OpenDaylight Command-Line Interface (CLI):Main

From Daylight Project

Contents

- 3.17.12 Show External-ports Commands
  - 3.17.12.1 Show switch ports connected to external l2 networks
- 3.17.13 Show Feature Commands
  - 3.17.13.1 Show enabled and disabled features
- 3.17.14 Show Firewall-rule Commands
  - 3.17.14.1 Show firewall rules for controller interfaces
- 3.17.15 Show Flow-entry Commands
  - 3.17.15.1 Show configured static flow-entries
- 3.17.16 Show Global-config Commands
  - 3.17.16.1 Show high availability configuration
- 3.17.17 Show Logging Commands
  - 3.17.17.1 Show various controller logs
- 3.17.18 Show Running-config Commands
  - 3.17.18.1 Show the current active configuration
- 3.17.19 Show Switch-cluster Commands
  - 3.17.19.1 Show groups of interconnected openflow switches
- 3.17.20 Show Tech-support Commands
  - 3.17.20.1 Show tech-support, collect output of various commands
- 3.17.21 Show This Commands
  - 3.17.21.1 Show the object associated with the current submode
- 3.17.22 Show Host Commands
  - 3.17.22.1 Show host details based on query
  - 3.17.22.2 Show various host related details by query
- 3.17.23 Show Link Commands
  - 3.17.23.1 Show links, controller managed switch to switch interfaces
- 3.17.24 Show Snmp-server-config Commands
  - 3.17.24.1 Show snmp configuration
- 3.17.25 Show Switches Commands
  - 3.17.25.1 Show switch summary
  - 3.17.25.2 Show realtime stats for switch
  - 3.17.25.3 Show stats for selected switch
  - 3.17.25.4 Show statistics for a given switch
  - 3.17.25.5 Show interfaces for selected switch
  - 3.17.25.6 Show switch tcpdump via controller
  - 3.17.25.7 Show switch details via query
- 3.17.26 Show Switch-interfaces Commands
  - 3.17.26.1 Show interfaces for switch associated with current submode
- 3.17.27 Show Switch Commands
  - 3.17.27.1 Show tunnels for all switches
  - 3.17.27.2 Show tunnels for selected switches
- 3.17.28 Show Tacacs-plus-config Commands
  - 3.17.28.1 Show tacacs operational state
- 3.17.29 Show Tag Commands
  - 3.17.29.1 Show configured tags
- 3.17.30 Show Tech-support-config Commands
  - 3.17.30.1 Show tech-support configuration
- 3.17.31 Show Tenant Commands
  - 3.17.31.1 Show defined tenants
  - 3.17.31.2 Show specific tenant, identified by name
  - 3.17.31.3 Show specific tenant, identified by name
- 3.17.32 Show Dvs Commands
  - 3.17.32.1 Show vcenter dvs details
- 3.17.33 Show Dvs-port-group Commands
  - 3.17.33.1 Show vcenter dvs port-group details
- 3.17.34 Show Vcenter Commands

OpenDaylight Command Line Interface (CLI) User Guide

The OpenDaylight Command-Line Interface (CLI) is a management interface to the OpenDaylight Network Virtualization and OpenDaylight SDN Controller Platform. The CLI is packaged along with OSCP in the same source repository. Please see the OSCP User Guide for instructions on installation and configuration of OSCP.

CLI concepts

Access and modes

The CLI can be accessed through the console window of the virtual machine or by using ssh to connect to the virtual machine. Logging in as the admin user gives access to the CLI.

The admin user will be logged into the CLI in login mode. The prompt will have a > to indicate this mode.

Enter enable mode by typing the enable command. The prompt will have a # to indicate this mode.

Enter config mode by typing the configure command. The prompt will have a (config)# to indicate this mode.

There are a number of config submodes which will be indicated in the prompt as (config-<submode>)#, and each submode allows configuration of a specific type of object in the database.

Type exit to return to the previous mode, or type end to exit all config modes and return to enable mode.

Navigation and help

There are a number of ways users can navigate the CLI and receive help at any time.

For navigation, the CLI implements a Linux/shell-style navigation - for example:

- Ctrl-B - back one character
- Ctrl-F - forward one character
- Ctrl-A - move to the start of the line
- Ctrl-E - move to the end of the line
- Ctrl-P - display the previous command - can be repeated to go through history
- Ctrl-R - search for text among previous commands
  Refer to http://tiswww.case.edu/php/chet/readline/readline.html for more information on all keyboard shortcuts and facilities.

Help is also accessible by typing the command help. The output will vary based on the specific mode the user is in.

Command completion at any point can be pressing tab once or twice. Pressing tab once will complete the value if possible, and pressing tab twice will show all possible completions. Type the ? character to show completions at any time.

The CLI will accept shortened versions of commands and options so long as there is no ambiguity in the commands. For example, sh run will be interpreted as show running-config.

CLI conveniences: pipes, watching commands, and other tricks
The output of any CLI command can be piped to common Unix shell utilities such as grep, awk, wc, tail, more, or less. This can make searching for data or browsing through output significantly easier. For example, to find all the ports of a given switch, type:

```
10.0.2.15> show port | grep :24
 00:0a:00:24:a8:c4:69:00 52    28    00:24:a8:c4:69:cc
 00:0a:00:24:a8:c4:69:00 54    30    00:24:a8:c4:69:ca
 00:0a:00:24:a8:c4:69:00 56    32    00:24:a8:c4:69:c8
 00:0a:00:24:a8:c4:69:00 57    33    00:24:a8:c4:69:c7
 00:0a:00:24:a8:c4:69:00 58    34    00:24:a8:c4:69:c6
 10.0.2.15>
```

The CLI provides begin, include, and exclude as pipe options that may be familiar to administrators of existing network devices. begin is useful for starting to view large output at a particular point. For example, to see just the lsof output of show tech-support, type:

```
10.0.2.15> show tech-support | begin lsof |more
 Executing os command: sudo lsof
 COMMAND    PID       USER   FD      TYPE             DEVICE SIZE/OFF       NODE NAME
 init         1       root  cwd       DIR               8,16     4096          2 /
 init         1       root  rtd       DIR               8,16     4096          2 /
 init         1       root  txt       REG               8,16   125640       4081 /sbin/init
 init         1       root  mem       REG               8,16    51712      19738 /lib/libnss_files-2.12.1.so
 init         1       root  mem       REG               8,16    43552      26679 /lib/libnss_nis-2.12.1.so
 init         1       root  mem       REG               8,16    97256      19579 /lib/libnsl-2.12.1.so
```

```
 ...
```

The CLI also allows users to repeatedly invoke a command by prepending the command watch. For example, watch show switch <dpid> flow will allow the user to monitor the flows on a specific switch. This may be familiar to Unix users who use top or the corresponding watch command in Unix.

Multiple CLI commands can be entered into the CLI on a single line, separated by a semi-colon. For example, to get all the way into a configuring a flow entry on a specific switch, type:

```
10.0.2.15> enable; conf; switch 00:0a:00:24:a8:c4:69:00; flow-entry foo
```

```
 10.0.2.15(config-flow-entry)#
```

Finally, the CLI allows redirecting the output of a command to a local file or to the URL of an HTTP server or an FTP server. The local file is stored in the controller database (as general access to the local filesystem is not given). This is convenient for storing away show output or versions of configurations. For example, to store the output of show host, type:

```
10.0.2.15# show host > config://hostoutput
```

```
user data created
```

```
10.0.2.15# show config
```

```
Name       Length Version Timestamp
```

```
----------|------|-------|-------------------
```

```
hostoutput 1775   1       2010-12-15.08:29:31
```

```
10.0.2.15# show config hostoutput | head -10
```

```
MAC Address       Name Switch ID               Ingress Port IP Address     Vendor
```

```
-----------------|----|----------------------|------------|--------------|---------------------------
```

```
00:0c:29:6a:83:8c      00:0a:00:24:a8:c4:69:00 38           192.168.10.207 VMware, Inc.
```

```
00:21:f7:de:e9:00      00:0c:00:21:f7:de:e9:00 local(65534) 192.168.12.102 ProCurve Networking by HP
```

```
00:25:90:08:94:d5      00:0a:00:21:f7:de:e9:00 2            128.8.109.145  Super Micro Computer, Inc.
```

```
00:30:48:f9:cc:f7      00:00:00:30:48:f9:cc:f7 local(65534) 192.168.2.99   Supermicro Computer, Inc.
```

```
00:30:48:f9:cd:0b      00:00:00:30:48:f9:cd:0b local(65534) 0.0.0.0        Supermicro Computer, Inc.
```

```
00:30:48:f9:cd:0d      00:00:00:00:00:00:00:01 17           192.168.11.2   Supermicro Computer, Inc.
```

```
00:50:8d:65:be:4b      00:00:00:00:00:00:00:01 22           192.168.11.1   ABIT COMPUTER CORPORATION
```

```
10:00:00:67:ad:07      00:00:00:30:48:f9:cc:f7 4            192.168.12.132 unknown
```

Conventions

Text in courier font indicates either text that should be typed in by the user or output from a command.

Text enclosed in parentheses ( ) indicates a set of required arguments. The options are separated by a vertical bar |.

Text enclosed in square brackets [ ] indicates optional arguments. The options are separated by a vertical bar |.

Text enclosed in angle brackets < > indicates a value that should be entered/substituted by the user.

CLI Commands

Boot Command

Configure system boot options

Command Mode: enable mode

Command Syntax: `boot factory-default`

Command Description:
This command allows you to configure system boot parameters. At the moment, you can restore the controller to its factory default configuration using the 'factory-default' parameter.

Next Keyword Descriptions:

- factory-default:
  Reset the controller to the factory default configuration. This will wipe out all configuration and logs files and restore the controller to its initial default configuration.

This will require rebooting the controller node. When the controller finishes rebooting, you will need to go through the normal "first time setup" process, including reconfiguring the network interface configuration.

Note that if you have customized the controller base image through debug facilities, then some configuration may not be restored through this command. If this is the case, it is recommended that you start with a fresh controller image.

Command Examples:

```
Reset controller to its factory default settings.
```

```
    node1(config)# boot factory-default
```

```
    Re-setting controller to factory defaults ...
```

```
    Warning: This will reset your controller to factory-default state
```

```
             and reboot it.  You will lose all node/controller
```

```
             configuration and the logs
```

```
    Do you want to continue [no]? yes
```

```
    Resetting cassandra state ...
```

```
    Removing existing log files ...
```

```
    rsyslog stop/waiting
```

```
    Resetting system state ...
```

```

```

```
    Current default time zone: 'Etc/UTC'
```

```
    Local time is now:      Wed Oct 24 00:05:41 UTC 2012.
```

```
    Universal Time is now:  Wed Oct 24 00:05:41 UTC 2012.
```

```

```

```
    passwd: password expiry information changed.
```

```

```

Clearterm Command

Clears and resets the terminal screen

Command Mode: login mode

Command Syntax: `clearterm`

Configure Command

Enter configure mode

Command Mode: enable mode

Command Syntax: `configure [terminal]`

Connect Command

Connect to a controller's rest api

Command Mode: login mode

Command Syntax: `connect {<controller-id> | <ip-address>} [port <port>]`

Command Description:
The connect command directs the CLI to issue REST API requests to the selected controller. The CLI uses the controller's REST API to perform all configuration, retrieve all the data for show commands, and also to generate the running config. The CLI is an application which presents the controller's REST API to the user, and can also provide some limited aid to an application write trying to use the REST API, since the requests issued by the CLI can be viewed with the 'debug rest' command (use 'help debug rest' for more details)

When the controller is running as a HA slave, many commands will not be available. If the master has configured port 80 to be available (controller-node interface submode's firewall command), then by using the connect command to use the master's REST API, ssh can be avoided. This would preserve the command history, allowing an easy way to re-issue failed commands.

The REST API for the controller is typically configured at port 80.

Completion for the command will display all the currently known controllers. If the connect succeeds, the list of known controllers will very likely be different.

Next Keyword Descriptions:

- controller-id:
  The alias or UUID to identfy the controller
- port <port>:
  Identify the tcp port number of the REST API
- ip-address:
  An ip address to identify the controller's REST API

Command Examples:

```
connect localhost
```

```
    Connect to the current controller (really a no-op)
```

```
connect controller-node-b
```

```
    Connect to the controller identified by an alias
```

```
connect 192.168.2.129 port 8000
```

```
    Connect to the identified ip address
```

Copy Command

Copy configs to other configs

Command Mode: enable mode

Command Syntax: `copy {<source specifier>} [<destination specifier>]`

Command Description:
The copy command with one parameters copies the contents of the configuraion file out for display.

The copy command with two parameters copies the contents of the requested source into the dessination.

Next Keyword Descriptions:

- dest:
  Destination specifier for the copy

The keyword 'running-config' can be used to specify the controller's current state (that is, overwrite the running state).

The 'config://' URI scheme can be used to save config files to the local controller's disk.

The 'http://' and 'ftp://' URI schemes can be used to reference remote resources.

If the destination specifier is omitted, the resource referenced by the source specifier is sent to the Cli console.

- source:
  Source specifier for the copy

The keyword 'running-config' can be used to specify the controller's current state.

The 'config://' URI scheme can be used to reference saved config files from a previous 'copy' command.

The 'http://' and 'ftp://' URI schemes can be used to reference remote resources.

Command Examples:

```
copy running-config
```

```
   Variation of the 'show running-config' command
```

```
 copy running-config config://xyz
```

```
  Copy the current running config into the configuration
```

```
  target named config://xyz
```

Enable Command

Enter enable mode

Command Mode: login mode

Command Syntax: `enable`

Ha Command

Perform actions related to high availability

Command Mode: enable mode

Command Syntax: `ha {failover | provision <ip> | decommission <id>}`

Command Description:
Configures failover parameters, or triggers a failover event.

The 'failover' token directs the current node to fail over to a configured slave controller.

The 'provision' token configures a slave controller by IP address that will function as a master controller after a failover event.

Next Keyword Descriptions:

- id:
  Specify the name of one of the nodes in the controller cluster.

Command Examples:

```
ha failover
```

```
  Fail this controller node, defer to a slave controller
```

```
ha provision 1.2.3.4
```

```
  Configure a new controller node to function as a slave
```

Ping Command

Ping a switch or ip address

Command Mode: login mode

Command Syntax: `ping [count <count>] <ip-address>`

Command Description:
The ping command sends ICMP echo requests validates basic network connectivity between the CLI and the requested target.

