Naddod N9000 Series Switches User Guide

AI Data Center Switch

NADDOD Pte.Ltd.

Contents

Installation of SONiC Software 1
Installation of SONiC Software 1
Managing SONiC Configuration Files 2
Zero Touch Provisioning (ZTP) 6
Displaying Feature States 8
New Default Configuration 13
REST Server Configuration 15
Configure Management Interface 20
System Management Configuration 23
Specifying a Device Name 23
CPU and Memory Utilization 23
Docker Warm Restart 26
Initiating a Warm Reboot 27
Using Container Auto-Restart 27
Network Time Protocol (NTP) 29
System Log (syslog) 31
Configuration Reloading 35
Management Framework (>>> programming) 37
MIB & Trap Support List 40
Hardware Reset 42
Protocol Down 43
Simple Network Management Protocol (SNMP) 45
Telnet Server 46
Users and Passwords 48
Interface Configuration 51
Shutting Down and Starting Up Interfaces 51
Link Aggregation Control Protocol (LACP) 54
Configure Interface Speed, Breakout Mode and Auto-Negotiation 57
Link-Training 61
Configure Laser Frequency and TX Power for ZR Transceiver 64
Auto-Negotiation 69
IP Configuration 71
IPv4/IPv6 Interfaces and Routes 71
Neighbor Discovery Protocol (NDP) 73
Static Anycast Gateway (SAG) 75
Layer 2 77
ARP 77
Configuring DHCP Relay 80
Configuring interface TPID 82
Control Plane Policing (CoPP) 83
Creating VLANs 85
DHCP Snooping User Guide 88
Forwarding Database (FDB) 90
MAC Aging Time 92
IGMP Snooping User Guide 92
Link Aggregation (LAG) 95
Link Layer Discovery Protocol (LLDP) 95
Multi-Chassis Link Aggregation Group (MC-LAG) 96
PVST and STP 102
XSTP (STP, RSTP, MSTP) 103
Dynamic Load Balancing (DLB) 113
Layer 3 115
SONiC Routing Stack 115

Border Gateway Protocol (BGP) 117
Bidirectional Forwarding Detection (BFD) 122
Virtual Routing and Forwarding (VRF) 123
Management VRF Configuration 124
Leaking VRF Routes 125
Displaying ARP Entries 126
In Band Management VRF 127
BGP Unnumbered 128
BGP Gracefaul Restart 129
Virtual Router Redundancy Protocol (VRRP) 132
Network Monitoring 136
Critical Resource Monitoring (CRM) 136
Flow Sampling (sFlow) 139
Monitor Link 140
Configuring Drop Counters 141
Setting the Watermark Telemetry Interval 143
Inband Network Telemetry (INT) 143
Mirror On Drop (MOD) 145
Counter Space Design 147
QoS Configuration 149
Class of Service (CoS) 149
QoS Marking 152
Explicit Congestion Notification (ECN) 155
Dynamic Explicit Congestion Notification（DECN） 157
Priority Flow Control (PFC) 159
Starting the PFC Watchdog 166
Displaying Queue and Priority-Group Counters 169
Queue Scheduling 171
Weighted Random Early Detection (WRED) 173
RDMA over Converged Ethernet (RoCE) 175
Port and Queue Shaping 179
Port Rate Limit (Traffic Policing) 181
Shared Buffer Configuration 182
DiffServ 185
Security 191
AAA and TACACS+ 191
Authorization & Accounting 194
Configuring TACACS+ Server Parameters 200
Configuring RADIUS Server Parameters 200
Access Control Lists (ACL) 201
Configuring a Control Plane ACL 208
ACL Rule Configuration File 210
ACL Rule Configuration File 210
PINS 218
Introduction 218
Setup the P4 Client on Your Local Environment 219
Check if the Docker Image of P4runtime-sh is Available in Your Local Repository 219
Start the Connection with P4 Runtime Sever and Push a Forwarding Pipeline Configuration with the Shell 220
An Example to Set a Route Entry on P4 Target through P4runtime Shell 220
Troubleshooting & Limitation 224
Appendix A: Compiling P4 Source Code 224

Installation of SONiC Software

Installation of SONiC Software

SONiC is a network operating system based on the SONiC sub-project (OCP Networking Project Group) and runs on switches. The SONiC software needs to be loaded onto the supported switch platforms using ONIE, as described in Install, Upgrade, and Remove SONiC.

After installing SONiC on a switch, you can connect to the console port of the device and configure it through one of the following methods:

- command Line Interface
- config_db.json configuration file
- minigraph.xml configuration file

See the Managing SONiC Configuration Files section for more information on JSON and XML configuration files.

SONiC also includes a command-line tool for installing the SONiC software as an alternative image in the partition. When a switch is already running a SONiC, you can use the sonic-installer install command to install another new image using the CLI.

Example

```
admin@sonic:~$ sudo sonic_installer install http://192.168.1.3/download/SONiC-OS-Naddod-SONiC_20200827_110345_ec201911_2020-aug_enhanced_178.bin
New image will be installed, continue? [y/N]: y
Downloading image...
...100%, 480 MB, 3357 KB/s, 146 seconds passed
Command: /tmp/sonic_image
Verifying image checksum ... OK.
Preparing image archive ... OK.
ONIE Installer: platform: XXXX
onie_platform:
Installing SONiC in SONiC
Installing SONiC to /host/image-xxxx
Directory /host/image-xxxx/ already exists. Cleaning up...
Archive: fs.zip
```

```
creating: /host/image-xxxx/boot/
inflating: /host/image-xxxx/boot/vmlinuz-3.16.0-4-amd64
inflating: /host/image-xxxx/boot/config-3.16.0-4-amd64
inflating: /host/image-xxxx/boot/System.map-3.16.0-4-amd64
inflating: /host/image-xxxx/boot/initrd.img-3.16.0-4-amd64
creating: /host/image-xxxx/platform/
extracting: /host/image-xxxx/platform/firsttime
inflating: /host/image-xxxx/fs.squashfs
inflating: /host/image-xxxx/dockerfs.tar.gz
Log file system already exists. Size: 4096MB
Installed SONiC base image SONiC-OS successfully
Command: cp /etc/sonic/minigraph.xml /host/
Command: grub-set-default --boot-directory=/host 0
Done
```

You can set the new image to boot by default using the sonic_installer set_default command, which will load the image at the next reboot and all subsequent reboots.

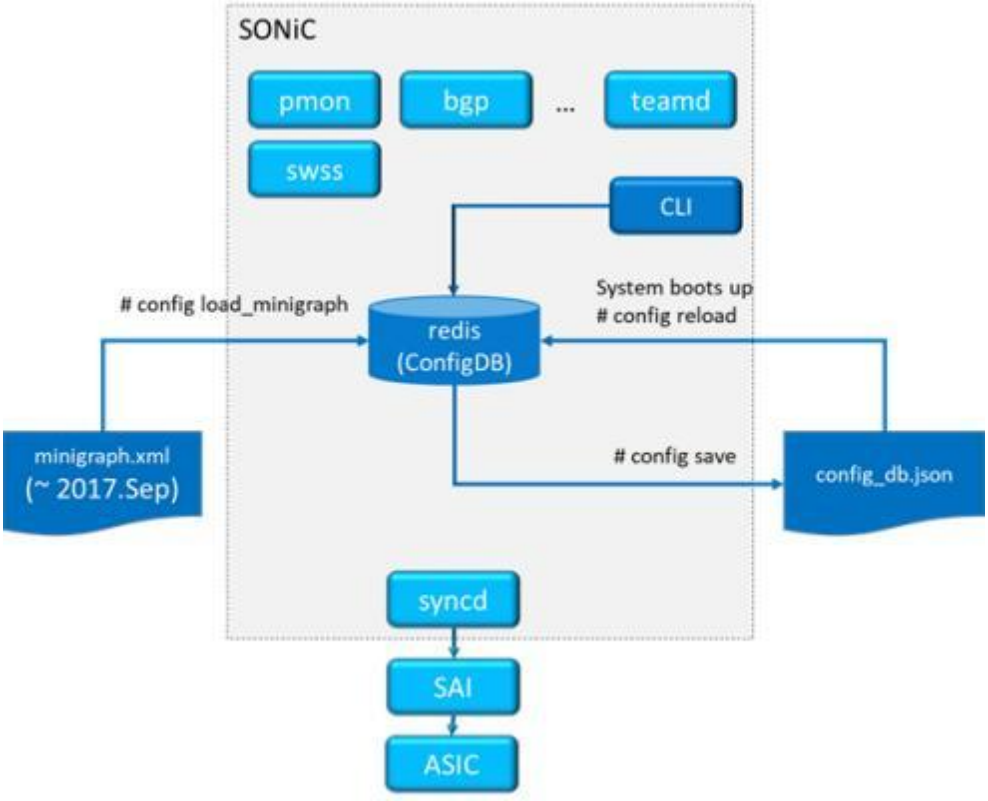
Example

```
admin@sonic:~$ sudo sonic_installer set_default SONiC-OS-HEAD.XXXX
```

Other commands related to software installation and management are documented in section of the CLI guide.

Managing SONiC Configuration Files

The SONiC configuration is dependent on two primary databases. A Redis database that can be considered as the switch “running configuration,” and a config_db.json file that functions like a “startup configuration” file.



Redis is an open source in-memory data structure store that maintains configuration changes from the various software features. The config_db.json file is text-readable in the JavaScript Object Notation (JSON) format. The config_db.json file configuration is imported into the Redis database at startup, but the Redis configuration is not written back to the config_db.json file unless a save command is manually executed from the CLI.

Note：

SONiC also supports loading a configuration from an XML formatted file called minigraph.xml stored in the Linux file system /etc/sonic/ location. The minigraph.xml file is a legacy method used before September 2017.

For further information on SONiC configuration files, see the section Configuration File Management in the CLI command reference.

- Loading and Saving a JSON Configuration File
- Loading the SONiC Configuration from a minigraph.xml File
- Reloading the SONiC Configuration
- Loading the Management Interface Configuration
- Displaying the Running Configuration

Loading and Saving a JSON Configuration File

You can load a switch configuration from a JSON file using the config load command. Either the default /etc/sonic/config_db.json file is loaded, or another file name can be specified.

The configuration in the file is loaded into the running Redis database.

This command does not flush out the Redis database before loading the new configuration, the input either modifies or is added to the configuration.

The following example loads the configuration from the default config_db.json file.

Example
admin@sonic:~$ sudo config load Load config from the file /etc/sonic/config_db.json? [y/N]: y Running command: /usr/local/bin/sonic-cfggen -j /etc/sonic/config_db.json --write-to-db

Use the config save command from the CLI to save the running Redis database (the running configuration) to the default /etc/sonic /config_db.json file or another user-specified file. This will save the current configuration after a reboot.

