NOKIA

7450 ETHERNET SERVICE SWITCH
7750 SERVICE ROUTER
7950 EXTENSIBLE ROUTING SYSTEM

MD-CLI USER GUIDE
RELEASE 20.2.R2

3HE 15820 AAAB TQZZA 01

Issue: 01

April 2020

Table of Contents

1 MD-CLI Overview

This guide provides information about the Model-Driven Command Line Interface (MD-CLI).

This guide is organized into functional sections and provides concepts and descriptions of the MD-CLI environment, the configuration workflow, and the syntax and command usage within the MD-CLI. It also describes how the MD-CLI interacts with the classic CLI to perform non-configuration operations.

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 outputs may differ depending on supported functionality and user configuration.

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.

1.1 Using the MD-CLI

All 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 management interface that can be used to manage Nokia SR OS routers. Some of the benefits of the MD-CLI include:

- follows the model-driven networking strategy, based on common YANG models for a structured configuration. Consistency is maintained between the MD-CLI, NETCONF, and the gRPC model-driven interfaces.
- uses the transactional configuration method which uses a candidate configuration to hold the current configuration changes before they are applied to the running configuration, and avoids configuration ordering requirements
- provides multiuser candidate configuration modes (global, exclusive, private, and read-only) that control access to the configuration, allowing a user exclusive access to the configuration such that no other configuration changes can be made

- allows the use of configuration groups with flexible templates that simplify the configuration process by applying the template instead of repeating the same configuration

For more information about NETCONF and gRPC, refer to the *7450 ESS, 7750 SR, 7950 XRS, and VSR System Management Guide*.

Table 1 describes command syntax symbols used in this guide.

Table 1 **Command Syntax Symbols**

Symbol	Description
|	A vertical bar represents an OR, indicating that only one of the parameters in the brackets or parentheses can be selected.
( )	Parentheses indicate that one of the parameters must be selected.
[ ]	Brackets indicate optional parameters.
Bold	Commands in **bold** indicate commands and keywords.
Italic	Commands in *italics* indicate that you must enter text based on the parameter.

In the following examples, **location** and **graceful-shutdown** are command names. For the **location** command, *keyword* must be one of the keywords **cf1**, **cf2**, or **cf3**. For the **graceful-shutdown** command, *boolean* must be one of the keywords **true** or **false**, although explicitly using the keyword **true** is optional.

```
location keyword
keyword - (cf1 | cf2 | cf3)

graceful-shutdown boolean
boolean - ([true] | false)
```

2 Controlling the Management Interface Configuration Mode

SR OS routers can be in different management interface configuration modes, which affects the management interfaces that can be used to configure the router. The following interfaces are available for configuration on SR OS:

- classic (default) — configuration via the classic CLI and SNMP, no model-driven interfaces are supported
- mixed — configuration via the classic CLI and model-driven interfaces: the MD-CLI, NETCONF, and gRPC/gNMI, read-only access via SNMP
- model-driven — configuration via model-driven interfaces: the MD-CLI, NETCONF, and gRPC/gNMI, read-only access via the classic CLI and SNMP

Refer to the *7450 ESS, 7750 SR, 7950 XRS, and VSR System Management Guide* for more information on management interface configuration mode features and the interactions between classic and model-driven modes.

2.1 Enabling the MD-CLI

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.

To enable the MD-CLI engine from the classic CLI, perform the following steps:

1. Set the configuration mode to model-driven and leave **cli-engine** unconfigured.

```
A:node-2>config>system>management-interface# configuration-mode model-driven
```

2. Log out and start a new CLI session to access the MD-CLI engine.

```
A:node-2>config>system>management-interface# logout
```

When a new user session begins, the MD-CLI engine is available and the MD-CLI prompt is displayed.

```
[]
A:admin@node-2#
```

When the configuration mode is changed to **model-driven**, the following applies:

- the configuration mode becomes immediately active
- access to configuration in the classic CLI is read-only (no modification)

- access to show configuration in the classic CLI is still available

2.2 Switching Between the Classic CLI and MD-CLI Engines

A single CLI command is available in both the classic CLI and MD-CLI engines to switch between the two engines in a user session. When authorized (**cli-engine** list contains both **classic-cli** and **md-cli**), the CLI engine switch command ("//", the double forward slash) can be executed from any CLI context in both engines to switch to the other CLI engine.

```
A:node-2# //
INFO: CLI #2052: Switching to the MD-CLI engine

[]
A:admin@node-2# //
INFO: CLI #2051: Switching to the classic CLI engine
A:node-2#
```

The context in which the CLI engine switch command is executed is saved when toggling between CLI engines and returns to the same context when toggling back.

```
[]
A:admin@node-2# edit-config read-only
INFO: CLI #2066: Entering read-only configuration mode

(ro)[]
A:admin@node-2# configure router

(ro)[configure router "Base"]
A:admin@node-2# //
INFO: CLI #2051: Switching to the classic CLI engine
A:node-2# configure system management-interface
A:node-2>config>system>management-interface# //
INFO: CLI #2052: Switching to the MD-CLI engine

(ro)[configure router "Base"]
A:admin@node-2# //
INFO: CLI #2051: Switching to the classic CLI engine
A:node-2>config>system>management-interface#
```

If switching engines is not authorized (when **cli-engine** is only [**classic-cli**] or [**md-cli**]), the command is rejected.

```
A:node-2# //
MINOR: CLI #2053 Switching CLI engine is not authorized
A:node-2#
```

2.2.1 Executing Classic CLI Commands from the MD-CLI Engine

When switching engines is authorized, all classic CLI engine commands can be executed from the MD-CLI engine. Entering a classic CLI engine command preceded by the "//" command executes the command in the classic CLI engine and returns immediately to the MD-CLI engine. The MD-CLI context is preserved before the switch to the classic CLI engine, and the context is restored when the session returns to the MD-CLI engine. In the following example, the classic CLI command is executed from the **configure system** context in the MD-CLI. When the session returns to the MD-CLI engine, it is returned to the same context.

```
(ex)[configure system]
A:admin@node-2# //file dir
INFO: CLI #2051: Switching to the classic CLI engine
A:admin@node-2# /file dir

Volume in drive cf3 on slot A is .

Volume in drive cf3 on slot A is formatted as FAT32

Directory of cf3:\

10/24/2019  01:04p      <DIR>          .ssh/
01/01/1980  12:00a                170 NVRAM.DAT
01/01/1980  12:00a                610 bof.cfg
10/24/2019  01:04p                317 nvsys.info
10/24/2019  01:04p                  1 restcntr.txt
               4 File(s)                   1098 bytes.
               1 Dir(s)                    2048 bytes free.
INFO: CLI #2052: Switching to the MD-CLI engine

(ex)[configure system]
A:admin@node-2#
```

It is acceptable to include a space between "//" and the CLI command. For example, **//file dir** and **// file dir** are equivalent commands.

User interactions, such as pagination, confirmation, or control characters (for example, CTRL-c to stop an ongoing command execution), are supported during CLI command execution. The CLI engine is switched back to the MD-CLI engine just before the CLI command prompt would normally appear.

Executing MD-CLI commands from the classic CLI engine works in the same way as described for executing classic CLI commands from the MD-CLI engine.

2.2.2 MD-CLI and Classic CLI Engine Interactions

The following describes MD-CLI engine interactions with the classic CLI when using the "//" command:

- uncommitted changes in the MD-CLI are kept when switching to the classic CLI
- "//" appears in the history of the CLI engine where it is executed

```
[]
A:admin@node-2# //
INFO: CLI #2051: Switching to the classic CLI engine
A:node-2# history
   1 history
A:node-2# //
INFO: CLI #2052: Switching to the MD-CLI engine
[]
A:admin@node-2# history
       1 //
[]
A:admin@node-2#
```

- "//*command*" appears in the history of both CLI engines

```
[]
A:admin@node-2# //file dir
INFO: CLI #2051: Switching to the classic CLI engine
A:node-2# /file dir

Volume in drive cf3 on slot A is .

Volume in drive cf3 on slot A is formatted as FAT32

Directory of cf3:\

06/26/2019  07:58p      <DIR>          .ssh/
01/01/1980  12:00a                170 NVRAM.DAT
01/01/1980  12:00a                610 bof.cfg
06/26/2019  07:58p                311 nvsys.info
06/26/2019  07:58p                  1 restcntr.txt
               4 File(s)                   1092 bytes.
               1 Dir(s)                    2048 bytes free.

INFO: CLI #2052: Switching to the MD-CLI engine

[]
A:admin@node-2# history
       1 //file dir

[]
A:admin@node-2# //
INFO: CLI #2052: Switching to the classic CLI engine
A:admin@node-2# history
       1 /file dir
       2 history
A:admin@node-2#
```

- command completion, ? help, and redirection are not supported for the command following the “//”
- all control characters added on the same line when entering a “//” command have an effect on the CLI engine where they are entered

```
[]
A:admin@node-2# //file dir   Press CTRL-w   # stay in the MD-CLI engine
                                            # delete word


[]
A:admin@node-2# //file dir   Press CTRL-c   # stay in the MD-CLI engine
                                            # stop current command
[]
A:admin@node-2#
```

CTRL-z is the equivalent of Enter and **exit all**. When used on a command line with “//”, CTRL-z is the equivalent of just pressing Enter. Because the originating CLI engine is no longer available, **exit all** can no longer be executed.

```
[]
A:admin@node-2# //file dir   Press CTRL-z
INFO: CLI #2051: Switching to the classic CLI engine
A:node-2# /file dir

Volume in drive cf3 on slot A is .

Volume in drive cf3 on slot A is formatted as FAT32

Directory of cf3:\

06/26/2019  07:58p      <DIR>          .ssh/
01/01/1980  12:00a                170 NVRAM.DAT
01/01/1980  12:00a                610 bof.cfg
06/26/2019  07:58p                311 nvsys.info
06/26/2019  07:58p                  1 restcntr.txt
               4 File(s)                   1092 bytes.
               1 Dir(s)                    2048 bytes free.

INFO: CLI #2052: Switching to the MD-CLI engine

[]
A:admin@node-2# history
```

A command history is maintained per CLI engine. CLI commands executed in the MD-CLI do not appear in the classic CLI history. CLI commands executed in the classic CLI do not appear in the MD-CLI history.

2.2.3 Switching Explicitly to the Classic CLI Engine

The **/!classic-cli** command is available in both the classic CLI and MD-CLI engines to explicitly switch to the classic CLI engine in a session, as long as **classic-cli** is an authorized CLI engine. If switching to the classic CLI engine is not authorized, the command is rejected. Issuing the **/!classic-cli** command in the classic CLI engine has no effect.

The **/!classic-cli** switch command can be executed from any CLI context in both engines and the context is preserved for both engines. When the command is executed, the session enters the last saved working context of the classic CLI engine.

```
A:node-2>config>system>management-interface# //
INFO: CLI #2052: Switching to the MD-CLI engine

(ex)[configure router "Base" bgp]
A:admin@node-2# /!classic-cli
INFO: CLI #2051: Switching to the classic CLI engine
A:node-2>config>system>management-interface#
```

2.2.4 Switching Explicitly to the MD-CLI Engine

The **/!md-cli** command is available in both the classic CLI and MD-CLI engines to explicitly switch to the MD-CLI engine in a session, as long as **md-cli** is an authorized CLI engine. If switching to the MD-CLI engine is not authorized, the command is rejected. Issuing the **/!md-cli** command in the MD-CLI engine has no effect.

The **/!md-cli** switch command can be executed from any CLI context in both engines and the context is preserved for both engines. When the command is executed, the session enters the last saved working context of the MD-CLI engine.

```
(ex)[configure router "Base" bgp]
A:admin@node-2# /!classic-cli
INFO: CLI #2051: Switching to the classic CLI engine
A:node-2>config>system>management-interface# /!md-cli
INFO: CLI #2052: Switching to the MD-CLI engine

(ex)[configure router "Base" bgp]
A:admin@node-2#
```

The **/!md-cli** and **/!classic-cli** commands can be useful when executing commands from a file, allowing the file to be executed in either CLI engine and ensuring the commands are run in the intended CLI engine.

3 Navigating in the MD-CLI

3.1 The MD-CLI Tree Structure

The MD-CLI tree contains the following elements from the Nokia YANG models:

- container — an element that contains other elements. In the following example, **tcp-keepalive** and **gnmi** are containers.

```
tcp-keepalive {
    admin-state disable
    idle-time 600
    interval 15
    retries 4
}
gnmi {
    admin-state enable
    auto-config-save false
}
```

- leaf — an element that does not contain any other elements and has a data type (for example, string or integer). A leaf can also be defined with no data type where the leaf takes no parameter value (that is, an empty leaf). The **bold** elements in the following example are leafs.

```
tcp-keepalive {
    admin-state disable
    idle-time 600
    interval 15
    retries 4
}
gnmi {
    admin-state enable
    auto-config-save false
}
```

- list entry — an element similar to a container with multiple instances where each list entry is identified by the values of its keys (for example, group “group-2”)

```
router "Base" {
    bgp {
        group "group-1" {
            connect-retry 600
            keepalive 33
        }
        group "group-2" {
            description "Text description for group-2"
            local-preference 8
        }
    }
}
```

- key — a unique identifier for a list entry (for example, “group-1” and “group-2”)

```
router "Base" {
    bgp {
        group "group-1" {
            connect-retry 600
            keepalive 33
        }
        group "group-2" {
            description "Text description for group-2"
            local-preference 8
        }
    }
}
```

- leaf-list — an element that contains a sequence of values of a particular data type (for example, “policy” is a leaf-list in the following example)

```
policy ["policy-a" "policy-b" "policy-c"]
```

- list — a sequence of list entries. In the preceding example, the entire set of interfaces is a list.

```
group "group-1" {
    connect-retry 600
    keepalive 33
}
group "group-2" {
    description "Text description for group-2"
    local-preference 8
}
```

- leaf-list entry — one of the values of a leaf-list. For example, “policy-a”, “policy-b”, and “policy-c” are leaf-list entries in the following example.

```
policy ["policy-a" "policy-b" "policy-c"]
```

The following terms are also used:

- keyword — an element with a name defined by SR OS; for example, enumerated values, leaf names, and container names)
- variable parameter — an element with a name defined by the user; for example, descriptions, names, integer or string leaf values)
- immutable element — an element that can only be configured in the transaction in which the parent element is created. It cannot be modified while the parent element exists.
- choice element — an element which is part of a set of mutually exclusive elements. Setting a choice element clears all configuration from the other choice elements.

In the following example, **admin-state** (leaf name), **enable** (enumerated value), and **connect-retry** (leaf name) are keywords, and “800” is a variable parameter.

```
*(ex)[configure router "Base" bgp]
A:admin@node-2# info
    admin-state enable
    connect-retry 800
```

Managing the router configuration using the MD-CLI involves accessing and configuring the appropriate elements (containers, lists, leafs, and leaf-lists).

The MD-CLI tree shows the commands and parameters (also known as elements) that are available in a hierarchical output. In the following **tree detail** command output, the bold elements are containers (or container lists) which contain leafs (or leaf-lists).

```
*[ex:configure system]
A:admin@dut-c# tree detail
+-- alarms
|   +-- admin-state <keyword>
|   +-- apply-groups <reference>
|   +-- max-cleared <number>
+-- allow-boot-license-violations <boolean>
+-- apply-groups <reference>
+-- boot-bad-exec <string>
+-- boot-good-exec <string>
+-- central-frequency-clock
|   +-- apply-groups <reference>
|   +-- bits
|   |   +-- input
|   |   |   +-- admin-state <keyword>
|   |   +-- interface-type <keyword>
|   |   +-- output
|   |   |   +-- admin-state <keyword>
|   |   |   +-- line-length <keyword>
|   |   |   +-- ql-minimum <keyword>
|   |   |   +-- source <keyword>
|   |   |   +-- squelch <boolean>
|   |   +-- ql-override <keyword>
|   |   +-- ssm-bit <number>
|   +-- ptp
|   |   +-- admin-state <keyword>
|   |   +-- ql-override <keyword>
|   +-- ql-minimum <keyword>
|   +-- ql-selection <boolean>
|   +-- ref-order
|   |   +-- first <keyword>
|   |   +-- fourth <keyword>
|   |   +-- second <keyword>
|   |   +-- third <keyword>
|   +-- ref1
|   |   +-- admin-state <keyword>
|   |   +-- ql-override <keyword>
|   |   +-- source-port <reference | connector-port>
|   +-- ref2
|   |   +-- admin-state <keyword>
|   |   +-- ql-override <keyword>

  <snip>
```

3.2 The MD-CLI Command Prompt

The MD-CLI command prompt displays on two lines. The first line contains the following information:

- **baseline status indicator**

  This indicator displays an exclamation mark (!) to indicate an out-of-date baseline when in configuration mode.
- **uncommitted changes indicator**

  This indicator displays an asterisk (*) to indicate uncommitted configuration changes when in configuration mode.
- **configuration mode reference**

  When in configuration mode, a configuration mode reference is displayed:
  - in round brackets for an explicit configuration workflow
  - prepended to the context, separated by a colon for an implicit configuration workflow

  The configuration mode reference can be one of the following:
  - pr — private mode
  - ex — exclusive mode
  - gl — global mode
  - ro — read-only mode
- **context**

  The present working context is displayed in square brackets ([]) when in operational or configuration mode.

For an explicit configuration workflow, the format of the first line is as follows:

<*baseline status indicator* > <*uncommitted changes indicator*> (<*configuration mode*>) [*context*]

Examples:

```
(ro)[]

(ex)[configure router "Base" bgp]
```

For an implicit configuration workflow, the format of the first line is as follows:

<*baseline status indicator* > <*uncommitted changes indicator*> [<*configuration mode*>:*context*]

Examples:

```
[ro:configure]

*[ex:configure]
```

The second line contains the following information:

- **CPM**

  The active CPM slot can be A or B on 7750 SR routers, and A,B,C, or D on 7950 XRS routers.
- **user**

  The user is the name of the current user for this session.
- **name**

  The name is the system name, as configured with the **configure system name** command. The system name can change dynamically during the session if it is configured to a different name.

The format of the second line is as follows:

CPM:*user@name#*

The following examples display the two-line prompt in different modes.

- prompt in operational mode

```
[]
A:admin@my-system#
```

- prompt in the operational root, with exclusive configuration mode

```
(ex)[]
A:admin@my-system#
```

- prompt in operational mode **show router bgp**

```
[show router "Base" bgp]
A:admin@my-system#
```

- prompt in exclusive configuration mode **configure router bgp**

```
(ex)[configure router "Base" bgp]
A:admin@my-system#
```

- prompt in exclusive configuration mode **configure router bgp** with uncommitted changes

```
*(ex)[configure router "Base" bgp]
A:admin@my-system#
```

- implicit configuration workflow prompt for a session in private configuration mode, with present working context of **configure router bgp** with uncommitted changes in the private candidate datastore, and the baseline datastore out-of-date

```
!*[pr:configure router "Base" bgp]
A:admin@my-system#
```

3.3 Environment Commands

The environment configuration for the MD-CLI is available in both the classic CLI and in the MD-CLI, but the configuration applies only to MD-CLI sessions.

In the MD-CLI, environment variables are found under the context **configure system management-interface cli md-cli**:

```
[gl:configure system management-interface cli md-cli environment]
A:admin@node-2# ?
 command-completion   + Enter the command-completion context
 console              + Enter the console context
 message-severity-    + Enter the message-severity-level context
  level
 more                 - Paging control of the output text
 progress-indicator   + Enter the progress-indicator context
 prompt               + Enter the prompt context
 time-display         - Time to display timestamp before prompt
 time-format          - Format in which time is to be displayed
```

In the classic CLI, the configuration context is as follows:

```
*A:node-2>config>system>management-interface>cli>md>env#?
      command-comple* + Configure keystrokes to trigger command completion
      console         + Configure console parameters
      message-severi* + Configure messages severity
 [no] more            - Configure paging of the output text
      progress-indic* + Settings for progress indicator during command
                        execution
      prompt          + Configure content of displayed prompt
      time-display    - Specify whether timestamp should be displayed in UTC or
                        local time
      time-format     - Specify format in which should be displayed
```

Changes made to the environment configuration apply only to new sessions and do not affect current sessions.

See the *MD-CLI Command Reference Guide* for information about the **environment** commands in the MD-CLI.

3.3.1 Customizing Per-Session Environment Settings

The environment can be customized for all sessions in the configuration under the **configure system management-interface cli md-cli environment** context, or per session using the **environment** command. When a new MD-CLI session is started, the per-session environment configuration is copied from the global environment configuration. Changes made to the global environment configuration after the session begins apply only to new sessions and do not affect current sessions. Changes made to the environment parameters for a session apply only for that session.

The per-session environment is accessed by entering **environment** at the operational root or with **/environment** from any other mode or context. Changes made in the per-session environment are immediate.

The **info** command displays the difference between the per-session environment and the configured global environment parameters. Therefore, for a new MD-CLI session, the **info** command has no output, as the per-session environment is the same as the global environment. The **info detail** command displays the current values in the global environment for all parameters.

3.3.2 Customizing the Session Prompt

3.3.2.1 Customizing the Uncommitted Changes Indicator

As the default setting of the environment configuration, the uncommitted changes indicator is displayed as part of the command prompt. This setting can be modified per session or it can be changed for all MD-CLI sessions by changing the environment configuration.

The **uncommitted-changes-indicator** command under the **environment prompt** context suppresses or displays the change indicator for an MD-CLI session. Environment changes are applied immediately and are lost when the session disconnects.

```
*[environment prompt]
A:admin@node-2# uncommitted-changes-indicator false

[environment prompt]
A:admin@node-2#

[environment prompt]
A:admin@node-2# uncommitted-changes-indicator true
```

```
*[environment prompt]
A:admin@node-2#
```

3.3.2.2 Customizing the Line Preceding the Command Prompt

By default, a blank line precedes the command prompt. This setting can be modified for each MD-CLI session.

The **newline** command under the **environment prompt** context suppresses or displays a new line before the prompt.

```
[]
A:admin@node-2# environment prompt

[environment prompt]
A:admin@node-2# newline false
[environment prompt]
A:admin@node-2# newline true

[environment prompt]
A:admin@node-2#
```

3.3.2.3 Customizing the Context Information in the Command Prompt

By default, the context is displayed in the command prompt. This setting can be modified for each MD-CLI session.

The **context** command under the **environment prompt** context suppresses or displays the current context.

```
[environment prompt]
A:admin@node-2# context false

[]
A:admin@node-2# context true

[environment prompt]
A:admin@node-2#
```

3.3.2.4 Customizing the Timestamp

By default, the timestamp is not displayed before the command prompt. This setting can be modified for each MD-CLI session.

The **timestamp** command under the **environment prompt** context suppresses or displays the timestamp.

```
[environment prompt]
A:admin@node-2# timestamp true

SUN 10 JUNE 2018 23:09:51 UTC
[environment prompt]
A:admin@node-2# timestamp false

[environment prompt]
A:admin@node-2#
```

The **environment time-display** command configures the time zone display to UTC or local time (as configured in **configure system time**).

```
[environment]
A:admin@node-2# time-display ?

 time-display <keyword>
 <keyword> - (local|utc)
 Default   - local

    Time to display timestamp before prompt
```

3.3.3 Customizing the Progress Indicator

The progress indicator appears on the line immediately following the command and disappears when the MD-CLI command completes or when output is available to display. The indicator is a display of dynamically changing dots.

```
(ex)[configure]
A:admin@node-2# compare
...                    # progress indicator displays here as dots
```

The delay interval can be configured with the **delay** command or the indicator can be disabled with the **admin-state disable** command under the **environment progress-indicator** context. For example, the user can disable the progress indicator for logged sessions.

```
[environment progress-indicator]
A:admin@node-2# ?

 admin-state          - Administrative state of the progress indicator
```

```
 delay                - Delay before progress indicator is displayed
 type                 - Progress indicator output style
```

3.3.4 Customizing the Pagination Setting

The **environment more** command enables pagination when configured to **true** and disables pagination when configured to **false**. With pagination enabled, the display output can be paused and continued, based on the “Press Q to quit, Enter to print next line or any other key to print next page” message at the bottom of the screen.

```
[]
A:admin@node-2# environment more true


[]
A:admin@node-2# show system security profile user-profile-name administrative
===============================================================================
User Profile
===============================================================================

User Profile          : administrative
Def. Action           : permit-all
LI                    : no
Netconf Kill
Authorization         : no
Netconf Lock
Authorization         : no
gRPC gNMI
Capabilities RPC
Authorization         : yes
gRPC gNMI Get RPC
Authorization         : yes
gRPC gNMI Set RPC
Authorization         : yes
gRPC gNMI Subscribe
RPC Authorization     : yes
-------------------------------------------------------------------------------
Cli Session Group     : no
Press Q to quit, Enter to print next line or any other key to print next page.
```

The pagination setting can be overridden by using **| no-more** for a single command. As with pagination disabled, the output is displayed completely without any prompts to continue.

```
[]
A:admin@node-2# show system security profile user-profile-name administrative | no-
more
```

3.3.5 Customizing the Console Settings

The default size for a console window is 24 lines long by 80 characters wide. The **environment console** command can be used to change these settings.

```
(ex)[environment]
A:admin@node-2# console ?

 length               - Number of lines displayed on the screen
 width                - Number of columns displayed on the screen
```

3.3.6 Customizing the Message Level Security Settings

The INFO: CLI messages are displayed by default. The **environment message-security-level** command suppresses the INFO messages by changing the setting to **warning**.

```
[environment message-severity-level]
A:admin@node-2# cli ?

 cli <keyword>
 <keyword> - (warning|info)
 Default   - info

    Message severity threshold for CLI messages
```

Following are examples of INFO: CLI messages that are suppressed when the setting is changed to **warning**:

```
INFO: CLI #2051: Switching to the classic CLI engine
INFO: CLI #2052: Switching to the MD-CLI engine
INFO: CLI #2054: Entering global configuration mode
INFO: CLI #2056: Exiting global configuration mode
INFO: CLI #2055: Uncommitted changes are present in the candidate configuration
INFO: CLI #2057: Uncommitted changes are kept in the candidate configuration
```

3.3.7 Preventing Changes to Environment Settings

The environment datastore is subject to AAA command authorization. A user can be prevented from modifying the global environment settings or the per-session environment settings, or both.

In the following configuration output, **entry 113** blocks user “tstuser” from modifying the global environment settings. In addition, **entry 114** prevents the user from changing the per-session environment settings.

```
(ro)[configure system security aaa local-profiles profile "tstuser"]
A:admin@node-2# info
    default-action permit-all
    entry 113 {
        action deny
        match "configure system management-interface cli md-cli environment"
    }
    entry 114 {
        action deny
        match "environment"
    }


(ex)[configure system management-interface cli md-cli environment]
A:tstuser@node-2# prompt timestamp
MINOR: MGMT_CORE #2020: Permission denied

(ex)[configure system management-interface cli md-cli environment]
A:tstuser@node-2# /environment
MINOR: MGMT_CORE #2020: Permission denied

(ex)[configure system management-interface cli md-cli environment]
A:tstuser@node-2#
```

3.4 Using Online Help

A short help description is displayed immediately when the question mark (?) is entered (without needing to press Enter). The following displays help from the operational root level.

```
[]
A:admin@node-2# ?

 admin                + Enter the administrative context for system operations
 clear                + Clear statistics or reset operational state
 configure            + Enter the configuration context
 environment          + Enter the environment configuration context
 li                   + Enter the lawful intercept context
 show                 + Show operational information
 tools                + Enter the tools context for troubleshooting and
                        debugging

 Global commands:
 back                 - Move back one or more levels
 delete               - Delete an element from the candidate datastore
 edit-config          - Enter a candidate configuration mode
 enable               - Enable administrative mode
 exec                 - Execute commands from a file
 exit                 - Return to the previous working context or to the
                        operational root
 history              - Show the most recently entered commands
 logout               - Exit the CLI session
 oam                  - Enter the oam context
 ping                 - Trigger ping of address or dns-name
```

```
 pwc                  - Show the present working context
 top                  - Move to the top level of the context
 traceroute           - Determine the route to a destination address
 tree                 - Show the command tree under the present working context
```

The ? help is context-sensitive. The following ? help output lists additional commands available in exclusive configuration mode.

```
(ex)[]
A:admin@node-2# ?

 admin                + Enter the administrative context for system operations
 clear                + Clear statistics or reset operational state
 configure            + Enter the configuration context
 environment          + Enter the environment configuration context
 show                 + Show operational information
 tools                + Enter the tools context for troubleshooting and
                        debugging

 Global commands:
 back                 - Move back one or more levels
 delete               - Delete an element from the candidate datastore
 edit-config          - Enter a candidate configuration mode
 enable               - Enable administrative mode
 exec                 - Execute commands from a file
 exit                 - Return to the previous working context or to the
                        operational root
 history              - Show the most recently entered commands
 insert               - Insert an element into a user-ordered list
 logout               - Exit the CLI session
 oam                  - Enter the oam context
 ping                 - Trigger ping of address or dns-name
 pwc                  - Show the present working context
 quit-config          - Leave the candidate configuration mode
 top                  - Move to the top level of the context
 traceroute           - Determine the route to a destination address
 tree                 - Show the command tree under the present working context

 Configuration commands:
 commit               - Commit changes to the running datastore
 compare              - Show changes between datastores
 discard              - Discard changes in the candidate datastore
 info                 - Show the configuration from the present working context
 update               - Update candidate baseline
 validate             - Validate changes in the candidate datastore
```

The help results may depend on the cursor position. The following example shows the **router** command syntax, followed by available commands after entering the **router** context.

```
[ex:configure]
A:admin@node-2# router?

 router [[router-name] <string>]

 [router-name]         - The administrative name for this virtual router. The
```

```
                          router name must be unique among all virtual routers
                          in the system.

 admin-state           - Administrative state of the router instance
 aggregates            + Enter the aggregates context
 allow-icmp-redirect   - Allow ICMP redirects on the management interface
 allow-icmp6-redirect  - Allow IPv6 ICMP redirects on the management interface
 apply-groups          - Apply a configuration group at this level
 autonomous-system     - AS number advertised to peers for this router
 bfd                   + Enter the bfd context
 bgp                   + Enter the bgp context

 <snip>
```

In the following ? output, similar information is shown, with more details provided for configuring the **router** command, including the allowable string length and default value for the command.

```
[ex:configure]
A:admin@node-2# router ?

 router [[router-name] <string>]

 [[router-name] <string>]
 <string>  - <1..64 characters>
 Default   - "Base"

    The administrative name for this virtual router. The router name must be
    unique among all virtual routers in the system.

 admin-state           - Administrative state of the router instance
 aggregates            + Enter the aggregates context
 allow-icmp-redirect   - Allow ICMP redirects on the management interface
 allow-icmp6-redirect  - Allow IPv6 ICMP redirects on the management interface
 apply-groups          - Apply a configuration group at this level
 autonomous-system     - AS number advertised to peers for this router
 bfd                   + Enter the bfd context
 bgp                   + Enter the bgp context

 <snip>
```

3.4.1 Indicators in the Online Help

Table 2 describes the meaning of the indicators displayed in the online help.

Table 2 **Root Commands**

Symbol	Description
+	Indicates a container or list

Table 2 **Root Commands (Continued)**

Symbol	Description
-	Indicates a leaf, a leaf-list, a list or container with no leafs, or a global command (if in the operational root)
^	Indicates a mandatory element (an element that must be configured before the configuration is considered valid)

In the following help display example, the containers are **eth-cfm**, **domain**, and **association**. The leafs are **dns**, **format**, **level**, **mac**, **md-index**, and **name**, while **level** is also a mandatory element.

```
[ex:configure]
A:admin@node-2# eth-cfm ?

 eth-cfm

 domain               + Enter the domain list instance

[ex:configure]
A:admin@node-2# eth-cfm

[ex:configure eth-cfm]
A:admin@node-2# domain ?

 [md-admin-name] <string>
 <string> - <1..64 characters>

    Unique domain name


[ex:configure eth-cfm]
A:admin@node-2# domain dom-name ?

domain

 Immutable fields      - level, dns, mac, name, format, md-index

 association           + Enter the association list instance
 dns                   - Domain name like text string derived from a DNS name
 format                - Maintenance domain name not to be provided
 level                 ^ Maintenance Domain Level (MD Level)
 mac                   - Maintenance domain MAC name
 md-index              - The index of the Maintenance Domain (MD)
 name                  - Maintenance domain name as an ASCII string
```

3.4.1.1 Descriptions and Format Guidelines for Leafs and Leaf-lists

When online help is entered for a leaf or leaf-list, a short description of the element is displayed after the element type. The valid input values for the element are also listed, as shown in the following examples.

The **description** string for the VPRN service can have a length of 1 to 80 characters:

```
*[ex:configure service vprn "5"]
A:admin@node-2# description ?

 description <string>
 <string> - <1..80 characters>

    Text description
```

The ? help for the **autonomous-system** parameter lists the valid number range, followed by a short description of the parameter:

```
*[ex:configure service vprn "5"]
A:admin@node-2# autonomous-system ?

 autonomous-system <number>
 <number> - <1..4294967295>

    AS number advertised to peers for this router
```

A parameter value may have a unit type associated with it, as shown in the following example of the **ingress-buffer-allocation** parameter:

```
*[ex:configure qos sap-ingress "sap-pname" policer 6]
A:admin@node-2# mbs ?

 mbs (<number> | <keyword>)
 <number>  - <0..16777216> - bytes
 <keyword> - auto          - bytes
 Default   - auto

    High priority for the violate threshold of PIR leaky bucket of this policer
```

This example shows a parameter that is a reference to another parameter. The **owner** command refers to the script policy name that is configured through the **configure system script-control script-policy** context. The name is a string of 1 to 32 characters.

```
*[ex:configure log event-handling handler "h-name" entry 5]
A:admin@node-2# script-policy owner ?

 owner <reference>
 <reference> - <1..32            - configure system script-control script-
               characters>         policy <owner>
```

```
    Name combined with value of owner to launch the script policy
```

3.4.1.2 Immutable Elements

An immutable element can only be configured in the transaction in which the parent element is created. It cannot be modified while the parent element exists. Any modification to an immutable element in model-driven interfaces causes SR OS to automatically delete the parent element and recreate it with the new value for the immutable element.

Immutable elements are identified in the online help, as seen in the following examples:

```
[ex:configure eth-cfm]
A:admin@node-2# domain dom-name ?

domain

 Immutable fields      - level, dns, mac, name, format, md-index

 association           + Enter the association list instance
 dns                   - Domain name like text string derived from a DNS name
 format                - Maintenance domain name not to be provided
 level                 ^ Maintenance Domain Level (MD Level)
 mac                   - Maintenance domain MAC name
 md-index              - The index of the Maintenance Domain (MD)
 name                  - Maintenance domain name as an ASCII string


[ex:configure eth-cfm]
A:admin@node-2# domain dom-name level ?

 level <number>
 <number>  - <0..7>

 'level' is: mandatory, immutable

    Maintenance Domain Level (MD Level)

    Warning: Modifying this element will cause the parent element
    'configure eth-cfm domain "dom-name"' to be recreated automatically for the
    new value to take effect.


<snip>

[ex:configure router "Base" isis 0]
A:admin@node-2# lsp-lifetime ?

 lsp-lifetime <number>
 <number> - <350..65535> - seconds
 Default  - 1200
```

```
    Amount of time during which an LSP is considered valid

    Note: Modifying this element requires
    'configure router "Base" isis 0 admin-state' to be toggled manually for the
    new value to take effect.
```

Immutable elements also exist in the classic CLI. They are parameters that are on the command line with the **create** keyword. For example, in the following classic CLI command, all the parameters shown on the command line are immutable. These parameters cannot be changed without deleting and recreating the service.

```
[]
A:admin@node-2# //
INFO: CLI #2051: Switching to the classic CLI engine
A:node-2# configure service ies
  - ies <service-id> [customer <customer-id>] [create] [vpn <vpn-id>] [name
    <name>]
  - no ies <service-id>
```

3.4.1.3 Optional Indicators in the Online Help

The following help display is an example of optional indicators.

The square brackets ([]) around **slot-number** indicate that the **slot-number** keyword is optional when entering the command.

```
[ex:configure]
A:admin@node-2# card ?

 [slot-number] <number>
 <number> - <1..20>

    IOM slot within a chassis
```

The **card** context can be entered as:

```
[ex:configure]
A:admin@node-2# card slot-number 5
```

or

```
[ex:configure]
A:admin@node-2# card 5
```

Angle brackets (<>) indicate a variable name and the pipe (|) indicates a choice. For the **sub-group** command, a number in the range of 1 to 8 can be entered, or one of the keywords **auto-iom** or **auto-mda**.

```
[ex:configure lag 8 port 1/1/1]
```

```
A:admin@node-2# sub-group ?

 sub-group (<number> | <keyword>)
 <number>  - <1..8>
 <keyword> - (auto-iom|auto-mda)
 Default   - 1

 'sub-group' is: immutable

    Subgroup of the port in the LAG

    Warning: Modifying this element will cause the parent element
    'configure lag 8 port 1/1/1' to be recreated automatically for the new
    value to take effect.
```

For an overall view of the configuration commands available in the MD-CLI, refer to the *MD-CLI Command Reference Guide*.

3.5 Operational Root and Global Commands

The commands in Table 3 are available at the operational root level of the MD-CLI hierarchy.

