notification is expected after the <replayComplete> if a <stopTime> was specified. The following is an example of a session with a <stopTime> specified:





- If a <stopTime> was not specified, the session will continue to send notifications as they arise in the system. The following is an example of a session without a <stopTime> specified:



- If neither <startTime> and <stopTime> arguments are present, no replay is present and notifications continue to be sent until the subscription is terminated.

5.3.1 NETCONF Notification Examples

This section provides examples of NETCONF notifications.

5.3.1.1 Example: <create-subscription> Operation

The following example shows a create-subscription> operation.

```
<?xml version="1.0" encoding="UTF-8"?>
<rpc message-id="101" xmlns="urn:ietf:params:xml:ns:netconf:base:1.0">
    <create-subscription>
    </create-subscription>
</rpc>
]]>]]>
```

The following examples show the reply.

```
<?xml version="1.0" encoding="UTF-8"?>
<rpc-reply message-id="101" xmlns="urn:ietf:params:xml:ns:netconf:base:1.0">
    <ok/>
</rpc-reply>
]]>]]>
```

5.3.1.2 Example: sros-config-change-event Notification

The following example shows an sros-config-change-event notification.

```
<notification xmlns="urn:ietf:params:xml:ns:netconf:notification:1.0">
    <eventTime>2017-06-12T09:12:43.376Z</eventTime>
    <sros-config-change-event xmlns="urn:nokia.com:sros:ns:yang:sr:notifications">
        <sequence-number>8447</sequence-number>
        <severity>warning</severity>
        <application>system</application>
        <event-id>2008</event-id>
        <event-name>tmnxConfigDelete</event-name>
        <router-name>Base</router-name>
        <subject>LDP</subject>
        <message>vRtrLdpNgSessionTable: Virtual Router 1, Peer 2.2.2.2:0. managed ob
ject deleted</message>
        <event-params>
```

```
            <tmnxNotifyRow>vRtrLdpNgSessState.1.1.6.2.2.2.2.0.0</tmnxNotifyRow>
            <tmnxNotifyEntryOID>vRtrLdpNgSessionEntry</tmnxNotifyEntryOID>
            <tmnxNotifyObjectName>vRtrLdpNgSessionTable: Virtual Router 1, Peer 2.2.
2.2:0.</tmnxNotifyObjectName>
        </event-params>
    </sros-config-change-event>
</notification>
```

5.3.1.3 Example: sros-state-change-event Notification

The following example shows an sros-state-change-event notification.

```
<notification xmlns="urn:ietf:params:xml:ns:netconf:notification:1.0">
    <eventTime>2017-06-12T09:16:36.781Z</eventTime>
    <sros-state-change-event xmlns="urn:nokia.com:sros:ns:yang:sr:notifications">
        <sequence-number>8460</sequence-number>
        <severity>warning</severity>
        <application>system</application>
        <event-id>2009</event-id>
        <event-name>tmnxStateChange</event-name>
        <router-name>Base</router-name>
        <subject>LDP</subject>
        <message>Status of vRtrLdpNgSessionTable: Virtual Router 1, Peer 2.2.2.2:0.
changed administrative state: inService, operational state: inService</message>
        <event-params>
            <tmnxNotifyRow>vRtrLdpNgSessState.1.1.6.2.2.2.2.0.0</tmnxNotifyRow>
            <tmnxNotifyRowAdminState>inService</tmnxNotifyRowAdminState>
            <tmnxNotifyRowOperState>inService</tmnxNotifyRowOperState>
            <tmnxNotifyEntryOID>vRtrLdpNgSessionEntry</tmnxNotifyEntryOID>
            <tmnxNotifyObjectName>vRtrLdpNgSessionTable: Virtual Router 1, Peer 2.2.
2.2:0.</tmnxNotifyObjectName>
        </event-params>
    </sros-state-change-event>
</notification>
```

5.3.1.4 Example: sros-cli-accounting-event Notification

The following example shows an sros-cli-accounting-event notification.

```
<notification xmlns="urn:ietf:params:xml:ns:netconf:notification:1.0">
    <eventTime>2017-06-12T09:11:45.476Z</eventTime>
    <sros-command-accounting-
event mlns="urn:nokia.com:sros:ns:yang:sr:notifications">
        <sequence-number>8462</sequence-number>
        <severity>minor</severity>
        <application>user</application>
        <event-id>2011</event-id>
        <event-name>cli_config_io</event-name>
        <router-name>Base</router-name>
        <subject>admin</subject>
        <message>User from CONSOLE: Dut-C>config>log>log-id#  /
```

```
configure router interface "toDutB_214" </message>
        <event-params>
            <srcAddr>CONSOLE</srcAddr>
            <prompt>Dut-C>config>log>log-id# </prompt>
            <message>/configure router interface "toDutB_214" </message>
        </event-params>
    </sros-command-accounting-event>
</notification>
```

5.3.1.5 Example: sros-log-generic-event Notification

The following shows an sros-log-generic-event notification.

```
<notification xmlns="urn:ietf:params:xml:ns:netconf:notification:1.0">
    <eventTime>2017-06-12T09:12:42.344Z</eventTime>
    <sros-log-generic-event xmlns="urn:nokia.com:sros:ns:yang:sr:notifications">
        <sequence-number>8443</sequence-number>
        <severity>warning</severity>
        <application>ospf</application>
        <event-id>2047</event-id>
        <event-name>tmnxOspfNgIfStateChange</event-name>
        <router-name>Base</router-name>
        <subject>VR:  1 OSPFv2 (0) </subject>
        <message>LCL_RTR_ID 1.1.1.1: Interface toDutB_214 state changed to down (eve
nt IF_DOWN)</message>
        <event-params>
            <vRtrID>1</vRtrID>
            <tmnxOspfVersion>version2</tmnxOspfVersion>
            <tmnxOspfInstance>0</tmnxOspfInstance>
            <tmnxOspfRouterId>16843009</tmnxOspfRouterId>
            <tmnxOspfNgIfIndex>0x00000007</tmnxOspfNgIfIndex>
            <tmnxOspfNgIfInstId>0</tmnxOspfNgIfInstId>
            <tmnxOspfNgIfAreaId>0</tmnxOspfNgIfAreaId>
            <tmnxOspfNgIfState>down</tmnxOspfNgIfState>
            <tmnxOspfIfIpName>toDutB_214</tmnxOspfIfIpName>
            <tmnxOspfIfEvent>IF_DOWN</tmnxOspfIfEvent>
            <ospfRouterIdIpAddr>1.1.1.1</ospfRouterIdIpAddr>
        </event-params>
    </sros-log-generic-event>
</notification>
```

5.3.1.6 Example: netconf-config-change Notification

The following example shows a netconf-config-change notification.

```
<notification xmlns="urn:ietf:params:xml:ns:netconf:notification:1.0"> <eventTime>20
16-01-01T19:17:33Z</eventTime>
<netconf-config-change
      xmlns="urn:ietf:params:xml:ns:yang:ietf-netconf-notifications"
      xmlns:notif="urn:nokia.com:sros:ns:yang:sr:notifications"
      xmlns:sros="urn:nokia.com:sros:ns:yang:sr:conf">
```

```
      <changed-by>
        <username>user_name</username>
        <session-id>8</session-id>
        <source-host>138.192.72.45</remote-host>
      </changed-by>
    <datastore>running</datastore>
    <edit>
      <target>/config/service/epipe[serviceId=1]</target>
      <operation>create</operation>
      <notif:value>anyValue</notif:value>
    </edit>
</netconf-config-change>
</notification>
```

5.3.1.7 Example: sros-md-rpc-accounting-event Notification

The following example shows an sros-md-rpc-accounting-event notification.

```
<?xml version="1.0" encoding="UTF-8"?>
<notification xmlns="urn:ietf:params:xml:ns:netconf:notification:1.0">
    <eventTime>2019-10-08T21:01:50.165Z</eventTime>
    <sros-md-rpc-accounting-
 event xmlns="urn:nokia.com:sros:ns:yang:sr:notifications">
        <sequence-number>124</sequence-number>
        <severity>minor</severity>
        <application>security</application>
        <event-id>2227</event-id>
        <event-name>netconf_auth</event-name>
        <router-name>management</router-name>
        <subject>admin</subject>
        <message>User admin from 192.168.7.229 port 44559 to port 830 session 7: edi
t-config RPC authorized</message>
        <event-params>
            <userName>admin</userName>
            <srcAddr>192.168.7.229</srcAddr>
            <srcPort>44559</srcPort>
            <dstPort>830</dstPort>
            <sessionId>7</sessionId>
            <rpcName>edit-config</rpcName>
        </event-params>
    </sros-md-rpc-accounting-event>
</notification>
]]>]]>
```

5.4 NETCONF Monitoring

The :ietf-netconf-monitoring capability is advertised in the SR OS NETCONF server <hello> as:

<capability>urn:ietf:params:xml:ns:yang:ietf-netconf-monitoring</capability>

The advertised capability provides information about the schemas supported by SR OS, which allows a NETCONF client to query and retrieve schema information from the SR OS NETCONF server.

SR OS supports the "/netconf-state/schemas" subtree only from the YANG model that is used to monitor the NETCONF protocol as per RFC 6022 (that is, ":ietf-netconf-monitoring" capability).

SR OS links retrieve the supported schemas for all the CLI commands that are used to enable and disable the YANG modules. The following are examples:

- A "/netconf-state/schemas" path returns all supported Nokia models (modules and sub-modules) when the **nokia-modules** parameter is set to **true**.
- A "/netconf-state/schemas" path returns the supported combined Nokia (flat) models when the **nokia-combined-modules** parameter is set to **true**.
- A "/netconf-state/schemas" path returns the ietf modules (for example, ietf-inet-types or ietf-yang-types) and the Nokia types in the returned list of schemas when either the **nokia-modules** or **nokia-combined-modules** is enabled.

The following example shows a request and the received response.

```
<?xml version="1.0" encoding="UTF-8"?>
<rpc message-id="101" xmlns="urn:ietf:params:xml:ns:netconf:base:1.0">
  <get>
    <filter>
      <netconf-state xmlns="urn:ietf:params:xml:ns:yang:ietf-netconf-monitoring">
        <schemas/>
      </netconf-state>
    </filter>
  </get>
</rpc>
]]>]]>
```

The following example shows the reply.

```
<?xml version="1.0" encoding="UTF-8"?>
<rpc-reply message-id="101" xmlns="urn:ietf:params:xml:ns:netconf:base:1.0">
    <data>
        <netconf-state xmlns="urn:ietf:params:xml:ns:yang:ietf-netconf-monitoring">
            <schemas>
                <schema>
                    <identifier>nokia-conf</identifier>
                    <version>2016-07-06</version>
                    <format>yang</format>
                    <namespace>urn:nokia.com:sros:ns:yang:sr:conf</namespace>
                    <location>NETCONF</location>
                </schema>
                <schema>
                    <identifier>nokia-conf-aa-group</identifier>
                    <version>2018-09-14</version>
                    <format>yang</format>
                    <namespace>urn:nokia.com:sros:ns:yang:sr:conf</namespace>
```

```
                    <location>NETCONF</location>
                </schema>
                <schema>
                    <identifier>nokia-conf-aaa</identifier>
                    <version>2018-08-27</version>
                    <format>yang</format>
                    <namespace>urn:nokia.com:sros:ns:yang:sr:conf</namespace>
                    <location>NETCONF</location>
                </schema>
                    ….
                    ….
                <schema>
                    <identifier>nokia-state</identifier>
                    <version>2016-07-06</version>
                    <format>yang</format>
                    <namespace>urn:nokia.com:sros:ns:yang:sr:state</namespace>
                    <location>NETCONF</location>
                </schema>
                <schema>
                    <identifier>nokia-state-aa-group</identifier>
                    <version>2018-09-14</version>
                    <format>yang</format>
                    <namespace>urn:nokia.com:sros:ns:yang:sr:state</namespace>
                    <location>NETCONF</location>
                </schema>
                <schema>
                    <identifier>nokia-state-aaa</identifier>
                    <version>2018-08-27</version>
                    <format>yang</format>
                    <namespace>urn:nokia.com:sros:ns:yang:sr:state</namespace>
                    <location>NETCONF</location>
                </schema>
                    …
                    …
      </schemas>
    </state>
  </data>
</rpc-reply>
```

A <get-schema> operation is supported for explicit schema retrieval using NETCONF (YANG data models' discovery and download as per RFC 6022). The following parameters are supported:

- **identifier**: A mandatory string. Specifies an identifier for the schema list entry (YANG file). It can be the name of a module or a submodule.
- **version**: An optional string. Specifies a version of the schema requested (for example, YANG file). It represents the most recent YANG **revision** statement in a module or submodule. Empty string if no **revision** statement is present. As multiple versions may be supported by the NETCONF server, each version must be reported individually in the schema list (it can have the same identifier but different versions).
- **format**: An optional string. Specifies the data modeling language that the schema is written in. Default value is “yang” when not specified; “yang” shall be the only value supported if specified.

Unless the user intentionally specifies a schema path destination from which to acquire the YANG schema files, the software upgrade process manages the YANG schema files to ensure the schema files are synchronized with the software image on both the primary and standby CPM.

When an SR OS image boots (from **bof primary-image**), the associated YANG files shall match the image. If the primary SR OS image fails to boot and the secondary (or tertiary) SR OS image (from **bof secondary-image** or **bof tertiary-image**) loads, the YANG schema files associated with the loaded image shall be installed and available to the <get-schema> NETCONF RPC. It is recommended that each of the **primary-image**, **secondary-image**, and **tertiary-image** strings do not exceed 120 characters for the <get-schema> request to work properly with all schema files.

The **configure system management-interface schema-path** CLI command can be used to configure the schema path. See the **schema-path** command description in *7450 ESS, 7750 SR, 7950 XRS, and VSR Classic CLI Command Reference Guide* for more information. If **schema-path** is configured, all the YANG files must be manually copied to the specified **schema-path** URL prior to using the <get-schema> RPC successfully.

When the requested schema does not exist, the <error-tag> returned is "invalid-value". The maximum length of a schema path URL is 180 characters, however, Nokia recommends that a URL string be less than or equal to 135 characters, to guarantee that a <get-schema> will work properly with the longest YANG module name in SR OS.

When more than one schema matches the requested parameters, the <error-tag> returned is "operation-failed".

The following example shows a <get-schema> request.

```
<?xml version="1.0" encoding="UTF-8"?>
<rpc message-id="101" xmlns="urn:ietf:params:xml:ns:netconf:base:1.0">
    <get-schema xmlns="urn:ietf:params:xml:ns:yang:ietf-netconf-monitoring">
        <identifier>nokia-conf</identifier>
    </get-schema>
</rpc>
]]>]]>
```

The following example shows the reply.

```
<?xml version="1.0" encoding="UTF-8"?>
<rpc-reply message-id="101" xmlns="urn:ietf:params:xml:ns:netconf:base:1.0">
    <data xmlns="urn:ietf:params:xml:ns:yang:ietf-netconf-
monitoring"><![CDATA[module nokia-conf {
    yang-version "1.1";
    namespace "urn:nokia.com:sros:ns:yang:sr:conf";
..
}
]]></data>
```

```
</rpc-reply>
```

5.5 YANG Library

SR OS supports a mechanism, a YANG library, to identify the YANG modules and submodules that are implemented by the NETCONF server. NETCONF clients should be able to query or cache the YANG library contents and identify whether their cache is out-of-date.

The SR OS NETCONF server supports the “/yang-library” state model and advertises the following capability in the <hello> message (in accordance with RFC 8525):

```
<capability>urn:ietf:params:netconf:capability:yang-
library:1.1?revision=<date>&content-id=<content-id-value></capability>
```

The following is the YANG tree diagram for the “/yang-library”model:

```
 module: ietf-yang-library
     +--ro yang-library
        +--ro module-set* [name]
        |  +--ro name                  string
        |  +--ro module* [name]
        |  |  +--ro name         yang:yang-identifier
        |  |  +--ro revision?    revision-identifier
        |  |  +--ro namespace    inet:uri
        |  |  +--ro location*    inet:uri
        |  |  +--ro submodule* [name]
        |  |  |  +--ro name        yang:yang-identifier
        |  |  |  +--ro revision?   revision-identifier
        |  |  |  +--ro location*   inet:uri
        |  |  +--ro feature*     yang:yang-identifier
        |  |  +--ro deviation*   -> ../../module/name
        |  +--ro import-only-module* [name revision]
        |     +--ro name         yang:yang-identifier
        |     +--ro revision     union
        |     +--ro namespace    inet:uri
        |     +--ro location*    inet:uri
        |     +--ro submodule* [name]
        |        +--ro name        yang:yang-identifier
        |        +--ro revision?   revision-identifier
        |        +--ro location*   inet:uri
        +--ro schema* [name]
        |  +--ro name          string
        |  +--ro module-set*   -> ../../module-set/name
        +--ro datastore* [name]
        |  +--ro name      ds:datastore-ref
        |  +--ro schema    -> ../../schema/name
        +--ro content-id    string
```

The SR OS NETCONF server advertises the following capability in the <hello> message:

```
<capability>urn:ietf:params:netconf:capability:yang-library:1.0?revision=<revision-
date>&amp;module-set-id=<string></capability>
```

The following is the YANG tree diagram for the **modules-state** model:

```
 +--ro modules-state
   +--ro module-set-id    string
   +--ro module* [name revision]
     +--ro name                yang:yang-identifier
     +--ro revision            union
     +--ro schema?             inet:uri
     +--ro namespace           inet:uri
     +--ro feature*            yang:yang-identifier
     +--ro deviation* [name revision]
     |  +--ro name        yang:yang-identifier
     |  +--ro revision    union
     +--ro conformance-type    enumeration
     +--ro submodule* [name revision]
       +--ro name        yang:yang-identifier
       +--ro revision    union
       +--ro schema?     inet:uri
```

The **module-set-id** is a mandatory leaf that identifies a set of YANG modules that the SR OS NETCONF server supports. The value of this leaf changes whenever there is a change in the set of modules or submodules in the YANG library. When this change occurs, SR OS changes the **module-set-id** value advertised in the NETCONF server <hello> message.

The **modules-state** can be used by the NETCONF client to fetch the YANG library, cache it, and re-fetch it only if the value of the **module-set-id** changes again. The YANG library is returned in the **module** list.

Example

If the SR OS NETCONF server advertises the following capability, the NETCONF client can use the advertised **module-set-id** to query the YANG library:

```
<capability>urn:ietf:params:netconf:capability:yang-library:1.0?revision=2018-
05-08&module-set-id=1234</capability>
```

The following example shows the NETCONF client using the advertised **module-set-id** to query the YANG library.

```
<rpc xmlns="urn:ietf:params:xml:ns:netconf:base:1.0" message-id="101">
<get>
 <filter type="subtree">
  <modules-state xmlns="urn:ietf:params:xml:ns:yang:ietf-yang-library">
     <module-set-id>1234</module-set-id>
     <module>
```

```
      </module>
    </modules-state>
  </filter>
 </get>
</rpc>
]]>]]>
```

The following example shows the reply.

```
<?xml version="1.0" encoding="UTF-8"?>
<rpc-reply message-id="101" xmlns="urn:ietf:params:xml:ns:netconf:base:1.0">
    <data>
        <modules-state xmlns="urn:ietf:params:xml:ns:yang:ietf-yang-library">
            <module-set-id>1234</module-set-id>
            <module>
                <name>iana-if-type</name>
                <revision>2014-05-08</revision>
                <schema></schema>
                <namespace>urn:ietf:params:xml:ns:yang:iana-if-type</namespace>
                <feature></feature>
                <conformance-type>implement</conformance-type>
            </module>
                        …
                        …
            <module>
                <name>nokia-conf</name>
                <revision>2016-07-06</revision>
                <schema></schema>
                <namespace>urn:nokia.com:sros:ns:yang:sr:conf</namespace>
                <feature></feature>
                <conformance-type>implement</conformance-type>
                <submodule>
                    <name>nokia-conf-aa-common</name>
                   <revision>2018-04-23</revision>
                    <schema></schema>
                </submodule>
                                …
                                …
            </module>
                        …
                        …
        </modules-state>
    </data>
</rpc-reply>
]]>]]>
```

5.6 NETCONF Operations Using the md-cli-raw-command Request

The md-cli-raw-command request allows a NETCONF client to execute a wide set of operations on the SR OS router.

The command input string accepts a command in the exact format as it would be entered in the MD-CLI. For example:

```
<rpc message-id="101" xmlns="urn:ietf:params:xml:ns:netconf:base:1.0">
  <action xmlns="urn:ietf:params:xml:ns:yang:1">
    <global-operations xmlns="urn:nokia.com:sros:ns:yang:sr:oper-global">
      <md-cli-raw-command>
        <md-cli-input-line>clear router Base interface system statistics</md-cli-
input-line>
      </md-cli-raw-command>
    </global-operations>
  </action>
</rpc>
```

Other examples of commands that can be used as input strings include:

- admin reboot now
- clear router Base interface system statistics
- file list cf1:
- oam eth-cfm loopback 28 md-admin-name "14" ma-admin-name "200" mep-id 32 send-count 5 interval 10
- ping 10.10.10.2
- show system information
- tools dump system-resources
- tools perform cflowd manual-export
- traceroute 192.168.10.1
- //debug router ldp interface foo

The SR OS NETCONF server workflow to process the md-cli-raw-command request is the following.

1. Open a new temporary MD-CLI session (with the same username as the NETCONF session).
2. Pass the input command to the MD-CLI engine.
3. Return the MD-CLI output to the NETCONF client as an unstructured block of text in the <rpc-reply> message.

The MD-CLI context for the operation is the root and the MD-CLI executes the command in operational mode, which is similar to a user newly logged into an MD-CLI session.

Interactive commands, or commands that prompt for input, are not supported and result in an error. For example, using “admin reboot” as an input string fails. Using “admin reboot now” is accepted. Other examples of interactive commands that are not supported include:

- enable
- password
- ssh
- telnet

The **cli-engine** command controls the engines allowed to process the input. For example, if **cli-engine** is set to allow only the MD-CLI engine (and not the classic CLI engine), any md-cli-raw-command input strings that start with "//" generate an error. Changes to the **cli-engine** configuration only take effect on raw-md-cli-command in NETCONF sessions that are started after the **cli-engine** configuration was changed.

Only a single operation is supported as the input to the md-cli-raw-command request. Multiple operations require multiple NETCONF RPCs.

The md-cli-raw-command request is not intended as a mechanism to read structured state data or to manage basic configuration. The YANG-modeled configuration and state data are managed and accessed using standard NETCONF operations, such as <edit-config>, <get-config>, and <get>.

The following MD-CLI commands and similar commands are not supported as input strings for md-cli-raw-command:

- admin show configuration
- bof
- configure
- debug
- edit-config
- environment
- exec
- info
- li
- //admin display-config
- //admin compare
- //admin rollback
- //admin view
- //bof
- //candidate
- //configure
- //environment
- //exec

Unstructured state information can be retrieved using md-cli-raw-command, for example, with **show** or **tools** dump commands as the input string. The output returned, however, is an unstructured block of text. Structured state information can be retrieved using the standard NETCONF <get> operation.

5.7 NETCONF Using the Legacy Alcatel-Lucent Base-R13 SR OS YANG Modules

For more details about the Alcatel-Lucent Base-R13 YANG modules, see SR OS YANG Data Models.

This section describes variations in the SR OS NETCONF behavior when it uses the legacy Base-R13 SR OS YANG data modules.

For access to the Alcatel-Lucent Base-R13 SR OS YANG data models, both console and NETCONF access must be configured for the NETCONF user.

Using SR OS NETCONF with the legacy Alcatel-Lucent Base-R13 SR OS YANG modules is not recommended. These modules are supported only until customers currently using them migrate to the NOKIA SR OS YANG modules, then will be deprecated.

5.7.1 Operations and Capabilities

A client establishing a NETCONF session must log into the router so user accounts must exist for NETCONF on SR OS. An access type “netconf” is provided. For access to the Base-R13 SR OS YANG data models, both **console** and **netconf** access must be configured for the NETCONF user.

Table 30 lists only the exceptions from Table 26 when using the Alcatel-Lucent Base-R13 SR OS YANG modules. For more details about the supported operations and capabilities, see Table 26.

Table 30 **Exceptions to Protocol Support for Base-R13 compared to Nokia YANG**

Protocol Operation	Example	Supported	Notes
get-config	<?xml version="1.0" encoding="UTF-8"?> <rpc message-id="101" xmlns="urn:ietf:params:xml:ns:netconf:ba se:1.0"> <get-config> <source> <candidate/> </source> </get-config> </rpc>]]>]]>	No	Not supported with the Alcatel-Lucent Base-R13 SR OS YANG modules.
get	<?xml version="1.0" encoding="UTF-8"?> <rpc message-id="101" xmlns="urn:ietf:params:xml:ns:netconf:ba se:1.0"> <get/> </rpc>]]>]]>	No	Not supported with the Alcatel-Lucent Base-R13 SR OS YANG modules.
edit-config	<target> <running/> </target>	Yes	Can be used when all of the following are true: The Alcatel-Lucent Base-R13 SR OS YANG modules are used. The writeable-running capability is set to "true". The configuration mode is set to "mixed-mode".
	<target> <candidate/> </target>	No	Not supported when using the Alcatel-Lucent Base-R13 SR OS YANG modules.

Table 30 **Exceptions to Protocol Support for Base-R13 compared to Nokia YANG (Continued)**

Protocol Operation	Example	Supported	Notes
	<default-operation>none</default-operation>	Yes	With the Alcatel-Lucent Base-R13 SR OS YANG modules, an operation of "none" on a leaf node (inherited or direct) causes that leaf statement to be ignored. No error is returned if the leaf does not exist in the data model.
get-schema	<?xml version="1.0" encoding="UTF-8"?> <rpc message-id="101" xmlns="urn:ietf:params:xml:ns:netconf:ba se:1.0"> <get-schema xmlns="urn:ietf:params:xml:ns:yang:ietf- netconf-monitoring"> </get-schema> </rpc>]]>]]>	No	Cannot be used to acquire the Alcatel-Lucent Base-R13 SR OS YANG modules.

The following rules apply to using the Alcatel-Lucent Base-R13 SR OS YANG models:

- The <get-config> operation returns non-default configurations by default (the “trim” mode as per RFC 6243, *With-defaults capability for NETCONF*).
- The SR OS NETCONF server supports the “trim” method as the default. Also, it supports the “report-all” method. When the base-r13-modules setting is enabled and all other modules are disabled, the following capability is advertised:

  <capability>urn:ietf:params:netconf:capability:with-defaults:1.0?basic-mode=trim&also-supported=report-all</capability>
- The user must specify at least a top-level tag and a namespace in the filter. If the namespace is not specified, SR OS returns an error.
- A <get-config> request that specifies a non-existent list node or presence container will result in a reply that contains no data for those list nodes or containers. An <rpc-error> is not sent in this case.
- An <edit-config> request must specify the namespace used in the top-level <configure> element because SR OS accepts only a single namespace at the top-level <configure> element. For example:

```
<configure xmlns="urn:alcatel-lucent.com:sros:ns:yang:conf-r13">
    <system>
        ....
```

- Any <edit-config> configuration changes can be made on the <running> datastore only and take immediate operational effect. The Alcatel-Lucent Base-R13 SR OS YANG modules are not applicable to the <candidate> datastore.

The supported <edit-config> operation attribute values for the Alcatel-Lucent Base-R13 SR OS YANG modules are listed in Table 31.

Table 31 **<edit-config> Operation Attribute Values for Alcatel-Lucent Base-R13 SR OS YANG Modules**

Command	Notes
urn:alcatel-lucent.com:sros:ns:yang:conf-*-r13 namespace Alcatel-Lucent Base-R13 SR OS YANG modules	
merge (Base-R13 SR OS modules)	For a merge operation, the operations and tags specified in an <edit-config> request are order-aware and order-dependent, and the sequence of merge operations must follow the required sequence of the equivalent CLI commands. The <edit-config> request is processed and executed in a top-down order. The same leaf can be enabled and disabled multiple times in the request and the final result is whatever was last specified for that leaf in the <edit-config> request.
remove (Base-R13 SR OS modules)	A <remove> operation is not supported for boolean leaves. For example, any of the following example commands will return an error: • <shutdown operation="remove"/> • <shutdown operation="remove">false</shutdown> • <interface operation="remove"> <interface-name>abc</interface-name> <shutdown>true</shutdown> </interface> For this last case, <shutdown operation="merge">true</shutdown> could be used instead to make the request valid. A <remove> operation is the equivalent of **no** command in the CLI. This **no** command is applied whether the default for the command is enabled (command), disabled (**no** command), or a specific value. The <remove> operation is not aware of the default value of the object or leaf being removed. A <remove> operation for a leaf, where the request also specifies a value for the leaf, will result in an error.

Table 31 **<edit-config> Operation Attribute Values for Alcatel-Lucent Base-R13 SR OS YANG Modules (Continued)**

Command	Notes
delete (Base-R13 SR OS modules)	A <delete> operation for a leaf or a presence container will not return an error if the item is already deleted. An error is returned if attempting to delete a list node that does not exist. A <delete> operation for a container without presence will return an error. A <delete> operation is not supported for boolean leaves. For example, any of the following example commands will return an error: • <shutdown operation="delete"/> • <shutdown operation="delete">false</shutdown> • <interface operation="delete"> <interface-name>abc</interface-name> <shutdown>true</shutdown> </interface> For this last case, <shutdown operation="merge">true</shutdown> could be used instead to make the request valid. A <delete> operation is the equivalent of **no** command in the CLI. This **no** command is applied whether the default for the command is enabled (command), disabled (**no** command), or a specific value. The <delete> operation is not aware of the default value of the object/leaf being deleted. A <delete> operation on a node will ignore any values provided for that node (it will not check if that value is configured or valid), and it will ignore any data below that node (it will not check if that data exists or is valid).
create (Base-R13 SR OS modules)	A <create> operation for a leaf or a presence container will not return an error if the item is being set to the same value. An error is returned if attempting to create a list node that already exists. A <create> operation for a container without presence will result in an "OK" response (no error) but will be silently ignored. For a <create> operation, the operations and tags specified in an <edit-config> request are order-aware and order-dependent, and the sequence of create operations must follow the required sequence of the equivalent CLI commands. The <edit-config> request is processed and executed in a top-down order. The same leaf can be enabled and disabled multiple times in the request and the final result is whatever was last specified for that leaf in the <edit-config> request.
replace (Base-R13 SR OS modules)	Not supported

For the <edit-config> operation <default-operation> value of "none", an operation of "none" on a leaf node (inherited or direct) causes that leaf statement to be ignored. No error will be returned if the leaf does not exist in the data model.

If the children branches of an object are required to be removed before deleting the object in the CLI, the equivalent delete request in a NETCONF <edit-config> request must contain all those children if they exist).

In the following example, SR OS shuts down the test interface, deletes the interface, shuts down the VPLS service, and removes it.

```
<?xml version="1.0" encoding="UTF-8"?>
<rpc message-id="101" xmlns="urn:ietf:params:xml:ns:netconf:base:1.0">
    <edit-config>
        <target><running/></target>
        <config>
            <configure xmlns="urn:alcatel-lucent.com:sros:ns:yang:conf-r13">
                <service>
                    <vpls operation="delete">
                        <service-id>11</service-id>
                        <interface>
                            <ip-int-name>test</ip-int-name>
                            <shutdown operation="merge">true</shutdown>
                        </interface>
                        <shutdown operation="merge">true</shutdown>
                    </vpls>
                </service>
            </configure>
        </config>
    </edit-config>
</rpc>
]]>]]>
```

Note: In the urn:alcatel-lucent.com:sros:ns:yang:conf-*-r13 namespace (the Alcatel-Lucent Base-R13 SR OS YANG modules), the operation="merge" is required in the shutdown nodes; otherwise, the inherited operation is delete, which is not supported on boolean leaves.

In the preceding example, if other children of VPLS service 11 exist in the configuration besides the interface test specified in the preceding delete request, and those children are required in the CLI to be deleted before removing VPLS service11, then the deletion request will fail. All configured children must be specified in the delete request.

SR OS supports the "/netconf-state/schemas" subtree but the Alcatel-Lucent Base-R13 SR OS YANG schema is not returned whether the **base-r13-modules** command is set to **true** or **false**.

The <configuration-region/> cannot be used with the Alcatel-Lucent Base-R13 SR OS YANG modules.

5.8 NETCONF Using the CLI Content Layer

When using the CLI format at the NETCONF content layer, configuration and state information is expressed as untagged (non-XML) CLI commands; for example, CLI script.

Access to various CLI config, **show** and **admin** commands using the CLI content layer is controlled by the user profile that is used to authenticate the underlying SSH session.

If a NETCONF request attempts to execute a CLI command that is outside the scope of its access profile, an error response will be sent.

The following example shows a user request where the **show** command usage is outside the scope of the user's access profile.

```
<?xml version="1.0" encoding="UTF-8"?>
<rpc message-id="101" xmlns="urn:ietf:params:xml:ns:netconf:base:1.0">
    <get>
        <filter>
            <oper-data-format-cli-block>
                <cli-show>system security profile</cli-show>
            </oper-data-format-cli-block>
        </filter>
    </get>
</rpc>
]]>]]>
```

The following example shows the reply.

```
 <?xml version="1.0" encoding="UTF-8"?>
<rpc-reply message-id="101" xmlns="urn:ietf:params:xml:ns:netconf:base:1.0">
    <rpc-error>
        <error-type>application</error-type>
        <error-tag>operation-failed</error-tag>
        <error-severity>error</error-severity>
        <error-info>
            <err-element>cli-show</err-element>
        </error-info>
        <error-message>
            command failed - 'show system security profile'
            MINOR: CLI Command not allowed for this user.
        </error-message>
    </rpc-error>
</rpc-reply>
]]>]]>
```

When the Alcatel-Lucent Base-R13 SR OS YANG modules are used, a user can save a rollback checkpoint using the CLI content layer (before doing an <edit-config> or a series of <edit-config>) and perform a rollback revert if needed later using the <cli-action> RPC.

The set of supported actions are as follows:

- **admin rollback compare** [to *checkpoint2*]
- **admin rollback compare** *checkpoint1* to *checkpoint2*
- **admin rollback delete** *checkpoint* | *rescue*
- **admin rollback save** [**comment** *comment*] [**rescue**]
- **admin rollback revert** *checkpoint* | *rescue* [**now**]
- **admin rollback view** [*checkpoint* | *rescue*]

The following example shows two rollback items with responses.

```
<?xml version="1.0" encoding="UTF-8"?>
<rpc message-id="101" xmlns="urn:ietf:params:xml:ns:netconf:base:1.0">
    <cli-action>
        <admin>rollback compare active-cfg to 1</admin>
        <admin>rollback compare</admin>
    </cli-action>
</rpc>
]]>]]>
```

The following example shows the reply.

```
<?xml version="1.0" encoding="UTF-8"?>
<rpc-reply message-id="101" xmlns="urn:ietf:params:xml:ns:netconf:base:1.0">
    <data xmlns="urn:alcatel-lucent.com:sros:ns:yang:cli-content-layer-r13">
        <cli-action>
            <item>
                <admin>rollback compare active-cfg to 1</admin>
                <response>
            0.150 s
            0.450 s
-----------------------------------------------
configure
    router
-       mpls
-           shutdown
-           interface "system"
-               no shutdown
-           exit
-           lsp "test"
-               shutdown
-           exit
-       exit
-       rsvp
-           shutdown
-           interface "system"
-               no shutdown
-           exit
-        exit
    exit
exit
-----------------------------------------------
Finished in 0.720 s
            </response>
```

```
            </item>
            <item>
                <admin>rollback compare</admin>
                <response>
            0.160 s
            0.070 s
-----------------------------------------------
configure
    router
-       mpls
-           shutdown
-            interface "system"
-                no shutdown
-            exit
-            lsp "test"
-                shutdown
-            exit
-        exit
-        rsvp
-            shutdown
-            interface "system"
-                no shutdown
-            exit
-        exit
    exit
    service
-       vpls "99" customer 1 create
-           shutdown
-           stp
-               shutdown
-           exit
-       exit
    exit
exit
-----------------------------------------------
Finished in 0.350 s
                    </response>
                </item>
            </cli-action>
        </data>
    </rpc-reply>
]]>]]>
```

The script must be correctly ordered and has the same dependencies and behavior as CLI. The location of CR/LF (ENTER) within the CLI for an <edit-config> is significant and affects the processing of the CLI commands, such as what CLI branch is considered the “working context”. In the following two examples, the “working context” after the commands are issued are different.

Example 1

```
exit all  [<-ENTER]
configure system time zone EST [<-ENTER]
```

Example 2

```
exit all [<-ENTER]
configure  [<-ENTER]
    system  [<-ENTER]
        time  [<-ENTER]
            zone EST [<-ENTER]
```

After Example 1, the CLI working context is the root and immediately sending “dst-zone CEST” would return an error. After Example 2, the CLI working context is config>system>time and sending “dst-zone CEST” would work as expected.

Configuration changes using NETCONF trigger the same “change” log events (for example, tmnxConfigCreate) as a normal CLI user doing the same changes.

The <with-defaults> tag (RFC 6243, *With-defaults capability for NETCONF*) is not supported in a CLI content layer request.

The operator can get a full configuration (equivalent to the CLI command **admin display-config** [**detail**]), including defaults for a CLI content layer, using an empty <cli-info-detail>. The full configuration can be obtained using a <get-config> request in a CLI content layer format with an empty <cli-info> or <cli-info-detail> tag inside a <config-format-cli-block>. <report-all> is not supported.

Post-processing commands are ignored: "| match" (pipe match), "| count" (pipe count), and ">" (redirect to file) and CLI ranges are not supported for any command; for example, show card [1..5].

5.8.1 CLI Content Layer Examples

The following examples can be used after a NETCONF session has been established including the exchange of the <hello> messages.

5.8.1.1 Example: Configuration Change

The following example shows a configuration change request and response.

Note:

- To successfully push configuration changes using the CLI content layer
  - The writeable-running capability must be set to "true".
  - The configuration mode should not be set to "mixed-mode".
- The **exit all** command is not required at the beginning of the CLI block; it is automatically assumed by the SR OS NETCONF server.

The following example shows a configuration change request and response.

```
<?xml version="1.0" encoding="UTF-8"?>
<rpc message-id="101" xmlns="urn:ietf:params:xml:ns:netconf:base:1.0">
    <edit-config>
    <target><running/></target>
        <config>
            <config-format-cli-block>
                configure system
                    time zone EST
                    location over-here
                exit all
            </config-format-cli-block>
        </config>
    </edit-config>
</rpc>
]]>]]>
```

The following example shows the reply.

```
<?xml version="1.0" encoding="UTF-8"?>
<rpc-reply message-id="101"
xmlns="urn:ietf:params:xml:ns:netconf:base:1.0">
    <ok/>
</rpc-reply>
]]>]]>
```

5.8.1.2 Example: Retrieving Configuration Information

The following example shows a <get-config> request and response to retrieve configuration information.

```
<?xml version="1.0" encoding="UTF-8"?>
<rpc message-id="101" xmlns="urn:ietf:params:xml:ns:netconf:base:1.0">
    <get-config>
        <source>
            <running/>
        </source>
        <filter>
            <config-format-cli-block>
                <cli-info>router</cli-info>
                <cli-info-detail>system login-control</cli-info-detail>
            </config-format-cli-block>
        </filter>
    </get-config>
</rpc>
]]>]]>
```

The following example shows the reply.

```
<?xml version="1.0" encoding="UTF-8"?>
<rpc-reply message-id="101" xmlns="urn:ietf:params:xml:ns:netconf:base:1.0">
    <data>
```

```
        <config-format-cli-block xmlns="urn:alcatel-lucent.com:sros:ns:yang:cli-
content-layer-r13">
            <item>
                <cli-info>router</cli-info>
                <response>
----------------------------------------------
#--------------------------------------------------
echo "IP Configuration"
#--------------------------------------------------
        interface "system"
            no shutdown
        exit
----------------------------------------------
                </response>
            </item>
            <item>
                <cli-info-detail>system login-control</cli-info-detail>
                <response>
----------------------------------------------
            ftp
                inbound-max-sessions 3
            exit
            ssh
                no disable-graceful-shutdown
                inbound-max-sessions 5
                outbound-max-sessions 5
                no ttl-security
            exit
            telnet
                no enable-graceful-shutdown
                inbound-max-sessions 5
                outbound-max-sessions 5
                no ttl-security
            exit
            idle-timeout 30
            no pre-login-message
            no motd
            login-banner
            no exponential-backoff
----------------------------------------------
                </response>
            </item>
        </config-format-cli-block>
    </data>
</rpc-reply>
]]>]]>
```

5.8.1.3 Example: Retrieving Full Configuration Information

The following example shows a <get-config> request and response to retrieve full configuration information.

Note: The <cli-info-detail/> request can be used to get the full configuration, including default settings.

```
<?xml version="1.0" encoding="UTF-8"?>
<rpc message-id="101" xmlns="urn:ietf:params:xml:ns:netconf:base:1.0">
   <get-config>
          <source>
               <running/>
          </source>
          <filter>
              <config-format-cli-block>
                  <cli-info/>
              </config-format-cli-block>
          </filter>
    </get-config>
</rpc>
]]>]]>
```

The following example shows the reply.

```
<?xml version="1.0" encoding="UTF-8"?>
<rpc-reply message-id="101" xmlns="urn:ietf:params:xml:ns:netconf:base:1.0">
    <data>
        <config-format-cli-block xmlns="urn:alcatel-lucent.com:sros:ns:yang:cli-
content-layer-r13">
            <item>
                <cli-info></cli-info>
                <response>

# Generated WED JAN 07 01:07:43 2015 UTC

exit all
configure
#--------------------------------------------------
echo "System Configuration"
#--------------------------------------------------
    system
        dns
        exit
        load-balancing
            lsr-load-balancing lbl-ip
            system-ip-load-balancing
        exit
        netconf
            no shutdown
        exit
        snmp
            shutdown
            engineID "deadbeefdeadbeef"
        exit
        time
            ntp
                authentication-key 1 key "OAwgNUlbzgI" hash2 type des
                no shutdown
            exit
```

```
            sntp
                shutdown
            exit
            zone EST
        exit
        thresholds
            rmon
            exit
        exit
#--------------------------------------------------
echo "Cron Configuration"
#--------------------------------------------------
        cron
            ...
            ...
            ...
        exit
    exit
#--------------------------------------------------
echo "System Security Configuration"
#--------------------------------------------------
    ...
    ...
    ...
#--------------------------------------------------
echo "System Time NTP Configuration"
#--------------------------------------------------
    system
        time
            ntp
            exit
        exit
    exit

exit all

# Finished WED JAN 07 01:07:43 2015 UTC
------------------------------------------------
------------------------------------------------
                </response>
            </item>
        </config-format-cli-block>
    </data>
</rpc-reply>
]]>]]>
```

5.8.1.4 Example: <get> Request

The following example shows a <get> request.

```
<?xml version="1.0" encoding="UTF-8"?>
<rpc message-id="101" xmlns="urn:ietf:params:xml:ns:netconf:base:1.0">
    <get>
        <filter>
            <oper-data-format-cli-block>
```

```
                <cli-show>system security ssh</cli-show>
            </oper-data-format-cli-block>
        </filter>
    </get>
</rpc>
]]>]]>
```

The following example shows the reply.

```
<?xml version="1.0" encoding="UTF-8"?>
<rpc-reply message-id="101" xmlns="urn:ietf:params:xml:ns:netconf:base:1.0">
    <data>
       <oper-data-format-cli-block xmlns="urn:alcatel-lucent.com:sros:ns:yang:cli-
content-layer-r13">
            <item>
                <cli-show>system security ssh</cli-show>
                <response>

===============================================================================
SSH Server
===============================================================================
Administrative State      : Enabled
Operational State         : Up
Preserve Key              : Enabled

SSH Protocol Version 1    : Disabled

SSH Protocol Version 2    : Enabled
DSA Host Key Fingerprint  : ca:ce:37:90:49:7d:cc:68:22:b3:06:2c:11:cd:3c:8e
RSA Host Key Fingerprint  : 49:7c:21:97:42:35:83:61:06:95:cd:a8:78:4c:1e:76

-------------------------------------------------------------------------------
Connection                                Username
    Version Cipher                        ServerName  Status
-------------------------------------------------------------------------------
10.121.143.254                           admin
    2       aes128-cbc                    netconf     connected
-------------------------------------------------------------------------------
Number of SSH sessions : 1
===============================================================================
                </response>
            </item>
        </oper-data-format-cli-block>
    </data>
</rpc-reply>
]]>]]>
```

5.8.1.5 Example: <get> Request with Non-Syntax Error in the Second Item

The following example shows a <get> request with a non-syntax error in the second item.

```
<?xml version="1.0" encoding="UTF-8"?>
<rpc message-id="101" xmlns="urn:ietf:params:xml:ns:netconf:base:1.0">
<get>
    <filter>
        <oper-data-format-cli-block>
            <cli-show>router interface "system"</cli-show>
            <cli-show>router mpls lsp</cli-show>
            <cli-show>system security ssh</cli-show>
        </oper-data-format-cli-block>
    </filter>
</get>
</rpc>
]]>]]>
```

The following example shows the reply.

```
<?xml version="1.0" encoding="UTF-8"?>
<rpc-reply message-id="101" xmlns="urn:ietf:params:xml:ns:netconf:base:1.0">
<data>
   <oper-data-format-cli-block xmlns="urn:alcatel-lucent.com:sros:ns:yang:cli-
content-layer-r13">
        <item>
            <cli-show>router interface "system"</cli-show>
            <response>
===============================================================================
Interface Table (Router: Base)
===============================================================================
Interface-Name    Adm    Opr(v4/v6)     Mode    Port/SapId
    IP-Address                                    PfxState
-------------------------------------------------------------------------------
 system            Up    Up/Down        Network system
    10.23.63.5/32                            n/a
-------------------------------------------------------------------------------
 Interfaces : 1
===============================================================================
                </response>
            </item>
            <item>
                <cli-show>router mpls lsp</cli-show>
                <response>
                    MINOR: CLI MPLS is not configured.
                </response>
                <rpc-error>
                    <error-type>application</error-type>
                    <error-tag>operation-failed</error-tag>
                    <error-severity>error</error-severity>
                    <error-info>
                        <err-element>cli-show</err-element>
                    </error-info>
                    <error-message>
                        command failed - 'show router mpls lsp'
                    </error-message>
                </rpc-error>
            </item>
        </oper-data-format-cli-block>
    </data>
</rpc-reply>
]]>]]>
```

6 Event and Accounting Logs

6.1 Logging Overview

The two primary types of logging supported in the OS are event logging and accounting logs.

Event logging controls the generation, dissemination and recording of system events for monitoring status and troubleshooting faults within the system. The OS groups events into four major categories or event sources.

- Security events — Events that pertain to attempts to breach system security.
- Change events — Events that pertain to the configuration and operation of the node.
- Main events — Events that pertain to applications that are not assigned to other event categories/sources.
- Debug events — Events that pertain to trace or other debugging information.

Events within the OS and have the following characteristics:

- A time stamp in UTC or local time.
- The generating application.
- A unique event ID within the application.
- A router name (also called a vrtr-name) identifying the associated routing context (for example, Base or vprn1000).
- A subject identifying the affected object for the event (e.g. interface name or port identifier).
- A short text description.

Event control assigns the severity for each application event and whether the event should be generated or suppressed. The severity numbers and severity names supported in the OS conform to ITU standards M.3100 X.733 & X.21 and are listed in Table 32.

Table 32 **Event Severity Levels**

Severity Number	Severity Name
1	cleared
2	indeterminate (info)

Table 32 **Event Severity Levels (Continued)**

Severity Number	Severity Name
3	critical
4	major
5	minor
6	warning

Events that are suppressed by event control will not generate any event log entries. Event control maintains a count of the number of events generated (logged) and dropped (suppressed) for each application event. The severity of an application event can be configured in event control.

An event log within the OS associates the event sources with logging destinations. Examples of logging destinations include, the console session, a specific telnet or SSH session, memory logs, file destinations, SNMP trap groups and syslog destinations. A log filter policy can be associated with the event log to control which events will be logged in the event log based on combinations of application, severity, event ID range, router name (vrtr-name), and the subject of the event.

The OS accounting logs collect comprehensive accounting statistics to support a variety of billing models. The routers collect accounting data on services and network ports on a per-service class basis. In addition to gathering information critical for service billing, accounting records can be analyzed to provide insight about customer service trends for potential service revenue opportunities. Accounting statistics on network ports can be used to track link utilization and network traffic pattern trends. This information is valuable for traffic engineering and capacity planning within the network core.

Accounting statistics are collected according to the parameters defined within the context of an accounting policy. Accounting policies are applied to customer Service Access Points (SAPs) and network ports. Accounting statistics are collected by counters for individual service queues defined on the customer’s SAP or by the counters within forwarding class (FC) queues defined on the network ports.

The type of record defined within the accounting policy determines where a policy is applied, what statistics are collected and time interval at which to collect statistics.

The supported destination for an accounting log is a compact flash system device. Accounting data is stored within a standard directory structure on the device in compressed XML format. It is recommended that accounting logs be configured on the cf1: or cf2: devices only. Accounting log files are not recommended on the cf3: device (cf3: is intended to be used primarily for software images and configuration related files).