Next Keyword Descriptions:

- count <count>:
  This integer field identifies the number of ping requests sent to the target before exiting. The default value is '5'.
  ip-address: type Resolvable-ip-address
  A dotted-quat ip address (192.168.2.129), or a domain name (localhost), which can be resolved may be included here. Additionally, switch alias or dpids may also be included if they're currently active (an ip address is associated with the switch)

Command Examples:

```
ping 1.2.3.4
```

```
  Ping a specific host with a default package count (5)
```

```
ping 10 1.2.3.4
```

```
  Ping a specific host with a fixed number of packets
```

```

```

Reload Command

Reload and reboot

Command Mode: enable mode

Command Syntax: `reload`

Command Description:
This command will restart this controller node. If you are running as part of a high availability cluster, this will trigger a failover to the remaining nodes, but you may wish to manually trigger this first using the "ha failover" command.

Rebooting the controller will allow you to access the boot menu if you wish to revert to an older version of the controller software following an upgrade. If you wish to do this, select the appropriate image from the boot loader prompt from the controller console to choose a different image version.

Command Examples:

```
Reboot the controller
```

```
    node1(config)# reload
```

```
    Confirm Reload (yes to continue) yes
```

```

```

Rollback Command

Rollback cluster to specified config

Command Mode: enable mode

Command Syntax: `rollback {images:// | saved-configs://} <file>`

Set Command

Manage cli sessions settings

Command Mode: login mode

Command Syntax: `set length {<length> | term}`

Command Description:
Set the terminal height for paging Cli command output.

Command Examples:

```
set length 50
```

```
  Set the terminal length to 50 lines.
```

```
set length term
```

```
  Set the terminal length to the natural screen height.
```

Traceroute Command

Determine the l3 path to some destination

Command Mode: login mode

Command Syntax: `traceroute <ip-address>`

Command Description:
Compute the L3 path between the local host and the destination by using increasing TTL's, and reporting back ICMP timed-out messages

Next Keyword Descriptions:
ip-address: type Resolvable-ip-address
A dotted-quat ip address (192.168.2.129), or a domain name (localhost, bigswtich.com), which can be resolved may be included here. Additionally, switch alias or dpids may also be included if they're currently active (an ip address is associated with the switch)

Command Examples:

```
traceroute 1.2.3.4
```

```
  Compute the L3 path the host with IP address 1.2.3.4
```

```
traceroute www.yahoo.com
```

```
  Compute the L3 path the host with a specific hostname
```

Upgrade Command

Manage the controller upgrade process

Command Mode: enable mode

Command Syntax: `upgrade abort`

Next Keyword Descriptions:

- abort:
  If you've already run an upgrade operation on this controller node, but want to abort the upgrade before you reboot into the new partition, run the "upgrade abort" command to configure the controller to boot by default from the current-active partition and set the controller status back to "Ready."

This can allow you to restart a failed upgrade process, for example if a node failure occurs while attempting to upgrade.

Command Examples:

```
upgrade abort
```

```
  Abort an in-progress or requested upgrade
```

```

```

Manage the controller upgrade process

Command Mode: enable mode

Command Syntax: `upgrade [force] [details]`

Command Description:
Upgrade the controller from an uploaded controller image. To perform upgrade, you will first need to upload an upgrade image package by scping the file using the "images" user.

Upgrade image package is a file with name of format "controller-upgrade-YYYY.MM.DD.XXXX.pkg". Following is an example to prepare upgrade for controller with IP address 192.168.67.141:"

scp $path/controller-upgrade-2013.02.13.0921.pkg images@192.168.67.141:"

After you run the upgrade, the new controller image will be installed on the second image partition, and will be configured as the default boot image. Running the "reload" command will boot the current node If you are upgrading a cluster with multiple nodes, begin by upgrading the slave nodes followed by the master node. You can find out which slave is the master by running "show controller-node all". You should upgrade each node by running the upgrade, then reloading the controller node.

For example, if you have two nodes in your cluster, node1 (the current master) and node2 (the current slave), you should:

1. Upgrade node2 using the "upgrade" command
2. Reboot node2 using the "reload" command
3. Upgrade node1 using the "upgrade" command
4. Reboot node1 using the "reload" command

Note that when you reboot the master controller node1 there may be a brief disruption in your network.

Please refer to the section on upgrading in the user guide for more detailed information, including information on how to revert if the upgrade fails.

Next Keyword Descriptions:

- force:
  The "force" option will run the upgrade but will ignore any validation errors. These errors include validating the package checksum, minimum system requirements, and connectivity. Note that if you choose this option, there is an increased chance that your controller upgrade will be unsuccessful.

- details:
  The "details" option will cause verbose information to be printed out during each upgrade step. This may be helpful in diagnosing failures during the upgrade process.

Command Examples:

```
Upgrade the controller node using an uploaded upgrade package
```

```
    node1> enable
```

```
    node1# upgrade
```

```
    Upgrade controller from image '/home/images/controller-upgrade.pkg'?
```

```
    (yes to continue) yes
```

```
    Executing upgrade...
```

```
    1 - Verifying package checksum
```

```
      Succeeded
```

```
    2 - Verifying connectivity to other nodes via ping
```

```
      Succeeded
```

```
    3 - Checking minimum system requirements
```

```
      Succeeded
```

```
    4 - Copying configuration
```

```
      Succeeded
```

```
    5 - Creating new filesystem
```

```
      Succeeded
```

```
    Controller node upgrade complete.
```

```
    Upgrade will not take effect until system is rebooted. Use 'reload' to
```

```
    reboot this controller node. To revert, select the appropriate image
```

```
    from the boot menu
```

```

```

Write Command

Write config to memory or terminal, or clear

Command Mode: enable mode

Command Syntax: `write terminal`

Command Description:
Write command allows you to view the current configuration or restore the system to its default configuration.

Next Keyword Descriptions:

- terminal:
  Display the current active configuration on the terminal. Equivalent to "show running-config".

Command Examples:

```
Display the current running-config:
```

```
    node1# write terminal
```

```
    !
```

```
    ! OS 1.0 - custom version
```

```
    ! Current Time: 2012-10-23.20:37:51
```

```
    !
```

```
    ...
```

```
Reset to factory defaults:
```

```
node1# write erase
```

```
Re-setting controller to factory defaults ...
```

```
...
```

Command Syntax: `write erase`

Next Keyword Descriptions:

- erase:
  Reset the controller to the factory default configuration. This will wipe out all configuration and logs files and restore the controller to its initial default configuration.

This will require rebooting the controller node. When the controller finishes rebooting, you will need to go through the normal "first time setup" process, including reconfiguring the network interface configuration.

Note that if you have customized the controller base image through debug facilities, then some configuration may not be restored through this command. If this is the case, it is recommended that you start with a fresh controller image.

Test Command

Perform various tests on the network

Command Mode: login mode

Command Syntax: `test packet-in src-host <src-host> dst-host <dst-host> [src-switch <src-switch> <src-switch-port>] [vlan <vlan>] [priority <priority>] [src-ip-address <src-ip-address>] [dst-ip-address <dst-ip-address>] [protocol <protocol>] [tos <tos>] [src-port <src-port>] [dst-port <dst-port>]`

Command Description:
The test command provides various tools to help perform root-cause analysis.

Next Keyword Descriptions:

- src-host <src-host>:
  This parameter identifies a host, used to identify an attachment point. The parameter's value is a mac address.
- vlan <vlan>:
  VLAN ID for tagging packets
- src-switch-port:
  This parameter identifies a physical switch port number, as part of the description of the attachment point
- dst-host <dst-host>:
- src-port <src-port>:
  Port number for injected packets

- src-ip-address <src-ip-address>:
  IP address for injected packets
- dst-ip-address <dst-ip-address>:
- src-switch <src-switch>:
  This parameter identifies a switch by DPID, as part of the description of the attachment point
- priority <priority>:
  Ether priority for injected packets
- protocol <protocol>:
  Protocol number for injected packets
- tos <tos>:
  TOS flags for injected packets
- dst-port <dst-port>:
- packet-in:
  The 'packet-in' test type provides a tools to determine whether a source and dest can transmit a frame. A frame is injected as if it originated from the indicated port on a source switch, and is directed to a destination switch and port. When it arrives, the command announces the path traversed.

Command Examples:

```
test packet-in src-host 00:00:00:00:00:01 dst-host 00:00:00:00:00:02
```

```
  Test packet injection between two hosts
```

```
test packet-in src-host 00:00:00:00:00:01 dst-host 00:00:00:00:00:02 src-switch 00:00:00:00:00:00:00:00 32
```

```
  Run the packet-in test, specifying a specific switch physical port
```

```
test packet-in src-host 00:00:00:00:00:01 dst-host 00:00:00:00:00:02 ... vlan 1001
```

```
  Run the packet-in test, tagging packets with a specific VLAN
```

```
test packet-in src-host 00:00:00:00:00:01 dst-host 00:00:00:00:00:02 ... priority 2
```

```
  Run the packet-in test, tagging packets with a specific ether priority
```

```
test packet-in src-host 00:00:00:00:00:01 dst-host 00:00:00:00:00:02 ... src-ip-address 1.2.3.4
```

```
  Run the packet-in test, tagging packets with a specific source IP address
```

```
test packet-in src-host 00:00:00:00:00:01 dst-host 00:00:00:00:00:02 ... dst-ip-address 1.2.3.4
```

```
  Run the packet-in test, tagging packets with a specific destination IP address
```

```
test packet-in src-host 00:00:00:00:00:01 dst-host 00:00:00:00:00:02 ... protocol 6
```

```
  Run the packet-in test, tagging packets with a specific IP protocol number (TCP)
```

```
test packet-in src-host 00:00:00:00:00:01 dst-host 00:00:00:00:00:02 ... tos 7
```

```
  Run the packet-in test, tagging packets with a specific set of TOS bits
```

```
test packet-in src-host 00:00:00:00:00:01 dst-host 00:00:00:00:00:02 ... src-port 80
```

```
  Run the packet-in test, tagging packets with a specific source port number
```

```
test packet-in src-host 00:00:00:00:00:01 dst-host 00:00:00:00:00:02 ... dst-port 443
```

```
  Run the packet-in test, tagging packets with a specific destination port number
```

```
```

Perform various tests on the network

Command Mode: login mode

Command Syntax: `test path {src-host <src-host> | src-ip <src-ip> | src-switch <src-switch> <src-switch-port>} {dst-host <dst-host> | dst-ip <dst-ip> | dst-switch <dst-switch> <dst-switch-port>}`

Command Description:
The test command provides various tools to help perform root-cause analysis.

Next Keyword Descriptions:

- src-host <src-host>:
  This parameter identifies the host, used to identify an attachment point. The parameter's value is a mac address.
- src-switch-port:
  This parameter identifies a specific switch's interface (physical port), to complete the attachment point description.
- dst-host <dst-host>:
- src-ip <src-ip>:
  This parameter identies an ip address used to determine an attachment point
- dst-switch-port:
- src-switch <src-switch>:
  This parameter identifies a switch via DPID, as part of the description of the attachment point
- dst-ip <dst-ip>:
- path:
  The 'test path' command requests the controller to compute the path between interfaces on switches, return, and display the result. No attempt it made to validate the connectivity between the two endpoints.

To compute the path, and source and a destination needs to be provided. These can be described in a variety of different forms: hosts mac addresses, ip addresses, or switch and interface names.

- dst-switch <dst-switch>:

Command Examples:

```
node1> test path src-ip 10.0.0.1 dst-ip 10.0.0.3
```

```
 # Switch                   IF      Rx Bytes Rx Pkts Rx Errs Tx Bytes Tx Pkts Tx Errs
```

```
 -|----------------------|-------|--------|-------|-------|--------|-------|-------
```

```
 1 00:00:00:00:00:00:00:06 s6-eth1 860      11      0       4762     74      0
```

```
 2 00:00:00:00:00:00:00:06 s6-eth3 2208     35      0       2208     35      0
```

```
 3 00:00:00:00:00:00:00:05 s5-eth1 2208     35      0       2208     35      0
```

```
 4 00:00:00:00:00:00:00:05 s5-eth2 2208     35      0       2391     38      0
```

```
 5 00:00:00:00:00:00:00:07 s7-eth3 2419     38      0       2208     35      0
```

```
 6 00:00:00:00:00:00:00:07 s7-eth1 860      11      0       4684     73      0
```

```

```

Command Syntax: `tunnel-link {verify <switch dpid or alias> <switch dpid or alias>}`

Configuration Commands

Address-space Commands

Enter address space submode

Command Mode: config mode

Command Syntax: `[no] address-space <name>`

Command Description:
Set up or tear down address space definitions.

Specifying an address space by name enters the address-space configuration sub-mode.

Next Keyword Descriptions:

- name:
  This is the name of the address-space. The address-space name "default" represents the default address space that will be used if no other address-space matches.

Command Examples:

```
addres-space MY-ADDRESS
```

```
  Define a new address space, and enter its configuration submode
```

```
no address-space MY-ADDRESS
```

```
  Delete an address space by name
```

```

```

Set address-space active

Command Mode: config-address-space mode

Command Syntax: `[no] active`

Command Description:
Set the address-space active. If an address-space is inactive the controller will not use it or its rules.

Command Examples:

```
active
```

```
  Within an address-space config sub-mode,
```

```
  set this address-space as active
```

```
no active
```

```
  Deactivate the currently-configured address space
```

```
```

Provide description for this address-space

Command Mode: config-address-space mode

Command Syntax: `[no] description <description>`

Command Description:
A user provided textual description for this address-space.

Command Examples:

```
description "THIS IS MY ADDRESS-SPACE"
```

```
  Associate a textual description with this address-space
```

```
```

Set an address-space identifier rule

Command Mode: config-address-space mode

Command Syntax: `[no] identifier-rule <rule>`

Command Description:
Set an address-space identifier rule and enter submode. Devices are assigned into address-spaces based on identifier-rules.

Next Keyword Descriptions:

- rule:

Command Examples:

```
identifier-rule rule42
```

```
  Define a new identifier rule for this address-space definition,
```

```
  and enter the identifier-rule submode
```

```
no identifier-rule rule42
```

```
  Delete an identifier rule associated with the
```

```
  currently-define address-space
```

```
```

Set rule to active

Command Mode: config-address-space-id-rule mode

Command Syntax: `[no] active`

Command Description:
Configure this address space identifier rule as 'active'

Command Examples:

```
active
```

```
  Configure this address space identifier rule as active
```

```
no active
```

```
  Deactivate this address space identifier rule
```

```
```

Provide description for identifier rule

Command Mode: config-address-space-id-rule mode

Command Syntax: `[no] description <description>`

Command Description:
A user provided textual description for this identifier-rule.