Example
admin@sonic:~$ sudo config save -y /etc/sonic/config2.json

Loading the SONiC Configuration from a minigraph.xml File

Use the config load_minigraph command to load the configuration from the /etc/sonic/minigraph.xml file. The XML file must be copied to the /etc/sonic/ location before executing the command. This command restarts various services running on the device and takes some time to complete.

Example
admin@sonic:~$ sudo config load_minigraph Reload config from minigraph? [y/N]: y Running command: /usr/local/bin/sonic-cfggen -j /etc/sonic/config_db.json --write-to-db

Reloading the SONiC Configuration

Along with loading a configuration file into the running Redis database, the config reload command also flushes out the current configuration before importing a new configuration. This command loads the default /etc/sonic/config_db.json file or another specified JSON-formatted file.

The config reload command takes some time to complete since it stops and restarts all services running on the device. In addition, this command can disrupt management connections to the device and they may require reconfiguration. In general, executing this command from the console port is recommended.

Example

```
admin@sonic:~$ sudo config reload
Clear current config and reload config from the file /etc/sonic/config_db.json? [y/N]: y
Running command: systemctl stop dhcp_relay
Running command: systemctl stop swss
Running command: systemctl stop snmp
Warning: Stopping snmp.service, but it can still be activated by: snmp.timer
Running command: systemctl stop lldp
Running command: systemctl stop pmon
Running command: systemctl stop bgp
Running command: systemctl stop teamd
Running command: /usr/local/bin/sonic-cfggen -H -k Force10-Z9100-C32 --write-to-db
Running command: /usr/local/bin/sonic-cfggen -j /etc/sonic/config_db.json --write-to-db
Running command: systemctl restart hostname-config
Running command: systemctl restart interfaces-config
Timeout, server 10.11.162.42 not responding.
```

Loading the Management Interface Configuration

To reconfigure a device's management interface and host name, you can use the config load_mgmt_config command. This command loads the management configuration from the /etc/sonic/device_desc.xml file or another specified file.

Example

```
admin@sonic:~$ sudo config load_mgmt_config
Reload config from minigraph? [y/N]: y
```

Running command: /usr/local/bin/sonic-cfggen -M /etc/sonic/device_desc.xml --write-to-db

Note：

device_desc.xml example file can be as below

```
<DeviceMiniGraph xmlns="Microsoft.Search.Autopilot.Evolution"
xmlns:i="http://www.w3.org/2001/XMLSchema-instance">
  <Hostname>SONIC-X</Hostname>
  <HwSku>NADDOD-HW-Y</HwSku>
  <Address xmlns:a="Microsoft.Search.Autopilot.NetMux">
    <a:IPPrefix>127.0.0.1/16</a:IPPrefix>
  </Address>
  <ManagementAddress xmlns:a="Microsoft.Search.Autopilot.NetMux">
    <a:IPPrefix>192.168.200.10/24</a:IPPrefix>
  </ManagementAddress>
</DeviceMiniGraph>
```

Displaying the Running Configuration

Use the show running configuration all command to display the current running configuration for all modules.

Example
admin@sonic:~$ show running configuration all

Alternatively, you can also specify one of the following modules: bgp, interfaces, ntp, snmp, acl, ports, or syslog along with show running configuration command to display the respective module details. See the Startup & Running Configuration section in the CLI reference guide for more details.

Zero Touch Provisioning (ZTP)

SONiC supports Zero Touch Provisioning (ZTP), which enables a large number of switches to be configured quickly and automatically.ZTP loads a JSON file at boot up that is essentially a script that performs the configuration steps needed to set up a switch on the network.

The ZTP JSON file can be installed on a device or on a provisioning server. When the ZTP JSON file is on a server, DHCP Option 67 can be used to obtain the URL of ZTP JSON file and enable it to be downloaded.

By default, the ZTP service is enabled in SONiC. The service can be disabled from the CLI using the config ztp disable command. Once the ZTP service has been disabled, it must be manually enabled again using the config ztp enable command.

Example

```
root@sonic:/home/admin# config ztp disable
Active ZTP session will be stopped and disabled, continue? [y/N]: y
Running command: ztp disable –y
root@sonic:/home/admin# config ztp enable
Running command: ztp enable
```

In addition, the config ztp run command can be used to start a new ZTP session. This command deletes the existing /etc/sonic /config_db.json file and erases the previous ZTP session data before loading a new ZTP JSON file.

Example

```
root@sonic:/home/admin# config ztp run
ZTP will be restarted. You may lose switch data and connectivity, continue? [y/N]: y
Running command: ztp run -y
Go Back To [<span style="color: #0000ff"><span style="text-decoration: underline; ">Beginning of the document</span></span> or [<span style="color: #0000ff"><span style="text-decoration: underline; ">Beginning of this section</span></span>
```

To check the status of the current ZTP session, use the show z tp status command. This command displays information related to all configuration sections as defined in the switch provisioning information discovered in a particular ZTP session.

For more information see the ZTPsection in the CLI reference guide.

Example

```
root@B1-SP1-7712:/home/admin# show ztp status
ZTP Admin Mode : True
ZTP Service : Inactive
ZTP Status : SUCCESS
ZTP Source : dhcp-opt67 (eth0)
Runtime : 05m 31s
Timestamp : 2019-09-11 19:12:24 UTC
ZTP Service is not running
```

```
01-configdb-json: SUCCESS
02-connectivity-check: SUCCESS
Use the verbose option to display more detailed information.
root@B1-SP1-7712:/home/admin# show ztp status --verbose
Command: ztp status --verbose
=======================================
ZTP
=======================================
ZTP Admin Mode : True
ZTP Service : Inactive
ZTP Status : SUCCESS
ZTP Source : dhcp-opt67 (eth0)
Runtime : 05m 31s
Timestamp : 2019-09-11 19:12:16 UTC
ZTP JSON Version : 1.0
ZTP Service is not running


---------------------------------------
  01-configdb-json
---------------------------------------
  Status       : SUCCESS
  Runtime      : 02m 48s
  Timestamp : 2019-09-11   19:11:55   UTC
  Exit Code : 0
  Ignore Result : False
---------------------------------------
  02-connectivity-check
---------------------------------------
Status : SUCCESS
Runtime : 04s
Timestamp : 2019-09-11 19:12:16 UTC
Exit Code : 0
Ignore Result : False
```

Displaying Feature States

- Introduction
- Feature List

Introduction

SONiC includes a capability in which a feature state can be enabled or disabled, which will make the corresponding feature docker container start or stop. In SONiC, the Feature List describes the default service container states, the default admin state features, and the commands to check its states.

Note：

After the 202006 branch, a new set of commands were introduced to check the container states. See - https://github.com/Azure/sonic-utilities/blob/master/doc/Command-Reference.md#feature for more information.

Until the show feature command is implemented, you can use the Feature List table to check the service states of SONiC features. Refer to Feature List to find the command's service state and admin's (configuration) state in the table.

For example, for the LACP feature, the default service state of the container is Enable , and the default admin state is Disable . You can use the below commands to check the states.

command to show service state
System ctl status teamd.service docker ps | grep teamd docker exec -i teamd ps -x

These commands can be issued under the SONiC Linux shell.

The system ctl command can display teamd (LACP service container) status related information, and the other two docker commands are an alternative method to check the container states.

Use the docker ps command to check the running state of the teamd container. For experienced users, who want more information on the docker status, the docker exec -i teamd ps -x command can display the processes running in the teamd container.

command to show admin state
show interfaces port channel

The show interfaces command is the SONiC CLI command that you can use to check the admin or operation states of the LACP feature.

Most of the commands used to check the admin states are SONiC CLI commands. In special cases, you may need to use other methodologies (e.g., other Linux shell commands or checking the config_db.json file) to check the feature states. For those cases, there will be a remark in the description.

Feature List

Category	Feature Name	Default Service Container	Default Admin State (Enable/Disable)
L2	LAG (LACP)	Enable systemctl status teamd.service docker ps | grep teamd docker exec -i teamd ps -x	Disable admin@sonic~$ show interfaces port channel
	LLDP	Enable systemctl status lldp.service docker ps | grep lldp docker exec -i lldp ps -x	Enable admin@sonic~$ show lldp table admin@sonic~$ show lldp neighbors
	MCLAG	Enable systemctl status iccpd.service docker ps | grep iccpd docker exec -i iccpd ps -x	Disable admin@sonic~$ show mc lag brief
	QinQ	Enable systemctl status swss.service	Disable Check config_db.json for VLAN stacking configuration.
	Storm Control	Enable systemctl status swss.service	Disable admin@sonic~$ show storm-control all
	VLAN	Enable systemctl status swss.service docker exec -i swss ps -x | grep v lanmgrd	Disable admin@sonic~$ show vlan brief
	VLAN Translation	Enable systemctl status swss.service	Disable Check config_db.json for VLAN stacking configuration.
L3	BGP	Enable systemctl status bgp.service docker ps | grep bgp docker exec -i bgp ps -x	Disable admin@sonic~$ vtysh Hello, this is FRRouting (version 7.2.1-sonic). Copyright 1996-2005 Kunihiro Ishiguro, et al. sonic# show ip bgp summary % BGP instance not found
	EVPN/VxLAN	Enable systemctl status bgp.service show vxlan interface	Disable Refer to EVPN HLD

	IS-IS	Enable systemctl status bgp.service docker ps | grep bgp	Disable admin@as7816-64x:~$ vtysh Hello, this is FRRouting (version 7.2.1-sonic). Copyright 1996-2005 Kunihiro Ishiguro, et al. as7816-64x# show isis summary (Number of area 0 means there is no active ISIS area)
	OSPF v2	Enable systemctl status bgp.service docker ps | grep bgp	Disable admin@sonic:~$ vtysh Hello, this is FRRouting (version 7.2.1-sonic). Copyright 1996-2005 Kunihiro Ishiguro, et al. as7816-64x# show ip ospf % OSPF instance not found
	Proxy ARP	Enable systemctl status swss.service	Disable root@sonic:~# cat /proc/sys/net/ipv4/conf/[interface] /proxy_arp_pvlan ('0' is disabled, '1' is enabled) This command is issued under the Linux shell.
	SAG (Static Anycast Gateway)	Enable systemctl status swss.service	Disable admin@sonic:~$ show sag
	VRF	Enable systemctl status swss.service	Disable admin@sonic:~$ show vrf
QoS	CoS	Enable systemctl status swss.service	Enable CoS mapping configuration uses the chip default configuration. Check config_db.json for DOT1P_TO_TC_MAP configuration if you change the configuration.
	DSCP	Enable systemctl status swss.service	Enable DSCP mapping configuration uses the chip default configuration. Check config_db.json for DSCP_TO_TC_MAP configuration if the user changes the configuration.
	ECN-WRED	Enable systemctl status swss.service	Disable admin@sonic:~$ show ecn
	PFC (Priority Flow Control)/Asym	Enable systemctl status swss.service	Disable admin@sonic:~$ show pfc priority