6.1.1 Logging Using the Management VPRN

When a management VPRN is configured, all authentication, authorization, and accounting servers, DNS server, or syslog servers are reachable using this management VPRN. The user can configure a syslog server within the management VPRN, log all SR OS events, and direct the logs to this syslog server. The command **config**>**log**>**services-all-events** *service id* enables the logging of all events under the management VPRN specified by the service ID. Within the specified VPRN, the command **config**>**service**>**vprn**>**log**>**log-id**>**from** directs the logs from the specified event sources (**main**, **change**, **security**, or **debug-trace**) to be saved at the log ID destination specified by the **config**>**service**>**vprn**>**log**>**log-id**>**to** command.

6.2 Log Destinations

Both event logs and accounting logs use a common mechanism for referencing a log destination. Routers support the following log destinations:

- Console
- Session
- CLI Logs
- Memory Logs
- Log Files
- SNMP Trap Group
- Syslog
- NETCONF

Only a single log destination can be associated with an event log or with an accounting log. An event log can be associated with multiple event sources, but it can only have a single log destination.

A file destination is the only type of log destination that can be configured for an accounting log.

6.2.1 Console

Sending events to a console destination means the message will be sent to the system console The console device can be used as an event log destination.

6.2.2 Session

A session destination is a temporary log destination which directs entries to the active telnet or SSH session for the duration of the session. When the session is terminated, for example, when the user logs out, the “to session” configuration is removed. Event logs configured with a session destination are stored in the configuration file but the “to session” part is not stored. Event logs can direct log entries to the session destination.

6.2.3 CLI Logs

A CLI log is a log that outputs log events to a CLI session. An operator can subscribe to a CLI log from within a CLI session using the **tools perform log subscribe-to log-id** command. The events are sent to the CLI session for the duration of that CLI session (or until an **unsubscribe-from** command is issued).

6.2.4 Memory Logs

A memory log is a circular buffer. When the log is full, the oldest entry in the log is replaced with the new entry. When a memory log is created, the specific number of entries it can hold can be specified, otherwise it will assume a default size. An event log can send entries to a memory log destination.

6.2.5 Log Files

Log files can be used by both event logs and accounting logs and are stored on the compact flash devices in the file system. It is recommended that event and accounting logs be configured on the cf1: or cf2: devices only. Log files are not recommended on the cf3: device (cf3: is intended to be used primarily for software images and configuration related files).

A log file is identified with a single log file ID, but a log file will generally be composed of a number individual files in the file system. A log file is configured with a rollover parameter, expressed in minutes, which represents the length of time an individual log file should be written to before a new file is created for the relevant log file ID. The rollover time is checked only when an update to the log is performed. Thus, complying to this rule is subject to the incoming rate of the data being logged. For example, if the rate is very low, the actual rollover time may be longer than the configured value.

The retention time for a log file specifies the amount of time the file should be retained on the system based on the creation date and time of the file.

When a log file is created, only the compact flash device for the log file is specified. Log files are created in specific subdirectories with standardized names depending on the type of information stored in the log file.

Event log files are always created in the **\log** directory on the specified compact flash device. The naming convention for event log files is:

log *eeff-timestamp*

where:

ee is the event log ID

ff is the log file destination ID

timestamp is the timestamp when the file is created in the form of *yyyymmdd-hhmmss* where:

yyyy is the four-digit year (for example, 2007)

mm is the two digit number representing the month (for example, 12 for December)

dd is the two digit number representing the day of the month (for example, 03 for the 3rd of the month)

hh is the two digit hour in a 24-hour clock (for example, 04 for 4 a.m.)

mm is the two digit minute (for example, 30 for 30 minutes past the hour)

ss is the two digit second (for example, 14 for 14)

Accounting log files are created in the **\act-collect** directory on a compact flash device (specifically *cf1* or *cf2*). The naming convention for accounting log files is nearly the same as for log files except the prefix **act** is used instead of the prefix **log**. The naming convention for accounting logs is:

act aaff-timestamp.xml.gz

where:

aa is the accounting policy ID

ff is the log file destination ID

timestamp is the timestamp when the file is created in the form of *yyyymmdd-hhmmss* where:

yyyy is the four-digit year (for example, 2007)

mm is the two digit number representing the month (for example, 12 for December)

dd is the two digit number representing the day of the month (for example, 03 for the 3rd of the month)

hh is the two digit hour in a 24-hour clock (for example, 04 for 4 a.m.)

mm is the two digit minute (for example, 30 for 30 minutes past the hour)

ss is the two digit second (for example, 14 for 14 seconds)

Accounting logs are .xml files created in a compressed format and have a .gz extension.

The **\act-collect** directory is where active accounting logs are written. When an accounting log is rolled over, the active file is closed and archived in the **\act** directory before a new active accounting log file created in **\act-collect**.

When creating a new log file on a Compact Flash disk card, the system will check the amount of free disk space and that amount must be greater than or equal to the lesser of 5.2 MB or 10% of the Compact Flash disk capacity.

6.2.6 SNMP Trap Group

An event log can be configured to send events to SNMP trap receivers by specifying an SNMP trap group destination.

An SNMP trap group can have multiple trap targets. Each trap target can have different operational parameters.

A trap destination has the following properties:

- The IP address of the trap receiver.

- The UDP port used to send the SNMP trap.
- SNMP version (v1, v2c, or v3) used to format the SNMP notification.
- SNMP community name for SNMPv1 and SNMPv2c receivers.
- Security name and level for SNMPv3 trap receivers.

For SNMP traps that will be sent out-of-band through the Management Ethernet port on the SF/CPM, the source IP address of the trap is the IP interface address defined on the Management Ethernet port. For SNMP traps that will be sent in-band, the source IP address of the trap is the system IP address of the router.

Each trap target destination of a trap group receives the identical sequence of events as defined by the log ID and the associated sources and log filter applied.

6.2.7 Syslog

An event log can be configured to send events to one syslog destination. Syslog destinations have the following properties:

- Syslog server IP address.
- The UDP port used to send the syslog message.
- The Syslog Facility Code (0 to 23) (default 23 - local 7).
- The Syslog Severity Threshold (0 to 7) - events exceeding the configured level will be sent.

Because syslog uses eight severity levels whereas the router uses six internal severity levels, the severity levels are mapped to syslog severities. Table 33 displays the severity level mappings to syslog severities.

Table 33 **Router to Syslog Severity Level Mappings**

SR OS Event Severity	Syslog Severity Numerical Code	Syslog Severity Name	Syslog Severity Definition
—	0	emergency	System is unusable
critical (3)	1	alert	Action must be taken immediately
major (4)	2	critical	Critical conditions
minor (5)	3	error	Error conditions
warning (6)	4	warning	Warning conditions
—	5	notice	Normal but significant condition

Table 33 **Router to Syslog Severity Level Mappings (Continued)**

SR OS Event Severity	Syslog Severity Numerical Code	Syslog Severity Name	Syslog Severity Definition
cleared (1) indeterminate (2)	6	info	Informational messages
—	7	debug	Debug-level messages

The general format of an SR OS syslog message is as follows (see RFC 3164, *The BSD Syslog Protocol*). The “<” and “>” are informational delimiters to make reading and understanding the format easier and they do not appear in the actual syslog message except as part of the PRI:

<PRI> <HEADER><MSG>

where:

- <*PRI*> (the “<” and “>” are included in the syslog message) is the configured facility*8+severity (as described in the *7450 ESS, 7750 SR, 7950 XRS, and VSR System Management Guide* and RFC3164).
- <*HEADER*> is "MMM DD HH:MM:SS <source IP addr>" (without the quotes). There are always 2 characters for the day (DD). Single digit days are preceded with a space character.
- <*MSG*> is <log-prefix>: <seq> <vrtr-name> <application>-<severity>-<Event Name>-<Event ID> [<subject>]: <message>\n

where:

- <*log-prefix*> is an optional 32 characters of text (default = 'TMNX') as configured in the log-prefix command.
- <*seq*> is the log event sequence number (always preceded by a colon and a space char)
- <*vrtr-name*> is vprn1, vprn2, … | Base | management | vpls-management
- <*subject*> may be empty resulting in []:
- \n is the standard ASCII new line character (hex 0A)

Examples (from different nodes):

default log-prefix (TMNX):

```
<188>Jan  2 18:43:23 10.221.38.108 TMNX: 17 Base SYSTEM-WARNING-tmnxStateChange-
2009 [CHASSIS]:  Status of Card 1 changed administrative state: inService,
operational state: outOfService\n
<186>Jan  2 18:43:23 10.221.38.108 TMNX: 18 Base CHASSIS-MAJOR-tmnxEqCardRemoved-
2003 [Card 1]:  Class IO Module : removed\n
```

no log-prefix:

```
<188>Jan 11 18:48:12 10.221.38.108 : 32 Base SYSTEM-WARNING-tmnxStateChange-2009
[CHASSIS]:  Status of Card 1 changed administrative state: inService,
operational state: outOfService\n
<186>Jan 11 18:48:12 10.221.38.108 : 33 Base CHASSIS-MAJOR-tmnxEqCardRemoved-
2003 [Card 1]:  Class IO Module : removed\n
```

log-prefix "test":

```
<186>Jan 11 18:51:22 10.221.38.108 test: 47 Base CHASSIS-MAJOR-tmnxEqCardRemoved-
2003 [Card 1]:  Class IO Module : removed\n
<188>Jan 11 18:51:22 10.221.38.108 test: 48 Base SYSTEM-WARNING-tmnxStateChange-
2009 [CHASSIS]:  Status of Card 1 changed administrative state: inService,
operational state: outOfService\n
```

6.2.8 NETCONF

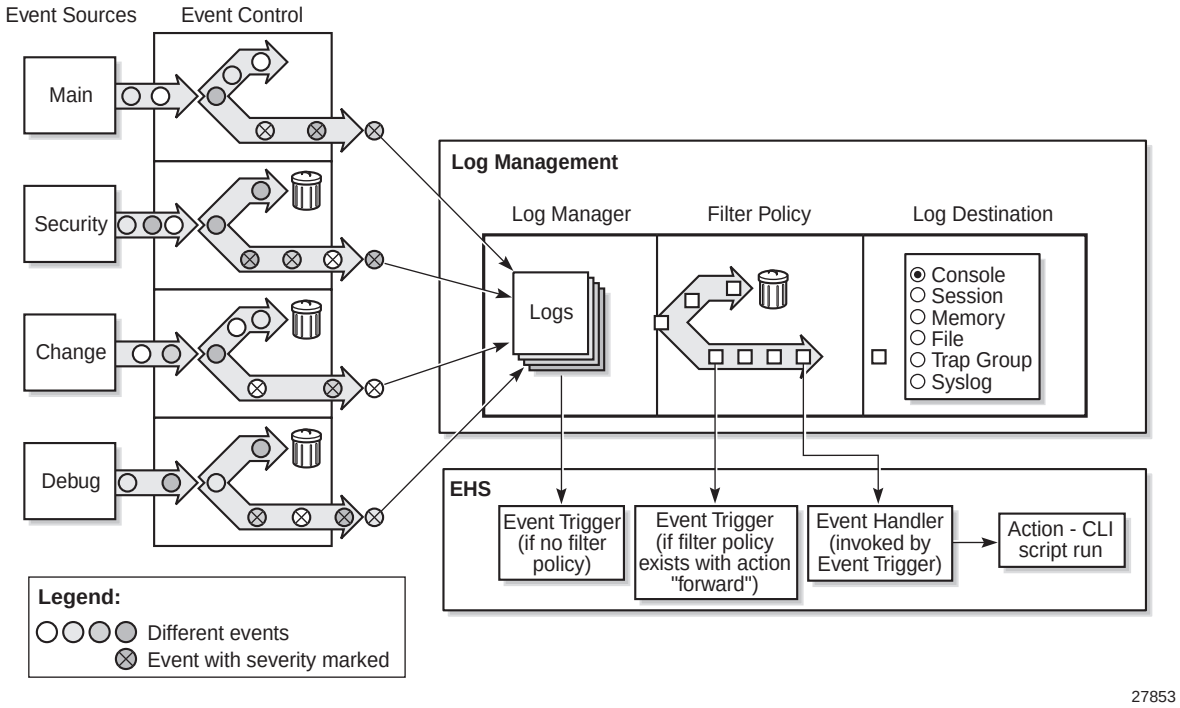
A NETCONF log is a log that outputs log events to a NETCONF session as notifications. A NETCONF client can subscribe to a NETCONF log using the configured **netconf-stream** *stream-name* for the log in a subscription request. See NETCONF Notifications for more details.

6.3 Event Logs

Event logs are the means of recording system generated events for later analysis. Events are messages generated by the system by applications or processes within the router.

Figure 17 depicts a function block diagram of event logging.

Figure 17 **Event Logging Block Diagram**



6.3.1 Event Sources

In Figure 17, the event sources are the main categories of events that feed the log manager.

- Security — The security event source is all events that affect attempts to breach system security such as failed login attempts, attempts to access MIB tables to which the user is not granted access or attempts to enter a branch of the CLI to which access has not been granted. Security events are generated by the SECURITY application and the authenticationFailure event in the SNMP application.
- Change — The change activity event source is all events that directly affect the configuration or operation of the node. Change events are generated by the USER application. The Change event stream also includes the tmnxConfigModify (#2006), tmnxConfigCreate (#2007), tmnxConfigDelete (#2008) and tmnxStateChange (#2009) change events from the SYSTEM application.

- Debug — The debug event source is the debugging configuration that has been enabled on the system. Debug events are generated when debug is enabled for various protocols under the **debug** branch of the CLI (for example, **debug system ntp**).
- Main — The main event source receives events from all other applications within the router.

Examples of applications within the system include IP, MPLS, OSPF, CLI, services, and so on. The following example displays a partial sample of the **show log applications** command output which displays all applications.

```
*A:ALA-48# show log applications
===================================
Log Event Application Names
===================================
Application Name
-----------------------------------
...
BGP
CCAG
CFLOWD
CHASSIS
...
MPLS
MSDP
NTP
...
USER
VRRP
VRTR
===================================
*A:ALA-48#
```

6.3.2 Event Control

Event control pre-processes the events generated by applications before the event is passed into the main event stream. Event control assigns a severity to application events and can either forward the event to the main event source or suppress the event. Suppressed events are counted in event control, but these events will not generate log entries as it never reaches the log manager.

Simple event throttling is another method of event control and is configured similarly to the generation and suppression options. See Simple Logger Event Throttling.

Events are assigned a default severity level in the system, but the application event severities can be changed by the user.

Application events contain an event number and description that explains why the event is generated. The event number is unique within an application, but the number can be duplicated in other applications.

The following example, generated by querying event control for application generated events, displays a partial list of event numbers and names.

```
router# show log event-control
=============================================================================
Log Events
=============================================================================
Application
 ID#    Event Name                          P   g/s     Logged     Dropped
-----------------------------------------------------------------------------

show
BGP:
   2001 bgpEstablished                     MI  gen          1           0
   2002 bgpBackwardTransition              WA  gen          7           0
   2003 tBgpMaxPrefix90                    WA  gen          0           0
...
CCAG:
CFLOWD:
   2001 cflowdCreated                      MI  gen          1           0
   2002 cflowdCreateFailure                MA  gen          0           0
   2003 cflowdDeleted                      MI  gen          0           0
...
CHASSIS:
   2001 cardFailure                        MA  gen          0           0
   2002 cardInserted                       MI  gen          4           0
   2003 cardRemoved                        MI  gen          0           0
...

,,,
DEBUG:
L  2001 traceEvent                         MI  gen          0           0
DOT1X:
FILTER:
   2001 filterPBRPacketsDropped            MI  gen          0           0
IGMP:
   2001 vRtrIgmpIfRxQueryVerMismatch       WA  gen          0           0
   2002 vRtrIgmpIfCModeRxQueryMismatch     WA  gen          0           0
IGMP_SNOOPING:
IP:
L  2001 clearRTMError                      MI  gen          0           0
L  2002 ipEtherBroadcast                   MI  gen          0           0
L  2003 ipDuplicateAddress                 MI  gen          0           0
...
ISIS:
   2001 vRtrIsisDatabaseOverload           WA  gen          0           0
```

6.3.3 Log Manager and Event Logs

Events that are forwarded by event control are sent to the log manager. The log manager manages the event logs in the system and the relationships between the log sources, event logs and log destinations, and log filter policies.

An event log has the following properties:

- A unique log ID — The log ID is a short, numeric identifier for the event log. A maximum of 30 logs can be configured at a time.
- One or more log sources — The source stream or streams to be sent to log destinations can be specified. The source must be identified before the destination can be specified. The events can be from the main event stream, events in the security event stream, or events in the user activity stream.
- One event log destination — A log can only have a single destination (for example, syslog or memory).
- An optional event filter policy — An event filter policy defines whether to forward or drop an event or trap-based on match criteria.

6.3.4 Event Filter Policies

The log manager uses event filter policies to allow fine control over which events are forwarded or dropped based on various criteria. Like other filter policies in the SR OS, filter policies have a default action. The default actions are either:

- Forward
- Drop

Filter policies also include a number of filter policy entries that are identified with an entry ID and define specific match criteria and a forward or drop action for the match criteria.

Each entry contains a combination of matching criteria that define the application, message, event number, router, severity, and subject conditions. The entry’s action determines how the packets should be treated if they have met the match criteria.

Entries are evaluated in order from the lowest to the highest entry ID. The first matching event is subject to the forward or drop action for that entry.

Valid operators are displayed in Table 34:

Table 34 **Valid Filter Policy Operators**

Operator	Description
eq	equal to
neq	not equal to
lt	less than
lte	less than or equal to
gt	greater than
gte	greater than or equal to

A match criteria entry can include combinations of:

- Equal to or not equal to a given system application.
- Equal to or not equal to an event message string or regular expression match.
- Equal to, not equal to, less than, less than or equal to, greater than or greater than or equal to an event number within the application.
- Equal to, not equal to, less than, less than or equal to, greater than or greater than or equal to a severity level.
- Equal to or not equal to a router name string or regular expression match.
- Equal to or not equal to an event subject string or regular expression match.

6.3.5 Event Log Entries

Log entries that are forwarded to a destination are formatted in a way appropriate for the specific destination whether it be recorded to a file or sent as an SNMP trap, but log event entries have common elements or properties. All application generated events have the following properties:

- A time stamp in UTC or local time.
- The generating application.
- A unique event ID within the application.
- A router name (vrtr-name, for example, vprn101 or Base) identifying the router instance that generated the event.
- A subject identifying the affected object.
- A short text description.

The general format for an event in an event log with either a memory, console or file destination is as follows.

```
nnnn <time> TZONE <severity>: <application> #<event_id> <vrtr-name>
 <subject>
<message>
```

The following is an event log example:

```
252 2013/05/07 16:21:00.761 UTC WARNING: SNMP #2005 Base my-interface-abc
"Interface my-interface-abc is operational"
```

The specific elements that compose the general format are described in Table 35.

Table 35 **Log Entry Field Descriptions**

Label	Description
nnnn	The log entry sequence number.
<time>	YYYY/MM/DD HH:MM:SS.SSS
YYYY/MM/DD	The UTC date stamp for the log entry. *YYYY* — Year *MM* — Month *DD* — Date
HH:MM:SS.SSS	The UTC time stamp for the event. *HH* — Hours (24 hour format) *MM* — Minutes *SS.SSS* — Seconds
TZONE	The timezone (for example, UTC, EDT) as configured by **configure log log-id** *x* **time-format**.
<severity>	The severity level name of the event. CLEARED — A cleared event (severity number 1). INFO — An indeterminate/informational severity event (severity level 2). CRITICAL — A critical severity event (severity level 3). MAJOR — A major severity event (severity level 4). MINOR — A minor severity event (severity level 5). WARNING — A warning severity event (severity 6).
<application>	The application generating the log message.
<event_id>	The application's event ID number for the event.
<vrtr-name>	The router name (vrtr-name, for example, vprn101 or Base) in a format used by the logging system representing the router instance that generated the event.

Table 35 **Log Entry Field Descriptions (Continued)**

Label	Description
<subject>	The subject/affected object for the event.
<message>	A text description of the event.

6.3.6 Simple Logger Event Throttling

Simple event throttling provides a mechanism to protect event receivers from being overloaded when a scenario causes many events to be generated in a very short period of time. A throttling rate, # events/# seconds, can be configured. Specific event types can be configured to be throttled. Once the throttling event limit is exceeded in a throttling interval, any further events of that type cause the dropped events counter to be incremented. Dropped events counts are displayed by the **show>log>event-control** context. Events are dropped before being sent to one of the logger event collector tasks. There is no record of the details of the dropped events and therefore no way to retrieve event history data lost by this throttling method.

A particular event type can be generated by multiple managed objects within the system. At the point this throttling method is applied the logger application has no information about the managed object that generated the event and cannot distinguish between events generated by object “A” from events generated by object “B”. If the events have the same event-id, they are throttled regardless of the managed object that generated them. It also does not know which events may eventually be logged to destination log-id <n> from events that will be logged to destination log-id <m>.

Throttle rate applies commonly to all event types. It is not configurable for a specific event-type.

A timer task checks for events dropped by throttling when the throttle interval expires. If any events have been dropped, a TIMETRA-SYSTEM-MIB::tmnxTrapDropped notification is sent.

6.3.7 Default System Log

Log 99 is a pre-configured memory-based log which logs events from the main event source (not security, debug, and so on). Log 99 exists by default.

The following example displays the log 99 configuration.

```
ALA-1>config>log# info detail
#------------------------------------------
echo "Log Configuration "
#------------------------------------------
...
        snmp-trap-group 7
        exit
...
        log-id 99
            description "Default system log"
            no filter
            from main
            to memory 500
            no shutdown
        exit
----------------------------------------------
ALA-1>config>log#
```

6.3.8 Event Handling System

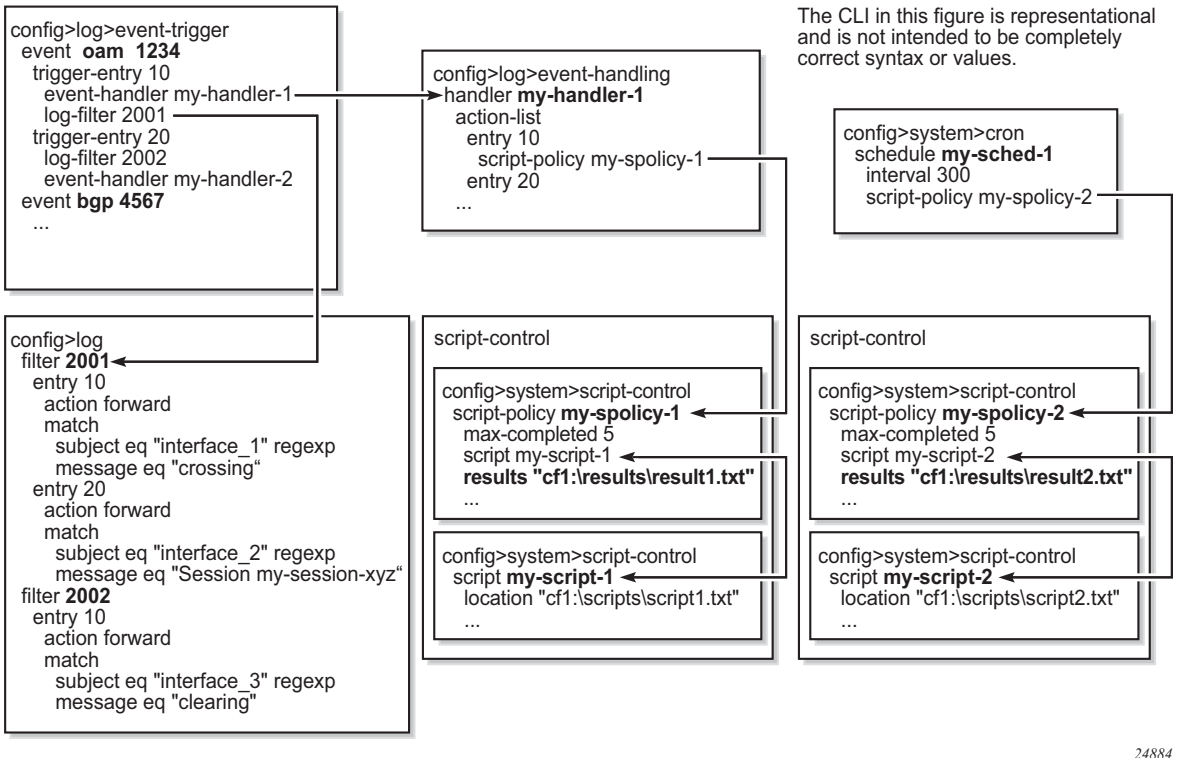
The Event Handling System (EHS) is a framework that allows operator-defined behavior to be configured on the router. EHS adds user-controlled programmatic exception handling by allowing a CLI script to be executed upon the detection of a log event (the 'trigger'). Regexp style expression matching is available on various fields in the log event to give flexibility in the trigger definition.

EHS handler objects are used to tie together:

- trigger events (typically log events that match some configurable criteria)
- a set of actions to perform (typically one or more CLI scripts)

EHS, along with CRON, uses the generic SR OS CLI script-control functions for scripts. Any command available in CLI (with some limited exceptions such as 'candidate' commands) can be executed in a script as the result of an EHS handler being triggered. Figure 18 illustrates the relationships between the different configurable objects used by EHS (and CRON).

Figure 18 **EHS Object Relationships**



6.3.8.1 EHS Configuration and Syntax Rules

Complex rules can be configured to match log events as a trigger for an EHS handler.

When a log event is generated in SR OS, it is subject to discard using suppression and throttling (**config**>**log**>**event-control**) before it is evaluated as a trigger for EHS, according to the following.

- EHS does not trigger on log events that are suppressed through **config**>**log**>**event-control**.
- EHS does not trigger on log events that are throttled by the logger.

EHS is triggered on log events that are dropped by user-configured log filters that are assigned to individual logs (**config**>**log**>**filter**). The EHS event trigger logic occurs before the distribution of log event streams into individual logs.

Varbinds are variable bindings that represent the variable number of values that are included in a log event.

The common parameters and varbinds for a triggering log event are passed in to the triggered EHS script and can be used within the EHS script as passed-in (dynamic) variables. Passed-in (dynamic) variables are:

- the common event parameters, for example: appid, name, eventid, severity, subject, and gentime
- the predefined varbinds in a log event

For example, the following are the passed-in (dynamic) variables for an event with *N* varbinds:

- appid
- eventid
- severity
- subject
- gentime
- event_varbind_1
- event_varbind_2

  ...

  ...
- event_varbind_*N*

Note:

- For more information about showing event parameters, see the **show** command in the *7450 ESS, 7750 SR, 7950 XRS, and VSR Clear, Show, and Tools Command Reference Guide*.
- Refer to the *7450 ESS, 7750 SR, 7950 XRS, and VSR Log Events Guide* for any event's predefined varbinds
- The passed-in event's **gentime** is always UTC
- The event's sequence number is not passed in to the script

When using the classic CLI, an EHS script has the ability to define local (static) variables and uses basic .if or .set syntax inside the script. The use of variables with .if or .set commands within an EHS script adds more logic to the EHS scripting and allows the reuse of a single EHS script for more than one trigger or action.

Both passed-in and local variables can be used within an EHS script either as part of the CLI commands or as part of the .if or .set commands.

The following applies to both CLI commands and .if or .set commands (where *X* represents a variable).

- Using $*X*, without using single or double quotes, replaces the variable *X* with its string or integer value.
- Using “*X*”, with double quotes, means the actual string *X*.
- Using “$*X*”, with double quotes, replaces the variable *X* with its string or integer value.
- Using ‘*X*’, with single quotes does not replace the variable X with its value but means the actual string $*X*.

This means the following interpretation of single and double quotes applies.

- All characters within single quotes are interpreted as string characters.
- All characters within double quotes are interpreted as string characters except for $, which replaces the variable with its value (for example, shell expansion inside a string).

6.3.8.2 Examples of EHS Syntax Supported in Classic CLI

This section describes the supported EHS syntax for the classic CLI.

Note: These scenarios use pseudo syntax.

- .if $*string_variable*==*string_value_or_string_variable* {
  CLI_commands_set1
  .} else {
  CLI_commands_set2
  .} endif
- .if ($*string_variable*==*string_value_or_string_variable*) {
  CLI_commands_set1
  .} else {
  CLI_commands_set2
  .} endif
- .if $*integer_variable*==*integer_value_or_integer_variable* {
  CLI_commands_set1
  .} else {
  CLI_commands_set2

.} endif

- .if ($*integer_variable*==*integer_value_or_integer_variable*) {
  CLI_commands_set1
  .} else {
  CLI_commands_set2
  .} endif
- .if $*string_variable*!=*string_value_or_string_variable* {
  CLI_commands_set1
  .} else {
  CLI_commands_set2
  .} endif
- .if ($*string_variable*!=*string_value_or_string_variable*) {
  CLI_commands_set1
  .} else {
  CLI_commands_set2
  .} endif
- .if $*integer_variable*!=*integer_value_or_integer_variable* {
  CLI_commands_set1
  .} else {
  CLI_commands_set2
  .} endif
- .if ($*integer_variable*!=*integer_value_or_integer_variable*) {
  CLI_commands_set1
  .} else {
  CLI_commands_set2
  .} endif
- .set $*string_variable* = *string_value_or_string_variable*
- .set ($*string_variable* = *string_value_or_string_variable*)
- .set $*integer_variable* = *integer_value_or_integer_variable*
- .set ($*integer_variable* = *integer_value_or_integer_variable)*

where:

- *CLI_commands_set1* is a set of one or more CLI commands
- *CLI_commands_set2* is a set of one or more CLI commands
- *string_variable* is a local (static) string variable

- *string_value_or_string_variable* is a string value/variable
- *integer_variable* is a local (static) integer variable
- *integer_value_or_integer_variable* is an integer value/variable

Note:

- A limit of 100 local (static) variables per EHS script is imposed. Exceeding this limit may result in an error and partial execution of the script.
- When a set statement is used to set a string_variable to a string_value, the string_value can be any non-integer value not surrounded by single or double quotes or it can be surrounded by single or double quotes.
- A "." preceding a directive (for example, if, set, and so on) is always expected to start a new line.
- An end of line is always expected after {.
- A CLI command is always expected to start a new line.
- Passed-in (dynamic) variables are always read-only inside an EHS script and cannot be overwritten using a set statement.
- .if commands support == and != operators only.
- .if and .set commands support addition, subtraction, multiplication, and division of integers.
- .if and .set commands support addition, which means concatenation, of strings.

6.3.8.3 Valid Examples of EHS Syntax in Classic CLI

This section provides a list of valid examples to trigger log events using EHS syntax in classic CLI.

- configure service epipe $serviceID

  where *$serviceID* is either a local (static) integer variable or passed-in (dynamic) integer variable
- echo srcAddr is $srcAddr

  where *$srcAddr* is a passed-in (dynamic) string variable
- .set $ipAddr = "10.0.0.1"

  where *$ipAddr* is a local (static) string variable
- .set $ipAddr = $srcAddr

  where *$srcAddr* is a passed-in (dynamic) string variable

  $ipAddr is a local (static) string variable.
- .set ($customerID = 50)

  where *$customerID* is a local (static) integer variable

- .set ($totalPackets = $numIngrPackets + $numEgrPackets)

  where *$totalPackets*, *$numIngrPackets*, *$numEgrPackets* are local (static) integer variables
- .set ($portDescription = $portName + $portLocation)

  where *$portDescription*, *$portName*, *$portLocation* are local (static) string variables
- if ($srcAddr == "CONSOLE") {

  CLI_commands_set1

  .else {

  CLI_commands_set2

  .} endif

  where *$srcAddr* is a passed-in (dynamic) string variable

  CLI_commands_set1 is a set of one or more CLI commands

  CLI_commands_set2 is a set of one or more CLI commands
- .if ($customerId == 10) {

  CLI_commands_set1

  .else {

  CLI_commands_set2

  .} endif

  where *$customerID* is a passed-in (dynamic) integer variable

  CLI_commands_set1 is a set of one or more CLI commands

  CLI_commands_set2 is a set of one or more CLI commands
- .if ($numIngrPackets == $numEgrPackets) {

  CLI_commands_set1

  .else {

  CLI_commands_set2

  .} endif

  where *$numIngrPackets* and *$numEgrPackets* are local (static) integer variables

  CLI_commands_set1 is a set of one or more CLI commands

  CLI_commands_set2 is a set of one or more CLI commands

6.3.8.4 Invalid Examples for EHS Syntax in Classic CLI

This section provides a list of invalid variable use in EHS syntax in classic CLI.

- .set $srcAddr = "10.0.0.1"
  where *$srcAddr* is a passed-in (dynamic) string variable
  Reason: passed-in variables are read only inside an EHS script.
- .set ($ipAddr = $numIngrPackets + $numEgrPackets)
  where *$ipAddr* is a local (static) string variable
  $numIngrPackets and *$numEgrPackets* are local (static) integer variables
  Reason: variable types do not match, cannot assign a string to an integer.
- .set ($numIngrPackets = $ipAddr + $numEgrPackets)
  where *$ipAddr* is a local (static) string variable
  $numIngrPackets and *$numEgrPackets* are local (static) integer variables
  Reason: variable types do not match, cannot concatenate a string to an integer.
- .set $ipAddr = "10.0.0.1"100
  where *$ipAddr* is a local (static) string variable
  Reason: when double quotes are used, they have to surround the entire string.
- .if ($totalPackets == "10.1.1.1") {
  .} endif
  where *$totalPackets* is a local (static) integer variables
  Reason: cannot compare an integer variable to a string value.
- .if ($ipAddr == 10) {
  .} endif
  where *$ipAddr* is a local (static) string variable
  Reason: cannot compare a string variable to an integer value.
- .if ($totalPackets == $ipAddr) {
  where *$totalPackets* is a local (static) integer variables
  $ipAddr is a local (static) string variable
  Reason: cannot compare an integer variable to a string variable.

6.3.8.5 EHS Debounce

EHS debounce (also called dampening) is the ability to trigger an action (for example an EHS script), if an event happens (N) times within a specific time window (S).

N = [2..15]

S = [1..604800]

Note:

- Triggering occurs with the Nth event not at the end of S.
- There is no sliding window (for example a trigger at Nth event, N+1 event, and so on), as N is reset after a trigger and count is restarted.
- When EHS debouncing or dampening is used, the varbinds passed in to an EHS script at script triggering time are from the Nth event occurrence (the Nth triggering event).
- If S is not specified then the SR OS will continue to trigger every Nth event.

For example:

When linkDown occurs N times in S seconds, an EHS script is triggered to shut down the port.

6.3.8.6 Executing EHS or CRON Scripts

The execution of EHS or CRON scripts depends on the CLI engine associated with the configuration mode. The EHS or CRON script execution engine is based on the primary CLI engine set by the CLI command **configure system management-interface cli cli-engine**.

For example, if **cli-engine** is configured to **classic-cli md-cli**, the script executes in the classic CLI infrastructure and disregards the configuration mode, even if it is model-driven.

The following is the default behavior of the EHS or CRON scripts, depending on the configuration mode.

- **Classic CLI configuration mode**

  EHS or CRON scripts execute in the classic CLI environment and an error occurs if any model-driven CLI commands exist.
- **Model-driven configuration mode**

EHS or CRON scripts execute in the MD-CLI environment and an error occurs if any classic CLI commands exist.

- **Mixed configuration mode**

  EHS or CRON scripts execute in the classic CLI environment and an error occurs if any model-driven CLI commands exist.

EHS or CRON scripts that contain MD-CLI commands can be used in the MD-CLI as follows:

- scripts can be configured
- scripts can be created, edited, and results read through FTP
- scripts can be triggered and executed
- scripts generate an error if there are any non MD-CLI commands or any.if or .set syntax in the script

User authorization for EHS or CRON scripts can be configured in either the classic CLI or the MD-CLI as follows:

- classic CLI

  config>**system**>**security**>**cli-script**>**authorization**>**event-handler**>**cli-user** *user-name*
- MD-CLI

  configure system security cli-script authorization event-handler cli-user *user-name*

When a user is not specified, an EHS or CRON script bypasses authorization and can execute all commands.

In all configuration modes, a script policy can be disabled (using the MD-CLI command **configure system script-control script-policy** *policy-name* **admin-state disable** or the classic CLI command **configure system script-control script-policy** *policy-name* **shutdown**) even if history exists. When the script policy is disabled, the following applies.

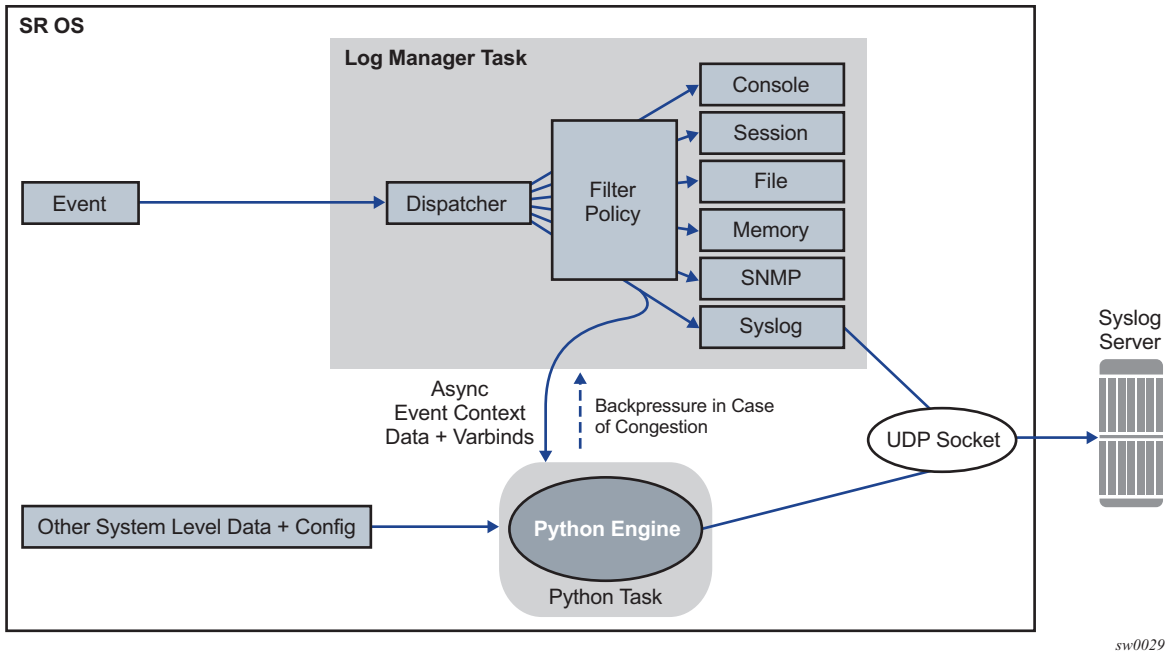
- Newly triggered EHS or CRON scripts are not allowed to execute or queue.
- In-progress EHS or CRON scripts are allowed to continue.
- Already queued EHS or CRON scripts are allowed to execute.

By default, a script policy is configured to allow an EHS or CRON script to override datastore locks from any model-driven interface (MD-CLI, NETCONF, and so on) in mixed and model-driven modes. A script policy can be configured to not allow EHS or CRON scripts to override datastore locks (using the MD-CLI command **configure system script-control script-policy** *policy-name* **lock-override false** command or the classic CLI **configure system script-control script-policy** *policy-name* **no lock-override** command).

6.4 Customizing Syslog Messages Using Python

Log events in SR OS can be customized by a Python script before they are sent to a syslog server. The log events that are subject to Python processing are selected via log filters. This allows only a preferred subset of log messages to be customized (Figure 19).

Figure 19 **Interaction between the Logger and the Python Engine**



6.4.1 Python Engine for Syslog

This section discusses syslog-specific aspects of Python processing. Refer to the “Python Script Support for ESM” section of the *7450 ESS, 7750 SR, and VSR Triple Play Service Delivery Architecture Guide* for an introduction to Python.

When an event is dispatched to the log manager in SR OS, the log manager asynchronously passes the event context data and varbinds to the Python engine, that is, the logger task is not waiting for feedback from Python. Varbinds are variable bindings that represent the variable number of values that are included in the event. Each varbind consists of a triplet (OID, type, value). Along with other system-level variables, the Python engine constructs a syslog message and sends it to the syslog destination. During this process, the operator can modify the format of the syslog message or leave it intact, as if it was generated by the syslog process within the log manager.

The tasks of the Python engine in a syslog context are as follows:

- assembles custom syslog messages (including PRI, HEADER and MSG fields) based on the received event context data, varbinds specific to the event, system-level data, and the configuration parameters (syslog server IP address, syslog facility, log-prefix and the destination UDP port)
- reformats timestamps in a syslog message
- sends the original or modified message to the syslog server
- drops the message

6.4.1.1 Python Syslog APIs

Python APIs are used to assemble a syslog message which, in SR OS, has the following generic format:

```
<PRI> <HEADER><MSG>
```

where:

- <**PRI**> (the “<” and “>” are included in the syslog message) is the configured facility x 8+severity (as described in the *7450 ESS, 7750 SR, 7950 XRS, and VSR System Management Guide* and RFC 3164)
- <**HEADER**> is MMM DD HH:MM:SS <hostname>. There are always two characters for the day (DD). Single digit days are preceded with a space character.

- <**MSG**> is <log-prefix>: <seq> <router-name> <application>-<severity>-<Event Name>-<Event ID> [<subject>]: <message>\n

where:

- <*log-prefix*> is an optional set of 32 characters (default = 'TMNX') as configured in the **log-prefix** command
- <*seq*> is the log event sequence number. It always preceded by a colon and a space character.
- <*router-name*> is the name of the router, for example, vprn1, vprn2, Base, management, vpls-management
- <*subject*> is the topic and can be empty, resulting in []:
- \n is the standard ASCII new line character (hex 0A)

Table 36 describes Python information that can be used to manipulate syslog messages.

Table 36 **Manipulating Python Syslog Messages**

Imported Nokia (ALC) Modules	Access Rights	Comments
event (from alc import event)	—	The method used to retrieve generic event information.
syslog (from alc import syslog)	—	The method used to retrieve syslog-specific parameters.
system (from alc import system)	—	The method used to retrieve system-specific information. Currently, the only parameter retrieved is the system name.
Events use the following format as they are written into memory, file, console, and system: nnnn <time> <severity>:<application> # <event_id> <router-name> <subject> <message> The event-related information received in the context data from the log manager is retrieved via the following Python methods:		
event.sequence	RO	The sequence number of the event (nnnn).
event.timestamp	RO	The timestamp of the event. (YYYY/MM/DD HH:MM:SS.SS).
event.routerName	RO	The router name, for example, BASE, VPRN1, and so on.
event.application	RO	The application generating the event, for example, NA.

Table 36 **Manipulating Python Syslog Messages (Continued)**

Imported Nokia (ALC) Modules	Access Rights	Comments
event.severity	RO	The severity of the event. This is configurable in SR OS (CLEARED [1], INFO [2], CRITICAL [3], MAJOR [4], MINOR [5], WARNING [6]).
event.eventId	RO	The event ID, for example, 2012.
event.eventName	RO	The event Name, for example, tmnxNatPlBloclAllocationLsn.
event.subject	RO	An optional field, for example, [NAT].
event.message	RO	The event-specific message, for example, "{2} Map 192.168.20.29 [2001-2005] MDA 1/2 -- 276824064 classic-lsn-sub %3 vprn1 10.10.10.101 at 2015/08/31 09:20:15".
Syslog Methods		
syslog.hostName	RO	The IP address of the SR OS node sending the syslog message. This is used in the Syslog HEADER.
syslog.logPrefix	RO	The log prefix which is configurable and optional, for example, TMNX:
syslog.severityToPRI(event.severity)	—	The Python method used to derive the PRI field in syslog header based on event severity and a configurable syslog facility.
syslog.severityToName(event.severity)	—	An SR OS event severity to syslog severity name. For more information, see the Syslog section.
syslog.timestampToUnix(timestamp)	—	The Python method that takes a timestamp in the format if YYYY/MM/DD HH:MM:SS and converts it into a UNIX-based format (seconds since Jan 01 1970 – UTC).
syslog.set(newSyslogPdu)	—	The Python method used to send the syslog message in the newSyslogPdu. This variable must be constructed manually via string manipulation. In the absence of the command, the SR OS assembles the default syslog message (as if Python was not configured) and sends it to the syslog server, assuming that the message is not explicitly dropped.

Table 36 **Manipulating Python Syslog Messages (Continued)**

Imported Nokia (ALC) Modules	Access Rights	Comments
syslog.drop()	—	The Python method used to drop a syslog message. This method must be called before the syslog.set<newSyslogPdu method.
System Methods		
system.name	RO	The Python method used to retrieve the system name

For example, assume that the syslog format is:

```
<PRI><timestamp> <hostname> <log-prefix>: <sequence> <router-name>  <appid>-
<severity>-<name>-<eventid> [<subject>]: <text>
```

Then the following is an example of the syslogPdu constructed via Python:

```
syslogPdu = "<" + syslog.severityToPRI(event.severity) + ">" \ + event.timestamp + "
 " \ + syslog.hostname + " " + syslog.logPrefix + ": " + \ event.sequence + " " + ev
ent.routerName + " " + \  event.application + "-
" + \ syslog.severityToName(event.severity) + "-" + \
               event.eventName + "-" + event.eventId + " [" + \
               event.subject + "]: " + event.message
```

6.4.1.2 Timestamp Format Manipulation

Certain logging environments require customized formatting of the timestamp. Nokia provides a timestamp conversion method in the alu.syslog Python module to convert a timestamp from the format YYYY/MM/DD hh:mm:ss into a UNIX-based timestamp format (seconds since Jan 01 1970 – UTC).

For example, an operator can use the following Python method to convert a timestamp from the YYYY/MM/DD hh:mm:ss.ss or YYYY/MM/DD hh:mm:ss (no centiseconds) format into either the UNIX timestamp format or the MMM DD hh:mm:ss format.

```
from alc import event
from alc import syslog
from alc import system
#input format: YYYY/MM/DD hh:mm:ss.ss  or YYYY/MM/DD hh:mm:ss
#output format 1: MMM DD hh:mm:ss
#output format 2: unixTimestamp (TBD)
def timeFormatConversion(timestamp,format):
    if format not in range(1,2):
        raise NameError('Unexpected format, expected:' \
                        '0<format<3 got: '+str(format))
```

```
    try:
        dat,tim=timestamp.split(' ')
    except:
        raise NameError('Unexpected timestamp format, expected:' \
                        'YYYY/MM/DD hh:mm:ss got: '+timestamp)
    try:
        YYYY,MM,DD=dat.split('/')
    except:
        raise NameError('Unexpected timestamp format, expected:' \
                        'YYYY/MM/DD hh:mm:ss got: '+timestamp)
    try:
        hh,mm,ss=tim.split(':')
        ss=ss.split('.')[0]   #just in case that the time format is hh:mm:ss.ss
    except:
        raise NameError('Unexpected timestamp format, expected:' \
                        'YYYY/MM/DD hh:mm:ss got: '+timestamp)
    if not (1970<=int(YYYY)<2100 and
            1<=int(MM)<=12 and
            1<=int(DD)<=31 and
            0<=int(hh)<=24 and
            0<=int(mm)<=60 and
            0<=int(ss)<=60):
        raise NameError('Unexpected timestamp format, or values out of the range' \
                        'Expected: YYYY/MM/DD hh:mm:ss got: '+timestamp)
    if format == 1:
        MMM={1:'Jan',
             2:'Feb',
             3:'Mar',
             4:'Apr',
             5:'May',
             6:'Jun',
             7:'Jul',
             8:'Aug',
             9:'Sep',
             10:'Oct',
             11:'Nov',
             12:'Dec'}[int(MM)]
        timestamp=MMM+' '+DD+' '+hh+':'+mm+':'+ss
    if format == 2:
        timestamp=syslog.timestampToUnix(timestamp)
    return timestamp
```

The timeFormatConversion method can accept the event.timestamp value in the format:

```
YYYY/MM/DD HH:MM:SS.SS
```

and return a new timestamp in the format determined by the format parameter:

```
1 ? MMM DD HH:MM:SS
2 ? Unix based time format
```

This method accepts the input format in either of the two forms YYYY/MM/DD HH:MM:SS.SS or YYYY/MM/DD HH:MM:SS and simply ignores the centisecond part in the former form.

6.4.2 Python Processing Efficiency

Python retrieves event-related variables from the log manager, as opposed to retrieving pre-assembled syslog messages. This eliminates the need for string parsing of the syslog message to manipulate it constituent parts increasing the speed of Python processing.

To further improve processing performance, Nokia recommends performing string manipulation via the Python native string method, when possible.

6.4.3 Python Backpressure

A Python task assembles syslog messages based on the context information received from the logger and sends them to the syslog server independent of the logger. If the Python task is congested due to a high volume of received data, the backpressure should be sent to the ISA so that the ISA stops allocating NAT resources. This behavior matches the current behavior in which NAT resources allocation is blocked if that logger is congested.