Command Examples:

```
description "This is rule42"
```

```
  Provide a description for the currently-defined address-space identifier rule
```

```

```

Associate switch with identifier rule

Command Mode: config-address-space-id-rule mode

Command Syntax: `match switch {<switch dpid or switch alias> [<switch interface, or range, or list>]}`

Next Keyword Descriptions:

- switch:
  Associate a switch or set of switch interfaces with this identifier rule. Devices present on the specified switch/interfaces will be matched.
- ports:
  Restrict the match to a switch interface or list/range of switch interfaces

Command Examples:

```
match switch 00:00:11:22:33:44:55:66
```

```
    Matches all interfaces on the switch with this DPID.
```

```
match switch 00:00:11:22:33:44:55:66 Ethernet1
```

```
    Matches interface Ethernet1 on the specified switch.
```

```
match switch 00:00:11:22:33:44:55:66 Ethernet1,Ethernet5-10,port2
```

```
    Matches interfaces Ethernet1, Ethernet5, Ethernet6, ... Ethernet10,
```

```
       and port2 on the specified switch.
```

```
match switch ToR-1-1
```

```
    Matches the switch with the alias 'ToR-1-1'
```

```
```

Associate tag with identifier rule

Command Mode: config-address-space-id-rule mode

Command Syntax: `match tags <tag>`

Next Keyword Descriptions:

- tags:
  Associate a tag or list of tags with this identifier-rule. If a list of tags is given **all** tags need to match.

Command Examples:

```
match tags com.bs.tenant=CustomerA
```

```
    Matches devices that match the given tag.
```

```
match tags com.bs.tenant=CustomerA,com.example.type=router
```

```
    Matches devices that match *all* of the given tags.
```

```
```

Associate vlans with identifier rule

Command Mode: config-address-space-id-rule mode

Command Syntax: `match vlans <Vlan number (1-4095) or range, or list>`

Next Keyword Descriptions:

- vlans:
  Associate VLANs with identifier-rule. In this software version the specified VLAN must be equal to value specified for 'vlan-tag-on-egress'

Command Examples:

```
match vlans 1001
```

```
  Associate one or more VLAN IDs with this identifer rule
```

```
no match vlans 1001
```

```
  Remove a VLAN ID association from this identifier rule
```

```
no match vlans
```

```
  Remove all VLAN ID associations from this identifier rule
```

```
```

Describe priority for identifier rule

Command Mode: config-address-space-id-rule mode

Command Syntax: `[no] priority <priority>`

Command Description:
The prirority of this identifier-rule. Higher numeric values represent higher priority. The highest priority identifier-rule that matches a given packet will be choosen.

Command Examples:

```
priority 100
```

```
  Set the priority for the currently-defined address-space identifier rule
```

```
```

Describe address-space origin

Command Mode: config-address-space mode

Command Syntax: `[no] origin <origin>`

Command Description:
Describe the origin of an address space

Command Examples:

```
origin rest
```

```
  Note that this address-space was configured via REST
```

```
no origin rest
```

```
  Remove the origin notation for this address-space
```

```

```

Set address-space priority

Command Mode: config-address-space mode

Command Syntax: `[no] priority <priority>`

Command Description:
The prirority of this address-space. Higher numeric values represent higher priority. All identifier-rules of the highest priority address-space will be evaluated before any rules of other address-spaces.

Command Examples:

```
priority 100
```

```
  Set the priority of this currently-defined address-space
```

```
no priority 100
```

```
  Remove the previous priority declaration,
```

```
  and reset this address-space's priority to the default value
```

```

```

Egress vlan tag

Command Mode: config-address-space mode

Command Syntax: `[no] vlan-tag-on-egress <vlan-tag-on-egress>`

Command Description:
Associates a VLAN with this address-space. This VLAN is used for internal disambiguation and for tagging packets that egress to other networks (according to identifier-rule configuration).

Command Examples:

```
vlan-tag-on-egress 42
```

```
  Associate a VLAN tag with this address-space.
```

```
  Each address-space must have an associated VLAN, else this address-space
```

```
  will be ignored by the controller
```

```

```

Onv Commands

Enter onv submode, manage access lists

Command Mode: config mode

Command Syntax: `onv <onvname>`

Command Description:
This command is used to enter a submode to manage properties associated with the virtual switch. This currently includes acl managment, and association of the acls rules to interfaces.

The named onv-id must already exist. See the onv-definition command to create new onv's.

Withing this submode, two other submodes can be entered. The access-list submode associates specific acl rules with an access list, while the interface submode allows association of named access rules to specific interfaces.

Next Keyword Descriptions:

- onvname:
  This is the name of the ONV. The ONV name "default" represents the default ONV in the default address space. A ONV name conforming to "<address-space-name>-default" represents the default ONV for the address-space "address-space-name"

Command Examples:

```
onv  my-onv-definition
```

```
  Enter ONV definition sub-mode.
```

```
  The ONV name corresponds to a prior onv-definition identifier.
```

```

```

Associate interface with access-list

Command Mode: config-tenant-onv-if mode

Command Syntax: `[no] access-group <onv-access-list> {in | out}`

Command Description:
Associate an access-list configuration with this interface rule.

Next Keyword Descriptions:

- out:
  Apply an access-list to outgoing traffic on this ONV interface.
- in:
  Apply an access-list to incoming traffic on this ONV interface.

Command Examples:

```
access-group pair-blocker in
```

```
  Associate the 'pair-blocker' ACL with input packets to this ONV
```

```
no access-group pair-blocker in
```

```
  Remove the input packet ACL for this ONV instance
```

```

```

Enter onv access-list submode

Command Mode: config-tenant-onv mode

Command Syntax: `[no] access-list <name>`

Command Description:
Enter submode to configure ONV access-list.

Next Keyword Descriptions:

- name:
  The name of the access list

Command Examples:

```
access-list access-list-1
```

```
  Define a new access list for this ONV,
```

```
  and enter its configuration sub-mode
```

```
no access-list access-list-1
```

```
  Remove the definition for this access list
```

```

```

Provide a description for a onv access list

Command Mode: config-tenant-onv-acl mode

Command Syntax: `[no] description <description>`

Command Description:
A user provided textual description for this access-list.

Command Examples:

```
description "Access list #1"
```

```
  Associate a text description with this access list
```

Define acl details for this access-list

Command Mode: config-tenant-onv-acl mode

Command Syntax: `<acl rule number> {permit | deny} {{{ip | tcp | udp} | <ip protocol>} {<src-ip> <src-ip-mask> | <src-ip> | <src-cidr> | any} [{eq | neq} {<src-tp-port> | {http | dns | https | ssh}] [{<dst-ip> <dst-ip-mask> | <dst-ip> | <dst-cidr> | any} [{eq | neq} {<dst-tp-port> | {http | dns | https | ssh}}]] | icmp {<src-ip> <src-ip-mask> | <src-ip> | <src-cidr> | any} [{eq | neq} {<src-tp-port> | {http | dns | https | ssh}}] [{<dst-ip> <dst-ip-mask> | <dst-ip> | <dst-cidr> | any} [{eq | neq} {<dst-tp-port> | {http | dns | https | ssh}}]] [<icmp-type>] | mac {any | <src-mac>} {any | <dst-mac>} [<ether-type> | {arp | lldp | 802.1Q | ip | mpls | rarp | mpls-mc | appletalk-aarp | ipv6 | novell | ipx}] [vlan <vlan>]}}`

Command Description:
Add an entry to this ONV access list

Next Keyword Descriptions:

- ip:
  Access list entry for IP packets.
- src-ip-mask:
  An inverse netmask in dotted decimal notation.
- tcp:
  Access list entry for TCP packets.
- eq:
  Port number equals.

- any:
  Represents any IP address.
- https:
  Specify a port by service name
- ether-type:
  Specify an ether type by number (hex or decimal)
- type:

- dst-ip-mask:
- neq:
  Port number does not equal.

- udp:
  Access list entry for UDP packets.
- http:
- dns:
- dst-ip:
  IP address in dotted decimal notation.
  IP address in dotted decimal notation.
  IP address with prefix length in CIDR format.
- mac:
  Filter based on source/destionation MAC addresses, ether types and VLAN IDs
- ssh:o:p>
- icmp:

- deny:
  Deny traffic matching this entry.
- src-tp-port:
  Port number in decimal or hex if prefixed with 0x.
- src-ip:
- dst-tp-port:
- permit:
  Permit traffic matching this entry.

Command Examples:

```
10 deny ip 10.0.0.1 10.0.0.2
```

```
  Add an ACL rule to deny IP traffic between two hosts
```

```
11 allow tcp 10.0.0.3
```

```
  Add an ACL rule to allow TCP traffic from a specific host
```

```
12 deny 51
```

```
  Add an ACL rule to deny AH packets
```

```
13 deny tcp any eq http
```

```
  Add an ACL rule to deny HTTP traffic
```

```
14 deny tcp any eq http
```

```
  Add an ACL rule to deny HTTP traffic
```

```
15 deny mac 00:00:00:00:00:01 any
```

```
  Add an ACL rule to deny traffic based on source MAC address
```

```
16 deny mac 00:00:00:00:00:01 00:00:00:00:00:02
```

```
  Add an ACL rule to deny traffic between MAC addresses
```

```
16 deny mac any any 0x0842
```

```
  Add an ACL rule to deny wake-on-LAN packets
```

```
17 deny mac any any vlan 42
```

```
  Add an ACL rule to deny packets from a specific VLAN
```

```
```

Set onv priority

Command Mode: config-tenant-onv-acl mode

Command Syntax: `[no] priority <priority>`

Command Description:
Within a ONV access list definition, set the priority

Command Examples:

```
priority 100
```

```
  Set the priority of this ACL to 100
```

```
```

Enter onv-if submode

Command Mode: config-tenant-onv mode

Command Syntax: `[no] interface <interface>`

Command Description:
Enter ONV interface submode, manage association to access lists.

Next Keyword Descriptions:

- interface:
  The name of the ONV interface. Interface names are derived from interface-rules.

Command Examples:

```
interface main-interface
```

```
  Enter the ONV interface configuration sub-mode.
```

```
  The interface is named based on a corresponding interface-rule specifier.
```

```
```

ONV-definition Commands

Enter onv definition submode

Command Mode: config mode

Command Syntax: `[no] onv-definition <onvname>`

Command Description:
This submode is used to create, then describe the membership of devices for the named ONV. Within this submode, properties of the ONV can be configured. Interface-rules are configured and managed, which configure the membership of devices.

The controller provides a ONV named 'default' to collect devices which are not associated with any ONV.

Part of the configuration associated with a ONV includes the association of a particular address-space. When not speficially configured, the 'default' address-space is associated with the ONV.

Each address-space also has an associated default ONV. IF the address space is called 'yellow', then the default ONV for this address space is called 'yellow-default'

Next Keyword Descriptions:

- onvname:
  This is the name of the ONV. The ONV name "default" represents the default ONV in the default address space. A ONV name conforming to "<address-space-name>-default" represents the default ONV for the address-space "address-space-name"

Command Examples:

```
onv-definition my-first-onv
```

```
  Define a new ONV instance by name,
```

```
  and enter the ONV definition sub-mode
```

```
no onv-definition my-first-onv
```

```
  Remove a named ONV
```

```
```

Set onv active

Command Mode: config-tenant-def-onv mode

Command Syntax: `[no] active`

Command Description:
Set the ONV active. If a ONV is inactive the controller will not use it or its rules.

Command Examples:

```
active
```

```
  Activate this ONV instance
```

```
no active
```

```
  Deactivate this ONV instance
```

```
```

Configure arp mode

Command Mode: config-tenant-def-onv mode

Command Syntax: `[no] arp-mode {always-flood | flood-if-unknown | drop-if-unknown}`

Command Description:
Configure the ARP behavior for this ONV instance.

Next Keyword Descriptions:

- always-flood:
  Always flood ARP packets on all switch interfaces. No active managment of ARP, will leak packets across ONV.
- drop-if-unknown:
  Drop ARP packets if the host is unknown
- flood-if-unknown:
  Flood ARP packets if the destination is unknown. Might leak packets across ONV.

Command Examples:

```
arp-mode always-flood
```

```
  Do not manage ARP packets, just flood them on all interfaces
```

```
arp-mode flood-if-unknown
```

```
  Only flood ARP packets for unknown destinations
```

```
arp-mode drop-if-unknown
```

```
  Drop ARP packets from unknown destinations
```

```

```

Configure broadcast mode

Command Mode: config-tenant-def-onv mode

Command Syntax: `[no] broadcast {always-flood | forward-to-known | drop}`

Command Description:
Configure broadcast characteristics of the ONV instance.

Next Keyword Descriptions:

- always-flood:
  Always flood all non-ARP, non-DHCP broadcast packets on all switch interfaces. Will leak packets across ONV.

- forward-to-known:
  Forward all non-ARP, non-DHCP broadcast packets to all known hosts in this ONV.
- drop:
  Drop all non-ARP, non-DHCP broadcast packets.

Command Examples:

```
broadcast always-flood
```

```
  Miscellaneious broadcast packets are sent on all interfaces
```

```
broadcast forward-to-known
```

```
  Miscellaneous broadcast packets are forwarded only to known hosts
```

```
broadcast drop
```

```
  Miscellaneous broadcast packets are dropped
```

```

```

```

```

Provide description for a onv instance

Command Mode: config-tenant-def-onv mode

Command Syntax: `[no] description <description>`

Command Description:
Within a ONV definition sub-mode, declare a friendly descriptor for the ONV instance.

Command Examples:

```
description "this is my first ONV instance"
```

```
  Add a textual description to a ONV
```

```

```

Configure dhcp ip address

Command Mode: config-tenant-def-onv mode

Command Syntax: `[no] dhcp-ip <dhcp-ip>`

Command Description:
Specify the IP address of the local DHCP server or local DHCP relay. Used by dhcp-mode 'static'.

Command Examples:

```
dhcp-ip 1.2.3.4
```

```
  In static DHCP mode, configure the local DHCP server or relay address
```

Set dhcp mode

Command Mode: config-tenant-def-onv mode

Command Syntax: `[no] dhcp-mode {always-flood | flood-if-unknown | static}`

Command Description:
Configure ONV handling of DHCP broadcast packets.

Next Keyword Descriptions:

- always-flood:
  Always flood all DHCP packets. Will leak packets across ONV.
- static:
  Forward DHCP packets to the configured local DHCP server or local DHCP relay.
- flood-if-unknown:
  Flood DHCP packets if the DHCP server location is unknown. DHCP server location will be discovered.