Table 3 **Operational Root Commands**

Command	Description
admin	Enter the administrative context for system operations
clear	Clear statistics or reset operational state
configure	Enter the configuration context
environment	Enter the environment configuration context
li	Enter the lawful intercept configuration context
show	Show operational information
tools	Enter the tools context for troubleshooting and debugging

The global commands in Table 4 are available from various levels of the MD-CLI hierarchy.

Table 4 **Global Commands**

Command	Description
back	Move back one or more levels
delete	Delete an element from the candidate datastore
edit-config	Enter a candidate configuration mode
enable	Enable administrative mode
exec	Execute commands from a file
exit	Return to the previous context or to the operational root
history	Show the most recently entered commands
logout	Exit the CLI session
oam	Perform OAM tests. See Using the oam Commands.
ping	Trigger ping of an IP address or DNS name. See Using the ping Command.
pwc	Show the present working context. See **pwc** under Navigating the MD-CLI Hierarchy Levels.
quit-config	Leave the candidate configuration mode
top	Move to the top level of the context
traceroute	Determine the route to a destination address. See Using the traceroute Command.
tree	Show the command tree under the present working context

Table 5 lists configuration commands that are available in configuration mode.

Table 5 **Configuration Commands**

Command	Description
commit	Commit changes to the running datastore
compare	Show changes between datastores
discard	Discard changes in the candidate datastore
info	Show the running configuration from the present working context
load	Load a configuration from a file
rollback	Rollback to a previous configuration

Table 5 **Configuration Commands (Continued)**

Command	Description
update	Update the candidate baseline
validate	Validate changes in the candidate datastore

3.5.1 Using the ping Command

Use the **ping** command in the MD-CLI to verify the reachability of a host. The syntax is as follows:

— **ping** [**destination-address**] {*ip-address* | *string* } [**bypass-routing**] [**count** *number*] [**do-not-fragment**] [**fc** *keyword*] [**interface** *string*] [**interval** {*number* | *decimal-number*}] [**next-hop-address** {*ipv4-address* | *ipv6-address*}] [**output-format** *keyword*] [**pattern** {*keyword* | *number*}] [**router-instance** *string*] [**size** *number*] [**source-address** {*ipv4-address* | *ipv6-address*}] [**subscriber** *string*] [**timeout** *number*] [**tos** *number*] [**ttl** *number*]

Example command output:

```
[]
A:admin@node-2# ping 10.251.72.68
PING 10.251.72.68 56 data bytes
64 bytes from 10.251.72.68: icmp_seq=1 ttl=64 time=1.79ms.
64 bytes from 10.251.72.68: icmp_seq=2 ttl=64 time=1.07ms.
64 bytes from 10.251.72.68: icmp_seq=3 ttl=64 time=1.19ms.
64 bytes from 10.251.72.68: icmp_seq=4 ttl=64 time=1.26ms.
64 bytes from 10.251.72.68: icmp_seq=5 ttl=64 time=1.16ms.

---- 10.251.72.68 PING Statistics ----
5 packets transmitted, 5 packets received, 0.00% packet loss
round-trip min = 1.07ms, avg = 1.29ms, max = 1.79ms, stddev = 0.254ms

[]
A:admin@node-2# ping 10.251.72.68 output-format summary
PING 10.251.72.68 56 data bytes !!!!!
---- 10.251.72.68 PING Statistics ----
5 packets transmitted, 5 packets received, 0.00% packet loss
round-trip min = 0.842ms, avg = 1.02ms, max = 1.34ms, stddev = 0.172ms
```

Table 6 describes the parameters of the **ping** command.

Table 6 **Ping Parameters**

Ping Parameter	Description
[**destination-address**] {*ip-address* | *string*}	IP address or DNS name of the remote host to ping, where the IP address can be one of: • *ipv4-address-with-zone* • *ipv4-address* • *ipv6-address-linklocal-with-zone* • *ipv6-address* • *ipv6-address-with-zone*
bypass-routing	Bypass the routing table when sending the ping request to a host on a directly-attached network; return an error if the host is not on a directly-attached network
count *number*	Number of ping requests to send to the remote host
do-not-fragment	Do not fragment the request frame, which is particularly useful in combination with the **size** parameter for maximum MTU determination (does not apply to ICMPv6)
fc *keyword*	Forwarding class options for the transmitted ICMP Echo Request packet: • **be** • **l2** • **af** • **l1** • **h2** • **ef** • **h1** • **nc**
interface *string*	Interface name
interval {*number* | *decimal-number*}	Time between consecutive ping requests
next-hop-address {*ipv4-address* | *ipv6-address*}	Disregard the routing table and send the packet to the specified next hop address, which must be on an adjacent router attached to a common subnet
output-format *keyword*	Keyword options are: • **summary** • **detail** - show extra information in error cases

Table 6 **Ping Parameters (Continued)**

Ping Parameter	Description
pattern {*keyword* | *number*}	16-bit pattern string to include in the packet (expressed as a decimal integer) or a system-generated sequential pattern (using the keyword **sequential**)
router-instance *string*	Router name, CPM router instance, or service ID
size *number*	Size of request packets, including the ICMP header data (8 bytes) and the ICMP payload of the ICMP Echo Request packets
source-address {*ipv4-address* | *ipv6-address*}	Source IP address used in the ICMP Echo Request packets
subscriber *string*	Subscriber ID used when sending ICMP Echo Request packets
timeout *number*	Time to wait for reply packet. The timer is started when the last ICMP Echo Request is sent.
tos *number*	Type-of-Service (ToS) bits in the IP header of the ICMP Echo Request packets
ttl *number*	Time To Live (TTL) value included in the ICMP Echo Request packets

3.5.2 Using the traceroute Command

Use the traceroute command in the MD-CLI to display the route that packets take to a specified host.

— **traceroute** [**destination-address**] {*ipv4-address* | *ipv6-address* | *string* } [**detail**] [**numeric**] [**router-instance** *string*] [**source-address** [*ipv4-address* | *ipv6-address*] [**tos** *number*] [**ttl** *number*] [**wait** *number*]

Example command output:

```
[]
A:admin@node-2# traceroute 10.251.72.68
traceroute to 10.251.72.68, 30 hops max, 40 byte packets
  1  10.251.72.1 (10.251.72.1)    1.64 ms  1.71 ms  1.49 ms
  2  10.251.72.68 (10.251.72.68)    1.21 ms  1.23 ms  1.12 ms
```

Table 7 describes the parameters of the **traceroute** command.

Table 7 **Traceroute Parameters**

Traceroute Parameter	Description
[**destination-address**] {*ip-address* | *string*}	Destination IP address or DNS name, where the IP address can be one of: • *ipv4-address* • *ipv6-address*
detail	Display the MPLS label stack information (if available)
numeric	Avoid looking up DNS names when displaying results
router-instance *string*	Router name, CPM router instance, or service ID
source-address {*ipv4-address* | *ipv6-address*}	Source address of the probe packets; return an error If the IP address is not one of the device's interfaces
tos *number*	ToS bits in the IP header of the probe packets
ttl *number*	TTL value included in the traceroute request
wait *number*	Time to wait for a response to a probe

3.5.3 Using the oam Commands

The following **oam** commands are available in the MD-CLI.

3.5.3.1 OAM EFM Commands

The following commands issue Ethernet in the First Mile (EFM) OAM loopback tests on the specified port.

— **oam efm local-loopback** {**start** | **stop**} **port-id** {*ethernet-satellite-client-port* | *connector-port* | *port* | *pxc-sub-port*}

— **oam efm remote-loopback** {**start** | **stop**} **port-id** {*ethernet-satellite-client-port* | *connector-port* | *port* | *pxc-sub-port*}

Table 8 describes the parameters of the available OAM EFM commands.

Table 8 **OAM EFM Parameters**

EFM Parameter	Description
local-loopback {**start** | **stop**} **port-id** {*ethernet-satellite-client-port* | *connector-port* | *port* | *pxc-sub-port*}	Start or stop the local loopback test on the specified port
remote-loopback {**start** | **stop**} **port-id** {*ethernet-satellite-client-port* | *connector-port* | *port* | *pxc-sub-port*}	Start or stop the remote loopback test on the specified port

3.5.3.2 OAM ETH-CFM Commands

The following command issues an Ethernet Connectivity Fault Management (ETH-CFM) test. The implementation supports a single ETH-TST PDU to check unidirectional reachability launched from a source Maintenance Association End Point (MEP) and terminated on the remote MEP with no response PDU toward the source.

- **oam eth-cfm eth-test** {*mac-address* | *number*} **mep-id** *number* **md-admin-name** *reference* **ma-admin-name** *reference* [**data-length** *number*] [**priority** *number*]

The following command issues a linktrace test.

- **oam eth-cfm linktrace** {*mac-address* | *number*} **mep-id** *number* **md-admin-name** *reference* **ma-admin-name** *reference* [**ttl** *number*]

The following command issues a loopback test.

- **oam eth-cfm loopback** {*mac-address* | **multicast** | *number*} **mep-id** *number* **md-admin-name** *reference* **ma-admin-name** *reference* [**interval** *number*] [**lbm-padding** *number*] [**priority** *number*] [**send-count** *number*] [**size** *number*] [**timeout** *number*]

The following command issues an Ethernet CFM one-way delay test.

- **oam eth-cfm one-way-delay-test** {*mac-address* | *number*} **mep-id** *number* **md-admin-name** *reference* **ma-admin-name** *reference* [**priority** *number*]

The following command issues an Ethernet CFM two-way delay test.

- **oam eth-cfm two-way-delay-test** {*mac-address* | *number*} **mep-id** *number* **md-admin-name** *reference* **ma-admin-name** *reference* [**priority** *number*]

The following command issues an Ethernet CFM two-way SLM test in SAA.

— **oam eth-cfm two-way-slm-test** {*mac-address* | *number*} **mep-id** *number* **md-admin-name** *reference* **ma-admin-name** *reference* [**interval** *number*] [**priority** *number*] [**send-count** *number*] [**size** *number*] [**timeout** *number*]

Table 9 describes the parameters of the available OAM ETH-CFM commands.

Table 9 **OAM ETH-CFM Parameters**

ETH-CFM Parameter	Description
mac-address | *number*	Unicast destination MAC address or the remote MEP ID of the peer within the association. For an ETH-CFM loopback test, the MAC address can be a multicast MAC address.
multicast	Build the class 1 destination multicast address based on the level of the local MEP. The last nibble of the multicast address must match the level of the local MEP or the command fails and the test is not instantiated.
mep-id *number*	Local MEP ID
md-admin-name *reference*	Referenced domain name
ma-admin-name *reference*	Referenced association name
data-length *number*	Size of the padding to be added to the frame
interval *number*	Time between probes within the test run
lbm-padding *number*	Size of the data portion of the data TLV which does not allow for an optional octet string. MSDU is not processed with this option. The **lbm-padding** and **size** options are mutually exclusive.
priority *number*	Priority of the frame, which can be manipulated by QoS policies
send-count *number*	Number of messages to send
size *number*	Size of the data portion of the data TLV allowing for an optional octet string to be specified. The **size** and l**bm-padding** options are mutually exclusive.

Table 9 **OAM ETH-CFM Parameters (Continued)**

ETH-CFM Parameter	Description
timeout *number*	Time that the router waits for a message reply after sending a message request. Upon expiration of the timeout, the router assumes that the message response is not received. Any response received after the timeout is silently discarded.
ttl *number*	Time to Live for a returned linktrace

3.5.3.3 OAM OAM-PM Commands

The following command issues an on-demand OAM Performance Monitoring (OAM-PM) test.

— **oam oam-pm action** {**start** | **stop**} **session** *reference* **test-type** {**dm** | **dmm** | **lmm** | **slm** | **twamp-light**}

Table 10 describes the parameters of the available OAM-PM commands.

Table 10 **OAM OAM-PM Parameters**

OAM-PM Parameter	Description
action {**start**	**stop**}	Start or stop an OAM-PM test
session *reference*	Referenced OAM-PM session name
test-type {**dm**	**dmm**	**lmm**	**slm**	**twamp-light** }	Test type • **dm** - MPLS Delay Measurement test • **dmm** - Ethernet Delay Measurement Message test • **lmm** - Ethernet Loss Measurement Message test • **slm** - Ethernet Synthetic Loss Measurement test • **twamp-light** - Two-Way Active Measurement Protocol (TWAMP) Light test

3.6 Navigating the MD-CLI Hierarchy Levels

The following commands can be used to navigate the MD-CLI hierarchy (context) levels:

- **pwc**

  The **pwc** command shows the present working context with all keyword and variable parameters. The syntax is as follows:

  — **pwc** [[**path-type**] {**model-path** | **xpath**}] [**previous**]

```
(ex)[]
A:admin@node-2# configure

(ex)[configure]
A:admin@node-2# card 1

(ex)[configure card 1]
A:admin@node-2# mda 2

*(ex)[configure card 1 mda 2]
A:admin@node-2# network

*(ex)[configure card 1 mda 2 network]
A:admin@node-2# pwc
Present Working Context:
  configure
  card 1
  mda 2
  network
```

  – **pwc previous**

    The **pwc previous** command displays the previous working context.

```
*(ex)[configure card 1 mda 2 network]
A:admin@node-2# pwc previous
Previous Working Context:
  configure
  card 1
  mda 2
```

  – **pwc path-type**

    The path can be displayed in alternate formats. The **model-path** format is a YANG-modeled path format that can be used with RESTCONF-based management systems. The **xpath** format can be used with telemetry systems.

```
*(ex)[configure card 1 mda 2 network]
A:admin@node-2# pwc model-path
Present Working Context:
/nokia-conf:configure/card=1/mda=2/network
```

```
*(ex)[configure card 1 mda 2 network]
A:admin@node-2# pwc xpath
Present Working Context:
/configure/card[slot-number=1]/mda[mda-slot=2]/network
```

- **back**

  The **back** command can be used to go back one or more levels. If no parameter value is specified for the number of levels to go back, the default is one level. Using **back** at the top of the current command tree moves the context to the operational root level. If the number of levels specified is greater than the current depth, the context moves to the operational root. A closing brace (}) can also be used to go back one level.

```
*(ex)[configure card 1 mda 2 network]
A:admin@node-2# back

*(ex)[configure card 1 mda 2]
A:admin@node-2# back 2

*(ex)[configure]
A:admin@node-2# back 5

*(ex)[]
A:admin@node-2#
```

- **top**

  The **top** command moves the context to the top of the current command tree without exiting the mode. This command can be used instead of executing the **back** command a number of times to move to the top of the command tree.

```
*(ex)[]
A:admin@node-2# configure

*(ex)[configure]
A:admin@node-2# card 1

*(ex)[configure card 1]
A:admin@node-2# mda 2

*(ex)[configure card 1 mda 2]
A:admin@node-2# network

*(ex)[configure card 1 mda 2 network]
A:admin@node-2# top

*(ex)[configure]
A:admin@node-2#
```

- **exit**

The **exit** command moves the context to the previous context in the current command tree. If the previous context was up one level, the **exit** command functions similarly to the **back** command. Using **exit all** moves the context to the operational root. A slash (/) can also be used instead of **exit all**. Using **exit** at the operational root has no effect. To log out of the system, the **logout** command must be used.

```
*(ex)[]
A:admin@node-2#

*(ex)[]
A:admin@node-2# configure card 1 mda 2

*(ex)[configure card 1 mda 2]
A:admin@node-2# atm

*(ex)[configure card 1 mda 2 network]
A:admin@node-2# exit all

*(ex)[]
A:admin@node-2# configure card 1 mda 2 network

*(ex)[configure card 1 mda 2 network]
A:admin@node-2# /

*(ex)[]
A:admin@node-2#
```

3.7 Using the tree Command

The **tree** command displays the command tree under the present working context, excluding the present working context element. Hierarchy is indicated with a pipe (|), and a "**+--** " separator precedes each element. The tree output is in alphabetical order of elements.

```
[ex:configure system security aaa remote-servers]
A:admin@node-2# tree
+-- apply-groups
+-- ldap
|   +-- admin-state
|   +-- apply-groups
|   +-- public-key-authentication
|   +-- server
|   |   +-- address
|   |   |   +-- apply-groups
|   |   |   +-- port
|   |   +-- admin-state
|   |   +-- apply-groups
|   |   +-- bind-authentication
|   |   |   +-- password
|   |   |   +-- root-dn
|   |   +-- search
```

```
|   |   |   +-- base-dn
|   |   +-- server-name
|   |   +-- tls-profile
|   +-- server-retry
|   +-- server-timeout
|   +-- use-default-template
+-- radius
|   +-- access-algorithm
|   +-- accounting
|   +-- accounting-port
|   +-- admin-state
|   +-- apply-groups
|   +-- authorization
|   +-- interactive-authentication
|   +-- port
|   +-- server
|   |   +-- address
|   |   +-- apply-groups
|   |   +-- secret
|   +-- server-retry
|   +-- server-timeout
|   +-- use-default-template
+-- tacplus
    +-- accounting
    |   +-- record-type
    +-- admin-control
    |   +-- tacplus-map-to-priv-lvl
    +-- admin-state
    +-- apply-groups
    +-- authorization
    |   +-- use-priv-lvl
    +-- interactive-authentication
    +-- priv-lvl-map
    |   +-- apply-groups
    |   +-- priv-lvl
    |       +-- apply-groups
    |       +-- user-profile-name
    +-- server
    |   +-- address
    |   +-- apply-groups
    |   +-- port
    |   +-- secret
    +-- server-timeout
    +-- use-default-template


[]
A:admin@node-2# tree
+-- admin
|   +-- clear
|   |   +-- security
|   |       +-- lockout
|   |       |   +-- all
|   |       |   +-- user
|   |       +-- password-history
|   |           +-- all
|   |           +-- user
|   +-- disconnect
|   |   +-- address
|   |   +-- session-id
```

```
|   |   +-- session-type
|   |   +-- username
|   +-- reboot
|   |   +-- now
|   +-- redundancy
|   |   +-- force-switchover
|   |       +-- now
|   +-- save
|   |   +-- nat
|   |       +-- deterministic-script
|   +-- set
|   |   +-- time
|   +-- show
|   |   +-- configuration
|   |       +-- json
|   +-- support-mode
|   +-- system
|       +-- license
|       |   +-- activate
|       |   |   +-- now
|       |   +-- validate
|       +-- security
|       |   +-- hash-control
|       |       +-- custom-hash
|       |           +-- algorithm
|       |           +-- key
|       +-- telemetry
|           +-- grpc
|               +-- cancel
|                   +-- all
|                   +-- subscription-id
+-- back
```

3.7.1 Using the flat Option

The **flat** option displays the command hierarchy under the present working context on one line, excluding the present working context element.

```
(ro)[]
A:admin@node-2# tree flat
admin
admin clear
admin clear security
admin clear security lockout
admin clear security lockout all
admin clear security lockout user
admin clear security password-history
admin clear security password-history all
admin clear security password-history user
admin disconnect
admin disconnect address
admin disconnect session-id
admin disconnect session-type
admin disconnect username
admin reboot
```

```
admin reboot now
admin redundancy
admin redundancy force-switchover
admin redundancy force-switchover now
admin save
```

3.7.2 Using the detail Option

The **detail** option displays all key and field values in the output on every line.

```
(ro)[]
A:admin@node-2# tree detail
+-- admin
|   +-- clear
|   |   +-- security
|   |       +-- lockout
|   |       |   +-- all
|   |       |   +-- user <string>
|   |       +-- password-history
|   |           +-- all
|   |           +-- user <string>
|   +-- disconnect
|   |   +-- address <ipv4-address | ipv6-address>
|   |   +-- session-id <number>
|   |   +-- session-type <keyword>
|   |   +-- username <string>
|   +-- reboot
|   |   +-- <keyword>
|   |   +-- now

  <snip>
```

The **flat** and **detail** options can be combined in any order.

```
(ro)[]
A:admin@dut-b# tree flat detail
admin
admin clear
admin clear security
admin clear security lockout
admin clear security lockout all
admin clear security lockout user <string>
admin clear security password-history
admin clear security password-history all
admin clear security password-history user <string>
admin disconnect
admin disconnect address <ipv4-address | ipv6-address>
admin disconnect session-id <number>
admin disconnect session-type <keyword>
admin disconnect username <string>
admin reboot
admin reboot <keyword>
admin reboot now
admin redundancy
```

```
admin redundancy force-switchover
admin redundancy force-switchover now

<snip>
```

3.8 Using Control Characters and Editing Keystrokes on the Command Line

Table 11 lists the control characters and keystrokes available to execute and edit commands.

Table 11 **Control Characters**

Command	Description
/ (Slash)	Return to the operational root (equivalent to **exit all**) if used without parameters. Navigate into context or set the value and remain in current context if used at the beginning of a line (equivalent to **exit all**, and then the command)
} (Closing Brace)	Go back one level
CTRL-z	Return to operational root. If using CTRL-z after a command, return to the operational root after executing the command (equivalent to pressing Enter after the command and **exit all** after the command has executed).
CTRL-c	Stop the current command
CTRL-d	Delete the current character
CTRL-w	Delete the word up to the cursor
CTRL-h	Delete the current character and move the cursor left
CTRL-u	Delete text up to the cursor and preserve the character under the cursor
CTRL-k	Delete the text after the cursor, without preserving the character under the cursor
CTRL-a (or Home)	Move to the beginning of the line
CTRL-e (or End)	Move to the end of the line

Table 11 **Control Characters (Continued)**

Command	Description
CTRL-p (or Up arrow)	Display prior command from history
CTRL-n (or Down arrow)	Display next command from history
CTRL-b (or Left arrow)	Move the cursor one space to the left
CTRL-f (or Right arrow)	Move the cursor one space to the right
ESC+b	Move back one word, or to the beginning of the current word if the cursor is not at the start of the word
CTRL-l	Clear the screen

3.9 Displaying Available Commands using Tab

Variables, keywords, global commands, and configuration commands and units are separated by a blank line in the output, in the following order:

- values or units (mutually exclusive)
- keywords
- global commands
- configuration commands

```
[ex:configure log]
A:admin@node-2# Press Tab

 accounting-policy        app-route-notifications  event-damping
 event-handling           event-trigger            file
 filter                   log-events               log-id
 route-preference         services-all-events      snmp-trap-group
 syslog                   throttle-rate

 back                     delete                   edit-config
 exec                     exit                     history
 logout                   ping                     pwc
 top                      traceroute               tree

 commit                   compare                  discard
 info                     update                   validate


[ex:configure log]
A:admin@node-2# event-damping Press Tab
```

```
 <event-damping>
 false
 true

 accounting-policy        app-route-notifications  apply-groups
 event-handling           event-trigger            file
 filter                   log-events               log-id
 route-preference         services-all-events      snmp-trap-group
 syslog                   throttle-rate

 delete
```

The ? help displays similar information but does not always display global or configuration commands.

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

 accounting-policy     + Enter the accounting-policy list instance
 app-route-            + Enter the app-route-notifications context
  notifications
 apply-groups          - Apply a configuration group at this level
 event-damping         - Allow event damping algorithm to suppress QoS or
                         filter change events
 event-handling        + Enter the event-handling context
 event-trigger         + Enter the event-trigger context
 file                  + Enter the file list instance
 filter                + Enter the filter list instance
 log-events            + Enter the log-events context
 log-id                + Enter the log-id list instance
 route-preference      + Enter the route-preference context
 services-all-events   + Enter the services-all-events context
 snmp-trap-group       + Enter the snmp-trap-group list instance
 syslog                + Enter the syslog list instance
 throttle-rate         + Enter the throttle-rate context

[ex:configure log]
A:admin@node-2# event-damping ?

 event-damping <boolean>
 <boolean>  - ([true]|false)
 Default    - true

    Allow event damping algorithm to suppress QoS or filter change events

 accounting-policy     + Enter the accounting-policy list instance
 app-route-            + Enter the app-route-notifications context
  notifications
 apply-groups          - Apply a configuration group at this level
 event-handling        + Enter the event-handling context
 event-trigger         + Enter the event-trigger context
 file                  + Enter the file list instance
 filter                + Enter the filter list instance
 log-events            + Enter the log-events context
 log-id                + Enter the log-id list instance
 route-preference      + Enter the route-preference context
 services-all-events   + Enter the services-all-events context
```

```
snmp-trap-group       + Enter the snmp-trap-group list instance
syslog                + Enter the syslog list instance
throttle-rate         + Enter the throttle-rate context
```

3.10 Using Command Completion

The MD-CLI supports both command abbreviation and command completion. When typing a command, Tab, Spacebar, or Enter invokes auto-completion. If the text entered is enough to match a specific command, auto-completion completes the command. If the text entered is not sufficient to identify a specific command, pressing Tab or Spacebar displays options in alphabetical order matching the text entered.

The **environment command-completion** command controls what keystrokes can trigger command completion. Each keystroke is independently controlled with its own Boolean value.

```
(ex)[environment command-completion]
A:admin@node-2# info detail
    enter true
    space true
    tab true
```

Note: If Spacebar completion has multiple matches and also matches an keyword, the space is considered a separator and auto-completion is not triggered.

- **configure por**+Spacebar displays auto-completion results
- **configure port**+Spacebar inserts a space and suppresses auto-completion results
- **configure por**+Tab displays auto-completion results
- **configure port**+Tab displays auto-completion results

3.10.1 Variable Parameter Completion

Variable parameter completion works only with the Tab key. All configured variables from the candidate and running configuration datastores are displayed. Line wrapping may occur for variables with long names. Parameters are displayed in alphabetical or numerical order. The variable parameter name is always displayed as the first line. In the following example, “interface-name” is the variable parameter name and “int-1” and “system” are configured names.

```
*(ex)[configure router "Base"]
A:admin@node-2# interface  Press Tab

 <interface-name>
```

```
 "int-1"
 "system"

*(ex)[configure router "Base"]
A:admin@node-2# interface
```

3.10.1.1 Completion for Lists with a Default Keyword

Some list elements have a default keyword defined, such as the **router** command, where the default keyword is “Base”. When the command completion parameters (**enter**, **space**, and **tab**) are at their default settings (**true**), and the initial input matches an element in the list and a unique command keyword, the matching keyword is completed instead of the variable.

For example, the **router** command has a default keyword defined as “Base”. If router “bf” is created using the command **configure router “bf”** (with quotation marks”), and there is an existing **bfd** command context, the variable completion is as follows.

The following displays for **router**+Spacebar+Tab:

```
*[ex:configure]
A:admin@node-2# router  Press Tab

 <router-name>
 "Base"
 "bf"
 "management"

 aggregates             allow-icmp-redirect    allow-icmp6-redirect
 apply-groups           autonomous-system      bfd
```

The following displays for **router bf**+Tab:

```
*[ex:configure]
A:admin@node-2# router bf  Press Tab

 "bf"

 bfd
```

Entering **router bf**+Enter completes to **router bfd** and enters the **router “Base” bfd** context:

```
*[ex:configure]
A:admin@node-2# router bfd  Press Enter

*[ex:configure router "Base" bfd]
A:admin@node-2#
```

Similarly, **router bf**+Spacebar completes to **router bfd** and enters the **router "Base" bfd** context when Enter is pressed:

```
*[ex:configure]
A:admin@node-2# router bfd  Press Space  Press Enter

*[ex:configure router "Base" bfd]
A:admin@node-2#
```

To enter the context for router "bf", use quotation marks to specify the variable:

```
[ex:configure]
A:admin@node-2# router "bf"

*[ex:configure router "bf"]
A:admin@node-2#
```

If the command completion for **enter** is set to **false**, then **router bf**+Enter allows the match to router "bf". Similarly, when the command completion for **space** is **false**, **router bf**+Spacebar also matches to router "bf" instead of the **bfd** context

```
*(ex)[environment command-completion]
A:admin@node-2# info detail
    enter true
    space true
    tab true

*(ex)[environment command-completion]
A:admin@node-2# enter false

*(ex)[environment command-completion]
A:admin@node-2# space false

*(ex)[environment command-completion]
A:admin@node-2#

*(ex)[]
A:admin@node-2# configure

*(ex)[configure]
A:admin@node-2# router bf  Press Enter

*(ex)[configure router "bf"]
A:admin@node-2# back

*(ex)[configure]
A:admin@node-2# router bf  Press Spacebar+Enter

*(ex)[configure router "bf"]
A:admin@node-2#
```

3.10.1.2 Completion for Keyword-based Leaf-lists

For keyword-based leaf-lists, command completion displays all possible values, not only those that are configured. When deleting values in a leaf-list, only the values that are currently configured are displayed. In the following example, when defining the forwarding traffic classes, all keyword values are listed. When deleting the forwarding traffic classes, only the configured classes are displayed.

```
*[ex:configure policy-options policy-statement "ss" entry 3 from]
A:admin@node-2# family ?

 family <value>
 family [<value>...] - 1..20 system-ordered values separated by spaces enclosed
                       by brackets
 <value> - <keyword>
<keyword> - (ipv4|vpn-ipv4|ipv6|mcast-ipv4|vpn-ipv6|l2-vpn|mvpn-ipv4|mdt-
              safi|ms-pw|route-target|mcast-vpn-ipv4|mvpn-ipv6|evpn|mcast-
              ipv6|label-ipv4|label-ipv6|bgp-ls|mcast-vpn-ipv6)

    Match address families to this condition

*[ex:configure policy-options policy-statement "ss" entry 3 from]
A:admin@node-2# family [ipv4 mcast-ipv4 mcast-vpn-ipv4 label-ipv4]

*[ex:configure policy-options policy-statement "ss" entry 3 from]
A:admin@node-2# info
    family [ipv4 mcast-ipv4 mcast-vpn-ipv4 label-ipv4]

*[ex:configure policy-options policy-statement "ss" entry 3 from]
A:admin@node-2# delete family Press Tab

 <family>
 ipv4
 mcast-ipv4
 mcast-vpn-ipv4
 label-ipv4
 *
```

3.10.1.3 Completion for Boolean Elements

The explicit use of the keyword **true** for a Boolean element is optional. If neither **true** or **false** is entered, the keyword **true** is assumed.

```
(ex)[environment]
A:admin@node-2# more ?

 more <boolean>
 <boolean> - ([true]|false)
 Default   - true

    Paging control of the output text
```

When Tab is used for command completion with Boolean elements, the values of **false** and **true** are displayed, along with the names of possible elements that can follow. In the following example of the **environment more** command, the commands **command-completion**, **console**, **message-severity-level**, and so on, can be defined following the **more** command.

```
(ex)[environment]
A:admin@node-2# more  Press Tab

 <more>
 false
 true

 command-completion     console                message-severity-level
 progress-indicator     prompt                 time-display

 delete

(ex)[environment]
A:admin@node-2# more
```

3.11 Modifying the Idle Timeout Value for CLI Sessions

A single idle timeout applies to all CLI engines in a CLI session (classic and MD-CLI). The idle timeout can be modified to a value between 1 and 1440 minutes.

The following points apply.

- The idle timeout only affects new CLI sessions. Existing and current sessions retain the previous idle timeout.
- The idle timeout can be disabled by setting the value to **none**.
- The “Idle time” column in the **show users** display is reset after an action in either CLI engine.

```
[]
A:admin@node-2# show users
===============================================================================
User                              Type      Login time           Idle time
  Session ID   From
===============================================================================
                                  Console         --             6d 19:38:00 --
  6            --
admin                             SSHv2     11OCT2018 14:35:19   0d 00:00:00 --
 #23           192.168.144.87
admin                             SSHv2     11OCT2018 14:35:55   0d 00:07:46 --
  24           192.168.144.87
-------------------------------------------------------------------------------
Number of users: 2
```

```
'#' indicates the current active session
===============================================================================
```

A warning message is displayed when a session reaches one-half the value of the idle timeout, and another message is displayed when the idle timeout expires.

3.11.1 Idle Timeout Interaction with the Classic CLI

The idle timeout configured in the classic CLI affects all new sessions as well as the current session. However, the current session is only affected if the classic CLI engine is active when the idle timeout expires. Configuration changes via the MD-CLI or any other interface, including SNMP, only affect new sessions that begin after the change.

3.12 Using Output Modifiers in the MD-CLI

Output modifiers provide support for post-processing of CLI output. Output modifiers are specified using a pipe (|) character. The following points apply when using output modifiers.

- Output modifiers can be appended to any CLI command in any command context.
- Output modifiers work across soft line breaks (visual lines) that are wrapped due to the terminal width; for example, using **match** or **count**. They do not work across hard line breaks (logical lines).
- Modifiers can be combined in any order. No hard limit exists for the number of combinations. Output is processed linearly and there is little impact on the system performance except to the operator session that entered the modifier combination.

3.12.1 Using | match Options

The following options are supported for use with the pipe (|) **match** command:

- **ignore-case** — specifies to ignore case in pattern match
- **invert-match** — specifies to invert the pattern match selection
- **max-count** — specifies the maximum number of displayed matches

- **post-lines** — specifies the number of lines to display following the matched line
- **pre-lines** — specifies the number of lines to display preceding the matched line

The following example matches on the pattern **autonomous-system** in the **tree detail** under the **configure router "Base"** context, and starts the display with seven lines preceding the pattern match.

```
(ex)[configure router "Base"]
A:admin@node-2# tree detail | match autonomous-system pre-lines 7
|   |   +-- indirect <unicast-ipv4-address | global-unicast-ipv6-address>
|   |   +-- local-preference <number>
|   |   +-- summary-only <boolean>
|   +-- apply-groups <reference>
+-- allow-icmp-redirect <boolean>
+-- allow-icmp6-redirect <boolean>
+-- apply-groups <reference>
+-- autonomous-system <number>

(ex)[configure router "Base"]
A:admin@node-2#
```

3.12.1.1 Using Regular Expressions with | match

Regular expressions (REs) used by the MD-CLI engine are delimited by apostrophes (‘); for example, ‘.*’. REs cannot be delimited by double quotation marks (“); for example, “.*”.

MD-CLI REs are based on a subset of The Open Group Base Specifications Issue 7 and IEEE Std 1003.1-2008, 2016 Edition REs, as defined in chapter 9. MD-CLI REs only support Extended Regular Expression (ERE) notation as defined in section 9.4. Basic Regular Expression (BRE) notation as defined in section 9.3 is not supported.

In ERE notation, a backslash (\) before a special character is treated as a literal character. Backslashes are not supported before ( ) or { }, as they are in BREs to indicate a bracket expression or marked expression.

Table 12 describes the special characters that are supported in EREs.

Table 12 **Special Characters in Extended Regular Expressions**

Special character	Description
.	Matches any single character
*	Matches the preceding expression zero or more times
?	Matches the preceding expression zero or one time

Table 12 **Special Characters in Extended Regular Expressions**

Special character	Description
+	Matches the preceding expression one or more times
[ ]	Matches a single character within the brackets
[^]	Matches a single character not within the brackets
^	Matches the starting position
$	Matches the ending position
( )	Defines a marked subexpression
{m,n}	Matches the preceding expression at least *m* and not more than *n* times
{m}	Matches the preceding expression exactly *m* times
{m, }	Matches the preceding expression at least *m* times
{ ,n}	Matches the preceding expression not more than *n* times
|	Matches either expression preceding or following the |
\	Treats the following character as a match criterion
-	Separates the start and end of a range

The following examples show the use of a bracket expression as a matching list expression.

The first output does not use any match expressions and therefore shows the entire output.

```
*(gl)[]
A:admin@node-2# show port

===============================================================================
Ports on Slot 1
===============================================================================
Port          Admin Link Port    Cfg  Oper LAG/ Port Port Port   C/QS/S/XFP/
Id            State      State   MTU  MTU  Bndl Mode Encp Type   MDIMDX
-------------------------------------------------------------------------------
1/1/1         Down  No   Ghost   8704 8704    - netw null xcme
1/1/2         Up    No   Ghost   1514 1514    - accs null xcme
1/1/3         Up    No   Ghost   1514 1514    - accs null xcme
1/1/4         Up    No   Ghost   1514 1514    - accs null xcme
1/1/5         Up    No   Ghost   1514 1514    - accs null xcme
1/1/6         Down  No   Ghost   8704 8704    - netw null xcme
1/1/7         Down  No   Ghost   8704 8704    - netw null xcme
1/1/8         Down  No   Ghost   8704 8704    - netw null xcme
1/1/9         Down  No   Ghost   8704 8704    - netw null xcme
1/1/10        Down  No   Ghost   8704 8704    - netw null xcme
```

```
1/1/11        Down  No   Ghost   8704 8704    - netw null xcme
1/1/12        Down  No   Ghost   8704 8704    - netw null xcme
1/2/1         Up    No   Ghost   8704 8704    - netw null xcme
1/2/2         Up    No   Ghost   1514 1514    - accs null xcme
1/2/3         Up    No   Ghost   1514 1514    - accs null xcme
1/2/4         Down  No   Ghost   8704 8704    - netw null xcme
Press Q to quit, Enter to print next line or any other key to print next page.
```

In this matching list expression, a match is any single character in the bracket expression, which in this case is 1, 3, or 5.

```
*(gl)[]
A:admin@node-2# show port | match '1/1/[135]'
1/1/1         Down  No   Ghost   8704 8704    - netw null xcme
1/1/3         Up    No   Ghost   1514 1514    - accs null xcme
1/1/5         Up    No   Ghost   1514 1514    - accs null xcme
1/1/10        Down  No   Ghost   8704 8704    - netw null xcme
1/1/11        Down  No   Ghost   8704 8704    - netw null xcme
1/1/12        Down  No   Ghost   8704 8704    - netw null xcme
```

In this non-matching list expression, a match is any single character not in the bracket expression, that is, not 1, 2, or 4.