	PFC		
	PFC Watchdog (WD)	Enable systemctl status swss.service	Disable admin@sonic:~$ show pfcwd config
	PFC Watermark (WM)	Enable systemctl status swss.service	Enable admin@sonic:~# show counterpoll show
	Port Rate Limit	Enable systemctl status swss.service	Disable Check config_db.json for SCHEDULER and PORT_QOS_MAP configuration.
Security	AAA	Enable systemctl status swss.service	Enable admin@sonic:~# show aaa
	CoPP	Enable systemctl status swss.service	Enable docker exec swss cat /etc/swss/config.d/00-copp.config.json This command is issued under the Linux shell.
	IPv4 ACL	Enable systemctl status swss.service	Disable admin@sonic:~# show acl table
	IPv6 ACL	Enable systemctl status swss.service	Disable admin@sonic:~# show acl table
	LDAP	Enable systemctl status swss.service	Disable admin@sonic:~# show aaa
	TACACS+	Enable systemctl status swss.service	Disable. admin@sonic:~# show aaa
Management & Monitor	Configurable drop counter	Enable systemctl status swss.service	Disable admin@sonic:~# show dropcounters configuration
	Critical Resource Monitoring (CRM)	Enable systemctl status swss.service	Enable admin@sonic:~# crm show resources
	DHCP Relay Agent	Enable systemctl status dhcp_relay . service docker exec -i dhcp_relay ps -x	Disable admin@sonic:~# show vlan brief Check DHCP Helper Address Field
	DNS Client	Enable systemctl status systemd - resolved.service	Disable
	Management Framework	Enable systemctl status mgmt - framework.service docker ps | grep mgmt-framework	Enable

		docker exec -i mgmt-framework ps -x	
	Mgmt VRF	Enable systemctl status swss.service	Disable admin@sonic:~# show mgmt-vrf
	Mirror	Enable systemctl status swss.service	Disable admin@sonic:~# show mirror_session
	NTP	Enable systemctl status ntp.service	Enable admin@sonic:~# show ntp
	Platform Monitor	Enable systemctl status pmon.service systemctl status * - platform - monitor.service	Enable admin@sonic:~# show platform syseeprom admin@sonic:~# show platform psustatus admin@sonic:~# show interfaces transceiver
	sFlow	Enable systemctl status sflow.service	Disable admin@sonic:~# show sflow
	SNMP	Enable systemctl status snmp.service	Enable
	Syslog	Enable systemctl status rsyslog . service	Enable
	Telemetry	Enable systemctl status telemetry . service docker ps | grep telemetry	Enable
	ZTP	Enable systemctl status z tp.service	Enable admin@sonic:~# sudo show ztp status

New Default Configuration

- What is "default configuration"?
- Why there is a distribution "default configuration"?
  - FRR configuration
  - REST Server configuration
  - Remove configurations

- How do you reset to "default configuration"?
  - Configuration manipulation
  - Command Line

What is "default configuration"?

When a device boots up for the first time, the default configuration is automatically generated and written to config_db.json , in case the configuration file config_db.json is missing. In this instance, according to the generic procedure, SONiC refers to the minigraph template, platform, and port configuration to generate the "Default Startup Configuration" described in the SONiC User Manual.

Why there is a distribution "default configuration"?

In the distribution, default configuration allows users to easily generate the configuration file as needed rather than adding numerous needless configurations to the file. Traditional switch users can expect only the most basic configuration, such as the port configuration, can be found in the default configuration file, which they can configure as required.

The ACL, IP, and QOS configurations in the SONiC Community "Default Startup Configuration" have increased the effort in the deployment (planning) stage. In response to this feedback, in Enterprise SONiC Distribution by NADDOD , we added an option called "factory" which generates configurations as simply as possible in the absence of a SONiC configuration file (config db.json). Even without the "factory" option, users can still use Community's "Default Startup Configuration" to generate configurations as they want.

To make the file easier to deploy for the users, The following changes are made to the "default configurations", compared to the Community's SONiC "Default Startup Configuration" file.

FRR configuration

The default configuration allows the FRR to use its original configuration and activate ISIS/FRR functionalities, enabling users to fully control the FRR via its VTYSH.

docker_routing_config_mode: This configuration can be set to "unified", or "split", or "separated" as per Community's definition. In default configuration, default changes to "split", to allow the user to fully control the FRR and store the configuration in the device.

frr_mgmt_framework_config: This configuration can be set to "true", or "false" as per Community's definition. However, In default configuration, the default changes to "true " to enable OSPF/ISIS features in the FRR.

The following is an example to show the changes in the configurations of DEVICE_METADATA.

Example


```
"DEVICE_METADATA": {
    "localhost": {
        "buffer_model": "traditional",
        "default_bgp_status": "up",
        "default_pfcwd_status": "disable",
        "docker_routing_config_mode": "split",
        "frr_mgmt_framework_config": "true",
        "hostname": "leaf-3",
        "hwsku": "NADDOD-N9200-64DC",
        "mac": "04:f8:f8:6a:fa:91",
        "platform": "x86_64-naddod_n9200_64dc-r0",
        "type": "LeafRouter"
    }
},
```


REST Server Configuration

In the Community SONiC's "Default Startup Configuration", the REST server does not enable an authentication mechanism. As a result, there will be an increase in risk and security issues. In NADDOD distribution, by default, the REST server enables the authentication to lower the security risk.

Example


```
"REST_SERVER": {
    "default": {
        "client_auth": "user"
    }
},
```


In the 202012 & 202111 versions, it is strongly recommended to enable authentication on the REST server to prevent access from anonymous.

This also applies to users who use existing configuration files on the 202012 & 202111 versions. It is strongly recommended to add this configuration to the existing configuration file to provide stronger protection for your device access.

Remove configurations

Users cannot use certain default Community SONiC configuration settings if the deployment environment is different. To address this and to maintain a cleaner basis for customers to quickly deploy them, these types of configurations are removed from the NADDOD distribution.

In default configuration, the "factory" setup will remove the below configurations.

- DEVICE_METADATA : "bgp_asn"
- LOOPBACK_INTERFACE
- BGP_NEIGHBOR
- DEVICE_NEIGHBOR
- INTERFACE from config_db.json
- DEVICE_METADATA ["localhost"]["bgp_asn"] -

This is only used when docker_routing_config_mode is set to "separated", to allow the user to have full control of FRR. The "bgp_asn" setting is ineffective if the docker_routing_config_mode is set to "split " and will be removed as shown in the below example. This original configuration is used to set AS number for BGP. In NADDOD Distribution, the user should specify the BGP AS number according to the deployment plan.

Example

```
"DEVICE_METADATA": {

"localhost": {

"bgp_asn": "65100",        <-- removed

"hostname": "sonic",

"hwsku": "NADDOD-N9200-64DC",

"mac": "04:f8:f8:6a:fa:91",

"platform": "x86_64-naddod_n9200_64dc-r0",

"type": "LeafRouter"

}

},
```

- LOOPBACK_INTERFACE - Loopback interface configuration should be decided by the user.

Example

```
"LOOPBACK_INTERFACE": {
"Loopback0": {},
"Loopback0| 10.1.0.1/32": {}
},
```

- BGP_NEIGHBOR - The BGP neighbor configuration should be decided by the user according to the deployment plan.

Example

```
"BGP_NEIGHBOR": {
"10.0.0.1": {
"asn": "65200",
"holdtime": "180",
"keepalive": "60",
"local_addr": "10.0.0.0",
"name": "ARISTA01T2",
"nhopself": "0",
"rrclient": "0"
},
"10.0.0.3": {
"asn": "65200",
"holdtime": "180",
"keepalive": "60",
"local_addr": "10.0.0.2",
"name": "ARISTA02T2",
"nhopself": "0",
```

```
"rrclient": "0"
},
"10.0.0.5": {
"asn": "65200",
"holdtime": "180",
"keepalive": "60",
"local_addr": "10.0.0.4",
"name": "ARISTA03T2",
"nhopself": "0",
"rrclient": "0"
},
```


- DEVICE_NEIGHBOR - The Community default configuration on this is empty. Remove it to make the configuration clean.

Example


```
"DEVICE_NEIGHBOR": {},
```


- INTERFACE - In the Community default configuration, every port is assigned an IP address. The IP address on the interface should be planned and configured by the user according to the deployment plan.

Example


```
"INTERFACE": {
"Ethernet0| 10.0.0.0/31": {},
"Ethernet1| 10.0.0.2/31": {},
"Ethernet2| 10.0.0.4/31": {},
"Ethernet3| 10.0.0.6/31": {},
"Ethernet4| 10.0.0.8/31": {},
"Ethernet5| 10.0.0.10/31": {},
.....
```

```
.....
"Ethernet51| 10.0.0.102/31": {},
"Ethernet52| 10.0.0.104/31": {},
"Ethernet56| 10.0.0.106/31": {}
}
```

How do you reset to "default configuration"?

The default configuration will only be generated on the first boot after installing the image file to the device through ONIE mode.

The better approach is to reset the device to the default configuration and then re-apply the essential configuration.

Configuration manipulation

The following example shows how to reset the device to the default configuration.

Step 1: Install the image from ONIE .

Step 2: Disable ZTP .

```
admin@sonic:~$ sudo config ztp disable
```

Step 3: Delete the old configuration file .

```
admin@sonic:~$ sudo rm /etc/sonic/config_db.json
```

Step 4: Reboot the device .

```
admin@sonic:~$ sudo reboot
```

After reboot, the new default configuration will be generated.

Command Line

The config-setup factory command helps the user quickly reset the device to the default configuration.

Default configuration
admin@sonic:~$ sudo rm /etc/sonic/config_db.json admin@sonic:~$ sudo config-setup factory

Configure Management Interface

- management interface
- static route on management interface

management interface

The management interface in SONiC is eth0 .