Command Examples:

```
dhcp-mode static
```

```
  ONV will forward DHCP traffic to a single known host
```

```
dhcp-mode flood-if-unknown
```

```
  Flood DHCP packets if the server is not known
```

```
dhcp-mode always-flood
```

```
  Do not manage DHCP traffice
```

```

```

Set rule to active

Command Mode: config-tenant-def-onv-if-rule mode

Command Syntax: `[no] active`

Command Description:
Mark the rule as active. Only active rules will be matched.

Command Examples:

```
active
```

```
  Mark this interface rule as active
```

```
no active
```

```
  Mark this interface rule as inactive
```

```

```

Enable multiple interface rule matches

Command Mode: config-tenant-def-onv-if-rule mode

Command Syntax: `[no] allow-multiple`

Command Description:
If allow-multiple is set devices matching this rule are allowed to be in multiple ONV at the same time.

Command Examples:

```
allow-multiple
```

```
  Devices matching this rule can be in more than one ONV
```

```
no allow-multiple
```

```
  Devices matching this rule can be in a single ONV
```

```

```

```

```

Provide description for interface rule

Command Mode: config-tenant-def-onv-if-rule mode

Command Syntax: `[no] description <description>`

Command Description:
A user provided textual description for this interface-rule.

Command Examples:

```
description "This is my first ONV interface rule"
```

```
  Describe a ONV interface rule
```

```

```

Associate ip-subnet (ip or cidr range) for interface rule

Command Mode: config-tenant-def-onv-if-rule mode

Command Syntax: `match ip-subnet <ip address (10.10.10.10), or cidr (10.20.30.0/24)>`

Command Description:
Associate an IP address or IP subnet with interface-rule.

Next Keyword Descriptions:

- ip-subnet:

Command Examples:

```
match ip-subnet 10.10.10.10
```

```
 match ip-subnet 10.42.10.0/24
```

```

```

Associate mac (host) with interface rule

Command Mode: config-tenant-def-onv-if-rule mode

Command Syntax: `match mac <host mac or alias>`

Command Description:
Associate MAC address (host) with interface-rule.

Next Keyword Descriptions:

- mac:

Command Examples:

```
match mac 00:00:00:00:00:01
```

```
   Match a specific MAC address
```

```

```

Associate switch with interface rule

Command Mode: config-tenant-def-onv-if-rule mode

Command Syntax: `match switch {<switch dpid or switch alias> [<switch interface, or range, or list>]}`

Command Description:
Associate a switch or set of switch interfaces with this interface-rule. Hosts present on the specified switch/interfaces will be matched.

Command Examples:

```
match switch 00:00:11:22:33:44:55:66
```

```
    Matches all interfaces on the switch with this DPID.
```

```
match switch 00:00:11:22:33:44:55:66 Ethernet1
```

```
    Matches interface Ethernet1 on the specified switch.
```

```
match switch 00:00:11:22:33:44:55:66 Ethernet1,Ethernet5-10,port2
```

```
    Matches interfaces Ethernet1, Ethernet5, Ethernet6, ... Ethernet10,

       and port2 on the specified switch.
```

```
match switch ToR-1-1
```

```
    Matches the switch with the alias 'ToR-1-1'
```

```

```

Associate tags with interface rule

Command Mode: config-tenant-def-onv-if-rule mode

Command Syntax: `match tags <tags>`

Command Description:
Associate a tag or list of tags with this interface-rule. If a list of tags is given **all** tags need to match.

Command Examples:

```
match tags com.bs.tenant=CustomerA
```

```
    Matches devices that match the given tag.
```

```
match tags com.bs.tenant=CustomerA,com.example.type=router
```

```
    Matches devices that match *all* of the given tags.
```

```

```

Associate vlans with interface rule

Command Mode: config-tenant-def-onv-if-rule mode

Command Syntax: `match vlans <Vlan number (0-4096) or range, or list>`

Command Description:
Associate a vlan (or list or range) with this interface rule.

Command Examples:

```
match vlans 42
```

```
  Match packets in VLAN 42
```

Describe priority for interface rule

Command Mode: config-tenant-def-onv-if-rule mode

Command Syntax: `[no] priority <priority>`

Command Description:
The priority of this interface-rule. Higher numeric values represent higher priority. The highest priority interface-rule that matches a given packet will be choosen.

Command Examples:

```
priority 100
```

```
  Assign a priority to this interface rule
```

Describe onv origin

Command Mode: config-tenant-def-onv mode

Command Syntax: `[no] origin <origin>`

Command Description:
Describe the origin of this ONV instance

Command Examples:

```
origin rest
```

```
  Note that this ONV instance was configured via REST
```

```

```

Associate address space

Command Mode: config-tenant-def-onv mode

Command Syntax: `[no] use address-space <address-space>`

Command Description:
Associate this ONV with the specified address-space.

Command Examples:

```
use address-space address-space-1
```

```
  Change the association of this ONV from 'default' to 'address-space-1'.
```

```
no address-space address-space-1
```

```
  Remove the address space association for this ONV.
```

```

```

====Enter interface-rule submode, configure onv

```
details====
```

Command Mode: config-tenant-def-onv mode

Command Syntax: `[no] interface-rule <rule>`

Command Description:
Set a ONV interface rule and enter submode. Devices are assigned into ONV based on interface-rules.

Command Examples:

```
interface-rule my-rule-1
```

```
  Define a new interface rule, and enter its configuration sub-mode
```

```
no interface-rule my-rule-1
```

```
  Delete an interface rule by name
```

```

```

Set onv priority

Command Mode: config-tenant-def-onv mode

Command Syntax: `[no] priority <priority>`

Command Description:
Assign a priority to this ONV instance

Command Examples:

```
priority 100
```

```
  Within a onv definition, assign a specific priority to this instance
```

```

```

Controller-node Commands

Enter configuration submode for controller-nodes

Command Mode: config mode

Command Syntax: `controller-node <id>`

Command Description:
Enter a submode to configure the settings for the controller node.

Next Keyword Descriptions:

- id:
  Specify the name of one of the nodes in the controller cluster.

Command Examples:

```
controller-node localhost
```

```
  Enter configuration sub-mode for the current controller
```

```
controller-node 39df9a30-0ce7-423b-98ba-86d543aecf05
```

```
  Enter configuration sub-mode for a specific (possibly remote) controller
```

```

```

Set clock

Command Mode: config-controller mode

Command Syntax: `clock set <time> <day-of-month> {January | February | March | April | May | June | July | August | September | October | November | December} <year>`

Command Description:
Configure the clock for a controller instance. Within the controller-node configuration sub-mode, the user can configure set the clock.

When operating in HA mode with multiple controllers, setting the clock of the controllers to different values will have an effect on database reconciliation. Since time stamps are used to determine which of the shared items are more up-to-date, if the time between controllers is very skewed, the any database updates applied may or may no be aplied correctly. It is better to try to deal with time skew in the multiple controller enviromenment by using ntpdate.

Command Examples:

```
clock set 17:30:00 1 January 1970
```

```
  Set the clock manually
```

```

```

Configure time zone

Command Mode: config-controller mode

Command Syntax: `[no] clock timezone <time-zone>`

Command Description:
Configure the timezone for a controller instance's clock. Within the controller-node configuration sub-mode, the user can configure the local timezone for the node.

Command Examples:

```
clock timezone America/Los_Angeles
```

```
  Configure a specific timezone for this controller
```

```
no clock timezone
```

```
  Remove any timezone setting; the default is to use UTC.
```

```

```

Associate dns, default gateway with the controller node

Command Mode: config-controller mode

Command Syntax: `[no] ip {domain {lookup | name <domain-name>} | {name-server <ip>} | default-gateway <default-gateway>}`

Command Description:
Configure the IP-related settings of the controller node.

Next Keyword Descriptions:

- domain-lookups-enabled:
  Configure whether or not DNS lookups are enabled on the controller node.
- name <domain-name>:
  Specify the default domain name for the controller.
- name-server <ip>:
  Specify the IP address of the DNS server.
- default-gateway <default-gateway>:
  Specify the IP address of the default gateway.

Attach alias to controller

Command Mode: config-controller mode

Command Syntax: `controller-alias <alias>`

Command Description:
Configure an alias for the controller node.

Next Keyword Descriptions:

- alias:
  Specify the name of the alias for the controller node.

Enter interface submode, configure controller interface

Command Mode: config-controller mode

Command Syntax: `[no] interface <interface-type> <interface-number>`

Command Description:
Enter a submode to configure a network interface of the controller node.

Next Keyword Descriptions:

- Ethernet:
  Specify the type of the network interface, e.g. Ethernet.
- number:
  Specify the number of the network interface.

Configure firewall rule for controller-node

Command Mode: config-controller-if mode

Command Syntax: `[no] firewall allow [from <src-ip>] [local-ip <vrrp-ip>] {{openflow | web | ssh | ssl} | {udp | tcp} {{openflow | web | ssh | ssl} | <port>} | vrrp}`

Command Description:
Configure a firewall rule to allow traffic to the specified port number and protocol (tcp, udp, or vrrp) of the controller node.

Next Keyword Descriptions:

- web:
  The 'web' keyword identifies port 80. This is not only the typical web interface, but also the port for REST API requests.
- udp:
  By selecting the 'udp' keyword, the matched ip protocol is udp. A port number must be included for this selection.
- from <src-ip>:
  Associate the firewall rule with a specific source ip address. The rule will apply only to ip frames with this originating ip address.
- openflow:
  The 'openflow' keyword identifies port 6633.
- local-ip <vrrp-ip>:
  Associate the firewall rule with a specific destination ip address. The rule will apply only to ip frames for this destination ip address.
- tcp:
  By selecting the 'tcp' keyword, the matched ip protocol is tcp. A port number must be included for this selection.
- ssl:
  The 'ssl' keyword identifies port 443.
- vrrp:

- ssh:
  The 'ssh' keyword identified port 22.
- allow:
  The 'allow' keyword configures a firewall rule which describes a match condition for traffic. When the condition is satisfied, the traffic is allowed.
- port:
  Specify the port to which traffic is allowed in the firewall rule. The port can be an explicit port number or one of the following named ports: 'openflow' (port 6633), 'web' (port 80), 'ssl' (port 443) or 'ssh' (port 22).

Associate ip address with interface

Command Mode: config-controller-if mode

Command Syntax: `[no] ip {address {<ip-address> <netmask> | <cidr-address>} | mode {dhcp | static}}`

Command Description:
Configure the IP-related settings of the controller node.

Next Keyword Descriptions:

- ip:
  Specify the statically-configured IP address of the controller node (e.g. 192.168.1.1).
- netmask:
  Specify the statically-configured IP netmask of the controller node (e.g. 255.255.255.0).
- static:
  Specify the mode for configuring the IP address, either 'static' to specify an explicit IP address or 'dhcp' to obtain the IP address from a DHCP server.
- cidr:
  Specify the statically-configured CIDR address of the controller node (e.g. 192.168.1.1/24).
- dhcp:

Configure logging (syslog) for controller-node

Command Mode: config-controller mode

Command Syntax: `logging {on | server <server-ip-or-domain> [level {emerg | alert | crit | err | warning | notice | info | debug | 0 | 1 | 2 | 3 | 4 | 5 | 6 | 7}]}`

Command Description:
The controller-node logging command allows configuration of outgoing syslog messages associated with the current controller

Next Keyword Descriptions:

- info:
  The logging level allows configuration of the syslog logging level. The keywords provided directly map to the syslog levels, although the command keywords are abbreviated versions of the typical syslog levels.
- 1:
- notice:
- err:
- 0:
- alert:
- logging-enabled:
  Using the 'on' keyword in this command enables syslog logging.

- debug:
- 3:
- emerg:
- 5:
- 4:
- 7:
- 6:
- crit:
- 2:
- server:
  The 'server' keyword prefixes the ip address of the syslog server.
- warning:

Command Examples:

```
logging on
```

```
  Enable logging
```

```
no logging on
```

```
  Disable logging
```

```
logging server 1.2.3.4
```

```
  Send syslog records to a specific server or domain name
```

```
logging server 1.2.2.4 level warning
```

```
  Send syslog messages at a specific level (numeric or symbolic)
```

```
no logging server 1.2.3.4
```

```

```

Configure ntp for controller-node

Command Mode: config-controller mode

Command Syntax: `[no] ntp server <ntp-server>`

Command Description:
Configure the NTP server for the controller node.

Next Keyword Descriptions:

- server <ntp-server>:
  Specify the host name or ip address of the NTP server.

Command Examples:

```
ntp server 1.2.3.4
```

```
  Set the NTP server address
```

```
no ntp server
```

```
  Disable the NTP server
```

```

```

Feature Commands

Enable features for controller

Command Mode: config mode

Command Syntax: `[no] feature {onv | flow-pusher | performance-monitor}`

Command Description:
Enable a named feature for the controller image

Next Keyword Descriptions:

- onv:
  Configure this controller as a ONV
- flow-pusher:
  Enable the static flow pusher feature
- performance-monitor:
  Configure the performance monitoring feature for this controller

Command Examples:

```
feature onv
```

```
  Enable the ONV feature
```

```
no feature onv
```

```
  Disable the ONV feature
```

```

```

Ha Commands

Configure high availability

Command Mode: config mode

Command Syntax: `[no] ha {cluster-number <cluster-number>}`

Command Description:
Configure parameters for the VRRP protocol

Next Keyword Descriptions:

- cluster-number <cluster-number>:

Command Examples:

```
ha cluster-number 42
```

```
  Set the cluster number (VRRP router ID)
```

```
no ha cluster-number
```

```
  Reset the VRRP router ID to the default ('1')
```

```

```

Version Commands

Move to a specific version of command syntax

Command Mode: config mode

Command Syntax: `version <version>`

Command Description:
Switch to a specific version of command syntax.

This command is reserved for future use (there are currently no alternate versions).

Command Examples:

```
version XYZ
```

```
  Switch to version XYZ.
```

```

```

Forwarding Commands

Configure forwarding service properties

Command Mode: config mode

Command Syntax: `forwarding {access-priority <access-priority> | core-priority <core-priority>}`

Command Description:
The forwarding command configures attributes associated with the forwarding service.

Next Keyword Descriptions:

- core-priority:
  This attribute set the proprity for flows created by the forwarding service on a core switch
- access-priority:
  This attribute sets the proprity for flows created by the forwarding service on an access switch

Command Examples:

```
  forwarding core-priority 100
```

```
       To set priority for forwarding flow-mods on core switches to 100, use:
```

```
   forwarding access-priority 100
```

```
        To set priority for forwarding flow-mods on access switches to 100, use:
```

```

```

```

```

Host Commands

Host submode, configure host details

Command Mode: config mode

Command Syntax: `[no] host [address-space <address-space>] [vlan <vlan>] <mac>`

Command Description:
The host command enters a submode which allows configuration for the identified device. The command's parameters identify a unique device, which may require the use of the optional address-space and vlan parameters.