6.4.4 Event Selection for Python Processing

Events destined for Python processing are configured through a log ID that references a Python policy. The selection of the events are performed via a filter associated with this log ID. The remainder of the events destined to the same syslog server can bypass Python processing by redirecting them to a different log ID. The following example clarifies this point:

1. Creating the Python policy

```
A:dut-a# configure python python-policy PyForLogEvents create
*A:dut-a>config>python>py-policy$
 [no] description    - Configure the description of this policy
 [no] dhcp           - Configure scripts to handle dhcp messages jritter
 [no] dhcp6          - Configure scripts to handle dhcp6 messages
 [no] diameter       - Configure scripts to handle diameter messages
 [no] gtpv1-c        - Configure scripts to handle GTPv1-C messages
 [no] gtpv2-c        - Configure scripts to handle GTPv2-C messages
 [no] pppoe          - Configure scripts to handle PPPoE messages
 [no] radius         - Configure scripts to handle RADIUS messages
 [no] vsd            - Configure scripts to handle VSD messages
 [no] syslog         - Configure a script to handle outgoing syslog messages
*A:dut-a>config>python>py-policy$ syslog
  - syslog script <name>
  - no syslog
<name> :[32 chars max]
```

The detailed Python policy description is explained in the “Python Script Support for ESM” section in the *7450 ESS, 7750 SR, and VSR Triple Play Service Delivery Architecture Guide*.

2. Log filters identify the events that are subject to Python processing

```
A:dut-a>config>log# info
----------------------------------------------
        filter 6
            default-action drop
            entry 1
                action forward
                match
                    application eq "nat"
                    number eq 2012
                exit
            exit
        exit
        filter 7
            default-action forward
            entry 1
                action drop
                match
                    application eq "nat"
                    number eq 2012
                exit
            exit
        exit
```

3. Syslog destination

```
 syslog 1
            address 192.168.1.1
        exit
```

4. Applying Python syslog policy to selected events via filter 6:

```
        log-id 33  Note: Process log events with id of 2012 with Python before
 sending them to syslog server.
            filter 6
            from main
            to syslog 1
            python-policy "PyForLogEvents"
            no shutdown
        exit
        log-id 34  Note: Log events that are not processed by Python.
            filter 7
            from main
            to syslog 1
            no shutdown
        exit
```

In the example above, the configuration-only event 2012 from application "nat" will be sent to log-id 33. All other events are forwarded to the same syslog destination via log-id 34, without any modification. As a result, all events (modified via log-id 33 and unmodified via log-id 34) are sent to the syslog 1 destination.

This configuration may cause reordering of syslog messages at the syslog 1 destination due to slight delay of messages processed by Python.

6.4.5 Modifying a Log File

The following displays the current log configuration:

```
ALA-12>config>log>log-id# info
----------------------------------------------
...
log-id 2
            description "This is a test log file."
            filter 1
            from main security
            to file 1
exit
...
----------------------------------------------
ALA-12>config>log>log-id#
```

The following displays an example to modify log file parameters:

```
Example:config# log
 config>log# log-id 2
 config>log>log-id# description "Chassis log file."
 config>log>log-id# filter 2
 config>log>log-id# from security
 config>log>log-id# exit
```

The following displays the modified log file configuration:

```
A:ALA-12>config>log# info
----------------------------------------------
...
log-id 2
            description "Chassis log file."
            filter 2
            from security
            to file 1
exit
...
----------------------------------------------
A:ALA-12>config>log#
```

6.4.6 Deleting a Log File

The log ID must be shutdown first before it can be deleted. In a previous example, **file 1** is associated with **log-id 2**.

```
A:ALA-12>config>log# info
----------------------------------------------
file-id 1
            description "LocationTest."
            location cf1:
            rollover 600 retention 24
        exit
...
log-id 2
            description "Chassis log file."
            filter 2
            from security
            to file 1
exit
...
----------------------------------------------
A:ALA-12>config>log#
```

The following displays an example to delete a log file:

```
Example:config# log
 config>log# log-id 2
 config>log>log-id# shutdown
 config>log>log-id# exit
 config>log# no log-id 2
```

6.4.7 Modifying a File ID

The following displays the current log configuration:

```
A:ALA-12>config>log# info
----------------------------------------
        file-id 1
            description "This is a log file."
            location cf1:
            rollover 600 retention 24
        exit
----------------------------------------
A:ALA-12>config>log#
```

The following displays an example to modify log file parameters:

```
Example:config# log
 config>log# file-id 1
 config>log>file-id# description "LocationTest."
 config>log>file-id# rollover 2880 retention 500
 config>log>file-id# exit
```

The following displays the file modifications:

```
A:ALA-12>config>log# info
----------------------------------------------
...
file-id 1
    description "LocationTest."
    rollover 2880 retention 500
exit
...
----------------------------------------------
A:ALA-12>config>log#
```

The following displays an example to modify log file parameters:

```
Example:config# log
 config>log# file-id 1
 config>log>file-id# description "LocationTest."
 config>log>file-id# location cf2:
 config>log>file-id# rollover 2880 retention 500
 config>log>file-id# exit
```

The following displays the file modifications:

```
A:ALA-12>config>log# info
----------------------------------------------
...
file-id 1
            description "LocationTest."
            location cf2:
            rollover 2880 retention 500
        exit
...
----------------------------------------------
A:ALA-12>config>log#
```

6.4.8 Modifying a Syslog ID

The following displays an example of the syslog ID modifications:

```
Example:config# log
 config>log# syslog 1
 config>log>syslog$ description "Test syslog."
 config>log>syslog# address 10.10.0.91
 config>log>syslog# facility mail
 config>log>syslog# level info
```

The following displays the syslog configuration:

```
A:ALA-12>config>log# info
----------------------------------------------
...
        syslog 1
            description "Test syslog."
            address 10.10.10.91
            facility mail
            level info
        exit
...
----------------------------------------------
A:ALA-12>config>log#
```

6.4.9 Modifying an SNMP Trap Group

The following displays the current SNMP trap group configuration:

```
A:ALA-12>config>log# info
----------------------------------------------
...
snmp-trap-group 10
    trap-target 10.10.10.104:5 "snmpv3" notify-community "coummunitystring"
exit
...
----------------------------------------------
A:ALA-12>config>log#
```

The following displays an example of the command usage to modify an SNMP trap group:

```
Example:config# log
 config>log# snmp-trap-group 10
 config>log>snmp-trap-group# no trap-target
 10.10.10.104:5
 config>log>snmp-trap-group# snmp-trap-group# trap-
 target 10.10.0.91:1 snmpv2c notify-community "com1"
```

The following displays the SNMP trap group configuration:

```
A:ALA-12>config>log# info
----------------------------------------------
...
        snmp-trap-group 10
            trap-target 10.10.0.91:1 "snmpv2c" notify-community "com1”
        exit
...
----------------------------------------------
A:ALA-12>config>log#
```

6.4.10 Deleting an SNMP Trap Group

The following displays the SNMP trap group configuration:

```
A:ALA-12>config>log# info
----------------------------------------------
...
       snmp-trap-group 10
           trap-target 10.10.0.91:1 "snmpv2c" notify-community "com1"
       exit
...
----------------------------------------------
A:ALA-12>config>log#
```

The following displays an example to delete a trap target and an SNMP trap group.

```
Example:config>log# snmp-trap-group 10
 config>log>snmp-trap-group# no trap-target 10.10.0.91:1
 config>log>snmp-trap-group# exit
 config>log# no snmp-trap-group 10
```

6.4.11 Modifying a Log Filter

The following output displays the current log filter configuration:

```
ALA-12>config>log# info
#------------------------------------------
echo "Log Configuration "
#------------------------------------------
...
        filter 1
            default-action drop
            description "This is a sample filter."
            entry 1
                action forward
                match
                    application eq "mirror"
                    severity eq critical
                exit
```

```
            exit
        exit
...
----------------------------------------
ALA-12>config>log#
```

The following displays an example of the log filter modifications:

```
Example:config# log
 config>log# filter 1
 config>log>filter# description "This allows <n>."
 config>log>filter# default-action forward
 config>log>filter# entry 1
 config>log>filter>entry$ action drop
 config>log>filter>entry# match
 config>log>filter>entry>match# application eq user
 config>log>filter>entry>match# number eq 2001
 config>log>filter>entry>match# no severity
 config>log>filter>entry>match# exit
```

The following displays the log filter configuration:

```
A:ALA-12>config>log>filter# info
----------------------------------------
...
        filter 1
            description "This allows <n>."
            entry 1
                action drop
                match
                    application eq "user"
                    number eq 2001
                exit
            exit
        exit
...
----------------------------------------
A:ALA-12>config>log>filter#
```

6.4.12 Modifying Event Control Parameters

The following displays the current event control configuration:

```
A:ALA-12>config>log# info
----------------------------------------------
...
event-control "bgp" 2014 generate critical
...
----------------------------------------------
A:ALA-12>config>log#
```

The following displays an example of an event control modification:

```
Example:config# log
 config>log# event-control bgp 2014 suppress
```

The following displays the log filter configuration:

```
A:ALA-12>config>log# info
----------------------------------------------
...
        event-control "bgp" 2014 suppress
...
----------------------------------------------
A:ALA-12>config>log#
```

The following displays the current event control configuration:

```
A:ALA-12>config>log# info
----------------------------------------------
...
event-control "ospf" 2014 generate critical
...
----------------------------------------------
A:ALA-12>config>log#
```

The following displays an example of an event control modification:

```
Example:config# log
 config>log# event-control ospf 2014 suppress
```

The following displays the log filter configuration:

```
A:ALA-12>config>log# info
----------------------------------------------
...
        event-control "ospf" 2014 suppress
...
----------------------------------------------
A:ALA-12>config>log#
```

6.4.13 Returning to the Default Event Control Configuration

The **no** form of the **event-control** command returns modified values back to the default values.

Use the following CLI syntax to modify event control parameters:

```
config>log
     no event-control application [event-name |event-
        number]
```

The following displays an example of the command usage to return to the default values:

```
Example:config# log
 config>log# no event-control "bgp" 2001
 config>log# no event-control "bgp" 2002
 config>log# no event-control "bgp" 2014
```

```
A:ALA-12>config>log# info detail
----------------------------------------------
#----------------------------------------
echo "Log Configuration"
#----------------------------------------
        event-control "bgp" 2001 generate minor
        event-control "bgp" 2002 generate warning
        event-control "bgp" 2003 generate warning
        event-control "bgp" 2004 generate critical
        event-control "bgp" 2005 generate warning
        event-control "bgp" 2006 generate warning
        event-control "bgp" 2007 generate warning
        event-control "bgp" 2008 generate warning
        event-control "bgp" 2009 generate warning
        event-control "bgp" 2010 generate warning
        event-control "bgp" 2011 generate warning
        event-control "bgp" 2012 generate warning
        event-control "bgp" 2013 generate warning
        event-control "bgp" 2014 generate warning
        event-control "bgp" 2015 generate critical
        event-control "bgp" 2016 generate warning
...
----------------------------------------------
A:ALA-12>config>log#
```

6.5 Accounting Logs

Before an accounting policy can be created a target log file must be created to collect the accounting records. The files are stored in system memory on compact flash (*cf1:* or *cf2:*) in a compressed (tar) XML format and can be retrieved using FTP or SCP.

A file ID can only be assigned to either one event log ID or one accounting log.

6.5.1 Accounting Records

An accounting policy must define a record name and collection interval. Only one record name can be configured per accounting policy. Also, a record name can only be used in one accounting policy.

The record name, sub-record types, and default collection period for service and network accounting policies are shown in Table 37. Table 39 (fields per policer stat-mode are given in the **stat-mode** command descriptions in the *7450 ESS, 7750 SR, 7950 XRS, and VSR Quality of Service Guide*), Table 40, and Table 41 provide field descriptions.

Table 37 **Accounting Record Name and Collection Periods**

Record Name	Sub-Record Types	Accounting Object	Platform	Default Collection Period (minutes)
service-ingress-octets	sio	SAP	All	5
service-egress-octets	seo	SAP	All	5
service-ingress-packets	sip	SAP	All	5
service-egress-packets	sep	SAP	All	5
network-ingress-octets	nio	Network port	All	15
network-egress-octets	neo	Network port	All	15
network-egress-packets	nep	Network port	All	15
network-ingress-packets	nio	Network port	All	15
compact-service-ingress-octets	ctSio	SAP	All	5
combined-service-ingress	cmSipo	SAP	All	5
combined-network-ing-egr-octets	cmNio & cmNeo	Network port	All	15
combined-service-ing-egr-octets	cmSio & cmSeo	SAP	All	5
complete-network-ingr-egr	cpNipo & cpNepo	Network port	All	15
complete-service-ingress-egress	cpSipo & cpSepo	SAP	All	5
combined-sdp-ingress-egress	cmSdpipo and cmSdpepo	SDP and SDP binding	All	5

Table 37 **Accounting Record Name and Collection Periods (Continued)**

Record Name	Sub-Record Types	Accounting Object	Platform	Default Collection Period (minutes)
complete-sdp-ingress-egress	cmSdpipo, cmSdpepo, cpSdpipo and cpSdpepo	SDP and SDP binding	All	5
complete-subscriber-ingress-egress	cpSBipo & cpSBepo	Subscriber profile	7750 SR	5
aa-protocol	aaProt	AA ISA Group	7750 SR	15
aa-application	aaApp	AA ISA Group	7750 SR	15
aa-app-group	aaAppGrp	AA ISA Group	7750 SR	15
aa-subscriber-protocol	aaSubProt	Special study AA subscriber	7750 SR	15
aa-subscriber-application	aaSubApp	Special study AA subscriber	7750 SR	15
custom-record-aa-sub	aaSubCustom	AA subscriber	All	15
combined-mpls-lsp-egress	mplsLspEgr	LSP	All	5
combined-mpls-lsp-ingress	mplsLspIn	LSP	All	5
saa	saa png trc hop	SAA or SAA test	All	5
complete-ethernet-port	enet	Ethernet port	All	15

When creating accounting policies, one service accounting policy and one network accounting policy can be defined as default. If statistics collection is enabled on a SAP or network port and no accounting policy is applied, then the respective default policy is used. If no default policy is defined, then no statistics are collected unless a specifically defined accounting policy is applied.

Each accounting record name is composed of one or more sub-records which is in turn composed of multiple fields.

Refer to the Application Assurance Statistics Fields Generated per Record table in the *7450 ESS, 7750 SR, and VSR Multiservice Integrated Service Adapter and Extended Services Appliance Guide* for fields names for Application Assurance records.

The availability of the records listed in Table 38 depends on the specific platform functionality and user configuration.

Table 38 **Accounting Record Name Details**

Record Name	Sub-Record	Field	Field Description
Service-ingress-octets (sio) [1]	sio	svc	SvcId
		sap	SapId
		qid	QueueId
		hoo	OfferedHiPrioOctets
		hod	DroppedHiPrioOctets
		loo	LowOctetsOffered
		lod	LowOctetsDropped
		uco	UncoloredOctetsOffered
		iof	InProfileOctetsForwarded
		oof	OutOfProfileOctetsForwarded
Service-egress-octets (seo) [1]	seo	svc	SvcId
		sap	SapId
		qid	QueueId
		iof	InProfileOctetsForwarded
		iod	InProfileOctetsDropped
		oof	OutOfProfileOctetsForwarded
		ood	OutOfProfileOctetsDropped

Table 38 **Accounting Record Name Details (Continued)**

Record Name	Sub-Record	Field	Field Description
Service-ingress-packets (sip) [1, 2]	sip	svc	SvcId
		sap	SapId
		qid	QueueId
		hpo	HighPktsOffered
		hpd	HighPktsDropped
		lpo	LowPktsOffered
		lpd	LowPktsDropped
		ucp	UncoloredPacketsOffered
		ipf	InProfilePktsForwarded
		opf	OutOfProfilePktsForwarded
Service-egress-packets (sep) [1, 2]	sep	svc	SvcId
		sap	SapId
		qid	QueueId
		ipf	InProfilePktsForwarded
		ipd	InProfilePktsDropped
		opf	OutOfProfilePktsForwarded
		opd	OutOfProfilePktsDropped
Network-ingress-octets (nio)	nio	port	PortId
		qid	QueueId
		iof	InProfileOctetsForwarded
		iod	InProfileOctetsDropped
		oof	OutOfProfileOctetsForwarded
		ood	OutOfProfileOctetsDropped

Table 38 **Accounting Record Name Details (Continued)**

Record Name	Sub-Record	Field	Field Description
Network-egress-octets (neo)	neo	port	PortId
		qid	QueueId
		iof	InProfileOctetsForwarded
		iod	InProfileOctetsDropped
		oof	OutOfProfileOctetsForwarded
		ood	OutOfProfileOctetsDropped
Network-ingress-packets (nip)	nip	port	PortId
		qid	QueueId
		ipf	InProfilePktsForwarded
		ipd	InProfilePktsDropped
		opf	OutOfProfilePktsForwarded
		opd	OutOfProfilePktsDropped
Network Egress Packets (nep)	nep	port	PortId
		qid	QueueId
		ipf	InProfilePktsForwarded
		ipd	InProfilePktsDropped
		opf	OutOfProfilePktsForwarded
		opd	OutOfProfilePktsDropped
Compact-service-ingress-octets (ctSio)	ctSio	svc	SvcId
		sap	SapId
		qid	QueueId
		hoo	OfferedHiPrioOctets
		hod	DroppedHiPrioOctets
		loo	LowOctetsOffered
		lod	LowOctetsDropped
		uco	UncoloredOctetsOffered

Table 38 **Accounting Record Name Details (Continued)**

Record Name	Sub-Record	Field	Field Description
Combined-service-ingress (cmSipo)	cmSipo	svc	SvcId
		sap	SapId
		qid	QueueId
		hpo	HighPktsOffered
		hpd	HighPktsDropped
		lpo	LowPktsOffered
		lpd	LowPktsDropped
		ucp	UncoloredPacketsOffered
		hoo	OfferedHiPrioOctets
		hod	DroppedHiPrioOctets
		loo	LowOctetsOffered
		lod	LowOctetsDropped
		uco	UncoloredOctetsOffered
		ipf	InProfilePktsForwarded
		opf	OutOfProfilePktsForwarded
		iof	InProfileOctetsForwarded
		oof	OutOfProfileOctetsForwarded

Table 38 **Accounting Record Name Details (Continued)**

Record Name	Sub-Record	Field	Field Description
Combined-network-ing-egr-octets (cmNio & cmNeo)	cmNio	port	PortId
		qid	QueueId
		iof	InProfileOctetsForwarded
		iod	InProfileOctetsDropped
		oof	OutOfProfileOctetsForwarded
		ood	OutOfProfileOctetsDropped
	cmNeo	port	PortId
		qid	QueueId
		iof	InProfileOctetsForwarded
		iod	InProfileOctetsDropped
		oof	OutOfProfileOctetsForwarded
		ood	OutOfProfileOctetsDropped

Table 38 **Accounting Record Name Details (Continued)**

Record Name	Sub-Record	Field	Field Description
Combined-service-ingr-egr-octets (cmSio & CmSeo)	cmSio	svc	SvcId
		sap	SapId
		qid	QueueId
		hoo	OfferedHiPrioOctets
		hod	DroppedHiPrioOctets
		loo	LowOctetsOffered
		lod	LowOctetsDropped
		uco	UncoloredOctetsOffered
		iof	InProfileOctetsForwarded
		oof	OutOfProfileOctetsForwarded
	cmSeo	svc	SvcId
		sap	SapId
		qid	QueueId
		iof	InProfileOctetsForwarded
		iod	InProfileOctetsDropped
		oof	OutOfProfileOctetsForwarded
		ood	OutOfProfileOctetsDropped

Table 38 **Accounting Record Name Details (Continued)**

Record Name	Sub-Record	Field	Field Description
Complete-network-ingr-egr (cpNipo & cpNepo)	cpNipo	port	PortId
		qid	QueueId
		ipf	InProfilePktsForwarded
		ipd	InProfilePktsDropped
		opf	OutOfProfilePktsForwarded
		opd	OutOfProfilePktsDropped
		iof	InProfileOctetsForwarded
		iod	InProfileOctetsDropped
		oof	OutOfProfileOctetsForwarded
		ood	OutOfProfileOctetsDropped
	cpNepo	port	PortId
		qid	QueueId
		ipf	InProfilePktsForwarded
		ipd	InProfilePktsDropped
		opf	OutOfProfilePktsForwarded
		opd	OutOfProfilePktsDropped
		iof	InProfileOctetsForwarded
		iod	InProfileOctetsDropped
		oof	OutOfProfileOctetsForwarded
		ood	OutOfProfileOctetsDropped

Table 38 **Accounting Record Name Details (Continued)**

Record Name	Sub-Record	Field	Field Description
Complete-service-ingress-egress (cpSipo & cpSepo)	cpSipo	svc	SvcId
		sap	SapId
		qid	QueueId
		hpo	HighPktsOffered
		hpd	HighPktsDropped
		lpo	LowPktsOffered
		lpd	LowPktsDropped
		ucp	UncoloredPacketsOffered
		hoo	OfferedHiPrioOctets
		hod	DroppedHiPrioOctets
		loo	LowOctetsOffered
		lod	LowOctetsDropped
		uco	UncoloredOctetsOffered
		apo	AllPacketsOffered
		aoo	AllOctetsOffered
		apd	AllPacketsDropped
		aod	AllOctetsDropped
		apf	AllPacketsForwarded
		aof	AllOctetsForwarded
		ipd	InProfilePktsDropped
		iod	InProfileOctetsDropped
		opd	OutOfProfilePktsDropped
		ood	OutOfProfileOctetsDropped
		hpf	HighPriorityPacketsForwarded
		hof	HighPriorityOctetsForwarded

Table 38 **Accounting Record Name Details (Continued)**

Record Name	Sub-Record	Field	Field Description
Complete-service-ingress-egress (cpSipo & cpSepo) (Continued)	cpSipo (Continued)	lpf	LowPriorityPacketsForwarded
		lof	LowPriorityOctesForwarded
		ipf	InProfilePktsForwarded
		opf	OutOfProfilePktsForwarded
		iof	InProfileOctetsForwarded
		oof	OutOfProfileOctetsForwarded
	cpSepo	svc	SvcId
		sap	SapId
		qid	QueueId
		ipf	InProfilePktsForwarded
		ipd	InProfilePktsDropped
		opf	OutOfProfilePktsForwarded
		opd	OutOfProfilePktsDropped
		iof	InProfileOctetsForwarded
		iod	InProfileOctetsDropped
		oof	OutOfProfileOctetsForwarded
		ood	OutOfProfileOctetsDropped
Complete-sdp-ingress-egress (cpSdpipo & cpSdpepo)	cpSdpipo	sdp	SdpID
		tpf	TotalPacketsForwarded
		tpd	TotalPacketsDropped
		tof	TotalOctetsForwarded
		tod	TotalOctetsDropped
	cpSdpepo	sdp	SdpID
		tpd	TotalPacketsDropped
		tod	TotalOctetsDropped

Table 38 **Accounting Record Name Details (Continued)**

Record Name	Sub-Record	Field	Field Description
Combined-sdp-ingress-egress (cmSdpipo & cmSdpepo)	cmSdpipo	svc	SvcID
		sdp	SdpID
		tpf	TotalPacketsForwarded
		tpd	TotalPacketsDropped
		tof	TotalOctetsForwarded
		tod	TotalOctetsDropped
	cmSdpepo	svc	SvcID
		sdp	SdpID
		tpf	TotalPacketsForwarded
		tof	TotalOctetsForwarded

Table 38 **Accounting Record Name Details (Continued)**

Record Name	Sub-Record	Field	Field Description
Complete-sdp-ingress-egress (cmSdpipo & cmsdpepo) (cpSdpip & cpSdpepo)	cmSdpipo	svc	SvcID
		sdp	SdpID
		tpf	TotalPacketsForwarded
		tpd	TotalPacketsDropped
		tof	TotalOctetsForwarded
		tod	TotalOctetsDropped
	cmSdpepo	svc	SvcID
		sdp	SdpID
		tpf	TotalPacketsForwarded
		tof	TotalOctetsForwarded
	cpSdpipo	sdp	SdpID
		tpf	TotalPacketsForwarded
		tpd	TotalPacketsDropped
		tof	TotalOctetsForwarded
		tod	TotalOctetsDropped
	cpSdpepo	sdp	SdpID
		tpf	TotalPacketsForwarded
		tof	TotalOctetsForwarded
Complete-subscriber-ingress-egress (cpSBipo & cpSBepo) (cpSBipooc & cpSBepooc) [3]	SubscriberInformation	subId	SubscriberId
		subProfile	SubscriberProfile
	Sla-Information [4]	svc	SvcId
		sap	SapId
		slaProfile	SlaProfile
		spiSharing	SPI sharing type and identifier

Table 38 **Accounting Record Name Details (Continued)**

Record Name	Sub-Record	Field	Field Description
Complete-subscriber-ingress-egress (cpSBipo & cpSBepo) (cpSBipooc & cpSBepooc) [3] (Continued)	cpSBipo	qid	QueueId
		hpo	HighPktsOffered [4]
		hpd	HighPktsDropped
		lpo	LowPktsOffered [4]
		lpd	LowPktsDropped
		ucp	UncolouredPacketsOffered
		hoo	OfferedHiPrioOctets [4]
		hod	DroppedHiPrioOctets
		loo	LowOctetsOffered [4]
		lod	LowOctetsDropped
		apo	AllPktsOffered [4]
		aoo	AllOctetsOffered [4]
		uco	UncolouredOctetsOffered
		ipf	InProfilePktsForwarded
		opf	OutOfProfilePktsForwarded
		iof	InProfileOctetsForwarded
		oof	OutOfProfileOctetsForwarded
		v4pf	IPv4PktsForwarded
		v6pf	IPv6PktsForwarded
		v4pd	IPv4PktsDropped
		v6pd	IPv6PktsDropped
		v4of	IPv4OctetsForwarded
		v6of	IPv6OctetsForwarded
		v4od	IPv4OctetsDropped
		v6od	IPv6OctetsDropped

Table 38 **Accounting Record Name Details (Continued)**

Record Name	Sub-Record	Field	Field Description
Complete-subscriber-ingress-egress (cpSBipo & cpSBepo) (cpSBipooc & cpSBepooc) [3] (Continued)	cpSBepo	qid	QueueId
		ipf	InProfilePktsForwarded
		ipd	InProfilePktsDropped
		opf	OutOfProfilePktsForwarded
		opd	OutOfProfilePktsDropped
		iof	InProfileOctetsForwarded
		iod	InProfileOctetsDropped
		oof	OutOfProfileOctetsForwarded
		ood	OutOfProfileOctetsDropped
		v4pf	IPv4PktsForwarded
		v6pf	IPv6PktsForwarded
		v4pd	IPv4PktsDropped
		v6pd	IPv6PktsDropped
		v4of	IPv4OctetsForwarded
		v6of	IPv6OctetsForwarded
		v4od	IPv4OctetsDropped
		v6od	IPv6OctetsDropped

Table 38 **Accounting Record Name Details (Continued)**

Record Name	Sub-Record	Field	Field Description
Complete-subscriber-ingress-egress (cpSBipo & cpSBepo) (cpSBipooc & cpSBepooc) [3] (Continued)	cpSBipooc [3]	cid	OverrideCounterId
		apo	AllPktsOffered
		hpd	HighPktsDropped
		lpd	LowPktsDropped
		aoo	AllOctetsOffered
		hod	DroppedHiPrioOctets
		lod	LowOctetsDropped
		ipf	InProfilePktsForwarded
		opf	OutOfProfilePktsForwarded
		iof	InProfileOctetsForwarded
		oof	OutOfProfileOctetsForwarded
		ucp	UncolouredPacketsOffered
		uco	UncolouredOctetsOffered
	cpSBepooc [3]	cid	OverrideCounterId
		ipf	InProfilePktsForwarded
		ipd	InProfilePktsDropped
		ofp	OutOfProfilePktsForwarded
		opd	OutOfProfilePktsDropped
		ipd	InProfileOctetsForwarded
		iod	InProfileOctetsDropped
		oof	OutOfProfileOctetsForwarded
		ood	OutOfProfileOctetsDropped

Table 38 **Accounting Record Name Details (Continued)**

Record Name	Sub-Record	Field	Field Description
saa	saa	tmd	TestMode
		own	OwnerName
		tst	TestName
		png	PingRun subrecord
		rid	RunIndex
		trr	TestRunResult
		mnr	MinRtt
		mxr	MaxRtt
		avr	AverageRtt
		rss	RttSumOfSquares
		pbr	ProbeResponses
		spb	SentProbes
		mnt	MinOutTt
		mxt	MaxOutTt
		avt	AverageOutTt
		tss	OutTtSumOfSquares
		mni	MinInTt
		mxi	MaxInTt
		avi	AverageInTt
		iss	InTtSumOfSqrs
		ojt	OutJitter
		ijt	InJitter
		rjt	RtJitter
		prt	ProbeTimeouts
		prf	ProbeFailures

Table 38 **Accounting Record Name Details (Continued)**

Record Name	Sub-Record	Field	Field Description
saa (Continued)	trc	rid	RunIndex
		trr	TestRunResult
		lgp	LastGoodProbe
	hop	hop	TraceHop
		hid	HopIndex
		mnr	MinRtt
		mxr	MaxRtt
		avr	AverageRtt
		rss	RttSumOfSquares
		pbr	ProbeResponses
		spb	SentProbes
		mnt	MinOutTt
		mxt	MaxOutTt
		avt	AverageOutTt
		tss	OutTtSumOfSquares
		mni	MinInTt
		mxi	MaxInTt
		avi	AverageInTt
		iss	InTtSumOfSqrs
		ojt	OutJitter
		ijt	InJitter
		rjt	RtJitter
		prt	ProbeTimeouts
		prf	ProbeFailures
		tat	TraceAddressType
		tav	TraceAddressValue

Table 38 **Accounting Record Name Details (Continued)**

Record Name	Sub-Record	Field	Field Description
Complete-ethernet-port (enet)	enet	port	PortId
		to	EtherStatsOctets
		tp	EtherStatsPkts
		de	EtherStatsDropEvents
		tbcp	EtherStatsBroadcastPkts
		mcp	EtherStatsMulticastPkts
		cae	EtherStatsCRCAlignErrors
		up	EtherStatsUndersizePkts
		op	EtherStatsOversizePkts
		fgm	EtherStatsFragments
		jab	EtherStatsJabbers
		col	EtherStatsCollisions
		p64o	EtherStatsPkts64Octets
		p127o	EtherStatsPkts65to127Octets
		p255o	EtherStatsPkts128to255Octets
		p511o	EtherStatsPkts256to511Octets
		p1023o	EtherStatsPkts512to1023Octets
		p1518o	EtherStatsPkts1024to1518Octets
		po1518o	EtherStatsPktsOver1518Octets
		ae	Dot3StatsAlignmentErrors
		fe	Dot3StatsFCSErrors
		scf	Dot3StatsSingleCollisionFrames
		mcf	Dot3StatsMultipleCollisionFrames
		sqe	Dot3StatsSQETestErrors
		dt	Dot3StatsDeferredTransmissions

Table 38 **Accounting Record Name Details (Continued)**

Record Name	Sub-Record	Field	Field Description
Complete-ethernet-port (enet) (Continued)	enet (Continued)	lcc	Dot3StatsLateCollisions
		exc	Dot3StatsExcessiveCollisions
		imt	Dot3StatsInternalMacTransmitErrors
		cse	Dot3StatsCarrierSenseErrors
		ftl	Dot3StatsFrameTooLongs
		imre	Dot3StatsInternalMacReceiveErrors
		se	Dot3StatsSymbolErrors
		ipf	Dot3InPauseFrames
		opf	Dot3OutPauseFrames

Notes:

1. The number of octets in an ATM sap excludes the Header Error Control (HEC) byte, thus meaning each packet/cell has only 52 bytes instead of the usual 53.
2. For a SAP in AAL5 SDU mode, packet counters refer to the number of SDU. For a SAP in N-to-1 cell mode, packet counters refer to the number of cells.
3. If override counters on the HSMDA are configured (see the *7450 ESS, 7750 SR, 7950 XRS, and VSR Quality of Service Guide*).
4. Not used to identify stats from HSMDA due to MDA architecture. If the statistics are from HSMDA: apo, aoo else lpo/hpo, loo/hoo.

Table 39, Table 40, and Table 41 provide field descriptions.

Table 39 **Policer Stats Field Descriptions**

Field	Field Description
pid	PolicerId
statmode	PolicerStatMode
aod	AllOctetsDropped
aof	AllOctetsForwarded
aoo	AllOctetsOffered
apd	AllPacketsDropped

Table 39 **Policer Stats Field Descriptions (Continued)**

Field	Field Description
apf	AllPacketsForwarded
apo	AllPacketsOffered
c1od	ConnectionOneOctetsDropped **
c1of	ConnectionOneOctetsForwarded **
c1oo	ConnectionOneOctetsOffered **
c1pd	ConnectionOnePacketsDropped **
c1pf	ConnectionOnePacketsForwarded **
c1po	ConnectionOnePacketsOffered **
c2od	ConnectionTwoOctetsDropped **
c2of	ConnectionTwoOctetsForwarded **
c2oo	ConnectionTwoOctetsOffered **
c2pd	ConnectionTwoPacketsDropped **
c2pf	ConnectionTwoPacketsForwarded **
c2po	ConnectionTwoPacketsOffered **
hod	HighPriorityOctetsDropped
hof	HighPriorityOctetsForwarded
hoo	HighPriorityOctetsOffered
hpd	HighPriorityPacketsDropped
hpf	HighPriorityPacketsForwarded
hpo	HighPriorityPacketsOffered
iod	InProfileOctetsDropped
iof	InProfileOctetsForwarded
ioo	InProfileOctetsOffered
ipd	InProfilePacketsDropped
ipf	InProfilePacketsForwarded
ipo	InProfilePacketsOffered
lod	LowPriorityOctetsDropped

Table 39 **Policer Stats Field Descriptions (Continued)**

Field	Field Description
lof	LowPriorityOctetsForwarded
loo	LowPriorityOctetsOffered
lpd	LowPriorityPacketsDropped
lpf	LowPriorityPacketsForwarded
lpo	LowPriorityPacketsOffered
opd	OutOfProfilePacketsDropped
opf	OutOfProfilePacketsForwarded
opo	OutOfProfilePacketsOffered
ood	OutOfProfileOctetsDropped
oof	OutOfProfileOctetsForwarded
ooo	OutOfProfileOctetsOffered
xpd	ExceedProfilePktsDropped
xpf	ExceedProfilePktsForwarded
xpo	ExceedProfilePktsOffered
xod	ExceedProfileOctetsDropped
xof	ExceedProfileOctetsForwarded
xoo	ExceedProfileOctetsOffered
ppd	InplusProfilePacketsDropped
ppf	InplusProfilePacketsForwarded
ppo	InplusProfilePacketsOffered
pod	InplusProfileOctetsDropped
pof	InplusProfileOctetsForwarded
poo	InplusProfileOctetsOffered
uco	UncoloredOctetsOffered
ucp	UncoloredPacketsOffered
v4po	IPv4PktsOffered *
v4oo	IPv4OctetsOffered *

Table 39 **Policer Stats Field Descriptions (Continued)**

Field	Field Description
v6po	IPv6PktsOffered *
v6oo	IPv6OctetsOffered *
v4pf	IPv4PktsForwarded *
v6pf	IPv6PktsForwarded *
v4pd	IPv4PktsDropped *
v6pd	IPv6PktsDropped *
v4of	IPv4OctetsForwarded *
v6of	IPv6OctetsForwarded *
v4od	IPv4OctetsDropped *
v6od	IPv6OctetsDropped *

* Enhanced Subscriber Management (ESM) only.

** Enhanced Subscriber Management (ESM) connection bonding only.

Table 40 **Queue Group Record Types**

Record Name	Description
qgone	PortQueueGroupOctetsNetworkEgress
qgosi	PortQueueGroupOctetsServiceIngress
qgose	PortQueueGroupOctetsServiceEgress
qgpne	PortQueueGroupPacketsNetworkEgress
qgpsi	PortQueueGroupPacketsServiceIngress
qgpse	PortQueueGroupPacketsServiceEgress
fpqgosi	ForwardingPlaneQueueGroupOctetsServiceIngress
fpqgoni	ForwardingPlaneQueueGroupOctetsNetworkIngress
fpqgpsi	ForwardingPlaneQueueGroupPacketsServiceIngress
fpqgpni	ForwardingPlaneQueueGroupPacketsNetworkIngress

Table 41 **Queue Group Record Type Fields**

Field	Field Description
data port	Port (used for port based Queue Groups)
member-port	LAGMemberPort (used for port based Queue Groups)
data slot	Slot (used for Forwarding Plane based Queue Groups)
forwarding-plane	ForwardingPlane (used for Forwarding Plane based Queue Groups)
queue-group	QueueGroupName
instance	QueueGroupInstance
qid	QueueId
pid	PolicerId
statmode	PolicerStatMode
aod...ucp	same as above

6.5.2 Accounting Files

When a policy has been created and applied to a service or network port, the accounting file is stored on the compact flash in a compressed XML file format. The router creates two directories on the compact flash to store the files. The following output displays a directory named **act-collect** that holds accounting files that are open and actively collecting statistics. The directory named **act** stores the files that have been closed and are awaiting retrieval.

```
ALA-1>file cf1:\# dir act*
12/19/2006 06:08a      <DIR>          act-collect
12/19/2006 06:08a      <DIR>          act

ALA-1>file cf1:\act-collect\ # dir
Directory of cf1:\act-collect#

12/23/2006 01:46a      <DIR>          .
12/23/2006 12:47a      <DIR>          ..
12/23/2006 01:46a                   112 act1111-20031223-014658.xml.gz
12/23/2006 01:38a                   197 act1212-20031223-013800.xml.gz
```

Accounting files always have the prefix **act** followed by the accounting policy ID, log ID and timestamp. The accounting log file naming and log file destination properties like rollover and retention are discussed in more detail in Log Files.

6.5.3 Design Considerations for Accounting Policies

The router has ample resources to support large scale accounting policy deployments. When preparing for an accounting policy deployment, verify that data collection, file rollover, and file retention intervals are properly tuned for the amount of statistics to be collected.

If the accounting policy collection interval is too brief there may be insufficient time to store the data from all the services within the specified interval. If that is the case, some records may be lost or incomplete. Interval time, record types, and number of services using an accounting policy are all factors that should be considered when implementing accounting policies.

The rollover and retention intervals on the log files and the frequency of file retrieval must also be considered when designing accounting policy deployments. The amount of data stored depends on the type of record collected, the number of services that are collecting statistics, and the collection interval that is used. For example, with a 1Gb CF and using the default collection interval, the system is expected to hold 48 hours' worth of billing information.

6.5.4 Reporting and Time-Based Accounting

SR OS on the 7750 SR platform has support for volume accounting and time-based accounting concepts, and provides an extra level of intelligence at the network element level in order to provide service models such as "prepaid access" in a scalable manner. This means that the network element gathers and stores per-subscriber accounting information and compares it with "pre-defined" quotas. Once a quota is exceeded, the pre-defined action (such as re-direction to a web portal or disconnect) is applied.

6.5.5 Overhead Reduction in Accounting: Custom Record

Custom records can be used to decrease accounting messaging overhead as follows:

- User Configurable Records
- Changed Statistics Only
- Configurable Accounting Records

- Significant Change Only Reporting

6.5.5.1 User Configurable Records

Users can define a collection of fields that make up a record. These records can be assigned to an accounting policy. These are user-defined records rather than being limited to pre-defined record types. The operator can select queues and policers and the counters within these queues and policers that need to be collected. Refer to the predefined records containing a given field for XML field name of a custom record field.

6.5.5.2 Changed Statistics Only

A record is only generated if a significant change has occurred to the fields being written in a given the record. This capability applies to both ingress and egress records regardless on the method of delivery (such as RADIUS and XML). The capability also applies to Application Assurance records; however without an ability to specify different significant change values and per-field scope (for example, all fields of a custom record are collected if any activity was reported against any of the statistics that are part of the custom record).

6.5.5.3 Configurable Accounting Records

6.5.5.3.1 XML Accounting Files for Service and ESM-Based Accounting

The **custom-record** command in the **config**>**log**>**accounting-policy** context provides the flexibility to reduce the volume of data generated by allowing network operators to specify the record needed for collection. This can eliminate queues and policers, or selected counters within these queues and policers, that are not relevant for billing.

ESM-based accounting applies to the 7750 SR only.

Record headers including information such as service-ID, SAP-ID, and so on, will always be generated.

6.5.5.3.2 XML Accounting Files for Policer Counters

Policer counters can be collected using custom records with the record within the accounting policy configured to be either **custom-record-service** or **custom-record-subscriber**. The policer identifier for which counters are collected must be configured under **custom-record**, specifying the required ingress (**i-counters**) and egress (**e-counters**) counters to be collected. A similar configuration is available for a reference policer (**ref-policer**) to define a reference counter used together with the **significant-change** command.

The counters collected are dependent on the **stat-mode** of the related policer, as this determines which statistics are collected by the system for the policer.

The ingress policer counters collected for each combination of XML accounting record name and policer **stat-mode** are given in Table 42.

The egress policer counters collected for each combination of XML accounting record name and policer **stat-mode** are given in Table 43.

Table 42 **Custom Record Policer Ingress Counter Mapping**

Policer i-counters CLI Name	Policer stat-mode	Custom Record Counter	Custom Record Field
in-profile-octets-discarded-count	minimal	—	—
	offered-limited-capped-cir offered-limited-profile-cir offered-priority-cir offered-profile-capped-cir offered-profile-cir offered-profile-no-cir offered-total-cir	In-Profile Octets Dropped	iod
	offered-priority-no-cir	High-Priority Octets Dropped	hod
	v4-v6	V4 Octets Dropped	v4od

Table 42 **Custom Record Policer Ingress Counter Mapping (Continued)**

Policer i-counters CLI Name	Policer stat-mode	Custom Record Counter	Custom Record Field
in-profile-octets-forwarded-count	minimal	—	—
	offered-limited-capped-cir offered-limited-profile-cir offered-priority-cir offered-profile-capped-cir offered-profile-cir offered-profile-no-cir offered-total-cir	In-Profile Octets Forwarded	iof
	offered-priority-no-cir	High-Priority Octets Forwarded	hof
	v4-v6	V4 Octets Forwarded	v4of
in-profile-octets-offered-count	minimal offered-limited-profile-cir offered-total-cir	—	—
	offered-limited-capped-cir offered-profile-capped-cir offered-profile-cir offered-profile-no-cir	In-Profile Octets Offered	ioo
	offered-priority-cir offered-priority-no-cir	High-Priority Octets Offered	hoo
	v4-v6	V4 Octets Offered	v4oo
in-profile-packets-discarded-count	minimal	—	—
	offered-limited-capped-cir offered-limited-profile-cir offered-priority-cir offered-profile-capped-cir offered-profile-cir offered-profile-no-cir offered-total-cir	In-Profile Packets Dropped	ipd
	offered-priority-no-cir	High-Priority Packets Dropped	hpd
	v4-v6	V4 Packets Dropped	v4pd

Table 42 **Custom Record Policer Ingress Counter Mapping (Continued)**

Policer i-counters CLI Name	Policer stat-mode	Custom Record Counter	Custom Record Field
in-profile-packets-forwarded-count	minimal	—	—
	offered-limited-capped-cir offered-limited-profile-cir offered-priority-cir offered-profile-capped-cir offered-profile-cir offered-profile-no-cir offered-total-cir	In-Profile Packets Forwarded	ipf
	offered-priority-no-cir	High-Priority Packets Forwarded	hpf
	v4-v6	V4 Packets Forwarded	v4pf
in-profile-packets-offered-count	minimal offered-limited-profile-cir offered-total-cir	—	—
	offered-limited-capped-cir offered-profile-capped-cir offered-profile-cir offered-profile-no-cir	In-Profile Packets Offered	ipo
	offered-priority-cir offered-priority-no-cir	High-Priority Packets Offered	hpo
	v4-v6	V4 Packets Offered	v4po
out-profile-octets-discarded-count	minimal	All Octets Dropped	aod
	offered-limited-capped-cir offered-limited-profile-cir offered-priority-cir offered-profile-capped-cir offered-profile-cir offered-profile-no-cir offered-total-cir	Out-of-Profile Octets Dropped	ood
	offered-priority-no-cir	Low-Priority Octets Dropped	lod
	v4-v6	V6 Octets Dropped	v6od

Table 42 **Custom Record Policer Ingress Counter Mapping (Continued)**

Policer i-counters CLI Name	Policer stat-mode	Custom Record Counter	Custom Record Field
out-profile-octets-forwarded-count	minimal	All Octets Forwarded	aof
	offered-limited-capped-cir offered-limited-profile-cir offered-priority-cir offered-profile-capped-cir offered-profile-cir offered-profile-no-cir offered-total-cir	Out-of-Profile Octets Forwarded	oof
	offered-priority-no-cir	Low-Priority Octets Forwarded	lof
	v4-v6	V6 Octets Forwarded	v6of
out-profile-octets-offered-count	minimal offered-total-cir	All Octets Offered	aoo
	offered-limited-capped-cir	—	—
	offered-limited-profile-cir offered-profile-capped-cir offered-profile-cir offered-profile-no-cir	Out-of-Profile Octets Offered	ooo
	offered-priority-cir offered-priority-no-cir	Low-Priority Octets Offered	loo
	v4-v6	V6 Octets Offered	v6oo
out-profile-packets-discarded-count	minimal	All Packets Dropped	apd
	offered-limited-capped-cir offered-limited-profile-cir offered-priority-cir offered-profile-capped-cir offered-profile-cir offered-profile-no-cir offered-total-cir	Out-of-Profile Packets Dropped	opd
	offered-priority-no-cir	Low-Priority Packets Dropped	lpd
	v4-v6	V6 Packets Dropped	v6pd

Table 42 **Custom Record Policer Ingress Counter Mapping (Continued)**

Policer i-counters CLI Name	Policer stat-mode	Custom Record Counter	Custom Record Field
out-profile-packets-forwarded-count	minimal	All Packets Forwarded	apf
	offered-limited-capped-cir offered-limited-profile-cir offered-priority-cir offered-profile-capped-cir offered-profile-cir offered-profile-no-cir offered-total-cir	Out-of-Profile Packets Forwarded	opf
	offered-priority-no-cir	Low-Priority Packets Forwarded	lpf
	v4-v6	V6 Packets Forwarded	v6pf
out-profile-packets-offered-count	minimal offered-total-cir	All Packets Offered	apo
	offered-limited-capped-cir	n/a	n/a
	offered-limited-profile-cir offered-profile-capped-cir offered-profile-cir offered-profile-no-cir	Out-of-Profile Packets Offered	opo
	offered-priority-cir offered-priority-no-cir	Low-Priority Packets Offered	lpo
	v4-v6	V6 Packets Offered	v6po
uncoloured-octets-offered-count	minimal offered-priority-cir offered-priority-no-cir offered-profile-no-cir offered-total-cir v4-v6	—	—
	offered-limited-capped-cir offered-limited-profile-cir offered-profile-capped-cir offered-profile-cir	Uncoloured Octets Offered	uco

Table 42 **Custom Record Policer Ingress Counter Mapping (Continued)**

Policer i-counters CLI Name	Policer stat-mode	Custom Record Counter	Custom Record Field
uncoloured-packets-offered-count	minimal offered-priority-cir offered-priority-no-cir offered-profile-no-cir offered-total-cir v4-v6	—	—
	offered-limited-capped-cir offered-limited-profile-cir offered-profile-capped-cir offered-profile-cir	Uncoloured Packets Offered	ucp

Table 43 **Custom Record Policer Egress Counter Mapping**

Policer e-counters CLI Name	Policer stat-mode	Custom Record Counter	Custom Record Field
exceed-profile-octets-discarded-count	bonding minimal offered-limited-capped-cir offered-profile-capped-cir offered-profile-cir offered-profile-no-cir offered-total-cir v4-v6	n/a	n/a
	offered-four-profile-no-cir offered-total-cir-exceed offered-total-cir-four-profile	Exceed-Profile Octets Dropped	xod

Table 43 **Custom Record Policer Egress Counter Mapping (Continued)**

Policer e-counters CLI Name	Policer stat-mode	Custom Record Counter	Custom Record Field
exceed-profile-octets-forwarded-count	bonding minimal offered-limited-capped-cir offered-profile-capped-cir offered-profile-cir offered-profile-no-cir offered-total-cir v4-v6	—	—
	offered-four-profile-no-cir offered-total-cir-exceed offered-total-cir-four-profile	Exceed-Profile Octets Forwarded	xof
exceed-profile-octets-offered-count	bonding minimal offered-limited-capped-cir offered-profile-capped-cir offered-profile-cir offered-profile-no-cir offered-total-cir offered-total-cir-exceed offered-total-cir-four-profile v4-v6	—	—
	offered-four-profile-no-cir	Exceed-Profile Octets Offered	xoo

Table 43 **Custom Record Policer Egress Counter Mapping (Continued)**

Policer e-counters CLI Name	Policer stat-mode	Custom Record Counter	Custom Record Field
exceed-profile-packets-discarded-count	bonding minimal offered-limited-capped-cir offered-profile-capped-cir offered-profile-cir offered-profile-no-cir offered-total-cir v4-v6	—	—
	offered-four-profile-no-cir offered-total-cir-exceed offered-total-cir-four-profile	Exceed-Profile Packets Dropped	xpd
exceed-profile-packets-forwarded-count	bonding minimal offered-limited-capped-cir offered-profile-capped-cir offered-profile-cir offered-profile-no-cir offered-total-cir v4-v6	—	—
	offered-four-profile-no-cir offered-total-cir-exceed offered-total-cir-four-profile	Exceed-Profile Packets Forwarded	xpf