```
*(gl)[]
A:admin@node-2# show port | match '1/1/[^124]'
1/1/3         Up    No   Ghost   1514 1514    - accs null xcme
1/1/5         Up    No   Ghost   1514 1514    - accs null xcme
1/1/6         Down  No   Ghost   8704 8704    - netw null xcme
1/1/7         Down  No   Ghost   8704 8704    - netw null xcme
1/1/8         Down  No   Ghost   8704 8704    - netw null xcme
1/1/9         Down  No   Ghost   8704 8704    - netw null xcme
```

The range operator (-) can be used in a matching or non-matching list expression.

```
*(ro)[]
A:admin@node-2# show port | match '1/1/[3-7]'
1/1/3         Up    No   Ghost   1514 1514    - accs null xcme
1/1/4         Up    No   Ghost   1514 1514    - accs null xcme
1/1/5         Up    No   Ghost   1514 1514    - accs null xcme
1/1/6         Down  No   Ghost   8704 8704    - netw null xcme
1/1/7         Down  No   Ghost   8704 8704    - netw null xcme
```

```
*(ro)[]
A:admin@node-2# show port | match '1/1/[^3-7]'
1/1/1         Down  No   Ghost   8704 8704    - netw null xcme
1/1/2         Up    No   Ghost   1514 1514    - accs null xcme
1/1/8         Down  No   Ghost   8704 8704    - netw null xcme
1/1/9         Down  No   Ghost   8704 8704    - netw null xcme
1/1/10        Down  No   Ghost   8704 8704    - netw null xcme
1/1/11        Down  No   Ghost   8704 8704    - netw null xcme
1/1/12        Down  No   Ghost   8704 8704    - netw null xcme
```

The alternation operator (|) can be used with or without a bracket expression to match against two or more alternative expressions.

```
*(ro)[]
A:admin@node-2# show port | match '1/1/[2-5|7-9]'
1/1/2         Up    No   Ghost   1514 1514    - accs null xcme
1/1/3         Up    No   Ghost   1514 1514    - accs null xcme
1/1/4         Up    No   Ghost   1514 1514    - accs null xcme
1/1/5         Up    No   Ghost   1514 1514    - accs null xcme
1/1/7         Down  No   Ghost   8704 8704    - netw null xcme
1/1/8         Down  No   Ghost   8704 8704    - netw null xcme
1/1/9         Down  No   Ghost   8704 8704    - netw null xcme
```

Without a bracket expression, an exact match is attempted against two or more alternative expressions.

```
*[ex:configure card 1]
A:admin@node-2# info | match '10g|100g'
        mda-type imm4-10gb-xp-xfp
        mda-type cx2-100g-cfp
```

MD-CLI REs match on the output format of an element, as shown in the configuration. For example, if the value of an element is shown in hexadecimal in **info** output, a decimal RE will not match the value. In the following example, the Ethertype is entered in decimal format, but is displayed in hexadecimal. Matching on the decimal format does not find a match.

```
*[ex:configure filter mac-filter "fn" entry 1 match]
A:admin@node-2# etype ?

 etype <number>
 <number> - <0x600..0xffff>

    Ethernet type

*[ex:configure filter mac-filter "fn" entry 1 match]
A:admin@node-2# etype 65535

*[ex:configure filter mac-filter "fn" entry 1 match]
A:admin@node-2# info
    etype 0xffff

*[ex:configure filter mac-filter "fn" entry 1 match]
A:admin@node-2# top

*[ex:configure]
A:admin@node-2# info | match 65535

*[ex:configure]
A:admin@node-2# info | match 0xffff
                    etype 0xffff

*[ex:configure]
A:admin@node-2#
```

MD-CLI REs are not implicitly anchored. The ^ or $ anchoring special characters can be used, as in the following example.

```
*(ex)[configure router "Base" bgp]
A:admin@node-2# info
    group "external" {
    }
    group "internal" {
    }
    neighbor 192.168.10.1 {
        group "external"
        keepalive 30
        peer-as 100
    }
    neighbor 192.168.10.2 {
        group "external"
        peer-as 100
        family {
            ipv4 true
        }
    }
```

This example uses the ^ anchor character to match on “group” preceded by four spaces at the beginning of the line.

```
*(ex)[configure router "Base" bgp]
A:admin@node-2# info | match '^    group' pre-lines 1
    group "external" {
    }
    group "internal" {
```

This example uses the ^ anchor character to match on “group” preceded by eight spaces at the beginning of the line.

```
*(ex)[configure router "Base" bgp]
A:admin@node-2# info | match '^        group' pre-lines 1
    neighbor 192.168.10.1 {
        group "external"
    neighbor 192.168.10.2 {
        group "external"
*(ex)[configure router "Base" bgp]
A:admin@node-2#
```

In the following configuration example using the **compare** command, the **| match** option filters out those commands to be deleted (configuration statements beginning with the minus sign (-)) and those to be added (configuration statements beginning with the plus sign (+)).

```
*(gl)[configure log accounting-policy 5]
A:admin@node-2# /compare
+   admin-state enable
-   collection-interval 105
+   collection-interval 75
-   include-system-info true
+   include-system-info false

*(gl)[configure log accounting-policy 5]
A:admin@node-2# /compare | match '^-'
```

```
-   collection-interval 105
-   include-system-info true
```

The backslash (\) is used to match the literal “+” character that denotes additions to the configuration seen in the **compare** command.

```
*(gl)[configure log accounting-policy 5]
A:admin@node-2# /compare | match '^\+'
+   admin-state enable
+   collection-interval 75
+   include-system-info false
```

A character class expression is expressed as a character class name enclosed within bracket colon (“[:” and “:]”) delimiters. Table 13 defines the character class expressions.

Table 13 **Character Class Expressions**

Character Class	Characters matched (delimited by ‘single quotation marks’)	Description
[:alnum:]	‘ABCDEFGHIJKLMNOPQRSTUVWXYZ abcdefghijklmnopqrstuvwxyz 0123456789’	Alphanumeric characters
[:alpha:]	‘ABCDEFGHIJKLMNOPQRSTUVWXYZ abcdefghijklmnopqrstuvwxyz’	Alphabetic characters
[:blank:]	‘ \t’	Space and Tab
[:cntrl:]	‘\007\b\t\n\v\f\r\1\2\3\4\5\6\16\17\20 \21\22\23\24\25\26\27\30 \31\32\33\34\35\36\37\177’	Control characters
[:digit:]	‘0123456789’	Digits
[:graph:]	‘ABCDEFGHIJKLMNOPQRSTUVWXYZ abcdefghijklmnopqrstuvwxyz 0123456789 !\"#$%&'()*+,-./ :;<=>?@[\\]^_`{|}~’	Visible characters
[:lower:]	‘abcdefghijklmnopqrstuvwxyz’	Lowercase letters
[:print:]	‘ABCDEFGHIJKLMNOPQRSTUVWXYZ abcdefghijklmnopqrstuvwxyz 0123456789!\"#$%&'()*+,-./ :;<=>?@[\\]^_`{|}~ ’	Visible characters and the Space character
[:punct:]	‘!\"#$%&'()*+,-./:;<=>?@[\\]^_`{|}~’	Punctuation characters
[:space:]	‘\t\n\v\f\r ‘	Whitespace (blank) characters

Table 13 **Character Class Expressions (Continued)**

Character Class	Characters matched (delimited by 'single quotation marks')	Description
[:upper:]	'ABCDEFGHIJKLMNOPQRSTUVWXYZ'	Uppercase letters
[:xdigit:]	'0123456789ABCDEFabcdef'	Hexadecimal digits

Character class expressions must be enclosed within brackets. The expression '[[:digit:]]' is treated as an RE containing the character class "digit", while '[:digit:]' is treated as an RE matching ":", "d", "i", "g", or "t".

Collating symbols and equivalence classes are not supported in MD-CLI REs.

3.12.2 Using the | count Option

The **| count** option displays the line count of the output.

```
[ex:configure router "Base"]
A:admin@node-2# tree flat detail | match seamless-bfd
bfd seamless-bfd
bfd seamless-bfd peer <unicast-ipv4-address | global-unicast-ipv6-address>
bfd seamless-bfd peer <unicast-ipv4-address | global-unicast-ipv6-address> apply-
groups <reference>
bfd seamless-bfd peer <unicast-ipv4-address | global-unicast-ipv6-
address> discriminator <number>

[ex:configure router "Base"]
A:admin@node-2# tree flat detail | match seamless-bfd | count
Count: 4 lines
```

Note: Error messages are not processed by output modifiers. They are always displayed and are not affected by the **count** or **match** modifiers.

3.12.3 Using the | no-more Option

The **| no-more** option displays the output with pagination disabled. This option is similar to the **environment more false** setting, where the entire output text is printed without page interruptions.

3.12.4 Using the File Redirect Option

The **>** option can be used to redirect output to a local or remote file. The **>** redirect must be specified at the end of a command and cannot be combined with other redirects.

```
*(ex)[configure router "Base"]
A:admin@node-2# info detail | match leak-export > ?

 [url] <string>
 <string> - <1..255 characters>

 The location where the output should be saved.
```

3.13 Navigating Contexts in the MD-CLI

3.13.1 Entering Contexts

Configuring a container navigates into the context. In the following example, the first container is **router**, and the next is **aggregates**. All containers are marked with a “+”.

```
[ex:configure]
A:admin@node-2# router

[ex:configure router "Base"]
A:admin@node-2# ?

 admin-state           - Administrative state of the router instance
 aggregates            + Enter the aggregates context
 allow-icmp-redirect   - Allow ICMP redirects on the management interface
 allow-icmp6-redirect  - Allow IPv6 ICMP redirects on the management interface
 apply-groups          - Apply a configuration group at this level
 autonomous-system     - AS number advertised to peers for this router
 bfd                   + Enter the bfd context
 bgp                   + Enter the bgp context
 bier                  + Enter the bier context
 class-forwarding      - Allow class-based forwarding over IGP shortcuts
...


(pr)[configure router "Base"]
A:admin@node-2# aggregates

(pr)[configure router "Base" aggregates]
A:admin@node-2#
```

Alternatively, the same context can be entered on one line:

```
(pr)[]
A:admin@node-2# configure router aggregates

(pr)[configure router "Base" aggregates]
A:admin@node-2#
```

Configuring a leaf element maintains the present working context if there is no explicit opening brace. Entering an explicit opening brace navigates into the specified context.

```
*[ex:configure card 1 mda 2]
A:admin@node-2# atm mode ?

 mode <keyword>
 <keyword> - (max8k-vc|max16k-vc)

    Mode of operation

*[ex:configure card 1 mda 2]
A:admin@node-2# atm mode max8k-vc

*[ex:configure card 1 mda 2]
A:admin@node-2# atm { mode max8k-vc

*[ex:configure card 1 mda 2 atm]
A:admin@node-2#
```

Configuring a container navigates into the context.

```
[ex:configure router "Base"]
A:admin@node-2# ?

 admin-state          - Administrative state of the router instance
 aggregates           + Enter the aggregates context
 allow-icmp-redirect  - Allow ICMP redirects on the management interface
 allow-icmp6-redirect - Allow IPv6 ICMP redirects on the management interface
 apply-groups         - Apply a configuration group at this level
 autonomous-system    - AS number advertised to peers for this router
 bfd                  + Enter the bfd context
 bgp                  + Enter the bgp context
 ...

[ex:configure router "Base"]
A:admin@node-2# bgp

[ex:configure router "Base" bgp]
A:admin@node-2# ?

 add-paths            + Enter the add-paths context
 admin-state          - Administrative state of the BGP instance
 advertise-external   + Enter the advertise-external context
 advertise-inactive   - Advertise an inactive BGP route to peers
...

[ex:configure router "Base" bgp]
A:admin@node-2# add-paths
```

```
[ex:configure router "Base" bgp add-paths]
A:admin@node-2#
```

Configuring an empty container or a list where the only children are keys does not navigate into the context. These elements are displayed with aggregated braces with a space ({ }) on the same line. It is possible to enter the element name with an opening brace; however, no options are available in this context.

For example, configuring the list element **sdp-include** with a key of “ref_group_name” does not change the existing context.

```
*(ex)[configure service pw-template "tt"]
A:admin@node-2# sdp-include ref_group_name

*(ex)[configure service pw-template "tt"]
A:admin@node-2# info
    sdp-include "ref_group_name" { }

*(ex)[configure service pw-template "tt"]
A:admin@node-2#
```

3.13.2 Exiting Contexts

The **back** and **top** commands are used to navigate contexts, but it is also possible to use closing braces (}) to navigate.

The behavior of an explicit closing brace depends on the contents of the current command line. If the command line contains an explicit opening brace, the closing brace exits to the parent context of the opening brace.

In the following example with an opening brace on the command line, the closing brace exits VPRN 1, and then enters the context of VPRN 2.

```
(ex)[]
A:admin@node-2# configure service vprn 1 { interface "intf1" description "vprn-
if" } vprn 2

*(ex)[configure service vprn "2"]
A:admin@node-2#
```

In the following example without an opening brace on the command line, the first closing brace exits interface “int1”, and the second closing brace exits VPRN 1 and enters the VPRN 2 context.

```
*(ex)[configure service]
A:admin@node-2# vprn 1 interface "int1" description vprn-if } } vprn 2

*(ex)[configure service vprn "2"]
A:admin@node-2#
```

3.14 Executing Commands with a File

The **exec** command executes commands from a file as if the user typed or pasted the input into the MD-CLI without command completion. The syntax can be seen as follows:

```
(ex)[configure]
A:admin@node-2# exec ?

 [url] <string>
 <string> - <0..255 characters>

    The location of the file to be executed.

(ex)[configure]
A:admin@node-2# exec my-url-fn ?

 exec

 echo                 - Displays the commands on screen as they are being
                        executed.
```

The **exec** command:

- errors if it detects an interactive input
- terminates in the CLI engine in which it completes execution as follows:
  - if there are no commands that switch CLI engines, the CLI engine is always the one in which **exec** started
  - if there are commands that switch CLI engines, **exec** ends in the last CLI engine that was entered
  - //**exec** returns to the engine in which it was started
- terminates execution and displays an error message if an error occurs, leaving the session in the same context as when the error occurred

The system executes the file as follows:

- disables pagination while the command is running
- disables command completion while the command is running
- suppresses the commands in the file from the command history

3.14.1 Using Commands that Switch Engines in an Executable File

When using commands that switch between CLI engines within an executable file, the following commands are recommended:

- use **/!classic-cli** to switch explicitly to the classic CLI engine and **/!md-cli** to switch explicitly to the MD-CLI engine, instead of **//** to toggle between engines
- use **exit all** to get to a known starting point: the operational root of the classic CLI or the MD-CLI engine
- include **edit-config** if the script needs to change the candidate configuration in the MD-CLI engine. Use **quit-config** after changes are committed in the script.

Note:

- An executable with **edit-config** may fail if other users have locked the configuration.
- Issuing the **quit-config** command with changes in the candidate configuration while the session is in exclusive configuration mode fails the executable because of the “discard changes” prompt.

3.15 Displaying Information in the MD-CLI

3.15.1 Using the info Command

The **info** command shows the configuration for the present context. The command can only be executed while in a configuration mode. By default, all configured parameters in the candidate configuration datastore are displayed.

- **info** [[**from**] (**candidate** | **running** | **baseline**)]
  - **converted**
  - **detail**
  - **flat**
  - **full-context**
  - **inheritance**
  - **json**

```
[ex:configure]
A:admin@node-2# info

 info [[from] <keyword>]
```

```
[[from] <keyword>]
<keyword>  - (candidate|running|baseline)

   Source datastore

converted             - Include converted configuration values from third party
detail                - Include default and unconfigured values in the output
flat                  - Show the hierarchy on each line
full-context          - Show full hierarchy on each line
inheritance           - Include configuration inherited from configuration
                        groups
json                  - Show the output in indented JSON format
```

Table 14 describes the **info** command options.

Table 14 **Info Command Options**

Option	Description
[**from**] (**candidate** | **running** | **baseline**)	Specify the source datastore (default is **from candidate**)
converted	Include converted configuration values from third party modules
detail	Include default and unconfigured values in the output
flat	Show the hierarchy on each line starting from the present working context
full-context	Show the full hierarchy on each line
inheritance	Include configuration inherited from configuration groups
json	Show the output in indented JSON format

Note: The **flat, full-context**, and **json** options are mutually exclusive. Other unsupported combinations include:

- **inheritance** and **converted**
- **inheritance** and **detail**

The order of the configuration output is as follows:

- keys are displayed on the same line as the command element
- **apply-groups** is displayed, if applicable
- **admin-state** is displayed, if applicable
- **description** is displayed, if applicable
- other top-level elements are displayed in alphabetical order

The following displays configured information for **configure router bgp**

```
*[ex:configure router "Base" bgp]
A:admin@node-2# info
    connect-retry 90
    local-preference 250
    add-paths {
        ipv4 {
            receive true
        }
    }
```

The following output displays the same information in JSON format:

```
*[ex:configure router "Base" bgp]
A:admin@node-2# info json
{
    "connect-retry": 90,
    "local-preference": 250,
    "nokia-conf:add-paths": {
        "ipv4": {
            "receive": true
        }
    }
}
```

The configuration output can display all elements that are configured, even if an element is set to the system default state or value.

The **detail** option displays all data for the context, including default configurations.

The double hash (##) indicates an unconfigured element or a dynamic default.

```
*[ex:configure router "Base" bgp]
A:admin@node-2# info detail
 ## apply-groups
    admin-state enable
 ## description
    connect-retry 90
    keepalive 30
    damping false
    local-preference 250
    loop-detect ignore-loop
    loop-detect-threshold 0
    selective-label-ipv4-install false
    min-route-advertisement 30
    aggregator-id-zero false
    preference 170
    block-prefix-sid false
 ## multihop
 ## med-out
 ## authentication-key
    client-reflect true
    vpn-apply-export false
    vpn-apply-import false
    asn-4-byte true
```

When using the **info** command with both **detail** and **json** options, the output does not include unconfigured items. Unconfigured items in the MD-CLI are denoted with ##, and there is no standard method of supplying comments within the JSON format.

```
*[ex:configure router "Base" bgp]
A:admin@node-2# info detail json
{
    "admin-state": "enable",
    "connect-retry": 90,
    "keepalive": 30,
    "damping": false,
    "local-preference": 250,
    "loop-detect": "ignore-loop",
    "loop-detect-threshold": 0,
    "selective-label-ipv4-install": false,
    "min-route-advertisement": 30,
    "aggregator-id-zero": false,
    "preference": 170,
    "block-prefix-sid": false,
    "client-reflect": true,
    "vpn-apply-export": false,
    "vpn-apply-import": false,
    "asn-4-byte": true,
```

The **flat** option displays the context of every element in the present working context on a single line. Braces ensure that the context stays in the present working context for copy and paste purposes.

```
*[ex:configure router "Base" bgp]
A:admin@node-2# info flat detail
 ## apply-groups
    admin-state enable
 ## description
    connect-retry 90
    keepalive 30
    damping false
    local-preference 250
...
 ## route-target-list
 ## outbound-route-filtering
 ## monitor
    rib-management { }
    rib-management { ipv4 }
    rib-management { ipv4 route-table-import }
 ## rib-management ipv4 route-table-import apply-groups
 ## rib-management ipv4 route-table-import policy-name
...
    rib-management { label-ipv4 }
    rib-management { label-ipv4 route-table-import }
 ## rib-management label-ipv4 route-table-import apply-groups
 ## rib-management label-ipv4 route-table-import policy-name
...
 ## segment-routing apply-groups
    segment-routing admin-state disable
 ## segment-routing prefix-sid-range
 ## group
 ## neighbor
```

The **full-context** option displays the full context of every element from the present working context on a single line.

```
*[ex:configure router "Base" bgp]
A:admin@node-2#  info full-context
    /configure router "Base" bgp connect-retry 90
    /configure router "Base" bgp local-preference 250
    /configure router "Base" bgp add-paths { }
    /configure router "Base" bgp add-paths ipv4 receive true

*[ex:configure router "Base" bgp]
A:admin@node-2# info full-context detail
 ## /configure router "Base" bgp apply-groups
    /configure router "Base" bgp admin-state enable
 ## /configure router "Base" bgp description
    /configure router "Base" bgp connect-retry 90
    /configure router "Base" bgp keepalive 30
    /configure router "Base" bgp damping false
    /configure router "Base" bgp local-preference 250
    /configure router "Base" bgp loop-detect ignore-loop
    /configure router "Base" bgp loop-detect-threshold 0
    /configure router "Base" bgp selective-label-ipv4-install false
    /configure router "Base" bgp min-route-advertisement 30
    /configure router "Base" bgp aggregator-id-zero false
    /configure router "Base" bgp preference 170
    /configure router "Base" bgp block-prefix-sid false
 ## /configure router "Base" bgp multihop
 ## /configure router "Base" bgp med-out
 ## /configure router "Base" bgp authentication-key
    /configure router "Base" bgp client-reflect true
```

3.15.1.1 Displaying Lists

The **info** command always displays all keys of the list on the same line. The first key of a list is unnamed in the MD-CLI, however, there are exceptions where the key is named and must be entered. (Refer to the online help for the correct syntax of the command, or the *MD-CLI Command Reference Guide*.) All other keys are named. For example, the **collector** list has two keys, **ip-address** and **port**. The name of the first key, **ip-address**, does not appear in the **info** display. The name of the second key and any subsequent keys are always displayed.

```
*(ex)[configure cflowd]
A:admin@node-2# info
    collector 10.10.20.30 port 7 {
    }
    collector 10.10.30.40 port 8 {
    }

*(ex)[configure cflowd]
A:admin@node-2#
```

3.15.2 Using the show Command

The classic CLI **show** commands can be used in the MD-CLI as well as in the classic CLI, in the following ways:

- use **/show** or **show** (while in the operational root []) in the MD-CLI engine
- use **show** in the classic CLI engine
- use **//** in the MD-CLI engine to switch to the classic CLI engine, then use **show** in the classic CLI engine
- use **//show** in the MD-CLI engine to execute **/show** in the classic CLI engine and switch back to the MD-CLI

3.15.2.1 Classic CLI Command Availability

Classic CLI commands that are accessible in the MD-CLI show outputs of the same information and provide the same functionality in the MD-CLI as they do in the classic CLI. No additional outputs or enhancements are included in the MD-CLI.

Note: Follow the classic CLI context when using the **show** command. For example, route policy information is displayed using the **show router policy** command in both the MD-CLI and classic CLI engines, even though this information is configured in the **configure policy-options** context in the MD-CLI and in the **configure router policy-options** context in the classic CLI.

3.15.2.1.1 Classic CLI show commands not available in the MD-CLI

The following classic CLI show commands are currently blocked in the MD-CLI:

- show alias
- show bof
- show config
- show debug
- show system candidate
- show system rollback

3.15.2.2 Using the show Command in the MD-CLI Engine

The **show** command in the MD-CLI is applicable only in the operational root []. The **/show** command can be used from the root or any configuration context.

```
(ex)[]
A:admin@node-2# show port port-id 1/1/1 description

===============================================================================
Port Descriptions on Slot 1
===============================================================================
Port Id        Description
-------------------------------------------------------------------------------
1/1/1          10-Gig Ethernet
===============================================================================

(ex)[]
A:admin@node-2# configure router

(ex)[configure router "Base"]
A:admin@node-2# show port port-id 1/1/1 description
                ^^^^
MINOR: MGMT_CORE #2201: Unknown element - 'show'

(ex)[configure router "Base"]
A:admin@node-2# /show port port-id 1/1/1 description
===============================================================================
Port Descriptions on Slot 1
===============================================================================
Port Id        Description
-------------------------------------------------------------------------------
1/1/1          10-Gig Ethernet
===============================================================================
```

3.15.3 Using Output Modifiers

Output modifiers (**match**, **count**, and **no-more**) can also be used with the **show** command. See Using Output Modifiers in the MD-CLI.

3.16 MD-CLI Admin Tree

The **admin** commands are available only in the operational mode of the MD-CLI, or they can be executed with **/admin** from a configuration branch.

See the *MD-CLI Command Reference Guide* for information about the **admin** commands in the MD-CLI.

The following outputs show the **admin show configuration** command for the default configuration region (**configure**).

```
[]
A:admin@node-2# admin show

[admin show]
A:admin@node-2# configuration

...

configure {
    card 1 {
        card-type imm48-sfp+2-qsfp28
        mda 1 {
        }
    }
    card 2 {
        card-type imm36-100g-qsfp28
        mda 1 {
        }
    }
    log {
        accounting-policy 12 {
        }
        log-events {
            port event tmnxPortEtherLoopbackStart {
                severity major
            }
            port event tmnxPortEtherLoopbackStop {
                severity major
            }
        }
        filter 1001 {
            entry 10 {
                description "Collect only events of major severity or higher"
                action forward
                match {
                    severity {
                        gte major
                    }
                }
            }
        }
        log-id 99 {
            description "Default System Log"
            source {
                main true
            }
            destination {
                memory {
                    max-entries 500
                }
            }
        }
        log-id 100 {
            description "Default Serious Errors Log"
            filter 1001
```

```
            source {
                main true
            }
            destination {
                memory {
                    max-entries 500
                }
            }
        }
    }
    port 1/1/1 {
    }
    port 1/1/2 {
    }
    port 1/1/3 {

...

[admin show]
A:admin@node-2# configuration json
{
    "nokia-conf:configure": {
        "card": [
            {
                "slot-number": 1,
                "card-type": "imm48-sfp+2-qsfp28",
                "mda": [
                    {
                        "mda-slot": 1
                    }
                ]
            },
            {
                "slot-number": 2,
                "card-type": "imm36-100g-qsfp28",
                "mda": [
                    {
                        "mda-slot": 1
                    }
                ]
            }
        ],
        "log": {
            "accounting-policy": [
                {
                    "policy-id": 12
                }
            ],
            "log-events": {
                "port": [
                    {
                        "event": "tmnxPortEtherLoopbackStart",
                        "severity": "major"
                    },
                    {
                        "event": "tmnxPortEtherLoopbackStop",
                        "severity": "major"
                    }
                ]
```

```
                },
                "filter": [
                    {
                        "filter-id": 1001,
                        "entry": [
                            {
                                "entry-id": 10,
                                "description": "Collect only events of major severity or
higher",
                                "action": "forward",
                                "match": {
                                    "severity": {
                                        "gte": "major"
                                    }
                                }
                            }
                        ]
                    }
                ],
                "log-id": [
                    {
                        "id": 99,
                        "description": "Default System Log",
                        "source": {
                            "main": true
                        },
                        "destination": {
                            "memory": {
                                "max-entries": 500
                            }
                        }
                    },
                    {
                        "id": 100,
                        "description": "Default Serious Errors Log",
                        "filter": 1001,
                        "source": {
                            "main": true
                        },
                        "destination": {
                            "memory": {
                                "max-entries": 500
                            }
                        }
                    }
                ]
            },
            "port": [
                {
                    "port-id": "1/1/1"
                },
                {
                    "port-id": "1/1/2"
                },
                {
                    "port-id": "1/1/3"
                },

      ...
```

4 Configuring in the MD-CLI

4.1 Configuration Workflow

4.1.1 MD-CLI Session Modes

There are two modes in the MD-CLI:

- operational — a user can run all commands to monitor or troubleshoot the router, but the router configuration cannot be changed
- configuration — a user can run all commands to monitor or troubleshoot the router. In private, exclusive, or global configuration mode, the router configuration can be changed. In read-only configuration mode, the user can only view the router configuration.

The first line of the user prompt indicates the active configuration mode. For example:

- [**pr**:configure] — indicates a user in private configuration mode (implicit configuration workflow)
- (**ex**) [configure] — indicates a user in exclusive configuration mode (explicit configuration workflow)

At login, an MD-CLI session always starts in operational mode. To configure the router, the user must enter a configuration mode using the explicit or implicit configuration workflow.

The configuration workflow (implicit vs explicit) determines if the user is restricted to the configure branch or if the user can navigate freely while in configuration mode. Configuration workflows are detailed in Implicit and Explicit Configuration Workflows.

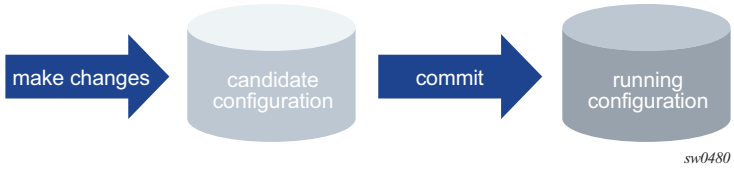
The configuration mode (private, exclusive, global, or read-only) determines the interaction with other simultaneous configuration sessions. Candidate configuration modes are detailed in Candidate Configuration Modes.

4.1.2 Transactional Configuration Method

The MD-CLI transactional configuration method is a two-step process in which configuration changes are made in a candidate configuration. When the configuration is committed, the changes are copied to the running configuration and become active.

Figure 1 shows the flow of configuration changes from the candidate configuration to the running configuration.

Figure 1 **Flow of Configuration Changes**



Other non-router configuration operations, such as changing the MD-CLI session environment are active immediately.

The MD-CLI configuration method differs from the classic CLI in the following ways:

- In the classic CLI, changes to the router configuration are immediately activated in the running configuration. A strict configuration order must be maintained or the configuration fails.
- In the MD-CLI, the transactional configuration method allows multiple configuration changes to be made in any order in the candidate configuration. The system applies the correct ordering when the configuration is activated with the **commit** command.

4.1.3 Implicit and Explicit Configuration Workflows

The MD-CLI supports two configuration workflows:

- Implicit configuration workflow
  - Navigation is restricted to the **configure** branch and its descendants.
  - Operational commands require an absolute path and error when incomplete.

  - **configure** {**private** | **exclusive** | **global** | **read-only**} enters configuration mode and navigates in the **configure** branch. There is no default configuration mode.
  - **exit all** leaves configuration mode and navigates to the operational root.
- Explicit configuration workflow
  - Navigation is unrestricted while in configuration mode.
  - Operational commands while in the **configure** branch require an absolute path and navigate when incomplete.
  - **edit-config** {**private** | **exclusive** | **global** | **read-only**} enters configuration mode without navigating. There is no default configuration mode.
  - **quit-config** leaves configuration mode without navigating. The **quit-config** command is not available in the **configure** branch.

Table 15 compares the implicit and explicit configuration workflows.

Table 15 **Implicit and Explicit Configuration Mode Features**

	Implicit Configuration Workflow	**Explicit Configuration Workflow**
Use	User focused on configuration tasks in the **configure** branch	Power user mode with unrestricted navigation capabilities
Flexibility	Run operational commands or configuration commands from the **configure** branch	Run operational commands or configuration commands anywhere
configure	Enters configuration mode [1] and navigates to the **configure** branch	Navigates to the **configure** branch (after **edit-config** command)
edit-config	Not applicable	Enters configuration mode [1]
exit all or CTRL-z or **/**	Leaves configuration mode and navigates to the operational root	Navigates to the operational root
quit-config	Not applicable	Leaves configuration mode
Commands that result in an action or display output	Execute the command	Execute the command
Commands that navigate out of the **configure** branch	Not allowed	Navigate

Table 15 **Implicit and Explicit Configuration Mode Features (Continued)**

	Implicit Configuration Workflow	**Explicit Configuration Workflow**
info and configuration commands in the **configure** branch	Allowed	Allowed
info and configuration commands out of the **configure** branch	Not allowed	Allowed

Notes:

1. Requires specifying the configuration mode (private | exclusive | global | read-only)

4.1.3.1 Using the Implicit Configuration Workflow

In the implicit configuration workflow, navigation while in configuration mode is restricted to the **configure** branch and its descendants.

The **configure** {**private** | **exclusive** | **global** | **read-only**} command places the user session in the specified configuration mode and navigates to the top of the configuration tree (/**configure**). The first line of the session prompt indicates the configuration mode prepended to the context and separated with a colon.

```
[]
A:admin@node-2# configure exclusive
INFO: CLI #2060: Entering exclusive configuration mode
INFO: CLI #2061: Uncommitted changes are discarded on configuration mode exit

[ex:configure]
A:admin@node-2#
```

When the MD-CLI session is in operational mode, the **configure** command only accepts a configuration mode parameter and cannot be followed by a path to navigate nor by a configuration element to edit the router configuration.

```
[]
A:admin@node-2# configure exclusive router
                                    ^^^^^^
MINOR: CLI #2069: Operation not allowed - currently in operational mode

[]
A:admin@node-2#
```

The following navigation commands leave configuration mode if they cause navigation outside the configuration branch.

- **back**, or **back** with a number greater than the present working context depth
- **exit**, or **exit all**
- CTRL-z
- **/**
- **}**

```
[ex:configure router "Base"]
A:admin@node-2# exit all
INFO: CLI #2064: Exiting exclusive configuration mode

[]
A:admin@node-2#
```

Commands that do not navigate outside the configure branch or that result in an action or display output are allowed.

```
[ex:configure]
A:admin@node-2# /show uptime
System Up Time         : 3 days, 00:27:49.35 (hr:min:sec)

[ex:configure]
A:admin@node-2#


[ex:configure]
A:admin@node-2# /environment more false

[ex:configure]
A:admin@node-2#
```

Commands that navigate out of a configure branch are not allowed.

```
[ex:configure]
A:admin@node-2# /show router
MINOR: CLI #2069: Operation not allowed -
  cannot navigate out of configuration  region

[ex:configure]
A:admin@node-2#


[ex:configure]
A:admin@node-2# /tools dump
MINOR: CLI #2069: Operation not allowed -
 cannot navigate out of configuration  region

[ex:configure]
A:admin@node-2#
```

4.1.3.2 Using the Explicit Configuration Workflow

In the explicit configuration workflow, navigation while in configuration mode is unrestricted. Operational and configuration commands can be executed from any context.

The **edit-config** {**private** | **exclusive** | **global** | **read-only**} command places the user session in the specified configuration mode. The present working context is not changed. The first line of the session prompt indicates the configuration mode between round brackets.

```
[show router]
A:admin@node-2# edit-config exclusive
INFO: CLI #2060: Entering exclusive configuration mode
INFO: CLI #2061: Uncommitted changes are discarded on configuration mode exit

(ex)[show router]
A:admin@node-2#
```

When the MD-CLI session is in configuration mode, the **configure** command can be followed by a path to navigate or by a configuration element to edit the router configuration.

```
(ex)[]
A:admin@node-2# show router

(ex)[show router]
A:admin@node-2# /configure system time zone standard name utc

*(ex)[show router]
A:admin@node-2# /configure router

*(ex)[configure router "Base"]
A:admin@node-2#
```

Commands that result in an action or display output can be executed in the configure branch. Navigation outside the configure branch is allowed and does not exit the configuration mode.

```
(ex)[configure router "Base"]
A:admin@node-2# /show uptime
System Up Time         : 8 days, 23:16:45.01 (hr:min:sec)

(ex)[configure router "Base"]
A:admin@node-2# /tools

(ex)[tools]
A:admin@node-2#
```

Configuration commands, such as **info** and **commit**, can be executed outside the **configure** branch.

```
(ex)[tools]
A:admin@node-2# info
    configure {
        log {

--- snip ---

(ex)[tools]
A:admin@node-2# commit

(ex)[tools]
A:admin@node-2#
```

The **quit-config** command exits configuration mode and places the session in operational mode. The **quit-config** command must be executed from the operational root. The present working context does not change.

```
(ex)[tools]
A:admin@node-2# exit all

(ex)[]
A:admin@node-2# quit-config
INFO: CLI #2064: Exiting exclusive configuration mode

[]
A:admin@ndoe-2#
```

4.1.3.3 Transitioning from an Implicit to an Explicit Configuration Workflow

An MD-CLI configuration session can transition from an implicit to an explicit configuration workflow using the **edit-config** command while in configuration mode.