The identified device does not currently need to be known to the controller, providing for pre-configuraion of hosts.

Various host configuration within the submode includes host-alias, and host security associations.

If completion is requested for the hosts, the resulting entries shown are the collection of currently known hosts, and the collection of configured hosts.

The mac address identifying the specific host is case insensitive.

The 'no' variation of the host command will remove all configured details for the identified device. If a 'show host' is issued afterwards, the host will still appear if the controller has any operational state assocaited with the host.

Next Keyword Descriptions:

- address-space <address-space>:
  The optional address-space association for host allows the identified mac to be bound to a specific isolated address space.

When this optional parameter is not included, the mac is associated with the address-space named 'default'.

- vlan <vlan>:
  The optional vlan parameter allows the identified mac address to be associated with a specific vlan.

When this optional parameter isn't included, no vlan is associated with the mac.

Currently, the vlan may only be associated with a mac for the associated address-space 'default'.

Command Examples:

```
host 00:00:00:00:00:01
```

```
  Define a host with a specific MAC, and enter its configuration sub-mode
```

```
no host 00:00:00:00:00:01
```

```
  Delete a definition for a specific host
```

```
host address-space default 00:00:00:00:00:02
```

```
  Define a host and bind it to a specific address space
```

```
host vlan 42 00:00:00:00:00:03
```

```
  Define a host and bind it to a specific VLAN.
```

```
  Note here that VLAN associations are only valid with the 'default' address space.
```

```

```

```

```

Attach alias to host

Command Mode: config-host mode

Command Syntax: `[no] host-alias <id>`

Command Description:
The host-alias command allows associating a more identifiable name with the host identified by entering the host submode. Host aliases must start with an alphabetic character, and can continue with alphanumerics, '_', or '-'. The maximum length of a host alias is 255 characters.

Once an alias is associated with a host, various show commands will provide the alias along with, or instead of the mac address to identify the host.

Command Examples:

```
host-alias my-mac-book
```

```
  Associate a friendly name with the current host definition
```

```
no host-alias my-mac-book
```

```
  Remove a hostname association from this host
```

```

```

Configure security policies for host

Command Mode: config-host mode

Command Syntax: `[no] security policy bind {ip-address <ip-address> | attachment-point {all | <switch dpid or alias>} <if-name-regex>}`

Command Description:

The security command within the host submode is used to bind ip address and attachment points for the hosts identified by the submode.

What a host is presented to the controller, the attachment point of the host is also identified. When the security command is used to constrain the attachment point, the controller can use the configured details to choose whether it will allow the host to join the network.

When an ip address is bound to the host, no other host may use the indicated ip address. This is implemented by snooping arp's and the dhcp protocol. It is still possible for the host to send frames with spoofed src ip address, but the destination will not be able to reply to these frames.

Next Keyword Descriptions:

- attachment-point:
  The attachment point portion is intended to identify the switch or interface name, otherwise the host cannot transmit or receive network traffic.
- if-name-regex:
  This field is a regular expression, which is used to match against an interface name associated with the switch.
- bind:
  The bind keywork of the security policy command is used to configure various associations restricting the behavior of the host.
- dpid:
  The switch is part of the attachment point description. This can be a switch dpid, or an alias identifying a single switch.
- policy:
  The policy keyword is used to configure security policies associated with this host.
- ip-address <ip-address>:
  This identifies an ip address, it will prevent other host's from using any other ip address.

Command Examples:

```
security policy bind ip-address 10.10.10.1
```

```
  When the host sends any ip frames, the src address
```

```
  of these frames must be 10.10.10.1.
```

```

```

```
no security policy bind ip-address 10.10.10.1
```

```
  Remove a source-address binding requirement for this host
```

```
security policy bind attachment-point ntgr-7328-3
```

```
  The host identified by this submode can only send
```

```
  and receive traffic when it attached to this switch.
```

```
no security policy bind attachment-point ntgr-7328-3
```

```
  Remove an attachment point requirement for this host
```

```
security policy bind attachment-point ntgr-7328-3 12
```

```
  The host identified by this submode can only send
```

```
  and receive traffic when it attached to this switch and
```

```
  interface named '12'
```

```
no security policy bind attachment-point ntgr-7328-3 12
```

```
  Remove an attachment point/interface requirement for this host
```

Snmp-server Commands

Smnp configuration, enable server, configure parameters

Command Mode: config mode

Command Syntax: `snmp-server {enable | community ro <community> | location <location> | contact <contact>}`

Command Description:
Configure this device to respond to SNMP queries.

Configure SNMP protocol parameters, and configure how responses to SNMP queries are composed.

Next Keyword Descriptions:

- enable:
  Enable this device for responding to SNMP.

Use the 'no' version of this command to disable SNMP features.

- location:
  Configure this device's location via the sysLocation SNMP MIB.

Reset the location to the system default with the 'no' version of this command.

- community:
  Configure the community string for simple read-only SNMP client authentication.

Reset the community string (default empty) with the 'no' version of this command.

- contact:
  Configure the adminstrative contact record (the SNMP sysContact MIB) for this device.

Reset the contact information to system defaults with the 'no' version of this command.

Command Examples:

```
snmp-server enable
```

```
  Enable SNMP support.
```

```
no snmp-server enable
```

```
  Disable SNMP support.
```

```
snmp-server community ro MY-SNMP
```

```
  Set the community string (for authenticating to this SNMP service)
```

```
  to "MY-SNMP"
```

```
no snmp-server community
```

```
  Reset the community string to default (the empty string)
```

```
snmp-server location snmp.example.com
```

```
  Set the server location reported during SNMP queries.
```

```
no snmp-server location
```

```
  Reset to the system default server location.
```

```
snmp-server contact admin@example.com
```

```
  Set the administrative contact reported during SNMP queries.
```

```
no snmp-server contact
```

```
  Reset to the system default administrative contact setting.
```

```
```

Switch Commands

Enter switch submode, configure switch details

Command Mode: config mode

Command Syntax: `[no] switch <dpid>`

Command Description:
The switch command enters the switch submode for a single identified switch. Within the submode, various configuraion can be performed on the switch, including setting the switch-alias, enable or disabling the tunnel features.

The 'no' variation of the switch command will not remove swtich's currently connected to the controller, instead it will remove any user configured details of the identified switch.

Command Examples:

```
switch 00:00:00:00:00:00:00:01
```

```
  Define a new switch, and enter the switch configuration sub-mode
```

```
no switch 00:00:00:00:00:00:00:01
```

```
  Delete a switch definition
```

```

```

Set actions for this flow

Command Mode: config-switch-flow-entry mode

Command Syntax: `[no] actions <actions>`

Next Keyword Descriptions:

- actions:

Set flow active

Command Mode: config-switch-flow-entry mode

Command Syntax: `[no] active {True | False}`

Command Description:
Enable or disable this flow entry

Next Keyword Descriptions:

- False:
  Make this flow entry inactive

- True:
  Make this flow entry active

Command Examples:

```
active True
```

```
  Make active
```

```
active False
```

```
  Make inactive
```

```

```

Associate cookie for flow

Command Mode: config-switch-flow-entry mode

Command Syntax: `[no] cookie <cookie>`

Command Description:
Assign a cookie value (32-bit integer) to the flow entry

Command Examples:

```
cookie 42
```

```
  Assign a cookie value to this entry
```

```

```

Configure dst-ip match for flow

Command Mode: config-switch-flow-entry mode

Command Syntax: `[no] dst-ip {<ip-address> | <cidr-range>}`

Command Description:
Associate a destination IP address with this flow entry

Next Keyword Descriptions:

- dst-ip:
  Enter an IP address or CIDR address range

Command Examples:

```
dst-ip 1.2.3.4
```

```
  Associate a specific host address with this flow entry
```

```
dst-ip 1.2.3.4/24
```

```
  Associate a destination address range with this flow entry
```

```

```

Configure dst-mac match for flow

Command Mode: config-switch-flow-entry mode

Command Syntax: `[no] dst-mac <dst-mac>`

Command Description:
Associate a destination MAC address with this flow entry

Next Keyword Descriptions:

- dst-mac:
  Enter a MAC address or host alias

Command Examples:

```
dst-mac 00:00:00:00:00:01
```

```
  Associate a host by MAC address
```

```
dst-mac my-computer
```

```
  Associate a host alias with this flow entry
```

Configure dst-port match for flow

Command Mode: config-switch-flow-entry mode

Command Syntax: `[no] dst-port {<dst-port> | {http | dns | https | ssh}}`

Command Description:
Associate a TCP or UDP port with this flow entry

Next Keyword Descriptions:

- dst-port:
  Enter a TCP or UDP port number, or well-known service name

Command Examples:

```
dst-port 80
```

```
  Associate a port by number
```

```
dst-port https
```

```
  Associate a port by service name
```

Configure ether-type match for flow

Command Mode: config-switch-flow-entry mode

Command Syntax: `[no] ether-type {<ether-type> | {arp | lldp | 802.1Q | ip | mpls | rarp | mpls-mc | appletalk-aarp | ipv6 | novell | ipx}}`

Command Description:
Match flow entries by ether type

Next Keyword Descriptions:

- ether-type:
  Specify an ether by by number or by alias

Command Examples:

```
ether-type 0x88a2
```

```
  Match AOE frames
```

```
ether-type arp
```

```
  Match ARP frames
```

```
```

Set hard-timeout for this flow

Command Mode: config-switch-flow-entry mode

Command Syntax: `[no] hard-timeout <hard-timeout>`

Command Description:
Associate a hard timeout with this flow entry

Next Keyword Descriptions:

- hard-timeout:
  Specify a timeout in seconds

Command Examples:

```
hard-timeout 30
```

```
  Time out this flow after 30s
```

```
```

Set idle-timout for this flow

Command Mode: config-switch-flow-entry mode

Command Syntax: `[no] idle-timeout <idle-timeout>`

Command Description:
Set an idle timeout for this flow entry

Next Keyword Descriptions:

- idle-timeout:
  Specify a timeout value in seconds

Command Examples:

```
idle-timeout 30
```

```
  Set the idle timeout for 30 seconds
```

Configure wildcards for flow

Command Mode: config-switch-flow-entry mode

Command Syntax: `[no] ingress-port <ingress-port>`

Command Description:
Associate an OF ingress port with this flow entry

Next Keyword Descriptions:

- ingress-port:
  Specify an OF ingress port (16-bit number)

Command Examples:

```
ingress-port 32
```

```
  Assiciate an ingress port by number
```

Set priority of the flow

Command Mode: config-switch-flow-entry mode

Command Syntax: `[no] priority <priority>`

Command Description:
Assign a priority to this flow entry

Next Keyword Descriptions:

- priority:
  Specify the priority as a 16-bit integer

Command Examples:

```
priority 1000
```

```
  Give this flow entry a fixed priority
```

Configure ether-type match for flow

Command Mode: config-switch-flow-entry mode

Command Syntax: `[no] protocol <protocol>`

Command Description:
Associate a specific prototype type to this flow entry

Next Keyword Descriptions:

- protocol:
  Specify a protocol by number

Command Examples:

```
prototol 17
```

```
  Associate TCP packets with this flow entry
```

Configure src-ip match for flow

Command Mode: config-switch-flow-entry mode

Command Syntax: `[no] src-ip {<ip-address> | <cidr-range>}`

Command Description:
Associate a source IP address or range with this flow entry

Next Keyword Descriptions:

- src-ip:
  Specify an IP address or address range

Command Examples:

```
src-ip 1.2.3.4
```

```
  Match a specific address
```

```
src-ip 1.2.3.4/23
```

```
  Match an IP address range
```

Configure src-mac match for flow

Command Mode: config-switch-flow-entry mode

Command Syntax: `[no] src-mac <src-mac>`

Command Description:
Associate a source MAC address with this flow entry

Next Keyword Descriptions:

- src-mac:
  Specify a MAC address or host alias

Command Examples:

```
src-mac 00:00:00:00:00:02
```

```
  Specify a host by MAC address
```

```
src-mac my-server
```

```
  Specify a host by alias
```

Configure src-port match for flow

Command Mode: config-switch-flow-entry mode

Command Syntax: `[no] src-port {<src-port> | {http | dns | https | ssh}}`

Command Description:
Associate a source TCP or UDP port with this flow entry

Next Keyword Descriptions:

- src-port:
  Specify a TCP or UDP port by number or service name

Command Examples:

```
src-port 119
```

```
  Associate a port by number
```

```
src-port ftp-data
```

```
  Associate a port by service name
```

```

```

Configure ether-type match for flow

Command Mode: config-switch-flow-entry mode

Command Syntax: `[no] tos-bits <tos-bits>`

Command Description:
Associate packets with this flow entry by TOS bits

Next Keyword Descriptions:

- tos-bits:
  Specify TOS bits as a numeric mask

Command Examples:

```
tos-bits 6
```

```
  Match specific TOS bits
```

Configure vlan-id match for flow

Command Mode: config-switch-flow-entry mode

Command Syntax: `[no] vlan-id <vlan-id>`

Command Description:
Match packets to this flow entry with a specific VLAN id

Next Keyword Descriptions:

- vlan-id:
  Specify a VLAN id (12-bit integer)

Command Examples:

```
vlan-id 10
```

```
  Match a specific VLAN id
```

```

```

Configure vlan-priority match for flow

Command Mode: config-switch-flow-entry mode

Command Syntax: `[no] vlan-priority <vlan-priority>`

Command Description:
Match packets with a specific VLAN priority field

Next Keyword Descriptions:

- vlan-priority:
  Specify a priority field as a 3-bit integer

Command Examples:

```
vlan-priority 4
```

```
  Match packets with by priority field (in the VLAN header)
```

```

```

Configure wildcards for flow

Command Mode: config-switch-flow-entry mode

Command Syntax: `[no] wildcards <wildcards>`

Command Description:
Associate a wildcard value with this flow entry

Next Keyword Descriptions:

- wildcards:
  Specify a wildcard value (32-bit integer)

Command Examples:

```
wildcards 1000
```

```
  Associate a wildcard value with the currently-defined flow entry
```

Enter flow-entry submode, configure single static flow entry

Command Mode: config-switch mode

Command Syntax: `[no] flow-entry <name>`

Command Description:
Define a flow-entry and enter its configuration sub-mode

Command Examples:

```
flow-entry example-1
```

```
  Define a new flow entry
```

```
no flow-entry example-1
```

```
  Delete a flow entry by name
```

```

```

Configure interface as connected to an external network

Command Mode: config-switch-if mode

Command Syntax: `[no] switchport mode <interface connects to external network>`

Command Description:
Configure this interface to connect to an external network

Command Examples:

```
switchport mode external
```

```
  Within a switch interface definiton, configure this interface to be
```

```
  connected to an external network
```

```
no switchport mode external
```

```
  Remove the external connection attribute for this interface
```

```
```

Enable core-switch property for this switch

Command Mode: config-switch mode

Command Syntax: `[no] core-switch`

Command Description:
This switch is a core switch.