Table 43 **Custom Record Policer Egress Counter Mapping (Continued)**

Policer e-counters CLI Name	Policer stat-mode	Custom Record Counter	Custom Record Field
exceed-profile-packets-offered-count	bonding minimal offered-limited-capped-cir offered-profile-capped-cir offered-profile-cir offered-profile-no-cir offered-total-cir offered-total-cir-exceed offered-total-cir-four-profile v4-v6	—	—
	offered-four-profile-no-cir	Exceed-Profile Packets Offered	xpo
in-plus-profile-octets-discarded-count	bonding minimal offered-limited-capped-cir offered-profile-capped-cir offered-profile-cir offered-profile-no-cir offered-total-cir offered-total-cir-exceed v4-v6	—	—
	offered-four-profile-no-cir offered-total-cir-four-profile	In-Plus-Profile Octets Dropped	pod

Table 43 **Custom Record Policer Egress Counter Mapping (Continued)**

Policer e-counters CLI Name	Policer stat-mode	Custom Record Counter	Custom Record Field
in-plus-profile-octets-forwarded-count	bonding minimal offered-limited-capped-cir offered-profile-capped-cir offered-profile-cir offered-profile-no-cir offered-total-cir offered-total-cir-exceed v4-v6	—	—
	offered-four-profile-no-cir offered-total-cir-four-profile	In-Plus-Profile Octets Forwarded	pof
in-plus-profile-octets-offered-count	bonding minimal offered-limited-capped-cir offered-profile-capped-cir offered-profile-cir offered-profile-no-cir offered-total-cir offered-total-cir-exceed offered-total-cir-four-profile v4-v6	—	—
	offered-four-profile-no-cir	In-Plus-Profile Octets Offered	poo

Table 43 **Custom Record Policer Egress Counter Mapping (Continued)**

Policer e-counters CLI Name	Policer stat-mode	Custom Record Counter	Custom Record Field
in-plus-profile-packets-discarded-count	bonding minimal offered-limited-capped-cir offered-profile-capped-cir offered-profile-cir offered-profile-no-cir offered-total-cir offered-total-cir-exceed v4-v6	—	—
	offered-four-profile-no-cir offered-total-cir-four-profile	In-Plus-Profile Packets Dropped	ppd
in-plus-profile-packets-forwarded-count	bonding minimal offered-limited-capped-cir offered-profile-capped-cir offered-profile-cir offered-profile-no-cir offered-total-cir offered-total-cir-exceed v4-v6	—	—
	offered-four-profile-no-cir offered-total-cir-four-profile	In-Plus-Profile Packets Forwarded	ppf

Table 43 **Custom Record Policer Egress Counter Mapping (Continued)**

Policer e-counters CLI Name	Policer stat-mode	Custom Record Counter	Custom Record Field
in-plus-profile-packets-offered-count	bonding minimal offered-limited-capped-cir offered-profile-capped-cir offered-profile-cir offered-profile-no-cir offered-total-cir offered-total-cir-exceed offered-total-cir-four-profile v4-v6	—	—
	offered-four-profile-no-cir	In-Plus-Profile Packets Offered	ppo
in-profile-octets-discarded-count	bonding	Connection 1 Octets Dropped	c1od
	minimal	—	—
	offered-four-profile-no-cir offered-limited-capped-cir offered-profile-capped-cir offered-profile-cir offered-profile-no-cir offered-total-cir offered-total-cir-exceed offered-total-cir-four-profile	In-Profile Octets Dropped	iod
	v4-v6	V4 Octets Dropped	v4od

Table 43 **Custom Record Policer Egress Counter Mapping (Continued)**

Policer e-counters CLI Name	Policer stat-mode	Custom Record Counter	Custom Record Field
in-profile-octets-forwarded-count	bonding	Connection 1 Octets Forwarded	c1of
	minimal	—	—
	offered-limited-capped-cir offered-profile-capped-cir offered-profile-cir offered-profile-no-cir offered-total-cir offered-total-cir-exceed offered-four-profile-no-cir offered-total-cir-four-profile	In-Profile Octets Forwarded	iof
	v4-v6	V4 Octets Forwarded	v4of
in-profile-octets-offered-count	bonding	Connection 1 Octets Offered	c1oo
	minimal offered-total-cir offered-total-cir-exceed offered-total-cir-four-profile	—	—
	offered-four-profile-no-cir offered-limited-capped-cir offered-profile-capped-cir offered-profile-cir offered-profile-no-cir	In-Profile Octets Offered	ioo
	v4-v6	V4 Octets Offered	v4oo

Table 43 **Custom Record Policer Egress Counter Mapping (Continued)**

Policer e-counters CLI Name	Policer stat-mode	Custom Record Counter	Custom Record Field
in-profile-packets-discarded-count	bonding	Connection 1 Packets Dropped	c1pd
	minimal	—	—
	offered-four-profile-no-cir offered-limited-capped-cir offered-profile-capped-cir offered-profile-cir offered-profile-no-cir offered-total-cir offered-total-cir-exceed offered-total-cir-four-profile	In-Profile Packets Dropped	ipd
	v4-v6	V4 Packets Dropped	v4pd
in-profile-packets-forwarded-count	bonding	Connection 1 Packets Forwarded	c1pf
	minimal	—	—
	offered-limited-capped-cir offered-profile-capped-cir offered-profile-cir offered-profile-no-cir offered-total-cir offered-total-cir-exceed offered-four-profile-no-cir offered-total-cir-four-profile	In-Profile Packets Forwarded	ipf
	v4-v6	V4 Packets Forwarded	v4pf

Table 43 **Custom Record Policer Egress Counter Mapping (Continued)**

Policer e-counters CLI Name	Policer stat-mode	Custom Record Counter	Custom Record Field
in-profile-packets-offered-count	bonding	Connection 1 Packets Offered	c1po
	minimal offered-total-cir offered-total-cir-exceed offered-total-cir-four-profile	—	—
	offered-four-profile-no-cir offered-limited-capped-cir offered-profile-capped-cir offered-profile-cir offered-profile-no-cir	In-Profile Packets Offered	ipo
	v4-v6	V4 Packets Offered	v4po
out-profile-octets-discarded-count	bonding	Connection 2 Octets Dropped	c2od
	minimal	All Octets Dropped	aod
	offered-limited-capped-cir offered-profile-capped-cir offered-profile-cir offered-profile-no-cir offered-total-cir offered-total-cir-exceed offered-four-profile-no-cir offered-total-cir-four-profile	Out-of-Profile Octets Dropped	ood
	v4-v6	V6 Octets Dropped	v6od

Table 43 **Custom Record Policer Egress Counter Mapping (Continued)**

Policer e-counters CLI Name	Policer stat-mode	Custom Record Counter	Custom Record Field
out-profile-octets-forwarded-count	bonding	Connection 2 Octets Forwarded	c2of
	minimal	All Octets Forwarded	aof
	offered-limited-capped-cir offered-profile-capped-cir offered-profile-cir offered-profile-no-cir offered-total-cir offered-total-cir-exceed offered-four-profile-no-cir offered-total-cir-four-profile	Out-of-Profile Octets Forwarded	oof
	v4-v6	V6 Octets Forwarded	v6of
out-profile-octets-offered-count	bonding	Connection 2 Octets Offered	c2oo
	minimal offered-total-cir offered-total-cir-exceed offered-total-cir-four-profile	All Octets Offered	aoo
	offered-limited-capped-cir offered-profile-capped-cir offered-profile-cir offered-profile-no-cir offered-four-profile-no-cir	Out-of-Profile Octets Offered	ooo
	v4-v6	V6 Octets Offered	v6oo

Table 43 **Custom Record Policer Egress Counter Mapping (Continued)**

Policer e-counters CLI Name	Policer stat-mode	Custom Record Counter	Custom Record Field
out-profile-packets-discarded-count	bonding	Connection 2 Packets Dropped	c2pd
	minimal	All Packets Dropped	apd
	offered-limited-capped-cir offered-profile-capped-cir offered-profile-cir offered-profile-no-cir offered-total-cir offered-total-cir-exceed offered-four-profile-no-cir offered-total-cir-four-profile	Out-of-Profile Packets Dropped	opd
	v4-v6	V6 Packets Dropped	v6pd
out-profile-packets-forwarded-count	bonding	Connection 2 Packets Forwarded	c2pf
	minimal	All Packets Forwarded	apf
	offered-limited-capped-cir offered-profile-capped-cir offered-profile-cir offered-profile-no-cir offered-total-cir offered-total-cir-exceed offered-four-profile-no-cir offered-total-cir-four-profile	Out-of-Profile Packets Forwarded	opf
	v4-v6	V6 Packets Forwarded	v6pf

Table 43 **Custom Record Policer Egress Counter Mapping (Continued)**

Policer e-counters CLI Name	Policer stat-mode	Custom Record Counter	Custom Record Field
out-profile-packets-offered-count	bonding	Connection 2 Packets Offered	c2po
	minimal offered-total-cir offered-total-cir-exceed offered-total-cir-four-profile	All Packets Offered	apo
	offered-limited-capped-cir offered-profile-capped-cir offered-profile-cir offered-profile-no-cir offered-four-profile-no-cir	Out-of-Profile Packets Offered	opo
	v4-v6	V6 Packets Offered	v6po
uncoloured-octets-offered-count	bonding minimal offered-four-profile-no-cir offered-limited-capped-cir offered-profile-no-cir offered-total-cir offered-total-cir-exceed offered-total-cir-four-profile v4-v6	—	—
	offered-profile-capped-cir offered-profile-cir	Uncoloured Octets Offered	uco

Table 43 **Custom Record Policer Egress Counter Mapping (Continued)**

Policer e-counters CLI Name	Policer stat-mode	Custom Record Counter	Custom Record Field
uncoloured-packets-offered-count	bonding minimal offered-four-profile-no-cir offered-limited-capped-cir offered-profile-no-cir offered-total-cir offered-total-cir-exceed offered-total-cir-four-profile v4-v6	—	—
	offered-profile-capped-cir offered-profile-cir	Uncoloured Packets Offered	ucp

6.5.5.3.3 RADIUS Accounting in Networks Using ESM

The **custom-record** command in the **config>subscr-mgmt>radius-accounting-policy** context provide the flexibility to include individual counters in RADIUS accounting messages. See the CLI tree for commands and syntax. This functionality applies to the 7750 SR only.

6.5.5.4 Significant Change Only Reporting

Another way to decrease accounting messaging related to overhead is to include only “active” objects in a periodical reporting. An “active object” in this context is an object which has seen a “significant” change in corresponding counters. A significant change is defined in terms of a cumulative value (the sum of all reference counters).

This concept is applicable to all methods used for gathering accounting information, such as an XML file and RADIUS, as well as to all applications using accounting, such as service-acct, ESM-acct, and Application Assurance.

Accounting records are reported at the periodical intervals. This periodic reporting is extended with an internal filter which omits periodical updates for objects whose counter change experienced lower changes than a defined (configurable) threshold.

Specific to RADIUS accounting the **significant-change** command does not affect ACCT-STOP messages. ACCT-STOP messages will be always sent, regardless the amount of change of the corresponding host.

For Application Assurance records, a significant change of 1 in any field of a customized record (send a record if any field changed) is supported. When configured, if any statistic field records activity, an accounting record containing all fields will be collected.

6.5.6 Immediate Completion of Records

6.5.6.1 Record Completion for XML Accounting

For ESM RADIUS accounting, an accounting stop message is sent when:

- A subscriber/subscriber-host is deleted.
- An SLA profile instance (non-HSMDA) or subscriber instance (HSMDA) is changed.

A similar concept is also used for XML accounting. In case the accounted object is deleted or changed, the latest information will be written in the XML file with a “final” tag indication in the record header. This functionality applies to the 7750 SR only.

6.5.7 AA Accounting per Forwarding Class

This feature allows the operator to report on protocol/application/app-group volume usage per forwarding class by adding a bitmap information representing the observed FC in the XML accounting files. In case the accounted object is deleted or changed, the latest information will be written in the XML file with a “final” tag indication in the record header.

6.6 Configuration Notes

This section describes logging configuration restrictions.

- A file or filter cannot be deleted if it has been applied to a log.

- File IDs, syslog IDs, or SNMP trap groups must be configured before they can be applied to a log ID.
- A file ID can only be assigned to *either* one log ID *or* one accounting policy.
- Accounting policies must be configured in the **config**>**log** context before they can be applied to a service SAP or service interface, or applied to a network port.
- The **snmp-trap-id** must be the same as the **log-id**.

6.7 Configuring Logging with CLI

This section provides information to configure logging with the command line interface.

6.7.1 Log Configuration Overview

Configure logging parameters to save information in a log file or direct the messages to other devices. Logging does the following:

- Provides you with logging information for monitoring and troubleshooting.
- Allows the selection of the types of logging information to be recorded.
- Allows the assignment of a severity to the log messages.
- Allows the selection of source and target of logging information.

6.7.2 Log Types

Logs can be configured in the following contexts:

- Log file — Log files can contain log event message streams or accounting/billing information. Log file IDs are used to direct events, alarms or traps and debug information to their respective targets.
- SNMP trap groups — SNMP trap groups contain an IP address and community names which identify targets to send traps following specified events.
- Syslog — Information can be sent to a syslog host that is capable of receiving selected syslog messages from a network element.
- Event control — Configures a particular event or all events associated with an application to be generated or suppressed.
- Event filters — An event filter defines whether to forward or drop an event or trap based on match criteria.
- Accounting policies — An accounting policy defines the accounting records that will be created. Accounting policies can be applied to one or more service access points (SAPs).
- Event logs — An event log defines the types of events to be delivered to its associated destination.
- Event throttling rate — Defines the rate of throttling events.

6.7.3 Basic Log Configuration

The most basic log configuration must have the following:

- Log ID or accounting policy ID
- A log source
- A log destination

The following displays a log configuration example for the 7750 SR.

```
A:ALA-12>config>log# info
#------------------------------------------
echo "Log Configuration "
#------------------------------------------
        event-control "bgp" 2001 generate critical
        file-id 1
            description "This is a test file-id."
            location cf1:
        exit
        file-id 2
            description "This is a test log."
            location cf1:
        exit
        snmp-trap-group 7
            trap-target 11.22.33.44 "snmpv2c" notify-community "public"
        exit
        log-id 2
            from main
            to file 2
        exit
----------------------------------------------
A:ALA-12>config>log#
```

6.7.4 Common Configuration Tasks

The following sections describe basic system tasks that must be performed.

6.7.4.1 Configuring an Event Log

A event log file contains information used to direct events, alarms, traps, and debug information to their respective destinations. One or more event sources can be specified. File IDs, SNMP trap groups, or syslog IDs must be configured before they can be applied to an event log ID.

Use the following CLI syntax to configure a log file:

```
config>log
     log-id log-id
   description description-string
   filter filter-id
   from {[main] [security] [change] [debug-trace]}
   to console
   to file file-id
   to memory [size]
   to session
   to snmp [size]
   to syslog syslog-id}
   time-format {local | utc}
   no shutdown
```

The following displays a log file configuration example:

```
ALA-12>config>log>log-id# info
----------------------------------------------
...
log-id 2
            description "This is a test log file."
            filter 1
            from main security
            to file 1
exit
...
----------------------------------------------
ALA-12>config>log>log-id#
```

6.7.4.2 Configuring a File ID

To create a log file a file ID is defined, specifies the target CF drive, and the rollover and retention interval period for the file. The rollover interval is defined in minutes and determines how long a file will be used before it is closed and a new log file is created. The retention interval determines how long the file will be stored on the CF before it is deleted.

When creating new log files in a compact flash disk card, the minimum amount of free space is the MINIMUM of 10% of Compact Flash disk capacity OR 5 Mb (5,242,880 = 5 * 1024 * 1024).

The following displays a log file configuration example:

```
A:ALA-12>config>log# info
----------------------------------------
        file-id 1
            description "This is a log file."
            location cf1:
            rollover 600 retention 24
```

```
        exit
-----------------------------------------------
A:ALA-12>config>log#
```

6.7.4.3 Configuring an Accounting Policy

Before an accounting policy can be created a target log file must be created to collect the accounting records. The files are stored in system memory of compact flash (cf1: or cf2:) in a compressed (tar) XML format and can be retrieved using FTP or SCP. See Configuring an Event Log and Configuring a File ID.

Accounting policies must be configured in the **config**>**log** context before they can be applied to a service SAP or service interface, or applied to a network port.

The default accounting policy statement cannot be applied to LDP nor RSVP statistics collection records.

An accounting policy must define a record type and collection interval. Only one record type can be configured per accounting policy.

When creating accounting policies, one service accounting policy and one network accounting policy can be defined as default. If statistics collection is enabled on a SAP or network port and no accounting policy is applied, then the respective default policy is used. If no default policy is defined, then no statistics are collected unless a specifically defined accounting policy is applied.

By default, the subscriber host volume accounting data are based on the 14-byte Ethernet DLC header, 4-byte or 8-byte VLAN Tag (optional), 20-byte IP header, IP payload, and the 4-byte CRC (everything except the preamble and inter-frame gap). See Figure 20. This default can be altered by the **packet-byte-offset** configuration option.

Figure 20 **Subscriber Host Volume Accounting Data**

Destination MAC	Source MAC	802.1Q tag (optional)	802.1Q tag (optional)	Ethertype (Ethernet II) or length (IEEE 802.3)	Payload	CRC/FCS
6 octets	6 octets	(4 octets)	(4 octets)	2 octets	46-1500 octets	4 octets

0971

The following displays an accounting policy configuration example:

```
A:ALA-12>config>log# info
----------------------------------------------
accounting-policy 4
description "This is the default accounting policy."
record complete-service-ingress-egress
default
```

```
to file 1
exit
accounting-policy 5
description "This is a test accounting policy."
record service-ingress-packets
to file 3
exit
```

6.7.4.4 Configuring Event Control

The following displays an example of an event control configuration:

```
A:ALA-12>config>log# info
#------------------------------------------
echo "Log Configuration"
#------------------------------------------
        throttle-rate 500 interval 10
        event-control "oam" 2001 generate throttle
        event-control "ospf" 2001 suppress
        event-control "ospf" 2003 generate cleared
        event-control "ospf" 2014 generate critical
..
-----------------------------------------------
A:ALA-12>config>log>filter#
```

6.7.4.5 Configuring a Log Filter

The following displays a log filter configuration example:

```
A:ALA-12>config>log# info
#------------------------------------------
echo "Log Configuration "
#------------------------------------------
        file-id 1
            description "This is our log file."
            location cf1:
            rollover 600 retention 24
        exit
        filter 1
            default-action drop
            description "This is a sample filter."
            entry 1
                action forward
                match
                    application eq "mirror"
                    severity eq critical
                exit
            exit
        exit
...
log-id 2
```

```
            shutdown
            description "This is a test log file."
            filter 1
            from main security
            to file 1
        exit
...
-----------------------------------------
A:ALA-12>config>log#
```

6.7.4.6 Configuring an SNMP Trap Group

The associated *log-id* does not have to configured before a **snmp-trap-group** can be created, however, the **snmp-trap-group** must exist before the *log-id* can be configured to use it.

The following displays a basic SNMP trap group configuration example:

```
A:ALA-12>config>log# info
----------------------------------------------
...
snmp-trap-group 2
trap-target 10.10.10.104:5 "snmpv3" notify-community "coummunitystring"
        exit
...
log-id 2
            description "This is a test log file."
            filter 1
            from main security
            to file 1
exit
...
----------------------------------------------
A:ALA-12>config>log#
```

The following displays a SNMP trap group, log, and interface configuration examples:

```
A:SetupCLI>config>log# snmp-trap-group 44
A:SetupCLI>config>log>snmp-trap-group# info
----------------------------------------------
      trap-target "xyz-test" address xx.xx.x.x snmpv2c notify-community "xyztesting"
      trap-target "test2" address xx.xx.xx.x snmpv2c notify-community "xyztesting"
----------------------------------------------
*A:SetupCLI>config>log>log-id# info
----------------------------------------------
      from main
      to snmp
----------------------------------------------
*A:SetupCLI>config>router# interface xyz-test
*A:SetupCLI>config>router>if# info
----------------------------------------------
      address xx.xx.xx.x/24
```

```
      port 1/1/1
-------------------------------------------
*A:SetupCLI>config>router>if#
```

6.7.4.6.1 Setting the Replay Parameter

For this example the replay parameter was set by a SNMP SET request for the trap-target address 10.10.10.3 which is bound to port-id 1/1/1.

```
A:SetupCLI>config>log>snmp-trap-group 44
A:SetupCLI>config>log>snmp-trap-group# info
-------------------------------------------
trap-target "xyz-test" address 10.10.10.3 snmpv2c notify-
community "xyztesting" replay
trap-target "test2" address 10.20.20.5 snmpv2c notify-community "xyztesting"
-------------------------------------------
A:SetupCLI>config>log>snmp-trap-group#
```

In the following output, the **Replay** field changed from disabled to enabled.

```
A:SetupCLI>config>log>snmp-trap-group# show log snmp-trap-group 44
===============================================================================
SNMP Trap Group 44
===============================================================================
Description : none
-------------------------------------------------------------------------------
Name        : xyz-test
Address     : 10.10.10.3
Port        : 162
Version     : v2c
Community   : xyztesting
Sec. Level  : none
Replay      : enabled
Replay from : n/a
Last replay : never
-------------------------------------------------------------------------------
Name        : test2
Address     : 10.20.20.5
Port        : 162
Version     : v2c
Community   : xyztesting
Sec. Level  : none
Replay      : disabled
Replay from : n/a
Last replay : never
===============================================================================
A:SetupCLI>config>log>snmp-trap-group#
```

Since no events are waiting to be replayed, the log displays as before.

```
A:SetupCLI>config>log>snmp-trap-group# show log log-id 44
===============================================================================
Event Log 44
===============================================================================
```

```
SNMP Log contents  [size=100   next event=3819  (wrapped)]

3818 2008/04/22 23:35:39.89 UTC WARNING: SYSTEM #2009 Base IP
"Status of vRtrIfTable: router Base (index 1) interface xyz-test (index 35) changed
administrative state: inService, operational state: inService"

3817 2008/04/22 23:35:39.89 UTC WARNING: SNMP #2005 Base xyz-test
"Interface xyz-test is operational"

3816 2008/04/22 23:35:39.89 UTC WARNING: SNMP #2005 Base 1/1/1
"Interface 1/1/1 is operational"

3815 2008/04/22 23:35:39.71 UTC WARNING: SYSTEM #2009 Base CHASSIS
"Status of Mda 1/1 changed administrative state: inService, operational state:
 inService"

3814 2008/04/22 23:35:38.88 UTC MINOR: CHASSIS #2002 Base Mda 1/2
"Class MDA Module : inserted"

3813 2008/04/22 23:35:38.88 UTC MINOR: CHASSIS #2002 Base Mda 1/1
```

6.7.4.6.2 Shutdown In-Band Port

A **shutdown** on the in-band port that the trap-target address is bound to causes the route to that particular trap target to be removed from the route table. When the SNMP module is notified of this event, it marks the trap-target as inaccessible and saves the sequence-id of the first SNMP notification that will be missed by the trap-target.

Example: config>log>snmp-trap-group# exit all
#configure port 1/1/1 shutdown
#
tools perform log test-event
#

The **Replay from** field is updated with the sequence-id of the first event that will be replayed when the trap-target address is added back to the route table.

```
*A:SetupCLI# show log snmp-trap-group 44
===============================================================================
SNMP Trap Group 44
===============================================================================
Description : none
-------------------------------------------------------------------------------
Name        : xyz-test
Address     : 10.10.10.3
Port        : 162
Version     : v2c
Community   : xyztesting
Sec. Level  : none
Replay      : enabled
Replay from : event #3819
```

```
Last replay : never
-------------------------------------------------------------------------------
Name        : test2
Address     : 10.20.20.5
Port        : 162
Version     : v2c
Community   : xyztesting
Sec. Level  : none
Replay      : disabled
Replay from : n/a
Last replay : never
===============================================================================
*A:SetupCLI#
```

A display of the event log indicates which trap targets are not accessible and waiting for notification replay and the sequence ID of the first notification that will be replayed.

Note: If there are more missed events than the log size, the replay will actually start from the first available missed event.

```
*A:SetupCLI# show log log-id 44
===============================================================================
Event Log 44
===============================================================================
SNMP Log contents  [size=100   next event=3821  (wrapped)]
Cannot send to SNMP target address 10.10.10.3.
Waiting to replay starting from event #3819

3820 2008/04/22 23:41:28.00 UTC INDETERMINATE: LOGGER #2011 Base Event Test
"Test event has been generated with system object identifier tmnxModelSR12Reg.
System description: TiMOS-B-0.0.private both/i386 Nokia 7750 SR Copyright (c)
2000-2016 Nokia. All rights reserved. All use subject to applicable license
agreements. Built on Tue Apr 22 14:41:18 PDT 2008 by test123 in /test123/ws/panos/
main"

3819 2008/04/22 23:41:20.37 UTC WARNING: MC_REDUNDANCY #2022 Base operational state
 of peer chan*
"The MC-Ring operational state of peer 2.2.2.2 changed to outOfService."

3818 2008/04/22 23:35:39.89 UTC WARNING: SYSTEM #2009 Base IP
"Status of vRtrIfTable: router Base (index 1) interface xyz-test (index 35) changed
administrative state: inService, operational state: inService"

3823 2008/04/22 23:41:49.82 UTC WARNING: SNMP #2005 Base xyz-test
"Interface xyz-test is operational"
```

6.7.4.6.3 No Shutdown Port

A **no shutdown** command executed on the in-band port to which the trap-target address is bound will cause the route to that trap target to be re-added to the route table. When the SNMP trap module is notified of this event, it resends the notifications that were missed while there was no route to the trap-target address.

Example: configure# port 1/1/1 no shutdown
#
tools perform log test-event

After the notifications have been replayed the **Replay from** field indicates n/a because there are no more notifications waiting to be replayed and the **Last replay** field timestamp has been updated.

```
*A:SetupCLI# show log snmp-trap-group 44
===============================================================================
SNMP Trap Group 44
===============================================================================
Description : none
-------------------------------------------------------------------------------
Name        : xyz-test
Address     : 10.10.10.3
Port        : 162
Version     : v2c
Community   : xyztesting
Sec. Level  : none
Replay      : enabled
Replay from : n/a
Last replay : 04/22/2008 18:52:36
-------------------------------------------------------------------------------
Name        : test2
Address     : 10.20.20.5
Port        : 162
Version     : v2c
Community   : xyztesting
Sec. Level  : none
Replay      : disabled
Replay from : n/a
Last replay : never
===============================================================================
*A:SetupCLI#
```

A display of the event log shows that it is no longer waiting to replay notifications to one or more of its trap target addresses. An event message has been written to the logger that indicates the replay to the trap-target address has happened and displays the notification sequence ID of the first and last replayed notifications.

```
*A:SetupCLI# show log log-id 44
===============================================================================
Event Log 44
===============================================================================
SNMP Log contents  [size=100   next event=3827  (wrapped)]
```

```
3826 2008/04/22 23:42:02.15 UTC MAJOR: LOGGER #2015 Base Log-id 44
"Missed events 3819 to 3825 from Log-id 44 have been resent to SNMP notification
 target address 10.10.10.3."

3825 2008/04/22 23:42:02.15 UTC INDETERMINATE: LOGGER #2011 Base Event Test
"Test event has been generated with system object identifier tmnxModelSR12Reg.
System description: TiMOS-B-0.0.private both/i386 Nokia 7750 SR Copyright (c)
2000-2016 Nokia.
All rights reserved. All use subject to applicable license agreements.
Built on Tue Apr 22 14:41:18 PDT 2008 by test123 in /test123/ws/panos/main"

3824 2008/04/22 23:41:49.82 UTC WARNING: SYSTEM #2009 Base IP
"Status of vRtrIfTable: router Base (index 1) interface xyz-test (index 35) changed
 administrative state: inService, operational state: inService"

3823 2008/04/22 23:41:49.82 UTC WARNING: SNMP #2005 Base xyz-test
"Interface xyz-test is operational"
```

6.7.4.7 Configuring a Syslog Target

Log events cannot be sent to a syslog target host until a valid syslog ID exists. The following displays a syslog configuration example:

```
A:ALA-12>config>log# info
----------------------------------------------
...
        syslog 1
            description "This is a syslog file."
            address 10.10.10.104
            facility user
            level warning
        exit
...
----------------------------------------------
```

6.7.4.7.1 Configuring an Accounting Custom Record

```
A:ALA-48>config>subscr-mgmt>acct-plcy# info
----------------------------------------------
..
            custom-record
                queue 1
                    i-counters
                        high-octets-discarded-count
                        low-octets-discarded-count
                        in-profile-octets-forwarded-count
                        out-profile-octets-forwarded-count
                    exit
                    e-counters
                        in-profile-octets-forwarded-count
                        in-profile-octets-discarded-count
                        out-profile-octets-forwarded-count
                        out-profile-octets-discarded-count
```

```
                    exit
                exit
                significant-change 20
                ref-queue all
                    i-counters
                        in-profile-packets-forwarded-count
                        out-profile-packets-forwarded-count
                    exit
                    e-counters
                        in-profile-packets-forwarded-count
                        out-profile-packets-forwarded-count
                    exit
                exit
..
-----------------------------------------------
```

The following is an example custom record configuration.

```
Dut-C>config>log>acct-policy>cr# info
-----------------------------------------------
                aa-specific
                    aa-sub-counters
                        short-duration-flow-count
                        medium-duration-flow-count
                        long-duration-flow-count
                        total-flow-duration
                        total-flows-completed-count
                    exit
                    from-aa-sub-counters
                        flows-admitted-count
                        flows-denied-count
                        flows-active-count
                        packets-admitted-count
                        octets-admitted-count
                        packets-denied-count
                        octets-denied-count
                        max-throughput-octet-count
                        max-throughput-packet-count
                        max-throughput-timestamp
                        forwarding-class
                    exit
                    to-aa-sub-counters
                        flows-admitted-count
                        flows-denied-count
                        flows-active-count
                        packets-admitted-count
                        octets-admitted-count
                        packets-denied-count
                        octets-denied-count
                        max-throughput-octet-count
                        max-throughput-packet-count
                        max-throughput-timestamp
                        forwarding-class
                    exit
                exit
                significant-change 1
                ref-aa-specific-counter any
-----------------------------------------------
```

7 Node Discovery Provisioning Using OSPF

Some operators use third-party or self-build NMS and need to discover the nodes using OSPF and its Type 10 opaque LSA TLV.

When a node is discovered through OSPF, the NMS will push the new configuration to the node and perform any other required modifications using CLI (SSH and/or Telnet).

7.1 Node Discovery Procedure

A node is configured with a network element profile. A network element profile has all of the necessary information for node discovery, including the node NEID, NEIP, Vendor Identifier, chassis type, and MAC address. The network element profile can be added to an OSPF area in VPRN and when it is, the network element information is advertised using OSPF type 10 opaque LSA to the rest of the network. These discovery procedures are only available in VPRN.

An aggregation node (node closest to the NMS) gathers all of the network element information received using OSPF and converts this information into a MIB. The aggregation node also generates traps as necessary to update the NMS when the **configure system network-element-discovery generate-traps** command is configured. In addition to the traps, the NMS can walk the MIB table to update its new view of the network.

The node can be discovered using an IPv4 or IPv6 NEIP but the advertisement protocol is always OSPFv2; OSPFv3 is not supported. If the NMS wants to discover a node in an IPv6 network, both OSPFv2 and OSPFv3 must be enabled. OSPFv2 will advertise the node information to the aggregation node to generate the traps and build the MIB table but OSPFv3 will be used to provide the routing information needed to reach that node using an IPv6 network.

The minimum configuration required on the node so that it can be discovered includes:

- a management VPRN
- a network element profile configured under the **config**>**system**>**network-element-discovery** context
- a loopback or physical L3 interface in the VPRN with same IP address as the network element profile NEIP

- a physical interface in the VPRN (SAP) with an optional unnumbered interface inheriting the IP address from the loopback IP (this is light VPRN where the uplink is a SAP and not a spoke-sdp)
- an optional VLAN DOT1Q support for the L3 interface
- a temporary username and password. This could be the default.
- OSPF on the physical L3 interface in the VPRN (SAP). OSPF must be configured as P2P and has an additional flag to include the LSA type 10 opaque value.
- SSH and/or Telnet management enabled on the VPRN, these are disabled by default

As all discovery is done using the VPRN SAP, there is no need for GRT configuration.

7.1.1 Network Element Profiles

A network element profile is created for the system with all the information needed for node discovery using the commands below. This information will be flooded to the network using IGP.

```
config>system>network-element-discovery
   profile <profile name>
      neid
      neip
      system-mac
      platform-type
      vendor-id
```

This profile can be assigned to a VPRN OSPFv2 area for advertisement. Only one profile can be created for each node.

If a value is not set, the following default values are used for these optional parameters:

- system-mac = chassis-mac
- platform-type = "chassis-name, chassis-type"
- vendor-id = "Nokia"

7.1.2 Assigning a Network Element Profile

When a network element profile is assigned to OSPFv2, its information will be advertised using LSA type 10 opaque. Use the following commands to assign a network element profile to OSPFv2:

```
config>service>vprn
  ospf
    area 0
      advertised-ne-profile <profile-name>
      interface loopback
      interface uplink
```

The network element information will not be added to the routing table (RIB) or forwarding table (FIB), this includes the NEIP. If the NEIP of the profile needs to be visible to the network then a loopback interface or a physical interface must be configured with the same IP address as the NEIP and added to the OSPFv2 area.

For example, if the following profile is created, the NEIP is duplicated in a loopback interface address. This is because the NEIP is only used in type 10 LSA and is not added to the RIB or FIB. Type 10 LSA is only used to relay the information to the NMS. As such, the loopback interface with the same IP address as the NEIP, must be assigned to the same OSPF area to ensure the address is injected into the RIB and FIB of all nodes and is reachable. If this loopback interface is not configured and not added to OSPF then SR OS will not have any route entry in the RIB or FIB for the NEIP.

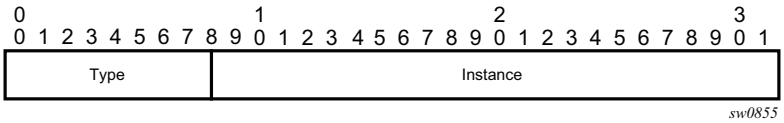
```
config>system>network-element-discovery
  profile <name-1>
    neid 0x091001
    neip 128.9.10.1
config>service>vprn 1
  ospf
    area 0
      advertised-ne-profile <name-1>
      interface loopback
      interface uplink
  interface loopback
    loopback
    address 128.9.10.1
```

7.1.3 OSPFv2 Opaque LSA Requirements

OSPF Opaque LSA packets are made up of the following, as shown in Figure 21.

- LSA ID: 202.255.238.0
- Opaque type value: 202
- Instance value: 255.238.0 (ffee00)

Figure 21 **OSPF Opaque LSA Packet**



Note: Type 10 opaque LSA can only be added to OSPFv2 and not OSPFv3.

OSPF Opaque LSA type 10 TLV contains the following information:

- Vendor identifier — vendor-id (vendor defined string)
  - type — 0x8000
  - length — length of identify string
  - Default value — "Nokia"
- Equipment identifier — product-id (vendor defined string)
  - type — 0x8001
  - length — length of identify string
  - default value — "chassis-name, chassis-type"
- System-Mac — node system MAC (node unique)
  - type — 0x8002
  - length — 6
  - default value — chassis mac address
- NEID — Node hostname (customer defined)
  - type — 0x8003
  - length — 4
- NEIP IPv4 — loopback IPv4 address (preconfigured)
  - type — 0x8004
  - length — 4
- NEIP IPv6: loopback IPv6 address (preconfigured)
  - type — 0x8005
  - length — 16

7.1.4 IPv4/IPv6

Both IPv4 and IPv6 node discovery is supported. However, LSA type 10 is only advertised on OSPFv2 and not OSPFv3. If the operator wants to use IPv6 or dual-stack, OSPFv2 must be configured under VPRN to advertise the node information using LSA type 10 opaque. In addition, OSPFV3 can also be enabled to advertise IPv6 routes and interfaces for IPv6 reachability.

If both IPv4 and IPv6 are configured on the loopback interface within the VPRN (dual-stack) then both values are sent in the OSPFv2 LSA type 10, NMS will prefer one over the other. In this case, all discovered routers must be configured with dual-stack OSPFv2 and OSPFv3.

7.2 Aggregation Node Configuration

An aggregation node, attached to NMS through a VPRN service, aggregates all nodes to be discovered using the same VPRN. The node is configured with:

- a management VPRN
- a loopback address in VPRN
- a physical interface in the VPRN with an optional unnumbered interface inheriting the IP address from the loopback IP (this is a light VPRN where the uplink is a SAP and not a spoke-sdp)
- optional VLAN DOT1Q support for the L3 interface
- OSPF on the physical L3 interface in the VPRN (SAP)
- SNMP (SNMPv3) on an L3 interface in the VPRN (SAP) toward the NMS
- a MIB table build based on the OSPF type 10 opaque LSA and stored in the OSPF opaque database

This node converts the OSPF LSA type 10 opaque information arriving from the discovered nodes into an SNMP MIB table and updates the NMS with the MIB. The NMS uses the MIB to discover the node and opens a SSH or Telnet session to configure the node accordingly.

The node to be discovered is provisioned with a management VRF as described in Node Discovery Procedure. All of the necessary node information needed for the node discovery is advertised by OSPF using LSA type 10 opaque. Nodes can be chained together, so the LSA type 10 needs to be forwarded all the way to the aggregation node.

NMS discovers the node using SNMP MIB traps or it can walk the MIB table.

NMS connects to the discovered node using SSH or Telnet to update the configuration by CLI. The operator must ensure that none of their commands will disable the management VPRN and disconnect the discovered node from NMS.

7.2.1 MIB Requirements on the Aggregation Node

A new vendor-proprietary MIB table for node discovery, tmnxVRtrNeInfoTable, is defined. This MIB table is used by the NMS to gather all node information by walking the table, for example, after an NMS reboot.

This MIB table is built on any node that receives OSPF type 10 LSA.

The MIB table should be built with a key of VRF and NEID and a data portion of neid, vendor identify, platform type, system-mac, neip-v4, and neip-v6.

The MIB parameters are:

- The company name is configurable using the CLI and is a string, but by default, "Nokia" is sent.

```
tmnxSysNEProfVendorId            OBJECT-TYPE
    SYNTAX      DisplayString
    MAX-ACCESS  read-create
    STATUS      current
    DESCRIPTION
       "The value of tmnxSysNEProfVendorId specifies the
 vendor identifier."
    DEFVAL      { "Nokia" }
    ::= { tmnxSysNEProfEntry 11 }
```

- The device type is configurable and is a string, but by default, the chassis-name and the chassis-type is sent.

```
tmnxSysNEProfPlatformType        OBJECT-TYPE
    SYNTAX      DisplayString
    MAX-ACCESS  read-create
    STATUS      current
    DESCRIPTION
       "The value of tmnxSysNEProfPlatformType specifies
 the product identifier.
        An empty string indicates this object is not
 configured, the chassis name and chassis type will be
 used."
    DEFVAL      { ''H }
    ::= { tmnxSysNEProfEntry 10 }
```

- The tcIPRanDcnIpv4 and tcIpRanDcnIpV6 objects are the NEIPs configured using the CLI.

```
tmnxSysNEProfNeipV4Type          OBJECT-TYPE
    SYNTAX      InetAddressType
    MAX-ACCESS  read-create
    STATUS      current
    DESCRIPTION
        "The value of tmnxSysNEProfNeipV4Type specifies
 the IP address type of tmnxSysNEProfNeipV4.
      The value of tmnxSysNEProfNeipV4Type can be either
 of 'ipv4(1)' or 'unknown (0)'.
        The value of 'unknown(0)' specifies no NEIP v4
 address is configured."
    DEFVAL      { unknown }
    ::= { tmnxSysNEProfEntry 5 }

tmnxSysNEProfNeipV4              OBJECT-TYPE
    SYNTAX      InetAddress (SIZE (0|4))
    MAX-ACCESS  read-create
    STATUS      current
    DESCRIPTION
        "The value of tmnxSysNEProfNeipV4 indicates the
 IPv4 address of the Network Element."
    DEFVAL      { ''H }
    ::= { tmnxSysNEProfEntry 6 }

tmnxSysNEProfNeipV6Type          OBJECT-TYPE
    SYNTAX      InetAddressType
    MAX-ACCESS  read-create
    STATUS      current
    DESCRIPTION
        "The value of tmnxSysNEProfNeipV6Type specifies
 the IP address type of tmnxSysNEProfNeipV6.
      The value of tmnxSysNEProfNeipV6Type can be either
 of 'ipv6(2)' or 'unknown (0)'.
        The value of 'unknown(0)' specifies no NEIP v6
 address is configured."
    DEFVAL      { unknown }
    ::= { tmnxSysNEProfEntry 7 }

tmnxSysNEProfNeipV6              OBJECT-TYPE
    SYNTAX      InetAddress (SIZE (0|16))
    MAX-ACCESS  read-create
    STATUS      current
    DESCRIPTION
        "The value of tmnxSysNEProfNeipV6 indicates the
 IPv6 address of the Network Element."
    DEFVAL      { ''H }
    ::= { tmnxSysNEProfEntry 8 }
```

```
tmnxSysNEProfSystemMac           OBJECT-TYPE
    SYNTAX      MacAddress
    MAX-ACCESS  read-create
    STATUS      current
    DESCRIPTION
        "The value of tmnxSysNEProfSystemMac specifies
 the system MAC address of the node.
         A value of all zeros indicates this object is
 not configured, the chassis MAC address will be used."
    DEFVAL      { '000000000000'H }
    ::= { tmnxSysNEProfEntry 9 }
```

The MIB walk is as follows:

tmnxVRtrNeInfoNeidHex.7.0.65.66.67 = STRING: 0:41:42:43

tmnxVRtrNeInfoNeipV4Type.7.0.65.66.67 = INTEGER: ipv4(1)

tmnxVRtrNeInfoNeipV4.7.0.65.66.67 = Hex-STRING: 80 09 0A 01

tmnxVRtrNeInfoNeipV4PrefixLen.7.0.65.66.67 = Gauge32: 32

tmnxVRtrNeInfoNeipV6Type.7.0.65.66.67 = INTEGER: ipv6(2)

tmnxVRtrNeInfoNeipV6.7.0.65.66.67 = Hex-STRING: 3F FE 00 00 00 00 00 00 00 00 00 00 80 09 0A 01

tmnxVRtrNeInfoNeipV6PrefixLen.7.0.65.66.67 = Gauge32: 128

tmnxVRtrNeInfoSystemMac.7.0.65.66.67 = STRING: e:0:0:0:0:1

tmnxVRtrNeInfoPlatformType.7.0.65.66.67 = STRING: Dut-B,7750 SR-12e_ndef

tmnxVRtrNeInfoVendorId.7.0.65.66.67 = STRING: NokiaNotDefault

7.2.2 SNMP Traps and Gets

The discovery table should be walkable by the NMS. The NMS can walk the entire table or get a specific row using the appropriate key. This is required when the NMS is rebooted or needs to update its entire database.

In addition, every time a node is updated, added, or removed from the OSPF opaque database (using LSA type 10 opaque update) a trap is sent to the NMS to notify the NMS of the change if the **configure system network-element-discovery generate-traps** command is configured. These traps are:

- {tmnxVRtrNeInfoNeidHex.4.0.65.66.67 00:41:42:43}
- {tmnxVRtrNeInfoNeipV4Type.4.0.65.66.67 ipv4}
- {tmnxVRtrNeInfoNeipV4.4.0.65.66.67 0x80:09:0a:01}
- {tmnxVRtrNeInfoNeipV4PrefixLen.4.0.65.66.67 32}
- {tmnxVRtrNeInfoNeipV6Type.4.0.65.66.67 ipv6}
- {tmnxVRtrNeInfoNeipV6.4.0.65.66.67 0x3f:fe:00:00:00:00:00:00:00:00:00:80:09:0a:01}
- {tmnxVRtrNeInfoNeipV6PrefixLen.4.0.65.66.67 128}
- {tmnxVRtrNeInfoSystemMac.4.0.65.66.67 0e:00:00:00:00:01}
- {tmnxVRtrNeInfoPlatformType.4.0.65.66.67 {Dut-B,7750 SR-12e_ndef}}
- {tmnxVRtrNeInfoVendorId.4.0.65.66.67 NokiaNotDefault}

8 sFlow

8.1 sFlow Overview

Some Layer 2 network deployments collect statistics on physical Ethernet ports and on Layer 2 interfaces at a high-frequency using a push model to, among others, monitor traffic, diagnose network issues, and/or provide billing. SR OS supports cflowd and XML accounting; however, those mechanisms are either Layer 3-specific, or focus on providing statistics at extremely large scale (thus use a pull model and cannot support high-frequency counter updates). To meet the statistics collection requirements of such Layer 2 deployments, SR OS supports sFlow statistics export using sFlow version 5.

The following list gives the main caveats for sFlow support:

- sFlow data sources require multi-core line cards (IOM), enabling sFlow on a card that is not a multi-core is not blocked and can be detected by SNMP trap/log generated by sFlow
- To meet high-frequency export of counters, sFlow implementation is targeted for low per-port VLL/VPLS SAP scale only. The configuration is blocked if the per-port VLL/VPLS SAP limit exceeds sFlow limit. Contact your Nokia representative for per-platform scaling limits applicable.

8.2 sFlow Features

This section describes sFlow functionality supported in SR OS.

8.2.1 sFlow Counter Polling Architecture

When sFlow is enabled on an SR OS router, the system takes upon a role of an sFlow network device as described in sFlow protocol version 5. A single sFlow agent can be configured for counter polling (flow sampling is not supported). There is no support for sub-agents.

The sFlow agent sends sFlow data to an operator-configured sFlow receiver. A single receiver is supported with configurable primary and backup IPv4 or IPv6 UDP destination sockets for redundancy (each sFlow packet exported is duplicated to both sockets when both are configured). The receiver's UDP sockets can be reachable either in-band or out-of-band (default) and must both be IPv4 or IPv6. An operator can also set the maximum size of the sFlow datagrams. Operators are expected to set this value to avoid IP fragmentation (Datagrams exceeding the specified size are fragmented before handed to IP layer).

The sFlow agent manages all sFlow data sources in the system. SR OS supports sFlow data that are physical ports. When a port is configured as an sFlow data source, counters for that port and all VPLS and Epipe SAPs on that port are collected and exported using sFlow (see later on section for record format). Flow data sources can only be configured when an sFlow receiver is configured. To remove the sFlow receiver, all sFlow data sources must first be deconfigured at the port level.

Each data source is processed at a 15-second, non-configurable interval. If multiple data sources exist on a line card, the line card distributes the processing of each data source within a 15 second interval to avoid sFlow storms. When a timer expires to trigger a data source processing, data is collected for the physical port and for all VLL and VPLS SAPs on that port and exported using sFlow version 5 records as described in later subsections of this document. Each port and all SAP records for a given data source for a given interval are collected and sent with the counter sequence number and the timestamp value (the time value corresponds to the time counters were actually collected by a line card). The timestamp value uses line card's sysUptime value, which is synchronized with CPM time automatically by the system. A line card sends the counters to a CPM card, where sFlow UDP datagrams are created, sequenced with the CPM sequence number and sent to the receiver. If no UDP sockets are configured, no errors are generated because data is not sent. If no UDP sockets are reachable, the created UDP sFlow datagrams are dropped.

Note: Line cards will reset the counter record sequence numbers if, as a result of configuration or operational change, the return statistics no longer provide continuity with the previous interval. This may occur when:

- The card hard or soft resets
- The MDA resets
- The sFlow agent counter map changes

Note: The CPM will reset the sFlow datagram sequence numbers if, as a result of configuration or operational change, the sFlow datagram to be sent no longer provides continuity with the previous datagram. The following lists examples of when this takes place:

- HA switch
- CTL reboot
- Creation of an sFlow receiver

8.2.2 sFlow Support on Logical Ethernet Ports

sFlow data sources operate in a context of physical Ethernet port. To enable sFlow on Ethernet logical ports and their SAPs, an operator must explicitly enable sFlow on every physical Ethernet port that is a member of the given logical port. Currently only LAG logical ports are supported (including MC-LAG).

Note: sFlow configuration does not change automatically when a port is added or removed to or from a LAG.

For SAPs on a LAG, egress statistics will increment based on ports used by each SAP on LAG egress while ingress statistics will increment based on ports used by each SAP on LAG ingress unless LAG features like, for example, per-fp-ingress-queuing or per-fp-sap-optimization result in SAP statistics collection against a single LAG port.

If logical-level view is required, for example, per LAG statistics, a receiver is expected to perform data correlation based on per-physical port interface and SAP records exported for the given logical port's physical ports and their SAPs. sFlow data records contain information that allows physical ports/SAP records correlation to a logical port. See sFlow Record Formats.

Note: Correlation of records must allow for small difference in timestamp values returned for member ports or SAP on a LAG because all ports run independent timestamps.

8.2.3 sFlow SAP Counter Map

To allow per SAP sFlow statistics export, operators must configure ingress and egress sFlow counter maps. The counter maps are required, because SR OS systems support more granular per policer/queue counters and not IF-MIB counters per VLL/VPLS SAPs. In an absence of a map configured, 0's will be returned in corresponding statistics records.

A single ingress and a single egress counter map are supported. The maps specify which ingress and which egress SAP QoS policy queue/policer statistics map to sFlow unicast, multicast, and broadcast counters returned in an sFlow SAP record. Multiple queues and/or policers can map to each of unicast, multicast, broadcast counters. A single queue/policer can only map to one type of traffic. Queues, policers configured in a SAP QoS policy but not configured in an sFlow map or vice-versa are ignored when sFlow statistics are collected.