```
[]
A:admin@node-2# configure exclusive
INFO: CLI #2060: Entering exclusive configuration mode
INFO: CLI #2061: Uncommitted changes are discarded on configuration mode exit
INFO: CLI #2075: Other configuration sessions are active

[ex:configure]
A:admin@node-2# /show
MINOR: CLI #2069: Operation not allowed -
 cannot navigate out of configuration  region

[ex:configure]
A:admin@node-2# edit-config exclusive

(ex)[configure]
A:admin@node-2# /show

(ex)[show]
A:admin@node-2#
```

Transitioning from an explicit to an implicit configuration workflow is not supported.

4.2 Candidate Configuration Modes

To configure the router using the MD-CLI, the user must enter a configuration mode using the explicit or implicit configuration workflow.

The configuration workflow (implicit vs explicit) determines if the user is restricted to the configure branch or if the user can navigate freely while in configuration mode. For more detailed information about configuration workflows, see Implicit and Explicit Configuration Workflows.

The configuration mode determines the interaction with other simultaneous configuration sessions. Table 16 provides an overview of the available configuration modes:

- private configuration mode — see Private Configuration Mode for details
- exclusive configuration mode — see Exclusive Configuration Mode for details
- global configuration mode — see Global Configuration Mode for details
- read-only configuration mode — see Read-Only Configuration Mode for details

Table 16 **Configuration Mode Overview**

	Private Configuration Mode	**Exclusive Configuration Mode**	**Global Configuration Mode**	**Read-only Configuration Mode**
Candidate configuration accessed	Private candidate configuration	Global candidate configuration	Global candidate configuration	Global candidate configuration
Single vs multiple users	Multiple users can simultaneously configure their own private candidate	Only one user can configure the global candidate	Multiple users can simultaneously configure the shared global candidate	Multiple users can have simultaneous read-only access to the global candidate
Privacy	User can see own changes. Changes are not visible for read-only sessions.	User can see own changes. Changes are visible for read-only sessions.	User can see changes from other global configuration sessions. Changes are visible for read-only sessions.	Users can see changes from global or exclusive configuration sessions

Table 16 **Configuration Mode Overview (Continued)**

	Private Configuration Mode	Exclusive Configuration Mode	Global Configuration Mode	Read-only Configuration Mode
Commits	Own changes are committed	Own changes are committed. Commits from other configuration changes are blocked.	Changes made by all global configuration sessions are committed	Users cannot commit
Update needed?	Yes - baseline can become out-of-date when another private or global configuration session commits	No - baseline is always up-to-date. Other configuration sessions cannot commit.	Yes - baseline can become out-of-date when a private configuration session commits	No - updates are not allowed in read-only configuration mode

Note: Private configuration mode should not be used in the MD-CLI when the router is also configured using NETCONF or gRPC for the following reasons:

- private candidate configurations are not visible over NETCONF or gRPC
- an equivalent function of the MD-CLI **update** command to manage an out-of-date baseline is not available in NETCONF or gRPC

4.2.1 Multiple Simultaneous Candidate Configurations

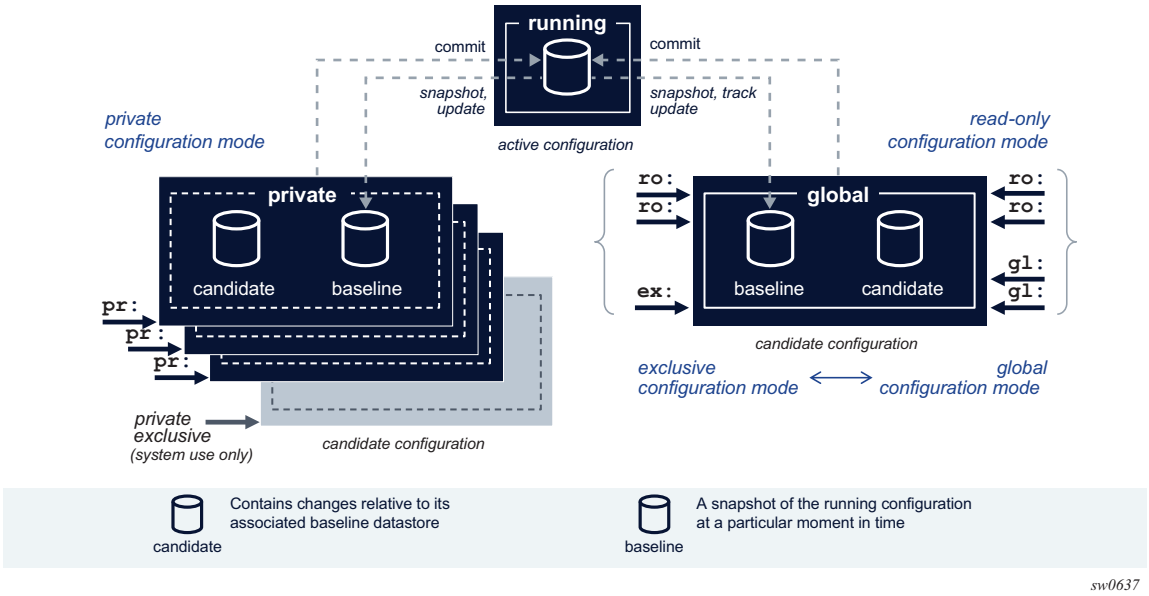
As introduced in Transactional Configuration Method, configuration changes are made in a candidate configuration and copied in the running configuration when the configuration changes are committed and become active.

This section describes:

- how the running configuration and a candidate configuration interact using a running datastore, a baseline datastore, and a candidate datastore
- how simultaneous configuration sessions access one or multiple candidate configurations as a function of their configuration mod

Figure 2 shows multiple candidate configurations.

Figure 2 **Multiple Candidate Configurations**



The running configuration is the active configuration of the router and is stored in the running datastore. There is only one running configuration in the router and therefore, only one running datastore. The running datastore is always instantiated.

The candidate configuration is a working configuration that contains changes before they are activated in the router. A candidate configuration uses two datastores:

- a baseline datastore that contains a snapshot copy of the running datastore at a given moment in time
- a candidate datastore that contains changes relative to its associated baseline datastore

Multiple candidate configurations can exist simultaneously in the router with one of the following:

- a single global candidate configuration that is accessed by one of the following:
  - a single session in exclusive configuration mode
  - one or multiple sessions in global configuration mode
  - one or multiple sessions in read-only configuration mode

  An exclusive configuration session is mutually exclusive with a global configuration session. Read-only configuration sessions can co-exist with an exclusive configuration session or with one or multiple global configuration sessions.

The global baseline datastore and global candidate datastore are always instantiated.

- up to eight private candidate configurations. A private candidate configuration is accessed by a single session in private configuration mode. The private baseline datastore and private candidate datastore are instantiated when the user enters the private configuration mode and the datastores are deleted from the router when the user exits the private configuration mode.
- one single private exclusive candidate configuration for system use only. Only one exclusive session can be active in the router at a time: either a user-started exclusive configuration session accessing the global candidate configuration, or a system-started private exclusive configuration session accessing a private candidate configuration. For more information, see Exclusive Private Configuration Session.

When a configuration session commits its candidate configuration, the router performs the following actions:

- verifies the running configuration has not been changed by another configuration session
- validates the candidate configuration by verifying the logic, constraints, and completeness of the candidate configuration
- activates the candidate configuration by sending the new candidate configuration to the corresponding applications

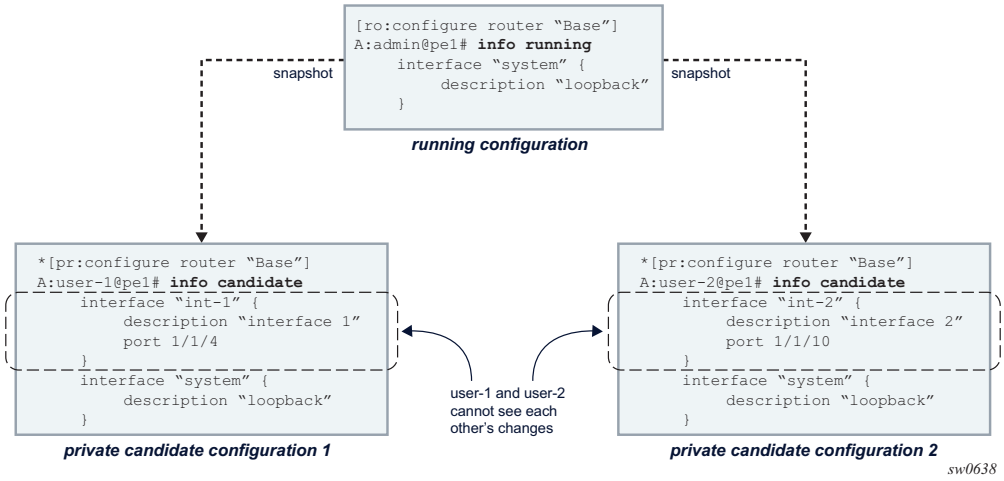
After a successful commit, the changes are copied to the running datastore, the baseline datastore contains a new copy of the running datastore, and the candidate datastore is empty.

Furthermore, when simultaneous configuration sessions access different candidate configurations:

- multiple private configuration sessions each access their own private candidate configuration
- one or multiple private configuration sessions each access their own private candidate configuration and one or multiple global configuration sessions all accessing the global candidate configuration
- one or multiple private configuration sessions each access their own private candidate configuration and one exclusive configuration session accessing the global candidate configuration
- one or multiple private configuration sessions each access their own private candidate configuration and one private exclusive configuration session accessing a private candidate configuration

Each configuration session adds changes in the candidate datastore relative to the baseline associated with the candidate configuration. The baseline datastore contains a snapshot copy of the running datastore at a given time. Therefore, multiple, simultaneous configuration sessions that are active in the router and that access different candidate configurations have their own unique view of the candidate configuration and cannot see other users' changes, as shown in Figure 3.

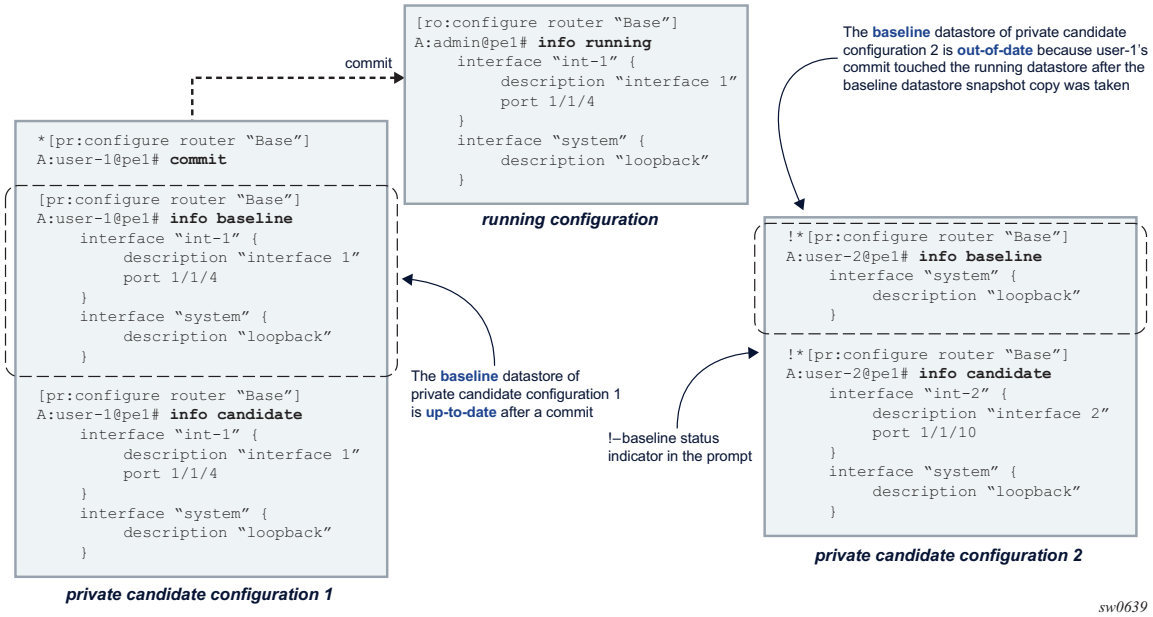
Figure 3 **Simultaneous Configuration Sessions**



Changes in a candidate configuration can only be committed when the running configuration has not been changed or touched after the baseline snapshot was taken. In other words, the baseline must be up to date to commit the changes.

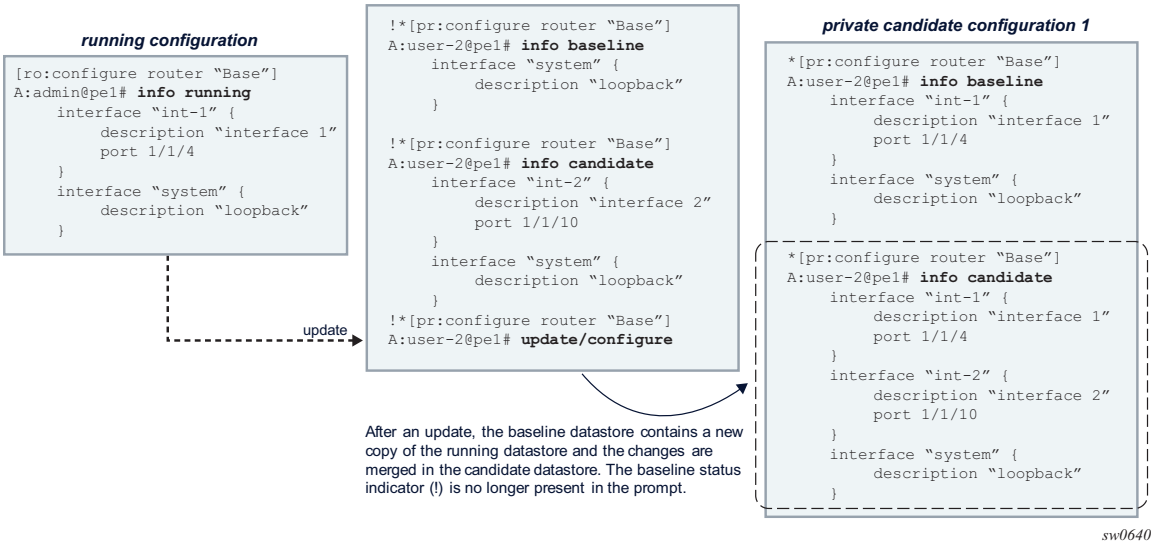
Figure 4 shows how the baseline datastore of user-2's candidate configuration is out-of-date after user-1 committed its changes. An exclamation mark (!) is shown in the prompt to indicate an out-of-date baseline status.

Figure 4 **Simultaneous Configuration Sessions - Baseline Out-of-Date**



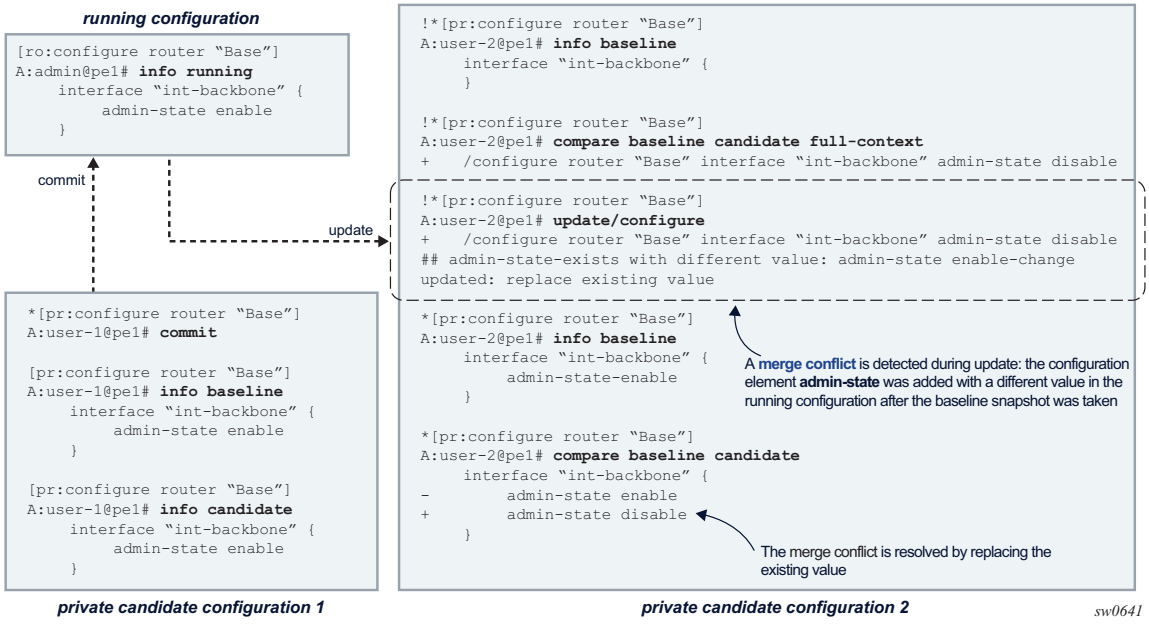
Because the baseline is out-of-date, user-2 must update its candidate configuration before committing. An update copies a new snapshot from the running datastore to the baseline datastore and merges the changes from the candidate datastore, as shown in Figure 5.

Figure 5 **Simultaneous Configuration Sessions - Update**



With more than one user working on the same part of the configuration, conflicts can occur when committed changes of one user's configuration session are merged into another user's candidate configuration. A merge conflict occurs when a configuration element is added, deleted, or modified in the candidate configuration and the same configuration element is also added, deleted, or modified in the running configuration after the baseline snapshot was taken. With the **update** command, the router resolves each merge conflict and installs the result in the candidate configuration, as shown in Figure 6.

Figure 6 **Simultaneous Configuration Sessions - Merge Conflict**



When a commit operation is executed in a configuration session while the baseline is out-of-date, the router first attempts to automatically update the candidate configuration. If a merge conflict is detected, the commit operation is canceled, to allow the administrator to resolve the merge conflicts manually. The candidate configuration remains in the same state as before the commit operation.

In configuration mode, the administrator can use the following tools to check and resolve potential merge conflicts:

- **compare baseline running** - list the changes that were made in the running datastore since a snapshot copy was stored in the baseline datastore
- **compare baseline candidate** or **compare** - list the candidate configuration changes
- **update check** - perform a dry run update. The router reports all merge conflicts as if an update was performed. The candidate configuration, that is, the baseline candidate datastore, is not changed with this command.

Conflict detection and resolution is detailed in Updating the Candidate Configuration.

4.2.2 Private Configuration Mode

In private configuration mode, a private candidate configuration is reserved for editing by a single private configuration session. Each private configuration session works on its own copy of the running configuration. Only the changes made in the private configuration session are visible and can be committed. Private configuration mode can be used when multiple users are configuring simultaneously on different parts of the router configuration.

A private configuration session has the following characteristics:

- Each private configuration session accesses its own private candidate configuration. The private candidate configuration is instantiated when the user enters private configuration mode and is deleted form the router when the user exits private configuration mode.
- Changes can only be entered in its own private candidate configuration.
- Configuration changes are visible only in the private candidate configuration in which the changes are entered.
- Uncommitted changes in the private candidate configuration cannot be seen by other private, exclusive, global, or read-only configuration sessions.
- When the **commit** command is issued, only those changes entered in its own private candidate configuration are committed.
- When a private configuration session is started, a new private candidate configuration is instantiated and has no uncommitted changes.
- When a user leaves private configuration mode, uncommitted changes are discarded and the private candidate configuration is deleted. The user is prompted for confirmation to exit when uncommitted changes are present.

For simultaneous configuration sessions:

- Up to eight simultaneous private configuration sessions can co-exist. Each private configuration session accesses its own private candidate configuration. Private candidate configurations can have uncommitted changes when another private configuration session starts. A private configuration session can edit and commit its private candidate configuration while another private configuration session is active.

- An exclusive configuration session can co-exist with a private configuration session. The private candidate configuration can have uncommitted changes when an exclusive configuration session starts. The exclusive session can edit and commit changes while a private configuration session is active. The private configuration session can still edit the private candidate configuration, but changes cannot be committed because the exclusive session holds a lock on the running datastore.
- Multiple global configuration sessions can co-exist with a private configuration session. A global configuration session accesses the global candidate configuration. The private candidate configuration can have uncommitted changes when the global configuration session starts.
- Multiple read-only configuration sessions can co-exist with a private configuration session. Read-only configuration sessions access the global candidate configuration. A read-only configuration session cannot view the changes in the private candidate configuration. The private candidate configuration can have uncommitted changes when a read-only configuration session starts.

Datastore interactions include the following characteristics:

- The private baseline datastore becomes out-of-date when another private, exclusive, global, or private exclusive configuration session commits changes to the running datastore after the private baseline snapshot was taken. An out-of-date baseline is indicated in the prompt with an exclamation mark.
- An update of the private candidate configuration is needed when its private baseline datastore is out-of-date. An update copies a new snapshot of the running datastore in the private baseline datastore and merges the changes from the private candidate datastore. Merge conflicts detected in a manual update are reported and resolved. Merge conflicts detected in an automatic update as part of a **commit** operation result in the cancellation of the **commit** operation.
- A snapshot of the running datastore is copied in the private baseline datastore:
  - at instantiation of the private candidate configuration when a user enters the private configuration mode
  - when a manual update is performed
  - after a commit, when no merge conflicts are detected during the automatic update and the updated candidate configuration is valid

When entering private configuration mode, the following messages are displayed:

```
[]
A:admin@node-2# configure private
INFO: CLI #2070: Entering private configuration mode
INFO: CLI #2061: Uncommitted changes are discarded on configuration mode exit
INFO: CLI #2075: Other configuration sessions are active
```

Note:

- CLI #2075 is shown only when applicable.
- To display the current active configuration sessions in the router, use the command **show system management-interface configuration-sessions**.

When leaving private configuration mode, the following messages are displayed.

- without uncommitted changes in the private candidate configuration:

```
[pr:configure]
A:admin@node-2# exit all
INFO: CLI #2074: Exiting private configuration mode
```

- with uncommitted changes present in the private candidate configuration:

```
*[pr:configure]
A:admin@node-2# exit all
INFO: CLI #2071: Uncommitted changes are present in the candidate configuration.  Ex
iting private configuration mode will discard those changes.

Discard uncommitted changes? [y,n] n
INFO: CLI #2072: Exit private configuration mode canceled

*[pr:configure]
A:admin@node-2# exit all
INFO: CLI #2071: Uncommitted changes are present in the candidate configuration.  Ex
iting private configuration mode will discard those changes.

Discard uncommitted changes? [y,n] y
WARNING: CLI #2073: Exiting private configuration mode -
 uncommitted changes are  discarded
```

Note: Private configuration mode should not be used in the MD-CLI when the router is also configured using NETCONF or gRPC for the following reasons:

- private candidate configurations are not visible over NETCONF or gRPC
- an equivalent function of the MD-CLI **update** command to manage an out-of-date baseline is not available in NETCONF or gRPC

4.2.3 Exclusive Configuration Mode

In exclusive configuration mode, the global configuration is reserved for editing by a single read-write configuration session. In addition, the running datastore is locked such that no other configuration session can commit changes. Exclusive configuration mode can be used when important router configuration changes must be implemented that cannot be interrupted or delayed, and to avoid the risk of committing other users' partial completed changes.

An exclusive configuration session has the following characteristics:

- An exclusive configuration session accesses the global candidate configuration.
- Only one user can enter exclusive configuration mode at a time.
- Configuration changes in the global candidate can only be entered by the user in exclusive configuration mode.
- Configuration changes in the global candidate are visible for read-only configuration sessions.
- Changes in the global candidate configuration can only be committed by the user in exclusive configuration mode
- Uncommitted changes cannot be present in the global candidate configuration when an exclusive configuration session starts.
- Uncommitted changes are discard from the global candidate configuration when a user leaves the exclusive configuration mode. The user is prompted for confirmation to exit when uncommitted changes are present.

For simultaneous configuration sessions:

- Multiple private configuration sessions can co-exist with an exclusive configuration session. Each private configuration session accesses its own private candidate configuration. The global candidate configuration can have uncommitted changes when a private configuration session starts. A private configuration session can edit its private candidate configuration but cannot commit the changes while an exclusive configuration session is active.
- Only one exclusive configuration session can be active in the router at a time.
- Global configuration sessions are mutually exclusive with an exclusive configuration session.
- Multiple read-only configuration sessions can co-exist with an exclusive configuration session. Read-only configuration sessions access the same global candidate configuration. The global candidate configuration can have uncommitted changes when a read-only configuration session starts.

Datastore interactions include the following characteristics:

- The global baseline datastore is always up to date. Commits from other configuration sessions are blocked while an exclusive configuration session is active.
- An update of the global candidate configuration is not needed in exclusive configuration mode.

When entering exclusive configuration mode, the following messages are displayed:

- with a global configuration session active:

```
[]
A:admin@node-2# configure exclusive
MINOR: MGMT_CORE #2052: Exclusive datastore access unavailable - model-
driven  interface editing global candidate
```

- with uncommitted changes present in the global candidate configuration:

```
[]
A:admin@node-2# configure exclusive
MINOR: MGMT_CORE #2052: Exclusive datastore access unavailable - model-
driven  interface has uncommitted changes in global candidate
```

- with a private configuration session active:

```
[]
A:admin@node-2# edit-config exclusive
INFO: CLI #2060: Entering exclusive configuration mode
INFO: CLI #2061: Uncommitted changes are discarded on configuration mode exit
INFO: CLI #2075: Other configuration sessions are active
```

Note:

- MGMT_CORE #2052 and CLI #2075 are shown only when applicable.
- To display the current active configuration sessions in the router, use the command **show system management-interface configuration-sessions**.

When leaving exclusive configuration mode, the following messages are displayed.

- without uncommitted changes in the global candidate configuration:

```
[ex:configure]
A:admin@node-2# exit all
INFO: CLI #2064: Exiting exclusive configuration mode
```

- with uncommitted changes in the global candidate configuration:

```
*[ex:configure]
A:admin@node-2# exit all
INFO: CLI #2063: Uncommitted changes are present in the candidate configuration.  Ex
iting exclusive configuration mode will discard those changes.

Discard uncommitted changes? [y,n] n
INFO: CLI #2065: Exit exclusive configuration mode canceled
```

```
*[ex:configure]
A:admin@node-2# exit all
INFO: CLI #2063: Uncommitted changes are present in the candidate configuration.  Ex
iting exclusive configuration mode will discard those changes.

Discard uncommitted changes? [y,n] y
WARNING: CLI #2062: Exiting exclusive configuration mode -
 uncommitted changes are  discarded
```

4.2.4 Global Configuration Mode

In global configuration mode, the global configuration is shared with all global configuration sessions. When a user commits their changes, the changes from all users are also committed. Global configuration mode can be used when multiple users are working together on the same part of the router configuration.

A global configuration session has the following characteristics:

- A global configuration session accesses the global candidate configuration.
- Multiple users can enter global configuration mode simultaneously.
- Configuration changes made by one user are visible to all other users in global or read-only configuration mode. Configuration changes in private candidate configurations are not visible.
- All changes in the global candidate configuration, from all users, are committed to the running configuration when a user commits the global candidate configuration.
- Uncommitted changes can be present in the global candidate configuration when a global configuration session starts.
- Uncommitted changes are kept in the global candidate configuration when a user leaves the global configuration mode.

For simultaneous configuration sessions:

- Multiple private configuration sessions can co-exist with a global configuration session. Each private configuration session accesses its own private candidate configuration. The global candidate configuration can have uncommitted changes when a private configuration session starts.
- An exclusive configuration session is mutually exclusive with a global configuration session.
- Multiple global configuration sessions can co-exist. All global configuration sessions access the same global candidate configuration. The global candidate configuration can have uncommitted changes when another global configuration session starts.

- Multiple read-only configuration sessions can co-exist with a global configuration session. Read-only configuration sessions access the same global candidate configuration. The global candidate configuration can have uncommitted changes when a read-only configuration session starts.

Datastore interactions include the following characteristics:

- The global baseline datastore becomes out-of-date when another private or private exclusive configuration session commits changes to the running datastore after the global baseline snapshot was taken. An out-of-date baseline is indicated in the prompt with an exclamation mark.
- An update of the global candidate configuration is needed when its global baseline datastore is out-of-date. An update copies a new snapshot of the running datastore in the global baseline datastore and merges the changes from the global candidate datastore. Merge conflicts detected in a manual update are reported and resolved. Merge conflicts detected in an automatic update as part of a **commit** operation result in the cancellation of the **commit** operation.
- The baseline datastore tracks the running datastore, that is, changes in the running datastore are automatically copied in the baseline datastore:
  - after a router reboot
  - after a successful commit
  - after a discard with an up to date global baseline
- A snapshot copy of the running datastore is copied in the global baseline datastore and tracking stops when the global candidate is touched, for example, when a configuration element has been added, deleted, or modified. A new snapshot of the running datastore is copied to the global baseline datastore when a manual update is performed.

When entering global configuration mode, the following messages are displayed:

```
[]
A:admin@node-2# configure global
INFO: CLI #2054: Entering global configuration mode
INFO: CLI #2055: Uncommitted changes are present in the candidate configuration
INFO: CLI #2075: Other configuration sessions are active
```

Note:

- CLI #2055 and CLI #2075 are shown only when applicable.
- To display the current active configuration sessions in the router, use the command **show system management-interface configuration-sessions**.

When leaving global configuration mode, the following messages are displayed.

```
*[gl:configure]
```

```
A:admin@node-2# exit all
INFO: CLI #2056: Exiting global configuration mode
INFO: CLI #2057: Uncommitted changes are kept in the candidate configuration
```

Note: CLI #2057 is shown only when applicable.

4.2.5 Read-Only Configuration Mode

In read-only configuration mode, no changes can be made to the global candidate configuration and no changes can be committed to the running configuration. Read-only configuration mode can be used when reviewing or monitoring configuration changes from other users in the global candidate configuration.

A read-only configuration session has the following characteristics:

- A read-only configuration session accesses the global candidate configuration.
- Multiple users can enter read-only configuration mode simultaneously.
- All configuration changes in the global candidate configuration are visible. Configuration changes in private candidate configurations are not visible.
- The global configuration cannot be edited and changes in the global configuration cannot be committed.
- Uncommitted changes can be present in the global candidate configuration when a read-only configuration session starts.
- Uncommitted changes are kept in the global candidate configuration when a user leaves a read-only configuration mode.

For simultaneous configuration sessions:

- Multiple private configuration sessions can co-exist with a read-only configuration session. Each private configuration session accesses its own private candidate configuration. The global candidate configuration can have uncommitted changes when a private configuration session starts.
- An exclusive configuration session can co-exist with a read-only configuration session. The exclusive configuration session accesses the same global candidate configuration. The global candidate configuration cannot have uncommitted changes when an exclusive configuration session starts.
- Multiple global configuration sessions can co-exist with a read-only configuration session. Global configuration sessions access the same global candidate configuration. The global candidate configuration can have uncommitted changes when another global configuration session starts.

- Multiple read-only configuration sessions can co-exist. Read-only configuration sessions access the same global candidate configuration. The global candidate configuration can have uncommitted changes when another read-only configuration session starts.

When entering read-only configuration mode, the following message is displayed:

```
[]
A:admin@node-2# configure read-only
INFO: CLI #2066: Entering read-only configuration mode
```

When leaving read-only configuration mode, the following message is displayed.

```
*[ro:configure]
A:admin@node-2# exit all
INFO: CLI #2067: Exiting read-only configuration mode
```

4.2.6 Transitioning Between Candidate Configuration Modes

Exclusive, global, and read-only configuration sessions that access the global candidate configuration can transition between these configuration modes without exiting and re-entering the configuration mode.

Transitions from and to private configuration mode are not allowed.

Figure 7 summarizes the configuration mode transitions and transitions to operational mode.

Figure 7 **Configuration and Operational Mode Transitions**

Configuration and Operational Mode Transition		To				
		Global	Exclusive	Read-Only	Private	Operational Mode
From	Global	X[1]	Allowed, no other exclusive or global configuration session can be active, uncommitted changes are kept	Allowed, uncommitted changes are kept	X	Allowed, uncommitted changes are kept
	Exclusive	Allowed, uncommitted changes are discarded	X[1]	Allowed, uncommitted changes are discarded	X	Allowed, uncommitted changes are discarded
	Read-Only	Allowed, no exclusive configuration session can be active, uncommitted changes are kept	Allowed, no other exclusive or global configuration session can be active, uncommitted changes are kept	X[1]	X	Allowed, uncommitted changes are kept
	Private	X	X	X	X[1]	Allowed, uncommitted changes are discarded
	Operational Mode	Allowed	Allowed	Allowed	Allowed	X

Note: 1. Allowed, but no functional value.

sw0654

Transitioning from exclusive to global or read-only configuration mode causes the candidate changes to be discarded.

```
[]
A:admin@node-2# edit-config exclusive
INFO: CLI #2060: Entering exclusive configuration mode
INFO: CLI #2061: Uncommitted changes are discarded on configuration mode exit

(ex)[]
A:admin@node-2# configure router interface my-int

*(ex)[configure router "Base" interface "my-int"]
A:admin@node-2# edit-config global
INFO: CLI #2063: Uncommitted changes are present in the candidate configuration.  Ex
iting exclusive configuration mode will discard those changes.

Discard uncommitted changes? [y,n] n
INFO: CLI #2065: Exit exclusive configuration mode canceled

*(ex)[configure router "Base" interface "my-int"]
```

```
A:admin@node-2# edit-config read-only
INFO: CLI #2063: Uncommitted changes are present in the candidate configuration.  Ex
iting exclusive configuration mode will discard those changes.

Discard uncommitted changes? [y,n] y
WARNING: CLI #2062: Exiting exclusive configuration mode -
 uncommitted changes are  discarded
INFO: CLI #2066: Entering read-only configuration mode

(ro)[configure router "Base" interface "my-int"]
A:admin@node-2#
```

Switching from global or read-only to exclusive configuration mode is allowed when no other global or exclusive configuration session is active. Uncommitted changes in the global candidate configuration are kept.

In the following example, the **admin disconnect** command is used to disconnect another active configuration session before the current session can switch to exclusive configuration.

```
[]
A:admin@node-2# edit-config global
INFO: CLI #2054: Entering global configuration mode
INFO: CLI #2075: Other configuration sessions are active

(gl)[]
A:admin@node-2# configure router interface new-int

*(gl)[configure router "Base" interface "new-int"]
A:admin@node-2# edit-config exclusive
MINOR: MGMT_CORE #2052: Exclusive datastore access unavailable - model-
driven  interface editing global candidate

*(gl)[configure router "Base" interface "new-int"]
A:admin@node-2# /show system management-interface configuration-sessions
===============================================================================
Session ID  Region                Datastore                    Lock State
   Username                       Session Mode                 Idle Time
   Session Type                   From
-------------------------------------------------------------------------------
#22         configure             Candidate                    Unlocked
   admin                          Global                       0d 00:00:00
   MD-CLI                         135.244.144.235
 23         configure             Candidate                    Unlocked
   user-1                         Global                       0d 00:00:42
   MD-CLI                         135.244.144.235
-------------------------------------------------------------------------------
Number of sessions: 2
'#' indicates the current active session
===============================================================================

*(gl)[configure router "Base" interface "new-int"]
A:admin@node-2#

*(gl)[configure router "Base" interface "new-int"]
A:admin@node-2# /admin disconnect session-id 23
```

```
*(gl)[configure router "Base" interface "new-int"]
A:admin@node-2# edit-config exclusive
INFO: CLI #2056: Exiting global configuration mode
INFO: CLI #2057: Uncommitted changes are kept in the candidate configuration
INFO: CLI #2060: Entering exclusive configuration mode
INFO: CLI #2061: Uncommitted changes are discarded on configuration mode exit

*(ex)[configure router "Base" interface "new-int"]
A:admin@node-2#
```

4.2.7 Exclusive Private Configuration Session

An exclusive private configuration session is reserved for system internal use.

Note: Exclusive private is not a configuration mode: an MD-CLI session cannot enter an exclusive private configuration mode.

Router configuration changes are made via an exclusive private configuration session as a result of the following scenarios:

- the management interface configuration mode is set to **mixed**, with one of the following actions:
  - an SNMP set operation
  - any (immediate) configuration performed in the classic CLI engine
  - a gNMI configuration operation
- the management interface configuration mode is set to **model-driven**, with the following action:
  - a gNMI configuration operation

It is important to be aware that an exclusive private configuration session can exist, as it interacts with other active configuration sessions in the following ways:

- An exclusive configuration session and a private exclusive configuration session are mutually exclusive, as they both require a lock on the running datastore.
- The global candidate configuration and private candidate configurations can become out-of-date when changes are committed via an exclusive private configuration session.
- Commits from global and private configuration sessions are blocked when an exclusive private configuration session is active.

- An exclusive private configuration session accesses its own private candidate configuration. Changes are not visible to other configuration sessions until they are committed and become active in the running configuration.

4.2.8 Restricting Configuration Mode Sessions

It may be desirable to deny a user the ability to use certain configuration modes. For example, denying the use of exclusive configuration mode prevents the user from locking the configuration datastore, or denying the use of the global configuration mode forces the user to work in a private candidate datastore.

It is possible to use AAA to deny access to particular configuration modes, as illustrated in the following configuration example.

In this example, the user *pr-user* has profile *admin-private*. Entries 3 and 4 in the local profile effectively deny users in the *admin-private* profile from entering the exclusive configuration mode in the MD-CLI.

```
[ex:configure system security aaa local-profiles profile "admin-private"]
A:admin@node-2# info detail
## cli-session-group
    default-action permit-all
 ... <snip> ...
    entry 3 {
     ## apply-groups
     ## description
        action deny
        match "edit-config exclusive"
    }
    entry 4 {
     ## apply-groups
     ## description
        action deny
        match "configure exclusive"
    }


[]
A:pr-user@node-2# configure exclusive
MINOR: MGMT_CORE #2020: Permission denied - unauthorized use of 'configure'
[]
A:pr-user@node-2# configure ?

 configure

 Configuration modes:
 global               - Enter global (shared) mode for candidate configuration.
 private              - Enter private mode for candidate configuration.
 read-only            - Enter read-only mode for candidate configuration.

          - Enter a candidate li configuration mode
```

```
[]
A:pr-user@node-2# edit-config exclusive
MINOR: MGMT_CORE #2020: Permission denied - unauthorized use of 'edit-config'

[]
A:pr-user@node-2# edit-config ?

 edit-config

 Configuration modes:
 global               - Enter global (shared) mode for candidate configuration.
 private              - Enter private mode for candidate configuration.
 read-only            - Enter read-only mode for candidate configuration.

 li                   - Enter a candidate li configuration mode
```

The following additional entries to the profile deny users from entering the global configuration mode in the MD-CLI.

```
[ex:configure system security aaa local-profiles profile "admin-pr"]
A:admin@node-2# info detail

 ...<snip>...

    entry 5 {
     ## apply-groups
     ## description
        action deny
        match "configure global"
    }
    entry 6 {
     ## apply-groups
     ## description
        action deny
        match "edit-config global"
    }


[]
A:pr-user@node-2# configure ?

 configure

 Configuration modes:
 private              - Enter private mode for candidate configuration.
 read-only            - Enter read-only mode for candidate configuration.