. to configure management interface ip on eth0 and remember to save it

```
sudo config interface ip add eth0 192.168.1.1/24 192.168.1.254
sudo config save -y
```

. to remove ip from management interface eth0

```
sudo config interface ip remove eth0 192.168.1.1/24
```

. to show current ip and default gateway setting about management interface, use following command. notice that this command only shows what settings user applied

show management_interface address

```
admin@sonic:~$ show management_interface address
Management IP address = 192.168.8.112/20
Management Network Default Gateway = 192.168.0.254
Management IP address = 2001:db8:2de::e13/64
Management Network Default Gateway = 2001:db8:2de::e15
```

. There is a sub "running " cmd , to show current ip and default gateway configured on linux. this command only for verification purpose. because "show management_interface address" only shows what settings user applied.

show management_interface address running

```
admin@sonic:~$ show management_interface address running
Management IP address = 192.168.8.112/20
Management Network Default Gateway = 192.168.0.254
Management IP address = 2001:db8:2de::e13/64
Management Network Default Gateway = 2001:db8:2de::e15
```

to check management default gateway, use following command, it shows only route in the default table on kernel

```
ip route show table default
```

it's different from data plane (ASIC) routing table, to show data plane (ASIC) routing table please use:

```
show ip route
```

when mgmt vrf is enabled, to check default gateway please use following command. default gateway will be move to "5000" route table when mgmt vrf is enabled.

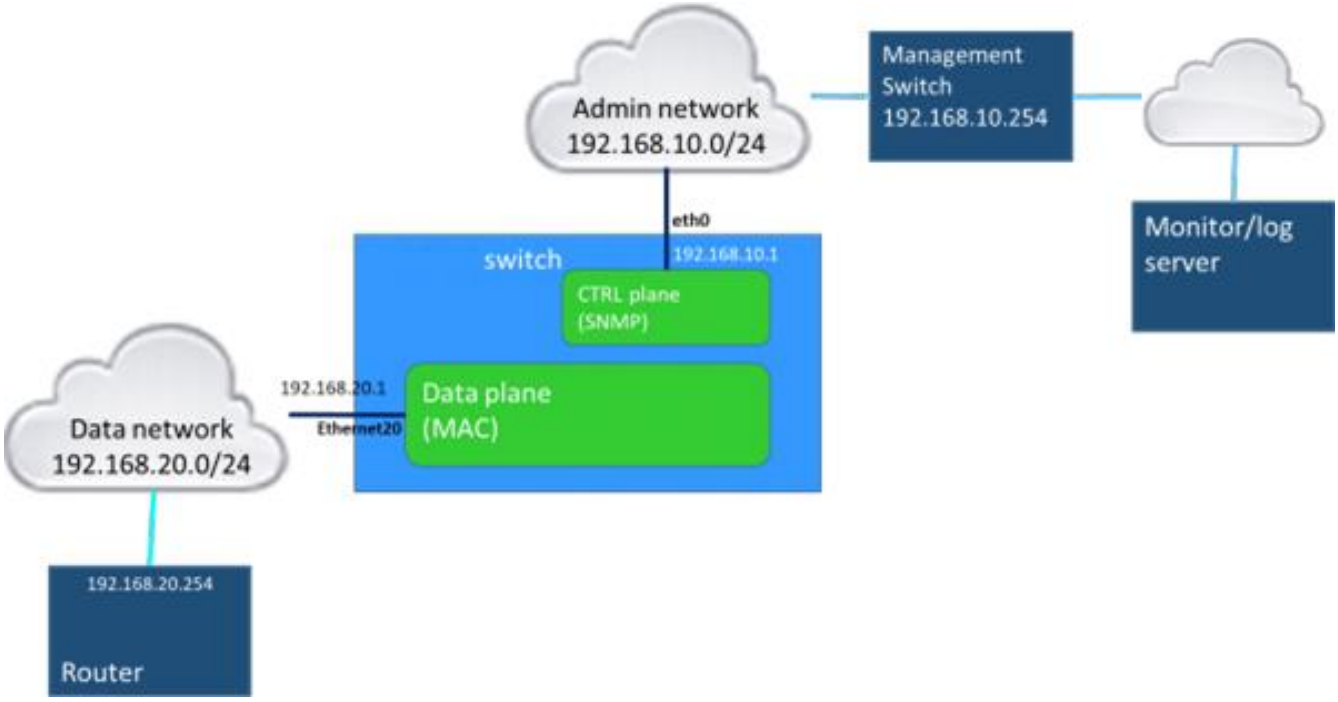
```
ip route show table 5000
```

. Please notice , user should not to use linux ip cmd to add default gateway, it might impact sonic and kernel and frr route entry synchronize scheme.

```
ip route add default via 192.168.1.254 (-->Not recommend)
```

static route on management interface

For some reason, if you want to add a static route on the management interface, you need to be carefully handling the routes on the switch which distinguish the default VRF and management VRF. For example, a network architecture as following: there are two routing interfaces, Ethernet20: 192.168.20.1/24 for data network and eth0: 192,168.10.1/24 for admin network. Due to management purpose, the trap server is located in remote monitor/log server which is behind the management switch 192.168.10.254.



In this application, you may use default gateway in adding IP address command, or use prefix option to specify the route. But please remember to enable management VRF before applying the "prefix" options.

configure default gateway of management interface

```
sudo config vrf add mgmt
sudo config interface ip add eth0 192.168.10.1/24 192.168.10.254
```

or

configure static route on management interface

```
sudo config vrf add mgmt
sudo config interface ip add eth0 192.168.10.1/24
sudo config route add prefix vrf mgmt 0.0.0.0/0 nexthop vrf mgmt 192.168.10.254
```

System Management Configuration

Specifying a Device Name

To specify or modify the name assigned to a switch system, use the config hostname command.

Example
admin@sonic:~$ sudo config hostname CSW06

For more information, see the (202111) Device Hostname section in the CLI reference guide.

NOTE:

- The format of the hostname must be conformed to the rules adopted by prevalent linux systems. For example, the rule of the host name described in https://man7.org/linux/man pages/man7/hostname.7.html(Before SONiC_20230518_055605_399)
- After the hostname is changed through the CLI, the hostname realized by the 'rsyslog' will not be changed automatically. It is required to restart 'rsyslog' service to let 'rsyslog' realize the change of the hostname. Note that besides the 'rsyslog' process running on the host, every docker container also contains a rsyslog process. The most simple way to let all of the 'rsyslog' processes realize the change is to reboot the DUT.
- (After SONiC_20230518_055605_399) The new hostname will be applied to the system and printing in the syslog without additional reload or reboot.

CPU and Memory Utilization

SONiC can display CPU and memory utilization for the system as well as currently active processes.

The "show process cpu" command

The show processes cpu command displays information about the active processes in the switch, and their corresponding CPU utilization statistics. This command uses the Linux "top -bn 1 -o %CPU" command to display the output. The following is a sample output of the show processes cpu command:

Example

```
admin@sonic:~$ show processes cpu
top - 23:50:08 up 1:18, 1 user, load average: 0.25, 0.29, 0.25
Tasks: 161 total, 1 running, 160 sleeping, 0 stopped, 0 zombie
%Cpu(s): 3.8 us, 1.0 sy, 0.0 ni, 95.1 id, 0.1 wa, 0.0 hi, 0.0 si, 0.0 st
KiB Mem: 8181216 total, 1161060 used, 7020156 free, 105656 buffers
KiB Swap: 0 total, 0 used, 0 free. 557560 cached Mem
PID USER PR NI VIRT RES SHR S %CPU %MEM TIME+ COMMAND
2047 root 20 0 683772 109288 39652 S 23.8 1.3 7:44.79 syncd
1351 root 20 0 43360 5616 2844 S 11.9 0.1 1:41.56 redis-server
10093 root 20 0 21944 2476 2088 R 5.9 0.0 0:00.03 top
1 root 20 0 28992 5508 3236 S 0.0 0.1 0:06.42 system
2 root 20 0 0 0 0 S 0.0 0.0 0:00.00 kthreadd
3 root 20 0 0 0 0 S 0.0 0.0 0:00.56 ksoftirqd/0
5 root 0 -20 0 0 0 S 0.0 0.0 0:00.00 kworker/0:0H
```

The "show processes memory" command

The show processes memory command displays information about the active processes in the switch, and the corresponding memory used. This command uses Linux's "top -bn 1 -o %MEM" command to display the output. The following is a sample output of the show processes memory command:

Example

```
admin@sonic:~$ show processes memory
top - 23:41:24 up 7 days, 39 min,  2 users,  load average: 1.21, 1.19, 1.18
Tasks: 191 total,   2 running, 189 sleeping,   0 stopped,   0 zombie
%Cpu(s):  2.8 us, 20.7 sy,  0.0 ni, 76.3 id,  0.0 wa,  0.0 hi,  0.2 si,  0.0 st
KiB Mem :  8162264 total,  5720412 free,   945516 used,  1496336 buff/cache
KiB Swap:        0 total,        0 free,        0 used.  6855632 avail Mem
```

PID	USER	PR	NI	VIRT	RES	SHR	S	%CPU	%MEM	TIME+	COMMAND
18051	root	20	0	851540	274784	8344	S	0.0	3.4	0:02.77	syncd
17760	root	20	0	1293428	259212	58732	S	5.9	3.2	96:46.22	syncd
508	root	20	0	725364	76244	38220	S	0.0	0.9	4:54.49	dockerd

30853	root	20	0	96348	56824	7880	S	0.0	0.7	0:00.98	show
17266	root	20	0	509876	49772	30640	S	0.0	0.6	0:06.36	docker
24891	admin	20	0	515864	49560	30644	S	0.0	0.6	0:05.54	docker
17643	admin	20	0	575668	49428	30628	S	0.0	0.6	0:06.29	docker
23885	admin	20	0	369552	49344	30840	S	0.0	0.6	0:05.57	docker
18055	root	20	0	509076	49260	30296	S	0.0	0.6	0:06.36	docker
17268	root	20	0	371120	49052	30372	S	0.0	0.6	0:06.45	docker
1227	root	20	0	443284	48640	30100	S	0.0	0.6	0:41.91	docker
23785	admin	20	0	443796	48552	30128	S	0.0	0.6	0:05.58	docker
17820	admin	20	0	435088	48144	29480	S	0.0	0.6	0:06.33	docker
506	root	20	0	1151040	43140	23964	S	0.0	0.5	8:51.08	containerd
18437	root	20	0	84852	26388	7380	S	0.0	0.3	65:59.76	python3.6

The following table lists and describes the fields in the show processes cpu and show processes memory output:

Field	Description
top	Display Linux processes.
Load average	The system load over the last one, five, and 15 minutes respectively.
Tasks	Shows total number of processes, and the number of running, sleeping, stopped, and zombie processess.
%cpu(s)	Shows the CPU usage separated by types. The data are the values between screen refreshes. The values are: . us: user processes . sy: system processes . ni: nice user processes . id: the CPU's idle time; a high idle time means there's not a lot going on otherwise . wa: wait time, or time spent waiting for I/O completion . hi: time spent waiting for hardware interrupts . si: time spent waiting for software interrupts . st: time stolen from this VM by the hypervisor
KiB Mem	Memory size
KiB Swap	Swapped size
PID	Process ID
USER	User name of the owner of the task.