8.2.4 sFlow Record Formats

Table 44 describes sFlow record used and exported:

Table 44 **sFlow Record Fields**

Record	Field	Value
sFlow Datagram Header (SAP and port)	Datagram version	5
	Agent Address	Active CPM IPv4 address (from BoF)
	Sub-agent ID	0
	Sequence number	CPM inserted sFlow datagram sequence number
	SysUptime	sysUptime when the counters for records included in the datagram were collected by the line card
	NumSamples	Number of counter records in the datagram

Table 44 **sFlow Record Fields (Continued)**

Record	Field	Value
Counter header (SAP and Port)	Enterprise	0 (standard sFlow)
	sFlow Sample Type	4 (Expanded counter sample)
	Sample Length	sFlow packet size excluding header
	Sequence number	Line card-inserted sequence number
	Source ID Type	0
	Source ID Index	tmnxPortId of the physical port (sFlow data source)
	Counter records	Count of counter records in the datagram
Ethernet Interface Counters (EIC) – port (Ethernet Layer)	Enterprise	Statistics returned are based on dot3StatsEntry in EtherLike-MIB.mib. Statistics support may depend on hardware type.
	Format	
	Flow data length	
	Alignment Errors	
	FCS Errors	
	Single Collision Frames	
	Multiple Collision Frames	
	SQE Test Errors	
	Deferred Transmissions	
	Late Collisions	
	Excessive Collisions	
	Internal Mac Transmit Errors	
	Carrier Sense Errors	
	Frame Too Longs	
	Internal Mac Receive Errors	
	Symbol Errors	

Table 44 **sFlow Record Fields (Continued)**

Record	Field	Value
Generic Interface Counters (GIC) – port/ SAP	Enterprise	0 (standard sFlow)
	Format	1 (GIC)
	Flow data length	88
	ifIndex	Port: ifIndex (tmnxPortId) of phys port SAP: SapEncapValue - part of SAP SNMP key
	ifType	Port: 6 (EthernetCsmacd) SAP: 1 (Other)
	ifSpeed	Port: Port speed value SAP: • top 32 bits: svcId for SAP (TIMETRA-SAP.mib) • lower 32 bits: sapPortId (TIMETRA-SAP.mib) The values plus ifIndex in the record are SAP SNMP key. SapPortId is LAG's tmnxPortId for SAPs on a LAG and port's tmnxPortId for SAPs on physical port
	ifDirection	Derived from MAU MIB (0 = unknown, 1 = full duplex, 2 = half duplex, 3 = in, 4 = out)
	ifAdminStatus	0 (down) 1 (up)
	ifOperStatus	0 (down) 1 (up)
	Input Octets	Statistics return for port are based on ifEntry or ifXEntry in IF-MIB.mib as applicable. Statistics returned for SAPs are sum of counters based on the sFlow ingress/ egress counter map configured.
	Input Packets	
	Input Multicast packets	
	Input Broadcast packets	
	Input Discarded packets	

Table 44 **sFlow Record Fields (Continued)**

Record	Field	Value
Generic Interface Counters (GIC) – port/ SAP (Continued)	Input Errors	Statistics return for port are based on ifEntry or ifXEntry in IF-MIB.mib as applicable. Statistics returned for SAPs are sum of counters based on the sFlow ingress/ egress counter map configured.
	Input Unknown Protocol Packets	
	Output Octets	
	Output Packets	
	Output Multicast packets	
	Output Broadcast packets	
	Output Discarded packets	
	Output Errors	
	Promiscuous Mode	0 (FALSE)

Notes:

- 0 is returned for statistics that are not supported by a given hardware type.
- If required, CPM executes rollover logic to convert internal 64-bit counters to a 32-bit sFlowd counter returned.

9 gRPC

gRPC is a modern, open-source, high-performance RPC framework that runs in any environment. In SR OS, this framework is used to implement the gRPC server, which can then be used for configuration management or telemetry.

The gRPC transport service uses HTTP/2 bidirectional streaming between the gRPC client (the data collector) and the gRPC server (the SR OS device). A gRPC session is a single connection from the gRPC client to the gRPC server over the TCP/TLS port.

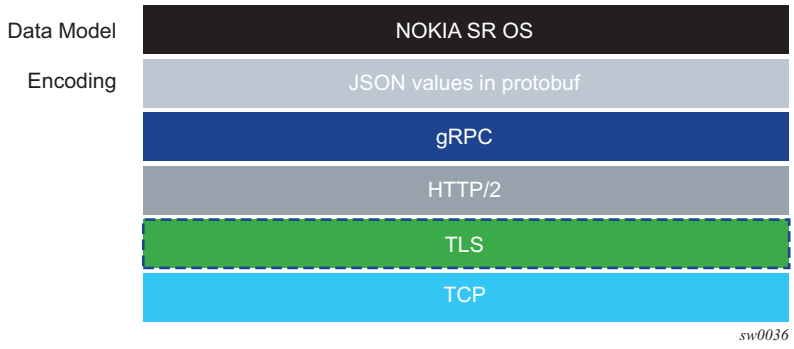
The gRPC service runs on port 57400 by default in SR OS. The service is not configurable.

A single gRPC server supports concurrent gRPC sessions and channels.

- There is a maximum of eight concurrent gRPC sessions for all of the gRPC clients.
- There is a maximum of 225 concurrent gRPC channels for all of the gRPC clients.

Figure 22 shows the gRPC protocol stack.

Figure 22 **Protocol Stack**



9.1 Security Aspects

9.1.1 TLS-Based Encryption

The gRPC server on SR OS can operate in two modes:

- without TLS encryption
- with TLS encryption

TLS encryption is used for added security. However, TLS encryption can be disabled in lab environments.

If TLS is not used, gRPC messages are not encrypted and user-names and passwords required in gRPC communication are visible to anyone capturing the packets. Therefore, Nokia recommends disabling TLS encryption only in a closed environment.

Before a gRPC connection will come up without TLS, the following conditions must both be met:

- no TLS server profile is assigned to the gRPC server
- the **allow-unsecure-connection** flag is set

The following summarizes the process of encryption:

- To use TLS encryption:
  - The gRPC session must be in an encrypted state.
  - If the gRPC client and gRPC server are unable to negotiate an encrypted gRPC session, the gRPC session fails and the gRPC server sends an error.
  - Fallback from an encrypted to an unencrypted gRPC session is not allowed.

  For information about how to configure TLS with gRPC, see the TLS chapter.

9.1.2 Authentication

The gRPC users can be authenticated using the local user database, RADIUS or TACACS+.

When using the local user database, the **access grpc** statement must be included for the user.

For RADIUS, the **access grpc** statement must be configured in **user-template radius_default** (and **radius use-default-template** must be enabled), or the RADIUS server must send the Timetra-Access VSA with a value that includes grpc access.

For TACACS+, the **access grpc** statement must be configured in **user-template tacplus_default** (and **tacplus use-default-template** must be enabled).

User authentication is based on following principles:

- Each RPC sent by the gRPC client carries a username and password.
- For the first RPC in the gRPC session, the gRPC server tries to authenticate the user using the specified authentication order, such as using the local user database, RADIUS, or TACACS+.

  For example, if TACACS+ is first in the authentication order, the gRPC server sends a request to the TACACs+ server to authenticate the gRPC user.
- For the subsequent RPCs on that same authenticated gRPC session, the username and password are re-authenticated only if changed.
- When no username and password are provided with the RPC, the gRPC server returns an error.
- If the RPC user is changed, any active subscriber RPCs on that same gRPC session are terminated by the gRPC server.
- If the RPC password is changed, the active gRPC session will continue to exist until a different username and password is sent in a subsequent RPC, or the gRPC session is terminated.
- Each message is carried over a gRPC session that was previously encrypted; the session is not re-encrypted.
- SR OS device authentication
  - The gRPC clients do not share gRPC sessions. Each gRPC client starts a separate gRPC session.
  - When a gRPC session is established, the gRPC server certificates are verified by the gRPC client to ensure that every gRPC server is authenticated by the gRPC client.
  - If gRPC is shut down on the gRPC server and a gRPC client is trying to establish a gRPC session, the gRPC client will get an error for every RPC sent.
  - If gRPC is shut down on the gRPC server and a gRPC session is established, all active RPCs are gracefully terminated and an error is returned for every active RPC.

9.2 gNMI Service

The gRPC Network Management Interface (gNMI) is a gRPC based protocol for network management functions, such as changing the configuration of network elements and retrieving state information. In addition, gNMI provides functionality necessary for supporting telemetry. The gNMI service is specified in the OpenConfig forum.

9.2.1 gNMI Service Definitions

The SR OS gRPC server supports gNMI version 0.7.0, and in particular, the following RPC operations:

- Capability RPC
- Set/Get RPCs
- Subscribe RPC

As in NETCONF RPCs, gNMI RPCs that are sent to the SR OS system are logged in security log and they are marked as authorized or unauthorized, and include information such as username, time, RPC type, and IP address of the client.

9.2.1.1 Capability Discovery

In gNMI service, the client discovers the capabilities of the gRPC server through a Capability-Discovery RPC, which consists of “CapabiltyRequest” and “CapabilityResponse” messages.

During this message exchange, the gRPC server informs the client about following attributes:

- supported gNMI version
- supported models
- supported encodings

The SR OS server announces the supported models based on the configuration under **config**>**system**>**management-interface**>**yang-modules**. The supported models includes the NOKIA-YANG or OpenConfig (OC) models.

The advertised module names and organizations are as follows:

- nokia-conf, org = "Nokia"
- nokia-state, org = "Nokia"
- openconfig, org = "OpenConfig working group" (as specified by the 'organization' in the YANG models)
- version - the version number is be defined as follows:
  - for NOKIA YANG models, the version number corresponds to an SR OS release number, for example, "16.0.r1"
  - for OC YANG models, the version number corresponds to a version number defined in "oc-ext:openconfig-version" that is included in the respective YANG models
  - for OC-YANG models, including NOKIA deviations, the version number corresponds to an SR OS release number, for example, "16.0.r1"

The following is an example of a “Capabilities Response Message”:

```
Going to send message of type gnmi.CapabilityResponse:
 .gnmi_version: 0.4.0
 .supported_encodings (1):
  .encoding: 0 = JSON
 .supported_models (47):
  { .name: 'nokia-conf', .organization: 'Nokia', .version: '16.0.r1' }
  { .name: 'nokia-state', .organization: 'Nokia', .version: '16.0.r1' }
  { .name: 'openconfig-
bgp', .organization: 'OpenConfig working group', .version: '4.0.1' }
<snip>
  { .name: 'nokia-sr-openconfig-if-ethernet-
deviations', .organization: 'Nokia', .version: '16.0.r1' }
  { .name: 'nokia-sr-openconfig-if-ip-
deviations', .organization: 'Nokia', .version: '16.0.r1'..."
```

9.2.1.2 Get/Set RPC

Information is retrieved from the NE using GET RPC messages, which consists of “GetRequest” and “GetResponse” messages. The client asks for a given information by specifying following:

- A set of paths — all rules to a path definition apply, as specified in the gNMI specification
- Type — configuration, state, or operational data
- Encoding — in accordance to server advertisement during capability discovery
- Use_models — this message is ignored

There is an upper limit on the size of the “GetResponse” message. This limit cannot exceed 100MB. If the limit is exceeded, the SR OS gRPC server responds with an error message.

In order to modify the information in an NE element, a SET gRPC message is used. This gRPC supports three types of transactions:

- delete
- replace
- update

With a gNMI SET RPC, SR OS authorizes all configuration changes, that is, it checks the YANG tree and authorizes every changed element.

The deletion of a container results in the deletion of any children containers that are authorized for deletion as well as their contents. Children containers that are not authorized for deletion, as well as their contents, are retained. For example, upon deletion of **configure system, configure system security** is not deleted because the deletion of that child container is not authorized.

Note: Only changes to configuration values are checked for authorization. A configuration command that simple writes the same value to a leaf will succeed even if the user does not have access to that leaf (but there will be no resulting change to the configuration).

For example, when a user is not authorized to change **access li,** but attempts to change it for another user who already has **access li**, SR OS allows that action because there is no change in value.

9.2.1.3 Subscribe RPC

The subscribe RPC is part of the telemetry support in gNMI.

A subscription is initiated from the gRPC client by sending a Subscribe RPC that contains a "SubscribeRequest" message to the gRPC server. A prefix can be specified to be used with all paths specified in the "SubscribeRequest". If a prefix is present, it is appended to the start of every path to provide a full path.

A subscription contains:

- a list of one or more paths. The following conditions apply:
  - A path represents the data tree as a series of repeated strings and elements. Each element represents a data tree node name and its associated attributes.

- A path must be syntactically valid within the set of schema modules that the gRPC server supports.
- The path list cannot be modified during the lifetime of the subscription.
- If the subscription path is to a container node, all child leafs of that container node are considered to be subscribed to.
- Any specified path must be unique within the list; paths cannot be repeated within the list. An error is returned if the same path is used more than one time in a single subscription.
- A specified path does not need to pre-exist within the current data tree on the gRPC server. If a path does not exist, the gRPC server continues to monitor for the existence of the path. Assuming that the path exists, the gRPC server transmits telemetry updates.
- The gRPC server does not send any data for a non-existent path; for example, if a path is non-existent at the time of subscription creation or if the path was deleted after the subscription was established.
- The maximum number of paths for all subscriptions on a single SR OS device is 14400. A path using a wildcard is still considered a single path.

• a subscription mode of one of the following types:

- ONCE mode — the server returns only one notification containing all information the client has subscribed to. In general, retrieving large amounts of information from the NE can be done using telemetry: “SubscribeRequest” message with ONCE subscription type.
- ON_CHANGE mode — the server returns notifications only when the value of the subscribed field changes. See ON_CHANGE Subscription Mode for more information.
- SAMPLE mode — the gRPC server sends notifications at the specified sampling interval
- TARGET_DEFINED mode — means ON_CHANGE for all states supporting ON_CHANGE notifications and SAMPLE mode for all other objects in the YANG tree

• a sample interval is supported for each path. If a sample interval of less than 1 s is specified, the gRPC server returns an error. If the sample interval is set to 0, the default value of 10 s is used. A sample interval is specified in nanoseconds (10 000 000 000 by default)

When a subscription is successfully initiated on the gRPC server, “SubscribeReponse” messages are sent from the gRPC server to the gRPC client. The “SubscribeResponse” message contains update notifications about the subscription's path list.

An update notification contains:

- a timestamp of the statistics collection time, represented in nanoseconds
- a prefix:
  - If a prefix is present, it is logically appended to the start of every path to provide the full path.
  - The presence of a prefix in the “SubscriptionResponse” message is not related to the presence of a prefix in the original “SubscriptionRequest” message. The prefix in the “SubscriptionResponse” message is optimized by the gRPC server.
- a list of updates (path and value pairs):
  - A path represents the data tree path as a series of repeated strings or elements, where each element represents a data tree node name and its associated attributes. See Schema Paths for more information.
  - The “TypedValue” message represents the value of the data tree node, where the encoding is “JSON”, “Bytes”, or “Protobuf” depending on the information in the “SubscribeRequest” message.

A sync response notification is sent one time, after the gRPC server sends all of the updates for the subscribed-to paths. The sync response must be set to “true” for the gRPC client to consider that the stream has synced one time. A sync response is used to signal the gRPC client that it has a full view of the subscribed-to data.

The gRPC server sends an error if required. The error contains a description of the problem.

Authorization checks are not performed by default for telemetry data. All configuration and state elements are available to authenticated telemetry subscriptions, with the exception of LI (Lawful Intercept) configuration and state elements, which are authorized separately based on the LI authorization configuration. To control telemetry data authorization, use the classic CLI **configure>system>security>managment-interface>output-authorization>telemetry-data** command or the MD-CLI **configure system security aaa management-interface output-authorization telemetry-data** command.

9.2.1.3.1 Bytes Encoding

Bytes encoding is performed by the gRPC server, which encodes the values of the leafs as typed values. Table 45 lists the mapping of the individual YANG types to the typed values defined in the gNMI specification in the GitHub repository.

Table 45 **Mapping of YANG Types to gNMI-specified Typed Values**

YANG Type	Typed Value
binary	bytes_val
bits	string_val
boolean	bool_val
decimal64	decimal_val
empty	bool_val
enumeration	string_val
identityref	string_val
instance-identifier	string_val
int8	int_val
int16	int_val
int32	int_val
int64	int_val
leafref	type of referenced leaf
string	string_val
uint8	uint_val
uint16	uint_val
uint32	uint_val
uint64	uint_val
union	type of active union member

9.2.1.3.2 ON_CHANGE Subscription Mode

SR OS supports ON_CHANGE subscription mode. This subscription mode indicates that Notification messages are sent as follows:

- after the “SubscriptionRequest” message is received
- every time the corresponding leaf value is changed

The notification message, as a response to an ON_CHANGE subscription, always contains the new value of the corresponding leaf, as defined in gNMI specification.

The ON_CHANGE subscription is supported for all configuration events as well as for selected state leafs. The **tools** command can display all state leafs supporting the ON_CHANGE subscription.

ON_CHANGE subscription is accepted for all valid paths. The server sends ON_CHANGE notifications only for leafs within this path that support ON_CHANGE notifications.

9.2.1.4 Publish RPC

With dial-out telemetry, where the SR OS node is the gRPC client instead of the gRPC server, the SR OS node sends a Publish RPC with a “SubscribeResponse” message to the gRPC server. (See Dial-out Telemetry.)

Because the current gnmi.proto definition does not support dial-out mode, a protobuf definition is introduced, with a separate gRPC service, as follows:

```
NOKIA-DialOut.proto
      option (dialout_service) = 0.1.0
      service gMIDialOut {
              rpc Publish(stream SubscribeResponse) returns (stream PublishResponse)
      }
      message PublishResponse {
      }
```

The preceding proto file definition reuses the “SubscribeResponse” message defined for dial-in telemetry, in accordance with the gNMI specification.

9.2.1.5 Schema Paths

Telemetry subscriptions include a set of schema paths used to identify which data nodes are of interest to the collector.

The paths in Telemetry Subscribe RPC requests follow the conventions described in the *OpenConfig gnmi-path-conventions.md* published on github.com (version 0.4.0, published June 21, 2017).

A path consists of a set of path segments often shown with a “/” character as a delimiter; for example, /configure/router[router-instance=Base]/interface[interface-name=my-interface1]/description.

These paths are encoded as a set of individual string segments in gnmi.proto (without any “/” characters); for example, ["configure", "router[router-name=Base]", "interface[interface-name=my-interface1]", "description"].

A path selects an entire subtree of the data model and includes all descendants of the node indicated in the path. Table 46 describes the types of schema paths that are supported in SR OS telemetry.

Table 46 **Schema Paths**

Path example	Description
/configure/router[router-name=Base]/ interface[interface-name=abc]	Selects all config leafs of interface abc and all descendants.
/configure/router[router-name=Base]/ interface[interface-name=abc]/description	Selects only the description leaf of interface abc.
/state/router[router-name=Base]/ interface[interface-name=*]	Selects all state information for all base router interfaces using a wildcard in a single segment of a path.
/configure/router[router-name=Base]/ interface[interface-name=*]/description	Selects all state information for all base router interfaces using a wildcard in a single segment of a path.
/	The root path. This selects all config and state data from all models (in all namespaces) supported on the router. Encoded as “” in gRPC/gPB.
/...	Expands all element levels in a given subtree schema, down to the leafs.
/*	Expands one level of a given subtree schema.

The following list describes the telemetry paths support in SR OS.

- The following wildcards are supported in the schema.
  - Specifying “/…” wildcard expands to multiple element levels in a path.
  - Specifying “/*” wildcard expands to only one level in a path.
- Wildcards for entire path segments are supported as follows:
  - For example: “/state/card/.../oper-state” expands to following paths
    /state/card[slot-number=*]/hardware-data/oper-state
    /state/card[slot-number=*]/mda[mda-slot=*]/hardware-data/oper-state
    /state/card[slot-number=*]/mda[mda-slot=*]/flex[group-index=*]/oper-state
  - For example: “/state/card/*/oper-state” expands to following path
    /state/card[slot-number=*]/hardware-data/oper-state

- If a wildcard is used for any key of a list, a wildcard must be used for all the keys of that list. In a single path segment, all keys must either have specific values or all keys must have wildcards. A mix of wildcards and specific values for different parts of a list key is not supported.

  For example:

  Supported:

  /a/b[key1=*][key2=*]/c[key1=foo]

  /a/b[key1=foo][key2=bar]/c[key1=*]

  Not supported:

  /a/b[key1=foo][key2=*]
- Functions such as “current()”, “last()” and mathematical operators, such as stat<5 or octets>3 are not supported in paths. The “|” (OR operator, used to select multiple paths) is not supported.
- Wildcards in multiple segments of a path are supported.

  For example: /state/card[slot-number=*]/mda[mda-slot=*]
- The paths with wildcards are expanded when a subscription is activated; this applies to dynamic and persistent subscriptions. In some cases, it is possible that a single path with wild cards can be expanded across both Nokia and Openconfig YANG models. However, this occurs only if both model types are enabled. If only one type is enabled, the path is expanded only within the enabled model. If the other type is enabled later, it is necessary to reset all subscriptions, which ensures that the expansion includes the newly enabled model type.

9.2.2 gNMI Service Use Cases

The gNMI Service can be used for the following:

- Telemetry
- NE Configuration Management

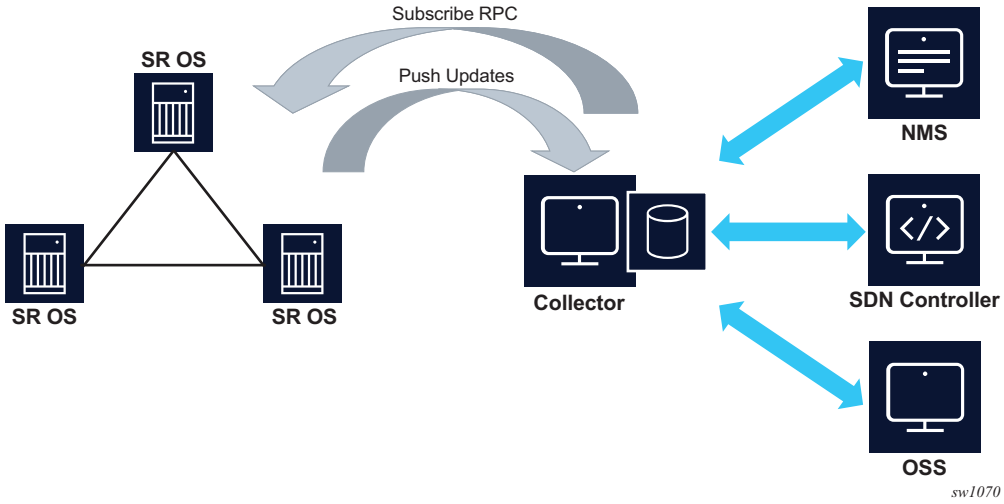
9.2.2.1 Telemetry

Telemetry is a network monitoring and fault management framework. Telemetry is driven by the need to use fresh data obtained from the network to make fast networking decisions such as traffic optimization and preventative troubleshooting.

9.2.2.1.1 Dial-in Telemetry

When the data collector initiates the gRPC connection, the SR OS node assumes the role of the gRPC server and the collector is the client. This is referred to as dial-in telemetry, where the SR OS node pushes data to the receiver (collector). Figure 23 shows the telemetry session initiated from the collector to the SR OS node via the Subscribe RPC.

Figure 23 **Dial-in Telemetry Session**



Dynamic Subscriptions

Dynamic subscriptions are created by the collector using the Subscribe RPC. These subscriptions are removed as soon as the gRPC session terminates. Dynamic subscriptions are currently supported only in dial-in mode.

Dial-in Telemetry Examples

This section contains examples of telemetry subscription requests and responses. The following examples are dumps of protobuf messages from a Python API. Formats may vary across different implementations.

Example 1 — Subscribe to a single path

```
2017-06-05 17:06:13,189 - SENT::SubscribeRequest
subscribe {
  subscription {
    path {
      element: "state"
```

```
      element: "router[router-instance=Base]"
      element: "interface[interface-name=test]"
      element: "statistics"
      element: "ip"
      element: "in-packets"
    }
    mode: SAMPLE
    sample_interval: 10000000000
  }
}

2017-06-05 17:06:13,190 - RCVD::SubsribeResponse
2017-06-05 17:06:23,492 - RCVD::Subscribe
2017-06-05 17:06:23,492 - update {
  timestamp: 1496675183491595139
  prefix {
    element: "state"
    element: "router[router-instance=Base]"
    element: "interface[interface-name=test]"
    element: "statistics"
    element: "ip"
  }
  update {
    path {
      element: "in-packets"
    }
    val {
      json_val: ""0""
    }
  }
}
2017-06-05 17:06:23,494 - RCVD::Subscribe
2017-06-05 17:06:23,494 - sync_response: true

2017-06-05 17:06:33,589 - RCVD::Subscribe
2017-06-05 17:06:33,589 - update {
  timestamp: 1496675213491595139
  prefix {
    element: "state"
    element: "router[router-instance=Base]"
    element: "interface[interface-name=test]"
    element: "statistics"
    element: "ip"
  }
  update {
    path {
      element: "in-packets"
    }
    val {
      json_val: ""28""
    }
  }
}
....
....
```

Example 2 — Subscribe to a single path with wildcard

```
2017-06-05 17:08:29,055 - SENT::SubscribeRequest
subscribe {
  subscription {
    path {
      element: "state"
      element: "router[router-instance=Base]"
      element: "interface[interface-name=*]"
      element: "statistics"
      element: "ip"
      element: "in-packets"
    }
    mode: SAMPLE
    sample_interval: 30000000000
  }
}

2017-06-05 17:08:29,056 - RCVD::SubsribeResponse
2017-06-05 17:08:59,133 - RCVD::Subscribe
2017-06-05 17:08:59,133 - update {
  timestamp: 1496675339132056575
  prefix {
    element: "state"
    element: "router[router-instance=Base]"
    element: "interface[interface-name=system]"
    element: "statistics"
    element: "ip"
  }
  update {
    path {
      element: "in-packets"
    }
    val {
      json_val: ""0""
    }
  }
}
2017-06-05 17:08:59,135 - RCVD::Subscribe
2017-06-05 17:08:59,135 - update {
  timestamp: 1496675339133006678
  prefix {
    element: "state"
    element: "router[router-instance=Base]"
    element: "interface[interface-name=to_node_B]"
    element: "statistics"
    element: "ip"
  }
  update {
    path {
      element: "in-packets"
    }
    val {
      json_val: ""0""
    }
  }
}
2017-06-05 17:08:59,135 - RCVD::Subscribe
2017-06-05 17:08:59,135 - update {
  timestamp: 1496675339133006678
  prefix {
```

```
    element: "state"
    element: "router[router-instance=Base]"
    element: "interface[interface-name=to_node_D]"
    element: "statistics"
    element: "ip"
  }
  update {
    path {
      element: "in-packets"
    }
    val {
      json_val: ""0""
    }
  }
}
2017-06-05 17:08:59,136 - RCVD::Subscribe
2017-06-05 17:08:59,136 - sync_response: true


2017-06-0517:09:29,139 - RCVD::Subscribe
2017-06-0517:09:29,139 - update {
  timestamp: 1496682569121314
  prefix {
    element: "state"
    element: "router[router-instance=Base]"
    element: "interface[interface-name=system]"
    element: "statistics"
    element: "ip"
  }
  update {
    path {
      element: "in-packets"
    }
    val {
      json_val: ""0""
    }
  }
}
2017-06-05 17:09:29,142 - RCVD::Subscribe
2017-06-05 17:09:29,142 - update {
  timestamp: 1496682569124342
  prefix {
    element: "state"
    element: "router[router-instance=Base]"
    element: "interface[interface-name=to_node_B]"
    element: "statistics"
    element: "ip"
  }
  update {
    path {
      element: "in-packets"
    }
    val {
      json_val: ""0""
    }
  }
}
2017-06-05 17:09:29,145 - RCVD::Subscribe
2017-06-05 17:09:29,145 - update {
  timestamp: 1496682569127344
```

```
  prefix {
    element: "state"
    element: "router[router-instance=Base]"
    element: "interface[interface-name=to_node_D]"
    element: "statistics"
    element: "ip"
  }
  update {
    path {
      element: "in-packets"
    }
    val {
      json_val: ""0""
    }
  }
}
....
....
```

Example 3: Subscribe to more than one path

```
2017-01-24 12:54:18,228 - SENT::SubscribeRequest
subscribe {
  subscription {
    path {
      element: "state"
      element: "router[router-instance=Base]"
      element: "interface[interface-name=to_node_B]"
    }
    mode: SAMPLE
    sample_interval: 30000000000
  }
  subscription {
    path {
      element: "state"
      element: "router[router-instance=Base]"
      element: "mpls"
      element: "statistics"
      element: "lsp-egress-stats[lsp-name=lsp_to_dest_f]"
    }
    mode: SAMPLE
    sample_interval: 30000000000
  }
}
```

Example 4: Subscribe to a list with wildcard

```
2017-01-24 13:45:30,947 - SENT::SubscribeRequest
subscribe {
  subscription {
    path {
      element: "state"
      element: "router[router-instance=Base]"
      element: "interface[interface-name=*]"
    }
    mode: SAMPLE
    sample_interval: 30000000000
```

```
  }
}
```

Example 5: Subscribe to path where the object did not exist before subscription

```
2017-01-24 13:53:50,165 - SENT::SubscribeRequest
subscribe {
  subscription {
    path {
      element: "state"
      element: "router[router-instance=Base]"
      element: "interface[interface-name=to_node_B]"
    }
    mode: SAMPLE
    sample_interval: 30000000000
  }
}


2017-01-24 13:53:50,166 - RCVD::SubsribeResponse
2017-01-24 13:54:20,169 - RCVD::Subscribe
2017-01-24 13:54:20,169 - sync_response: true


2017-01-24 13:54:50,174 - RCVD::Subscribe
2017-01-24 13:54:50,174 - update {
  timestamp: 1485262490169309451
  prefix {
    element: "state"
    element: "router[router-instance=Base]"
    element: "interface[interface-name=to_node_B]"
  }
  update {
...
...
  }
}
```

Example 6: Subscribe to a path where the object existed before subscription and then deleted after subscription

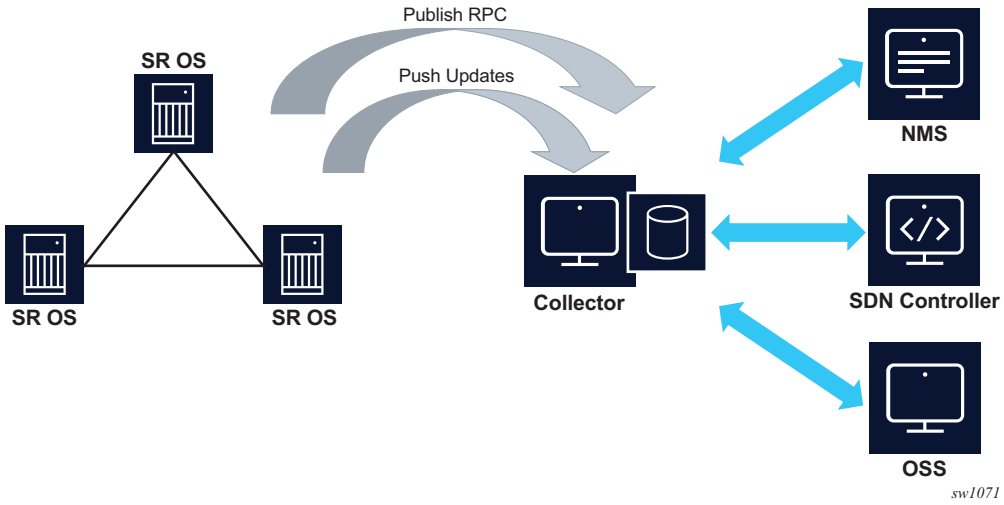
```
2017-01-24 14:00:41,292 - SENT::SubscribeRequest
subscribe {
  subscription {
    path {
      element: "state"
      element: "router[router-instance=Base]"
      element: "interface[interface-name=to_node_B]"
    }
    mode: SAMPLE
    sample_interval: 30000000000
  }
}


2017-01-24 14:00:41,294 - RCVD::SubsribeResponse
2017-01-24 14:01:11,295 - RCVD::Subscribe
2017-01-24 14:01:11,295 - update {
  timestamp: 1485262871290064704
```

```
  prefix {
    element: "state"
    element: "router[router-instance=Base]"
    element: "interface[interface-name=to_node_B]"
  }
  update {
...
...
  }
}
2017-01-24 14:01:11,359 - RCVD::Subscribe
2017-01-24 14:01:11,359 - sync_response: true

2017-01-24 14:01:41,293 - RCVD::Subscribe

2017-01-24 14:02:11,296 - RCVD::Subscribe
```

9.2.2.1.2 Dial-out Telemetry

When the SR OS node initiates the gRPC connection, the SR OS node assumes the role of the gRPC client. This is referred to as dial-out telemetry. Figure 24 shows the telemetry session initiated from the SR OS node to the collector via a Publish RPC.

Figure 24 **Dial-out Telemetry Session**



Persistent Subscriptions

Persistent subscriptions are configured on the SR OS node and they are not cleared when the gRPC session terminates. Persistent subscriptions are supported only in dial-out mode.

A persistent subscription associates one or more paths with corresponding destinations via sensor groups.

Every subscription has an associated administrative state as well as an operational state. If a connection is lost, the operational state goes down. If the collector does not receive the data but the SR OS appears to have a connection and the subscription is up, the connection can be reset by setting the administrative state down and then back up.

Destinations are defined in the form of destination groups. A destination group supports up to two destinations, where the destinations are served in a round-robin fashion. SR OS attempts to connect the first destination, and if successful, the telemetry data is sent to that destination. If the connection to the first destination fails (initially or during operation), SR OS attempts to connect or reconnect to the second destination, if it is configured. All configured destinations and local addresses should be reachable in the specified routing instances.

When the SR OS node initiates the gRPC connection via the Publish RPC, it includes the subscription name and the configured system name in the metadata. The collector can use this information to associate individual notification messages with the node and subscription.

Modifying any parameter of the active subscription causes the SR OS node to close the gRPC connection before attempting a reconnection.

When a gRPC connection is lost, the SR OS node continually attempts to establish a new session with the collector.

QoS Marking

The QoS marking of the IP packets carrying notifications can be configured under persistent subscription. IP packets to a specified destination are marked according to the configuration of the first subscription opened to the destination. This DSCP marking is maintained, regardless of any configuration changes, as long as the dial-out connection to the specified destination is open. If the destination is disconnected for any reason, the DSCP marking must be redefined when the connection is reestablished.

Configuring Dial-out Telemetry

The dial-out telemetry configuration process includes the following elements:

- **sensor group**

The sensor group specifies one or more schema paths from which data is streamed to the collector.

- **destination group**

  The destination group specifies the destination addresses (and ports) that the router uses to send the telemetry data.

- **persistent subscription**

  Persistent subscription associates a sensor group with a destination group and specifies streaming parameters for the telemetry data. For example, the subscription mode can be specified (ON_CHANGE, SAMPLE, or TARGET_DEFINED) for the subscription.

Dial-out telemetry can be configured via the MD-CLI or the classic CLI. For more information about using the MD-CLI, refer to the *7450 ESS, 7750 SR, and 7950 XRS MD-CLI User Guide*. For more information about the MD-CLI configuration commands, refer to the *7450 ESS, 7750 SR, and 7950 XRS MD-CLI Command Reference Guide*.

For more information about the classic CLI configuration commands, refer to the *7450 ESS, 7750 SR, 7950 XRS, and VSR Classic CLI Command Reference Guide* and the *7450 ESS, 7750 SR, 7950 XRS, and VSR Clear, Show, and Tools Command Reference Guide*.

The following example shows a sample dial-out telemetry configuration in the MD-CLI.

```
[ex:configure system telemetry]
A:admin@node-2# info
    destination-group "quick_cfg_dg_1" {
        description "Destination Group 1"
        allow-unsecure-connection
        destination 192.168.65.5 port 40001 {
            router-instance "Base"
        }
        destination 192.168.65.5 port 40002 {
            router-instance "Base"
        }
    }
    persistent-subscriptions {
        subscription "quick_cfg_sub_1" {
            admin-state enable
            description "Subscription 1"
            sensor-group "quick_cfg_sg"
            mode sample
            sample-interval 1234
            destination-group "quick_cfg_dg_1"
            local-source-address 1.2.3.4
            originated-qos-marking cp19
            encoding bytes
        }
    }
    sensor-groups {
```

```
        sensor-group "quick_cfg_sg" {
            description "Sensor Group"
            path "/state/router[router-name=Base]/interface[interface-name=test]/
statistics/ip" { }
        }
    }
```

The following sample outputs show telemetry information using the **show** command.

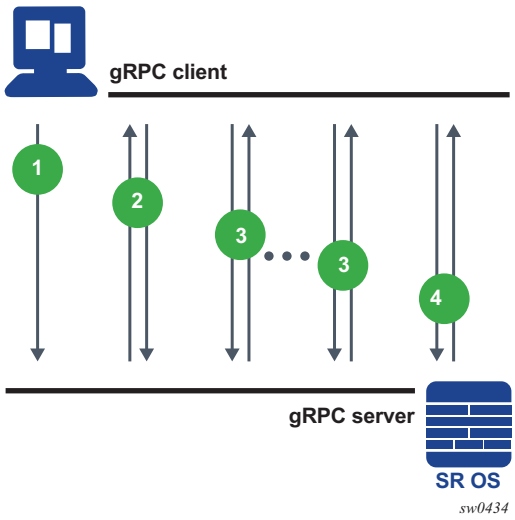
```
*A:node-2>show>system>telemetry>persistent# subscription "quick_cfg_sub_2"
===============================================================================
Telemetry persistent subscription
===============================================================================
Subscription Name     : quick_cfg_sub_2
Administrative State  : Enabled
Operational State     : Up
Subscription Id       : 198
Description           :
Sensor Group          : quick_cfg_sg_2
Destination Group     : quick_cfg_dg_2
Path Mode             : sample
Sample Interval       : 1000 ms
Encoding              : bytes
===============================================================================
*A:node-2>show>system>telemetry>persistent# subscription "quick_cfg_sub_2" paths
===============================================================================
Telemetry persistent subscriptionrsistent# subscription "quick_cfg_sub_2" paths
===============================================================================
Subscription Name     : quick_cfg_sub_1nt# subscription "quick_cfg_sub_2" paths
Administrative State  : Enabled
Operational State     : Up
Subscription Id       : 198
Description           :
Sensor Group          : quick_cfg_sg_2
Destination Group     : quick_cfg_dg_2
Path Mode             : sample
Sample Interval       : 1000 ms
Encoding              : bytes
-------------------------------------------------------------------------------
Paths
-------------------------------------------------------------------------------
Path                  : /state
Finished Samples      : 178
Deferred Samples      : 402
Total Collection Time : 405223 ms
Min Collection Time   : 2021 ms
Avg Collection Time   : 2276 ms
Max Collection Time   : 2956 ms
-------------------------------------------------------------------------------
No. of paths          : 1
===============================================================================
*A:node-
2>show>system>telemetry>persistent# subscription "quick_cfg_sub_2" destinations
===============================================================================
Telemetry persistent subscription
===============================================================================
Subscription Name     : quick_cfg_sub_2
Administrative State  : Enabled
```

```
Operational State     : Up
Subscription Id       : 198
Description           :
Sensor Group          : quick_cfg_sg_2
Destination Group     : quick_cfg_dg_2
Path Mode             : sample
Sample Interval       : 1000 ms
Encoding              : bytes
-------------------------------------------------------------------------------
Destinations
-------------------------------------------------------------------------------
Destination           : 192.168.65.1
Port                  : 40001
Operational State     : Down
Last Oper Down Reason : MINOR: TELEMETRY #2353: RPC refused by peer
Last Oper Change      : 2020/04/06 21:19:29
Connection Attempts   : 22
Notification Count    : 0
Total Notification Co*: 2315653
-------------------------------------------------------------------------------
Destination           : 192.168.65.1
Port                  : 40002
Operational State     : Up
Last Oper Down Reason : MINOR: TELEMETRY #2356: Canceled by config change
Oper Router Instance  : management
Last Oper Change      : 2020/04/06 21:19:30
Connection Attempts   : 22
Notification Count    : 3783151
Total Notification Co*: 5034573
-------------------------------------------------------------------------------
No. of destinations   : 2
* indicates that the corresponding row element may have been truncated.
===============================================================================
```

9.2.2.2 NE Configuration Management

Figure 25 shows NE configuration and information retrieval using the gNMI service.

Figure 25 **NE Configuration and Information Retrieval using gNMI Service**



In the context of gNMI, every SET RPC appears as an single commit operation, regardless of the number of paths included in the message. Both, NOKIA and OC models are supported by gNMI SET/GET RPC.

An example of the SET RPC command (including the response message from the gRPC server) follows:

```
 gNMI_rpc - DEBUG - SENT::SetRequest
prefix {
}
update {
  path {
    elem {
      name: "configure"
    }
    elem {
      name: "system"
    }
  }
  val {
    json_val: {"location": "zurich"}
  }
}
gMI_rpc - DEBUG - RCVD::SetResponse
prefix {
}
response {
  path {
    elem {
```

```
      name: "configure"
    }
    elem {
      name: "system"
    }
  }
  op: UPDATE
}
```

An example of the GET RPC command (including the response message from the gRPC server) follows:

```
gNMI_rpc - INFO - SENT::GetRequest GET140550212650064
path {
  elem {
    name: "configure"
  }
  elem {
    name: "system"
  }
  elem {
    name: "location"
  }
}
type: CONFIG
2017-12-06 12:17:28,639 - gMI_rpc - INFO -
 RCVD::GetResponse GET140550212650064
notification {
  timestamp: 1512559048634751055
  update {
    path {
      elem {
        name: "configure"
      }
      elem {
        name: "system"
      }
      elem {
        name: "location"
      }
    }
    val {
      json_val: "zurich"
    }
  }
}
```

9.3 gNOI Services

The gRPC Network Operations Interface (gNOI) defines a set of gRPC-based micro-services for executing operational commands on network devices. This includes the gNOI CERT service, that provides certificate management. The individual RPCs and messages that perform the operations required for certificate management on the node are defined in the Git repository hosting service (GitHub).

9.3.1 Certificate Management for TLS Connections

This section describes the gNOI services certificates that SR OS supports for managing secure TLS connections.

The SR OS supports the following RPCs for managing certificates for secure TLS connections:

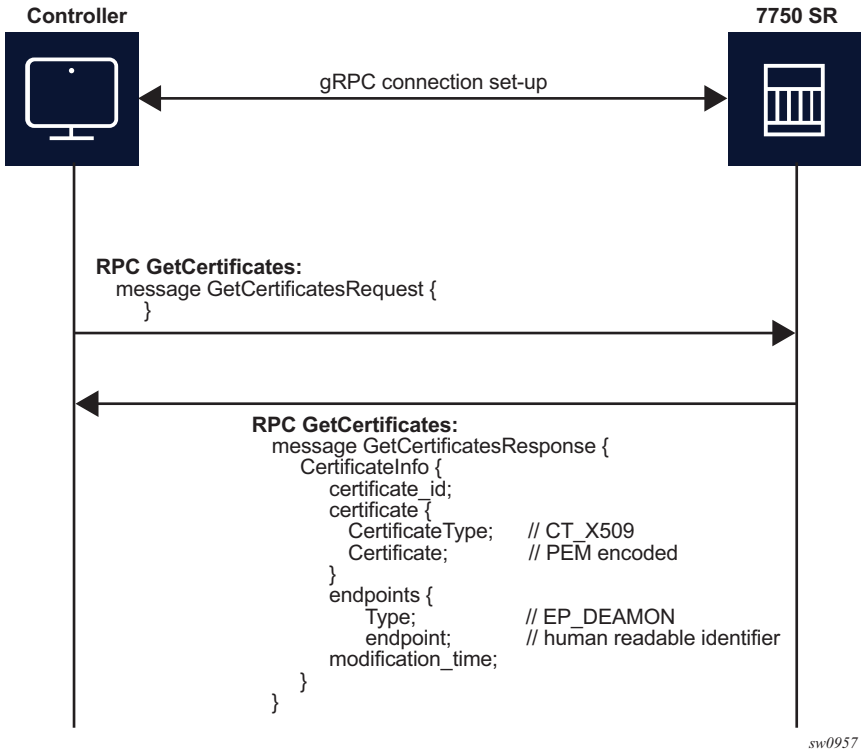
- RPC GetCertificates
- RPC CanGenerateCSR
- RPC Rotate
- RPC Install

Note: The gNOI RPCs, are by default, disabled in the user profile.

9.3.1.1 RPC GetCertificates

RPC GetCertificates provide information to the controller about all active certificates on the server (SR OS node). Figure 26 shows the message sequence.

Figure 26 **RPC GetCertificates Message Flow**



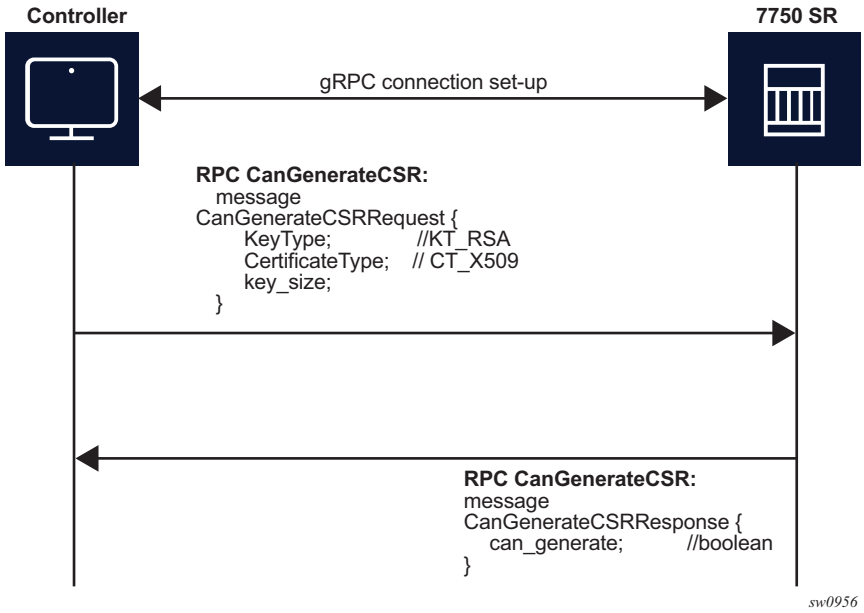
The RPC GetCertificates messages include a GetCertificateRequest and a GetCertificateRespone message. The GetCertificatesResponse message shown in Figure 26 includes the following information:

- certificate_id — The SR OS uses a certificate file name as the certificate ID.
- CertificateType — This is always set to X509, because it is the only type that the SR OS supports.
- endpoint — Indicates the CERT profiles in the SR OS node that use this certificate; if multiple CERT profiles use the certificate, the names are concatenated with the separation character “/”.

9.3.1.2 RPC CanGenerateCSR

The RPC CanGenerateCSR message can be used to determine if the gRPC server (SR OS node) can generate a Certificate Signing Request (CSR). It is a simple request and response operation as shown in Figure 27.

Figure 27 **RPC CanGenerateCSR Message Flow**



The SR OS only supports RSA keys and X509 certificates, so it only responds positively if those values are filled in the respective fields. The key size must be between 512 and 8192. In all other cases, the SR OS responds negatively to the CanGenerateCSRRequest message.

9.3.1.3 RPC Rotate

RPC Rotate allows the controller to rotate an active certificate on the server. After the rotation is completed, a new certificate can be used without affecting existing TLS connections.

The following cases are supported for a certificate rotation:

- server capable of generating a CSR (see Figure 28)
- server not capable of generating a CSR (see Figure 29)

The SR OS supports both scenarios, although it is assumed that in most cases the CSR is generated on SR OS node.

The following steps apply to both scenarios:

1. Generate the CSR

2. Sign the CSR by the Certificate Authority (CA)
3. Load the new certificate on the server
4. Verify the new certificate by creating a new connection
5. Finalize by confirming that the new certificate is being used

After the RPC Rotate is completed, all new connections use new keys.