[]
A:pr-user@node-2# edit-config ?

 edit-config

 Configuration modes:
 private              - Enter private mode for candidate configuration.
 read-only            - Enter read-only mode for candidate configuration.

 li                   - Enter a candidate li configuration mode
```

```
[]
A:pr-user@node-2# configure global
MINOR: MGMT_CORE #2020: Permission denied - unauthorized use of 'configure'

[]
A:pr-user@node-2# edit-config global
MINOR: MGMT_CORE #2020: Permission denied - unauthorized use of 'edit-config'
```

4.3 Modifying the Configuration

To modify the router configuration using the MD-CLI, enter (private, exclusive, or global) configuration mode and use the available configuration commands as described in the *MD-CLI Command Reference Guide*.

To add a new configuration or make changes to the existing configuration, see Adding Configuration Elements. To remove a particular configuration or to return a functionality to its default condition, see Deleting Configuration Elements.

Note: When entering commands in the MD-CLI, whether from a load file or explicitly in the CLI prompt, all input after a hash (#) is treated as a comment and is ignored.

4.4 Adding Configuration Elements

To add configuration statements using the MD-CLI, enter the command or parameter name with a valid value for the parameter as specified by the data type. For some parameters, it is sufficient to type the parameter name to set the parameter configuration.

The current configuration of a parameter is available via the **info detail** command, even if it is the default value or if the parameter is in an unconfigured state (indicated by ##). The display of default values allows an administrator to view the configuration, particularly in a multi-vendor network with different default settings. An operator may choose to explicitly configure a setting that persists rather than using the default, in case the default changes.

Refer to the *MD-CLI Command Reference Guide* for configuration commands and their appropriate syntax.

4.4.1 Default Values for Key Leafs

A leaf is an element that does not contain any other elements and has a data type, for example, a string, an integer, or an IP address.

Key leafs may have an optional default value that can be used as shorthand notation where a certain default is assumed. For example, **configure router bgp** with no instance value expands to **configure router “Base” bgp**. Default values are implemented as follows:

- default values cannot be used in a reference
- multiple keys in a list can have default values
- the first, last, or any key in a list may have a default value
- if the first key has a default value, the other keys must be named keys
- default values can be used multiple times in any combination; for example, **configure router isis** expands to **configure router “Base” isis 0**, and **configure router foo isis** expands to **configure router “foo” isis 0**.
- the expansion is automatic and displayed in the command prompt context and **pwc**

```
(ex)[]
A:admin@node-2# configure router

(ex)[configure router "Base"]
A:admin@node-2#

(ex)[]
A:admin@node-2# configure router isis

(ex)[configure router "Base" isis 0]
A:admin@node-2#

(ex)[]
A:admin@node-2# configure router ospf

(ex)[configure router "Base" ospf 0]
A:admin@node-2#pwc


Present Working Context:
  configure
  router "Base"
  ospf 0

(ex)[configure router "Base" ospf 0]
A:admin@node-2#
```

4.4.2 Entering Integer Values

Integer values can be entered in any of the following formats:

- **decimal**

  Enter an integer (whole number) without spaces; for example, **123456**.
- **binary**

  Enter **0b** followed by the binary value without spaces; for example, **0b1111000100100000**. Negative values are not accepted.
- **hexadecimal**

  Enter **0x** followed by the hexadecimal value in lowercase or uppercase without spaces; for example, **0x1E240** or **0x1e240**. Negative values are not accepted.

Integer values are displayed in decimal format, unless a different output format is specified internally by the system.

```
*(ex)[configure router "Base" bgp]
A:admin@node-2# connect-retry 0b100100101001

*(ex)[configure router "Base" bgp]
A:admin@node-2# info | match connect-retry
    connect-retry 2345

*(ex)[configure router "Base" bgp]
A:admin@node-2# connect-retry 0xd80

*(ex)[configure router "Base" bgp]
A:admin@node-2# info | match connect-retry
    connect-retry 3456

*(ex)[configure router "Base" bgp]
A:admin@node-2#
```

In this example, the **etype** parameter is a hexadecimal output value. A decimal value can be entered, but the value is displayed in hexadecimal format.

```
*[ex:configure filter mac-filter "fn" entry 1 match]
A:admin@node-2# etype ?

 etype <number>
 <number> - <0x600..0xffff>

   Ethernet type

*[ex:configure filter mac-filter "fn" entry 1 match]
A:admin@node-2# etype 65535

*[ex:configure filter mac-filter "fn" entry 1 match]
A:admin@node-2# info
    etype 0xffff
```

Note: Unions of integer and enumerated values do not support binary or hexadecimal input.

In the following example of a command with a union of data types, the **pir** command can have an integer value or it can be defined with the **max** enumerated value. If a numerical value is entered for **pir**, it must be entered as a decimal number.

```
*[ex:configure qos sap-ingress "sstest" queue 8 rate]
A:admin@node-2# pir ?
 pir (<number> | <keyword>)

 <number>  - <1..6400000000> - kilobps
 <keyword> - max             - kilobps
 Default   - max

    Administrative PIR

*[ex:configure qos sap-ingress "sstest" queue 8 rate]
A:admin@node-2# pir 88

*[ex:configure qos sap-ingress "sstest" queue 8 rate]
A:admin@node-2# info
    pir 88

*[ex:configure qos sap-ingress "sstest" queue 8 rate]
A:admin@node-2# pir 0b0010
                   ^^^^^^
MINOR: MGMT_CORE #2301: Invalid element value -
 'pir' expected number  '<1..6400000000>' (kilobps) or keyword 'max' (kilobps)

*[ex:configure qos sap-ingress "sstest" queue 8 rate]
A:admin@node-2# info
    pir 88

*[ex:configure qos sap-ingress "sstest" queue 8 rate]
A:admin@node-2# pir 2

*[ex:configure qos sap-ingress "sstest" queue 8 rate]
A:admin@node-2# info
    pir 2
```

4.4.3 Configuring Lists

A list is a sequence of list entries, and all keys of a list are entered on the same line as the list command. In general, the first key of a list is unnamed in the MD-CLI. All other keys are named. The name of the first key is shown in square brackets in ? help. Entering the name of the first key is optional when it is shown in brackets. In the following example, **ip-address** is the first key and **port** is the second key. Entering **ip-address** in the MD-CLI is optional; entering **port** and any subsequent key names is mandatory.

```
*(ex)[configure cflowd]
A:admin@node-2# collector ?

 [ip-address] (<unicast-ipv4-address> | <global-unicast-ipv6-address>)
 <unicast-ipv4-address>        - <d.d.d.d>
 <global-unicast-ipv6-address> - (<x:x:x:x:x:x:x:x>|<x:x:x:x:x:x:d.d.d.d>)

    IP address of a remote Cflowd collector host to receive the exported Cflowd
    data

*(ex)[configure cflowd]
A:admin@node-2# collector 10.20.30.40 ?

 port <number>
 <number> - <1..65535>

    UDP port number on the remote Cflowd collector host to receive the exported
    Cflowd data
```

The IP address and port number can be entered in one of the following ways:

```
*(ex)[configure cflowd]
A:admin@node-2# collector ip-address 10.10.20.30 port 7

*(ex)[configure cflowd]
A:admin@node-2# collector 10.10.20.30 port 7
```

There are some exceptions where the first key of a list is named. In these cases, the key name must be entered. In the following example, the key name **index** must be entered.

```
*[ex:configure cflowd collector 10.20.30.40 port 7 export-filter interface-
list service]
A:admin@node-2# ies-interface ?

 service-name <reference>
 <reference> - <1..64 characters> - configure service ies <service-name>

    IES service name

*[ex:configure cflowd collector 10.20.30.40 port 7 export-filter interface-
list service]
A:admin@node-2# ies-interface service-name svc-test interface-name ?
```

```
 interface-name <reference>
 <reference> - <1..32            - configure service ies <./service-name>
               characters>         interface <interface-name>

    ies interface name

*[ex:configure cflowd collector 10.20.30.40 port 7 export-filter interface-
list service]
A:admin@node-2# ies-interface service-name svc-test interface-name int-name-test

*[ex:configure cflowd collector 10.20.30.40 port 7 export-filter interface-
list service]
A:admin@node-2# info
    ies-interface service-name "svc-test" interface-name "int-name-test" { }
```

Auto-completion does not select or complete the name of the first key if it is optional. In the following example, the key name for **ma-admin-name** is optional as indicated by the square brackets, and is not auto-completed when Tab is entered.

```
*[ex:configure eth-cfm domain "dmtest"]
A:admin@node-2# association ?

 [ma-admin-name] <string>
 <string> - <1..64 characters>

    Unique domain association name

*[ex:configure eth-cfm domain "dmtest"]
A:admin@node-2# association Press Tab

 <ma-admin-name>
```

If the name of the first key is optional and is not entered as part of the command, the key name can be used as the actual value of the key if it is enclosed in quotation marks.

```
*[ex:configure eth-cfm domain "dmtest"]
A:admin@node-2# association "ma-admin-name"

*[pr:configure eth-cfm domain "dmtest" association "ma-admin-name"]
A:admin@node-2# pwc
Present Working Context:
  configure
  eth-cfm
  domain "dmtest"
  association "ma-admin-name"
```

If the optional key name is entered, it can be specified as the actual value of the key with or without the quotation marks.

```
*[ex:configure eth-cfm domain "dmtest"]
A:admin@node-2# association ma-admin-name ma-admin-name

*[pr:configure eth-cfm domain "dmtest" association "ma-admin-name"]
```

```
A:admin@node-2# pwc
Present Working Context:
  configure
  eth-cfm
  domain "dmtest"
  association "ma-admin-name"
```

4.4.3.1 System-Ordered Lists

For system-ordered lists, list entries are automatically reordered. In the following example, the list is reordered based on the alphabetical order of the string name identifying the list instance.

```
[ex:configure]
A:admin@node-2# eth-cfm ?

 eth-cfm

 domain                + Enter the domain list instance

[ex:configure]
A:admin@node-2# eth-cfm

[ex:configure eth-cfm]
A:admin@node-2# domain ?

 [md-admin-name] <string>
 <string>  - <1..64 characters>

    Unique domain name

[ex:configure eth-cfm]
A:admin@node-2# domain zero } domain two } domain four } domain five }

*[ex:configure eth-cfm]
A:admin@node-2# info
    domain "five" {
    }
    domain "four" {
    }
    domain "two" {
    }
    domain "zero" {
    }
```

4.4.3.2 User-Ordered Lists

For user-ordered lists, new entries are appended to the end of the list.

```
*(ex)[configure policy-options policy-statement "plcy_str" entry 29]
A:admin@node-2# from prefix-list [ plcy5 plcy1 ]
```

```
*(ex)[configure policy-options policy-statement "plcy_str" entry 29]
A:admin@node-2# info
    from {
        prefix-list ["plcy5" "plcy1"]
    }

*(ex)[configure policy-options policy-statement "plcy_str" entry 29]
A:admin@node-2# from prefix-list plcy3

*(ex)[configure policy-options policy-statement "plcy_str" entry 29]
A:admin@node-2# info
    from {
        prefix-list ["plcy5" "plcy1" "plcy3"]
    }

*(ex)[configure policy-options policy-statement "plcy_str" entry 29]
A:admin@node-2# from prefix-list plcy1

*(ex)[configure policy-options policy-statement "plcy_str" entry 29]
A:admin@node-2# info
    from {
        prefix-list ["plcy5" "plcy1" "plcy3"]
    }

*(ex)[configure policy-options policy-statement "plcy_str" entry 29]
A:admin@node-2#
```

To reorder a user-ordered list, the list can be deleted and recreated using the desired order. Alternatively, the tilde (~) character can be used to replace a list, effectively deleting and recreating the leaf-list in one step.

```
(ex)[]
A:admin@node-2# configure router isis 5

*(ex)[configure router "Base" isis 5]
A:admin@node-2# export-policy [test5 test3 test2]

*(ex)[configure router "Base" isis 5]
A:admin@node-2# info
    export-policy ["test5" "test3" "test2"]

*(ex)[configure router "Base" isis 5]
A:admin@node-2# ~ export-policy [test1 test2 test3 test5]

*(ex)[configure router "Base" isis 5]
A:admin@node-2# info
    export-policy ["test1" "test2" "test3" "test5"]

*(ex)[configure router "Base" isis 5]
A:admin@node-2#
```

It is possible to insert entries into an existing user-ordered list by using the **insert** command.

In the following configuration example, the list begins with two entries, named-entry “one” and named-entry “ten”.

```
*[ex:configure policy-options policy-statement "my-ordered-list"]
A:admin@node-2# entry-type named

*[ex:configure policy-options policy-statement "my-ordered-list"]
A:admin@node-2# named-entry one

*[ex:configure policy-options policy-statement "my-ordered-list" named-entry "one"]
A:admin@node-2# back

*[ex:configure policy-options policy-statement "my-ordered-list"]
A:admin@node-2# named-entry ten

*[ex:configure policy-options policy-statement "my-ordered-list" named-entry "ten"]
A:admin@node-2# back

*[ex:configure policy-options policy-statement "my-ordered-list"]
A:admin@node-2# info
    entry-type named
    named-entry "one" {
    }
    named-entry "ten" {
    }
```

The **insert** command is used with the following commands:

```
*[ex:configure policy-options policy-statement "my-ordered-list"]
A:admin@node-2# insert named-entry four ?

 Global commands:
 after                - Insert a named-entry in the user-ordered list after
                        another specified named-entry
 before               - Insert a named-entry in the user-ordered list before
                        another specified named-entry
 beginning            - Insert a named-entry at the beginning of the user-
                        ordered list
 end                  - Insert a named-entry at the end of the user-ordered list

*[ex:configure policy-options policy-statement "my-ordered-list"]
A:admin@node-2# insert named-entry four after one

*[ex:configure policy-options policy-statement "my-ordered-list"]
A:admin@node-2# insert named-entry six before ten

*[ex:configure policy-options policy-statement "my-ordered-list"]
A:admin@node-2# info
    entry-type named
    named-entry "one" {
    }
    named-entry "four" {
    }
    named-entry "six" {
    }
    named-entry "ten" {
    }
```

```
*[ex:configure policy-options policy-statement "my-ordered-list"]
A:admin@node-2# insert named-entry zero beginning

*[ex:configure policy-options policy-statement "my-ordered-list"]
A:admin@node-2# insert named-entry twenty end

*[ex:configure policy-options policy-statement "my-ordered-list"]
A:admin@node-2# info
    entry-type named
    named-entry "zero" {
    }
    named-entry "one" {
    }
    named-entry "four" {
    }
    named-entry "six" {
    }
    named-entry "ten" {
    }
    named-entry "twenty" {
    }
```

List entries can still be deleted.

```
*[ex:configure policy-options policy-statement "my-ordered-list"]
A:admin@node-2# delete named-entry six

*[ex:configure policy-options policy-statement "my-ordered-list"]
A:admin@node-2# info
    entry-type named
    named-entry "zero" {
    }
    named-entry "one" {
    }
    named-entry "four" {
    }
    named-entry "ten" {
    }
    named-entry "twenty" {
    }
```

The default behavior of the **insert** command is to return immediately to the present working context. To drop into the newly-inserted entry, add the **{** symbol, as shown in the following example.

```
*[ex:configure policy-options policy-statement "my-ordered-list"]
A:admin@node-2# insert named-entry five after four {

*[ex:configure policy-options policy-statement "my-ordered-list" named-entry "five"]
A:admin@node-2#
```

4.4.3.3 Special Handling for Lists with all Key Leafs

For lists in which the leafs are all keys (“key-only lists”), the creation of a single entry returns the user to the same context; that is, the MD-CLI session does not enter the context of the list member. This allows the user to enter multiple list items without the need to exit after each item. For example, **station** is a list with a single leaf that is the key. After each **station** entry, the session maintains the same context and other **station** entries can be added without applying the **back** or **exit** command.

```
*[ex:configure router "Base" bgp monitor]
A:admin@node-2# ?

 admin-state          - Administrative state of BMP monitoring
 all-stations         - Send BMP messages to all configured stations
 apply-groups         - Apply a configuration group at this level
 route-monitoring     + Enter the route-monitoring context
 station              - Add a list entry for station

*[ex:configure router "Base" bgp monitor]
A:admin@node-2# station stn1

*[ex:configure router "Base" bgp monitor]
A:admin@node-2# station stn2

*[ex:configure router "Base" bgp monitor]
A:admin@node-2# station stn3

*[ex:configure router "Base" bgp monitor]
A:admin@node-2# info
    station "stn1" { }
    station "stn2" { }
    station "stn3" { }
```

4.4.4 Configuring Leaf-Lists

A leaf-list is an element that contains a sequence of values of a particular data type. Specifying a leaf-list entry in the MD-CLI is additive. New entries are added to existing entries and previous entries are not removed. If a duplicate entry is specified, the order remains. To minimize the number of CLI warnings, no message is displayed.

Single or multiple leaf-list entries can be added in a single command line with the use of brackets ([]).

4.4.4.1 System-Ordered Leaf-Lists

For leaf-lists ordered by the system, the leaf-list entries are automatically reordered, as shown in the following example.

```
*[ex:configure port 1/1/2 ethernet eth-cfm mep md-admin-name "md-test" ma-admin-
name "ma-test" mep-id 8 ais]
A:admin@node-2# client-meg-level ?

 client-meg-level <value>
 client-meg-level [<value>...] - 1..7 system-ordered values separated by spaces
                                 enclosed by brackets

 <value> - <number>
 <number> - <1..7>
    Client MEG level for AIS message generation

*[ex:configure port 1/1/2 ethernet eth-cfm mep md-admin-name "md-test" ma-admin-
name "ma-test" mep-id 8 ais]
A:admin@node-2# client-meg-level [7 5 2 3]

*[ex:configure port 1/1/2 ethernet eth-cfm mep md-admin-name "md-test" ma-admin-
name "ma-test" mep-id 8 ais]
A:admin@node-2# info
    client-meg-level [2 3 5 7]

*[ex:configure port 1/1/2 ethernet eth-cfm mep md-admin-name "md-test" ma-admin-
name "ma-test" mep-id 8 ais]
A:admin@dut-b# client-meg-level [4 6]

*[ex:configure port 1/1/2 ethernet eth-cfm mep md-admin-name "md-test" ma-admin-
name "ma-test" mep-id 8 ais]
A:admin@node-2# info
    client-meg-level [2 3 4 5 6 7]
```

The following system-ordered leaf-list is reordered based on the enumerated value of the entered keywords.

```
*[ex:configure policy-options policy-statement "s" entry 9 from]
A:admin@node-2# family ?

 family <value>
 family [<value>...] - 1..20 system-ordered values separated by spaces enclosed
                       by brackets

 <value> - <keyword>
 <keyword> - (ipv4|vpn-ipv4|ipv6|mcast-ipv4|vpn-ipv6|l2-vpn|mvpn-ipv4|mdt-
             safi|ms-pw|flow-ipv4|route-target|mcast-vpn-ipv4|mvpn-ipv6|
             flow-ipv6|evpn|mcast-ipv6|label-ipv4|label-ipv6|bgp-ls|mcast-
             vpn-ipv6|sr-policy-ipv4|sr-policy-ipv6)

    Match address families to this condition

*[ex:configure policy-options policy-statement "s" entry 9 from]
A:admin@node-2# family [mcast-vpn-ipv4 bgp-ls l2-vpn]
```

```
*[ex:configure policy-options policy-statement "s" entry 9 from]
A:admin@node-2# info
    family [l2-vpn mcast-vpn-ipv4 bgp-ls]
```

4.4.4.2 User-Ordered Leaf-Lists

For user-ordered leaf-lists, new entries are appended to the end of the list.

```
*[ex:configure router "Base" isis 0]
A:admin@node-2# export-policy pol_1

*[ex:configure router "Base" isis 0]
A:admin@node-2# export-policy pol_5

*[ex:configure router "Base" isis 0]
A:admin@node-2# export-policy pol_3

*[ex:configure router "Base" isis 0]
A:admin@node-2# info
    export-policy ["pol_1" "pol_5" "pol_3"]
```

To reorder a user-ordered leaf-list, the leaf-list can be deleted and recreated using the desired order. Alternatively, the tilde (~) character can be used to replace a leaf-list, effectively deleting and recreating the leaf-list in one step.

```
(ex)[]
A:admin@node-2# configure router isis 5

*(ex)[configure router "Base" isis 5]
A:admin@node-2# export-policy [test5 test3 test2]

*(ex)[configure router "Base" isis 5]
A:admin@node-2# info
    export-policy ["test5" "test3" "test2"]

*(ex)[configure router "Base" isis 5]
A:admin@node-2# ~ export-policy [test1 test2 test3 test5]

*(ex)[configure router "Base" isis 5]
A:admin@node-2# info
    export-policy ["test1" "test2" "test3" "test5"]

*(ex)[configure router "Base" isis 5]
A:admin@node-2#
```

4.4.5 Configuring Leafs with Units

If a leaf is defined by a number value and an associated unit, the user can enter the value in a different base unit than is defined. For example, if a timer is defined in seconds, it is possible to enter a value based on the number of minutes, or a combination of minutes and seconds. These dynamic units in the MD-CLI can be entered in a format that is converted into the base unit based on a conversion factor.

Static units that have no conversion factor must always be entered in the base unit value; for example, a unit of packets per second, or bit errors.

Units are supported for:

- memory sizes, for example, bytes
- rates, for example, bps
- durations, for example, seconds
- dates, for example, FRI 11 MAY 2018 15:15:35 UTC

Dynamic units can be entered as a number in one of the following ways:

- as a value without a unit — the value is interpreted as the defined base unit. Decimal, binary, and hexadecimal numbers are supported. For example, **transmit-interval** has a base unit of deciseconds. Entering **transmit-interval 10,** without specifying a unit, configures the interval to 10 deciseconds.

```
*[ex:configure port 1/1/1 ethernet efm-oam]
A:admin@node-2# transmit-interval ?

 transmit-interval <number>
 <number> - <1..600> - deciseconds
 Default  - 10

    Transmit interval of OAMPDUs

*[ex:configure port 1/1/1 ethernet efm-oam]
A:admin@node-2# transmit-interval 10

*[ex:configure port 1/1/1 ethernet efm-oam]
A:admin@node-2# info
    transmit-interval 10
```

- as unique value-unit tuples — the units are separated by a space in any order, and the same unit cannot be used more than once. The value is interpreted as the specified unit and can only be entered as a decimal number. For example, there are many acceptable formats to enter 55 deciseconds for **transmit-interval**, including the following:

```
*[ex:configure port 1/1/1 ethernet efm-oam]
A:admin@node-2# info
    transmit-interval 55
```

```
*[ex:configure port 1/1/1 ethernet efm-oam]
A:admin@node-2# transmit-interval 5 seconds 5 deciseconds

*[ex:configure port 1/1/1 ethernet efm-oam]
A:admin@node-2# info
    transmit-interval 55

*[ex:configure port 1/1/1 ethernet efm-oam]
A:admin@node-2# transmit-interval 5 seconds 500 milliseconds

*[ex:configure port 1/1/1 ethernet efm-oam]
A:admin@node-2# info
    transmit-interval 55
```

The configured value is displayed as a positive integer in the defined base unit. Because the unit for **transmit-interval** is defined as deciseconds, the value displayed in the **info** command is in deciseconds, regardless of the format in which it was entered.

```
*[ex:configure port 1/1/1 ethernet efm-oam]
A:admin@node-2# info
    transmit-interval 55
```

The input value is calculated based on the input of all input tuples and validated after Enter is pressed. For example, entering 900 (deciseconds) for **transmit-interval** results in an error display, as 990 deciseconds is not in the element range.

```
*[ex:configure port 1/1/1 ethernet efm-oam]
A:admin@node-2# transmit-interval 950
                                  ^^^
MINOR: MGMT_CORE #2301: Invalid element value - 900 out of range 1..600
```

Entering a value followed by Space and Tab displays valid units for the value, as in the following example. For a value of 900 for **transmit-interval,** the system displays valid unit possibilities, listed in alphabetical order.

```
*[ex:configure port 1/1/1 ethernet efm-oam]
A:admin@node-2# transmit-interval 900 Press Tab
 milliseconds            centiseconds
```

If a unit is already present in the input, it is suppressed for any further input.

```
*[ex:configure port 1/1/1 ethernet efm-oam]
A:admin@node-2# transmit-interval 900 centiseconds 100 Press Tab

 milliseconds  deciseconds
```

The unit names can be singular or plural, depending on the numerical value entered. For a numerical value of 1, the unit names displayed are their singular form.

```
*[ex:configure port 1/1/1 ethernet efm-oam]
A:admin@node-2# transmit-interval 1 Press Tab
```

```
 decisecond             second                 minute

...

*[ex:configure port 1/1/1 ethernet efm-oam]
A:admin@node-2# transmit-interval 10 Press Tab

 centiseconds           deciseconds            seconds
```

Auto-completion is supported for valid units entered after a value.

Table 17, Table 18, and Table 19 list units that have a conversion factor that allows a leaf with a specific base unit to be defined in a dynamic unit. The valid unit keywords for each unit name are also provided.

Table 17 shows the valid inputs for memory sizes based on the dynamic unit.

Table 17 **Dynamic Units for Memory Sizes**

Unit Name	Valid MD-CLI Input
bytes	• bytes • byte
kilobytes	• kilobytes • kilobytes • kbytes • kbyte
megabytes	• megabytes • megabyte • mbytes • mbyte
gigabytes	• gigabytes • gigabyte • gbytes • gbyte
terabytes	• terabytes • terabyte • tbytes • tbyte

Table 18 shows the valid inputs for rates of speed based on the dynamic unit.

Table 18 **Dynamic Units for Rates**

Unit Name	Valid MD-CLI Input
bps (bits per second)	• bps
kilobps (kilobits per second)	• kilobps • kbps
megabps (megabits per second)	• megabps • mbps
gigabps (gigabits per second	• gigabps • gbps
terabps (terabits per second)	• terabps • tbps
petabps (petabits per second)	• petabps • pbps
exabps (exabits per second)	• exabps • ebps
zettabps (zettabits per second)	• zettabps • zbps

Table 19 shows the valid inputs for time durations based on the dynamic unit.

Table 19 **Dynamic Units for Duration**

Unit Name	Valid MD-CLI Input
picoseconds	• picoseconds • picosecond • psecs • psec
nanoseconds	• nanoseconds • nanosecond • nsecs • nsec
microseconds	• microseconds • microsecond • usecs • usec

Table 19 **Dynamic Units for Duration (Continued)**

Unit Name	Valid MD-CLI Input
milliseconds	• milliseconds • millisecond • msecs • msec
centiseconds	• centiseconds • centisecond • csecs • csec
deciseconds	• deciseconds • decisecond • dsecs • dsec
seconds	• seconds • second • secs • sec
minutes	• minutes • minute • mins • min
hours	• hours • hour • hrs • hr
days	• days • day
weeks	• weeks • week • wks • wk

Table 20 shows the valid inputs for dates based on the time format.

Table 20 **Dynamic Units for Dates**

Time Format	Valid MD-CLI Input
“yyyy-mm-dd hh:mm[:ss] [TZ]” For example: “2018-06-01 13:12:59 EDT”	yyyy is RFC 3339 date-fullyear mm is RFC 3339 date-month dd is RFC 3339 date-mday hh is RFC 3339 time-hour mm is RFC 3339 time-minute, requires preceding zeros ss is RFC 3339 time-second, requires preceding zeros (optional) TZ is the time-zone name (optional) This format follows ISO 9601 and must be enclosed in quotation marks.
“[DAY] dd MON yyyy hh:mm[:ss] [TZ]” For example: “FRI 11 MAY 2018 13:21:11 EDT”	DAY is the name of the day of the week (SUN, MON, TUE, WED, THU, FR, SAT),(optional) dd is RFC 3339 date-mday MON is the name of the month (JAN, FEB, MAR, APR, MAY, JUN, JUL, AUG, SEP, OCT, NOV, DEC) yyyy is RFC 3339 date-fullyear hh is RFC 3339 time-hour mm is RFC 3339 time-minute, requires preceding zeros ss is RFC 3339 time-second, requires preceding zeros (optional) TZ is the time-zone name (optional) This format follows RFC 1123 and must be enclosed in quotation marks.
yyyy-mm-ddThh:mm:ss[.fr][(Z|(+|-)hh:mm)] For example: 2018-05-11T13:21:11-0400 or 2018-05-11T17:21:11Z	This format follows RFC 3339 and can be enclosed in quotation marks.

4.4.6 Flexible Input for MAC and IPv6 Addresses

Flexible input is available for MAC and IPv6 addresses, where both uppercase and lowercase hexadecimal digits are accepted.

This example shows the hexadecimal digits in an IPv6 address entered in both uppercase and lowercase. IPv6 addresses are displayed in lowercase hexadecimal digits using zero compression, according to RFC 5952, *A Recommendation for IPv6 Address Text Representation*.

```
*[ex:configure service vprn "vprn1" dns]
A:admin@node-2# ipv6-source-address 2001:db8:aaa3::8a2e:3710:7335

*[ex:configure service vprn "vprn1" dns]
A:admin@node-2# info
    ipv6-source-address 2001:db8:aaa3::8a2e:3710:7335
```

For MAC addresses, the dash (-) separator can also be used in place of the colon (:).

```
*[ex:configure qos sap-ingress "s" mac-criteria entry 5 match]
A:admin@node-2# dst-mac address aa-BB-cc-DD-eE-Ff

*[ex:configure qos sap-ingress "s" mac-criteria entry 5 match]
A:admin@node-2# info
    dst-mac {
        address aa:bb:cc:dd:ee:ff
```

Flexible input is also available for MAC addresses using dot (.) notation:

```
*(ex)[configure filter mac-filter "str" entry 33 match]
A:admin@node-2# dst-mac address aaBB.ccDD.eEFf

*(ex)[configure filter mac-filter "str" entry 33 match]
A:admin@node-2# info
    dst-mac {
        address aa:bb:cc:dd:ee:ff
    }

*(ex)[configure filter mac-filter "str" entry 33 match]
A:admin@node-2#
```

4.4.7 Input Translation

The MD-CLI supports the following input translation for UTF-8 character encoding:

- curly quotation mark to ASCII quotation mark (")
- curly apostrophe to ASCII apostrophe (')
- hyphens and dashes, including minus sign, en dash, em dash, and others to ASCII hyphen-minus (-)

The input translation allows copy and paste functionality from word processing applications that use UTF-8 curly quotation marks, hyphens, or dashes.

4.5 Deleting Configuration Elements

The **delete** command removes explicit configuration and returns the element configuration to the system default state or value. If there is no defined default for an element, the element returns to an unconfigured state.

Note: The minus sign (**-**) can be used instead of the **delete** command.

The **delete** command can be used to delete any configuration element, such as:

- leafs
- containers
- lists
- leaf-lists

If an element has sub-elements (for example, a container with more containers and leafs), all of the sub-elements are also deleted as part of the parent deletion.

Note: If the configuration element to be removed does not exist, no warning messages are displayed.

4.5.1 Deleting Leafs

The following configuration example deletes three leafs; **admin-state** and **connect-retry** return to their default values, and **description** returns to an unconfigured state.

```
*(ex)[configure router "Base" bgp]
A:admin@node-2# info
    admin-state disable
    description "BGP description"
    connect-retry 65535

*(ex)[configure router "Base" bgp]
A:admin@node-2# delete admin-state

*(ex)[configure router "Base" bgp]
A:admin@node-2# delete description

*(ex)[configure router "Base" bgp]
A:admin@node-2# delete connect-retry
```

```
*(ex)[configure router "Base" bgp]
A:admin@node-2# info detail
    admin-state enable
 ## description
    connect-retry 120
    keepalive 30
    damping false
    local-preference 100
    loop-detect ignore-loop
<snip>
```

4.5.2 Deleting Containers

To remove a container, the **delete** command is specified before the container name. The following examples show the deletion of a **vprn** instance from two different contexts.

This example removes the instance from context **configure service vprn**:

```
*[ex:configure service]
A:admin@node-2# info
    vprn "vprn1" {
        description "VPRN instance 01"
        dns {
            ipv6-source-address 2001:db8:aaa3::8a2e:3710:7335
        }
        bgp {
            min-route-advertisement 50
        }
        interface "int1" {
            ipv6 {
                dhcp6 {
                    relay {
                        server ["2001:db8::" "2001:db9::" "2001:dba::" "2001:dc1::"]
                    }
                }
            }
        }
    }
*[ex:configure service]
A:admin@node-2# delete vprn "vprn1"

[ex:configure service]
A:admin@node-2# info detail | match vprn
 ## vprn
```

This example shows the deletion of the instance from context **configure**:

```
*[ex:configure service]
A:admin@node-2# info
    vprn "vprn1" {
        description "VPRN instance 01"
        dns {
```

```
            ipv6-source-address 2001:db8:aaa3::8a2e:3710:7335
        }
        bgp {
            min-route-advertisement 50
        }
        interface "int1" {
            ipv6 {
                dhcp6 {
                    relay {
                        server ["2001:db8::" "2001:db9::" "2001:dba::" "2001:dc1::"]
                    }
                }
            }
        }
    }

*[ex:configure service]
A:admin@node-2# back

*[ex:configure]
A:admin@node-2# delete service vprn "vprn1"

[ex:configure]
A:admin@node-2# service

[ex:configure service]
A:admin@node-2# info detail | match vprn
 ## vprn
```

In the preceding examples, the container is returned to an unconfigured state, as indicated by the ##.

In the following example, the **timers** element is a container, which contains sub-elements that are also containers; the **lsa-generate** and **spf-wait** elements. The placement of the **delete** command determines whether the **timers** element (and all of its sub-elements) are deleted, or one of the sub-elements.

```
*(ex)[configure router "Base" ospf 0]
A:admin@node-2# info
    timers {
        lsa-generate {
            max-lsa-wait 8000
            lsa-initial-wait 10
            lsa-second-wait 1000
        }
        spf-wait {
            spf-max-wait 2000
            spf-initial-wait 50
            spf-second-wait 100
        }
    }
    area 0.0.0.0 {
    }
```

To delete the **lsa-generate** element and its parameters, the **delete** command is specified before the **lsa-generate** element. The **info** command shows that the **spf-wait** parameters are still configured.

```
*(ex)[configure router "Base" ospf 0]
A:admin@node-2# timers delete lsa-generate

*(ex)[configure router "Base" ospf 0]
A:admin@node-2# info
    timers {
        spf-wait {
            spf-max-wait 2000
            spf-initial-wait 50
            spf-second-wait 100
        }
    }
    area 0.0.0.0 {
    }
```

If the **delete** command is placed before the **timers** element, all elements within the **timers** element are also deleted.

```
*(ex)[configure router "Base" ospf 0]
A:admin@node-2# info
    timers {
        lsa-generate {
            max-lsa-wait 8000
            lsa-initial-wait 10
            lsa-second-wait 1000
        }
        spf-wait {
            spf-max-wait 2000
            spf-initial-wait 50
            spf-second-wait 100
        }
    }
    area 0.0.0.0 {
    }

*(ex)[configure router "Base" ospf 0]
A:admin@node-2# delete timers

(ex)[configure router "Base" ospf 0]
A:admin@node-2# info
    area 0.0.0.0 {
    }
```

4.5.3 Deleting List Entries and Lists

To remove a list entry, the **delete** operation is specified before the list name and the entry to be removed.

```
*(ex)[configure service]
A:admin@node-2# info | match pw-template
    pw-template "pw-1" {
    pw-template "pw-3" {
    pw-template "pw-5" {
    pw-template "pw-8" {

*(ex)[configure service]
A:admin@node-2# delete pw-template “pw-3”

*(ex)[configure service]
A:admin@node-2# info | match pw-template
    pw-template "pw-1" {
    pw-template "pw-5" {
    pw-template "pw-8" {

*(ex)[configure service]
A:admin@node-2#
```

An explicit wildcard (*) deletes all members of a list.

```
*(ex)[configure service]
A:admin@node-2# info | match pw-template
    pw-template "pw-1" {
    pw-template "pw-5" {
    pw-template "pw-8" {

*(ex)[configure service]
A:admin@node-2# delete pw-template *

*(ex)[configure service]
A:admin@node-2# info | match pw-template

*(ex)[configure service]
A:admin@node-2#
```

If the list is a multi-key list, a combination of specific members and wildcards (*) can be used. In the following example, **mep** is a multikey list, where the keys are **md-admin-name**, **ma-admin-name**, and **mep-id**.

```
*(ex)[configure service epipe "svc-name" sap 1/1/4:1 eth-cfm]
A:admin@node-2# info
    mep md-admin-name "ref1" ma-admin-name "ref2" mep-id 5 {
    }
    mep md-admin-name "ref1" ma-admin-name "ref3" mep-id 5 {
    }
    mep md-admin-name "ref6" ma-admin-name "ref3" mep-id 99 {
    }

*(ex)[configure service epipe "svc-name" sap 1/1/4:1 eth-cfm]
A:admin@node-2#
```

The following **delete** operation deletes all lists with **mep-id** of 5, regardless of the **md-admin-name** or **ma-admin-name**.