Command Examples:

```
core-switch
```

```
  Configure the currently-configured switch as a core switch
```

```
no core-switch
```

```
  Remove the core-switch property
```

```
```

Enter switch-if submode, configure switch interface

Command Mode: config-switch mode

Command Syntax: `[no] interface <name>`

Command Description:
Specify a switch interface by name, and enter its configuration sub-mode

Command Examples:

```
interface Ethernet1
```

```
  Configure a specific port on this switch
```

```
no interface Ethernet1
```

```
  Delete a switch interface configuration
```

```

```

Attach alias to switch interface

Command Mode: config-switch-if mode

Command Syntax: `[no] interface-alias <id>`

Command Description:
Configure an alias for this switch interface.

Command Examples:

```
interface-alias defintf
```

```
  Assign an alias to this switch interface
```

```
no interface-alias defintf
```

```
  Remove an alias for this switch interface
```

```

```

```

```

Attach alias to switch

Command Mode: config-switch mode

Command Syntax: `[no] switch-alias <id>`

Command Description:
Assign an alias to this switch

Command Examples:

```
alias switch-1
```

```
  Attach a name to this switch
```

```
no alias switch-1
```

```
  Remove a switch alias
```

```

```

Enable/disable tunnel creation for this switch

Command Mode: config-switch mode

Command Syntax: `tunnel termination {enabled | disabled}`

Command Description:
Configure tunnel attributes for this switch

Command Examples:

```
tunnel termination enabled
```

```
  Enable tunnel termination for this switch
```

```
tunnel termination disabled
```

```
  Disable tunnel termination for this switch
```

```
no tunnel termination
```

```
  Disable tunnel termination for this switch
```

```

```

Aaa Commands

Configure accounting parameters

Command Mode: config mode

Command Syntax: `aaa accounting exec default {none | start-stop group tacacs+}`

Command Description:
Configure accounting (logging for user CLI events). Accounting is configured for a specific role, via a specific channel, for specific events, using one or more accounting services.

Next Keyword Descriptions:

- tacacs+:
  Configure the currently-defined TACACS+ servers (the 'tacacs+' group) as an accounting service.
- none:
  Disable accounting services for this role and channel.

Note that local (/var/log) accounting is always enabled.

- start-stop:
  Configure accounting services to record 'start' (login) and 'stop' (logout) events.
- group:
  Configure accounting services to use a specific group of accounting servers.
- exec:
  Configure accounting services for exec mode (CLI access).
- default:
  Configure authorization sources while connecting via the default channel (currently SSH).

Reset the authorization sources (currently to 'local' only) with the 'no' version of this command.

Command Examples:

```
aaa accounting exec default none
```

```
  Turn off accounting services for SSH CLI accesses.
```

```
  The native (/var/log) accounting cannot be disabled.
```

```
no aaa accounting exec default
```

```
  Reset the accounting services for SSH CLI accesses to defaults.
```

```
  This disables all accounting services, with the exception
```

```
  of native (/var/log) accounting, which is always enabled.
```

```
aaa accounting exec default start-stop group tacacs+
```

```
  Configure the currently-defined set of TACACS+ servers
```

```
  as an accounting service for SSH CLI login and logout.
```

```

```

```

```

Tacacs Commands

Tacacs timeout, ip server address

Command Mode: config mode

Command Syntax: `tacacs server {timeout <timeout> | key <key> | host <ip> [key {<key>}]}`

Command Description:
Configure server and protocol parameters for TACACS+. Specify remote TACACS+ servers, along with protocol timeouts and security settings.

TACACS+ specific configuration here is enabled by a corresponding 'aaa' command to enable authentication, authorization, or accounting via the TACACS+ protocol.

Next Keyword Descriptions:

- host <ip>:
  Configure TACACS+ servers by IP address for aaa roles.Hosts are not active for aaa rules unless the associated 'aaa' command is used. Delete the host with the 'no' form of this command.
- key:
  Configure the default TACACS+ protocol key used to secure the communications channel with the TACACS+ server(s).

The default key is used if there is no key defined for an individual server.

The default key can be reset (to an empty key) with the 'no' form of this command.
Specify the TACACS+ encryption key used to communicate with this TACACS+ server. If no key is specified, the global key is used.

Reset the per-host key (and fall back to the global key) with the 'no' version of this command.

- timeout:
  Configure the global timeout for communicating with TACACS+ servers. The timeout is specified in seconds.

If the timeout expires before the TACACS+ server responds, then the TACACS+ aaa action is considered 'failed', and other methods can be attempted.

The server timeout is reset to the system default using the 'no' command prefix.

Command Examples:

```
tacacs server timeout 50
```

```
  Configure the global TACACS+ protocol timeout to 50s.
```

```
no tacacs server timeout
```

```
  Configure the default TACACS+ prototol timeout.
```

```
tacacs server key SECRET-KEY
```

```
  Configure the (plaintext) key used to communicate with
```

```
  the TACACS+ server, if no key is specified for a given server.
```

```
no tacacs server key
```

```
  Reset the global TACACS+ server key.
```

```
  If no key is specified, an empty key is assumed.
```

```
tacacs server host AA.BB.CC.DD
```

```
  Add a TACACS+ server with the specified IP address.
```

```
no tacacs server host AA.BB.CC.DD
```

```
  Delete the specified TACACS+ server
```

```
tacacs server host AA.BB.CC.DD key SECRET-KEY
```

```
  Specify an encryption key for this TACACS+ server.
```

```
no tacacs server host AA.BB.CC.DD key
```

```
  Remove any key associated with this server,
```

```
  and use the global TACACS+ key instead.
```

```
```

```
```

```
```

Tag Commands

Enter tag, configure switch details

Command Mode: config mode

Command Syntax: `[no] tag <id>`

Command Description:
The tag command enters the tag submode for a single tag-value. The tag-value is composed of a tag and the value assigned to it, tag itself is composed of namespace that the tag belongs to and the tag's name itself. The syntax of tag-value - <namespace>.<name>=<value>. Example of tag-value is - "tenant.name=coke". Within the tag submode, this tag-value can be associated with hosts through match command.

The 'no' variation of the tag command will remove the tag-value and untag the hosts.

Command Examples:

```
tag example-com.role=primary
```

```
  Define a tag association, and enter the tag configuration sub-mode
```

```
no tag example-com.role=primary
```

```
  Remove a tag definition
```

```

```

Set the match rule for this tag

Command Mode: config-tag mode

Command Syntax: `[no] match [mac <mac>] [vlan <vlan>] [switch <dpid> [<ifname>]]`

Command Description:

The match command is used to tag hosts. MAC-address, VLAN, switch, switch-port are the fields that can be matched to tag hosts. These fields can be matched individually or in conjunction. For instance, if match is specified to match a MAC-Address, VLAN and switch-port, then the logical host that has that MAC-Address, sending packets tagged with that VLAN and connected to that switch-port is tagged. Whereas, if the match is just on a switch, then all the hosts connected to that switch are tagged. Tag-value can be associated with multiple such match statements, thus tagging hosts that match atleast one of the match statements.

The 'no' variation of the match command will remove the match statement and untag all the hosts that were tagged as a result of match on that match statement.

Command Examples:

```
localhost(config)# tag tenant.name=coke
```

```
localhost(config-tag)#
```

```
localhost(config-tag)# match mac 00:00:00:00:00:01
```

```
localhost(config-tag)#
```

```
localhost(config-tag)# match vlan 2
```

```
localhost(config-tag)# match switch 00:00:00:00:00:00:00:01
```

```
localhost(config-tag)# match switch 00:00:00:00:00:00:00:01 eth1
```

```
localhost(config-tag)# match mac 00:00:00:00:00:02 vlan 3
```

```
localhost(config-tag)# match mac 00:00:00:00:00:03 switch 00:00:00:00:00:00:00:02
```

```
localhost(config-tag)# match mac 00:00:00:00:00:03 switch 00:00:00:00:00:00:00:02 eth2
```

```
localhost(config-tag)# match mac 00:00:00:00:00:03 vlan 3 switch 00:00:00:00:00:00:00:02 eth3
```

Tech-support-config Commands

Manage command output for show tech-support

Command Mode: config mode

Command Syntax: `[no] tech-support-config {cli | shell} <cmd>`

Command Description:
This command sets a specific shell or cli command to be included in the show tech-support output. A 'no' prepended to this command will delete the shell/cli command from the show tech-support output.

Next Keyword Descriptions:

- shell:
  Including the following as a shell command in the tech-support output
- cli:
  Include the following as a Cli command in the tech-support output

Command Examples:

```
localhost(config)# tech-support-config shell date
```

```
localhost(config)#
```

```

```

Tenant Commands

Enter tenant definition submode

Command Mode: config mode

Command Syntax: `[no] tenant <name>`

Command Description:
This submode is used to create a named tenant. Within this submode, properties of the tenant can be configured. ONV are configured and managed, which configure the membership of devices. Virtual router is configured and managed, which defines the connectivity within and among tenants.

The controller provides a tenant named 'default' to collect devices which are not associated with any tenant.

The controller provides a tenant named 'system' to hold system virtual router which is used to define the connectivity among all tenants.

The controller provides a tenant named 'external' to hold configuration for external network access.

Next Keyword Descriptions:

- name:

Command Examples:

```
tenant my-first-tenant
```

```
  Define a new tenant instance by name,
```

```
  and enter the tenant sub-mode
```

```
no tenant my-first-tenant
```

```
  Remove a named tenant
```

Set tenant active

Command Mode: config-tenant mode

Command Syntax: `[no] active`

Command Description:
Set the tenant active. If a tenant is inactive the controller will not use it or its rules.

Command Examples:

```
active
```

```
  Activate this tenant instance
```

```
no active
```

```
  Deactivate this tenant instance
```

```

```

Provide description for a tenant instance

Command Mode: config-tenant mode

Command Syntax: `[no] description <description>`

Command Description:
Within a tenant definition sub-mode, declare a friendly descriptor for the tenant instance.

Command Examples:

```
description "this is my first tenant instance"
```

```
  Add a textual description to a tenant
```

```

```

Describe tenant origin

Command Mode: config-tenant mode

Command Syntax: `[no] origin <origin>`

Set routing rule

Command Mode: config-tenant-router mode

Command Syntax: `[no] route from {tenant <source tenant> [onv <source onv>] | onv <source onv> | {<src-ip> <src-ip-mask> | <src-ip> | <src-cidr> | any` to {tenant <destination tenant> [onv <destination onv>] | onv <destination onv> | {<dst-ip> <dst-ip-mask> | <dst-ip> | <dst-cidr> | any}} [<next hop ip address> | gw-pool <gateway pool name>] [<outgoing interface>] {deny | permit}}}

Command Description:
This command is used to create a new routing rule for a given tenant virtual router.

Next Keyword Descriptions:

- outgoing-intf:

outgoing interface name
- src-ip-mask:
  An inverse netmask in dotted decimal notation.
- gateway-pool:
  gateway pool name
- nh-ip:
  IP address in dotted decimal notation.
- src-ip:
  IP address in dotted decimal notation.
  IP address with prefix length in CIDR format.
- any:
  Represents any IP address.
- dst-ip:
- dst-ip-mask:

Command Examples:

```
route from tenant A to tenant B permit
```

```
  create a routing rule to permit packets from tenant A to tenant B
```

```
no route from tenant A to tenant B permit
```

```
  remove the created routing rule
```

Provide description for a virtual router instance

Command Mode: config-tenant-router mode

Command Syntax: `[no] description <description>`

Command Description:
Within a tenant virtual router definition sub-mode, declare a friendly descriptor for the virtual router instance.

Command Examples:

```
localhost(config-tenant-router)# description "this is the virtual router for testing"
```

```
  Add a textual description to a tenant virtual router
```

Describe virtual router origin

Command Mode: config-tenant-router mode

Command Syntax: `[no] origin <origin>`

Describe virtual router interface origin

Command Mode: config-tenant-router-intf mode

Command Syntax: `[no] origin <origin>`

Add ip address to the gateway pool

Command Mode: config-tenant-router-gw mode

Command Syntax: `[no] ip <ip-address>`

Command Description:
Add an IP address to a gateway pool

Next Keyword Descriptions:

- ip-address:
  IP address in dotted decimal notation.

Command Examples:

```
localhost(config-tenant-router-gw)# ip  10.0.1.1/24
```

```
localhost(config-tenant-router-gw)# no ip  10.0.1.1/24
```

```

```

Set virtual router interface active

Command Mode: config-tenant-router-intf mode

Command Syntax: `[no] active`

Command Description:
Set the tenant virtual router active. If a tenant virtual router is inactive the controller will not use it or its rules.

Command Examples:

```
localhost(config-tenant-router-intf)# active
```

```
  Activate this interface instance
```

```
localhost(config-tenant-router-intf)# no active
```

```
  Deactivate this interface instance
```

```

```

Set virtual router interface ip address

Command Mode: config-tenant-router-intf mode

Command Syntax: `[no] ip {<ip-address> <subnet-mask> | <src-cidr>}`

Command Description:
Associate an IP address and IP subnet with an virtual router interface.

Next Keyword Descriptions:

- subnet-mask:
  An inverse netmask in dotted decimal notation.
- ip-address:
  IP address in dotted decimal notation.
  IP address with prefix length in CIDR format.

Command Examples:

```
localhost(config-tenant-router-intf)# ip  10.0.1.1/24
```

```
localhost(config-tenant-router-intf)# no ip  10.0.1.1/24
```

Enter virtual router definition submode

Command Mode: config-tenant mode

Command Syntax: `[no] router <vrname>`

Command Description:
This submode is used to create, then describe the connectivity and routing rules for the named virtual router for a given tenant. Within this submode, properties of the virtual router can be configured.

Command Examples:

```
router my-first-router
```

```
  Define a new router instance by name,
```

```
  and enter the tenant-router sub-mode
```

```
no router my-first-router
```

```
  Remove a named router
```

Enter virtual router gateway pool definition submode

Command Mode: config-tenant-router mode

Command Syntax: `[no] gateway-pool <vrgwname>`

Command Description:
This command is used to create a new next hop gateway pool. Any IP address within this gateway pool can be used as a next hop for a routing rule configured with the next hop as this gateway pool.