Figure 28 **RPC Rotate Message Flow for CSRs Generated on the SR OS Node**

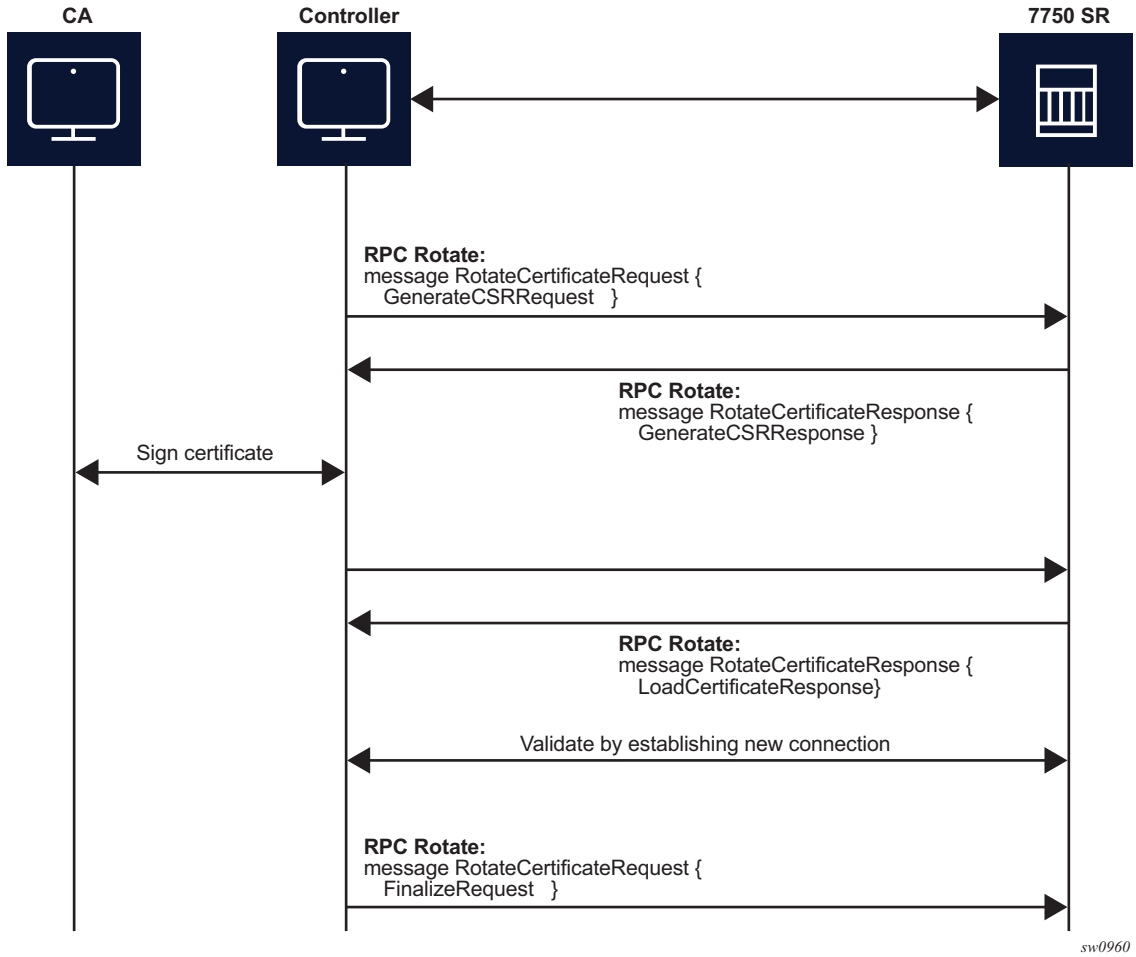
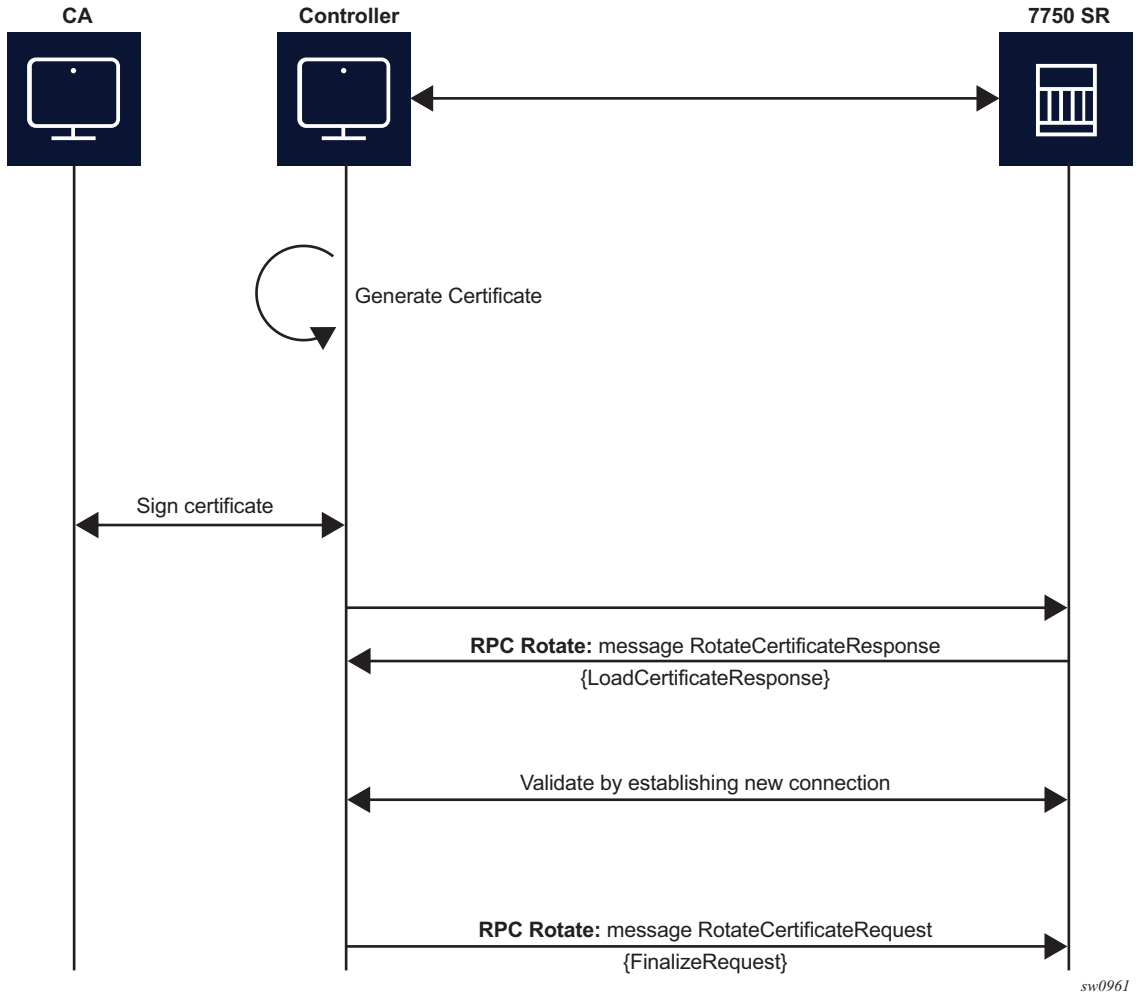


Figure 29 **RPC Rotate Message Flow When CSRs are Not Generated on the SR OS Node**

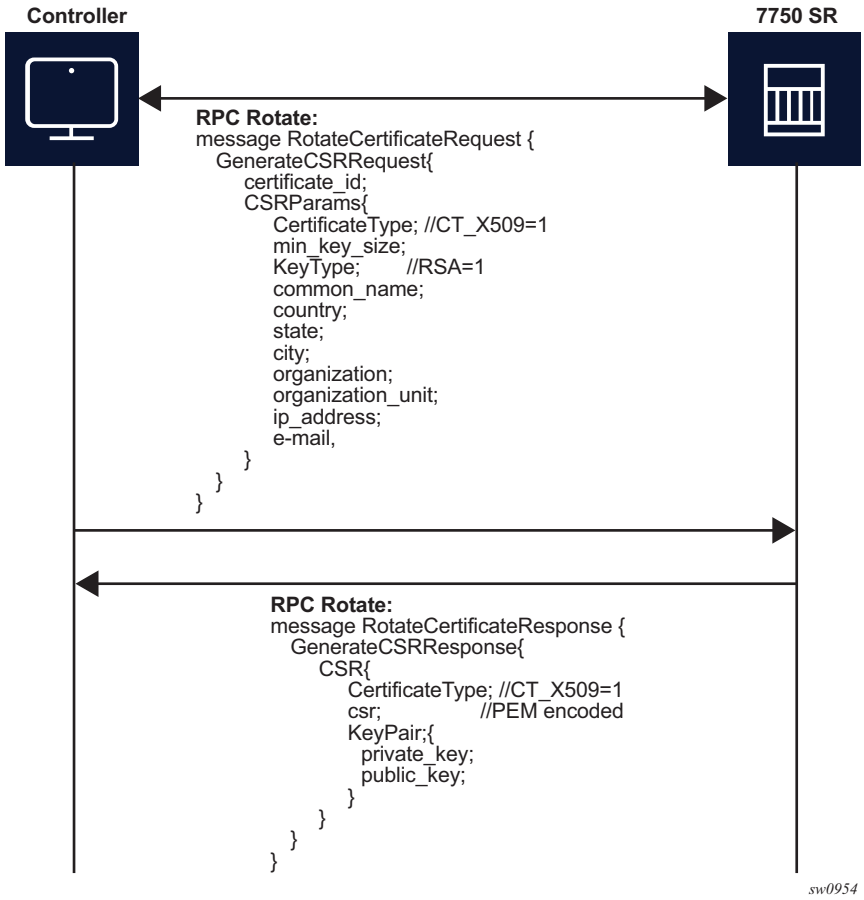


From the perspective of the interaction of the controller and the server (SR OS) two stages are the most important:

- message exchange to generate the CSR
- message exchange to load the new certificates on the server

Figure 30 shows a detailed content of the messages that are exchanged for CSR generation. The SR OS accepts requests only for the X509 certificate type, RSA key type, and a minimum key length of 512 bits.

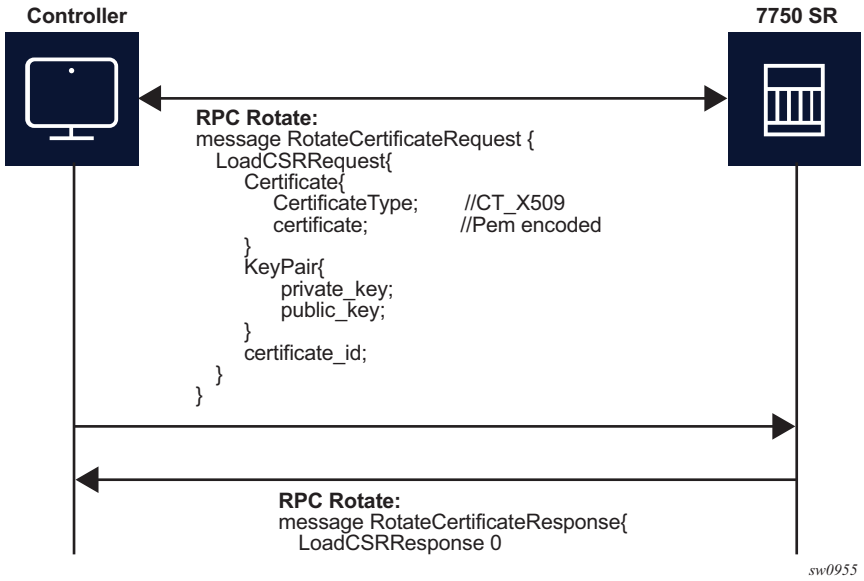
Figure 30 **GenerateCSR Message Flow**



For RPC Rotate, the certificate_id points to an existing certificate on the node. All the other parameters in the GenerateCSRRequest message are not checked by the SR OS software explicitly. They are used by the internal API to generate the CSR and that result is transparently passed to the controller.

After the CA signs the certificates, the files are loaded to the server using LoadCSRRequest and LoadCSRResponse message exchange, as shown in Figure 31. If this message exchange is used in the context of RPC Rotate, the certificate_id should not be present in LoadCSRRequest message. When the SR OS receives the message, it performs all the necessary steps to load this certificate, including storing the certificate and key files on the disk.

Figure 31 **LoadCSRRequest/Response Message Flow**



The controller is responsible for verifying the connection with the new certificate (Step 4 in Figure 28 and Figure 29); SR OS treats this as an optional step.

After the whole RPC is successfully closed, the system can use the new certificate to start new TLS connections.

9.3.1.4 RPC Install

The controller can use RPC Install to install a new certificate on the server. After the certificate is installed, the server must be configured (assign a certificate and key files in the CERT profile) before the new certificate can be used.

The following two possible cases are supported for installing a certificate:

- server capable of generating a CSR (see Figure 32)
- server is not capable of generating a CSR (see Figure 33)

The SR OS supports both scenarios, although it is assumed that in most cases the CSR is generated on the SR OS node.

Both scenarios require the following steps:

1. Generate the CSR

2. Sign the CSR by the Certificate Authority (CA)
3. Load the new certificate on the server

Figure 32 **RPC Install Message Flow for CSRs Generated on the SR OS Node**

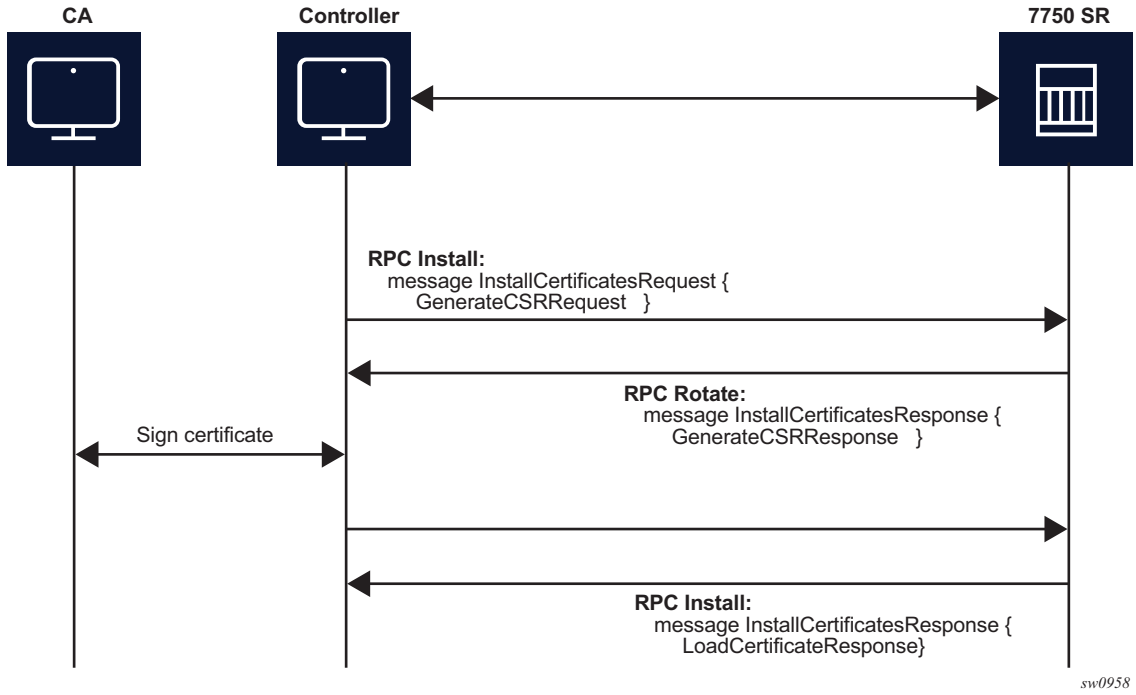
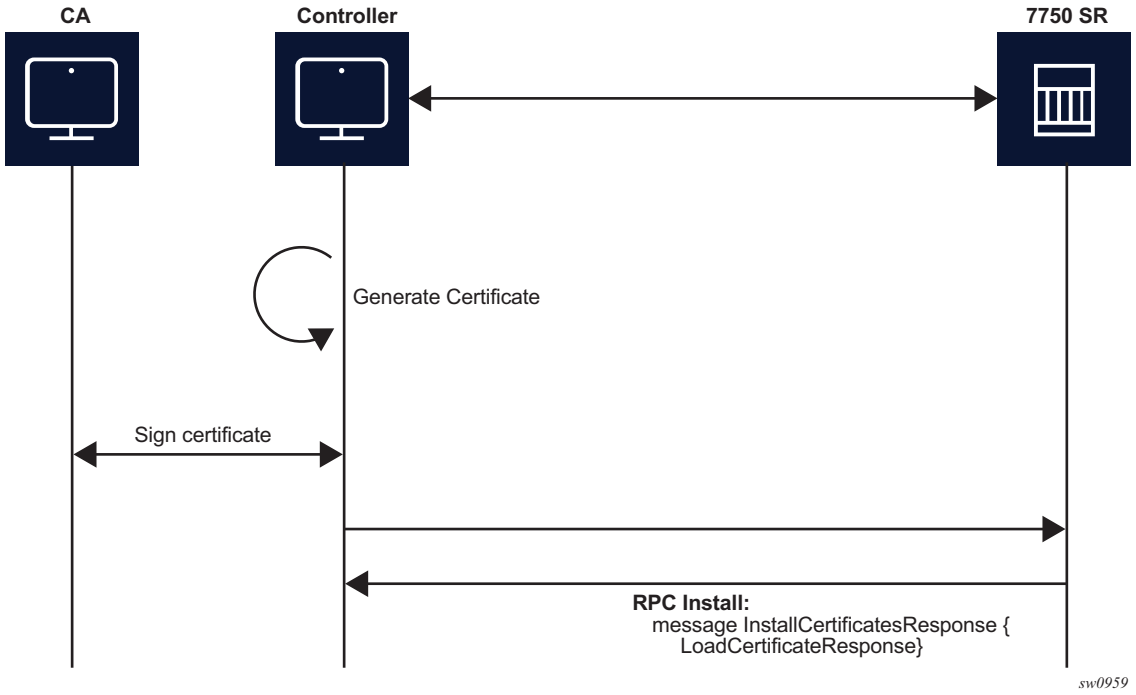


Figure 33 **RPC Install Message Flow if CSRs are Not Generated on the SR OS Node)**



The message exchange during phases 1 and 3 is the same as shown in Figure 30 and Figure 31. The only difference, in the case of RPC Install, is that, a new certificate_id is used.

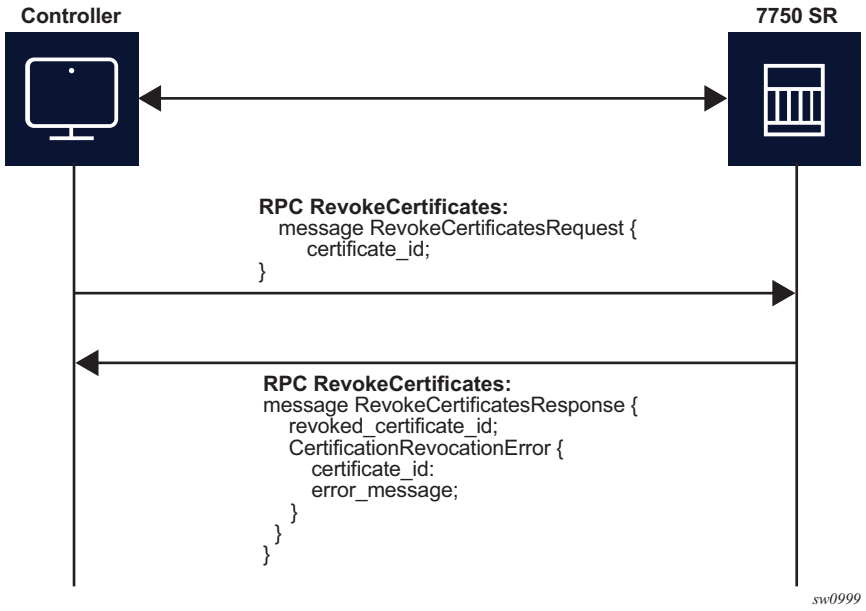
After new certificates are installed, the system must be configured before it can be used. Configuration is supported using the following methods:

- an existing gRPC session
- a CLI session, SNMP, and/or NETCONF

9.3.1.5 RPC RevokeCertificates

The purpose of the RPC RevokeCertificates is to render the existing certificate unusable by any client. In cases where the certificate being revoked by the client does not exist on the SR OS node, the corresponding RPC silently succeeds. The message flow is shown in Figure 34.

Figure 34 **RPC RevokeCertificates Message Flow**



9.4 gNOI System

In the gNOI System service, OpenConfig defines a generic interface to perform operational tasks on target network nodes. The specification can be found in following link: *https://github.com/openconfig/gnoi/blob/master/system/system.proto*.

These operations can be performed on individual targets, regardless of vendor. SR OS supports the following gNOI System RPCs:

- SetPackage RPC
- Reboot RPC
- CancelReboot RPC
- RebootStatus RPC
- SwitchControlProcessor RPC

9.4.1 SetPackage RPC

The SetPackage RPC allows the controller to place a software package on the target node. The file transfer is protected by the checksum. SR OS supports options where the controller can directly stream files to the target node. The remote download option is not supported.

9.4.2 Reboot, CancelReboot, and RebootStatus RPC

The Reboot RPC causes a target node to reboot. The controller can, in a RebootRequest message, specify the delay of the reboot and the actions that should be done with the system during reboot. SR OS supports only cold reboot, which means the whole node is shutdown and restarted.

The CancelReboot RPC allows the cancellation of any pending reboots.

The RebootStatus RPC allows the controller to query the status of a reboot for an individual component specified in the RebootStatus request. SR OS supports querying on a single component at the time.

9.4.3 SwitchControlProcessor RPC

The SwitchControlProcessor RPC allows the switching of the active Control-Processor to the Control-Processor that is provided in the request message. Since SR OS supports two Control Processors, one of the following paths should be given in the request message, depending which Control Processor is standby at that moment.

/state/cpm[cpm-slot=A]

/state/cpm[cpm-slot=B]

9.5 MD-CLI Service

The SR OS provides a proprietary management interface to use with the Network Interface Shell (NISH) tool which allows an MD-CLI style interface from a remote location in order to manage one or more SR OS nodes.

This feature is applicable only on SR OS platforms that support MD-CLI in Model-Driven mode.

This service operates using gRPC and therefore, the main gRPC service must also be enabled.

When enabled, the MD-CLI gRPC service provides MD-CLI schema information to the NISH client allowing users to remotely operate the SR OS device.

The MD-CLI gRPC service and the main gRPC service must be enabled on all nodes that will be managed using the nish client.

9.5.1 Remote Management Using a Remote Network Interface Shell Manager

When used together with the MD-CLI gRPC service, the remote management feature allows SR OS nodes to initiate communication with a remote NISH manager and announce to their availability to be managed using the NISH client.

This provides the NISH client with a dynamic view of the available nodes that it can manage. The remote management service does not perform or enable the actual management of the SR OS node using NISH. This communication is achieved directly from the NISH client to the SR OS node using the MD-CLI gRPC service.

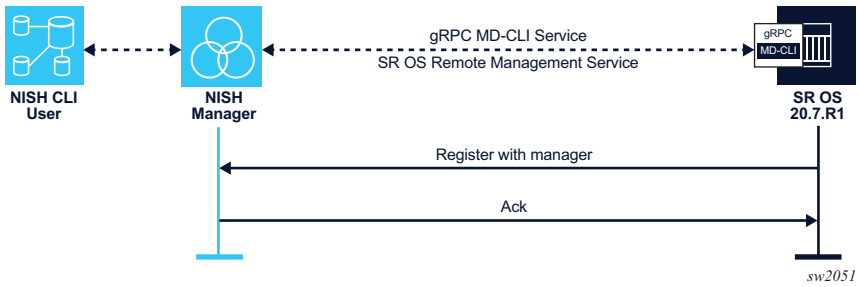
This feature is particularly useful when deploying clusters of SR OS nodes that may dynamically join or leave a cluster, such as in scenarios using the Control and User Plane Separation (CUPS) BNG application with Virtualized Service Routers (VSRs).

A working NISH manager service is required on an external server to use the remote management feature. The absence of a working NISH manager will not stop remote management from being enabled within SR OS, nor will it stop the SR OS node from announcing its presence to the configured IP address or addresses of the NISH manager.

When a remote NISH manager is configured, the SR OS node initiates a gRPC session with the configured manager. The SR OS node sends a message to communicate its name, IP address (IPv4 and IPv6 are supported) and gRPC port to the NISH manager. The NISH manager responds with an acknowledgment message. The SR OS node periodically checks in with the NISH manager.

Figure 35 depicts the remote management initiation.

Figure 35 **Remote Management Service Initiation**



If the connection is interrupted, the SR OS node immediately attempts reconnection with the configured NISH managers.

10 TLS

10.1 TLS Overview

Transport Layer Security (TLS) is used for two primary purposes:

- authentication of an end device (client or server) using a digital signature (DS)

  TLS uses PKI for device authentication. DSs are used to authenticate the client or the server. The server typically sends a certificate with a DS to the client.

  In certain situations, the server can request a certificate from the client to authenticate it. The client has a certificate (called a Trust Anchor) from the certificate authority (CA) which is used to authenticate server certificate and its DS. After the client provides a digitally signed certificate to the server and both parties are authenticated, the encryption PDUs can then be transmitted.

  When SR OS is acting as a server and it requests a certificate from the client, the client must provide the certificate. If the client fails to provide a certificate for authentication, SR OS will terminate the TLS session. The server TLS settings can be configured to not request certificates, in which case the client is not obligated to send the server a certificate for authentication.
- encryption and authentication of application PDUs

  After the clients and server have been successfully authenticated, the cipher suite is negotiated between the server and clients, and the PDUs will be encrypted based on the agreed cipher protocol.

10.2 TLS Server Interaction with Applications

TLS is a standalone configuration. The user must configure TLS server profiles with certificates and trust anchors, and then assign the TLS server profiles to the appropriate applications. When a TLS server profile is assigned to an application, the application should not send any clear text PDUs until the TLS handshake has been successfully completed and the encryption ciphers have been negotiated between the TLS server and the TLS client.

After successful negotiation and handshake, the TLS will be operationally up, and the TLS will notify the application which will begin transmitting PDUs. These PDUs will be encrypted using TLS based on the agreed ciphers. If, at any point, the TLS becomes operationally down, the application should stop transmitting PDUs.

For example, a TLS connection with the gMI application would operate as follows:

1. A TLS server profile is assigned to the gMI application.
2. gMI stops sending clear text PDUs because a TLS server profile has been assigned and TLS is not ready to encrypt.
3. The TLS server begins the handshake.
4. Authentication occurs at the TLS layer.
5. The TLS server and TLS client negotiate ciphers.
6. SALTs are negotiated for the symmetric key. A SALT is a seed for creating AES encryption keys.
7. When negotiations are successfully completed, the handshake finishes and gMI is notified.
8. TLS becomes operationally up, and gMI can resume transmitting PDUs. Until TLS becomes operationally up, gMI PDUs arriving from the client are dropped on ingress.

10.2.1 TLS Application Support

Table 47 lists the applications that support TLS.

Table 47 **TLS Application Support**

Application	TLS Server Supported	TLS Client Supported
LDAP		✓
gRPC	✓	

10.3 TLS Handshake

Figure 36 shows the TLS handshake.

Figure 36 **TLS Handshake**

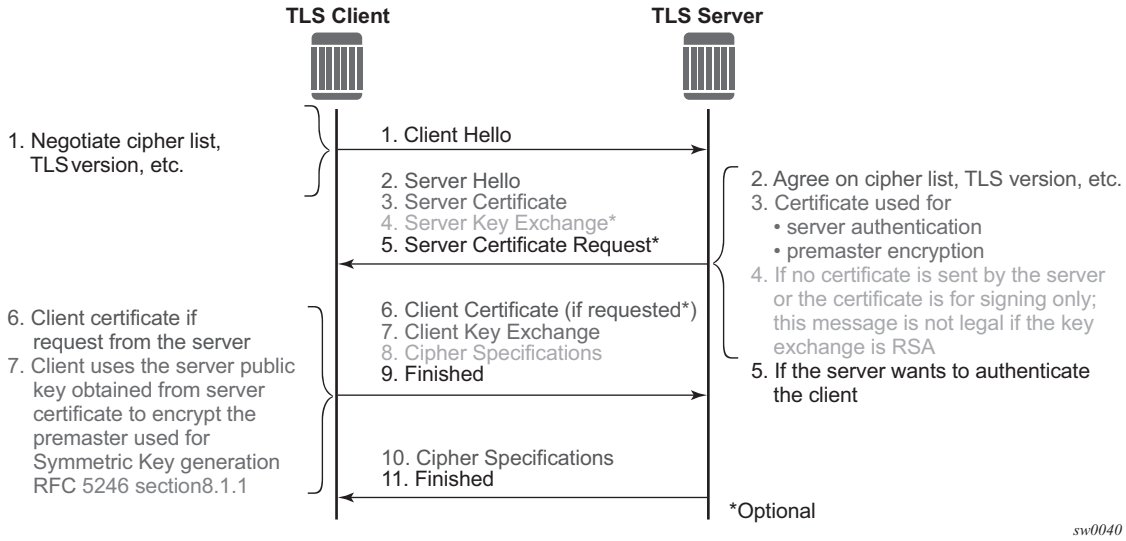


Table 48 describes the steps in the TLS handshake.

Table 48 **TLS Handshake Step Descriptions**

Step	Description
1	The TLS handshake begins with the client Hello message. This message includes the cipher list that the client wishes to use and negotiate, among other information.
2	The TLS server sends back a server Hello message, along with the first common cipher found on both the client cipher list and the server cipher list. This agreed cipher will be used for data encryption.
3	The TLS server continues by sending a server certificate message, where the server provides a certificate to the client so that the client can authenticate the server identity. The public key of this certificate (RSA key) can also be used for encryption of the symmetric key seed that will be used by the client and server to create the symmetric encryption key. This occurs only if the PKI is using RSA for asymmetric encryption.
4	Server key exchange is not supported by SR OS. SR OS only uses RSA keys; Diffie-Hellman key exchange is not supported.
5	The server can optionally be configured to request a certificate from the client to authenticate the client.
6	If the server has requested a certificate, the client should provide a certificate using a client certificate message. If the client does not provide a certificate, the server will drop the TLS session.

Table 48 **TLS Handshake Step Descriptions (Continued)**

Step	Description
7	The client uses the server public RSA key that was included in the server certificate to encrypt a seed used for creating the symmetric key. This seed is used by the client and server to create the identical symmetric key for encrypting and decrypting the data plane traffic.
8	The client sends a cipher spec to switch encryption to this symmetric key.
9	The client successfully finishes the handshake.
10	The server sends a cipher spec to switch encryption to this symmetric key.
11	The server successfully finishes the handshake.

After a successful handshake, TLS will be operationally up, and applications can then use it for application encryption.

10.4 TLS Client Certificate

TLS protocol is used for authentication, and as such, the server can ask to authenticate the client via PKI. If the server requests authentication from the client, the client must provide an X.509v3 certificate to the server so that it can be authenticated via the digital signature of its client. SR OS allows the configuration of an X.509v3 certificate for TLS clients. When the server requests a certificate via the server's Hello message, the client will transmit its certificate to the server using a client certificate message.

10.5 TLS Symmetric Key Rollover

SR OS supports key rollover via HelloRequest messages as detailed in RFC 5246, section 7.4.1.1. Some applications have a longer live time than other applications, in which case SR OS can use a timer that prompts the HelloRequest negotiation for the symmetric key rollover. This timer is configurable using CLI.

If an application does not support the HelloRequest message, the **no tls-re-negotiate-timer** command should be configured under the **config**>**system**>**security**>**tls** context. For example, the gRPC application does not support HelloRequest messages.

When **no tls-re-negotiate-timer** is configured, the HelloRequest message is not generated, and symmetric keys are not renegotiated.

10.6 Supported TLS Ciphers

As shown in Figure 36, TLS negotiates the supported ciphers between the client and the server.

The client sends the supported cipher suites in the client Hello message and the server compares them with the server cipher list. The top protocol on both lists is chosen and returned from the server within the server Hello message.

The 7750 SR supports the following ciphers as a TLS client or TLS server:

- tls-rsa-with-null-md5
- tls-rsa-with-null-sha
- tls-rsa-with-null-sha256
- tls-rsa-with3des-ede-cbc-sha
- tls-rsa-with-aes128-cbc-sha
- tls-rsa-with-aes256-cbc-sha
- tls-rsa-with-aes128-cbc-sha256
- tls-rsa-with-aes256-cbc-sha256

10.7 SR OS Certificate Management

SR OS implements a centralized certificate management protocol that can be used by TLS and IPsec. Refer to the *7450 ESS, 7750 SR, and VSR Multiservice Integrated Service Adapter and Extended Services Appliance Guide* for information about the configuration of the certificates and the corresponding protocols, such as OCSP, CMPv2, and CRL.

The main certificate configurations are:

- certificate configuration and management, configured using the **admin**>**certificate** commands
- PKI configuration (including creating a CA profile), configured using the **config**>**system**>**security**>**pki** commands

The two main configuration sub-trees for certificates are displayed below.

CLI Syntax:

```
admin>certificate
   clear-ocsp-cache
   cmpv2
   crl-update
   display
   export
   gen-keypair
   gen-local-cert-req
   import
   reload

config>system>security>pki
   [no] ca-profile
   certificate-display-format
   [no] certificate-expiration-warning
   [no] crl-expiration-warning
   [no] maximum-cert-chain-depth
```

10.7.1 Certificate Profile

The certificate profile is available for both the TLS server and the TLS client. The **cert-profile** command is configured for the server or client to transmit the provider certificate and its DS to the peer so that the peer can authenticate it via the **trust-anchor** and CA certificate.

Multiple provider certificates can be configured on SR OS; however, SR OS currently uses the smallest index as the active provider certificate, and will only send the certificate to the peer.

10.7.2 TLS Server Authentication of the Client Certificate CN Field

If the client provides a certificate upon request by the server, SR OS checks the certificate's common name (CN) field against local CN configurations. The CN is validated via the client IPv4/IPv6 address or FQDN.

If **cn-authentication** is not enabled, SR OS will not authenticate via the CN field and will only rely on certificate signature authentication.

10.7.3 CN Regexp Format

CN entries are configured by using the **config>system>security>pki>common-name-list** command. Entries should use regular expression (regexp), FQDN, or the IP address.

For information about regexp, refer to the CLI Usage section in *7450 ESS, 7750 SR, 7950 XRS, and VSR Basic System Configuration Guide*.

10.8 Operational Guidelines

10.8.1 Server Authentication Behavior

Following the Hello messages, the server sends its certificate in a certificate message if it is to be authenticated. If required, a ServerKeyExchange message may also be sent. Refer to RFC 5246, section 7.3, for more information about the authentication behavior on the LDAP server.

The **trust-anchor-profile** command determines whether or not the server must be authenticated by the client.

CLI Syntax:
```
config>system>security>tls
   client-tls-profile ldap create
      [no] trust-anchor-profile
```

Note: If the **trust-anchor-profile** is configured and the **ca-certificate** or **ca-profile** is missing from this **trust-anchor-profile**, the TLS connection will fail and an “unknown_ca” error will be generated, as per RFC 5246 section 7.2.2.

One of the following two configurations can be used to establish server connectivity.

a. If **trust-anchor-profile** is configured under the TLS **client-tls-profile** context, the server must be authenticated via the **trust-anchor-profile** command before a trusted connection is established between the server and the client.
b. If there is no **trust-anchor-profile** under the **client-tls-profile** context, the trusted connection can be established without server authentication. The RSA key of the certificate will be used for public key encryption, requiring basic certificate checks to validate the certificate. These basic checks are:

- time validity—the certificate is checked to ensure that it is neither expired nor not yet valid
- certificate type—the certificate is not a CA certificate
- keyUsage extension—if present, this must contain a digital signature and key encryption
- host verification—the IP address or DNS name of the server is looked up, if available (for LDAP, only the IP address is used), in the common name (cn) or subjectAltName extension. This is to verify that the certificate was issued to that server and not to another.

10.8.2 Client TLS Profile and Trust Anchor Behavior and Scale

SR OS allows the creation of client TLS profiles, which can be assigned to applications such as LDAP to encrypt the application layer.

The **client-tls-profiles** command is used for negotiating and authenticating the server. After the server is authenticated via the trust anchor profile (configured using the **trust-anchor-profile** command) of a client TLS profile, it negotiates the ciphers and authentication algorithms to be used for encryption of the data.

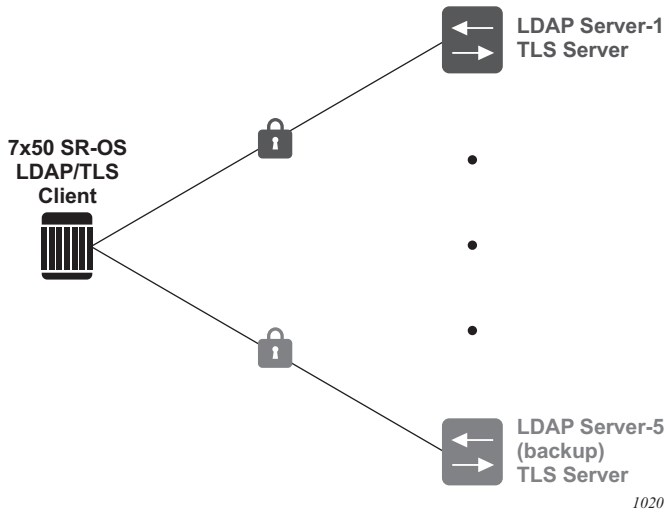
The client TLS profile must be assigned to an application for it to start encrypting. Up to 16 client TLS profiles can be configured. Because each of these client TLS profiles needs a trust anchor profile to authenticate the server, up to 16 trust anchor profiles can be configured. A trust anchor profile holds up to 8 trust anchors (configured using the **trust-anchor** command), which each hold a CA profile (**ca-profile**).

A CA profile is a container for installing CA certificates (**ca-certificates**). These CA certificates are used to authenticate the server certificate. When the client receives the server certificate, it reads through the trust anchor profile CA certificates and tries to authenticate the server certificate against each CA certificate. The first CA certificate that authenticates the server is used.

10.9 LDAP Redundancy and TLS

LDAP supports up to five redundant (backup) servers, as shown in Figure 37 and the configuration examples below. Depending on the **timeout** and **retry** configurations, if an LDAP server is determined to be out of service or operationally down, SR OS will switch to the redundant servers. SR OS will select the LDAP server with the next largest configured server index.

Figure 37 **LDAP and TLS Redundancy**



Configuration of Server-1:

```
A*:SwSim14>config>system>security>ldap# info
    public-key-authentication
    server 1 create
        address 1.1.1.1
        ldap-server "active-server"
        tls-profile "server-1-profile"

A*:SwSim14>config>system>security>tls# info
    client-tls-profile "server-1-profile" create
        cipher-list "to-active-server"
        trust-anchor-profile "server-1-ca"
        no shutdown
    exit
```

Configuration of Server-5 (backup):

```
A*:SwSim14>config>system>security>ldap# info
    public-key-authentication
    server 5 create
        address 5.5.5.1
```

```
        ldap-server "backup-server-5"
        tls-profile "server-5-profile"

A*:SwSim14>config>system>security>tls# info
    client-tls-profile "server-5-profile" create
        cipher-list "to-backup-server-5"
        trust-anchor-profile "server-5-ca"
        no shutdown
    exit
```

Each LDAP server can have its own TLS profile, each of which can have its own configuration of **trust-anchor** and **cipher-list**. For security reasons, the LDAP servers may be in different geographical areas and, as such, each will be assigned its own server certificate and trust anchor. The design is open to allow the user to mix and match all components.

10.10 Basic TLS Configuration

Basic TLS server configuration must have the following:

- a cipher list created using the **config**>**system**>**security**>**tls**>**server-cipher-list** command, and assigned to the TLS server profile using the **config**>**system**>**security**>**tls**>**server-tls-profile**>**cipher-list** command
- a certificate profile created using the **config**>**system**>**security**>**tls**>**cert-profile** command, and assigned to the TLS server profile using the **config**>**system**>**security**>**tls**>**server-tls-profile**>**cert-profile** command

Basic TLS client configuration must have a cipher list created using the **config**>**system**>**security**>**tls**>**client-cipher-list** command, and assigned to the TLS client profile using the **config**>**system**>**security**>**tls**>**client-tls-profile**>**cipher-list** command.

TLS imports the trust anchor certificate for (TLS) peer certificate authentication and public key retrieval.

The following displays the CLI syntax for TLS:

CLI Syntax:

```
config>system>security>tls
   cert-profile profile-name [create]
   no cert-profile profile-name
   client-cipher-list name [create]
   no client-cipher-list name
   client-tls-profile name [create]
   no client-tls-profile name
   server-cipher-list name [create]
   no server-cipher-list name
   server-tls-profile name [create]
   no server-tls-profile name
   trust-anchor-profile name [create]
   no trust-anchor-profile name
```

The following displays a TLS configuration example.

```
config>system>security>tls# info
----------------------------------------------
        trust-anchor-profile "server-1-ca" create
            trust-anchor "tls-server-1-ca"
        exit
        client-cipher-list "to-active-server" create
            cipher 1 name tls-rsa-with-aes256-cbc-sha256
            cipher 2 name tls-rsa-with-aes128-cbc-sha256
            cipher 3 name tls-rsa-with-aes256-cbc-sha
        exit
        client-tls-profile "server-1-profile" create
```

```
            cipher-list "to-active-server"
            trust-anchor-profile "server-1-ca"
            no shutdown
        exit
-----------------------------------------
```

10.11 Common Configuration Tasks

10.11.1 Configuring a Server TLS Profile

The following displays the CLI syntax for a server TLS profile.

CLI Syntax:

```
config>system>security>tls
   server-tls-profile name [create]
   no server-tls-profile name
      authenticate-client
         trust-anchor-profile ca-profile-name
         no trust-anchor-profile
      cert-profile name
      no cert-profile
      cipher-list name
      no cipher-list
      [no] shutdown
      tls-re-negotiate-timer [0 to 65000]
      no tls-re-negotiate-timer
```

10.11.2 Configuring a Client TLS Profile

The following displays the CLI syntax for a client TLS profile, which also configures the server authentication behavior:

CLI Syntax:

```
config>system>security>tls
   client-tls-profile name [create]
   no client-tls-profile name
      trust-anchor-profile name
      no trust-anchor-profile
```

10.11.3 Configuring a TLS Client or TLS Server Certificate

The following displays the CLI syntax for TLS certificate management:

CLI Syntax:

```
config>system>security>tls
   cert-profile profile-name [create]
   no cert-profile profile-name
      entry entry-id [create]
      no entry entry-id
         cert cert-filename
         no cert
         key key-filename
         no key
         [no] send-chain
            [no] ca-profile name
      [no] shutdown
   client-tls-profile name [create]
   no client-tls-profile name
      cert-profile name
      no cert-profile
   server-tls-profile name [create]
   no server-tls-profile name
      cert-profile name
      no cert-profile
```

10.11.4 Configuring a TLS Trust Anchor

The following displays the CLI syntax for a TLS trust anchor:

CLI Syntax:

```
config>system>security>pki
   [no] ca-profile
   certificate-display-format
   [no] certificate-expiration-warning hours
   [no] crl-expiration-warning
   [no] maximum-cert-chain-depth

config>system>security>tls
   [no] trust-anchor-profile
   [no] client-tls-profile
      [no] cipher-list
      [no] shutdown
      [no] trust-anchor-profile-profile
```

The following displays a TLS trust anchor configuration example:

```
*B:SeGW-1>config>system>security>pki# info
----------------------------------------------
        ca-profile "tls-server-1-ca" create
            cert-file "tls-1-Root-CERT"
            crl-file "tls-1-CRL-CERT"
            no shutdown
        exit
----------------------------------------------
*A:SwSim8>config>system>security>tls# info
----------------------------------------------
        trust-anchor-profile "server-1-ca" create
            trust-anchor "tls-server-1-ca"
        exit
        client-tls-profile "server-1-profile" create
            cipher-list "to-active-server"
            trust-anchor-profile "server-1-ca"
            no shutdown
        exit
```

11 Facility Alarms

11.1 Facility Alarms Overview

Facility Alarms provide a useful tool for operators to easily track and display the basic status of their equipment facilities. Facility Alarm support is intended to cover a focused subset of router states that are likely to indicate service impacts (or imminent service impacts) related to the overall state of hardware assemblies (cards, fans, links, and so on).

In the CLI, for brevity, the keyword or command **alarm** is used for commands related to Facility Alarms. This chapter may occasionally use the term **alarm** as a short form for **facility alarm**.

The CLI display for `show` routines allows the system operator to easily identify current facility alarm conditions and recently cleared facility alarms without searching event logs or monitoring various card and port show commands to determine the health of basic equipment in the system such as cards and ports.

The SR OS alarm model is based on RFC 3877, *Alarm Management Information Base (MIB)*, (which evolved from the IETF Disman drafts).

11.2 Facility Alarms vs. Log Events

Facility Alarms are different than log events. Facility Alarms have a state (at least two states: active and clear) and a duration, and can be modeled with state transition events (raised, cleared). A log event occurs when the state of some object in the system changes. Log events notify the operator of a state change (for example, a port going down, an IGP peering session coming up, and so on). Facility alarms show the list of hardware objects that are currently in a bad state. Facility alarms can be examined at any time by an operator, whereas log events can be sent by a router asynchronously when they occur (for example, as an SNMP notification or trap, or a syslog event).

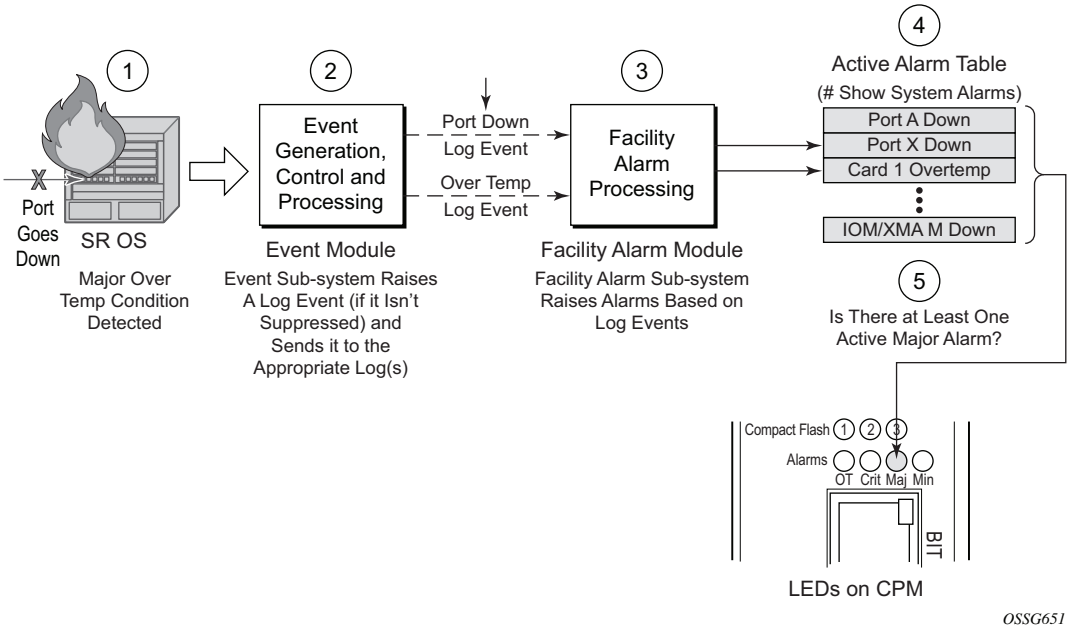
While log events provide notifications about a large number of different types of state changes in SR OS, facility alarms are intended to cover a focused subset of router states that are likely to indicate service impacts (or imminent service impacts) related to the overall state of hardware assemblies (cards, fans, links, and so on).

The facility alarm module processes log events in order to generate the raised and cleared state for the facility alarms. If a raising log event is suppressed under event-control, then the associated facility alarm will not be raised. If a clearing log event is suppressed under event-control, then it is still processed for the purpose of clearing the associated facility alarm. If a log event is a raising event for a Facility Alarm, and the associated Facility Alarm is raised, then changing the log event to **suppress** will clear the associated Facility Alarm.

Log event filtering, throttling and discarding of log events during overload do not affect facility alarm processing. In all cases, non-suppressed log events are processed by the facility alarm module before they are discarded.

Figure 38 illustrates the relationship of log events, facility alarms and the LEDs.

Figure 38 **Log Events, Facility Alarms and LEDs**



Facility Alarms are different and independent functionality from other uses of the term alarm in SR OS such as:

- Log events that use the term **alarm** (tmnxEqPortSonetAlarm)
- **configure card fp hi-bw-mcast-src** [**alarm**]
- **configure mcast-management multicast-info-policy bundle channel source-override video analyzer alarms**
- **configure port ethernet report-alarm**
- **configure system thresholds no memory-use-alarm**
- **configure system thresholds rmon no alarm**

- **configure system security cpu-protection policy alarm**

11.3 Facility Alarm Severities and Alarm LED Behavior

The Alarm LEDs on the CPM/CCM reflects the current status of the Facility Alarms:

- The Critical Alarm LED is lit if there is 1 or more active Critical Facility Alarms
- Similarly with the Major and Minor alarm LEDs
- The OT Alarm LED is not controlled by the Facility Alarm module

The supported alarm severities are as follows:

- Critical (with an associated LED on the CPM/CCM)
- Major (with an associated LED on the CPM/CCM)
- Minor (with an associated LED on the CPM/CCM)
- Warning (no LED)

Facility alarms inherit their severity from the raising log event.

A raising log event for a facility alarm configured with a severity of *indeterminate* or *cleared* will result in the facility alarm not being raised. But, a clearing log event is processed in order to clear facility alarms, regardless of the severity of the clearing log event.

Changing the severity of a raising log event only affects subsequent occurrences of that log event and facility alarms. Facility alarms that are already raised when their raising log event severity is changed maintain their original severity.

11.4 Facility Alarm Hierarchy

Facility Alarms for children objects is not raised for failure of a parent object. For example, when an MDA or XMA fails (or is shutdown) there is not a set of port facility alarms raised.