```
*(ex)[configure service epipe "svc-name" sap 1/1/4:1 eth-cfm]
```

```
A:admin@node-2# delete mep md-admin-name * ma-admin-name * mep-id 5

*(ex)[configure service epipe "svc-name" sap 1/1/4:1 eth-cfm]
A:admin@node-2# info
    mep md-admin-name "ref6" ma-admin-name "ref3" mep-id 99 {
    }

*(ex)[configure service epipe "svc-name" sap 1/1/4:1 eth-cfm]
A:admin@node-2#
```

The following **delete** operation removes all lists where **ma-admin-name** is “ref3” and **mep-id** is 5.

```
*(ex)[configure service epipe "svc-name" sap 1/1/4:1 eth-cfm]
A:admin@node-2# info
    mep md-admin-name "ref1" ma-admin-name "ref2" mep-id 5 {
    }
    mep md-admin-name "ref1" ma-admin-name "ref3" mep-id 5 {
    }
    mep md-admin-name "ref6" ma-admin-name "ref3" mep-id 99 {
    }

*(ex)[configure service epipe "svc-name" sap 1/1/4:1 eth-cfm]
A:admin@node-2# delete mep md-admin-name * ma-admin-name “ref3” mep-id 5

*(ex)[configure service epipe "svc-name" sap 1/1/4:1 eth-cfm]
A:admin@node-2# info
    mep md-admin-name "ref1" ma-admin-name "ref2" mep-id 5 {
    }
    mep md-admin-name "ref6" ma-admin-name "ref3" mep-id 99 {
    }

*(ex)[configure service epipe "svc-name" sap 1/1/4:1 eth-cfm]
A:admin@node-2#
```

The following **delete** operation removes all lists where **md-admin-name** is “ref1”.

```
*(ex)[configure service epipe "svc-name" sap 1/1/4:1 eth-cfm]
A:admin@node-2# info
    mep md-admin-name "ref1" ma-admin-name "ref2" mep-id 5 {
    }
    mep md-admin-name "ref1" ma-admin-name "ref3" mep-id 5 {
    }
    mep md-admin-name "ref6" ma-admin-name "ref3" mep-id 99 {
    }

*(ex)[configure service epipe "svc-name" sap 1/1/4:1 eth-cfm]
A:admin@node-2# delete mep md-admin-name “ref1” ma-admin-name * mep-id *

*(ex)[configure service epipe "svc-name" sap 1/1/4:1 eth-cfm]
A:admin@node-2# info
    mep md-admin-name "ref6" ma-admin-name "ref3" mep-id 99 {
    }

*(ex)[configure service epipe "svc-name" sap 1/1/4:1 eth-cfm]
A:admin@node-2#
```

4.5.3.1 Deleting Leaf-List Entries and Leaf-Lists

To remove a leaf-list entry, the **delete** operation is specified before the leaf-list name and the entry to be removed.

```
*(ex)[configure system security user-params local-user user "test" console]
A:admin@node-2# info
    member ["profile-a" "profile-b" "profile-x"]

*(ex)[configure system security user-params local-user user "test" console]
A:admin@node-2# delete member "profile-a"

*(ex)[configure system security user-params local-user user "test" console]
A:admin@node-2# info
    member ["profile-b" "profile-x"]

*(ex)[configure system security user-params local-user user "test" console]
A:admin@node-2#
```

Multiple leaf-list entries can be deleted in a single command with the use of brackets. The entries do not need to be in any specific order.

```
*(ex)[configure system security user-params local-user user "test" console]
A:admin@node-2# info
    member ["profile-a" "profile-b" "profile-f" "profile-x" "profile-c"]

*(ex)[configure system security user-params local-user user "test" console]
A:admin@node-2# delete member ["profile-c" "profile-f"]

*(ex)[configure system security user-params local-user user "test" console]
A:admin@node-2# info
    member ["profile-a" "profile-b" "profile-x"]

*(ex)[configure system security user-params local-user user "test" console]
A:admin@node-2#
```

An explicit wildcard (*) deletes all members of a leaf-list.

```
*(ex)[configure system security user-params local-user user "test" console]
A:admin@node-2# info
    member ["profile-b" "profile-x"]

*(ex)[configure system security user-params local-user user "test" console]
A:admin@node-2# delete member *

*(ex)[configure system security user-params local-user user "test" console]
A:admin@node-2# info

*(ex)[configure system security user-params local-user user "test" console]
A:admin@node-2#
```

The wildcard can optionally be enclosed in brackets.

```
*(ex)[configure system security user-params local-user user "test" console]
```

```
A:admin@node-2# delete member [*]
```

Deleting all members of a leaf-list sets the list to the unconfigured state (as indicated in the **info detail** display by the “##”).

```
*(ex)[configure system security user-params local-user user "test" console]
A:admin@node-2# delete member *

*(ex)[configure system security user-params local-user user "test" console]
A:admin@node-2# info detail | match member
 ## member

*(ex)[configure system security user-params local-user user "test" console]
A:admin@node-2#
```

4.6 Copying Configuration Elements

The output from the **info** commands can be copied and pasted and used as a direct input to another MD-CLI session, or loaded from a file.

The following example shows the output from the **info** command, displaying the following configuration for the profile of the user “guest1”.

```
*(ex)[configure system security aaa local-profiles profile "guest1"]
A:admin@node-2# info
    default-action permit-all
    entry 10 {
        action deny
        match "configure system security"
    }
    entry 20 {
        action deny
        match "configure li"
    }
    entry 30 {
        action deny
        match "show li"
    }
    entry 40 {
        action deny
        match "tools"
    }
```

The output can be copied and pasted to configure an identical profile for another user; for example, “guest2”. The working context must be at the same hierarchy level, as the **info** command output is context-sensitive.

Enter the context for configuring the profile for guest2:

```
(ex)[]
```

```
A:admin@node-2# configure system security aaa local-profiles profile guest2

*(ex)[configure system security aaa local-profiles profile "guest2"]
A:admin@node-2#
```

Copy the **info** command output and paste each line into the command line:

```
*(ex)[configure system security aaa local-profiles profile "guest2"]
A:admin@node-2#     default-action permit-all

*(ex)[configure system security aaa local-profiles profile "guest2"]
A:admin@node-2#     entry 10 {

*(ex)[configure system security aaa local-profiles profile "guest2" entry 10]
A:admin@node-2#         action deny

*(ex)[configure system security aaa local-profiles profile "guest2" entry 10]
A:admin@node-2#         match "configure system security"

*(ex)[configure system security aaa local-profiles profile "guest2" entry 10]
A:admin@node-2#     }

*(ex)[configure system security aaa local-profiles profile "guest2"]
A:admin@node-2#     entry 20 {

*(ex)[configure system security aaa local-profiles profile "guest2" entry 20]
A:admin@node-2#         action deny

*(ex)[configure system security aaa local-profiles profile "guest2" entry 20]
A:admin@node-2#         match "configure li"

*(ex)[configure system security aaa local-profiles profile "guest2" entry 20]
A:admin@node-2#     }

*(ex)[configure system security aaa local-profiles profile "guest2"]
A:admin@node-2#     entry 30 {

*(ex)[configure system security aaa local-profiles profile "guest2" entry 30]
A:admin@node-2#         action deny

*(ex)[configure system security aaa local-profiles profile "guest2" entry 30]
A:admin@node-2#         match "show li"

*(ex)[configure system security aaa local-profiles profile "guest2" entry 30]
A:admin@node-2#     }

*(ex)[configure system security aaa local-profiles profile "guest2"]
A:admin@node-2#     entry 40 {

*(ex)[configure system security aaa local-profiles profile "guest2" entry 40]
A:admin@node-2#         action deny

*(ex)[configure system security aaa local-profiles profile "guest2" entry 40]
A:admin@node-2#         match "tools"

*(ex)[configure system security aaa local-profiles profile "guest2" entry 40]
A:admin@node-2#     }
```

```
*(ex)[configure system security aaa local-profiles profile "guest2"]
A:admin@node-2#
```

The **info** command displays the configuration changes for profile “guest2”, which are identical to the configuration for profile “guest1”.

```
*(ex)[configure system security aaa local-profiles profile "guest2"]
A:admin@node-2# info
    default-action permit-all
    entry 10 {
        action deny
        match "configure system security"
    }
    entry 20 {
        action deny
        match "configure li"
    }
    entry 30 {
        action deny
        match "show li"
    }
    entry 40 {
        action deny
        match "tools"
    }

*(ex)[configure system security aaa local-profiles profile "guest2"]
A:admin@node-2#
```

Similarly, the **info flat** command output can be copied and pasted for the user profile for “guest3”; for example:

```
*(ex)[configure system security aaa local-profiles profile "guest2"]
A:admin@node-2# info flat
    default-action permit-all
    entry 10 action deny
    entry 10 match "configure system security"
    entry 20 action deny
    entry 20 match "configure li"
    entry 30 action deny
    entry 30 match "show li"
    entry 40 action deny
    entry 40 match "tools"
*(ex)[configure system security aaa local-profiles profile "guest2"]
A:admin@node-2#


*(ex)[configure system security aaa local-profiles profile "guest1"]
A:admin@node-2# /configure system security aaa local-profiles profile "guest3"

*(ex)[configure system security aaa local-profiles profile "guest3"]
A:admin@node-2#  default-action permit-all

*(ex)[configure system security aaa local-profiles profile "guest3"]
A:admin@node-2#     entry 10 action deny

*(ex)[configure system security aaa local-profiles profile "guest3"]
```

```
A:admin@node-2#     entry 10 match "configure system security"

*(ex)[configure system security aaa local-profiles profile "guest3"]
A:admin@node-2#     entry 20 action deny

*(ex)[configure system security aaa local-profiles profile "guest3"]
A:admin@node-2#     entry 20 match "configure li"

*(ex)[configure system security aaa local-profiles profile "guest3"]
A:admin@node-2#     entry 30 action deny

*(ex)[configure system security aaa local-profiles profile "guest3"]
A:admin@node-2#     entry 30 match "show li"

*(ex)[configure system security aaa local-profiles profile "guest3"]
A:admin@node-2#     entry 40 action deny

*(ex)[configure system security aaa local-profiles profile "guest3"]
A:admin@node-2#     entry 40 match "tools"

*(ex)[configure system security aaa local-profiles profile "guest3"]
A:admin@node-2# info
    default-action permit-all
    entry 10 {
        action deny
        match "configure system security"
    }
    entry 20 {
        action deny
        match "configure li"
    }
    entry 30 {
        action deny
        match "show li"
    }
    entry 40 {
        action deny
        match "tools"
    }

*(ex)[configure system security aaa local-profiles profile "guest3"]
A:admin@node-2#
```

The output from the **info full-context** command contains the full configuration path for the configuration statements. This output can be used to reconfigure the same user profile on another router, or to rebuild the user profile if it was deleted or discarded. The following example begins with a “guest1” user profile, which is subsequently deleted and re-added using the output from the **info full-context** command.

The following output shows the “guest1” user profile:

```
*(ex)[configure system security aaa local-profiles profile "guest1"]
A:admin@node-2# info full-context
    /configure system security aaa local-
profiles profile "guest1" defaultaction    permit-all
```

```
    /configure system security aaa local-
profiles profile "guest1" entry 10 action   deny
    /configure system security aaa local-
profiles profile "guest1" entry 10 match  "configure system security"
    /configure system security aaa local-
profiles profile "guest1" entry 20 action   deny
    /configure system security aaa local-
profiles profile "guest1" entry 20 match  "configure li"
    /configure system security aaa local-
profiles profile "guest1" entry 30 action   deny
    /configure system security aaa local-
profiles profile "guest1" entry 30 match  "show li"
    /configure system security aaa local-
profiles profile "guest1" entry 40 action   deny
    /configure system security aaa local-
profiles profile "guest1" entry 40 match   "tools"

*(ex)[configure system security aaa local-profiles profile "guest1"]
A:admin@node-2#
```

The “guest1” user profile is deleted, and the **info full-context** command after the delete shows no matches for profile “guest1”:

```
*(ex)[configure system security aaa local-profiles profile "guest1"]
A:admin@node-2# back

*(ex)[configure system security aaa local-profiles]
A:admin@node-2# delete profile “guest1”

*(ex)[configure system security aaa local-profiles]
A:admin@node-2# info full-context | match guest1

*(ex)[configure system security aaa local-profiles]
A:admin@node-2#
```

In the next step, the original full-context output for “guest1” is copied and pasted. Since the output contains the full configuration path, the statements can be pasted from any configuration context.

```
*(ex)[configure system security aaa local-profiles]
A:admin@node-2# top

*(ex)[configure]
A:admin@node-2#  /configure system security aaa local-
profiles profile "guest1" default-action permit-all

(ex)[configure]
A:admin@node-2#     /configure system security aaa local-
profiles profile "guest1"   entry 10 { }

*(ex)[configure system security aaa local-profiles profile "guest1"]
A:admin@node-2#     /configure system security aaa local-
profiles profile "guest1"   entry 10 action deny

*(ex)[configure system security aaa local-profiles profile "guest1"]
A:admin@node-2#     /configure system security aaa local-
```

```
profiles profile "guest1"   entry 10 match "configure system security"

*(ex)[configure system security aaa local-profiles profile "guest1"]
A:admin@node-2#     /configure system security aaa local-
profiles profile "guest1"   entry 20 { }

*(ex)[configure system security aaa local-profiles profile "guest1"]
A:admin@node-2#     /configure system security aaa local-
profiles profile "guest1"   entry 20 action deny

*(ex)[configure system security aaa local-profiles profile "guest1"]
A:admin@node-2#     /configure system security aaa local-
profiles profile "guest1"   entry 20 match "configure li"

*(ex)[configure system security aaa local-profiles profile "guest1"]
A:admin@node-2#     /configure system security aaa local-
profiles profile "guest1"   entry 30 { }

*(ex)[configure system security aaa local-profiles profile "guest1"]
A:admin@node-2#     /configure system security aaa local-
profiles profile "guest1"   entry 30 action deny

*(ex)[configure system security aaa local-profiles profile "guest1"]
A:admin@node-2#     /configure system security aaa local-
profiles profile "guest1"   entry 30 match "show li"

*(ex)[configure system security aaa local-profiles profile "guest1"]
A:admin@node-2#     /configure system security aaa local-
profiles profile "guest1"   entry 40 { }

*(ex)[configure system security aaa local-profiles profile "guest1"]
A:admin@node-2#     /configure system security aaa local-
profiles profile "guest1"   entry 40 action deny

*(ex)[configure system security aaa local-profiles profile "guest1"]
A:admin@node-2#     /configure system security aaa local-
profiles profile "guest1"   entry 40 match "tools"

*(ex)[configure system security aaa local-profiles profile "guest1"]
A:admin@node-2#

*(ex)[configure system security aaa local-profiles profile "guest1"]
A:admin@node-2# info
    default-action permit-all
    entry 10 {
        action deny
        match "configure system security"
    }
    entry 20 {
        action deny
        match "configure li"
    }
    entry 30 {
        action deny
        match "show li"
    }
    entry 40 {
        action deny
        match "tools"
```

```
    }

*(ex)[configure system security aaa local-profiles profile "guest1"]
A:admin@node-2#
```

The displayed output from the **compare** command can also be used to copy and paste statements in the MD-CLI. See section 4.7.1 for information about using the **compare** command.

4.7 Committing a Configuration

4.7.1 Viewing the Uncommitted Configuration Changes

The **compare** command in the MD-CLI compares configurations and displays the difference in one output. The command can only be executed from within the configuration context.

- **compare** [[**from**] *string*] [[**to**] *string*]
  - **flat**
  - **full-context**
  - **lines** *number*
  - **summary**

Table 21 provides a description of the **compare** command options.

Table 21 **Compare Command Options**

Option	Description
[**from**] *string*	Specify the reference datastore or configuration source to compare (default is **from baseline**). Options are: • **running** • **candidate** • **baseline** • **url** *string* • **rollback** *checkpoint-id* • **startup**

Table 21 **Compare Command Options (Continued)**

Option	Description
[**to**] *string*	Specify the datastore or configuration source to compare against (default is **to candidate**). Options are: • **running** • **candidate** • **baseline** • **url** *string* • **rollback** *checkpoint-id* • **startup**
flat	Show the context starting from the present working context
full-context	Show the context starting at the branch root
lines *number*	Show the specified number of lines before and after the changed element
summary	Suppress specific differences and display a summarized comparison

The following characters are used at the beginning of the output lines, to indicate the status of the element in the configuration:

- - (minus) — element is only in the first (from) configuration, displayed first
- + (plus) — element is only in the second (to) configuration, displayed second
- (space) — element is unchanged
- ~ (tilde) — new value of the element that changed (only used in the **summary** option)
- {...} — deleted elements compressed to its highest container (only used in the **summary** option)

```
*(ex)[configure]
A:admin@node-2# compare
    log {
+       accounting-policy 5 {
+           description "For SIO statistics"
+           collection-interval 69
+           include-system-info true
+           record service-ingress-octets
+       }
+       accounting-policy 8 {
+       }
    }
```

Note: The +/-/~ output from the **compare** command can be copied and pasted, or loaded from a file. Refer to section 4.7.1.1 for an example.

Because the **compare** command uses the default **from running**, the command **compare to candidate** is equivalent to **compare from running to candidate**. Executing **compare to running**, without specifying the **from** option is equivalent to **compare from running to running**, which shows no differences.

```
*(ex)[configure]
A:admin@node-2# compare to running

*(ex)[configure]
A:admin@node-2# compare to candidate
    log {
+       accounting-policy 5 {
+           description "For SIO statistics"
+           collection-interval 69
+           include-system-info true
+           record service-ingress-octets
+       }
+       accounting-policy 8 {
+       }
    }

*(ex)[configure]
A:admin@node-2# compare from running to candidate
    log {
+       accounting-policy 5 {
+           description "For SIO statistics"
+           collection-interval 69
+           include-system-info true
+           record service-ingress-octets
+       }
+       accounting-policy 8 {
+       }
    }
```

The following displays the output using the **flat** and **full-context** options.

```
*(ex)[configure]
A:admin@node-2# compare flat
+   log { accounting-policy 5 }
+   log accounting-policy 5 description "For SIO statistics"
+   log accounting-policy 5 collection-interval 69
+   log accounting-policy 5 include-system-info true
+   log accounting-policy 5 record service-ingress-octets
+   log { accounting-policy 8 }

*(ex)[configure]
A:admin@node-2# compare full-context
+   /configure log { accounting-policy 5 }
+   /configure log accounting-policy 5 description "For SIO statistics"
+   /configure log accounting-policy 5 collection-interval 69
```

```
+   /configure log accounting-policy 5 include-system-info true
+   /configure log accounting-policy 5 record service-ingress-octets
+   /configure log { accounting-policy 8 }

*(ex)[configure]
A:admin@node-2#
```

The following example shows the difference between the **compare** and **compare summary** commands. The **compare** command shows the deletion and addition of configuration changes, each on its own line, and the **compare summary** command shows the configuration change summarized on one line with a ~ character.

```
*(ex)[]
A:admin@node-2# compare
    router "Base" {
        interface "system" {
            ipv4 {
                primary {
-                   address 1.1.1.1
+                   address 10.243.5.96
                }
            }
        }
    }

*(ex)[]
A:admin@node-2# compare summary
    router "Base" {
        interface "system" {
            ipv4 {
                primary {
~                   address 10.243.5.96
                }
            }
        }
    }

*(ex)[]
A:admin@node-2#
```

4.7.1.1 Using the compare Outputs to Copy and Paste

In the following example, the **compare** command shows the timers that have been modified. After the **commit** command has been issued to add these to the running configuration, the **lsa-generate** container is deleted. The following displays the output for the **compare** command.

```
*(ex)[configure router "Base" ospf 0 timers]
A:admin@node-2# compare
+   lsa-generate {
+       max-lsa-wait 500000
+       lsa-initial-wait 100000
+       lsa-second-wait 200000
```

```
+   }
+   spf-wait {
+       spf-max-wait 120000
+       spf-initial-wait 50000
+       spf-second-wait 60000
+   }
```

The **compare** command, using the candidate configuration as the reference, displays the same configuration statements with a preceding minus (-) character. These statements will be used in a subsequent copy and paste function to delete some of the configuration. The minus (-) character at the beginning of the configuration statement takes the place of the **delete** keyword.

```
*(ex)[configure router "Base" ospf 0 timers]
A:admin@node-2# compare from candidate to running full-context
-   /configure router "Base" ospf 0 timers lsa-generate max-lsa-wait 500000
-   /configure router "Base" ospf 0 timers lsa-generate lsa-initial-wait 100000
-   /configure router "Base" ospf 0 timers lsa-generate lsa-second-wait 200000
-   /configure router "Base" ospf 0 timers spf-wait spf-max-wait 120000
-   /configure router "Base" ospf 0 timers spf-wait spf-initial-wait 50000
-   /configure router "Base" ospf 0 timers spf-wait spf-second-wait 60000

*(ex)[configure router "Base" ospf 0 timers]
A:admin@node-2# validate

*(ex)[configure router "Base" ospf 0 timers]
A:admin@node-2# commit
```

In the next step, the **lsa-generate** parameters are deleted, using a copy and paste of the first three configuration statements:

```
(ex)[configure]
A:admin@node-2# -   /configure router "Base" ospf 0 timers lsa-generate max-lsa-
wait 500000

*(ex)[configure]
A:admin@node-2# -   /configure router "Base" ospf 0 timers lsa-generate lsa-initial-
wait 100000

*(ex)[configure]
A:admin@node-2# -   /configure router "Base" ospf 0 timers lsa-generate lsa-second-
wait 200000
```

The **compare summary** command shows that the deleted **lsa-generate** parameters are compressed to its highest container, shown with an ellipsis in braces ({}).

```
*(ex)[configure]
A:admin@node-2# compare summary
    router "Base" {
        ospf 0 {
            timers {
-               lsa-generate { ... }
            }
        }
    }
```

If the **timers** container is deleted, which holds both the **lsa-generate** and **spf-wait** containers, the **compare summary** command now shows the **timers** container as the highest deleted container:

```
*(ex)[configure router "Base" ospf 0]
A:admin@node-2# delete timers
*(ex)[configure router "Base" ospf 0]
A:admin@node-2# compare
-   timers {
-       lsa-generate {
-           max-lsa-wait 500000
-           lsa-initial-wait 100000
-           lsa-second-wait 200000
-       }
-       spf-wait {
-           spf-max-wait 120000
-           spf-initial-wait 50000
-           spf-second-wait 60000
-       }
-   }

*(ex)[configure router "Base" ospf 0]
A:admin@node-2# compare summary
-   timers { ... }

*(ex)[configure router "Base" ospf 0]
A:admin@node-2#
```

4.7.2 Discarding Configuration Changes

The **discard** command in configuration mode cancels *all* changes made to the candidate configuration without impacting the running configuration or applications. The command is available only when the MD-CLI session is in a read/write configuration mode (private, exclusive, or global configuration mode) and only from the top of the **configure** branch (that is, **/configure**).

The following example shows the error that occurs when the **discard** operation is attempted from read-only configuration mode. The command is successful when the session is in global configuration mode, but only from the top of the configuration branch.

```
*(ro)[configure]
A:admin@node-2# compare
    log {
+       accounting-policy 5 {
+           description "For SIO statistics"
+           collection-interval 69
+           include-system-info true
+           record service-ingress-octets
+       }
+       accounting-policy 8 {
```

```
+       }
    }

*(ro)[configure]
A:admin@node-2# discard
MINOR: CLI #2069: Operation not allowed - currently in read-only mode

*(ro)[configure]
A:admin@node-2# exit

*(ro)[]
A:admin@node-2# quit-config
INFO: CLI #2067: Exiting read-only configuration mode

[]
A:admin@node-2# edit-config global
INFO: CLI #2054: Entering global configuration mode
INFO: CLI #2055: Uncommitted changes are present in the candidate configuration

*(gl)[]
A:admin@node-2# compare
    log {
+       accounting-policy 5 {
+           description "For SIO statistics"
+           collection-interval 69
+           include-system-info true
+           record service-ingress-octets
+       }
+       accounting-policy 8 {
+       }
    }

*(gl)[]
A:admin@node-2# configure log

*(gl)[configure log]
A:admin@node-2# discard
MINOR: MGMT_CORE #2203: Invalid element - 'discard' not allowed in 'log'

*(gl)[configure log]
A:admin@node-2# discard /configure

(gl)[configure log]
A:admin@node-2# /compare

(gl)[configure log]
A:admin@node-2#
```

Uncommitted changes from a global configuration session are kept in the candidate configuration when leaving configuration mode. Uncommitted changes from an exclusive or private configuration session are discarded when leaving configuration mode and a confirmation message is displayed:

```
*(ex)[]
A:admin@node-2# quit-config
INFO: CLI #2063: Uncommitted changes are present in the candidate configuration. Exi
ting exclusive configuration mode will discard those changes.
```

```
Discard uncommitted changes? [y,n] y
WARNING: CLI #2062: Exiting exclusive configuration mode -
 uncommitted changes are  discarded
INFO: CLI #2064: Exiting exclusive configuration mode
```

It is possible to discard the changes made by a session that obtained an explicit lock by disconnecting the remote session. Uncommitted changes from an exclusive configuration mode session are discarded when the session disconnects. See Viewing the Status of the Local Datastores for information about disconnecting a session.

4.7.3 Validating the Candidate Configuration

The **validate** command verifies the logic, constraints, and completeness of the candidate configuration without activating any changes. A successful validation returns no errors. If the validation fails, detailed failure reasons are provided. The **validate** command can be executed from any working directory and in any configuration mode.

```
*(ro)[]
A:admin@node-2# compare
    log {
+       accounting-policy 7 {
+           description "seven"
+           collection-interval 77
+       }
    }

*(ro)[]
A:admin@node-2# validate

*(ro)[]


*(ex)[]
A:admin@node-2# compare
+   eth-cfm {
+       domain "mdn" {
+           association "man" {
+               ccm-interval 10ms
+           }
+       }
+   }

*(ex)[]
A:admin@node-2# validate
MINOR: MGMT_CORE #236: configure eth-cfm domain "mdn" level -
 Missing mandatory fields
MINOR: ETH_CFM #12: configure eth-cfm domain "mdn" format -
 Inconsistent Value  error - One of dns, mac, name or format must be provided
```

```
*(ex)[]
A:admin@node-2#
```

The **commit** command also runs validation on the configuration. Therefore, it is not necessary to execute the **validate** command as a separate step when committing the candidate configuration.

4.7.4 Updating the Candidate Configuration

As described in Multiple Simultaneous Candidate Configurations, a candidate configuration uses two datastores:

- a baseline datastore that contains a snapshot copy of the running configuration at a specific time
- a candidate datastore that contains changes relative to its associated baseline datastore

For a private candidate configuration, access by MD-CLI sessions in private configuration mode, a snapshot of the running configuration is copied in the private baseline datastore:

- when a private candidate configuration is instantiated, when a user enters the private configuration mode
- when a manual update is performed
- after a commit, when no merge conflicts are detected during the automatic update and the updated candidate configuration is valid

For the global candidate configuration, accessed by MD-CLI sessions in global and exclusive configuration mode, a tracking mechanism exists.

- The baseline datastore tracks the running datastore, that is, changes in the running datastore are automatically copied in the baseline datastore:
  - after a router reboot
  - after a successful commit
  - after a discard with an up-to-date global baseline
- Tracking stops and a snapshot of the running datastore is copied in the global baseline datastore when the global candidate is touched (for example, a configuration element is added, deleted, or modified). A new snapshot of the running datastore is copied in the global baseline datastore when a manual update is performed.

With two simultaneous active configuration sessions that access different candidate configurations, a commit from one configuration session changes the running configuration and causes the candidate configuration of the other session to be out of date and must be updated.

To update a candidate configuration, the following tasks are performed.

- a new snapshot of the running configuration is copied in the baseline datastore
- the candidate configuration changes are merged in the new baseline:
  - The changes in the candidate datastore are applied to the new baseline datastore.
  - Merge conflicts are detected and resolved. A merge conflict occurs when a configuration element is added, deleted, or modified in the candidate configuration and the same configuration element was also added, deleted, or modified in the running configuration after the baseline snapshot was taken.
  - The resulting changes are stored in the candidate datastore as new changes relative to the updated baseline.

An update can be performed manually with the **update** command. The update must be executed at the configuration root (**/configure**). Merge conflicts are reported and resolved according to the conflict resolution rules. The **update** command does not provide output when no conflicts are detected.

The following is an example of a merge conflict reported in an update:

```
+   /configure router "Base" interface "int-1" ipv4 primary address 10.2.3.4
##  address - exists with different value: address 10.1.2.3 -
 change updated:   replace existing value
```

The first line lists the candidate configuration change that caused the merge conflict, in this case, adding an interface IPv4 address.

The second line describes the merge conflict and starts with a double hash (##) followed by the description:

- A merge conflict is detected for the configuration element **address**.
- The address already exists in the running configuration, but has a different value.
- The candidate configuration change as shown on the first line is updated; instead of adding an interface address, the interface address is replaced.

An update is automatically started when the candidate configuration is committed. The commit is canceled when merge conflicts are detected to give the administrator the opportunity to resolve the conflicts before committing again. The update, in this case, is not executed, the candidate configuration is unchanged, and the baseline datastore is not updated.

The **update check** command performs a dry-run update of the candidate configuration. Merge conflicts are reported the same way as for the **update** command, but the update is not executed. The **update check** command must be executed at the configuration root (**/configure**) or it can be executed in any configure branch descendant as **update check /configure**.

4.7.4.1 Example Update Scenario With Merge Conflicts

The private candidate configuration of user-1 is out-of-date. The running configuration has interface **backbone-1** configured. The private baseline datastore does not have the interface configured. The interface **backbone-1** configured by user-1 has a different address in its candidate configuration.

```
!*[pr:configure router "Base"]
A:user-1@node-2# info running
    interface "backbone-1" {
        ipv4 {
            primary {
                address 10.2.2.2
                prefix-length 24
            }
        }
    }

!*[pr:configure router "Base"]
A:user-1@node-2# info baseline

!*[pr:configure router "Base"]
A:user-1@node-2# info
    interface "backbone-1" {
        ipv4 {
            primary {
                address 10.1.1.1
                prefix-length 24
            }
        }
    }
```

The following is the list of changes entered in the private candidate configuration of user-1:

```
!*[pr:configure router "Base"]
A:user-1@node-2# compare baseline candidate full-context summary
+   /configure router "Base" interface "backbone-1" { }
```

```
+   /configure router "Base" interface "backbone-1" { ipv4 primary }
+   /configure router "Base" interface "backbone-1" ipv4 primary address 10.1.1.1
+   /configure router "Base" interface "backbone-1" ipv4 primary prefix-length 24
```

A **commit** command starts an automatic update. Because merge conflicts are detected, the **commit** is canceled:

```
!*[pr:configure router "Base"]
A:user-1@node-2# commit
MINOR: MGMT_CORE #2703: Commit canceled - conflicts detected, use update
```

A dry-run update detects the merge conflicts without executing the update. Each configuration element that is changed in both the candidate configuration and the running configuration after the last baseline snapshot was taken results in a conflict and is reported.

```
!*[pr:configure]
A:user-1@node-2# update check
+   /configure router "Base" { interface "backbone-1" }
##  interface "backbone-1" { } - already exists - change removed

+   /configure router "Base" { interface "backbone-1" ipv4 primary }
##  primary { } - already exists - change removed

+   /configure router "Base" interface "backbone-1" ipv4 primary address 10.1.1.1
##  address - exists with different value: address 10.2.2.2 -
 change updated:   replace existing value

+   /configure router "Base" interface "backbone-1" ipv4 primary prefix-length 24
##  prefix-length - exists with same value - change removed
```

After verifying that the merge conflict resolution is acceptable, the update can be executed. The reporting is the same as for a dry-run update.

```
!*[pr:configure]
A:user-1@node-2# update
+   /configure router "Base" { interface "backbone-1" }
##  interface "backbone-1" { } - already exists - change removed

+   /configure router "Base" { interface "backbone-1" ipv4 primary }
##  primary { } - already exists - change removed

+   /configure router "Base" interface "backbone-1" ipv4 primary address 10.1.1.1
##  address - exists with different value: address 10.2.2.2 -
 change updated:  replace existing value

+   /configure router "Base" interface "backbone-1" ipv4 primary prefix-length 24
##  prefix-length - exists with same value - change removed
```

The candidate configuration is now updated: the baseline datastore equals the running datastore and the candidate datastore contains the updated list of changes as described in the update report.

```
*[pr:configure router "Base"]
A:user-1@node-2# compare baseline candidate
    interface "backbone-1" {
        ipv4 {
            primary {
-               address 10.2.2.2
+               address 10.1.1.1
            }
        }
    }

*[pr:configure router "Base"]
A:user-1@node-2# info
    interface "backbone-1" {
        ipv4 {
            primary {
                address 10.1.1.1
                prefix-length 24
            }
        }
    }

*[pr:configure router "Base"]
A:user-1@node-2# info baseline
    interface "backbone-1" {
        ipv4 {
            primary {
                address 10.2.2.2
                prefix-length 24
            }
        }
    }

*[pr:configure router "Base"]
A:user-1@node-2# info running
    interface "backbone-1" {
        ipv4 {
            primary {
                address 10.2.2.2
                prefix-length 24
            }
        }
    }
```

4.7.4.2 Example Update Scenario Without Merge Conflicts

The private candidate configuration of user-1 is out-of-date. The running configuration has interface **backbone-1** configured. The private baseline datastore does not have the interface configured. The interface **backbone-2** is configured by user-1.

```
!*[pr:configure router "Base"]
A:user-1@node-2# info running
    interface "backbone-1" {
```

```
        ipv4 {
            primary {
                address 10.1.1.1
                prefix-length 24
            }
        }
    }

!*[pr:configure router "Base"]
A:user-1@node-2# info baseline

!*[pr:configure router "Base"]
A:user-1@node-2# info
    interface "backbone-2" {
        ipv4 {
            primary {
                address 10.2.2.2
                prefix-length 24
            }
        }
    }
```

The following shows the list of changes entered in the private candidate configuration of user-1:

```
!*[pr:configure]
A:user-1@node-2# compare baseline candidate full-context summary
+   /configure router "Base" { }
+   /configure router "Base" { interface "backbone-2" }
+   /configure router "Base" { interface "backbone-2" ipv4 primary }
+   /configure router "Base" interface "backbone-2" ipv4 primary address 10.2.2.2
+   /configure router "Base" interface "backbone-2" ipv4 primary prefix-length 24
```

A dry-run update detects merge conflicts without executing the update. There are no conflicts detected in this case.

```
!*[pr:configure]
A:user-1@node-2# update check

!*[pr:configure]
A:user-1@node-2#
```

A **commit** operation starts an automatic update. Without merge conflicts, the commit succeeds.

```
!*[pr:configure]
A:user-1@node-2# commit

[pr:configure]
A:user-1@node-2#
```

After a **commit** operation, the candidate configuration is updated; the baseline datastore equals the running datastore and the candidate datastore is empty.

```
[pr:configure]
A:user-1@node-2# compare baseline candidate

[pr:configure]
A:user-1@node-2# compare baseline running

[pr:configure router "Base"]
A:user-1@node-2# info
    interface "backbone-1" {
        ipv4 {
            primary {
                address 10.1.1.1
                prefix-length 24
            }
        }
    }
    interface "backbone-2" {
        ipv4 {
            primary {
                address 10.2.2.2
                prefix-length 24
            }
        }
    }
```

4.7.5 Committing the Candidate Configuration

The **commit** command can be executed from any hierarchy level within any configuration branch.

- **commit**
  - **confirmed**
    - [**timeout**] *minutes*
    - **accept**
    - **cancel**
    - **persist-id** *string*

Note: The **confirmed** option of the **commit** command is only available for the **configure** configuration region.

When a **commit** operation is initiated while the baseline is out-of-date, the router first attempts to update the candidate configuration. When a merge conflict is detected, the commit operation is canceled to allow the administrator to resolve the merge conflicts manually.

```
!*[pr:configure]
A:admin@node-2# commit
MINOR: MGMT_CORE #2703: Commit canceled - conflicts detected, use update
```

```
!*[pr:configure]
A:admin@node-2#
```

The update is executed and the **commit** operation proceeds when no merge conflict is detected. See Updating the Candidate Configuration for the update process.

A validation is subsequently performed on the candidate configuration.

With a successful validation, the changes are copied to the running configuration, which becomes the current, operational router configuration. The candidate configuration is reset to its initial state; an empty candidate datastore and an up-to-date baseline.

If the **commit** operation fails, an automatic rollback occurs, which returns the running state to the state before the **commit** was applied. An automatic rollback does not use a rollback checkpoint file, so is not dependent on persistency to be enabled. Instead, a list of changes is kept in memory until the automatic rollback is completed. The uncommitted changes remain in the candidate configuration.

4.7.5.1 Using the commit confirmed Command

Executing the **commit** command with no options performs the operation immediately. the confirmed option can be used to activate configuration changes without making them persistent, to give the user time to verify that the configuration is working as intended. By default, the **commit confirmed** command executes the **commit** operation with an automatic rollback of 10 minutes. Within this time, an explicit confirmation (**commit confirmed accept**) must be issued for the changes to become persistent. Other configuration commands issued during this time interval are blocked.

While the **commit confirmed** timer is running, the remaining time before an automatic rollback is shown before each prompt of all active MD-CLI sessions.