Command Examples:

```
localhost(config-tenant-router)# gateway-pool pool1
```

```
  create a gateway pool pool1
```

```
localhost(config-tenant-router)# no gateway-pool pool1
```

```
  remove the gateway pool named pool1
```

```

```

Enter virtual router interface definition submode

Command Mode: config-tenant-router mode

Command Syntax: `[no] interface <vriname> [onv <onv-connected> | tenant <router-connected-tenant> <router-connected>]`

Command Description:
This command is used to create a new virtual router interface and connect the named interface to a defined ONV or another virtual router.

Next Keyword Descriptions:

- onv:
  the 'onv' keyword specifies the named virtual router interface connects to a ONV within the same tenant scope.
- tenant:
  the 'tenant' keyword specifies the named virtual router interface connects to another tenant virtual router.

Command Examples:

```
localhost(config-tenant-router)# interface if1 onv A1
```

```
  create a virtual router interface if1 and connect it to ONV A1
```

```

```

```
localhost(config-tenant-router)# no interface if1
```

```
  remove the named virtual router interface if1
```

Topology Commands

Enable features for controller

Command Mode: config mode

Command Syntax: `[no] topology {autoportfast}`

Command Description:
The topology command configures attributes associated with the toplogy management of the controller.

Next Keyword Descriptions:

- autoportfast:

Command Examples:

```
   topology autoportfast
```

```
       Enable autoportfast and suppress link discovery on fast ports.
```

```
    no topology autoportfast
```

```
        Do not suppress link discovery on fast ports.
```

```

```

Vcenter Commands

Enter vcenter submode, configure vcenter details

Command Mode: config mode

Command Syntax: `[no] vcenter <vcenter-name>`

Enable vcenter connect

Command Mode: config-vcenter mode

Command Syntax: `[no] connect`

Enter vcenter-dvs submode, describe port groups

Command Mode: config-vcenter mode

Command Syntax: `[no] dvs datacenter-name <datacenter> dvs-name <dvs-name> [switch-class <switch-class>]`

Associate ip address for vcenter connection

Command Mode: config-vcenter mode

Command Syntax: `ip {address <ip>}`

Configure vcenter password for login

Command Mode: config-vcenter mode

Command Syntax: `password <password>`

Describe dvs portgroup

Command Mode: config-vcenter-dvs mode

Command Syntax: `portgroup <portgroup-name>`

Associate http port for vcenter connection

Command Mode: config-vcenter mode

Command Syntax: `port <port>`

Configure vcenter username for login

Command Mode: config-vcenter mode

Command Syntax: `username <username>`

Arp Commands

Set static arp

Command Mode: config mode

Command Syntax: `[no] arp <ip> <mac>`

Command Description:
The 'arp' command is used to create static arp <-> ip address bindings.

Command Examples:

```
arp 10.0.0.1 11:22:33:44:55:66
```

```
  create a static arp entry
```

```
no arp 10.0.0.1 11:22:33:44:55:66
```

```
  remove a static arp entry
```

Show Commands

Show Address-space Commands

Show all address spaces

Command Mode: login mode

Command Syntax: `show address-space`

Show a specific address space

Command Mode: login mode

Command Syntax: `show address-space {<address-space-name> | all} [{details | brief}]`

Command Description:
This 'show address-space' command variation is used to identify a specific address-space (the 'all' token provides display for every address-spaceJ), to describe particular operational or configured details.

Next Keyword Descriptions:

- details:
  The details keyword requests a more verbose version of various show commands.
- brief:
  The brief keywork requests a less verbose version of output for various show commands.

Show the configured identifier-rules for a specific address space

Command Mode: login mode

Command Syntax: `show address-space {<address-space-name> | all} {identifier-rules}`

Command Description:
The 'identifier-rules' keyword requests the display of the configured identifier-rules for the named adderss-space. The address-space identifier-rules describe the membership rules associated with an address-space.

Show onv-definition Commands

Show all defined onvs belong to current tenant

Command Mode: login mode

Command Syntax: `show onv`

Show specific onv, identified by name

Command Mode: login mode

Command Syntax: `show onv {<onv-id> | all} [{details | brief}]`

Command Description:
This 'show onv <id>' command variation is used to identify a specific onv or ONVs defined under current tenant mode. This 'show onv all' command variation is used to display all ONVs defined across all tenants. Being backward compatible, 'show onv' command under non config-tenant mode is also accepted and displays the ONVs for default tenant only.

Next Keyword Descriptions:

- details:
  The details keyword requests a more verbose version of various show commands.
- brief:
  The brief keywork requests a less verbose version of output for various show commands.

Show onv associated details based on name

Command Mode: login mode

Command Syntax: `show onv {<onv-id> | all} {interfaces | mac-address-table | interface-rules | access-lists | running-config | switch | flow [{brief | full-detail | details | summary}]}`

Command Description:
This 'show onv <id>' command variation is used to identify a specific onv or ONVs defined under current tenant mode. This 'show onv all' command variation is used to display all ONVs defined across all tenants. Being backward compatible, 'show onv' command under non config-tenant mode is also accepted and displays the ONVs for default tenant only.

Next Keyword Descriptions:

- access-lists:
  The 'access-list' keyword requests display of the configured access-lists associated with the onv. The output includes not only the access-list, but a brief descripion of the acl rules associated with the access-list.
- full-detail:
  For a few show commands, the 'full-detail' is a request to display more information than the 'defails' keyword.
- flow:
  The 'flow' keyword requests the display of all openflow flow-match entries inserted into various switches to implement the onv isolation.
- brief:
  The brief keywork requests a less verbose version of output for various show commands.
- summary:
  A more terse output format.
- interface-rules:
  The 'interface-rules' keyword requests the display of the configured interface-rules for the named onv. The onv interface-rules describe the membership rules associated with a onv.
- details:
  The details keyword requests a more verbose version of various show commands.

Show onv-interface-access-list Commands

Show access-group details

Command Mode: config-tenant-onv-if mode

Command Syntax: `show access-group`

Show onv-access-list Commands

Show onv access lists

Command Mode: config-tenant-onv- mode

Command Syntax: `show access-list`

Show onv-access-list-entry Commands

Show onv access list rules

Command Mode: config-tenant-onv mode

Command Syntax: `show access-list-entry`

Show onv-interface Commands

Show onv associated interfaces

Command Mode: config-tenant-onv config-tenant-onv-if mode

Command Syntax: `[no] show interfaces`

Command Description:
Show ONV interfaces associated with this ONV and its state and configuration.

Show Controller-interface Commands

Show controller-node associated interfaces

Command Mode: config-controller mode

Command Syntax: `show interfaces [type <type>] [number <number>]`

Command Description:
Display all the interfaces associated with the specified controller-node.

Next Keyword Descriptions:

- type <type>:
  Specify the type of the network interface to show, e.g. Ethernet.
- number <number>:
  Specify the number of the network interface to show.

Show Controller-node Commands

Show controller nodes summaries

Command Mode: login mode

Command Syntax: `show controller-node`

Command Description:
Show controller-node displays operational details for the identified controller nodes. Among the items displayed are the controller-alias, the HA role, and also a column labeled '@' to identify the controller currently connected to the CLI

Show controller-node associated details by name

Command Mode: login mode

Command Syntax: `show controller-node {<id> | all} [interfaces | firewall | summary | switches | alias] [{details | brief}]`

Command Description:
Show controller node associated information, for example the interfaces or switches related to a controller.

Next Keyword Descriptions:

- all:
  The 'all' token is used in place of an identifier's value to request every item associated with the command. The 'all' token is reserved, identifiers may not use the 'all' token as their name, no switch-alias or host-alias may be called 'all'
- firewall:
  Show the firewall rules for the specified controller node.
- interfaces:
  Display all the interfaces associated with the specified controller-node.
- brief:
  Specify the detail level for the show command. Value is either 'detail' or 'brief'.
- summary:
  Show a summary of the configuration for the specified controller node.
- switches:
  Show the switches connected to the specified controller node.
- alias:
  Show the alias for the specified controller node.
- details:
- id:
  Specify the id or alias of the controller node.

Show detailed controller-node related statistics

Command Mode: login mode

Command Syntax: `show controller-node {<id> | all} stats {cpu-user | disk-root | mem-used | mem-free | cli-cpu | database-cpu | swap-used | disk-boot | statd-cpu | cpu-system | cpu-idle | apache-cpu | cpu-nice | disk-log | fl-cpu} [start-time <start-time>] [end-time <end-time>] [duration <duration>] [sample-interval <sample-interval>] [sample-count <sample-count>] [sample-window <sample-window>] [data-format {value | rate}] [display {latest-value | graph | table}]`

Next Keyword Descriptions:

- cpu-user:
  Specify the type of controller stats to show.
- end-time <end-time>:
  Specify the end time for displaying the controller stats.
- all:
  The 'all' token is used in place of an identifier's value to request every item associated with the command. The 'all' token is reserved, identifiers may not use the 'all' token as their name, no switch-alias or host-alias may be called 'all'
- cpu-idle:
- swap-used:
- database-cpu:
- latest-value:
  Show the specified stats as either the latest value, graph or table.
- table:
- sample-interval <sample-interval>:
  Specify the interval between sample points.
- disk-root:
- disk-boot:
- graph:

- cli-cpu:
- sample-window <sample-window>:
  Specify the sample window for showing the stats. The sample window is the number of raw stat values to average around a down-sampled data point.
- mem-used:
- mem-free:
- sample-count <sample-count>:
  Specify the number of data points to show across the specified time range.
- start-time <start-time>:
  Specify the start time for displaying the stats.
- apache-cpu:
- cpu-system:
- rate:
  Show either the actual value or the rate of change of the value for the specified stats type.
- value:
- duration <duration>:
  Specify the duration for which to display stats. If this value is specified you should specify either the start-time or end-time but not both.
- statd-cpu:
- cpu-nice:
- disk-log:
- fl-cpu:

f

Show statistics for a given controller node

Command Mode: login mode

Command Syntax: `show controller-node {<id> | all} stats`

Next Keyword Descriptions:

- all:
  The 'all' token is used in place of an identifier's value to request every item associated with the command. The 'all' token is reserved, identifiers may not use the 'all' token as their name, no switch-alias or host-alias may be called 'all'
- id:
  Specify the id or alias of the controller node.

Show Config Commands

Show saved configs (ex: startup-config, etc)

Command Mode: login mode

Command Syntax: `show config [<first> diff <second> [<version>] | <config> [<version>]]`

Next Keyword Descriptions:

- second:
  Second configuration file selection.
- version:

- config:
  When a single config file is selected, the complete contents of the file is displayed
- first:
  When a pair of configuration files are selected, with the 'diff' keyword separating the two files, this positional parammeter identifies the first of the two config files to diff.

Show Config-file Commands

Show a specific saved config file

Command Mode: login mode

Command Syntax: `show config-file [<config>]`

Show Event-history Commands

Show recent network or system events

Command Mode: login mode

Command Syntax: `show event-history {topology-link | topology-switch | topology-cluster} [last <count>]`

Next Keyword Descriptions:

- last <count>:
  Limit the output to indicated number of lines
- topology-switch:
- topology-link:
- topology-cluster:

Show External-ports Commands

Show switch ports connected to external l2 networks

Command Mode: login mode

Command Syntax: `show external-ports`

Command Description:
The external-ports command displays the set of switch ports connected to external L2 networks.

Show Feature Commands

Show enabled and disabled features

Command Mode: login mode

Command Syntax: `show feature`

Show Firewall-rule Commands

Show firewall rules for controller interfaces

Command Mode: login mode

Command Syntax: `show firewall-rule [controller <controller>] [type <type>] [number <number>] [port <port>] [proto {tcp | udp | vrrp}] [src-ip <src-ip>] [local-ip <vrrp-ip>]`

Next Keyword Descriptions:

- udp:
  Filter the show command output by the indicated protocol.
- local-ip <vrrp-ip>:
  Select the rules which match this source ip address.
- src-ip <src-ip>:
- tcp:
- type <type>:
  Filter the show command output by the indicated interface type
- vrrp:
- port <port>:
  Filter the show command output by the indicated filtered port.
- number <number>:
  Filter the show command output by the indicated interface number.
- controller <controller>:
  Filter the show command output to the indicateed controller.

Show Flow-entry Commands

Show configured static flow-entries

Command Mode: login mode

Command Syntax: `show flow-entry [<name>] [switch <switch>]`

Show Global-config Commands

Show high availability configuration

Command Mode: login mode

Command Syntax: `show ha [details]`

Next Keyword Descriptions:

- details:
  The details keyword requests a more verbose version of various show commands.

Show Logging Commands

Show various controller logs

Command Mode: login mode

Command Syntax: `show logging [controller {all | <controller-node>}] <log-name>`

Next Keyword Descriptions:

- log-name:

Show Running-config Commands

Show the current active configuration

Command Mode: login mode

Command Syntax: `show running-config [feature | controller-node [<word>] | switch [<word>] | host [<word>] | onv [<word>] | tenant [<word>] | static-arp | address-space [<word>] | forwarding | snmp | tacacs | tag [<word>] | tech-support | topology]`

Command Description:
The 'show running-config' command displays the complete configuration for the controller. When the controller is operating with HA enabled, 'show running-config' displays the configuraion of the HA cluster.

The running-config is intended to show the configuration necessary to update a default configuration to the current configuration. The running-config then displays deviations from the default configuration; configuration whice arises from the default configuration will not be shown in the running-config

While operating with HA enabled, a replay of the running config will not cause a slave to be (re)configured into HA-enabled if it is not already operating as HA-enabled.