When a parent facility alarm is cleared, children facility alarms that are still in occurrence on the node appears in the active facility alarms list. For example, when a port fails there is a port facility alarm, but if the MDA or XMA is later shutdown the port alarm is cleared (and a card alarm will be active for the MDA or XMA). If the MDA or XMA comes back into service, and the port is still down, then a port alarm becomes active once again.

The supported facility alarm hierarchy is as follows (parent objects that are down cause alarms in all children to be masked):

- CPM -> Compact Flash
- CCM -> Compact Flash
- IOM/IMM -> MDA -> Port -> Channel
- XCM -> XMA -> Port

Note: A masked facility alarm is not the same as a cleared facility alarm. The cleared facility alarm queue does not display entries for previously raised facility alarms that are currently masked. If the masking event goes away, then the previously raised facility alarms will once again be visible in the active facility alarm queue.

11.5 Facility Alarm List

Table 49 and Table 50 show the supported Facility Alarms.

Table 49 **Facility Alarm, Facility Alarm Name, Raising Log Event, Sample Details String and Clearing Log Event**

Facility Alarm	Facility Alarm Name/Raising Log Event	Sample Details String	Clearing Log Event
59-2004-1	linkDown	Interface intf-towards-node-B22 is not operational	linkUp
64-2091-1	tmnxSysLicenseInvalid	Error - <reason> record. <hw> will reboot the chassis <timeRemaining>	tmnxSysLicenseValid
64-2092-1	tmnxSysLicenseExpiresSoon	The license installed on <hw> expires <timeRemaining>	tmnxSysLicenseValid
64-2221-1	tmnxSysStandbyLicensingError	CPM B is not licensed; license record not found	tmnxSysStandbyLicensingReady

Table 49 **Facility Alarm, Facility Alarm Name, Raising Log Event, Sample Details String and Clearing Log Event (Continued)**

Facility Alarm	Facility Alarm Name/Raising Log Event	Sample Details String	Clearing Log Event
64-2226-1	tmnxSysLicenseUpdateRequired	System license update is required	tmnxSysLicenseValid
93-2006-1	tmnxSatSyncIfTimHoldover	Synchronous timing interface on satellite esat-1 is in holdover state	tmnxSatSyncIfTimHoldoverClear
93-2008-1	tmnxSatSyncIfTimRef1Alarm with attribute tmnxSyncIfTimingNotifyAlarm == 'los(1)'	Synchronous timing interface on satellite, alarm on reference 1	tmnxSatSyncIfTimRef1AlarmClear
93-2008-2	tmnxSatSyncIfTimRef1Alarm with attribute tmnxSyncIfTimingNotifyAlarm == 'oof(2)'	Synchronous timing interface on satellite, alarm on reference 1	same as 93-2008-1
93-2008-3	tmnxSatSyncIfTimRef1Alarm with attribute tmnxSyncIfTimingNotifyAlarm == 'oopir(3)'	Synchronous timing interface on satellite, alarm on reference 1	same as 93-2008-1
93-2010-x	same as 93-2008-x but for ref2	same as 93-2008-x but for ref2	same as 93-2008-x but for ref2
7-2001-1	tmnxEqCardFailure	Class MDA Module: failed, reason: Mda 1 failed startup tests	tmnxChassisNotificationClear
7-2003-1	tmnxEqCardRemoved	Class CPM Module: removed	tmnxEqCardInserted
7-2004-1	tmnxEqWrongCard	Class IOM Module: wrong type inserted	tmnxChassisNotificationClear
7-2005-1	tmnxEnvTempTooHigh	Chassis 1: temperature too high	tmnxChassisNotificationClear
7-2011-1	tmnxEqPowerSupplyRemoved	Power supply 1, power lost	tmnxEqPowerSupplyInserted
7-2017-1	tmnxEqSyncIfTimingHoldover	Synchronous Timing interface in holdover state	tmnxEqSyncIfTimingHoldoverClear
7-2019-1	tmnxEqSyncIfTimingRef1Alarm with attribute tmnxSyncIfTimingNotifyAlarm == 'los(1)'	Synchronous Timing interface, alarm los on reference 1	tmnxEqSyncIfTimingRef1AlarmClear

Table 49 **Facility Alarm, Facility Alarm Name, Raising Log Event, Sample Details String and Clearing Log Event (Continued)**

Facility Alarm	Facility Alarm Name/Raising Log Event	Sample Details String	Clearing Log Event
7-2019-2	tmnxEqSyncIfTimingRef1Alarm with attribute tmnxSyncIfTimingNotifyAlarm == 'oof(2)'	Synchronous Timing interface, alarm oof on reference 1	same as 7-2019-1
7-2019-3	tmnxEqSyncIfTimingRef1Alarm with attribute tmnxSyncIfTimingNotifyAlarm == 'oopir(3)'	Synchronous Timing interface, alarm oopir on reference 1	same as 7-2019-1
7-2021-x	same as 7-2019-x but for ref2	same as 7-2019-x but for ref2	same as 7-2019-x but for ref2
7-2030-x	same as 7-2019-x but for the BITS input	same as 7-2019-x but for the BITS input	same as 7-2019-x but for the BITS input
7-2033-1	tmnxChassisUpgradeInProgress	Class CPM Module: software upgrade in progress	tmnxChassisUpgradeComplete
7-2073-x	same as 7-2019-x but for the BITS2 input	same as 7-2019-x but for the BITS2 input	same as 7-2019-x but for the BITS2 input
7-2092-1	tmnxEqPowerCapacityExceeded	The system has reached maximum power capacity <x> watts	tmnxEqPowerCapacityExceededClear
7-2094-1	tmnxEqPowerLostCapacity	The system can no longer support configured devices. Power capacity dropped to <x> watts	tmnxEqPowerLostCapacityClear
7-2096-1	tmnxEqPowerOverloadState	The system has reached critical power capacity. Increase available power now	tmnxEqPowerOverloadStateClear
7-2104-1	tmnxEqLowSwitchFabricCap	The switch fabric capacity is less than the forwarding capacity of IOM 1 due to errors in fabric links	tmnxEqLowSwitchFabricCapClear
7-2134-1	tmnxSyncIfTimBITS2048khzUnsup	The revision of 1/1 does not meet the specifications to support the 2048kHz BITS interface type	tmnxSyncIfTimBITS2048khzUnsupClr
7-2136-1	tmnxEqMgmtEthRedStandbyRaise	The standby CPM's management Ethernet port A/1 is serving as the system's management Ethernet port	tmnxEqMgmtEthRedStandbyClear

Table 49 **Facility Alarm, Facility Alarm Name, Raising Log Event, Sample Details String and Clearing Log Event (Continued)**

Facility Alarm	Facility Alarm Name/Raising Log Event	Sample Details String	Clearing Log Event
7-2138-1	tmnxEqPhysChassPowerSupOvrTmp	Power supply 2 over temperature	tmnxEqPhysChassPowerSupOvrTmpClr
7-2140-1	tmnxEqPhysChassPowerSupAcFail	Power supply 1 AC failure	tmnxEqPhysChassPowerSupAcFailClr
7-2142-1	tmnxEqPhysChassPowerSupDcFail	Power supply 2 DC failure	tmnxEqPhysChassPowerSupDcFailClr
7-2144-1	tmnxEqPhysChassPowerSupInFail	Power supply 1 input failure	tmnxEqPhysChassPowerSupInFailClr
7-2146-1	tmnxEqPhysChassPowerSupOutFail	Power supply 1 output failure	tmnxEqPhysChassPowerSupOutFailClr
7-2148-1	tmnxEqPhysChassisFanFailure	Fan 2 failed	tmnxEqPhysChassisFanFailureClear
7-2153-1	tmnxCpmMemSizeMismatch	The standby CPM A has a different memory size than the active B	tmnxCpmMemSizeMismatchClear
7-2156-1	tmnxPhysChassPwrSupWrgFanDir	The front to back fan direction for chassis 1 power supply 1 is not supported	tmnxPhysChassPwrSupWrgFanDirClr
7-2157-1	tmnxPhysChassPwrSupPemACRect	Chassis 1 power supply 1 acRec1 failed or missing	tmnxPhysChassPwrSupPemACRectClr
7-2159-1	tmnxPhysChassPwrSupInputFeed	Chassis 1 power supply 1 inputFeedA not supplying power	tmnxPhysChassPwrSupInputFeedClr
7-2161-1	tmnxEqBpEpromFail	The active CPM is no longer able to access any of backplane EPROMs due to a hardware defect	tmnxEqBpEpromFailClear
7-2163-1	tmnxEqBpEpromWarning	The active CPM is no longer to access one backplane EPROM due to a hardware defect but a redundant EPROM is present and accessible.	tmnxEqBpEpromWarningClear
7-2165-1	tmnxPhysChassisPCMInputFeed	Chassis 1 pcm 1 1 not supplying power	tmnxPhysChassisPCMInputFeedClr
7-2190-1	tmnxPhysChassisPMOutFail	Chassis 1 Power Shelf 1 Power Module 4 output failure	tmnxPhysChassisPMOutFailClr

Table 49 **Facility Alarm, Facility Alarm Name, Raising Log Event, Sample Details String and Clearing Log Event (Continued)**

Facility Alarm	Facility Alarm Name/Raising Log Event	Sample Details String	Clearing Log Event
7-2192-1	tmnxPhysChassisPMInputFeed	Chassis 1 Power Shelf 1 Power Module 3 inputFeedA inputFeedB not supplying power	tmnxPhysChassisPMInputFeedClr
7-2194-1	tmnxPhysChassisFilterDoorOpen	Filter door is missing or open	tmnxPhysChassisFilterDoorClosed
7-2196-1	tmnxPhysChassisPMOverTemp	Chassis 1 Power Shelf 1 over temperature	tmnxPhysChassisPMOverTempClr
7-2203-x	same as 7-2019-x but for SyncE	same as 7-2019-x but for SyncE	same as 7-2019-x but for SyncE
7-2205-x	same as 7-2019-x but for E2	same as 7-2019-x but for E2	same as 7-2019-x but for E2
7-4001-1	tmnxInterChassisCommsDown	Control communications disrupted between the Active CPM and the chassis	tmnxInterChassisCommsUp
7-4003-1	tmnxCpmIcPortDown	CPM Interconnect Port is not operational. Error code = invalid-connection	tmnxCpmIcPortUp
7-4006-1	tmnxCpmIcPortSFFRemoved	CPM interconnect port SFF removed	tmnxCpmIcPortSFFInserted
7-4007-1	tmnxCpmNoLocalIcPort	CPM A can not reach the chassis using its local CPM interconnect ports	tmnxCpmLocalIcPortAvail
7-4017-1	tmnxSfmIcPortDown	SFM interconnect Port is not operational. Error code = invalid-connection to Fabric 10 IcPort 2	tmnxSfmIcPortUp
7-6002-1	tmnxPowerShelfCommsDown	Chassis 1 Power Shelf 1 lost communication with cpmA	tmnxPowerShelfCommsUp
7-6005-1	tmnxPowerShelfOutputStatusDown	Chassis 1 Power Shelf 2 output status switched to off	tmnxPowerShelfOutputStatusUp

Table 50 **Facility Alarm Name/Raising Log Event, Cause, Effect and Recovery**

Facility Alarm	Facility Alarm Name/Raising Log Event	Cause	Effect	Recovery
59-2004-1	linkDown	A linkDown trap signifies that the SNMP entity, acting in an agent role, has detected that the ifOperStatus object for one of its communication links is about to enter the down state from some other state (but not from the notPresent state).	The indicated interface is taken down.	If the ifAdminStatus is down then the interface state is deliberate and there is no recovery. If the ifAdminStatus is up then try to determine that cause of the interface going down: cable cut, distal end went down, and so on.
64-2091-1	tmnxSysLicenseInvalid	Generated when the license becomes invalid for the reason specified in the log event/alarm.	The system will reboot at the end of the time remaining.	Configure a valid license file location and file name.
64-2092-1	tmnxSysLicenseExpiresSoon	Generated when the license is due to expire soon.	The system will reboot at the end of the time remaining.	Configure a valid license file location and file name.
64-2221-1	tmnxSysStandbyLicensingError	Generated when the standby detects a licensing failure. The reason is specified in tmnxSysLicenseErrorReason.	The standby CPM may not synchronized and may be put into a failed state.	Configure a valid license file location and file name, given the value of tmnxSysLicenseErrorReason.
64-2226-1	tmnxSysLicenseUpdateRequired	The tmnxSysLicenseUpdateRequired notification is generated once after the system boots up and the license is determined by the system to be valid, but requires to be updated to the correct software version.	The system will use the license until it is updated.	Update and activate the updated license.

Table 50 **Facility Alarm Name/Raising Log Event, Cause, Effect and Recovery (Continued)**

Facility Alarm	Facility Alarm Name/Raising Log Event	Cause	Effect	Recovery
93-2006-1	tmnxSatSyncIfTimHoldover	The tmnxSatSyncIfTimHoldover notification is generated when the synchronous equipment timing subsystem of the satellite transitions into a holdover state	The transmit timing of all synchronous interfaces on the satellite are no longer synchronous with the host. This could result in traffic loss.	Investigate the state of the two input timing references on the satellite and the links between the host and the satellite (i.e. the uplinks) that drive them for failures.
93-2008-1	tmnxSatSyncIfTimRef1Alarm with attribute tmnxSyncIfTimingNotifyAlarm == 'los(1)'	The tmnxSatSyncIfTimRef1Alarm notification is generated when an alarm condition on the first timing reference is detected.	If the other timing reference is free of faults, the satellite no longer has a backup timing reference. If the other timing reference also has a fault, the satellite will likely no longer be synchronous with the host.	Investigate the state of the link between the host and the satellite (i.e. the uplink) that drives the first timing reference on the satellite for faults.
93-2008-2	tmnxSatSyncIfTimRef1Alarm with attribute tmnxSyncIfTimingNotifyAlarm == 'oof(2)'	The same cause as 93-2008-1	The same effect as 93-2008-1	Investigate the state of the link between the host and the satellite (i.e. the uplink) that drives the first timing reference on the satellite for faults.
93-2008-3	tmnxSatSyncIfTimRef1Alarm with attribute tmnxSyncIfTimingNotifyAlarm == 'oopir(3)'	The same cause as 93-2008-1	The same effect as 93-2008-1	Investigate the state of the link between the host and the satellite (i.e. the uplink) that drives the first timing reference on the satellite for faults.
93-2010-x	same as 93-2008-x but for ref2	The same cause as 93-2008-x but for ref2	The same as 93-2008-x but for ref2	The same as 93-2008-x but for ref2

Table 50 **Facility Alarm Name/Raising Log Event, Cause, Effect and Recovery (Continued)**

Facility Alarm	Facility Alarm Name/Raising Log Event	Cause	Effect	Recovery
7-2001-1	tmnxEqCardFailure	Generated when one of the cards in a chassis has failed. The card type may be IOM (or XCM), MDA (or XMA), SFM, CCM, CPM, Compact Flash, and so on. The reason is indicated in the details of the log event or alarm, and also available in the tmnxChassisNotifyCardFailureReason attribute included in the SNMP notification.	The effect is dependent on the card that has failed. IOM (or XCM) or MDA (or XMA) failure will cause a loss of service for all services running on that card. A fabric failure can impact traffic to and from all cards. 7750 SR, 7450 ESS — If the IOM/IMM fails then the two associated MDAs for the slot will also go down. 7950 XRS — If one out of two XMAs fails in a XCM slot then the XCM will remain up. If only one remaining operational XMA within a XCM slot fails, then the XCM will go into a booting operational state.	Before taking any recovery steps collect a tech-support file, then try resetting (clear) the card. If unsuccessful, try removing and re-inserting the card. If that does not work then replace the card.
7-2003-1	tmnxEqCardRemoved	Generated when a card is removed from the chassis. The card type may be IOM (or XCM), MDA (or XMA), SFM, CCM, CPM, Compact Flash, and so on.	The effect is dependent on the card that has been removed. IOM (or XCM) or MDA (or XMA) removal will cause a loss of service for all services running on that card. A fabric removal can impact traffic to and from all cards.	Before taking any recovery steps collect a tech-support file, then try re-inserting the card. If unsuccessful, replace the card.
7-2004-1	tmnxEqWrongCard	Generated when the wrong type of card is inserted into a slot of the chassis. Even though a card may be physically supported by the slot, it may have been administratively configured to allow only certain card types in a particular slot location. The card type may be IOM (or XCM), MDA (or XMA), SFM, CCM, CPM, Compact Flash, and so on.	The effect is dependent on the card that has been incorrectly inserted. Incorrect IOM (or XCM) or MDA (or XMA) insertion will cause a loss of service for all services running on that card.	Insert the correct card into the correct slot, and ensure the slot is configured for the correct type of card.

Table 50 **Facility Alarm Name/Raising Log Event, Cause, Effect and Recovery (Continued)**

Facility Alarm	Facility Alarm Name/Raising Log Event	Cause	Effect	Recovery
7-2005-1	tmnxEnvTempTooHigh	Generated when the temperature sensor reading on an equipment object is greater than its configured threshold.	This could be causing intermittent errors and could also cause permanent damage to components.	Remove or power down the affected cards, or improve the cooling to the node. More powerful fan trays may also be required.
7-2011-1	tmnxEqPowerSupplyRemoved	Generated when: • one of the power supplies is removed from the chassis • low input voltage is detected. The operating voltage range for the 7750 SR-7/12 and the 7450 ESS-7/12 is -40 to -72 VDC. The alarm is raised if the system detects that the voltage of the power supply has dropped to -42.5 VDC.	Reduced power can cause intermittent errors and could also cause permanent damage to components.	Re-insert the power supply or raise the input voltage to above -42.5 VDC.
7-2017-1	tmnxEqSyncIfTimingHoldover	Generated when the synchronous equipment timing subsystem transitions into a holdover state.	Any node-timed ports will have very slow frequency drift limited by the central clock oscillator stability. The oscillator meets the holdover requirements of a Stratum 3 and G.813 Option 1 clock.	Address issues with the central clock input references.

Table 50 **Facility Alarm Name/Raising Log Event, Cause, Effect and Recovery (Continued)**

Facility Alarm	Facility Alarm Name/Raising Log Event	Cause	Effect	Recovery
7-2019-1	tmnxEqSyncIfTimingRef1Alarm with attribute tmnxSyncIfTimingNotifyAlarm == 'los(1)'	Generated when an alarm condition on the first timing reference is detected. The type of alarm (los, oof, and so on) is indicated in the details of the log event or alarm, and is also available in the tmnxSyncIfTimingNotifyAlarm attribute included in the SNMP notification. The SNMP notification will have the same indices as those of the tmnxCpmCardTable.	Timing reference 1 cannot be used as a source of timing into the central clock.	Address issues with the signal associated with timing reference 1.
7-2019-2	tmnxEqSyncIfTimingRef1Alarm with attribute tmnxSyncIfTimingNotifyAlarm == 'oof(2)'	The same cause as 7-2019-1	The same effect as 7-2019-1	Address issues with the signal associated with timing reference 1.
7-2019-3	tmnxEqSyncIfTimingRef1Alarm with attribute tmnxSyncIfTimingNotifyAlarm == 'oopir(3)'	The same cause as 7-2019-1	The same effect as 7-2019-1	Address issues with the signal associated with timing reference 1.
7-2021-x	same as 7-2019-x but for ref2	The same cause as 7-2019-x but for the second timing reference	The same as 7-2019-x but for the second timing reference	The same as 7-2019-x but for the second timing reference
7-2030-x	same as 7-2019-x but for the BITS input	The same cause as 7-2019-x but for the BITS timing reference	The same as 7-2019-x but for the BITS timing reference	The same as 7-2019-x but for the BITS timing reference

Table 50 **Facility Alarm Name/Raising Log Event, Cause, Effect and Recovery (Continued)**

Facility Alarm	Facility Alarm Name/Raising Log Event	Cause	Effect	Recovery
7-2033-1	tmnxChassisUpgradeInProgress	The tmnxChassisUpgradeInProgress notification is generated only after a CPM switchover occurs and the new active CPM is running new software, while the IOMs or XCMs are still running old software. This is the start of the upgrade process. The tmnxChassisUpgradeInProgress notification will continue to be generated every 30 minutes while at least one IOM or XCM is still running older software.	A software mismatch between the CPM and IOM or XCM is generally fine for a short duration (during an upgrade) but may not allow for correct long term operation.	Complete the upgrade of all IOMs or XCMs.
7-2073-x	same as 7-2019-x but for the BITS2 input	The same as 7-2019-x but for the BITS 2 timing reference	The same as 7-2019-x but for the BITS 2 timing reference	The same as 7-2019-x but for the BITS 2 timing reference
7-2092-1	tmnxEqPowerCapacityExceeded	Generated when a device needs power to boot, but there is not enough power capacity to support the device.	A non-powered device will not boot until the power capacity is increased to support the device.	Add a new power supply to the system, or change the faulty power supply with a working one.
7-2094-1	tmnxEqPowerLostCapacity	Generated when a power supply fails or is removed which puts the system in an overloaded situation.	Devices are powered off in order of lowest power priority until the available power capacity can support the powered devices.	Add a new power supply to the system, or change the faulty power supply with a working one.
7-2096-1	tmnxEqPowerOverloadState	Generated when the overloaded power capacity can not support the power requirements and there are no further devices that can be powered off.	The system runs a risk of experiencing brownouts while the available power capacity does not meet the required power consumption.	Add power capacity or manually shutdown devices until the power capacity meets the power needs.

Table 50 **Facility Alarm Name/Raising Log Event, Cause, Effect and Recovery (Continued)**

Facility Alarm	Facility Alarm Name/Raising Log Event	Cause	Effect	Recovery
7-2104-1	tmnxEqLowSwitchFabricCap	The tmnxEqLowSwitchFabricCap alarm is generated when the total switch fabric capacity becomes less than the IOM capacity due to link failures. At least one of the taps on the IOM is below 100% capacity.	There is diminished switch fabric capacity to forward service-impacting information.	If the system does not self-recover, the IOM must be rebooted.
7-2134-1	tmnxSyncIfTimBITS2048khzUnsup	The tmnxSyncIfTimBITS2048khzUnsup notification is generated when the value of tSyncIfTimingAdmBITSIfType is set to 'g703-2048khz (5)' and the CPM does not meet the specifications for the 2048kHz BITS output signal under G.703.	The BITS input will not be used as the Sync reference and the 2048kHz BITS output signal generated by the CPM is squelched.	Replace the CPM with one that is capable of generating the 2048kHz BITS output signal, or set tSyncIfTimingAdmBITSIfType to a value other than 'g703-2048khz (5)'.
7-2136-1	tmnxEqMgmtEthRedStandbyRaise	The tmnxEqMgmtEthRedStandbyRaise notification is generated when the active CPM's management Ethernet port goes operationally down and the standby CPM's management Ethernet port is operationally up and now serving as the system's management Ethernet port.	The management Ethernet port is no longer redundant. The node can be managed via the standby CPM's management Ethernet port only.	Bring the active CPM's management Ethernet port operationally up.

Table 50 **Facility Alarm Name/Raising Log Event, Cause, Effect and Recovery (Continued)**

Facility Alarm	Facility Alarm Name/Raising Log Event	Cause	Effect	Recovery
7-2138-1	tmnxEqPhysChassPowerSupOvrTmp	Generated when the temperature sensor reading on a power supply module is greater than its configured threshold.	This could be causing intermittent errors and could also cause permanent damage to components.	Remove or power down the affected power supply module or improve the cooling to the node. More powerful fan trays may also be required. The power supply itself may be faulty so replacement may be necessary.
7-2140-1	tmnxEqPhysChassPowerSupAcFail	Generated when an AC failure is detected on a power supply.	Reduced power can cause intermittent errors and could also cause permanent damage to components.	First try re-inserting the power supply. If unsuccessful, replace the power supply.
7-2142-1	tmnxEqPhysChassPowerSupDcFail	Generated when an DC failure is detected on a power supply.	Reduced power can cause intermittent errors and could also cause permanent damage to components.	First try re-inserting the power supply. If unsuccessful, then replace the power supply.
7-2144-1	tmnxEqPhysChassPowerSupInFail	Generated when an input failure is detected on a power supply.	Reduced power can cause intermittent errors and could also cause permanent damage to components.	First try re-inserting the power supply. If that does not work, then replace the power supply.
7-2146-1	tmnxEqPhysChassPowerSupOutFail,	Generated when an output failure is detected on a power supply.	Reduced power can cause intermittent errors and could also cause permanent damage to components.	First try re-inserting the power supply. If that does not work, then replace the power supply.
7-2148-1	tmnxEqPhysChassisFanFailure	Generated when one of the fans in a fan tray has failed.	This could cause the temperature to rise and result in intermittent errors and potentially permanent damage to components.	Replace the fan tray immediately, improve the cooling to the node, or reduce the heat being generated in the node by removing cards or powering down the node.

Table 50 **Facility Alarm Name/Raising Log Event, Cause, Effect and Recovery (Continued)**

Facility Alarm	Facility Alarm Name/Raising Log Event	Cause	Effect	Recovery
7-2153-1	tmnxCpmMemSizeMismatch	A tmnxCpmMemSizeMismatch notification is generated when the RAM memory size of the standby CPM (that is, tmnxChassisNotifyCpmCardSlotNum) is different from the active CPM (that is, tmnxChassisNotifyHwIndex).	There is an increased risk of the memory overflow on the standby CPM during the CPM switchover.	Use CPMs with the same memory size.
7-2156-1	tmnxPhysChassPwrSupWrgFanDir	The tmnxPhysChassPwrSupWrgFanDirClr notification is generated when the airflow direction of the power supply's fan is corrected.	The fan is cooling the power supply in the proper direction.	No recovery required.
7-2157-1	tmnxPhysChassPwrSupPemACRect	The tmnxPhysChassPwrSupPemACRect notification is generated if any one of the AC rectifiers for a given power supply is in a failed state or is missing.	There is an increased risk of the power supply failing, causing insufficient power to the system.	Bring the AC rectifiers back online.
7-2159-1	tmnxPhysChassPwrSupInputFeed	The tmnxPhysChassPwrSupInputFeed notification is generated if any one of the input feeds for a given power supply is not supplying power.	There is an increased risk of system power brown-outs or black-outs.	Restore all of the input feeds that are not supplying power.

Table 50 **Facility Alarm Name/Raising Log Event, Cause, Effect and Recovery (Continued)**

Facility Alarm	Facility Alarm Name/Raising Log Event	Cause	Effect	Recovery
7-2161-1	tmnxEqBpEpromFail	The tmnxEqBpEpromFail alarm is generated when the active CPM is no longer able to access any of backplane EPROMs due to a hardware defect.	The active CPM is at risk of failing to initialize after node reboot due to not being able to access the BP EPROM to read the chassis type.	The system does not self-recover and Nokia Support has to be contacted for further instructions.
7-2163-1	tmnxEqBpEpromWarning	The tmnxEqBpEpromWarning alarm is generated when the active CPM is no longer to access one backplane EPROM due to a hardware defect but a redundant EPROM is present and accessible.	There is no effect on system operation.	No recovery action required.
7-2165-1	tmnxPhysChassisPCMInputFeed	The tmnxPhysChassisPCMInputFeed notification is generated if any one of the input feeds for a given PCM has gone offline.	There is an increased risk of system power brown-outs or black-outs.	Restore all of the input feeds that are not supplying power.
7-2190-1	tmnxPhysChassisPMOutFail	The tmnxPhysChassisPMOutFail notification is generated when an output failure occurs on the power module.	The power module is no longer operational.	Insert a new power module.
7-2192-1	tmnxPhysChassisPMInputFeed	The tmnxPhysChassisPMInputFeed notification is generated if any one of the input feeds for a given power module is not supplying power.	There is an increased risk of system power brownouts or blackouts.	Restore all of the input feeds that are not supplying power.

Table 50 **Facility Alarm Name/Raising Log Event, Cause, Effect and Recovery (Continued)**

Facility Alarm	Facility Alarm Name/Raising Log Event	Cause	Effect	Recovery
7-2194-1	tmnxPhysChassisFilterDoorOpen	The tmnxPhysChassisFilterDoorOpen notification is generated when the filter door is either open or not present.	Power shelf protection may be compromised.	If the filter door is not installed, install it. Close the filter door.
7-2196-1	tmnxPhysChassisPMOverTemp	The tmnxPhysChassisPMOverTemp notification is generated when a power module's temperature surpasses the temperature threshold.	The power module is no longer operational.	Check input feed and/or insert a new power module.
7-2203-x	same as 7-2019-x but for SyncE	The same cause as 7-2019-x but for SyncE	same as 7-2019-x but for SyncE	same as 7-2019-x but for SyncE
7-2205-x	same as 7-2019-x but for E2	The same cause as 7-2019-x but for E2	same as 7-2019-x but for E2	same as 7-2019-x but for E2
7-4001-1	tmnxInterChassisCommsDown	The tmnxInterChassisCommsDown alarm is generated when the active CPM cannot reach the far-end chassis.	The resources on the far-end chassis are not available. This event for the far-end chassis means that the CPM, SFM, and XCM cards in the far-end chassis will reboot and remain operationally down until communications are re-established.	Ensure that all CPM interconnect ports in the system are properly cabled together with working cables.

Table 50 **Facility Alarm Name/Raising Log Event, Cause, Effect and Recovery (Continued)**

Facility Alarm	Facility Alarm Name/Raising Log Event	Cause	Effect	Recovery
7-4003-1	tmnxCpmIcPortDown	The tmnxCpmIcPortDown alarm is generated when the CPM interconnect port is not operational. The reason may be a cable connected incorrectly, a disconnected cable, a faulty cable, or a misbehaving CPM interconnect port or card.	At least one of the control plane paths used for inter-chassis CPM communication is not operational. Other paths may be available.	A manual verification and testing of each CPM interconnect port is required to ensure fully functional operation. Physical replacement of cabling may be required.
7-4006-1	tmnxCpmIcPortSFFRemoved	The tmnxCpmIcPortSFFRemoved notification is generated when the SFF (eg. QSFP) is removed from the CPM interconnect port. Removing an SFF causes both this trap, and also a tmnxCpmIcPortDown event.	Removing the SFF will cause the CPM interconnect port to go down. This port will no longer be able to be used as part of the control plane between chassis but other paths may be available.	Insert a working SFF into the port.

Table 50 **Facility Alarm Name/Raising Log Event, Cause, Effect and Recovery (Continued)**

Facility Alarm	Facility Alarm Name/Raising Log Event	Cause	Effect	Recovery
7-4007-1	tmnxCpmNoLocalIcPort	The tmnxCpmNoLocalIcPort alarm is generated when the CPM cannot reach the other chassis using its local CPM interconnect ports.	Another control communications path may still be available between the CPM and the other chassis via the mate CPM in the same chassis. If that alternative path is not available then complete disruption of control communications to the other chassis will occur and the tmnxInterChassisCommsDown alarm is raised. A tmnxCpmNoLocalIcPort alarm on the active CPM indicates that a further failure of the local CPM interconnect ports on the standby CPM will cause complete disruption of control communications to the other chassis and the tmnxInterChassisCommsDown alarm is raised. A tmnxCpmNoLocalIcPort alarm on the standby CPM indicates that a CPM switchover may cause temporary disruption of control communications to the other chassis while the rebooting CPM comes back into service.	Ensure that all CPM interconnect ports in the system are properly cabled together with working cables.

Table 50 **Facility Alarm Name/Raising Log Event, Cause, Effect and Recovery (Continued)**

Facility Alarm	Facility Alarm Name/Raising Log Event	Cause	Effect	Recovery
7-4017-1	tmnxSfmIcPortDown	The tmnxSfmIcPortDown alarm is generated when the SFM interconnect port is not operational. The reason may be a cable connected incorrectly, a disconnected cable, a faulty cable, or a misbehaving SFM interconnect port or SFM card.	This port can no longer be used as part of the user plane fabric between chassis. Other fabric paths may be available resulting in no loss of capacity.	A manual verification and testing of each SFM interconnect port is required to ensure fully functional operation. Physical replacement of cabling may be required.
7-6002-1	tmnxPowerShelfCommsDown	The tmnxPowerShelfCommsDown is generated when there is a loss of communications with the power shelf controller.	If there is a power failure, it will not be detected since the power modules cannot be polled. The system will continue to report the state of the power modules as they were when last seen.	Correct the power shelf controller communications problem.
7-6005-1	tmnxPowerShelfOutputStatusDown	The tmnxPowerShelfOutputStatusSwitch is generated when the physical output switch on the power shelf is set to Standby.	The power output from the identified power shelf is switched off and does not supply power to the system.	Set output switch to On to restore power output.

The linkDown Facility Alarm is supported for the objects listed in Table 51 (note that all objects may not be supported on all platforms):

Table 51 **linkDown Facility Alarm Support**

Object	Supported
Ethernet Ports	Yes
Sonet Section, Line and Path (POS)	Yes
TDM Ports (E1, T1, DS3) including CES MDAs	Yes
TDM Channels (DS3 channel configured in an STM-1 port)	Yes

Table 51 **linkDown Facility Alarm Support (Continued)**

Object	Supported
ATM Ports	Yes
Ethernet LAGs	No
APS groups	No
Bundles (MLPPP, IMA, and so on)	No
ATM channels, Ethernet VLANs, Frame Relay DLCIs	No

11.6 Configuring Logging with CLI

This section provides information to configure logging using the command line interface.

11.6.1 Basic Facility Alarm Configuration

The most basic facility alarm configuration must have the following:

- Log ID or accounting policy ID
- A log source
- A log destination

The following displays an alarm configuration example.

```
A:ALA-12>config>system# alarms
#------------------------------------------
        no shutdown
        exit
----------------------------------------------
```

11.6.2 Common Configuration Tasks

11.6.2.1 Configuring the Maximum Number of Alarms to Clear

The number of alarms to clear can be configured using the command listed below.

Use the following CLI syntax to configure a log file:

CLI Syntax:

```
config>system
     alarms
   max-cleared max-alarms
```

The following displays facility alarm configuration example:

```
ALA-12>config>system# alarms
----------------------------------------------
...
max-cleared 100
exit
...
----------------------------------------------
```

12 Standards and Protocol Support

Note: The information presented is subject to change without notice.

Nokia assumes no responsibility for inaccuracies contained herein.

Access Node Control Protocol (ANCP)

draft-ietf-ancp-protocol-02, *Protocol for Access Node Control Mechanism in Broadband Networks*

RFC 5851, *Framework and Requirements for an Access Node Control Mechanism in Broadband Multi-Service Networks*

Application Assurance (AA)

3GPP Release 12, *ADC rules over Gx interfaces*

RFC 3507, *Internet Content Adaptation Protocol (ICAP)*

Asynchronous Transfer Mode (ATM)

AF-ILMI-0065.000 Version 4.0, *Integrated Local Management Interface (ILMI)*

AF-PHY-0086.001 Version 1.1, *Inverse Multiplexing for ATM (IMA) Specification*

AF-TM-0121.000 Version 4.1, *Traffic Management Specification*

GR-1113-CORE Issue 1, *Asynchronous Transfer Mode (ATM) and ATM Adaptation Layer (AAL) Protocols Generic Requirements*

GR-1248-CORE Issue 3, *Generic Requirements for Operations of ATM Network Elements (NEs)*

RFC 1626, *Default IP MTU for use over ATM AAL5*

RFC 2684, *Multiprotocol Encapsulation over ATM Adaptation Layer 5*

Bidirectional Forwarding Detection (BFD)

draft-ietf-idr-bgp-ls-sbfd-extensions-01, *BGP Link-State Extensions for Seamless BFD*

RFC 5880, *Bidirectional Forwarding Detection (BFD)*

RFC 5881, *Bidirectional Forwarding Detection (BFD) IPv4 and IPv6 (Single Hop)*

RFC 5882, *Generic Application of Bidirectional Forwarding Detection (BFD)*

RFC 5883, *Bidirectional Forwarding Detection (BFD) for Multihop Paths*

RFC 7130, *Bidirectional Forwarding Detection (BFD) on Link Aggregation Group (LAG) Interfaces*

RFC 7880, *Seamless Bidirectional Forwarding Detection (S-BFD)*

RFC 7881, *Seamless Bidirectional Forwarding Detection (S-BFD) for IPv4, IPv6, and MPLS*

RFC 7883, *Advertising Seamless Bidirectional Forwarding Detection (S-BFD) Discriminators in IS-IS*

RFC 7884, *OSPF Extensions to Advertise Seamless Bidirectional Forwarding Detection (S-BFD) Target Discriminators*

Border Gateway Protocol (BGP)

draft-hares-idr-update-attrib-low-bits-fix-01, *Update Attribute Flag Low Bits Clarification*

draft-ietf-idr-add-paths-guidelines-08, *Best Practices for Advertisement of Multiple Paths in IBGP*

draft-ietf-idr-best-external-03, *Advertisement of the best external route in BGP*

draft-ietf-idr-bgp-flowspec-oid-03, *Revised Validation Procedure for BGP Flow Specifications*

draft-ietf-idr-bgp-gr-notification-01, *Notification Message support for BGP Graceful Restart*

draft-ietf-idr-bgp-ls-app-specific-attr-01, *Application Specific Attributes Advertisement with BGP Link-State*

draft-ietf-idr-bgp-optimal-route-reflection-10, *BGP Optimal Route Reflection (BGP-ORR)*

draft-ietf-idr-error-handling-03, *Revised Error Handling for BGP UPDATE Messages*

draft-ietf-idr-flowspec-interfaceset-03, *Applying BGP flowspec rules on a specific interface set*

draft-ietf-idr-flowspec-path-redirect-05, *Flowspec Indirection-id Redirect* (localised ID)

draft-ietf-idr-flowspec-redirect-ip-02, *BGP Flow-Spec Redirect to IP Action*

draft-ietf-idr-link-bandwidth-03, *BGP Link Bandwidth Extended Community*

draft-ietf-idr-long-lived-gr-00, *Support for Long-lived BGP Graceful Restart*

draft-ietf-sidr-origin-validation-signaling-04, *BGP Prefix Origin Validation State Extended Community*

RFC 1772, *Application of the Border Gateway Protocol in the Internet*

RFC 1997, *BGP Communities Attribute*

RFC 2385, *Protection of BGP Sessions via the TCP MD5 Signature Option*

RFC 2439, *BGP Route Flap Damping*

RFC 2545, *Use of BGP-4 Multiprotocol Extensions for IPv6 Inter-Domain Routing*

RFC 2858, *Multiprotocol Extensions for BGP-4*

RFC 2918, *Route Refresh Capability for BGP-4*

RFC 3107, *Carrying Label Information in BGP-4*

RFC 3392, *Capabilities Advertisement with BGP-4*

RFC 4271, *A Border Gateway Protocol 4 (BGP-4)*

RFC 4360, *BGP Extended Communities Attribute*

RFC 4364, *BGP/MPLS IP Virtual Private Networks (VPNs)*

RFC 4456, *BGP Route Reflection: An Alternative to Full Mesh Internal BGP (IBGP)*

RFC 4486, *Subcodes for BGP Cease Notification Message*

RFC 4659, *BGP-MPLS IP Virtual Private Network (VPN) Extension for IPv6 VPN*

RFC 4684, *Constrained Route Distribution for Border Gateway Protocol/ MultiProtocol Label Switching (BGP/MPLS) Internet Protocol (IP) Virtual Private Networks (VPNs)*

RFC 4724, *Graceful Restart Mechanism for BGP* (helper mode)

RFC 4760, *Multiprotocol Extensions for BGP-4*

RFC 4798, *Connecting IPv6 Islands over IPv4 MPLS Using IPv6 Provider Edge Routers (6PE)*

RFC 4893, *BGP Support for Four-octet AS Number Space*

RFC 5004, *Avoid BGP Best Path Transitions from One External to Another*

RFC 5065, *Autonomous System Confederations for BGP*

RFC 5291, *Outbound Route Filtering Capability for BGP-4*

RFC 5396, *Textual Representation of Autonomous System (AS) Numbers* (asplain)

RFC 5549, *Advertising IPv4 Network Layer Reachability Information with an IPv6 Next Hop*

RFC 5575, *Dissemination of Flow Specification Rules*

RFC 5668, *4-Octet AS Specific BGP Extended Community*

RFC 6286, *Autonomous-System-Wide Unique BGP Identifier for BGP-4*

RFC 6810, *The Resource Public Key Infrastructure (RPKI) to Router Protocol*

RFC 6811, *Prefix Origin Validation*

RFC 6996, *Autonomous System (AS) Reservation for Private Use*

RFC 7311, *The Accumulated IGP Metric Attribute for BGP*

RFC 7607, *Codification of AS 0 Processing*

RFC 7674, *Clarification of the Flowspec Redirect Extended Community*

RFC 7752, *North-Bound Distribution of Link-State and Traffic Engineering (TE) Information Using BGP*

RFC 7854, *BGP Monitoring Protocol (BMP)*

RFC 7911, *Advertisement of Multiple Paths in BGP*

RFC 7999, *BLACKHOLE Community*

RFC 8092, *BGP Large Communities Attribute*

RFC 8212, *Default External BGP (EBGP) Route Propagation Behavior without Policies*

RFC 8571, *BGP - Link State (BGP-LS) Advertisement of IGP Traffic Engineering Performance Metric Extensions*

Broadband Network Gateway (BNG) - Control and User Plane Separation (CUPS)

3GPP 23.007, *Restoration procedures*

3GPP 29.244, *Interface between the Control Plane and the User Plane nodes*

3GPP 29.281, *General Packet Radio System (GPRS) Tunnelling Protocol User Plane (GTPv1-U)*

BBF TR-459, *Control and User Plane Separation for a Disaggregated BNG*

RFC 8300, *Network Service Header (NSH)*

Circuit Emulation

RFC 4553, *Structure-Agnostic Time Division Multiplexing (TDM) over Packet (SAToP)*

RFC 5086, *Structure-Aware Time Division Multiplexed (TDM) Circuit Emulation Service over Packet Switched Network (CESoPSN)*

RFC 5287, *Control Protocol Extensions for the Setup of Time-Division Multiplexing (TDM) Pseudowires in MPLS Networks*

Ethernet

IEEE 802.1AB, *Station and Media Access Control Connectivity Discovery*

IEEE 802.1ad, *Provider Bridges*

IEEE 802.1ag, *Connectivity Fault Management*

IEEE 802.1ah, *Provider Backbone Bridges*

IEEE 802.1ak, *Multiple Registration Protocol*

IEEE 802.1aq, *Shortest Path Bridging*

IEEE 802.1ax, *Link Aggregation*

IEEE 802.1D, *MAC Bridges*

IEEE 802.1p, *Traffic Class Expediting*

IEEE 802.1Q, *Virtual LANs*

IEEE 802.1s, *Multiple Spanning Trees*

IEEE 802.1w, *Rapid Reconfiguration of Spanning Tree*

IEEE 802.1X, *Port Based Network Access Control*

IEEE 802.3ac, *VLAN Tag*

IEEE 802.3ad, *Link Aggregation*

IEEE 802.3ah, *Ethernet in the First Mile*

IEEE 802.3x, *Ethernet Flow Control*

ITU-T G.8031/Y.1342, *Ethernet Linear Protection Switching*

ITU-T G.8032/Y.1344, *Ethernet Ring Protection Switching*

ITU-T Y.1731, *OAM functions and mechanisms for Ethernet based networks*

Ethernet VPN (EVPN)

draft-ietf-bess-evpn-igmp-mld-proxy-05, *IGMP and MLD Proxy for EVPN*

draft-ietf-bess-evpn-irb-mcast-04, *EVPN Optimized Inter-Subnet Multicast (OISM) Forwarding* (ingress replication)

draft-ietf-bess-evpn-pref-df-06, *Preference-based EVPN DF Election*

draft-ietf-bess-evpn-prefix-advertisement-11, *IP Prefix Advertisement in EVPN*

draft-ietf-bess-evpn-proxy-arp-nd-08, *Operational Aspects of Proxy-ARP/ND in EVPN Networks*

draft-ietf-bess-pbb-evpn-isid-cmacflush-00, *PBB-EVPN ISID-based CMAC-Flush*

RFC 7432, *BGP MPLS-Based Ethernet VPN*

RFC 7623, *Provider Backbone Bridging Combined with Ethernet VPN (PBB-EVPN)*

RFC 8214, *Virtual Private Wire Service Support in Ethernet VPN*

RFC 8317, *Ethernet-Tree (E-Tree) Support in Ethernet VPN (EVPN) an Provider Backbone Bridging EVPN (PBB-EVPN)*

RFC 8365, *A Network Virtualization Overlay Solution Using Ethernet VPN (EVPN)*

RFC 8560, *Seamless Integration of Ethernet VPN (EVPN) with Virtual Private LAN Service (VPLS) and Their Provider Backbone Bridge (PBB) Equivalents*

RFC 8584, *DF Election and AC-influenced DF Election*

Frame Relay

ANSI T1.617 Annex D, *DSS1 - Signalling Specification For Frame Relay Bearer Service*

FRF.1.2, *PVC User-to-Network Interface (UNI) Implementation Agreement*

FRF.12, *Frame Relay Fragmentation Implementation Agreement*

FRF.16.1, *Multilink Frame Relay UNI/NNI Implementation Agreement*

FRF.5, *Frame Relay/ATM PVC Network Interworking Implementation*

FRF2.2, *PVC Network-to-Network Interface (NNI) Implementation Agreement*

ITU-T Q.933 Annex A, *Additional procedures for Permanent Virtual Connection (PVC) status management*

Generalized Multiprotocol Label Switching (GMPLS)

draft-ietf-ccamp-rsvp-te-srlg-collect-04, *RSVP-TE Extensions for Collecting SRLG Information*

RFC 3471, *Generalized Multi-Protocol Label Switching (GMPLS) Signaling Functional Description*

RFC 3473, *Generalized Multi-Protocol Label Switching (GMPLS) Signaling Resource ReserVation Protocol-Traffic Engineering (RSVP-TE) Extensions*

RFC 4204, *Link Management Protocol (LMP)*

RFC 4208, *Generalized Multiprotocol Label Switching (GMPLS) User-Network Interface (UNI): Resource ReserVation Protocol-Traffic Engineering (RSVP-TE) Support for the Overlay Model*

RFC 4872, *RSVP-TE Extensions in Support of End-to-End Generalized Multi-Protocol Label Switching (GMPLS) Recovery*

RFC 5063, *Extensions to GMPLS Resource Reservation Protocol (RSVP) Graceful Restart* (helper mode)

gRPC Remote Procedure Calls (gRPC)

cert.proto Version 0.1.0, *gRPC Network Operations Interface (gNOI) Certificate Management Service*

gnmi.proto Version 0.7.0, *gRPC Network Management Interface (gNMI) Service Specification*

PROTOCOL-HTTP2, *gRPC over HTTP2*

system.proto Version 1.0.0, *gRPC Network Operations Interface (gNOI) System Service*

Intermediate System to Intermediate System (IS-IS)

draft-ietf-isis-mi-02, *IS-IS Multi-Instance*

draft-kaplan-isis-ext-eth-02, *Extended Ethernet Frame Size Support*

ISO/IEC 10589:2002 Second Edition, *Intermediate system to Intermediate system intra-domain routeing information exchange protocol for use in conjunction with the protocol for providing the connectionless-mode Network Service (ISO 8473)*

RFC 1195, *Use of OSI IS-IS for Routing in TCP/IP and Dual Environments*

RFC 2973, *IS-IS Mesh Groups*

RFC 3359, *Reserved Type, Length and Value (TLV) Codepoints in Intermediate System to Intermediate System*

RFC 3719, *Recommendations for Interoperable Networks using Intermediate System to Intermediate System (IS-IS)*

RFC 3787, *Recommendations for Interoperable IP Networks using Intermediate System to Intermediate System (IS-IS)*

RFC 4971, *Intermediate System to Intermediate System (IS-IS) Extensions for Advertising Router Information*

RFC 5120, *M-ISIS: Multi Topology (MT) Routing in IS-IS*

RFC 5130, *A Policy Control Mechanism in IS-IS Using Administrative Tags*

RFC 5301, *Dynamic Hostname Exchange Mechanism for IS-IS*

RFC 5302, *Domain-wide Prefix Distribution with Two-Level IS-IS*

RFC 5303, *Three-Way Handshake for IS-IS Point-to-Point Adjacencies*

RFC 5304, *IS-IS Cryptographic Authentication*

RFC 5305, *IS-IS Extensions for Traffic Engineering TE*

RFC 5306, *Restart Signaling for IS-IS* (helper mode)

RFC 5307, *IS-IS Extensions in Support of Generalized Multi-Protocol Label Switching (GMPLS)*

RFC 5308, *Routing IPv6 with IS-IS*

RFC 5309, *Point-to-Point Operation over LAN in Link State Routing Protocols*

RFC 5310, *IS-IS Generic Cryptographic Authentication*

RFC 6119, *IPv6 Traffic Engineering in IS-IS*

RFC 6213, *IS-IS BFD-Enabled TLV*

RFC 6232, *Purge Originator Identification TLV for IS-IS*

RFC 6233, *IS-IS Registry Extension for Purges*

RFC 6329, *IS-IS Extensions Supporting IEEE 802.1aq Shortest Path Bridging*

RFC 7775, *IS-IS Route Preference for Extended IP and IPv6 Reachability*

RFC 7794, *IS-IS Prefix Attributes for Extended IPv4 and IPv6 Reachability*

RFC 7987, *IS-IS Minimum Remaining Lifetime*

RFC 8202, *IS-IS Multi-Instance* (single topology)

RFC 8570, *IS-IS Traffic Engineering (TE) Metric Extensions* (delay metric)