```
*(gl)[configure log accounting-policy 5]
A:admin@node-2# commit confirmed

INFO: CLI #2090: Commit confirmed - automatic rollback in 9 minutes 59 seconds
(gl)[configure log accounting-policy 5]
A:admin@node-2#

INFO: CLI #2090: Commit confirmed - automatic rollback in 9 minutes 47 seconds
(gl)[configure log accounting-policy 5]
A:admin@node-2# pwc
Present Working Context:
  configure
  log
  accounting-policy 5
```

```
INFO: CLI #2090: Commit confirmed - automatic rollback in 9 minutes 45 seconds
(gl)[configure log accounting-policy 5]
A:admin@node-2# back

INFO: CLI #2090: Commit confirmed - automatic rollback in 9 minutes 41 seconds
(gl)[configure log]
A:admin@node-2# accounting-policy 9
MINOR: MGMT_CORE #2604: Commit confirmed in progress -
 changes to the candidate   configuration are not allowed

INFO: CLI #2090: Commit confirmed - automatic rollback in 9 minutes 38 seconds
(gl)[configure log]
A:admin@node-2#

INFO: CLI #2090: Commit confirmed - automatic rollback in 8 minutes 44 seconds
(gl)[configure log]
A:admin@node-2#
```

If the initial **commit** fails, the **commit confirmed** operation is canceled and no timer is started.

```
*(ex)[configure log accounting-policy 5]
A:admin@node-2# collection-interval 3

*(ex)[configure log accounting-policy 5]
A:admin@node-2# commit confirmed
MINOR: LOG #12: configure log accounting-policy 5 collection-interval -
 Inconsistent Value error - Minimum value is 5 minutes for this record type.

*(ex)[configure log accounting-policy 5]
A:admin@node-2#
```

The **timeout** option for the **commit confirmed** operation can override the default value of 10 minutes. While a **commit confirmed** timer is running, a subsequent **commit confirmed** or **commit confirmed** operation with a timeout option restarts the timer.

```
*(ex)[configure log accounting-policy 5]
A:admin@node-2# commit confirmed

INFO: CLI #2090: Commit confirmed - automatic rollback in 10 minutes
(ex)[configure log accounting-policy 5]
A:admin@node-2# commit confirmed 33

INFO: CLI #2090: Commit confirmed - automatic rollback in 33 minutes
(ex)[configure log accounting-policy 5]
A:admin@node-2#
```

Once the **commit confirmed** operation is underway, the timer starts. A **commit confirmed cancel** command terminates an ongoing confirmed commit and immediately performs an automatic rollback to the previous state before the initial **commit confirmed** command was issued.

If the **commit confirmed accept** command is not issued within the specified timeout period after a successful commit, all changes are automatically discarded from the running configuration. If the configuration session from which the commit confirmed was initiated is still active, the candidate configuration maintains all uncommitted configuration changes.

4.7.5.1.1 Non-persistent Operation

The **commit confirmed** and **commit confirmed accept** or **commit confirmed cancel** commands must be executed from the same MD-CLI configuration session. Commit commands executed from another configuration session while the **commit confirmed** timer is running generate an error.

Leaving the configuration mode or logging out from the MD-CLI session cancels the ongoing **commit confirmed** and starts an automatic rollback. The user must acknowledge the request to exit configuration mode or logout.

```
*(ex)[]
A:admin@node-2# commit confirmed
INFO: CLI #2090: Commit confirmed - automatic rollback in 10 minutes
(ex)[]
A:admin@test-node# exit all

INFO: CLI #2090: Commit confirmed - automatic rollback in 9 minutes 55 seconds
(ex)[]
A:admin@test-node# quit-config
INFO: CLI #2095: Commit confirmed in progress -
 exiting configuration mode will    cancel the commit confirmed and start configurat
ion rollback

Cancel commit confirmed and rollback immediately? [y,n] n
INFO: CLI #2076: Exit global configuration mode canceled

INFO: CLI #2090: Commit confirmed - automatic rollback in 9 minutes 48 seconds
(ex)[]
A:admin@test-node# logout
INFO: CLI #2095: Commit confirmed in progress -
 logout will cancel the commit   confirmed and start configuration rollback

Cancel commit confirmed and rollback immediately? [y,n] y
WARNING: CLI #2077: Exiting global configuration mode - commit confirmed canceled
INFO: CLI #2057: Uncommitted changes are kept in the candidate configuration
```

4.7.5.1.2 Persistent Identifier

Note: In private configuration mode, **commit confirmed** with a persistent identifier cannot be used. Instead, use the non-persistent **commit confirmed** command.

A persistence identifier can be specified with the initial **commit confirmed** command. A **commit confirmed accept** or **cancel** command can then be executed from the same or a different MD-CLI configuration session, NETCONF, or gRPC session, from where the **commit confirmed persist-id** command was initiated. The persistence identifier must then be included with the subsequent **commit confirmed** commands. The persistence identifier is a user-defined string of up to 255 characters or an empty string (“”).

```
*(ex)[configure]
A:admin@node-2# commit confirmed persist-id my-commit

INFO: CLI #2090: Commit confirmed - automatic rollback in 10 minutes
(ex)[configure]
A:admin@node-2# commit confirmed cancel
MINOR: MGMT_CORE #2603: Commit confirmed - persist-id expected

INFO: CLI #2090: Commit confirmed - automatic rollback in 9 minutes 53 seconds
(ex)[configure]
A:admin@node-2# commit confirmed accept
MINOR: MGMT_CORE #2603: Commit confirmed - persist-id expected

INFO: CLI #2090: Commit confirmed - automatic rollback in 9 minutes 45 seconds
(ex)[configure]
A:admin@node-2# commit confirmed cancel persist-id my-commit

*(ex)[configure]
A:admin@node-2#
```

4.8 Saving Changes

The running configuration can be saved to a local or remote file location with the **admin save** [**url**] *location* command, where *location* is a character string specifying the local or remote location where the configuration is to be saved.

To make the running configuration persistent, the configuration should be saved to the startup configuration location specified in the Boot Options File (BOF) as primary-config. This is achieved with the **admin save** command without specifying a location that defaults to the BOF primary-config.

The MD-CLI has an implicit persistency option linked to the **commit** command: the **auto-config-save** command in **configure system management-interface cli md-cli**. When candidate configuration changes are successfully committed, the configuration is automatically saved if **auto-config-save** is set to **true**.

```
*(ex)[configure system management-interface]
A:admin@node-2# info detail

    <snip>
        md-cli {
```

```
        auto-config-save true
        environment {
            more true
            time-display local
            command-completion {
                enter true
                space true
                tab true
            }
```

When **auto-config-save** is set to **false**, the **admin save** command must be issued to make the configuration persistent.

4.9 Rolling Back a Configuration from a Checkpoint File

A rollback checkpoint is an MD-CLI configuration file that can be loaded in the candidate configuration with the **rollback** command.

A rollback checkpoint is created automatically after every successful commit when automatic save is enabled via the MD-CLI **auto-config-save** command.

```
*(ex)[configure system management-interface]
A:admin@node-2# info detail

    <snip>

        md-cli {
            auto-config-save true
            environment {
                more true
                time-display local
                command-completion {
                    enter true
                    space true
                    tab true
                }
```

A rollback checkpoint is also created if an operator issues the **admin save** command, regardless of the MD-CLI **auto-config save** setting.

The **rollback** command loads a previously saved MD-CLI configuration file in the candidate configuration. Loading the file does not automatically initiate a **commit** command, which means that the file can be examined before committing. This **rollback** command is the equivalent of a **load full-replace** of the configuration file, but is identified with a checkpoint identifier. If no identifier is specified, the latest saved configuration file identified with index identifier 0 is used as the default.

The **rollback** command is available only for the **model-driven** management interface configuration mode.

```
[ex:configure]
A:admin@node-2# rollback

 rollback

 [checkpoint] (<number> | <keyword>)
 <number>   - <0..200>
 <keyword>  - startup

    ID corresponding to the configuration file to roll back

 [checkpoint]          - ID corresponding to the configuration file to roll back
```

Configuration files loaded with the **rollback** *checkpoint-id* command are identified with a number that corresponds to the configuration file and location specified as primary-config in the active Boot Option File (BOF). For example, the configuration file executed for a **rollback 3** command corresponds to the file named config.cfg.3. The checkpoint identifier 0 corresponds to the last saved configuration file and does not have a suffix. This is also the default when no checkpoint identifier is specified with the **rollback** command. By default, five configuration files are saved. The **configuration-backups** command can be used to save a different number of configuration files.

The **startup** option of the **rollback** command loads the contents of the current **admin save** file set with the primary configuration and not the version of the startup file that was booted.

```
[ex:configure system management-interface configuration-save]
A:admin@node-2# info detail
 ## apply-groups
 ## apply-groups-exclude
    configuration-backups 5

[ex:configure system management-interface configuration-save]
A:admin@node-2# configuration-backups

 configuration-backups <number>
 <number>  - <1..200>
 Default   - 5

    Maximum number of backup versions maintained
```

The **//show bof** command executed from the MD-CLI shows the name of the file as config.cfg.

```
(ro)[]
A:admin@node-2# //show bof
INFO: CLI #2051: Switching to the classic CLI engine
A:node-2# /show bof
===============================================================================
```

```
BOF (Memory)
===============================================================================
    primary-image     <snip>
    primary-config    <snip>/config.cfg
    license-file      <snip>/license
```

In the MD-CLI, the **rollback** command references the same filename with an appended suffix of the checkpoint identifier, in this case, identifier 3:

```
(ex)[configure]
A:admin@node-2# rollback 3
Executed 386 lines in 0.4 seconds from file <snip>/config.cfg.3
```

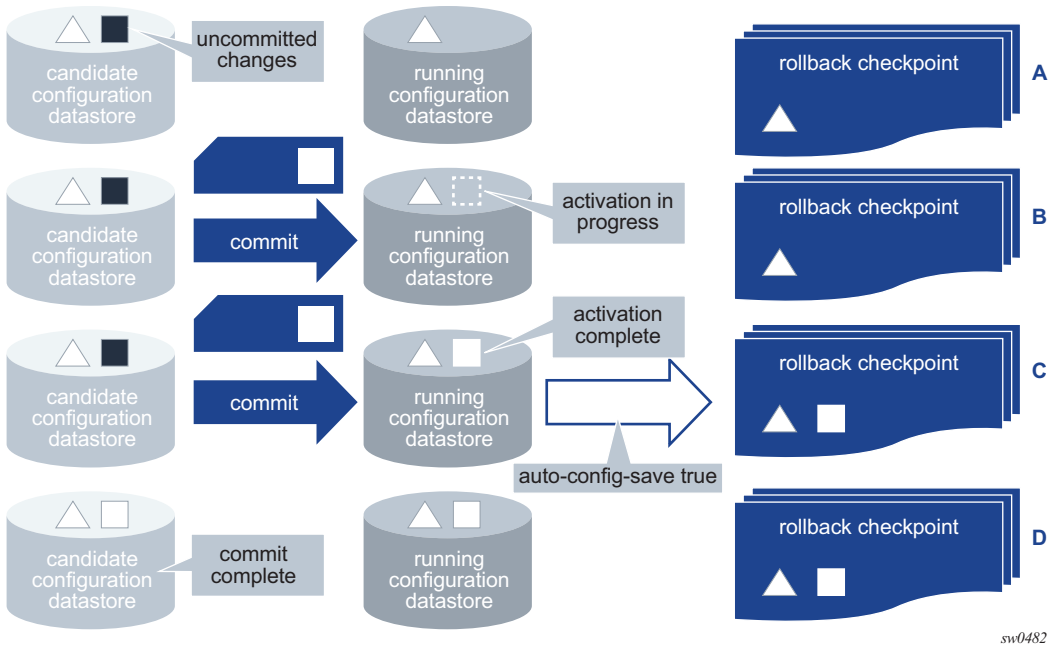
The **rollback** command is available in global or exclusive configuration mode and can only be executed from the root of the configuration branch.

When the **auto-config-save** parameter is set to **true**, the **rollback** command (without an index) is the equivalent of executing the **discard** command for the current candidate configuration changes.

The following figures show the relationship between the candidate and running configurations, the **commit** command, the setting of the **auto-config-save** parameter, and the rollback checkpoint files.

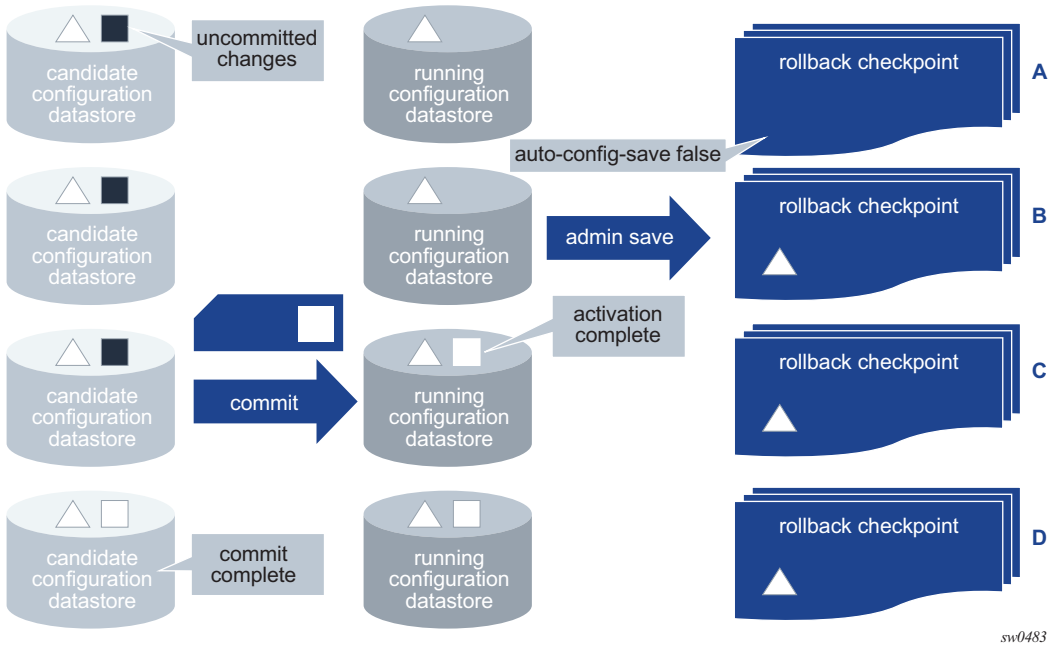
In Figure 8, the **auto-config-save** parameter is set to **true**. With a successful commit, a rollback checkpoint is created.

Figure 8 **Successful Commit with auto-config-save true**



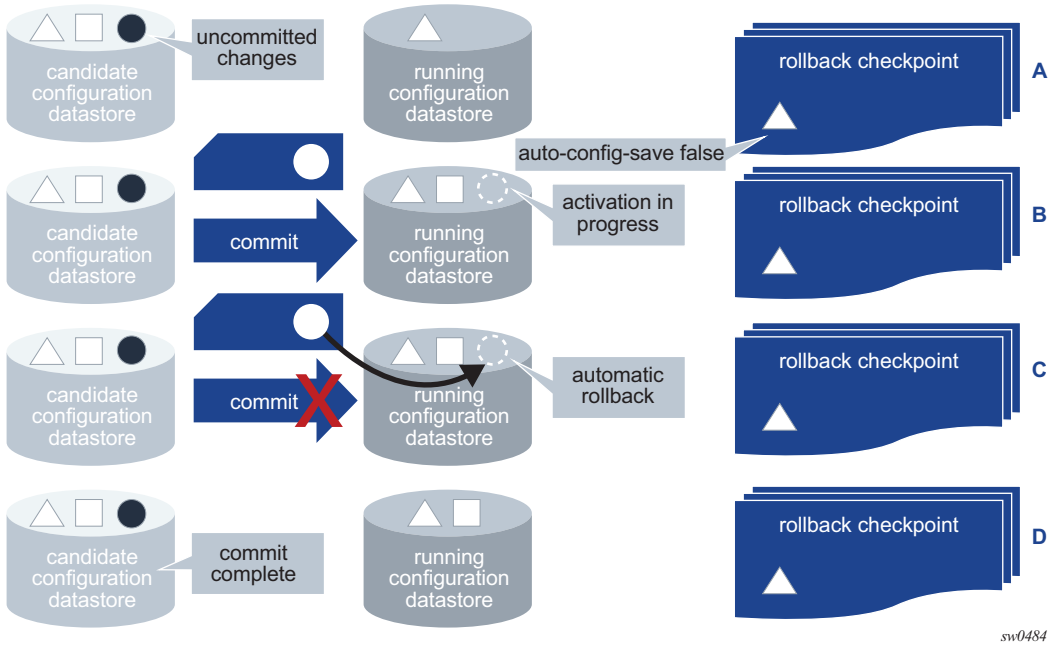
In Figure 9, the **auto-config-save** parameter is set to **false**. The **admin save** command creates a rollback checkpoint of the running configuration before the commit. However, a rollback checkpoint is not created after the successful commit.

Figure 9 **Successful commit with auto-config-save false**



In Figure 10, the commit fails and no rollback checkpoint is created, regardless of the setting of the **auto-config-save** parameter.

Figure 10 **Failed commit**



4.10 Loading a Configuration File

The **load** command loads the contents of a local or remote file into the candidate configuration. The command can only be executed at the top of the **configure** region when the MD-CLI session is in private, exclusive, or global configuration mode and does not result in a context change. The command can be issued regardless of whether uncommitted changes are present in the candidate configuration datastore.

The syntax of the **load** command is as follows:

load [**mode**] (**full-replace** | **merge**) [**url**] *filename*

The **full-replace** option replaces the current candidate configuration with the specified file.

The **merge** option merges the contents of the specified file into the candidate configuration. If there are conflicts, the configuration statements in the specified file override the existing configuration statements.

The file to be loaded is not a CLI script to be executed and cannot include:

- MD-CLI commands such as **commit**, **delete**, or **tools**

- navigation commands such as **exit**, **back**, or **top**

See Executing Commands with a File to perform such actions from a file.

If the loaded file encounters errors, parsing terminates at the first error. Statements before the error are loaded into the candidate configuration. Configuration statements in the loaded file are also subject to AAA command authorization. An authorization check failure also terminates the execution of further statements in the file.

Note: If the router fails to boot due to an invalid configuration syntax, it is recommended to correct the syntax and reboot the router, which also reloads persistent indices. This procedure is preferred over using **load full-replace** to restore the configuration without a reboot.

4.10.1 Using info Outputs in Load Files

The output from the **info full-context** or **info** commands can be copied and pasted into a load file. Both the **full-replace** and **merge** options support this type of content.

The following shows the output from the **info full-context** command. This output can be copied and pasted into a file; for example, cf3:\testbgp.cfg.

```
*[ro:configure router "Base" bgp]
A:admin@node-2# info full-context
    /configure router "Base" bgp group "external" { }
    /configure router "Base" bgp neighbor "192.168.89.8" group "external"
    /configure router "Base" bgp neighbor "192.168.89.8" prefix-
limit ipv4  maximum  200
    /configure router "Base" bgp neighbor "192.168.89.8" prefix-limit ipv4 log-
only  true
    /configure router "Base" bgp neighbor "192.168.89.8" prefix-
limit ipv4   threshold 80

A:admin@node-2# info full-context > cf3:testbgp.cfg


*(ex)[configure router "Base" bgp]
A:admin@node-2# //file dir
INFO: CLI #2051: Switching to the classic CLI engine
*A:node-2# /file dir

Volume in drive cf3 on slot A is .

Volume in drive cf3 on slot A is formatted as FAT32

Directory of cf3:\

10/24/2019  04:02a      <DIR>          .ssh/
01/01/1980  12:00a                170 NVRAM.DAT
```

```
01/01/1980  12:00a                 679 bof.cfg
10/24/2019  04:02a                 314 nvsys.info
10/24/2019  04:02a                   1 restcntr.txt
10/27/2019  02:57p                 257 testbgp.cfg
               5 File(s)                   1421 bytes.
               1 Dir(s)                       0 bytes free.

INFO: CLI #2052: Switching to the MD-CLI engine
```

From the MD-CLI, the **//file type** command displays the contents of the file:

```
*[ro:configure router "Base" bgp]
A:admin@node-2# //file type cf3:testbgp.cfg
INFO: CLI #2051: Switching to the classic CLI engine
*A:node-2>config>profile# /file type cf3:testbgp.cfg
File: testbgp.cfg
-------------------------------------------------------------------------------
    /configure router "Base" bgp group "external" { }
    /configure router "Base" bgp neighbor "192.168.89.8" group "external"
    /configure router "Base" bgp neighbor "192.168.89.8" prefix-
limit ipv4 maximum 200
    /configure router "Base" bgp neighbor "192.168.89.8" prefix-limit ipv4 log-
only true
    /configure router "Base" bgp neighbor "192.168.89.8" prefix-
limit ipv4 threshold 80

===============================================================================
INFO: CLI #2052: Switching to the MD-CLI engine
```

The **load merge** command can be used to merge the contents of the file into the candidate configuration. The following example shows no current candidate configuration changes for BGP before the command is executed. The **compare** command shows the candidate configuration changes after the file is merged.

```
[ex:configure]
A:admin@node-2# load merge cf3:testbgp.cfg
Loaded 5 lines in 0.0 seconds from file cf3:\testbgp.cfg

*[ex:configure]
A:admin@node-2# compare
    router "Base" {
        bgp {
+           group "external" {
+           }
+           neighbor "192.168.89.8" {
+               group "external"
+               prefix-limit ipv4 {
+                   maximum 200
+                   log-only true
+                   threshold 80
+               }
+           }
        }
    }
```

The output from the **info flat** command can also be copied into a file.

```
*[ro:configure router "Base" bgp]
A:admin@node-2# info flat
    group "external" { }
    neighbor "192.168.89.8" group "external"
    neighbor "192.168.89.8" prefix-limit ipv4 maximum 200
    neighbor "192.168.89.8" prefix-limit ipv4 log-only true
    neighbor "192.168.89.8" prefix-limit ipv4 threshold 80

*[ro:configure router "Base" bgp]
A:admin@node-2# info flat > cf3:testbgp.cfg

*[ro:configure router "Base" bgp]
A:admin@node-2# //file type cf3:testbgp.cfg
INFO: CLI #2051: Switching to the classic CLI engine
*A:node-2>config>profile# /file type cf3:testbgp.cfg
File: testbgp.cfg
-------------------------------------------------------------------------------
    group "external" { }
    neighbor "192.168.89.8" group "external"
    neighbor "192.168.89.8" prefix-limit ipv4 maximum 200
    neighbor "192.168.89.8" prefix-limit ipv4 log-only true
    neighbor "192.168.89.8" prefix-limit ipv4 threshold 80

===============================================================================
```

An additional context line is added to specify the context /configure router “Base” bgp, as shown in the file display:

```
*[ro:configure router "Base" bgp]
A:admin@node-2# //file type cf3:testbgp.cfg
INFO: CLI #2051: Switching to the classic CLI engine
*A:node-2>config>profile# /file type cf3:testbgp.cfg
File: testbgp.cfg
-------------------------------------------------------------------------------
/configure router bgp
    group "external" { }
    neighbor "192.168.89.8" group "external"
    neighbor "192.168.89.8" prefix-limit ipv4 maximum 200
    neighbor "192.168.89.8" prefix-limit ipv4 log-only true
    neighbor "192.168.89.8" prefix-limit ipv4 threshold 80

===============================================================================
```

The file is merged and the **compare** command shows the resulting candidate configuration changes.

```
[ex:configure router "Base" bgp]
A:admin@node-2# info

[ex:configure router "Base" bgp]
A:admin@node-2# top

[ex:configure]
A:admin@node-2# load merge cf3:testbgp.cfg
```

```
Loaded 6 lines in 0.0 seconds from file cf3:\testbgp.cfg

*[ex:configure]
A:admin@node-2# compare
    router "Base" {
        bgp {
+           group "external" {
+           }
+           neighbor "192.168.89.8" {
+               group "external"
+               prefix-limit ipv4 {
+                   maximum 200
+                   log-only true
+                   threshold 80
+               }
+           }
        }
    }
```

The following shows the output from the **info** command. To use the output in a load file, the context must be added through a manual edit, similar to the edit of file testbgp.cfg in the preceding example, or use the output from the **info full-context** command.

```
*[ro:configure router "Base" bgp]
A:admin@node-2# info
    group "external" {
    }
    neighbor "192.168.89.8" {
        group "external"
        prefix-limit ipv4 {
            maximum 200
            log-only true
            threshold 80
        }
    }
```

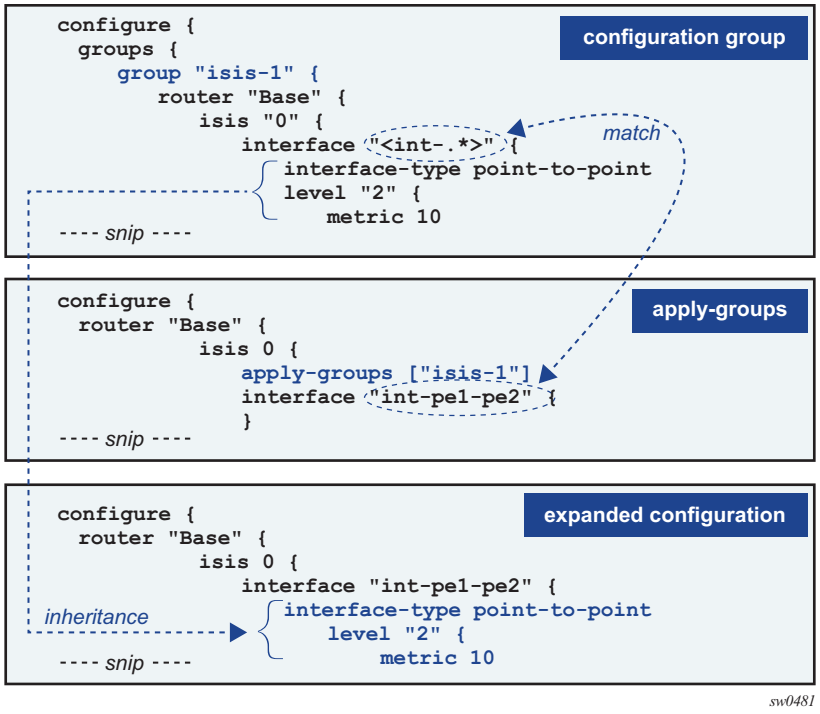
The contents of the load file with the **info** output include the following:

```
/configure router "Base" bgp
    group "external" {
    }
    neighbor "192.168.89.8" {
        group "external"
        prefix-limit ipv4 {
            maximum 200
            log-only true
            threshold 80
        }
    }
```

4.11 Using Configuration Groups

The MD-CLI supports the creation of configuration templates called configuration groups, which can be applied at different branches in the configuration, where the configuration elements are inherited. This is shown in Figure 11.

Figure 11 **Configuration Groups**



The advantage of using configuration groups is that similar configurations can be grouped in a template that is applied at multiple branches in the configuration tree. Subsequent configuration updates are only required in one location. Using groups, configurations can be organized in a logical fashion, such as regional (East vs West) or functional (core-facing vs access-facing parameters). The result is a more compact configuration that is easier to maintain and that reduces the number of configuration and operational errors.

Configuration groups are supported for all configuration branches and its descendants (this includes the configuration groups definition and applying the groups with the **apply-groups** command), with some exceptions

Configuration branches with no configuration groups support include:

- **log event-trigger li**

- **log filter** *filter-id* **entry** *entry-id* **match application**
- **log log-events li**
- **service vprn** *service-name* **log filter** *filter-id* **entry** *entry-id* **match application**
- **system security aaa local-profiles**
- **system security aaa user-template** *user-template-name* **access**
- **system security snmp access** *group*
- **system security snmp usm-community** *community-string*
- **system security user-params local-user user** *user-name*

4.11.1 Creating Configuration Groups

Configuration groups are created in the groups branch of the configuration tree.

```
(ex)[configure]
A:admin@pe1# info
    groups {
        group "isis-backbone" {
            router "Base" {
                isis "0" {
                    interface "int-pe1-pe2" {
                        hello-authentication-key "KrbVPnF6Dg13PM/biw6ErHmrkAHk" hash
                        hello-authentication-type message-digest
                        hello-authentication true
                        interface-type point-to-point
                    }
                }
            }
        }
    }
```

Multiple configuration groups can be created, each with a unique name.

```
(ex)[configure]
A:admin@pe1# info
    groups {
        group "isis-backbone" {
            router "Base" {
                # configuration elements
            }
        }
        group "isis-access" {
            router "Base" {
                # configuration elements
            }
        }
        group "qos-backbone" {
            card "1" {
                # configuration elements
            }
```

```
            port "1/1/1" {
                # configuration elements
            }
            qos {
                # configuration elements
            }
            router "Base" {
                # configuration elements
            }
        }
    }
```

The configuration elements in a configuration group always start at a top-level configuration branch, such as **router**, **qos**, or **card**.

To match on a key of a list entry in a configuration group, an exact match or a regular expression match can be used.

4.11.1.1 Exact Match

With an exact match, configuration elements can only be inherited by the list entry that matches the specified key value. When no list entry is matched, a new list entry is created with the specified key value.

In the following example, interface “int-pe1-pe2” is an exact match. When the group is applied and IS-IS interface “int-pe1-pe2” exists in IS-IS instance 0, the **interface-type** leaf is inherited. If the IS-IS interface does not exist, it is created with the **interface-type** set to **point-to-point**.

```
(ex)[configure]
A:admin@pe1# info
    groups {
        group "isis-backbone" {
            router "Base" {
                isis "0" {
                    interface "int-pe1-pe2" {
                        interface-type point-to-point
                    }
                }
            }
        }
    }
```

4.11.1.2 Regular Expression Match

With a regular expression match, configuration elements can be inherited by all list entries for which the key value matches the regular expression. A list entry cannot be created with a regular expression match.

In the following example, **interface “<.*>”** is a regular expression match that matches any interface name. When the group is applied, all configured IS-IS interfaces in IS-IS instance 0 inherit the **interface-type** leaf.

```
(ex)[configure]
A:admin@pe1# info
    groups {
        group "isis-backbone" {
            router "Base" {
                isis "0" {
                    interface "<.*>" {
                        interface-type point-to-point
                    }
                }
            }
        }
    }
```

4.11.1.2.1 Regular Expression Match Format

A regular expression match is specified as a string with the regular expression enclosed in angle brackets: “<regex-match>”.

The regular expression match is implicitly anchored: a ^ (match-starting position) is added at the beginning of the regular expression and a $ (match-ending position) is added at the end.

The regular expression is a subset of the Extended Regular Expression (ERE) notation as described in section 3.12.1.1.

For example:

- interface “<int-.*>” — matches all interfaces that start with “int-”
- interface “<.*>” — matches all interfaces
- interface “<.*pe[1-3].*>” — matches all interfaces that have “pe1”, “pe2”, or “pe3” in their name

Note: A regular expression match on an encrypted leaf is restricted to a match all: “<.*>”. Any other string enclosed in angle brackets (“<*string*>”) is accepted as an exact match for the encrypted leaf and displayed as a hashed value in the configuration.

4.11.1.3 Conflicting Match Criteria Within a Configuration Group

With a regular expression match, a match criteria conflict can occur if two regular expressions match or if a regular expression and an exact match both match on the same list entry. Conflicting matches within a configuration group are not supported and result in a validation error.

In the following configuration example, both **interface “<int-.*>”** and **interface “int-pe1-pe2”** are matching **isis 0 interface “int-pe1-pe2”**. At validation, this results in a configuration group inheritance failure because of conflicting match criteria:

```
(ex)[configure]
A:admin@pe1# info
    groups {
        group "isis-backbone" {
            router "Base" {
                isis "0" {
                    interface "<int-.*>" {
                        interface-type point-to-point
                        level-capability 2
                    }
                    interface "int-pe1-pe2" {
                        level 2 {
                            hello-interval 1
                        }
                    }
                }
            }
        }
    }

---snip---

    router "Base" {

---snip---

        isis 0 {
            apply-groups ["isis-backbone"]

---snip---
            interface "int-pe1-pe2" {
---snip---

           }
        }
    }

(ex)[configure]
A:admin@pe1# validate
MINOR: MGMT_CORE #2901: configure router "Base" isis 0 interface "int-pe1-pe2" -
 Configuration group inheritance failed -
 conflicting match criteria within group   "isis-backbone"
```

Conflicting match criteria within a configuration group can be avoided by applying multiple configuration groups.

```
*(ex)[configure]
A:admin@pe1# info
    groups {
        group "isis-backbone-common" {
            router "Base" {
                isis "0" {
                    interface "<int-.*>" {
                        interface-type point-to-point
                        level-capability 2
                    }
                }
            }
        }
        group "isis-backbone-custom" {
            router "Base" {
                isis "0" {
                    interface "int-pe1-pe2" {
                        level 2 {
                            hello-interval 1
                        }
                    }
                }
            }
        }
    }

---snip---

    router "Base" {

---snip---

        isis 0 {
            apply-groups ["isis-backbone-custom" "isis-backbone-common"]

---snip---

            interface "int-pe1-pe2" {
---snip---
           }
        }
    }

*(ex)[configure router "Base" isis 0]
A:admin@pe1# validate

*(ex)[configure router "Base" isis 0]
A:admin@pe1# info inheritance
    apply-groups ["isis-backbone-custom" "isis-backbone-common"]
    interface "int-pe1-pe2" {
        ## 'interface-type' inherited from group "isis-backbone-common"
        interface-type point-to-point
        ## 'level-capability' inherited from group "isis-backbone-common"
        level-capability 2
        level 2 {
```

```
            ## 'hello-interval' inherited from group "isis-backbone-custom"
            hello-interval 1
        }
    }
```

4.11.2 Applying Configuration Groups

To inherit configuration elements from a configuration group, apply the group in a branch of the configuration tree with the **apply-groups** statement. For example:

```
(ex)[configure router "Base" isis 0]
A:admin@pe1# info
    apply-groups ["isis-1"]
```

Configuration elements from the corresponding branches where the group is applied are inherited. In the following example, the configuration group “isis-3” has configuration elements in both the **router isis interface** and **router isis level** branch. Because the configuration group is applied at the **router isis interface** branch, only these configuration elements are inherited.

```
(ex)[configure]
A:admin@pe1# info
    groups {
        group "isis-3" {
            router "Base" {
                isis "0" {
                    interface "<int-.*>" {
                        interface-type point-to-point
                        level "2" {
                            metric 30
                        }
                    }
                    level "2" {
                        wide-metrics-only true
                    }
                }
            }
        }
    }

<snip>

   router "Base" {
        isis 0 {
            admin-state enable
            level-capability 2
            area-address [49.0001.0001]
            interface "int-pe1-pe2" {
                apply-groups ["isis-3"]
            }
        }
    }
```

The resulting expanded configuration can be shown with the **info inheritance** command:

```
(ex)[configure]
A:admin@pe1# info inheritance
   router "Base" {
        isis 0 {
            admin-state enable
            level-capability 2
            area-address [49.0001.0001]
            interface "int-pe1-pe2" {
                apply-groups ["isis-3"]
                ## 'interface-type' inherited from group "isis-3"
                interface-type point-to-point
                level 2 {
                    ## 'metric' inherited from group "isis-3"
                    metric 30
                }
            }
        }
    }
```

The following notes apply to configuration groups and the **apply-groups** statements:

- configuration groups cannot be nested; therefore, **apply-groups** statements cannot be part of a configuration group
- configuration groups that are not applied in the configuration do not functionally change the configuration
- configuration groups and **apply-groups** statements are part of the running configuration and are saved in the MD-CLI configuration file

4.11.3 Inheritance Rules

Local configuration elements have precedence over configuration group inheritance.

In the following example, the configuration group “isis-1” contains the configuration element **level-capability 1**, which is not inherited because a corresponding local configuration element exists.

```
(ex)[configure]
A:admin@pe1# info
    groups {
        group "isis-1" {
            router "Base" {
                isis "0" {
                    level-capability 1
                    interface "<int-.*>" {
                        interface-type point-to-point
                        level "2" {
                            metric 10
```

```
                        }
                    }
                }
            }
        }
    }

<snip>

   router "Base" {
        isis 0 {
            apply-groups ["isis-1"]
            admin-state enable
            level-capability 2
            area-address [49.0001.0001]
            interface "int-pe1-pe2" {
            }
        }
    }
```

The resulting expanded configuration after inheritance is shown as follows:

```
(ex)[configure]
A:admin@pe1# info inheritance
   router "Base" {
        isis 0 {
            apply-groups ["isis-1"]
            admin-state enable
            level-capability 2
            area-address [49.0001.0001]
            interface "int-pe1-pe2" {
                ## 'interface-type' inherited from group "isis-1"
                interface-type point-to-point
                level 2 {
                    ## 'metric' inherited from group "isis-1"
                    metric 10
                }
            }
        }
    }
```

Up to eight configuration groups can be applied to a configuration branch. The configuration order determines the inheritance precedence:

- configuration elements in the first listed group have the highest precedence
- configuration elements in the last listed group have the lowest precedence

In the following example, both configuration groups “isis-1” and “isis-2” set an interface **level 2 metric**. Because configuration group “isis-2” is listed first in the **apply-groups**, its configuration elements have precedence. The **interface-type** configuration element is inherited from group “isis-1” because a corresponding configuration element is not present in group “isis-2” nor is it locally configured.