Next Keyword Descriptions:

- onv:
  The 'onv' keyword filters the running-config output to only display the onv configuration.
- address-space:
  The 'address-space' keyword filters the running-config output to only display the address-space related configuration.
- controller-node:
  The 'controller-node' keyword filter the running config to only display controller-node configuration. This includes descriptions for the controller node interfaces, and firewall rules.
- snmp:
  The 'snmp' keyword filters the running-config output to only display the snmp configuration.
- switch:
  The 'switch' keyword filters the running-config output to only display switch related configuration.
- feature:
  The 'feature' keyword filters the running-config output to display the features enabled or disabled which differ from the default features.
- host:
  The 'host' keyword filters the running-config output to only display the host related configuration. This includes host aliases, and any host security associations.
- tag:
  The 'tag' keyword filters the running-config to only display the tag submode related configuration. Uses of the tag within onv, address-space, or other submodes is not included.
- tacacs:

- tech-support:
- forwarding:
- static-arp:
- tenant:
- topology:

Command Examples:

```
show running-config
```

```
    Show the complete running-config
```

```
show running-config swtich 00:00:00:00:00:73:28:03
```

```
    Show the running cofig details for a particular switch
```

```
    !
```

```
    switch 00:00:00:00:00:73:28:03
```

```
      switch-alias ntgr-7328-3
```

```

```

```

```

Show Switch-cluster Commands

Show groups of interconnected openflow switches

Command Mode: login mode

Command Syntax: `show switch-cluster`

Show Tech-support Commands

Show tech-support, collect output of various commands

Command Mode: login mode

Command Syntax: `show tech-support`

Show This Commands

Show the object associated with the current submode

Command Mode: config- mode

Command Syntax: `show this`

Show Host Commands

Show host details based on query

Command Mode: login mode

Command Syntax: `show host {<host mac or alias> | address-space <address space> | ip-address <ip address> | switch <switch dpid or alias> | all} [by last-seen] [{details | brief}]`

Command Description:
This command variation allows filtering based on the provided mac address. In addition, the hosts can be selected either by the switch of their attachment point, or the host's associated ip address

Next Keyword Descriptions:

- all:
  The 'all' keyword is a positional parameter replacement which allows additional parameters to be included. 'all' may not be used as a host alias.
- address-space <address-space>:
  This positional parameter allows filtering on the address space associated with the hosts.
- mac:
  This positional parameter allows filtering on the mac address.
- last-seen:
  Order the output by the last seen time of the attachment point
- brief:
  The brief keywork requests a less verbose version of output for various show commands.
- switch <dpid>:
  The 'switch <dpid>' keyword pair can be used to restrict the hosts displayed by the identified <dpid>. The hosts displayed will all have attachement points associated with the requested <dpid>.

If completion is requested after the 'switch' keyword, the list of switches displayed are the switches associated with the active attachment points of the known hosts.

- details:
  The details keyword requests a more verbose version of various show commands.
- ip-address <ipv4>:
  The 'ip-address <ip-address>' keyword pair can be used to restrict the host's displayed by the ip-address associaited with the currently known hosts.

If completion is requested after the ip-address keyword, the list of ip-addresses displayed are the ip-addresses currently associated with the known hosts.

- by:

Show various host related details by query

Command Mode: login mode

Command Syntax: `show host {<host mac or alias> | ip-address <ip address> | switch <switch dpid or alias> | all} {attachment-point [by {host-last-seen | last-seen}] [{details | brief}] | ip-address [by {host-last-seen | last-seen}] [{details | brief}] | alias}`

Command Description:
This 'show host' variant allows the selection of particular related host details, for example the attachment point or the ip address.

The variant exists to more completely list the host's associated attributes. In the more basic version of the command, these columns may have abbreviated information (displaying only one ip address, for example), to provide a more concise display.

Next Keyword Descriptions:

- attachment-point:

- all:
  The 'all' keyword is a positional parameter replacement which allows additional parameters to be included. 'all' may not be used as a host alias.
- switch <dpid>:
  The 'switch <dpid>' keyword pair can be used to restrict the hosts displayed by the identified <dpid>. The hosts displayed will all have attachement points associated with the requested <dpid>.

If completion is requested after the 'switch' keyword, the list of switches displayed are the switches associated with the active attachment points of the known hosts.

- host-last-seen:
  Order the output by the time the host was last seen by the controller
- brief:
  The brief keywork requests a less verbose version of output for various show commands.
- mac:
  This positional parameter allows filtering on the mac address.
- alias:

- details:
  The details keyword requests a more verbose version of various show commands.
- ip-address:

- last-seen:
  Order the output by the last seen time of the attachment point
- ip-address <ipv4>:
  The 'ip-address <ip-address>' keyword pair can be used to restrict the host's displayed by the ip-address associaited with the currently known hosts.

If completion is requested after the ip-address keyword, the list of ip-addresses displayed are the ip-addresses currently associated with the known hosts.

- by:

Show Link Commands

Show links, controller managed switch to switch interfaces

Command Mode: login mode

Command Syntax: `show link`

Show Snmp-server-config Commands

Show snmp configuration

Command Mode: login mode

Command Syntax: `show snmp`

Command Description:
Display operational parameters of this device's SNMP support.

Show the configurable parameters for the SNMP MIBs, and show the status of the network firewall (which might be blocking some SNMP requests)

Command Examples:

```
show snmp
```

```
  Display SNMP operational state.
```

```

```

Show Switches Commands

Show switch summary

Command Mode: login mode

Command Syntax: `show switch`

Command Description:
The show switch commands displays operational state for switches currently connected to the controller.

Next Keyword Descriptions:

- switch:

Show realtime stats for switch

Command Mode: login mode

Command Syntax: `show switch {<switch dpid or alias> | all} {{aggregate | flow | port | table | desc | queue} [{details | brief}] | features [{details | brief}]}`

Next Keyword Descriptions:

- all:
  The 'all' token is used in place of an identifier's value to request every item associated with the command. The 'all' token is reserved, identifiers may not use the 'all' token as their name, no switch-alias or host-alias may be called 'all'
- features:

- flow:
  The flow selection request the controller to query the selected switch for all the flows currently active.
- brief:
  The brief keywork requests a less verbose version of output for various show commands.
- queue:
  The queue option is a request for the controller to query the indicated switch to acquire the switch queue details The returned value is an association between the interface names, the queue numbers, and some statitics for each of the queues.
- details:
  The details keyword requests a more verbose version of various show commands.
- aggregate:
  The aggregate option displays aggregated flow statistics. It is a request for the controller to activly query the indicated switch to acquire and display the results.
- table:
  The table option is a request for the controller to query the indicated switch, for the switch's table details. The returned values the names of the tables, the size of the tables, and some usage statistics
- port:
  The port option is a request for the controller to send to the indicated switch, the reply is the openflow protocol port reply, which lists all the interfaces and provides tx and rx statistics.
- desc:
  The desc option is a request for the controller to query the indicated switch. to acquire the switch description. The returned values include the switch model and version, the switch vendor, serial number, and software version currently running.

Show stats for selected switch

Command Mode: login mode

Command Syntax: `show switch <switch dpid or alias> stats`

Show statistics for a given switch

Command Mode: login mode

Command Syntax: `show switch <switch dpid or alias> stats {OFActiveFlow | OFFlowMod | OFPacketIn} [start-time <start-time>] [end-time <end-time>] [duration <duration>] [sample-interval <sample-interval>] [sample-count <sample-count>] [sample-window <sample-window>] [data-format {value | rate}] [display {latest-value | graph | table}]`

Show interfaces for selected switch

Command Mode: login mode

Command Syntax: `show switch {<switch dpid or alias> | all} {interfaces [stats] [alias] | onv | alias} [{details | brief}]`

Next Keyword Descriptions:

- alias:
  Interfaces may have aliases assigned to them, this keyword diesplays any aliases associated with the switch
  This selection displays the alias associated with the requested switch (or all the swtiches when all is selected)
- interfaces:
  The 'interfaces' keyword lists the interfaces on the switch. This information is posed by the switch to the controller when the switch first connects, and then updated by the switch if new interfaces are created or deleted.

Show switch tcpdump via controller

Command Mode: login mode

Command Syntax: `show switch <switch dpid or alias> trace [oneline] [single-session] [echo-reply] [echo-request] [features-rep] [flow-mod] [flow-removed] [get-config-rep] [hello] [packet-in] [packet-out] [port-status] [set-config] [stats-reply] [stats-request] [detail]`

Show switch details via query

Command Mode: login mode

Command Syntax: `show switch {<switch dpid or alias> | all} [by {ip-address | connect-time}] [{details | brief}]`

Next Keyword Descriptions:

- brief:
  The brief keyword requests a less verbose version for the output of various show commands
- by:
  The 'by' keyword describes a sort-by token, The sort selection is the next keyword
- details:
  The details keyword requests a more verbose version of the output format.

Show Switch-interfaces Commands

Show interfaces for switch associated with current submode

Command Mode: config-switch mode

Command Syntax: `show interfaces`

Show Switch Commands

Show tunnels for all switches

Command Mode: login mode

Command Syntax: `show tunnel`

Show tunnels for selected switches

Command Mode: login mode

Command Syntax: `show tunnel {all | <dpid>}`

Show Tacacs-plus-config Commands

Show tacacs operational state

Command Mode: login mode

Command Syntax: `show tacacs`

Command Description:
Show TACACS+ aaa operational state.

Print out the global aaa sources and services, as well as any configured TACACS+ servers and parameters.

Command Examples:

```
show tacacs
```

```
  Display the TACACS+ aaa operational state.
```

```

```

Show Tag Commands

Show configured tags

Command Mode: login mode

Command Syntax: `show tag [namespace <namespace>] [name <name>] [value <value>]`

Command Description:
The show tag command lists all the tags configured. Each tag's namespace, name and value are shown. Also shown is whether this tag is persisted or not, meaning, whether this tag was created internally by the controller itself or the user created this.

Show Tech-support-config Commands

Show tech-support configuration

Command Mode: login mode

Command Syntax: `show tech-support-config [{cli | shell}] [cmd <cmd>]`

Command Description:
This commands shows customized commands shown in 'show tech-support' output.

Command Examples:

```
Meis-MacBook-Pro.local(config)# tech-support-config shell date
```

```
Meis-MacBook-Pro.local(config)# show tech-support-config
```

```
Type of command Command name
```

```
---------------|------------
```

```
shell           date
```

```
Meis-MacBook-Pro.local(config)#
```

```

```

Show Tenant Commands

Show defined tenants

Command Mode: login mode

Command Syntax: `show tenant`

Show specific tenant, identified by name

Command Mode: login mode

Command Syntax: `show tenant <tenant-id> {onv [<onv-id> [interfaces | mac-address-table | interface-rules | access-lists | running-config | switch | flow [{brief | full-detail | details | summary}]]] | router [<virtualrouter-id> [ip-address-pool | route | interfaces | gateway-pools | gw-address-pool]] | running-config | {details | brief}}`

Command Description:
This 'show tenant <tenant-name>' command is used to identify a specific tenant, to describe particular configured details. 'show tenant' command provides display for every defined tenant.

Next Keyword Descriptions:

- onv:
  The 'onv' keyword requests the display of the configured onvs for the named tenant.
- access-lists:
  The 'access-list' keyword requests display of the configured access-lists associated with the onv. The output includes not only the access-list, but a brief descripion of the acl rules associated with the access-list.
- full-detail:

For a few show commands, the 'full-detail' is a request to display more information than the 'defails' keyword.

- route:
  The 'route' keyword requests the display of the configured routing rule for the named virtual router.
- gw-address-pool:
  The 'gw-address-pool' keyword requests the display of the configured gateway ip addresses for all the gateway pools of the named virtual router.

- flow:
  The 'flow' keyword requests the display of all openflow flow-match entries inserted into various switches to implement the onv isolation.
- brief:
  The brief keywork requests a less verbose version of output for various show commands.
- summary:
  A more terse output format.
- interface-rules:
  The 'interface-rules' keyword requests the display of the configured interface-rules for the named onv. The onv interface-rules describe the membership rules associated with a onv.
- gateway-pools:
  The 'gateway-pools' keyword requests the display of the configured next hop gateway pools for the named virtual router.
- details:
  The details keyword requests a more verbose version of various show commands.
- router:
  The 'router' keyword requests the display of the configured virtual router for the named or current tenant.
- ip-address-pool:
  The 'ip-address-pool' keyword requests the display of the configured ip address/subnet for all the interfaces of the named virtual router.
- interfaces:
  The 'interfaces' keyword requests the display of the configured interfaces for the named virtual router.

Show specific tenant, identified by name

Command Mode: login mode

Command Syntax: `show tenant {<tenant-id> | all}`

Command Description:

This 'show tenant <tenant-name>' command is used to identify a specific tenant, to describe particular configured details. 'show tenant' command provides display for every defined tenant.

Command Examples:

```
localhost# show tenant all
```

```
      # Tenant ID Active Description Router ID
```

```
      -|---------|------|-----------|---------
```

```
       1 A         True               vrA
```

```
       2 default   True
```

```
       3 red       True
```

```
       4 system    True               vrsystem
```

```
localhost# show tenant A
```

```
       # Tenant ID Active Description Router ID
```

```
       -|---------|------|-----------|---------
```

```
1 A         True               vrA
```

```
```

Show Dvs Commands

Show vcenter dvs details

Command Mode: login mode

Command Syntax: `show dvs [vcenter <vcenter>] [datacenter <datacenter>] [dvs <dvs>]`

Show Dvs-port-group Commands

Show vcenter dvs port-group details

Command Mode: login mode

Command Syntax: `show dvs-port-group [vcenter <vcenter>] [datacenter <datacenter>] [dvs <dvs>] [portgroup <portgroup>]`

Show Vcenter Commands

Show vcenter configurations

Command Mode: login mode

Command Syntax: `show vcenter`

Show vcenter description by name

Command Mode: login mode

Command Syntax: `show vcenter {<vcenter-name> | all} [dvs | dvs-port-group | status | details]`

Next Keyword Descriptions:

- status:
- dvs-port-group:
- dvs:

Show vcenter operational status by name

Command Mode: login mode

Command Syntax: `show vcenter <vcenter-name> dvs <dvs-name> status`

Show Static-arp Commands

Show all configured static arps

Command Mode: login mode

Command Syntax: `show arp`

Command Description:
The 'show arp' command shows the configured static arp table.

Command Examples:

```
localhost# show arp
```

```
# Ip       Mac
```

```
-|--------|-----------------
```

```
1 10.0.0.1 11:22:33:44:55:66
```

```

```

Show Virtualrouter Commands

Show specific virtual router, identified by name

Command Mode: config-tenant mode

Command Syntax: `show router`

Command Description:
The 'router' keyword requests the display of the configured virtual router for the named or current tenant.

Show specific virtual router, identified by name

Command Mode: config-tenant mode

Command Syntax: `show router <virtualrouter-id> [ip-address-pool | route | interfaces [<vriname> [ip-address-pool]] | gateway-pools [<vrgwname> [gw-address-pool]]]`

Next Keyword Descriptions:

- gateway-pools:
  The 'gateway-pools' keyword requests the display of the configured next hop gateway pools for the named virtual router.
- route:
  The 'route' keyword requests the display of the configured routing rule for the named virtual router.
- ip-address-pool:
  The 'ip-address-pool' keyword requests the display of the configured ip address/subnet for all the interfaces of the named virtual router.
- interfaces:
  The 'interfaces' keyword requests the display of the configured interfaces for the named virtual router.

- This page was last modified on 17 April 2013, at 18:16.