PR	The process's priority. The lower the number, the higher the priority.
NI	Represents a nice value of task. A negative nice value implies higher priority, and a positive nice value means lower priority.
VIRT	Total virtual memory used by the task.
RES	How much physical RAM the process is using, measured in kilobytes.
SHR	Represents the shared memory size (kb) used by a task.
S	Process status. The status of the task which can be one of the following: • D = uninterruptible sleep • I = idle, R = running • S = sleeping • T = stopped by job control signal • t = stopped by debugger during trace • Z = zombie
%CPU	Represents the CPU usage
%MEM	Shows the memory usage of the task.
TIME+	CPU time, the same as 'TIME', but reflecting more granularity through hundredths of a second.
COMMAND	The name of the command that started the process.

Docker Warm Restart

In addition to a device-level warm reboot, SONiC also supports a Docker-based warm restart for the following Dockers: BGP, teamD, and SWSS. This feature enables you to restart a particular Docker with no interruption of packet forwarding and no effect on other services.

The config warm_restart enable/disable command is used to enable or disable the warm restart for a particular service that supports it.

When you restart a particular service using "systemctl restart <service_name>", the warm restart configuration is checked to see whether it is enabled or disabled. If the configuration is enabled for that service, a warm reboot is performed. Otherwise, a cold restart is performed for the service.

Example

```
admin@sonic:~$ sudo config warm_restart enable swss
admin@sonic:~$ sudo config warm_restart enable teamd
```

For more detailed information on performing warm restarts, setting timers, and displaying the status, see the CLI reference guide.

Initiating a Warm Reboot

To initiate a warm reboot of the system in SONiC, use the warm-reboot command as shown below.

Example

```
admin@sonic:~$ sudo warm-reboot -v
Tue Oct 22 23:20:53 UTC 2019 Pausing orchagent ...
Tue Oct 22 23:20:53 UTC 2019 Stopping radv ...
Tue Oct 22 23:20:54 UTC 2019 Stopping bgp ...
Tue Oct 22 23:20:54 UTC 2019 Stopped bgp ...
Tue Oct 22 23:20:57 UTC 2019 Initialize pre-shutdown ...
Tue Oct 22 23:20:58 UTC 2019 Requesting pre-shutdown ...
Tue Oct 22 23:20:58 UTC 2019 Waiting for pre-shutdown ...
Tue Oct 22 23:20:59 UTC 2019 Pre-shutdown succeeded ...
Tue Oct 22 23:20:59 UTC 2019 Backing up database ...
Tue Oct 22 23:21:00 UTC 2019 Stopping teamd ...
Tue Oct 22 23:21:00 UTC 2019 Stopped teamd ...
Tue Oct 22 23:21:00 UTC 2019 Stopping syncd ...
Tue Oct 22 23:21:11 UTC 2019 Stopped syncd ...
Tue Oct 22 23:21:11 UTC 2019 Stopping all remaining containers ...
Tue Oct 22 23:21:13 UTC 2019 Stopped all remaining containers ...
Tue Oct 22 23:21:15 UTC 2019 Rebooting with /sbin/kexec -e to SONiC-OS-20191021.01 ...
```

For more information, see the Warm Reboot section in the CLI reference guide.

Using Container Auto-Restart

The SONiC software is built on a modular concept where features and network applications are installed as “Docker containers.” A container allows an application to run isolated from its environment, and

Docker is a standard method for building containers. This enables applications to be added or replaced without impacting the whole SONiC system.

Additionally, SONiC allows Docker containers to be automatically shut down and restarted when unexpected conditions occur. Restarting a container makes sure that the configuration is reloaded and all processes are restarted and can run in a healthy state.

To enable or disable the auto-restart feature for a container, use the config feature auto restart command from the CLI and specify the container name.

Example

```
admin@sonic:~$ sudo config feature auto restart database disabled
```

To display the status of the auto-restart feature for all containers or a specific container, use the show feature auto restart command.

Example

```
admin@sonic:~$ show feature autorestart
Container Name   Status
--------------   --------
database         enabled
syncd            enabled
teamd            disabled
dhcp_relay       enabled
lldp             enabled
pmon             enabled
bgp              enabled
swss             disabled
telemetry        enabled
sflow            enabled
snmp             enabled
radv             disabled
```

For more information, see the Container Auto-restart section in the CLI reference guide.

Network Time Protocol (NTP)

The Network Time Protocol (NTP) enables a switch to synchronize its internal clock based on periodic updates from NTP time servers.

This synchronized time is used to record accurate dates and times for events on the device.

The multiple NTP time servers can be specified that a switch will poll for regular time updates. All responses received from multiple NTP time servers are filtered and compared to determine the most reliable and accurate time update.

To add a server to a switch's NTP server list, use the config ntp add command and specify the IP address.

To remove a server IP address from the list, use the config ntp del command.

Example

```
admin@sonic:~$ sudo config ntp add 9.9.9.9
NTP server 9.9.9.9 added to configuration
Restarting ntp-config service...

admin@sonic:~$ sudo config ntp del 9.9.9.9
NTP server 9.9.9.9 removed from configuration
Restarting ntp-config service...
```

When the NTP service is started, it listens on eth0 by default. This means that the NTP server must be accessible from eth0.

To configure an ntp source interface, run the config ntp source-interface set or config ntp source-interface unset command.

example：

```
admin@sonic:~$ sudo config ntp source-interface set Ethernet18
Restarting ntp-config service...
admin@sonic:~$ sudo config ntp source-interface unset Ethernet18
Restarting ntp-config service...
```

Run the show running configuration ntp command to display the current NTP configuration

example：

```
admin@sonic:~$ show running configuration ntp
NTP source-interface    : Ethernet18
NTP Servers
-------------
```

```
9.9.9.9
```

Run the show ntp command to display the NTP time update status

example：

```
admin@sonic:~$ show ntp
synchronised to NTP server (204.2.134.164) at stratum 3
  time correct to within 326797 ms
  polling server every 1024 s

  remote           refid      st t when poll reach   delay   offset jitter
==============================================================================

 23.92.29.245    .XFAC.          16 u    - 1024    0    0.000    0.000    0.000

*204.2.134.164   46.233.231.73    2 u 916 1024 377    3.079    0.394    0.128
```

Limitation：

Description The DHCP server cannot obtain the IP address of eth0. To work around this limitation, add the following code snippet to the configuration in "/etc/son/config_db.json". After the file is updated, run the systemctl restart ntp config command to restart the ntp service.

The steps are shown below.

The following shows the steps Step 1, add the following section in "/etc/son/config_db.json" :

```
{
...
  "NTP": {
      "global": {
        "src_intf": "eth0"
      }
    }
...
}
```

Step 2 Restart the NTP service：

```
sudo systemctl restart ntp-config
```

Fixes long drift clock:

sonic default NTP server is variable clock jump. If the current system time shifts too much (1000 seconds) to that of the upstream NTP server. It's never synchronized.

It is best to manually synchronize the time before adding the NTP server configuration.

```
sudo systemctl stop ntp.service
sudo /usr/sbin/ntpd -gq -u 0 time.google.com
sudo systemctl start ntp.service

#add your upstream ntp server
sudo config ntp add 9.9.9.9
```

System Log (syslog)

Syslog

SONiC supports the logging of error messages from a switch and sending them to a remote System Log (syslog) server. You can specify multiple syslog servers using the config syslog add command to add server IP addresses to the syslog server list. To remove a syslog server from the list, use the config syslog delete command.

Example

```
admin@sonic:~$ sudo config syslog add 1.1.1.1
Syslog server 1.1.1.1 added to configuration
Restarting rsyslog-config service...

admin@sonic:~$ sudo config syslog del 1.1.1.1
Syslog server 1.1.1.1 removed from configuration
Restarting rsyslog-config service...
```

For more information, see the (202111) Syslog section in the CLI reference guide.

How Change the Syslog Format

This document describes how to change the syslog message format. The default syslog format is RFC 3164, and SONiC supports changing the format to RFC 5424. The following are the general steps to change the format:

1. Remove all configured servers.
2. Edit the file /usr/share/sonic/templates/rsyslog.conf.j2 to change the format.
3. Add the syslog server.

Here's the example of how to change the format:

Step 1, Remove all syslog servers

Use the show syslog command to list all the servers. Then use the config syslog del command to delete them.

Step 2, Edit the /usr/share/sonic/templates/rsyslog.conf.j2 file to change the format

In the file, find the text #Remote syslog logging . After this text, you will see two lines that show the current used format in Templa te="SONiC File Format_RFC3164" .

/usr/share/sonic/templates/rsyslog.conf.j2

```
$template SONiCFileFormat,"%timegenerated%.%timegenerated:::date-subseconds% %HOSTNAME%
%syslogseverity-text:::uppercase% %syslogtag%%msg:::sp-if-no-1st-sp%%msg:::drop-last-lf%\n"
$template SONiCFileFormat_RFC3164,"<%PRI%>%timegenerated%.%timegenerated:::date-
subseconds% %HOSTNAME% %
syslogseverity-text:::uppercase% %syslogtag%%msg:::sp-if-no-1st-sp%%msg:::drop-last-lf%\n"
$template SONiCFileFormat_RFC5424,"<%PRI%>1 %TIMESTAMP:::date-rfc3339% %HOSTNAME%
%APP-NAME% %PROCID% %
MSGID% %STRUCTURED-DATA% %msg%\n"
$ActionFileDefaultTemplate SONiCFileFormat

#
# Remote syslog logging
#

# The omfwd plug-in provides the core functionality of traditional message forwarding via UDP and plain
TCP.
# It is a built-in module that does not need to be loaded.

{% if SYSLOG_SERVER is defined %}
{% for server, data in SYSLOG_SERVER.items() %}
{% set params_list = [] %}
{% if 'source' in data %}
{% set dummy = params_list.append('address=' + '"' + data.source|string + '"') %}
{% endif %}
{% if 'port' in data %}
{% set dummy = params_list.append('port=' + '"' + data.port|string + '"') %}
{% endif %}
{% if 'vrf' in data and data['vrf'] != "default" %}
{% set dummy = params_list.append('device=' + '"' + data.vrf|string + '"') %}
```

```
{% endif %}
{% if params_list %}
*.* action(type="omfwd" target="{{ server }}" protocol="udp" {{ params_list|join('  ') }} template="
SONiCFileFormat_RFC3164")
{% else %}
*.* action(type="omfwd" target="{{ server }}" protocol="udp" template="SONiCFileFormat_RFC3164")
{% endif %}
{% endfor %}
{% endif %}
```