RFC 8919, *IS-IS Application-Specific Link Attributes*

Internet Protocol (IP) — Fast Reroute

draft-ietf-rtgwg-lfa-manageability-08, *Operational management of Loop Free Alternates*

RFC 5286, *Basic Specification for IP Fast Reroute: Loop-Free Alternates*

RFC 7431, *Multicast-Only Fast Reroute*

RFC 7490, *Remote Loop-Free Alternate (LFA) Fast Reroute (FRR)*

Internet Protocol (IP) — General

draft-grant-tacacs-02, *The TACACS+ Protocol*

RFC 768, *User Datagram Protocol*

RFC 793, *Transmission Control Protocol*

RFC 854, *Telnet Protocol Specifications*

RFC 1350, *The TFTP Protocol (revision 2)*

RFC 2347, *TFTP Option Extension*

RFC 2348, *TFTP Blocksize Option*

RFC 2349, *TFTP Timeout Interval and Transfer Size Options*

RFC 2428, *FTP Extensions for IPv6 and NATs*

RFC 2784, *Generic Routing Encapsulation (GRE)*

RFC 2818, *HTTP Over TLS*

RFC 2890, *Key and Sequence Number Extensions to GRE*

RFC 3164, *The BSD syslog Protocol*

RFC 4250, *The Secure Shell (SSH) Protocol Assigned Numbers*

RFC 4251, *The Secure Shell (SSH) Protocol Architecture*

RFC 4252, *The Secure Shell (SSH) Authentication Protocol* (publickey, password)

RFC 4253, *The Secure Shell (SSH) Transport Layer Protocol*

RFC 4254, *The Secure Shell (SSH) Connection Protocol*

RFC 4511, *Lightweight Directory Access Protocol (LDAP): The Protocol*

RFC 4513, *Lightweight Directory Access Protocol (LDAP): Authentication Methods and Security Mechanisms* (TLS)

RFC 4632, *Classless Inter-domain Routing (CIDR): The Internet Address Assignment and Aggregation Plan*

RFC 5082, *The Generalized TTL Security Mechanism (GTSM)*

RFC 5246, *The Transport Layer Security (TLS) Protocol Version 1.2* (TLS client, RSA public key)

RFC 5656, *Elliptic Curve Algorithm Integration in the Secure Shell Transport Layer* (ECDSA)

RFC 5925, *The TCP Authentication Option*

RFC 5926, *Cryptographic Algorithms for the TCP Authentication Option (TCP-AO)*

RFC 6398, *IP Router Alert Considerations and Usage* (MLD)

RFC 6528, *Defending against Sequence Number Attacks*

RFC 7011, *Specification of the IP Flow Information Export (IPFIX) Protocol for the Exchange of Flow Information*

RFC 7012, *Information Model for IP Flow Information Export*

RFC 7230, *Hypertext Transfer Protocol (HTTP/1.1): Message Syntax and Routing*

RFC 7231, *Hypertext Transfer Protocol (HTTP/1.1): Semantics and Content*

RFC 7232, *Hypertext Transfer Protocol (HTTP/1.1): Conditional Requests*

Internet Protocol (IP) — Multicast

cisco-ipmulticast/pim-autorp-spec01, *Auto-RP: Automatic discovery of Group-to-RP mappings for IP multicast* (version 1)

draft-dolganow-bess-mvpn-expl-track-01, *Explicit Tracking with Wild Card Routes in Multicast VPN*

draft-ietf-bier-mvpn-11, *Multicast VPN Using BIER*

draft-ietf-bier-pim-signaling-08, *PIM Signaling Through BIER Core*

draft-ietf-idmr-traceroute-ipm-07, *A "traceroute" facility for IP Multicast*

draft-ietf-l2vpn-vpls-pim-snooping-07, *Protocol Independent Multicast (PIM) over Virtual Private LAN Service (VPLS)*

RFC 1112, *Host Extensions for IP Multicasting*

RFC 2236, *Internet Group Management Protocol, Version 2*

RFC 2365, *Administratively Scoped IP Multicast*

RFC 2375, *IPv6 Multicast Address Assignments*

RFC 2710, *Multicast Listener Discovery (MLD) for IPv6*

RFC 3306, *Unicast-Prefix-based IPv6 Multicast Addresses*

RFC 3376, *Internet Group Management Protocol, Version 3*

RFC 3446, *Anycast Rendevous Point (RP) mechanism using Protocol Independent Multicast (PIM) and Multicast Source Discovery Protocol (MSDP)*

RFC 3590, *Source Address Selection for the Multicast Listener Discovery (MLD) Protocol*

RFC 3618, *Multicast Source Discovery Protocol (MSDP)*

RFC 3810, *Multicast Listener Discovery Version 2 (MLDv2) for IPv6*

RFC 3956, *Embedding the Rendezvous Point (RP) Address in an IPv6 Multicast Address*

RFC 3973, *Protocol Independent Multicast - Dense Mode (PIM-DM): Protocol Specification (Revised)* (auto-RP groups)

RFC 4541, *Considerations for Internet Group Management Protocol (IGMP) and Multicast Listener Discovery (MLD) Snooping Switches*

RFC 4604, *Using Internet Group Management Protocol Version 3 (IGMPv3) and Multicast Listener Discovery Protocol Version 2 (MLDv2) for Source-Specific Multicast*

RFC 4607, *Source-Specific Multicast for IP*

RFC 4608, *Source-Specific Protocol Independent Multicast in 232/8*

RFC 4610, *Anycast-RP Using Protocol Independent Multicast (PIM)*

RFC 4611, *Multicast Source Discovery Protocol (MSDP) Deployment Scenarios*

RFC 5059, *Bootstrap Router (BSR) Mechanism for Protocol Independent Multicast (PIM)*

RFC 5186, *Internet Group Management Protocol Version 3 (IGMPv3) / Multicast Listener Discovery Version 2 (MLDv2) and Multicast Routing Protocol Interaction*

RFC 5384, *The Protocol Independent Multicast (PIM) Join Attribute Format*

RFC 5496, *The Reverse Path Forwarding (RPF) Vector TLV*

RFC 6037, *Cisco Systems' Solution for Multicast in MPLS/BGP IP VPNs*

RFC 6512, *Using Multipoint LDP When the Backbone Has No Route to the Root*

RFC 6513, *Multicast in MPLS/BGP IP VPNs*

RFC 6514, *BGP Encodings and Procedures for Multicast in MPLS/IP VPNs*

RFC 6515, *IPv4 and IPv6 Infrastructure Addresses in BGP Updates for Multicast VPNs*

RFC 6516, *IPv6 Multicast VPN (MVPN) Support Using PIM Control Plane and Selective Provider Multicast Service Interface (S-PMSI) Join Messages*

RFC 6625, *Wildcards in Multicast VPN Auto-Discover Routes*

RFC 6826, *Multipoint LDP In-Band Signaling for Point-to-Multipoint and Multipoint-to-Multipoint Label Switched Path*

RFC 7246, *Multipoint Label Distribution Protocol In-Band Signaling in a Virtual Routing and Forwarding (VRF) Table Context*

RFC 7385, *IANA Registry for P-Multicast Service Interface (PMSI) Tunnel Type Code Points*

RFC 7716, *Global Table Multicast with BGP Multicast VPN (BGP-MVPN) Procedures*

RFC 7761, *Protocol Independent Multicast - Sparse Mode (PIM-SM): Protocol Specification (Revised)*

RFC 8279, *Multicast Using Bit Index Explicit Replication (BIER)*

RFC 8296, *Encapsulation for Bit Index Explicit Replication (BIER) in MPLS and Non-MPLS Networks* (MPLS encapsulation)

RFC 8401, *Bit Index Explicit Replication (BIER) Support via IS-IS*

RFC 8444, *OSPFv2 Extensions for Bit Index Explicit Replication (BIER)*

RFC 8487, *Mtrace Version 2: Traceroute Facility for IP Multicast*

Internet Protocol (IP) — Version 4

RFC 791, *Internet Protocol*

RFC 792, *Internet Control Message Protocol*

RFC 826, *An Ethernet Address Resolution Protocol*

RFC 951, *Bootstrap Protocol (BOOTP)* (relay)

RFC 1034, *Domain Names - Concepts and Facilities*

RFC 1035, *Domain Names - Implementation and Specification*

RFC 1191, *Path MTU Discovery* (router specification)

RFC 1519, *Classless Inter-Domain Routing (CIDR): an Address Assignment and Aggregation Strategy*

RFC 1534, *Interoperation between DHCP and BOOTP*

RFC 1542, *Clarifications and Extensions for the Bootstrap Protocol*

RFC 1812, *Requirements for IPv4 Routers*

RFC 1918, *Address Allocation for Private Internets*

RFC 2003, *IP Encapsulation within IP*

RFC 2131, *Dynamic Host Configuration Protocol*

RFC 2132, *DHCP Options and BOOTP Vendor Extensions*

RFC 2401, *Security Architecture for Internet Protocol*

RFC 3021, *Using 31-Bit Prefixes on IPv4 Point-to-Point Links*

RFC 3046, *DHCP Relay Agent Information Option (Option 82)*

RFC 3768, *Virtual Router Redundancy Protocol (VRRP)*

RFC 4884, *Extended ICMP to Support Multi-Part Messages* (ICMPv4 and ICMPv6 Time Exceeded)

Internet Protocol (IP) — Version 6

RFC 2464, *Transmission of IPv6 Packets over Ethernet Networks*

RFC 2529, *Transmission of IPv6 over IPv4 Domains without Explicit Tunnels*

RFC 3122, *Extensions to IPv6 Neighbor Discovery for Inverse Discovery Specification*

RFC 3315, *Dynamic Host Configuration Protocol for IPv6 (DHCPv6)*

RFC 3587, *IPv6 Global Unicast Address Format*

RFC 3596, *DNS Extensions to Support IP version 6*

RFC 3633, *IPv6 Prefix Options for Dynamic Host Configuration Protocol (DHCP) version 6*

RFC 3646, *DNS Configuration options for Dynamic Host Configuration Protocol for IPv6 (DHCPv6)*

RFC 3736, *Stateless Dynamic Host Configuration Protocol (DHCP) Service for IPv6*

RFC 3971, *SEcure Neighbor Discovery (SEND)*

RFC 3972, *Cryptographically Generated Addresses (CGA)*

RFC 4007, *IPv6 Scoped Address Architecture*

RFC 4193, *Unique Local IPv6 Unicast Addresses*

RFC 4291, *Internet Protocol Version 6 (IPv6) Addressing Architecture*

RFC 4443, *Internet Control Message Protocol (ICMPv6) for the Internet Protocol Version 6 (IPv6) Specification*

RFC 4861, *Neighbor Discovery for IP version 6 (IPv6)*

RFC 4862, *IPv6 Stateless Address Autoconfiguration* (router functions)

RFC 4890, *Recommendations for Filtering ICMPv6 Messages in Firewalls*

RFC 4941, *Privacy Extensions for Stateless Address Autoconfiguration in IPv6*

RFC 5007, *DHCPv6 Leasequery*

RFC 5095, *Deprecation of Type 0 Routing Headers in IPv6*

RFC 5722, *Handling of Overlapping IPv6 Fragments*

RFC 5798, *Virtual Router Redundancy Protocol (VRRP) Version 3 for IPv4 and IPv6* (IPv6)

RFC 5952, *A Recommendation for IPv6 Address Text Representation*

RFC 6092, *Recommended Simple Security Capabilities in Customer Premises Equipment (CPE) for Providing Residential IPv6 Internet Service* (Internet Control and Management, Upper-Layer Transport Protocols, UDP Filters, IPsec and Internet Key Exchange (IKE), TCP Filters)

RFC 6106, *IPv6 Router Advertisement Options for DNS Configuration*

RFC 6164, *Using 127-Bit IPv6 Prefixes on Inter-Router Links*

RFC 8021, *Generation of IPv6 Atomic Fragments Considered Harmful*

RFC 8200, *Internet Protocol, Version 6 (IPv6) Specification*

RFC 8201, *Path MTU Discovery for IP version 6*

Internet Protocol Security (IPsec)

draft-ietf-ipsec-isakmp-mode-cfg-05, *The ISAKMP Configuration Method*

draft-ietf-ipsec-isakmp-xauth-06, *Extended Authentication within ISAKMP/Oakley (XAUTH)*

RFC 2401, *Security Architecture for the Internet Protocol*

RFC 2403, *The Use of HMAC-MD5-96 within ESP and AH*

RFC 2404, *The Use of HMAC-SHA-1-96 within ESP and AH*

RFC 2405, *The ESP DES-CBC Cipher Algorithm With Explicit IV*

RFC 2406, *IP Encapsulating Security Payload (ESP)*

RFC 2407, *IPsec Domain of Interpretation for ISAKMP (IPsec DoI)*

RFC 2408, *Internet Security Association and Key Management Protocol (ISAKMP)*

RFC 2409, *The Internet Key Exchange (IKE)*

RFC 2410, *The NULL Encryption Algorithm and Its Use With IPsec*

RFC 3526, *More Modular Exponential (MODP) Diffie-Hellman group for Internet Key Exchange (IKE)*

RFC 3566, *The AES-XCBC-MAC-96 Algorithm and Its Use With IPsec*

RFC 3602, *The AES-CBC Cipher Algorithm and Its Use with IPsec*

RFC 3706, *A Traffic-Based Method of Detecting Dead Internet Key Exchange (IKE) Peers*

RFC 3947, *Negotiation of NAT-Traversal in the IKE*

RFC 3948, *UDP Encapsulation of IPsec ESP Packets*

RFC 4106, *The Use of Galois/Counter Mode (GCM) in IPsec ESP*

RFC 4210, *Internet X.509 Public Key Infrastructure Certificate Management Protocol (CMP)*

RFC 4211, *Internet X.509 Public Key Infrastructure Certificate Request Message Format (CRMF)*

RFC 4301, *Security Architecture for the Internet Protocol*

RFC 4303, *IP Encapsulating Security Payload*

RFC 4307, *Cryptographic Algorithms for Use in the Internet Key Exchange Version 2 (IKEv2)*

RFC 4308, *Cryptographic Suites for IPsec*

RFC 4434, *The AES-XCBC-PRF-128 Algorithm for the Internet Key Exchange Protocol (IKE)*

RFC 4543, *The Use of Galois Message Authentication Code (GMAC) in IPsec ESP and AH*

RFC 4868, *Using HMAC-SHA-256, HMAC-SHA-384, and HMAC-SHA-512 with IPSec*

RFC 4945, *The Internet IP Security PKI Profile of IKEv1/ISAKMP, IKEv2 and PKIX*

RFC 5019, *The Lightweight Online Certificate Status Protocol (OCSP) Profile for High-Volume Environments*

RFC 5280, *Internet X.509 Public Key Infrastructure Certificate and Certificate Revocation List (CRL) Profile*

RFC 5282, *Using Authenticated Encryption Algorithms with the Encrypted Payload of the IKEv2 Protocol*

RFC 5903, *ECP Groups for IKE and IKEv2*

RFC 5998, *An Extension for EAP-Only Authentication in IKEv2*

RFC 6379, *Suite B Cryptographic Suites for IPsec*

RFC 6380, *Suite B Profile for Internet Protocol Security (IPsec)*

RFC 6712, *Internet X.509 Public Key Infrastructure -- HTTP Transfer for the Certificate Management Protocol (CMP)*

RFC 6960, *X.509 Internet Public Key Infrastructure Online Certificate Status Protocol - OCSP*

RFC 7296, *Internet Key Exchange Protocol Version 2 (IKEv2)*

RFC 7321, *Cryptographic Algorithm Implementation Requirements and Usage Guidance for Encapsulating Security Payload (ESP) and Authentication Header (AH)*

RFC 7383, *Internet Key Exchange Protocol Version 2 (IKEv2) Message Fragmentation*

RFC 7427, *Signature Authentication in the Internet Key Exchange Version 2 (IKEv2)*

RFC 7468, *Textual Encodings of PKIX, PKCS, and CMS Structures*

Label Distribution Protocol (LDP)

draft-pdutta-mpls-ldp-adj-capability-00, *LDP Adjacency Capabilities*

draft-pdutta-mpls-ldp-v2-00, *LDP Version 2*

draft-pdutta-mpls-mldp-up-redundancy-00, *Upstream LSR Redundancy for Multi-point LDP Tunnels*

draft-pdutta-mpls-multi-ldp-instance-00, *Multiple LDP Instances*

draft-pdutta-mpls-tldp-hello-reduce-04, *Targeted LDP Hello Reduction*

RFC 3037, *LDP Applicability*

RFC 3478, *Graceful Restart Mechanism for Label Distribution Protocol* (helper mode)

RFC 5036, *LDP Specification*

RFC 5283, *LDP Extension for Inter-Area Label Switched Paths (LSPs)*

RFC 5443, *LDP IGP Synchronization*

RFC 5561, *LDP Capabilities*

RFC 5919, *Signaling LDP Label Advertisement Completion*

RFC 6388, *Label Distribution Protocol Extensions for Point-to-Multipoint and Multipoint-to-Multipoint Label Switched Paths*

RFC 6512, *Using Multipoint LDP When the Backbone Has No Route to the Root*

RFC 6826, *Multipoint LDP in-band signaling for Point-to-Multipoint and Multipoint-to-Multipoint Label Switched Paths*

RFC 7032, *LDP Downstream-on-Demand in Seamless MPLS*

RFC 7473, *Controlling State Advertisements of Non-negotiated LDP Applications*

RFC 7552, *Updates to LDP for IPv6*

Layer Two Tunneling Protocol (L2TP) Network Server (LNS)

draft-mammoliti-l2tp-accessline-avp-04, *Layer 2 Tunneling Protocol (L2TP) Access Line Information Attribute Value Pair (AVP) Extensions*

RFC 2661, *Layer Two Tunneling Protocol "L2TP"*

RFC 2809, *Implementation of L2TP Compulsory Tunneling via RADIUS*

RFC 3438, *Layer Two Tunneling Protocol (L2TP) Internet Assigned Numbers: Internet Assigned Numbers Authority (IANA) Considerations Update*

RFC 3931, *Layer Two Tunneling Protocol - Version 3 (L2TPv3)*

RFC 4719, *Transport of Ethernet Frames over Layer 2 Tunneling Protocol Version 3 (L2TPv3)*

RFC 4951, *Fail Over Extensions for Layer 2 Tunneling Protocol (L2TP) "failover"*

Multiprotocol Label Switching (MPLS)

draft-ietf-mpls-lsp-ping-ospfv3-codepoint-02, *OSPFv3 CodePoint for MPLS LSP Ping*

RFC 3031, *Multiprotocol Label Switching Architecture*

RFC 3032, *MPLS Label Stack Encoding*

RFC 3443, *Time To Live (TTL) Processing in Multi-Protocol Label Switching (MPLS) Networks*

RFC 4023, *Encapsulating MPLS in IP or Generic Routing Encapsulation (GRE)*

RFC 4182, *Removing a Restriction on the use of MPLS Explicit NULL*

RFC 5332, *MPLS Multicast Encapsulations*

RFC 5884, *Bidirectional Forwarding Detection (BFD) for MPLS Label Switched Paths (LSPs)*

RFC 6374, *Packet Loss and Delay Measurement for MPLS Networks* (Delay Measurement, Channel Type 0x000C)

RFC 6424, *Mechanism for Performing Label Switched Path Ping (LSP Ping) over MPLS Tunnels*

RFC 6425, *Detecting Data Plane Failures in Point-to-Multipoint Multiprotocol Label Switching (MPLS) - Extensions to LSP Ping*

RFC 6790, *The Use of Entropy Labels in MPLS Forwarding*

RFC 7510, *Encapsulating MPLS in UDP*

RFC 7746, *Label Switched Path (LSP) Self-Ping*

RFC 7876, *UDP Return Path for Packet Loss and Delay Measurement for MPLS Networks* (Delay Measurement)

RFC 8029, *Detecting Multiprotocol Label Switched (MPLS) Data-Plane Failures*

Multiprotocol Label Switching — Transport Profile (MPLS-TP)

RFC 5586, *MPLS Generic Associated Channel*

RFC 5921, *A Framework for MPLS in Transport Networks*

RFC 5960, *MPLS Transport Profile Data Plane Architecture*

RFC 6370, *MPLS Transport Profile (MPLS-TP) Identifiers*

RFC 6378, *MPLS Transport Profile (MPLS-TP) Linear Protection*

RFC 6426, *MPLS On-Demand Connectivity and Route Tracing*

RFC 6427, *MPLS Fault Management Operations, Administration, and Maintenance (OAM)*

RFC 6428, *Proactive Connectivity Verification, Continuity Check and Remote Defect indication for MPLS Transport Profile*

RFC 6478, *Pseudowire Status for Static Pseudowires*

RFC 7213, *MPLS Transport Profile (MPLS-TP) Next-Hop Ethernet Addressing*

Network Address Translation (NAT)

draft-ietf-behave-address-format-10, *IPv6 Addressing of IPv4/IPv6 Translators*

draft-ietf-behave-v6v4-xlate-23, *IP/ICMP Translation Algorithm*

draft-miles-behave-l2nat-00, *Layer2-Aware NAT*

draft-nishitani-cgn-02, *Common Functions of Large Scale NAT (LSN)*

RFC 4787, *Network Address Translation (NAT) Behavioral Requirements for Unicast UDP*

RFC 5382, *NAT Behavioral Requirements for TCP*

RFC 5508, *NAT Behavioral Requirements for ICMP*

RFC 6146, *Stateful NAT64: Network Address and Protocol Translation from IPv6 Clients to IPv4 Servers*

RFC 6333, *Dual-Stack Lite Broadband Deployments Following IPv4 Exhaustion*

RFC 6334, *Dynamic Host Configuration Protocol for IPv6 (DHCPv6) Option for Dual-Stack Lite*

RFC 6887, *Port Control Protocol (PCP)*

RFC 6888, *Common Requirements For Carrier-Grade NATs (CGNs)*

RFC 7753, *Port Control Protocol (PCP) Extension for Port-Set Allocation*

RFC 7915, *IP/ICMP Translation Algorithm*

Network Configuration Protocol (NETCONF)

RFC 5277, *NETCONF Event Notifications*

RFC 6020, *YANG - A Data Modeling Language for the Network Configuration Protocol (NETCONF)*

RFC 6022, *YANG Module for NETCONF Monitoring*

RFC 6241, *Network Configuration Protocol (NETCONF)*

RFC 6242, *Using the NETCONF Protocol over Secure Shell (SSH)*

RFC 6243, *With-defaults Capability for NETCONF*

RFC 8342, *Network Management Datastore Architecture (NMDA)* (Startup, Candidate, Running and Intended datastores)

RFC 8525, *YANG Library*

RFC 8526, *NETCONF Extensions to Support the Network Management Datastore Architecture* (<get-data> operation)

Open Shortest Path First (OSPF)

RFC 1586, *Guidelines for Running OSPF Over Frame Relay Networks*

RFC 1765, *OSPF Database Overflow*

RFC 2328, *OSPF Version 2*

RFC 3101, *The OSPF Not-So-Stubby Area (NSSA) Option*

RFC 3509, *Alternative Implementations of OSPF Area Border Routers*

RFC 3623, *Graceful OSPF Restart Graceful OSPF Restart* (helper mode)

RFC 3630, *Traffic Engineering (TE) Extensions to OSPF Version 2*

RFC 4203, *OSPF Extensions in Support of Generalized Multi-Protocol Label Switching (GMPLS)*

RFC 4222, *Prioritized Treatment of Specific OSPF Version 2 Packets and Congestion Avoidance*

RFC 4552, *Authentication/Confidentiality for OSPFv3*

RFC 4576, *Using a Link State Advertisement (LSA) Options Bit to Prevent Looping in BGP/MPLS IP Virtual Private Networks (VPNs)*

RFC 4577, *OSPF as the Provider/Customer Edge Protocol for BGP/MPLS IP Virtual Private Networks (VPNs)*

RFC 5185, *OSPF Multi-Area Adjacency*

RFC 5187, *OSPFv3 Graceful Restart* (helper mode)

RFC 5243, *OSPF Database Exchange Summary List Optimization*

RFC 5250, *The OSPF Opaque LSA Option*

RFC 5309, *Point-to-Point Operation over LAN in Link State Routing Protocols*

RFC 5340, *OSPF for IPv6*

RFC 5642, *Dynamic Hostname Exchange Mechanism for OSPF*

RFC 5709, *OSPFv2 HMAC-SHA Cryptographic Authentication*

RFC 5838, *Support of Address Families in OSPFv3*

RFC 6549, *OSPFv2 Multi-Instance Extensions*

RFC 6987, *OSPF Stub Router Advertisement*

RFC 7684, *OSPFv2 Prefix/Link Attribute Advertisement*

RFC 7770, *Extensions to OSPF for Advertising Optional Router Capabilities*

RFC 8362, *OSPFv3 Link State Advertisement (LSA) Extensibility*

RFC 8920, *OSPF Application-Specific Link Attributes*

OpenFlow

TS-007 Version 1.3.1, *OpenFlow Switch Specification* (OpenFlow-hybrid switches)

Path Computation Element Protocol (PCEP)

draft-alvarez-pce-path-profiles-04, *PCE Path Profiles*

draft-dhs-spring-pce-sr-p2mp-policy-00, *PCEP extensions for p2mp sr policy*

draft-ietf-pce-segment-routing-08, *PCEP Extensions for Segment Routing*

RFC 5440, *Path Computation Element (PCE) Communication Protocol (PCEP)*

RFC 8281, *PCEP Extensions for PCE-initiated LSP Setup in a Stateful PCE Model*

RFC 8321, *Path Computation Element Communication Protocol (PCEP) Extensions for Stateful PCE*

Point-to-Point Protocol (PPP)

RFC 1332, *The PPP Internet Protocol Control Protocol (IPCP)*

RFC 1377, *The PPP OSI Network Layer Control Protocol (OSINLCP)*

RFC 1661, *The Point-to-Point Protocol (PPP)*

RFC 1662, *PPP in HDLC-like Framing*

RFC 1877, *PPP Internet Protocol Control Protocol Extensions for Name Server Addresses*

RFC 1989, *PPP Link Quality Monitoring*

RFC 1990, *The PPP Multilink Protocol (MP)*

RFC 1994, *PPP Challenge Handshake Authentication Protocol (CHAP)*

RFC 2153, *PPP Vendor Extensions*

RFC 2516, *A Method for Transmitting PPP Over Ethernet (PPPoE)*

RFC 2615, *PPP over SONET/SDH*

RFC 2686, *The Multi-Class Extension to Multi-Link PPP*

RFC 2878, *PPP Bridging Control Protocol (BCP)*

RFC 4638, *Accommodating a Maximum Transit Unit/Maximum Receive Unit (MTU/MRU) Greater Than 1492 in the Point-to-Point Protocol over Ethernet (PPPoE)*

RFC 5072, *IP Version 6 over PPP*

Policy Management and Credit Control

3GPP TS 29.212 Release 11, *Policy and Charging Control (PCC); Reference points* (Gx support as it applies to wireline environment (BNG))

RFC 4006, *Diameter Credit-Control Application*

RFC 6733, *Diameter Base Protocol*

Pseudowire

draft-ietf-l2vpn-vpws-iw-oam-04, *OAM Procedures for VPWS Interworking*

MFA Forum 9.0.0, *The Use of Virtual trunks for ATM/MPLS Control Plane Interworking*

MFA Forum 12.0.0, *Multiservice Interworking - Ethernet over MPLS*

MFA Forum 13.0.0, *Fault Management for Multiservice Interworking v1.0*

MFA Forum 16.0.0, *Multiservice Interworking - IP over MPLS*

RFC 3916, *Requirements for Pseudo-Wire Emulation Edge-to-Edge (PWE3)*

RFC 3985, *Pseudo Wire Emulation Edge-to-Edge (PWE3)*

RFC 4385, *Pseudo Wire Emulation Edge-to-Edge (PWE3) Control Word for Use over an MPLS PSN*

RFC 4446, *IANA Allocations for Pseudowire Edge to Edge Emulation (PWE3)*

RFC 4447, *Pseudowire Setup and Maintenance Using the Label Distribution Protocol (LDP)*

RFC 4448, *Encapsulation Methods for Transport of Ethernet over MPLS Networks*

RFC 4619, *Encapsulation Methods for Transport of Frame Relay over Multiprotocol Label Switching (MPLS) Networks*

RFC 4717, *Encapsulation Methods for Transport Asynchronous Transfer Mode (ATM) over MPLS Networks*

RFC 4816, *Pseudowire Emulation Edge-to-Edge (PWE3) Asynchronous Transfer Mode (ATM) Transparent Cell Transport Service*

RFC 5085, *Pseudowire Virtual Circuit Connectivity Verification (VCCV): A Control Channel for Pseudowires*

RFC 5659, *An Architecture for Multi-Segment Pseudowire Emulation Edge-to-Edge*

RFC 5885, *Bidirectional Forwarding Detection (BFD) for the Pseudowire Virtual Circuit Connectivity Verification (VCCV)*

RFC 6073, *Segmented Pseudowire*

RFC 6310, *Pseudowire (PW) Operations, Administration, and Maintenance (OAM) Message Mapping*

RFC 6391, *Flow-Aware Transport of Pseudowires over an MPLS Packet Switched Network*

RFC 6575, *Address Resolution Protocol (ARP) Mediation for IP Interworking of Layer 2 VPNs*

RFC 6718, *Pseudowire Redundancy*

RFC 6829, *Label Switched Path (LSP) Ping for Pseudowire Forwarding Equivalence Classes (FECs) Advertised over IPv6*

RFC 6870, *Pseudowire Preferential Forwarding Status bit*

RFC 7023, *MPLS and Ethernet Operations, Administration, and Maintenance (OAM) Interworking*

RFC 7267, *Dynamic Placement of Multi-Segment Pseudowires*

Quality of Service (QoS)

RFC 2430, *A Provider Architecture for Differentiated Services and Traffic Engineering (PASTE)*

RFC 2474, *Definition of the Differentiated Services Field (DS Field) in the IPv4 and IPv6 Headers*

RFC 2598, *An Expedited Forwarding PHB*

RFC 3140, *Per Hop Behavior Identification Codes*

RFC 3260, *New Terminology and Clarifications for Diffserv*

Remote Authentication Dial In User Service (RADIUS)

RFC 2865, *Remote Authentication Dial In User Service (RADIUS)*

RFC 2866, *RADIUS Accounting*

RFC 2867, *RADIUS Accounting Modifications for Tunnel Protocol Support*

RFC 2868, *RADIUS Attributes for Tunnel Protocol Support*

RFC 2869, *RADIUS Extensions*

RFC 3162, *RADIUS and IPv6*

RFC 4818, *RADIUS Delegated-IPv6-Prefix Attribute*

RFC 5176, *Dynamic Authorization Extensions to RADIUS*

RFC 6911, *RADIUS attributes for IPv6 Access Networks*

RFC 6929, *Remote Authentication Dial-In User Service (RADIUS) Protocol Extensions*

Resource Reservation Protocol — Traffic Engineering (RSVP-TE)

draft-newton-mpls-te-dynamic-overbooking-00, *A Diffserv-TE Implementation Model to dynamically change booking factors during failure events*

RFC 2702, *Requirements for Traffic Engineering over MPLS*

RFC 2747, *RSVP Cryptographic Authentication*

RFC 2961, *RSVP Refresh Overhead Reduction Extensions*

RFC 3097, *RSVP Cryptographic Authentication -- Updated Message Type Value*

RFC 3209, *RSVP-TE: Extensions to RSVP for LSP Tunnels*

RFC 3473, *Generalized Multi-Protocol Label Switching (GMPLS) Signaling Resource ReserVation Protocol-Traffic Engineering (RSVP-TE) Extensions* (IF_ID RSVP_HOP object with unnumbered interfaces and RSVP-TE graceful restart helper procedures)

RFC 3477, *Signalling Unnumbered Links in Resource ReSerVation Protocol - Traffic Engineering (RSVP-TE)*

RFC 3564, *Requirements for Support of Differentiated Services-aware MPLS Traffic Engineering*

RFC 3906, *Calculating Interior Gateway Protocol (IGP) Routes Over Traffic Engineering Tunnels*

RFC 4090, *Fast Reroute Extensions to RSVP-TE for LSP Tunnels*

RFC 4124, *Protocol Extensions for Support of Diffserv-aware MPLS Traffic Engineering*

RFC 4125, *Maximum Allocation Bandwidth Constraints Model for Diffserv-aware MPLS Traffic Engineering*

RFC 4127, *Russian Dolls Bandwidth Constraints Model for Diffserv-aware MPLS Traffic Engineering*

RFC 4561, *Definition of a Record Route Object (RRO) Node-Id Sub-Object*

RFC 4875, *Extensions to Resource Reservation Protocol - Traffic Engineering (RSVP-TE) for Point-to-Multipoint TE Label Switched Paths (LSPs)*

RFC 4950, *ICMP Extensions for Multiprotocol Label Switching*

RFC 5151, *Inter-Domain MPLS and GMPLS Traffic Engineering -- Resource Reservation Protocol-Traffic Engineering (RSVP-TE) Extensions*

RFC 5712, *MPLS Traffic Engineering Soft Preemption*

RFC 5817, *Graceful Shutdown in MPLS and Generalized MPLS Traffic Engineering Networks*

Routing Information Protocol (RIP)

RFC 1058, *Routing Information Protocol*

RFC 2080, *RIPng for IPv6*

RFC 2082, *RIP-2 MD5 Authentication*

RFC 2453, *RIP Version 2*

Segment Routing (SR)

draft-bashandy-rtgwg-segment-routing-uloop-06, *Loop avoidance using Segment Routing*

draft-ietf-idr-bgp-ls-segment-routing-ext-16, *BGP Link-State extensions for Segment Routing*

draft-ietf-idr-bgp-ls-segment-routing-msd-09, *Signaling MSD (Maximum SID Depth) using Border Gateway Protocol Link-State*

draft-ietf-idr-segment-routing-te-policy-09, *Advertising Segment Routing Policies in BGP*

draft-ietf-isis-mpls-elc-10, *Signaling Entropy Label Capability and Entropy Readable Label Depth Using IS-IS* (advertising ELC)

draft-ietf-lsr-flex-algo-08, *IGP Flexible Algorithm*

draft-ietf-ospf-mpls-elc-12, *Signaling Entropy Label Capability and Entropy Readable Label-stack Depth Using OSPF* (advertising ELC)

draft-ietf-rtgwg-segment-routing-ti-lfa-01, *Topology Independent Fast Reroute using Segment Routing*

draft-ietf-spring-conflict-resolution-05, *Segment Routing MPLS Conflict Resolution*

draft-ietf-spring-segment-routing-policy-08, *Segment Routing Policy Architecture*

draft-ietf-teas-sr-rsvp-coexistence-rec-02, *Recommendations for RSVP-TE and Segment Routing LSP co-existence*

draft-voyer-pim-sr-p2mp-policy-02, *Segment Routing Point-to-Multipoint Policy*

draft-voyer-spring-sr-p2mp-policy-03, *SR Replication Policy for P2MP Service Delivery*

RFC 8287, *Label Switched Path (LSP) Ping/Traceroute for Segment Routing (SR) IGP-Prefix and IGP-Adjacency Segment Identifiers (SIDs) with MPLS Data Planes*

RFC 8476, *Signaling Maximum SID Depth (MSD) Using OSPF* (node MSD)

RFC 8491, *Signaling Maximum SID Depth (MSD) Using IS-IS* (node MSD)

RFC 8661, *Segment Routing MPLS Interworking with LDP*

RFC 8665, *OSPF Extensions for Segment Routing*

RFC 8666, *OSPFv3 Extensions for Segment Routing*

RFC 8667, *IS-IS Extensions for Segment Routing*

RFC 8669, *Segment Routing Prefix Segment Identifier Extensions for BGP*

Simple Network Management Protocol (SNMP)

RFC 1157, *A Simple Network Management Protocol (SNMP)*

RFC 1215, *A Convention for Defining Traps for use with the SNMP*

RFC 1901, *Introduction to Community-based SNMPv2*

RFC 3410, *Introduction and Applicability Statements for Internet Standard Management Framework*

RFC 3411, *An Architecture for Describing Simple Network Management Protocol (SNMP) Management Frameworks*

RFC 3412, *Message Processing and Dispatching for the Simple Network Management Protocol (SNMP)*

RFC 3413, *Simple Network Management Protocol (SNMP) Applications*

RFC 3414, *User-based Security Model (USM) for version 3 of the Simple Network Management Protocol (SNMPv3)*

RFC 3415, *View-based Access Control Model (VACM) for the Simple Network Management Protocol (SNMP)*

RFC 3416, *Version 2 of the Protocol Operations for the Simple Network Management Protocol (SNMP)*

RFC 3417, *Transport Mappings for the Simple Network Management Protocol (SNMP)* (SNMP over UDP over IPv4)

RFC 3584, *Coexistence between Version 1, Version 2, and Version 3 of the Internet-standard Network Management Framework*

RFC 3826, *The Advanced Encryption Standard (AES) Cipher Algorithm in the SNMP User-based Security Model*

Simple Network Management Protocol (SNMP) - Management Information Base (MIB)

draft-ieft-snmpv3-update-mib-05, *Management Information Base (MIB) for the Simple Network Management Protocol (SNMP)*

draft-ietf-isis-wg-mib-06, *Management Information Base for Intermediate System to Intermediate System (IS-IS)*

draft-ietf-mboned-msdp-mib-01, *Multicast Source Discovery protocol MIB*

draft-ietf-mpls-ldp-mib-07, *Definitions of Managed Objects for the Multiprotocol Label Switching, Label Distribution Protocol (LDP)*

draft-ietf-mpls-lsr-mib-06, *Multiprotocol Label Switching (MPLS) Label Switching Router (LSR) Management Information Base Using SMIv2*

draft-ietf-mpls-te-mib-04, *Multiprotocol Label Switching (MPLS) Traffic Engineering Management Information Base*

draft-ietf-ospf-mib-update-08, *OSPF Version 2 Management Information Base*

draft-ietf-vrrp-unified-mib-06, *Definitions of Managed Objects for the VRRP over IPv4 and IPv6* (IPv6)

ianaaddressfamilynumbers-mib, *IANA-ADDRESS-FAMILY-NUMBERS-MIB*

ianagmplstc-mib, *IANA-GMPLS-TC-MIB*

ianaiftype-mib, *IANAifType-MIB*

ianaiprouteprotocol-mib, *IANA-RTPROTO-MIB*

IEEE8021-CFM-MIB, *IEEE P802.1ag(TM) CFM MIB*

IEEE8021-PAE-MIB, *IEEE 802.1X MIB*

IEEE8023-LAG-MIB, *IEEE 802.3ad MIB*

LLDP-MIB, *IEEE P802.1AB(TM) LLDP MIB*

RFC 1212, *Concise MIB Definitions*

RFC 1213, *Management Information Base for Network Management of TCP/IP-based Internets: MIB-II*

RFC 1724, *RIP Version 2 MIB Extension*

RFC 2021, *Remote Network Monitoring Management Information Base Version 2 using SMIv2*

RFC 2115, *Management Information Base for Frame Relay DTEs Using SMIv2*

RFC 2206, *RSVP Management Information Base using SMIv2*

RFC 2213, *Integrated Services Management Information Base using SMIv2*

RFC 2494, *Definitions of Managed Objects for the DS0 and DS0 Bundle Interface Type*

RFC 2514, *Definitions of Textual Conventions and OBJECT-IDENTITIES for ATM Management*

RFC 2515, *Definitions of Managed Objects for ATM Management*

RFC 2578, *Structure of Management Information Version 2 (SMIv2)*

RFC 2579, *Textual Conventions for SMIv2*

RFC 2580, *Conformance Statements for SMIv2*

RFC 2787, *Definitions of Managed Objects for the Virtual Router Redundancy Protocol*

RFC 2819, *Remote Network Monitoring Management Information Base*

RFC 2856, *Textual Conventions for Additional High Capacity Data Types*

RFC 2863, *The Interfaces Group MIB*

RFC 2864, *The Inverted Stack Table Extension to the Interfaces Group MIB*

RFC 2933, *Internet Group Management Protocol MIB*

RFC 3014, *Notification Log MIB*

RFC 3165, *Definitions of Managed Objects for the Delegation of Management Scripts*

RFC 3231, *Definitions of Managed Objects for Scheduling Management Operations*

RFC 3273, *Remote Network Monitoring Management Information Base for High Capacity Networks*

RFC 3419, *Textual Conventions for Transport Addresses*

RFC 3498, *Definitions of Managed Objects for Synchronous Optical Network (SONET) Linear Automatic Protection Switching (APS) Architectures*

RFC 3592, *Definitions of Managed Objects for the Synchronous Optical Network/ Synchronous Digital Hierarchy (SONET/SDH) Interface Type*

RFC 3593, *Textual Conventions for MIB Modules Using Performance History Based on 15 Minute Intervals*

RFC 3635, *Definitions of Managed Objects for the Ethernet-like Interface Types*

RFC 3637, *Definitions of Managed Objects for the Ethernet WAN Interface Sublayer*

RFC 3877, *Alarm Management Information Base (MIB)*

RFC 3895, *Definitions of Managed Objects for the DS1, E1, DS2, and E2 Interface Types*

RFC 3896, *Definitions of Managed Objects for the DS3/E3 Interface Type*

RFC 4001, *Textual Conventions for Internet Network Addresses*

RFC 4022, *Management Information Base for the Transmission Control Protocol (TCP)*

RFC 4113, *Management Information Base for the User Datagram Protocol (UDP)*

RFC 4220, *Traffic Engineering Link Management Information Base*

RFC 4273, *Definitions of Managed Objects for BGP-4*

RFC 4292, *IP Forwarding Table MIB*

RFC 4293, *Management Information Base for the Internet Protocol (IP)*

RFC 4631, *Link Management Protocol (LMP) Management Information Base (MIB)*

RFC 4878, *Definitions and Managed Objects for Operations, Administration, and Maintenance (OAM) Functions on Ethernet-Like Interfaces*

RFC 7420, *Path Computation Element Communication Protocol (PCEP) Management Information Base (MIB) Module*

SFLOW-MIB Version 1.3 (Draft 5), *sFlow MIB*

Timing

GR-1244-CORE Issue 3, *Clocks for the Synchronized Network: Common Generic Criteria*

GR-253-CORE Issue 3, *SONET Transport Systems: Common Generic Criteria*

IEEE 1588-2008, *IEEE Standard for a Precision Clock Synchronization Protocol for Networked Measurement and Control Systems*

ITU-T G.781, *Synchronization layer functions*

ITU-T G.813, *Timing characteristics of SDH equipment slave clocks (SEC)*

ITU-T G.8261, *Timing and synchronization aspects in packet networks*

ITU-T G.8262, *Timing characteristics of synchronous Ethernet equipment slave clock (EEC)*

ITU-T G.8264, *Distribution of timing information through packet networks*

ITU-T G.8265.1, *Precision time protocol telecom profile for frequency synchronization*

ITU-T G.8275.1, *Precision time protocol telecom profile for phase/time synchronization with full timing support from the network*

RFC 3339, *Date and Time on the Internet: Timestamps*

RFC 5905, *Network Time Protocol Version 4: Protocol and Algorithms Specification*

Two-Way Active Measurement Protocol (TWAMP)

RFC 5357, *A Two-Way Active Measurement Protocol (TWAMP)* (server, unauthenticated mode)

RFC 5938, *Individual Session Control Feature for the Two-Way Active Measurement Protocol (TWAMP)*

RFC 6038, *Two-Way Active Measurement Protocol (TWAMP) Reflect Octets and Symmetrical Size Features*

RFC 8545, *Well-Known Port Assignments for the One-Way Active Measurement Protocol (OWAMP) and the Two-Way Active Measurement Protocol (TWAMP)* (TWAMP)

RFC 8762, *Simple Two-Way Active Measurement Protocol* (Unauthenticated)

Virtual Private LAN Service (VPLS)

RFC 4761, *Virtual Private LAN Service (VPLS) Using BGP for Auto-Discovery and Signaling*

RFC 4762, *Virtual Private LAN Service (VPLS) Using Label Distribution Protocol (LDP) Signaling*

RFC 5501, *Requirements for Multicast Support in Virtual Private LAN Services*

RFC 6074, *Provisioning, Auto-Discovery, and Signaling in Layer 2 Virtual Private Networks (L2VPNs)*

RFC 7041, *Extensions to the Virtual Private LAN Service (VPLS) Provider Edge (PE) Model for Provider Backbone Bridging*

RFC 7117, *Multicast in Virtual Private LAN Service (VPLS)*

Voice and Video

DVB BlueBook A86, *Transport of MPEG-2 TS Based DVB Services over IP Based Networks*

ETSI TS 101 329-5 Annex E, *QoS Measurement for VoIP - Method for determining an Equipment Impairment Factor using Passive Monitoring*

ITU-T G.1020 Appendix I, *Performance Parameter Definitions for Quality of Speech and other Voiceband Applications Utilizing IP Networks - Mean Absolute Packet Delay Variation & Markov Models*

ITU-T G.107, *The E Model - A computational model for use in planning*

ITU-T P.564, *Conformance testing for voice over IP transmission quality assessment models*

RFC 3550, *RTP: A Transport Protocol for Real-Time Applications* (Appendix A.8)

RFC 4585, *Extended RTP Profile for Real-time Transport Control Protocol (RTCP)-Based Feedback (RTP/AVPF)*

RFC 4588, *RTP Retransmission Payload Format*

Wireless Local Area Network (WLAN) Gateway

3GPP TS 23.402, *Architecture enhancements for non-3GPP accesses* (S2a roaming based on GPRS)

Yet Another Next Generation (YANG)

RFC 6991, *Common YANG Data Types*

RFC 7950, *The YANG 1.1 Data Modeling Language*

RFC 7951, *JSON Encoding of Data Modeled with YANG*

Yet Another Next Generation (YANG) - OpenConfig Modules

openconfig-aaa.yang Version 0.4.0, *OpenConfig AAA Module*

openconfig-aaa-radius.yang Version 0.3.0, *OpenConfig AAA RADIUS Module*

openconfig-aaa-tacacs.yang Version 0.3.0, *OpenConfig AAA TACACS+ Module*

openconfig-acl.yang Version 1.0.0, *OpenConfig ACL Module*

openconfig-bfd.yang Version 0.1.0, *OpenConfig BFD Module*

openconfig-bgp.yang Version 3.0.1, *OpenConfig BGP Module*

openconfig-bgp-common.yang Version 3.0.1, *OpenConfig BGP Common Module*

openconfig-bgp-common-multiprotocol.yang Version 3.0.1, *OpenConfig BGP Common Multiprotocol Module*

openconfig-bgp-common-structure.yang Version 3.0.1, *OpenConfig BGP Common Structure Module*

openconfig-bgp-global.yang Version 3.0.1, *OpenConfig BGP Global Module*

openconfig-bgp-neighbor.yang Version 3.0.1, *OpenConfig BGP Neighbor Module*

openconfig-bgp-peer-group.yang Version 3.0.1, *OpenConfig BGP Peer Group Module*

openconfig-bgp-policy.yang Version 4.0.1, *OpenConfig BGP Policy Module*

openconfig-if-aggregate.yang Version 2.0.0, *OpenConfig Interfaces Aggregated Module*

openconfig-if-ethernet.yang Version 2.0.0, *OpenConfig Interfaces Ethernet Module*

openconfig-if-ip.yang Version 2.0.0, *OpenConfig Interfaces IP Module*

openconfig-if-ip-ext.yang Version 2.0.0, *OpenConfig Interfaces IP Extensions Module*

openconfig-interfaces.yang Version 2.0.0, *OpenConfig Interfaces Module*

openconfig-isis.yang Version 0.3.0, *OpenConfig IS-IS Module*

openconfig-isis-policy.yang Version 0.3.0, *OpenConfig IS-IS Policy Module*

openconfig-isis-routing.yang Version 0.3.0, *OpenConfig IS-IS Routing Module*

openconfig-lacp.yang Version 1.1.0, *OpenConfig LACP Module*

openconfig-lldp.yang Version 0.1.0, *OpenConfig LLDP Module*

openconfig-local-routing.yang Version 1.0.1, *OpenConfig Local Routing Module*

openconfig-network-instance.yang Version 0.8.0, *OpenConfig Network Instance Module*

openconfig-mpls.yang Version 2.3.0, *OpenConfig MPLS Module*

openconfig-mpls-rsvp.yang Version 2.3.0, *OpenConfig MPLS RSVP Module*

openconfig-mpls-te.yang Version 2.3.0, *OpenConfig MPLS TE Module*

openconfig-packet-match.yang Version 1.0.0, *OpenConfig Packet Match Module*

openconfig-relay-agent.yang Version 0.1.0, *OpenConfig Relay Agent Module*

openconfig-routing-policy.yang Version 3.0.0, *OpenConfig Routing Policy Module*

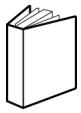
openconfig-system-logging.yang Version 0.3.1, *OpenConfig System Logging Module*

openconfig-system-terminal.yang Version 0.3.0, *OpenConfig System Terminal Module*

openconfig-telemetry.yang Version 0.5.0, *OpenConfig Telemetry Module*

openconfig-vlan.yang Version 2.0.0, *OpenConfig VLAN Module*

Customer Document and Product Support



Customer Documentation

Customer Documentation Welcome Page

Technical Support

Product Support Portal

Documentation Feedback

Customer Documentation Feedback

3HE 15829 AAAD TQZZA 01