```
(ex)[configure]
```

```
A:admin@pe1# info
    groups {
        group "isis-1" {
            router "Base" {
                isis "0" {
                    level-capability 1
                    interface "<int-.*>" {
                        interface-type point-to-point
                        level "2" {
                            metric 10
                        }
                    }
                }
            }
        }
        group "isis-2" {
            router "Base" {
                isis "0" {
                    interface "<int-.*>" {
                        level "2" {
                            metric 20
                        }
                    }
                }
            }
        }
    }

<snip>

   router "Base" {
        isis 0 {
            apply-groups ["isis-2" "isis-1"]
            admin-state enable
            level-capability 2
            area-address [49.0001.0001]
            interface "int-pe1-pe2" {
            }
        }
    }
```

The resulting expanded configuration after inheritance is shown as follows:

```
(ex)[configure]
A:admin@pe1# info inheritance
   router "Base" {
        isis 0 {
            apply-groups ["isis-2" "isis-1"]
            admin-state enable
            level-capability 2
            area-address [49.0001.0001]
            interface "int-pe1-pe2" {
                ## 'interface-type' inherited from group "isis-1"
                interface-type point-to-point
                level 2 {
                    ## 'metric' inherited from group "isis-2"
                    metric 20
                }
```

```
            }
        }
    }
```

Configuration groups can be applied at different hierarchical branches. The hierarchy determines the inheritance precedence.

Configuration elements in groups applied at a lower-level branch have precedence over configuration elements in groups applied at a higher-level branch.

In the following example, all configuration groups set an interface **level 2 metric**. Because configuration group “isis-3” is applied at the lowest level, its configuration elements have precedence. The **interface-type** configuration element is also inherited from group “isis-3” for the same reason. As explained earlier, the **level-capability** configuration element from group “isis-1” has lower precedence than the local configured value. The **wide-metric-only** configuration element from group “isis-3” is not inherited because the group is applied at the interface branch and only configuration elements at that level or lower can be inherited.

```
(ex)[configure]
A:admin@pe1# info
    groups {
        group "isis-1" {
            router "Base" {
                isis "0" {
                    level-capability 1
                    interface "<int-.*>" {
                        interface-type point-to-point
                        level "2" {
                            metric 10
                        }
                    }
                }
            }
        }
        group "isis-2" {
            router "Base" {
                isis "0" {
                    interface "<int-.*>" {
                        level "2" {
                            metric 20
                        }
                    }
                }
            }
        }
        group "isis-3" {
            router "Base" {
                isis "0" {
                    interface "<int-.*>" {
                        interface-type point-to-point
                        level "2" {
                            metric 30
                        }
                    }
```

```
                    level "2" {
                        wide-metrics-only true
                    }
                }
            }
        }
    }

<snip>

   router "Base" {
        isis 0 {
            apply-groups ["isis-2" "isis-1"]
            admin-state enable
            level-capability 2
            area-address [49.0001.0001]
            interface "int-pe1-pe2" {
                apply-groups ["isis-3"]
            }
        }
    }
```

The resulting expanded configuration after inheritance is shown as follows:

```
(ex)[configure]
A:admin@pe1# info inheritance
   router "Base" {
        isis 0 {
            apply-groups ["isis-2" "isis-1"]
            admin-state enable
            level-capability 2
            area-address [49.0001.0001]
            interface "int-pe1-pe2" {
                apply-groups ["isis-3"]
                ## 'interface-type' inherited from group "isis-3"
                interface-type point-to-point
                level 2 {
                    ## 'metric' inherited from group "isis-3"
                    metric 30
                }
            }
        }
    }
```

Inheritance rules for leaf-lists are the same as for a single leaf, whether the list is a system-ordered leaf list (for example, **configure router interface** *interface-name* **if-attribute admin-group** *value*) or a user-ordered leaf list (for example, **configure router bgp export policy** *value*). It is not possible to add values to an existing leaf-list through configuration group inheritance.

Inheritance rules for user-ordered (container) lists (for example, **configure policy-options policy-statement** *name* **named-entry** *entry-name*) are:

- list order is ignored for user-ordered list entry matching

- unmatched list entries in the configuration group definition and its descendant configuration elements are inherited in the locally-configured user-ordered list. Newly-created list entries are appended at the end in the order they appear in the configuration group definition.
- descendant configuration elements of matched user-ordered list entries are candidates for inheritance

4.11.4 Displaying the Expanded Configuration

After configuring and applying configuration groups, the expanded configuration should be reviewed before commit. The expanded configuration at a configuration branch can be displayed with the **info inheritance** command. By default, this command displays the expanded candidate configuration. To display the expanded running configuration, use **info running inheritance**.

All statements that are inherited from a configuration group are tagged with a system comment.

```
(ex)[configure router "Base" isis 0 interface "int-pe1-pe2"]
A:admin@pe1# info inheritance
    ## 'interface-type' inherited from group "isis-1"
    interface-type point-to-point
    level 2 {
        ## 'metric' inherited from group "isis-2"
        metric 20
    }
```

Use the regular expression pattern match **info inheritance | match '^[ ]*##' invert-match** to suppress the system comments in the output of **info inheritance**.

```
(ex)[configure router "Base" isis 0 interface "int-pe1-pe2"]
A:admin@pe1# info inheritance | match '^[ ]*##' invert-match
    interface-type point-to-point
    level 2 {
        metric 20
    }
```

Note: Conflicting matches are detected at validation. The **info inheritance** command may display an inherited configuration element that is part of a conflicting match criteria.

4.11.5 Authentication, Authorization, and Accounting (AAA) in Configuration Groups

User profiles can restrict the configuration branches that a user can change. When configuration groups are used, these user profiles should be enhanced to restrict the creation or inheritance of configuration elements in these branches.

In the following example, user admin2 has no access to the **sap-ingress** configuration branch.

```
(ex)[configure qos]
A:admin2@pe1# sap-ingress high-bw
MINOR: MGMT_CORE #2020: Permission denied - unauthorized use of 'sap-ingress'
```

This is enforced via the following entry in the local user profile:

```
(ro)[configure system security aaa local-profiles profile "restricted-admin"]
A:admin@pe1# info
<snip>
    entry 200 {
        action deny
        match "configure qos sap-ingress"
    }
```

Using configuration groups, user admin2 can still create or change **sap-ingress** QoS policies:

```
(ex)[configure groups]
A:admin2@pe1# info
    group "qos-1" {
        qos {
            sap-ingress "high-bw" {
                queue "1" {
                    rate {
                        pir 200000
                    }
                }
            }
        }
    }

(ex)[configure qos]
A:admin2@pe1# info
    apply-groups ["qos-1"]

<snip>
```

The result of the inheritance is not visible to user admin2 because the **info** command is also subject to the user profile rules.

```
(ex)[configure qos]
A:admin2@pe1# info inheritance
```

```
    apply-groups ["qos-1"]
    md-auto-id {
        qos-policy-id-range {
            start 1000
            end 2000
        }
    }
```

The admin user who has full privileges can see the inherited configuration, which includes the **sap-ingress** policy created by user admin2.

```
(ro)[configure qos]
A:admin@pe1# info inheritance
    apply-groups ["qos-1"]
    md-auto-id {
        qos-policy-id-range {
            start 1000
            end 2000
        }
    }
    ## 'sap-ingress "high-bw"' inherited from group "qos-1"
    sap-egress "high-bw" {
        ## 'queue 1' inherited from group "qos-1"
        queue 1 {
            ## 'rate' inherited from group "qos-1"
            rate {
                ## 'pir' inherited from group "qos-1"
                pir 200000
            }
        }
    }
```

To prevent user admin2 from creating sap-ingress QoS policies using configuration groups, the AAA profile of the user can be enhanced. For example, an entry can be added in the local user profile:

```
(ro)[configure system security aaa local-profiles profile "restricted-admin"]
A:admin@pe1# info
<snip>
    entry 200 {
        action deny
        match "configure qos sap-ingress"
    }
    entry 210 {
        action deny
        match "configure groups group qos sap-ingress"
    }
```

This configuration removes the privileges for user admin2 to create **sap-ingress** QoS policies using configuration groups:

```
(ex)[configure groups group "qos-1" qos]
A:admin2@pe1# sap-ingress high-bw
MINOR: MGMT_CORE #2020: Permission denied - unauthorized use of 'sap-ingress'
```

4.11.6 Configuration Group Example

The following configuration is an example of configuring IS-IS interface parameters using configuration groups.

In this example, all backbone IS-IS interface configuration parameters are part of the “isis-bb-interface” configuration group. A regular expression match “<int-.*>” is used to match on all backbone IS-IS interface names that start with “int-“. The system loopback interface does not match the regular expression, so cannot inherit the configuration elements from the group.

The “isis-bb-interface” configuration group is applied at the router “Base”, IS-IS instance 0 branch. When a new IS-IS backbone interface is added with a name that starts with “int-“, it also inherits the configuration elements from the configuration group.

```
(ex)[configure]
A:admin@pe1# info
    groups {
        group "isis-bb-interface" {
            router "Base" {
                isis "0" {
                    interface "<int-.*>" {
                        hello-authentication-key "KrbVPnF6Dg13PM/biw6ErHmrkAHk" hash
                        hello-authentication-type message-digest
                        hello-padding adaptive
                        hello-authentication true
                        interface-type point-to-point
                    }
                }
            }
        }
    }

<snip>

    router "Base" {
        isis 0 {
            apply-groups ["isis-bb-interface"]
            admin-state enable
            ipv4-routing true
            ipv6-routing native
            level-capability 2
            area-address [49.0001.0001]
            multi-topology {
                ipv6-unicast true
            }
            interface "int-pe1-pe2" {
            }
            interface "int-pe1-pe3" {
```

```
            }
            interface "system" {
                passive true
            }
            level 2 {
                wide-metrics-only true
            }
        }
    }
```

The resulting expanded configuration after inheritance is shown as follows:

```
(ex)[configure router "Base" isis 0]
A:admin@pe1# info inheritance
    apply-groups ["isis-bb-interface"]
    admin-state enable
    ipv4-routing true
    ipv6-routing native
    level1-capability 2
    area-address [49.0001.0001]
    multi-topology {
        ipv6-unicast true
    }
    interface "int-pe1-pe2" {
        ## 'hello-authentication-key' inherited from group "isis-bb-interface"
        hello-authentication-key "KrbVPnF6Dg13PM/biw6ErHmrkAHk" hash
        ## 'hello-authentication-type' inherited from group "isis-bb-interface"
        hello-authentication-type message-digest
        ## 'hello-padding' inherited from group "isis-bb-interface"
        hello-padding adaptive
        ## 'hello-authentication' inherited from group "isis-bb-interface"
        hello-authentication true
        ## 'interface-type' inherited from group "isis-bb-interface"
        interface-type point-to-point
    }
    interface "int-pe1-pe3" {
        ## 'hello-authentication-key' inherited from group "isis-bb-interface"
        hello-authentication-key "KrbVPnF6Dg13PM/biw6ErHmrkAHk" hash
        ## 'hello-authentication-type' inherited from group "isis-bb-interface"
        hello-authentication-type message-digest
        ## 'hello-padding' inherited from group "isis-bb-interface"
        hello-padding adaptive
        ## 'hello-authentication' inherited from group "isis-bb-interface"
        hello-authentication true
        ## 'interface-type' inherited from group "isis-bb-interface"
        interface-type point-to-point
    }
    interface "system" {
        passive true
    }
    level 2 {
        wide-metrics-only true
    }
```

The resulting expanded configuration after inheritance is shown as follows, without system comments:

```
(ex)[configure router "Base" isis 0]
A:admin@pe1# info inheritance | match '^[ ]*##' invert-match
    apply-groups ["isis-bb-interface"]
    admin-state enable
    ipv4-routing true
    ipv6-routing native
    level-capability 2
    area-address [49.0001.0001]
    multi-topology {
        ipv6-unicast true
    }
    interface "int-pe1-pe2" {
        hello-authentication-key "KrbVPnF6Dg13PM/biw6ErHmrkAHk" hash
        hello-authentication-type message-digest
        hello-padding adaptive
        hello-authentication true
        interface-type point-to-point
    }
    interface "int-pe1-pe3" {
        hello-authentication-key "KrbVPnF6Dg13PM/biw6ErHmrkAHk" hash
        hello-authentication-type message-digest
        hello-padding adaptive
        hello-authentication true
        interface-type point-to-point
    }
    interface "system" {
        passive true
    }
    level 2 {
        wide-metrics-only true
    }
```

4.11.7 Caveats

The following caveats apply to configuration groups.

- Configuration groups are available only for the model-driven management interface configuration mode and the supported Nokia YANG models.
- When configuration groups are used with NETCONF, the <get-config> operation returns the pre-expanded configuration, including the configuration groups definitions and **apply-groups** configuration elements. The expanded configuration, including the inherited configuration elements, cannot be returned with NETCONF.
- When configuration groups are used with gNMI, the Get RPC command returns the pre-expanded configuration, including the configuration groups definitions and **apply-groups** configuration elements. The expanded configuration, including the inherited configuration elements, cannot be returned with gNMI.

For more information about NETCONF and gNMI, refer to the *7450 ESS, 7750 SR, 7950 XRS, and VSR System Management Guide*.

4.12 Viewing the Status of the Local Datastores

An MD-CLI session in exclusive configuration mode acquires an explicit lock for both the global candidate and running configuration datastores. This is achieved by executing the **configure exclusive** command (in the implicit configuration workflow) or the **edit-config exclusive** command (in the explicit configuration workflow).

An explicit lock can also be obtained via:

- NETCONF or gRPC sessions. Refer to the *7450 ESS, 7750 SR, 7950 XRS, and VSR System Management Guide* for more information.
- a private exclusive configuration session. See Exclusive Private Configuration Session for more information.

To view the lock status of the datastores, the following **show** command is available:

show system management-interface datastore-locks [**detail**]

The **detail** option displays information about any model-driven interface session that impacts the datastore locks. MD-CLI read-only sessions, for example, do not impact the datastore locks.

```
(ro)[]
A:admin@node-2# show system management-interface datastore-locks detail
===============================================================================
Session ID  Region                Datastore                    Lock State
   Username                       Session Mode                 Idle Time
   Session Type                   From
-------------------------------------------------------------------------------
 69         configure             Candidate, Running           Locked
   admin                          Exclusive                    0d 00:01:48
   MD-CLI                         192.168.144.87
-------------------------------------------------------------------------------
Number of sessions: 1
'#' indicates the current active session
===============================================================================
```

The **configuration-sessions** command displays the same information as the **datastore-locks detail** command, but for all configuration sessions regardless of whether the session has a lock on the datastore.

```
(ro)[]
A:admin@node-2# show system management-interface configuration-sessions
===============================================================================
Session ID  Region                Datastore                    Lock State
   Username                       Session Mode                 Idle Time
   Session Type                   From
-------------------------------------------------------------------------------
#65         configure             Candidate                    Unlocked
   admin                          Private                      0d 00:00:00
```

```
   MD-CLI                          192.168.144.87
 66         configure              Candidate                      Unlocked
   admin                           Private                        0d 00:05:41
   MD-CLI                          192.168.144.87
 67         configure              Candidate                      Unlocked
   admin                           Private                        0d 00:05:08
   MD-CLI                          192.168.144.87
 68         configure              Candidate                      Unlocked
   admin                           Read-Only                      0d 00:02:25
   MD-CLI                          192.168.144.87
 69         configure              Candidate, Running             Locked
   admin                           Exclusive                      0d 00:01:54
   MD-CLI                          192.168.144.87
-------------------------------------------------------------------------------
Number of sessions: 5
'#' indicates the current active session
===============================================================================
```

4.12.1 Unlocking a Locked Datastore

A datastore lock that has been acquired by any model-driven session can be administratively removed by using the following **admin** command:

admin disconnect session-id *session-id*

For example, to disconnect the MD-CLI session indicated in the preceding **show** command output, issue the **admin** command as follows:

```
[]
A:admin@node-2# admin disconnect session-id 10
```

Disconnecting an MD-CLI session (or any model-driven session, including NETCONF and gRPC) that acquired a datastore lock has the following results:

- any uncommitted changes in the candidate configuration datastore are discarded
- the session is terminated
- the explicit lock is released

5 Troubleshooting

5.1 Debug commands

The **debug** command is not natively supported in the MD-CLI. The command can be executed from the classic CLI. The **//** command can be used to switch to the classic CLI engine from the MD-CLI engine. Both debug and /debug are supported in the classic CLI.

```
[]
A:admin@node-2# //
INFO: CLI #2051: Switching to the classic CLI engine
A:node-2# debug router bgp packets
A:node-2#
A:node-2# /debug router bgp packets
A:node-2#
```

5.2 Logging Debug Events in the MD-CLI

The following MD-CLI commands can be used to log debug events to an active CLI session.

```
— configure
    — log
        — log-id [id] number
            — source
                — debug boolean
            — destination
                — cli
                    — max-entries number
```

The following example shows the configuration for debug events to be stored in destination CLI log identifier 7. The log entries wrap at 50 entries (the configured value of **max-entries**).

```
(ex)[configure log]
A:admin@node-2# log-id 7

*(ex)[configure log log-id 7]
A:admin@node-2# source debug

*(ex)[configure log log-id 7]
A:admin@node-2# destination cli max-entries 50

*(ex)[configure log log-id 7]
```

```
A:admin@node-2# info
    source {
        debug true
    }
    destination {
        cli {
            max-entries 50
        }
    }
```

After the **commit** command has been issued to include the log in the running configuration, the following **tools** command can be executed in the CLI session that will be used to display outputs of the debug events. Refer to the *7450 ESS, 7750 SR, 7950 XRS, and VSR System Management Guide* for more information about the **tools** command.

```
(ex)[tools perform log]
A:admin@node-2# subscribe-to log-id 7

(ex)[]
A:admin@node-2#
```

The events can be displayed using the */***show log** command and cleared using the */* **clear log** command.

```
[]
A:admin@node-2# show log log-id 7
===============================================================================
Event Log 7
===============================================================================
Description : (Not Specified)
Log contents  [size=50   next event=2  (not wrapped)]

<snip>

[]
A:admin@node-2# clear log log-id 7

[]
A:admin@node-2#
```

To terminate the output of the logs to the CLI session, use the **unsubscribe-from** command as shown.

```
(ex)[]
A:admin@node-2# tools perform log unsubscribe-from log-id 7

(ex)[]
A:admin@node-2#
```

6 Advanced Tips and Features

6.1 Discarding Changes in Specific Contexts

The **discard** command can be used only from the top level of the configuration branch. From any working context (including the configure context), the **discard / configure** command can be used.

```
*(ex)[configure router "Base" ospf 0 timers spf-wait]
A:admin@node-2# discard
MINOR: MGMT_CORE #2203: Invalid element - 'discard' not allowed in 'spf-wait'

*(ex)[configure router "Base" ospf 0 timers spf-wait]
A:admin@node-2# discard /configure

(ex)[configure router "Base" ospf 0 timers spf-wait]
A:admin@node-2#
```

However, the **discard /configure** command removes all configuration statements from the candidate configuration datastore. To discard changes from a specific context, the output from the **compare** command can be used to copy and paste configuration statements from within a working context.

By default, the **compare** command uses the baseline datastore as the base reference. Therefore, any new configuration in the candidate datastore is displayed with a preceding plus (+) sign. When the **compare** command uses the candidate datastore as the base reference, the **compare** output displays any new configuration in the candidate datastore with a preceding minus (-) sign, indicating that these configuration elements are not present in the baseline datastore. The configuration elements preceded with a minus (-) sign can be used to discard configurations from the specific context from which the **compare** command was issued.

In the following configuration example, the **lsa-generate** timers are modified.

```
(ex)[configure router "Base" ospf 0 timers lsa-generate]
A:admin@node-2# info detail
    max-lsa-wait 5000
    lsa-initial-wait 5000
    lsa-second-wait 5000

(ex)[configure router "Base" ospf 0 timers lsa-generate]
A:admin@node-2# lsa-initial-wait 1000

*(ex)[configure router "Base" ospf 0 timers lsa-generate]
A:admin@node-2# lsa-second-wait 2000

*(ex)[configure router "Base" ospf 0 timers lsa-generate]
A:admin@node-2# max-lsa-wait 3000
```

```
*(ex)[configure router "Base" ospf 0 timers lsa-generate]
A:admin@node-2# info
    max-lsa-wait 3000
    lsa-initial-wait 1000
    lsa-second-wait 2000
```

By default, the **compare** command shows the new configuration using the baseline datastore as the reference:

```
*(ex)[configure router "Base" ospf 0 timers lsa-generate]
A:admin@node-2# compare
+   max-lsa-wait 3000
+   lsa-initial-wait 1000
+   lsa-second-wait 2000
```

The following shows the **compare** command output when the command is executed with the candidate datastore as the reference.

```
*(ex)[configure router "Base" ospf 0 timers lsa-generate]
A:admin@node-2# compare from candidate to baseline
-   max-lsa-wait 3000
-   lsa-initial-wait 1000
-   lsa-second-wait 2000
```

To discard the **max-lsa-wait** and **lsa-initial-wait** timer changes, the first two lines from the **compare** command output can be copied and pasted while in the specified context. The **info detail** command shows that the timer changes have reverted to their default values.

```
*(ex)[configure router "Base" ospf 0 timers lsa-generate]
A:admin@node-2# -   max-lsa-wait 3000

*(ex)[configure router "Base" ospf 0 timers lsa-generate]
A:admin@node-2# -   lsa-initial-wait 1000

*(ex)[configure router "Base" ospf 0 timers lsa-generate]
A:admin@node-2#

*(ex)[configure router "Base" ospf 0 timers lsa-generate]
A:admin@node-2# info detail
    max-lsa-wait 5000
    lsa-initial-wait 5000
    lsa-second-wait 2000

*(ex)[configure router "Base" ospf 0 timers lsa-generate]
A:admin@node-2
```

7 Converting Classic CLI to MD-CLI Configuration

When mixed or model-driven interface configuration mode is enabled, the SR OS router automatically converts the running configuration to the MD-CLI format in memory. Any additional configuration entered in the classic CLI can easily be converted to the MD-CLI format. Converting the classic CLI configuration avoids the need to recreate the configuration in the MD-CLI and ensures all applicable configuration elements are properly converted to the MD-CLI format by the system. Any Nokia SR OS router, including a virtual simulator, Virtualized Service Router (VSR), or lab router, can be used to convert configurations.

The following procedure converts additional configuration snippets that are entered in the classic CLI or copied and pasted into the classic CLI from other sources, such as from another router, that were not present when the configuration was first converted. The converted configurations can then be copied and pasted in the MD-CLI on another target router or saved to a file for later use. For these additional conversions, mixed configuration mode must be enabled.

Refer to Management Interface Configuration Mode in the *7450 ESS, 7750 SR, 7950 XRS, and VSR System Management Guide* for more information about the management interface configuration mode.

The following example converts a configuration snippet which changes the system name.

Step 1. Ensure mixed management interface configuration mode is enabled and persistent (from the classic CLI engine) and log out of the current session.

```
*A:node-2# /configure system management-interface configuration-mode mixed

Applying Changes to Model-Driven Database ... OK
*A:node-2#logout
```

Step 2. Log in to a new CLI session.

```
Login: admin
Password:
```

Step 3. (Optional) Save the configuration in the classic CLI format. The saved configuration can be used to return to a known baseline configuration.

```
*A:node-2# admin save
Writing configuration to cf3:config.cfg
Saving configuration ... OK
Completed.
```

Step 4. Switch to the MD-CLI engine and capture the configuration in the MD-CLI format to a file that can be used for later comparison. Because the **admin save** command is not supported in mixed configuration mode, use the **admin show configuration** command and redirect the output to the file.

```
*A:node-2# //
INFO: CLI #2052: Switching to the MD-CLI engine

[]
A:admin@node-2# admin show configuration > cf3:md-config.cfg
```

Step 5. Return to the classic CLI engine and enter the configuration to be converted.

```
[]
A:admin@node-2# //
INFO: CLI #2051: Switching to the classic CLI engine
A:node-2# configure system name new-node-7
```

Step 6. Switch to the MD-CLI engine. Compare the running configuration to the saved configuration file to display the additional configuration in the MD-CLI format.

```
*A:new-node-7# //
INFO: CLI #2052: Switching to the MD-CLI engine

[]
A:admin@new-node-7# configure global
INFO: CLI #2054: Entering global configuration mode

[gl:configure]
A:admin@new-node-7# compare from url cf3:md-config.cfg
    system {
-       name "node-2"
+       name "new-node-7"
    }
```

Step 7. Copy and paste the differences in the MD-CLI engine on the target router or redirect to a file for later use. See Copying Configuration Elements and Using the File Redirect Option for more information.

The following example illustrates the conversion of a longer and more complex configuration that was pasted into the classic CLI using the above procedure.

```
*A:node-2>config>service# info
----------------------------------------------
        vpls 2000 name "2000" customer 1 create
            vxlan instance 1 vni 2000 create
            exit
            bgp
                route-distinguisher 65001:2000
                route-target export target:65000:2000 import target:65000:2000
            exit
            bgp-evpn
                unknown-mac-route
```

```
                vxlan bgp 1 vxlan-instance 1
                    shutdown
                exit
            exit
            stp
                shutdown
            exit
            site "site-1" create
                site-id 1
                sap 1/1/1:1
                no shutdown
            exit
            sap 1/1/1:1 create
                no shutdown
            exit
            no shutdown
        exit
-------------------------------------------

*[ex:configure]
A:admin@node-2# compare from url cf3:md-config.cfg
+   service {
+       vpls "2000" {
+           admin-state enable
+           customer "1"
+           vxlan {
+               instance 1 {
+                   vni 2000
+               }
+           }
+           bgp 1 {
+               route-distinguisher "65001:2000"
+               route-target {
+                   export "target:65000:2000"
+                   import "target:65000:2000"
+               }
+           }
+           bgp-evpn {
+               routes {
+                   mac-ip {
+                       unknown-mac true
+                   }
+               }
+               vxlan 1 {
+                   admin-state enable
+                   vxlan-instance 1
+               }
+           }
+           bgp-mh-site "site-1" {
+               admin-state enable
+               id 1
+               sap 1/1/1:1
+           }
+           sap 1/1/1:1 {
+           }
+       }
+   }
```

Note: Consider the following when converting classic CLI to MD-CLI configuration:

- Revert to a saved classic CLI configuration or save changes to the classic CLI configuration in each CLI engine when converting a configuration snippet so that the MD-CLI comparison displays only the snippet that is converted.
- Avoid configuration snippets that overlap with the existing configuration, including default configuration snippets, because they may not be displayed as a difference. A minimum configuration is recommended to ensure that all differences are displayed.
- Use the same type of router for the conversion as the router the configuration is intended for, because some CLI commands are specific to a product family or chassis.

8 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, *Integrated Local Management Interface (ILMI) Version 4.0*

AF-PHY-0086.001, *Inverse Multiplexing for ATM (IMA) Specification Version 1.1*

AF-TM-0121.000, *Traffic Management Specification Version 4.1*

AF-TM-0150.00, *Addendum to Traffic Management v4.1 optional minimum desired cell rate indication for UBR*

GR-1113-CORE, *Asynchronous Transfer Mode (ATM) and ATM Adaptation Layer (AAL) Protocols Generic Requirements, Issue 1*

GR-1248-CORE, *Generic Requirements for Operations of ATM Network Elements (NEs), Issue 3*

ITU-T I.432.1, *B-ISDN user-network interface - Physical layer specification: General characteristics (02/99)*

ITU-T I.610, *B-ISDN operation and maintenance principles and functions (11/95)*

RFC 1626, *Default IP MTU for use over ATM AAL5*

RFC 2684, *Multiprotocol Encapsulation over ATM Adaptation Layer 5*

Bidirectional Forwarding Detection (BFD)

RFC 5880, *Bidirectional Forwarding Detection (BFD)*

RFC 5881, *Bidirectional Forwarding Detection (BFD) IPv4 and IPv6 (Single Hop)*

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*

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* (IS-IS)

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-sidr-origin-validation-signaling-04, *BGP Prefix Origin Validation State Extended Community*

draft-uttaro-idr-bgp-persistence-03, *Support for Long-lived BGP Graceful Restart*

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 8751, *BGP - Link State (BGP-LS) Advertisement of IGP Traffic Engineering Performance Metric Extensions*

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 802.3ab, *1000BASE-T*

IEEE 802.3ac, *VLAN Tag*

IEEE 802.3ad, *Link Aggregation*

IEEE 802.3ae, *10 Gb/s Ethernet*

IEEE 802.3ah, *Ethernet in the First Mile*

IEEE 802.3ba, *40 Gb/s and 100 Gb/s Ethernet*

IEEE 802.3i, *Ethernet*

IEEE 802.3u, *Fast Ethernet*

IEEE 802.3x, *Ethernet Flow Control*

IEEE 802.3z, *Gigabit Ethernet*

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-ac-df-01, *AC-Influenced Designated Forwarder Election for EVPN*

draft-ietf-bess-evpn-irb-mcast-04, *EVPN Optimized Inter-Subnet Multicast (OISM) Forwarding* (ingress replication)

draft-ietf-bess-evpn-pref-df-04, *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*

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*

Intermediate System to Intermediate System (IS-IS)

draft-ietf-isis-te-app-07, *IS-IS TE Attributes per application*

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, Nov. 2002, *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)

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 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 4632, *Classless Inter-domain Routing (CIDR): The Internet Address Assignment and Aggregation Plan*

RFC 5082, *The Generalized TTL Security Mechanism (GTSM)*

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 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-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 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)*

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"*

Management

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*
openconfig-acl.yang version 1.0.0, *ACL Module*
openconfig-bgp.yang version 3.0.1, *BGP Module*
openconfig-bgp-common.yang version 3.0.1, *BGP Common Module*
openconfig-bgp-common-multiprotocol.yang version 3.0.1, *BGP Common Multiprotocol Module*
openconfig-bgp-common-structure.yang version 3.0.1, *BGP Common Structure Module*
openconfig-bgp-global.yang version 3.0.1, *BGP Global Module*
openconfig-bgp-neighbor.yang version 3.0.1, *BGP Neighbor Module*
openconfig-bgp-peer-group.yang version 3.0.1, *BGP Peer Group Module*
openconfig-bgp-policy.yang version 4.0.1, *BGP Policy Module*
openconfig-if-aggregate.yang version 2.0.0, *Interfaces Aggregated Model*
openconfig-if-ethernet.yang version 2.0.0, *Interfaces Ethernet Model*
openconfig-if-ip.yang version 2.0.0, *Interfaces IP Module*
openconfig-if-ip-ext.yang version 2.0.0, *Interfaces IP Extensions Module*
openconfig-interfaces.yang version 2.0.0, *Interfaces Module*
openconfig-isis.yang version 0.3.0, *IS-IS Module*
openconfig-isis-lsp.yang version 0.3.0, *IS-IS LSP Module*
openconfig-isis-routing.yang version 0.3.0, *IS-IS Routing Module*
openconfig-lacp.yang version 1.1.0, *LACP Module*
openconfig-lldp.yang version 0.1.0, *LLDP Module*
openconfig-local-routing.yang version 1.0.1, *Local Routing Module*
openconfig-network-instance.yang version 0.8.0, *Network Instance Module*
openconfig-packet-match.yang version 1.0.0, *Packet Match Module*
openconfig-relay-agent.yang version 0.1.0, *Relay Agent Module*
openconfig-routing-policy.yang version 3.0.0, *Routing Policy Module*
openconfig-telemetry.yang version 0.4.1, *Telemetry Module*
openconfig-vlan.yang version 2.0.0, *VLAN Module*
RFC 1157, *A Simple Network Management Protocol (SNMP)*

RFC 1212, *Concise MIB Definitions*

RFC 1213, *Management Information Base for Network Management of TCP/IP-based Internets: MIB-II*

RFC 1215, *A Convention for Defining Traps for use with the SNMP*

RFC 1724, *RIP Version 2 MIB Extension*

RFC 1901, *Introduction to Community-based SNMPv2*

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 3164, *The BSD syslog Protocol*

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 3339, *Date and Time on the Internet: Timestamps*

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 3419, *Textual Conventions for Transport Addresses*

RFC 3498, *Definitions of Managed Objects for Synchronous Optical Network (SONET) Linear Automatic Protection Switching (APS) Architectures*

RFC 3584, *Coexistence between Version 1, Version 2, and Version 3 of the Internet-standard Network Management Framework*

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 3826, *The Advanced Encryption Standard (AES) Cipher Algorithm in the SNMP User-based Security Model*

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 4511, *Lightweight Directory Access Protocol (LDAP): The Protocol*

RFC 4513, *Lightweight Directory Access Protocol (LDAP): Authentication Methods and Security Mechanisms* (TLS)

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 5246, *The Transport Layer Security (TLS) Protocol Version 1.2* (TLS client, RSA public key)

RFC 6991, *Common YANG Data Types*

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 7420, *Path Computation Element Communication Protocol (PCEP) Management Information Base (MIB) Module*

RFC 7950, *The YANG 1.1 Data Modeling Language*

RFC 7951, *JSON Encoding of Data Modeled with YANG*

SFLOW-MIB, *sFlow MIB Version 1.3 (Draft 5)*

Multiprotocol Label Switching (MPLS)

draft-ietf-teas-sr-rsvp-coexistence-rec-02, *Recommendations for RSVP-TE and Segment Routing LSP co-existence*

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 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 7895, *YANG Module Library*

Open Shortest Path First (OSPF)

draft-ietf-ospf-ospfv3-lsa-extend-13, *OSPFv3 LSA Extendibility*

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*

OpenFlow

TS-007, *OpenFlow Switch Specification Version 1.3.1* (OpenFlow-hybrid switches)

Path Computation Element Protocol (PCEP)

draft-alvarez-pce-path-profiles-04, *PCE Path Profiles*

draft-ietf-pce-segment-routing-08, *PCEP Extensions for Segment Routing*

draft-ietf-pce-stateful-pce-14, *PCEP Extensions for Stateful PCE*

RFC 5440, *Path Computation Element (PCE) Communication Protocol (PCEP)*

RFC 8281, *PCEP Extensions for PCE-initiated LSP Setup in a Stateful PCE Model*

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 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*

MFA Forum 9.0.0, *The Use of Virtual trunks for ATM/MPLS Control Plane Interworking*

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-filsfils-spring-segment-routing-policy-05, *Segment Routing Policy for Traffic Engineering*

draft-gredler-idr-bgp-ls-segment-routing-ext-03, *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-bgp-prefix-sid-26, *Segment Routing Prefix SID extensions for BGP*

draft-ietf-idr-segment-routing-te-policy-02, *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-isis-segment-routing-extensions-04, *IS-IS Extensions for Segment Routing*

draft-ietf-ospf-mpls-elc-12, *Signaling Entropy Label Capability and Entropy Readable Label-stack Depth Using OSPF* (advertising ELC)

draft-ietf-ospf-segment-routing-extensions-04, *OSPF Extensions for Segment Routing*

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-ldp-interop-09, *Segment Routing interworking with LDP*

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)

Synchronous Optical Networking (SONET)/Synchronous Digital Hierarchy (SDH)

ANSI T1.105.03, *Jitter Network Interfaces*

ANSI T1.105.06, *Physical Layer Specifications*

ANSI T1.105.09, *Network Timing and Synchronization*

ITU-T G.703, *Physical/electrical characteristics of hierarchical digital interfaces*

ITU-T G.707, *Network node interface for the synchronous digital hierarchy (SDH)*

ITU-T G.813, *Timing characteristics of SDH equipment slave clocks (SEC)*

ITU-T G.823, *The control of jitter and wander within digital networks which are based on the 2048 kbit/s hierarchy*

ITU-T G.824, *The control of jitter and wander within digital networks which are based on the 1544 kbit/s hierarchy*

ITU-T G.825, *The control of jitter and wander within digital networks which are based on the synchronous digital hierarchy (SDH)*

ITU-T G.841, *Types and Characteristics of SDH Networks Protection Architecture, issued in October 1998 and as augmented by Corrigendum 1, issued in July 2002*

ITU-T G.957, *Optical interfaces for equipments and systems relating to the synchronous digital hierarchy*

Time Division Multiplexing (TDM)

ANSI T1.403, *DS1 Metallic Interface Specification*

ANSI T1.404, *DS3 Metallic Interface Specification*

Timing

GR-1244-CORE, *Clocks for the Synchronized Network: Common Generic Criteria, Issue 3, May 2005*

GR-253-CORE, *SONET Transport Systems: Common Generic Criteria. Issue 3, September 2000*

IEEE 1588-2008, *IEEE Standard for a Precision Clock Synchronization Protocol for Networked Measurement and Control Systems*

ITU-T G.781, *Synchronization layer functions, issued 09/2008*

ITU-T G.813, *Timing characteristics of SDH equipment slave clocks (SEC), issued 03/2003*

ITU-T G.8261, *Timing and synchronization aspects in packet networks, issued 04/2008*

ITU-T G.8262, *Timing characteristics of synchronous Ethernet equipment slave clock (EEC), issued 08/2007*

ITU-T G.8264, *Distribution of timing information through packet networks, issued 10/2008*

ITU-T G.8265.1, *Precision time protocol telecom profile for frequency synchronization, issued 10/2010*

ITU-T G.8275.1, *Precision time protocol telecom profile for phase/time synchronization with full timing support from the network, issued 07/2014*

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*

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 Appendix A.8, *RTP: A Transport Protocol for Real-Time Applications* (estimating the interarrival jitter)

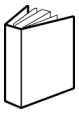
RFC 4585, *Extended RTP Profile for Real-time Transport Control Protocol (RTCP)-Based Feedback (RTP/AVPF)*

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)

Customer Document and Product Support



Customer Documentation

Customer Documentation Welcome Page

Technical Support

Product Support Portal

Documentation Feedback

Customer Documentation Feedback

3HE 15820 AAAB TQZZA 01