Edit the following line to choose the desired format:

- RFC 3164: use SONiCFileFormat_RFC3164
- RFC 5424: use SONiCFileFormat_RFC5424
- SONiC original community format: use SONiCFileFormat

For example, change the format to RFC 5424 as shown below:

/usr/share/sonic/templates/rsyslog.conf.j2 (example)

```
#
# Remote syslog logging
#

# The omfwd plug-in provides the core functionality of traditional message forwarding via UDP and plain
TCP.
# It is a built-in module that does not need to be loaded.

{% if SYSLOG_SERVER is defined %}
{% for server, data in SYSLOG_SERVER.items() %}
{% set params_list = [] %}
{% if 'source' in data %}
{% set dummy = params_list.append('address=' + '"' + data.source|string + '"') %}
{% endif %}
{% if 'port' in data %}
{% set dummy = params_list.append('port=' + '"' + data.port|string + '"') %}
{% endif %}
{% if 'vrf' in data and data['vrf'] != "default" %}
{% set dummy = params_list.append('device=' + '"' + data.vrf|string + '"') %}
```

```
{% endif %}
{% if params_list %}
*.* action(type="omfwd" target="{{ server }}" protocol="udp" {{ params_list|join('  ') }} template="
SONiCFileFormat_RFC5424")
{% else %}
*.* action(type="omfwd" target="{{ server }}" protocol="udp" template="SONiCFileFormat_RFC5424")
{% endif %}
{% endfor %}
{% endif %}
```

Note: Any changes to /usr/share/sonic/templates/rsyslog.conf.j2 not mentioned in this document are outside the scope of the maintenance service.

Step 3, add the syslog server

Example

```
admin@sonic:~$ sudo config syslog add 192.168.1.1
```

Troubleshooting Information

SONiC can collect relevant information about the device's status for troubleshooting and debugging purposes. This data includes syslog entries, database state, routing-stack state, e.t.c. The EC SONiC development team can review this data to troubleshoot, which has been compressed into an archive file. The syntax of the archived file generated is "/var/dump/DEVICE HOST NAME> YYYYMMDD HHMMSS. tar.gz".

You can use the show techsupport command to create an archive file for troubleshooting.

Example

```
admin@sonic:~$ show techsupport --since=yesterday # Will collect syslog and core files for the last 24
hours

admin@sonic:~$ show techsupport --since='hour ago' # Will collect syslog and core files for the last one
hour
```

For more information, see the (202111) Troubleshooting section in the CLI reference guide.

Configuration Reloading

config reload

With this command, you can import a new configuration from the input file or /etc/sonic/config db.json while clearing the existing one. This command stops all services before clearing the configuration and then restarts those services.

The execution of the config reload takes some time since it restarts a lot of services that are currently running on the device.

Note：

If a user has logged in through SSH, he may get disconnected depending upon the new management IP address and need to reconnect their SSH sessions. In general, it is recommended to execute this command from the console port after disconnecting all SSH sessions to the device.

When users do “config reload”, the newly loaded config may or may not have a management IP address. If there is a management IP in the newly loaded config file, it might be different or identical to the previously defined value.

The possible outcomes of executing config reload are as follows.

Case 1: Previously configured management IP is the same as the newly loaded management IP. The SSH session might be completely unaffected, but it might experience a temporary pause. However, assuming the client’s timeout value isn’t on the order of a couple of seconds, the session would likely just resume again as soon as the interface is reconfigured and up with the same IP.

Case 2: Previously configured management IP is different from newly loaded management IP. Users' SSH connections will be lost. . Case 3: Newly loaded config does not have any management IP. Users' SSH connections will be lost.

Note：

Management interface IP address and a default route (or specific route) may require reconfiguration in case those parameters are not part of the minigraph.xml.

Example:

```
admin@sonic:~$ sudo config reload
Clear current config and reload config from the file /etc/sonic/config_db.json? [y/N]: y
Running command: systemctl stop dhcp_relay
Running command: systemctl stop swss
Running command: systemctl stop snmp
Warning: Stopping snmp.service, but it can still be activated by: snmp.timer
Running command: systemctl stop lldp
Running command: systemctl stop pmon
Running command: systemctl stop bgp
Running command: systemctl stop teamd
Running command: /usr/local/bin/sonic-cfggen -H -k Force10-Z9100-C32 --write-to-db
Running command: /usr/local/bin/sonic-cfggen -j /etc/sonic/config_db.json --write-to-db
Running command: systemctl restart hostname-config
Running command: systemctl restart interfaces-config
Timeout, server 10.11.162.42 not responding.
```

Error Condition:

```
admin@sonic:~$ sudo config reload -y
System is not up. Retry later or use -f to avoid system checks

admin@sonic:~$ sudo config reload -y
Relevant services are not up. Retry later or use -f to avoid system checks

admin@sonic:~$ sudo config reload -y
SwSS container is not ready. Retry later or use -f to avoid system checks
```

Note：

Config reload has two system sanity checks.

(1) Checks the services which are grouped under delayed target up.

(2) Checks swss is running for at least 120 seconds. If you want to ignore these two sanity checks, please use "config reload -f".

Other command options related to config reload are documented in section (202111) Reloading Configuration of the CLI guide.

Management Framework (>>> programming)

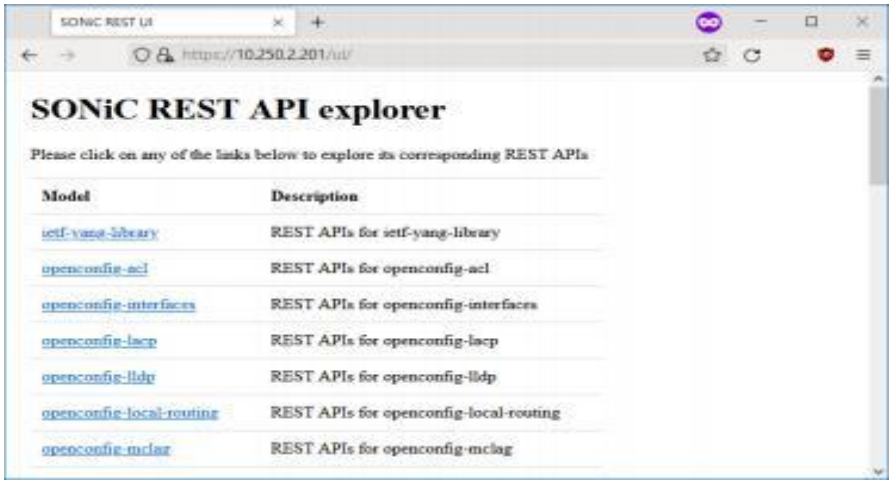
Management framework use REST server to manage configuration and status on SONiC switches.

REST server will provide Swagger UI based online documentation and test UI for all
REST APIs it supports. Documentation can be accessed by launching URL
https://REST_SERVER_IP/ui in a browser.
This page will list all supported OpenAPI definition files (both YANG generated and manual) along with link to open Swagger UI for them.

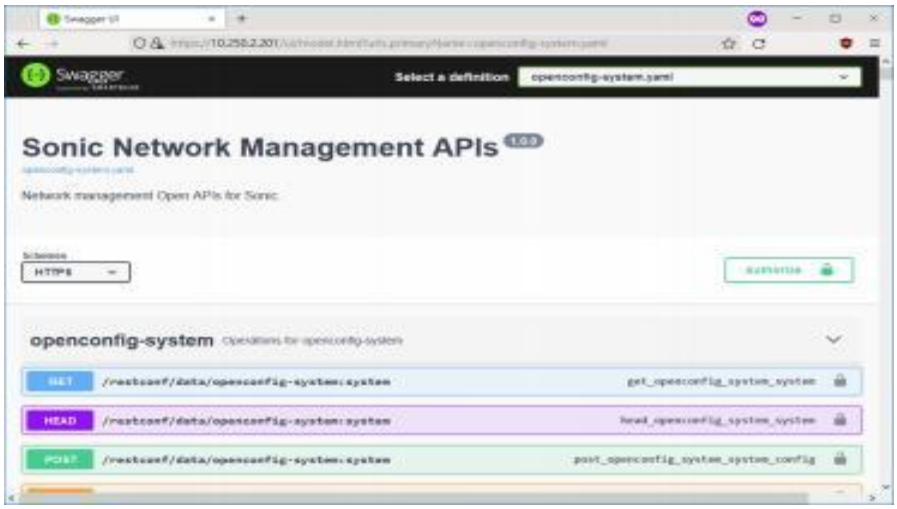
Swagger Web UI

View of web UI

- Homepage



- Page inside each model



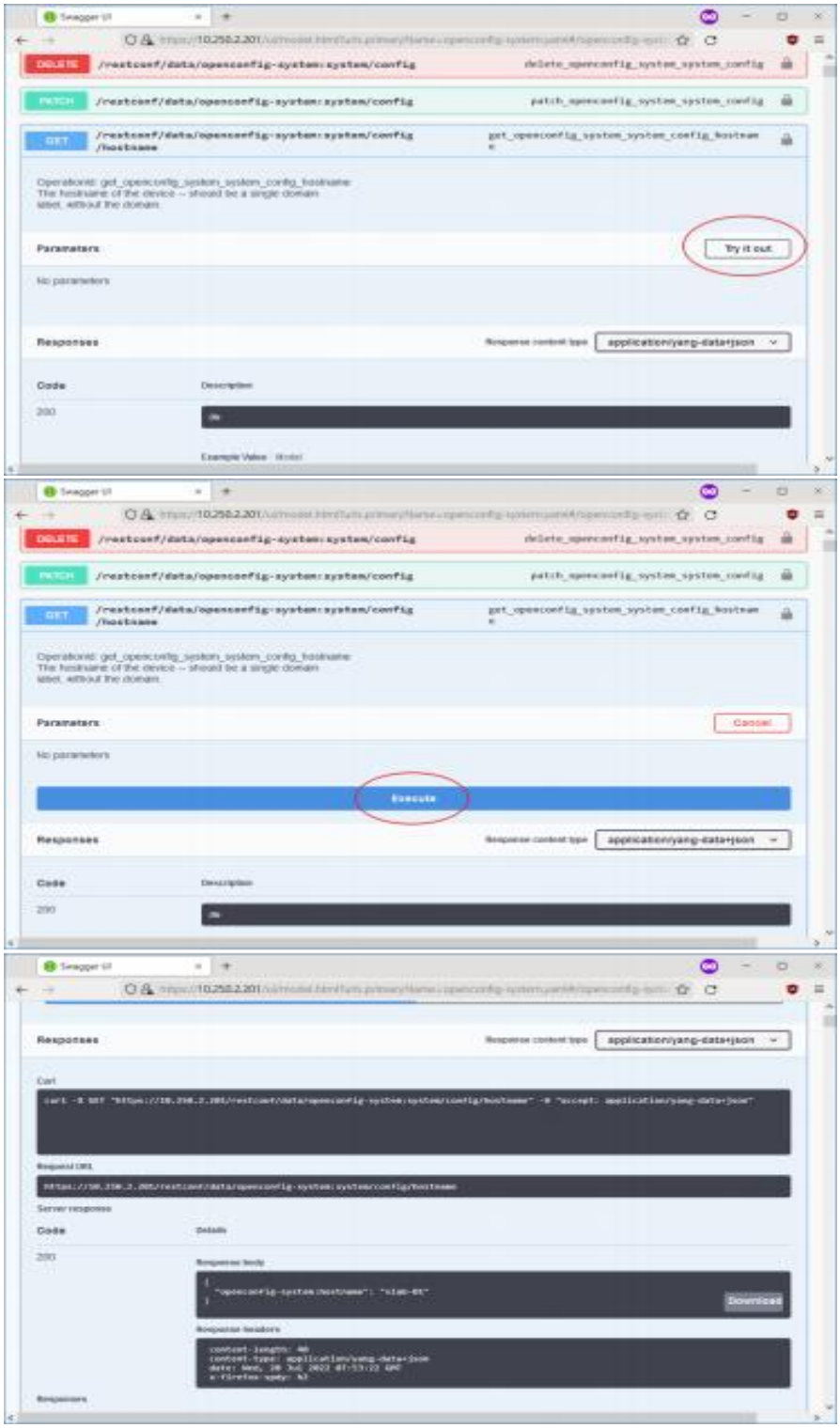
Usage

In web UI, user can click item in the page, to expand the corresponding API operation block.

In this block, user can click "Try it out", to fill the necessary parameter and then execute the API operation.

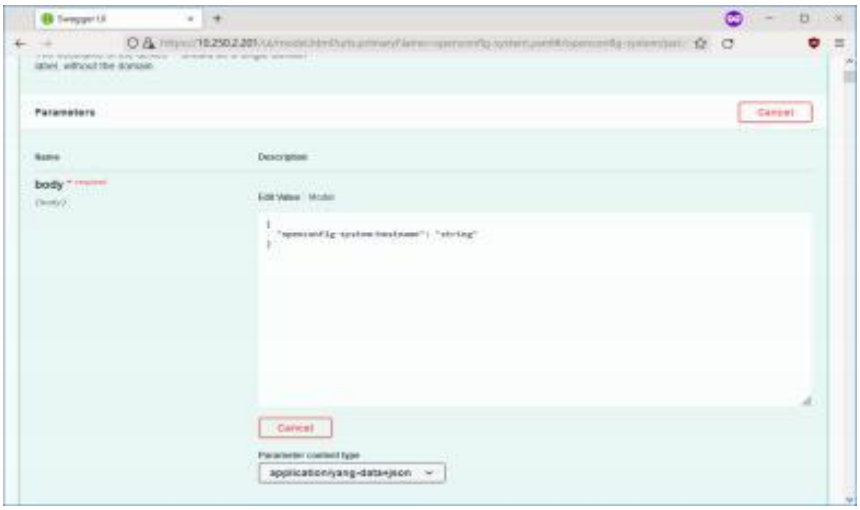
For example, if user want to GET the path /restconf/data/openconfig system:system/config/hostname Click `Try it out` button, and then click `Execute` button.

Then the result of this operation will return in this page.

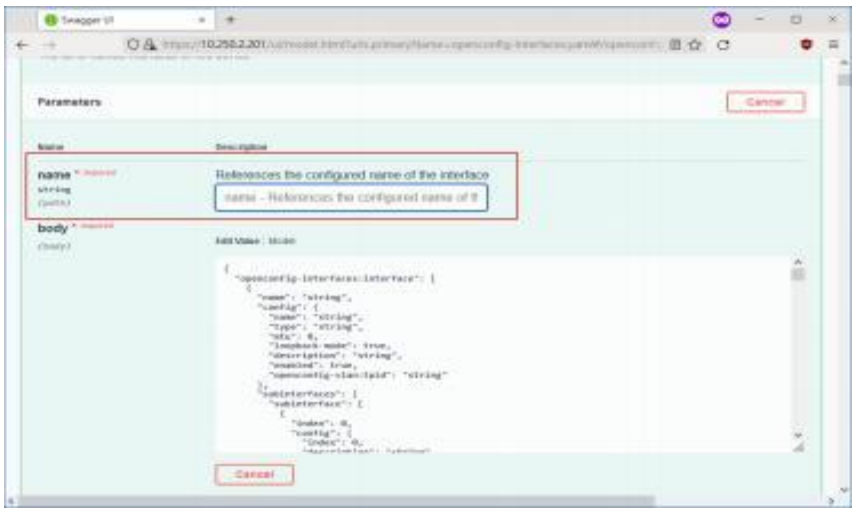


Some operations need the parameter as payload, which is , such as PATCH, PUT, POST. User needs to fill the parameter in input box.

And the input box will be filled in the sample value in advance, user can modify it as user need.



If the parameter is in the request path, such as /restconf/data/openconfig-interfaces:interfaces/interface={name}, the corresponding input box will show separately.



REST API by CURL command

Apart from Web UI, user can also use curl command to send REST request to do the same operation. for test, add parameter "-k" to allow insecure server connections when using SSL

Usage of CURL to Access REST API
curl -X {METHOD} {URL} -d {payload}

For example, to GET the path /restconf/data/openconfig-system:system/config/hostname

```
$ curl -k -X GET "https://10.250.2.201/restconf/data/openconfig-system:system/config/hostname" -H "accept: application/yang-data+json"
{"openconfig-system:hostname":"vlab-01"}
```

- **Example for operation with payload**

```
$ curl -k -X PATCH "https://10.250.2.201/restconf/data/openconfig-system:system/config/hostname" -H "accept: application/yang-data+json" -H  "Content-Type: application/yang-data+json" -d "{  \" openconfig-system:hostname\": \"vlab-01-test\"}"
```

- **Example of URL path with parameter**

  path: /restconf/data/openconfig-interfaces:interfaces/interface={name}/openconfig-vlan:routed-vlan /openconfig-if-ip:ipv4

  Use parameter "Vlan1000" to replace "{Name}" in the path

```
$ curl -k -X GET "https://10.250.2.201/restconf/data/openconfig-interfaces:interfaces /interface=Vlan1000/openconfig-vlan:routed-vlan/openconfig-if-ip:ipv4" -H  "accept: application/yang-
data+json"
{"openconfig-if-ip:ipv4":{"addresses":{"address":[{"config":{"ip":"192.168.1.1","openconfig-interfaces-ext:secondary":false,"prefix-length":24},"ip":"192.168.1.1","state":{"ip":"192.168.1.1","openconfig-interfaces-ext:secondary":false,"prefix-length":24}}]}}}
```

- **Example of both payload and path which contains parameter**

```
$ curl -k -X PATCH "https://10.250.2.201/restconf/data/openconfig-interfaces:interfaces /interface=Vlan1000/openconfig-vlan:routed-vlan/openconfig-if-ip:ipv4" -H"accept: application/yang-data+json" -H"Content-Type: application/yang-data+json" -d "{\"openconfig-if-ip:ipv4\": {\" addresses\": {\"address\": [{\"ip\": \"192.168.1.1\",\"config\": {\"ip\": \"192.168.1.1\",\"prefix-length\": 24}}]}}}"
```

Supported REST APIs

The above is the usage and example, the full supported REST APIs list can access the related pdf file.

MIB & Trap Support List

Standard MIB and Trap

MIB List:

	MIB	RFC	OID	status
1	lldpMIB (ieee8021lldp)	IEEE 802.1AB	1.0.8802.1.1.2	only 1.0.8802.1.1.2.1.3.1~8 1.0.8802.1.1.2.1.4.1.1.4~12

				1.0.8802.1.1.2.1.4.2.1.3~5
2	mib-2 (mib)	RFC 1213	1.3.6.1.2.1	only 1.3.6.1.2.1.1.1~9 1.3.6.1.2.1.3.1.1.1~3 1.3.6.1.2.1.4.1.1,2,25,26,31~37 1.3.6.1.2.1.4.21.1.7 1.3.6.1.2.1.4.22.1.2 1.3.6.1.2.1.4.34.1.3.1 1.3.6.1.2.1.5.1~30 1.3.6.1.2.1.6.1~15,19,20 1.3.6.1.2.1.7.1~5,7 1.3.6.1.2.1.10.7.2 1.3.6.1.2.1.11.1~32 1.3.6.1.2.1.15.1~4 1.3.6.1.2.1.25.1~6 1.3.6.1.2.1.28.1~2 1.3.6.1.2.1.55.1 1.3.6.1.2.1.88.1
3	snmpModules	RFC 2578	1.3.6.1.6.3	only 1.3.6.1.6.3.1,10~12,15,16
4	ifMIB	RFC 2863	1.3.6.1.2.1.2	only 1.3.6.1.2.1.2.1 1.3.6.1.2.1.2.2.1.1~21 1.3.6.1.2.1.31.1.1.1.1~19
5	ipCidrRouteDest (IpCidrRouteTable)	RFC 4292	1.3.6.1.2.1.4.24.4.1.1	only 1.3.6.1.2.1.4.24.4.1.1,16
6	dot1qTpFdbPort (in Q-BRIDGE-MIB)	RFC 4363	1.3.6.1.2.1.17.7.1.2.2.1.2	only 1.3.6.1.2.1.17.7.1.2.2.1.2
7	dot1qPvid(in Q-BRIDGE-MIB)	RFC 4363	1.3.6.1.2.1.17.7.1.4.5.1.1	only 1.3.6.1.2.1.17.7.1.4.5.1.1

8	entityMIB	RFC 2737 (v2)	1.3.6.1.2.1.47	only 1.3.6.1.2.1.47.1.1.1.1.2~16
9	entitySensorMIB	RFC 3433	1.3.6.1.2.1.99	only 1.3.6.1.2.1.99.1.1.1.1~5

Trap List:

Trap Name	snmpTrapOID
coldStart	1.3.6.1.6.3.1.1.5.1
linkDown	1.3.6.1.6.3.1.1.5.3
linkUp	1.3.6.1.6.3.1.1.5.4
authenticationFailure	1.3.6.1.6.3.1.1.5.5
entConfigChange	1.3.6.1.2.1.47.2.0.1
ospfVirtIfStateChange	1.3.6.1.2.1.14.16.2.1
ospfIfStateChange	1.3.6.1.2.1.14.16.2.16
ospfNbrStateChange	1.3.6.1.2.1.14.16.2.2
ospfVirtNbrStateChange	1.3.6.1.2.1.14.16.2.3
bgpEstablishedNotification	1.3.6.1.2.1.15.0.1
bgpBackwardTransNotification	1.3.6.1.2.1.15.0.2

Hardware Reset

Prerequisites Hardware Reset
Hardware Reset
Execute the Command on the Device that Supports Hardware Reset
Execute the Command on the Device that Does Not Support Hardware Reset

Hardware Reset is a feature that allows network administrators to restart the hardware of network devices through the CLI without performing a power cycle. This user guide provides brief instructions on how to perform Hardware Reset on your devices with SONiC system.

Prerequisites Hardware Reset

The hardware reset feature might not support on every hardware model. Please check the release note to know whether the hardware model you used is in the support list.

Hardware Reset

Execute the Command on the Device that Supports Hardware Reset

After the command is issued, the device gets reset through the hardware reset method immediately, and it might take several seconds to see the messages from console which is printed by BIOS.

Example
admin@sonic:~$ sudo hardware-reset

Execute the Command on the Device that Does Not Support Hardware Reset

When the command is issued on the device that does not support hardware reset, the error message shown below will be printed.

Example
admin@sonic:~$ sudo hardware-reset Hardware Reset is not supported on this platform xxxxxx

Protocol Down

Introduction

Protocol down is a feature to shutdown a port. Compare to the admin state which is controlled by the user, the protocol down is controlled by the program/process run inside the SONiC.

User should only use this feature to trace the port shutdown reason.

Below shows the possible reasons for the protocol down to shutdown a port in section 1. And in section 2, we show three related commands to check the result.

In section 3, we show the procedure to analyze the possible cause of a down port.

1. Protocol Down Reasons

Currently, there are four reasons in protocol down. Each protocol down reason corresponds to a program/process.

If any reason is set to TRUE, the protocol down will be enabled and the port will be shutdown.

1. EVPN
2. PMS
3. OBJTRK
4. STP

2. Commands

There are three commands to show the protocol down condition.

show interfaces status (new columns)

```
admin@n9200-64dc:~$ show interfaces status
Interface          Lanes    Speed    MTU    Oper FEC          Alias          Vlan                                  Oper
Admin    ProtoDown    Eff Admin                Type    Asym PFC    Oper Speed                                      ------    -------
--------------  ---------------  -------  -----  ----------  -------------  --------------                        down
-----------  -----------  ----------------  ----------  ------------
Ethernet0          49,50,51,52    100G               9100    rs    Eth1(Port1)    routed                          up
down       False    down                *QSFP28 or         later    off         100G
Ethernet4          53,54,55,56    100G               9100    rs    Eth2(Port2)    trunk
up         False     up    *QSFP28 or later          off         100G
```

ProtoDown: The combination result for all the protocol down reasons. True in this column means at least one of the protocol down reasons is True.

Eff Admin: The actual admin_state set to the SAI layer. It is a synthesized result of ProtoDown and Admin.

show protocol-down brief (new command)

```
admin@n9200-64dc:/usr$ show protocol-down brief
  Interface    ProtoDown    Admin    Effective Admin

  Ethernet0       False       up             down
  Ethernet4       False       up               up
```

show protocol-down status (new command)

```
admin@n9200-64dc:~$ show protocol-down status
      Interface    EVPN    PMS    OBJTRK    STP    ProtoDown    Admin    Effective Admin

      Ethernet0    False  False     False  False       False     down             down
      Ethernet4    False  False     False  False       False       up               up
```

3. Guide for checking port down status

1. Check Eff Admin(Effective Admin) if its value is down. If not, jump to 5.
2. Check if the Admin is down. If it is down, try to startup the interface first.
3. Check if the ProtoDown is True. If it is True, check the reason by the "show protocol-down status" command.
4. If the port is a trunk member, check if the trunk port has any protocol down reason been set.
5. Check other FEC setting, pre-emphasis or even the physical connection condition.

Notice

It is not recommended to enable the feature that will shutdown the port at the same time except that there is an explicit declaration that claims the features can be applied at the same time.

Simple Network Management Protocol (SNMP)

Simple Network Management Protocol (SNMP) is a standard protocol defined by the Internet Engineering Task Force (IETF) for managing and monitoring network devices. The SNMP agent is a software application that runs on a managed device, collecting and storing management information about the device and responding to requests from SNMP managers. Management Information Bases (MIBs) are hierarchical databases that define the structure and attributes of managed objects within a device. They provide a standardized way to represent and access management information using SNMP.

The supported MIBs can be found in (202111) MIB & Trap Support List.

To add the SNMP community, use config snmp community add command. To display the SNMP configuration, use show runningconfiguration snmp command.

Configuring SNMP Community

To add a SNMP community, use the config snmp community add command and specify the community string.

To remove a SNMP community, use the config snmp community del command.

Example
admin@sonic:~$ sudo config snmp community add testcomm ro

Management VRF

By default, the SNMP service runs in the default VRF. However, when the management VRF is enabled, the SNMP service must be manually moved from the default VRF to the management VRF using the snmp agent address add command.

Once the management VRF is enabled, use the snmp agent address add command to specify the listening IP address obtained from the show ip interfaces command and the management VRF, as shown in the following example:

Example
admin@sonic:~$ sudo config snmp agent address add 1.2.3.4 -v mgmt

When the management VRF is disabled, use the snmp agent address del command to remove the listening IP address, which will move the SNMP service back to the default VRF. If you want the SNMP service to run on a specific VRF, use the snmp agent address add command to specify the IP address and VRF name.

For more information, see the (202111) SNMP section in the CLI reference guide.

Telnet Server

A Telnet server allows remote clients to connect to the device over a network using the Telnet protocol. Telnet provides a text-based communication interface through which users can execute commands on the remote machine as if they were logged in locally. Here's a detailed explanation of its features and functionalities:

Key Features of a Telnet Server:

1. Remote Access:
    - Telnet servers enable users to log in to a remote system from any location, given they have the necessary credentials and network access.
2. Command Execution:
    - Once connected, users can execute commands on the remote machine. This is typically used for administrative tasks, configuring devices, and troubleshooting.
3. Text-Based Interface:
    - Telnet provides a simple, text-based interface. Users interact with the server using command-line instructions.
4. Network Communication:
    - The Telnet protocol operates over TCP (Transmission Control Protocol), usually on port 23. This ensures reliable communication between the client and the server.
5. Session Management:
    - The server manages multiple simultaneous sessions, allowing various clients to connect and interact with the remote machine independently.

How Telnet Works:

1. Client-Server Model:
    - Telnet operates on a client-server model. The Telnet client initiates a connection to the Telnet server, which then prompts the user for authentication (username and password).
2. Connection Establishment:
    - The client connects to the server using the Telnet protocol, typically specifying the server's IP address or hostname and the desired port number (default is 23).
3. Authentication:
    - The server requests authentication information from the client. This usually involves entering a username and password.
4. Command Execution:
    - Once authenticated, the user can enter commands through the Telnet client, which are sent to the server for execution. The server processes these commands and returns the output to the client.
5. Session Termination:
    - The session remains active until the user logs out or the connection is terminated. The server then closes the connection and releases any associated resources.

Telnet server is only supported on the following models:

- N9200-64DC
- N9500-128QC
- N9500-64OC

By default, the telnet server starts at boot. If you want to disable it, use the following command: systemctl disable inetd.service

Example1: stop telnet server

The following example shows how to stop telnet server, using systemctl stop inetd.service command.

Example

systemctl stop inetd.service

Example2: start telnet server

The following example shows how to start telnet server, using systemctl start inetd.service

Example

systemctl start inetd.service

Users and Passwords

Sonic uses and passwords are stored as follows:

- /etc/group stores group information

- /etc/passwd stores user information

- /etc/shadow stores encrypted passwords

These files, being part of the Linux filesystem, will be replaced during version upgrades. This means that after an upgrade, the original usernames and passwords will be lost and revert to default values. Moreover, unlike other Sonic configurations, config save/reload operations do not affect these files. In other words, usernames and passwords cannot be restored to factory settings or backed up through these methods.

To avoid the above issues and enhance the security of user passwords, we provide the following features:

1. To implement password preservation so that passwords are not lost during version upgrades.

2. To enable configuration save and reload, allowing username and password configurations to be handled along with other Sonic configurations through config save/reload.

3. To avoid the need for copying additional files during configuration migration.

4. Under the default configuration, logging in as admin will prompt a password change. After entering the original password, a new password must be set. The new password must be at least 8 characters long and include at least one number, one uppercase letter, and one special character, simple patterns like 123456 should be avoided.

5. Add password fail lock. When a user try to login with incorrect password 5 times in a row, the user will be locked out for 5 minutes, during which any login for this user will be denied.

Example1: First login

Under the default configuration (may be caused by ONIE installation, factory restore, and config ztp disabled), logging in as admin will prompt a password change. The rule for the new password refers to “Change Password”

logging as below:

Example

sonic login: admin

Password:

You are required to change your password immediately (administrator enforced).

Changing password for admin.

Current password:

New password:

Retype new password:

After modifying the password, the configuration needs to be saved as follows:

sudo config save

Example2: Change password

The new password must be at least 8 characters long and include at least one number, one uppercase letter, and one special character, simple patterns like 123456 should be avoided.

Example

The following examples demonstrate various error messages for password validation failures:

admin@sonic:~$ passwd

Changing password for admin.

Current password:

New password:

BAD PASSWORD: it is too simplistic/systematic

New password:

BAD PASSWORD: is too simple

New password:

The password has not been changed.

passwd: Have exhausted maximum number of retries for service

passwd: password unchanged

admin@sonic:~$

We get the message “it is too simplistic/systematic” when type “abc123456”; get “is too simple” when it is “@abcxyz3u1”(there is no uppercase letter); get “The password has not been changed” when type the old password.

Example3: Load config

When load config from a config file, the users and passwords are also loaded.

Example

config load /etc/sonic/config_db.json

Interface Configuration

Shutting Down and Starting Up Interfaces

Use the config interface shutdown command to administratively shut down either a physical interface or port-channel interface and to enable a physical or port-channel interface, use the config interface startup command. These two commands will also allow configuring multiple interfaces in one command.

- Shutting Down The Interface
- Starting Up the Interfaces

Shutting Down The Interface

Step1: Use the show interfaces status command to check the status of the interfaces.

Example

```
admin@sonic:~$ show interfaces status
Interface      Lanes    Speed    MTU    FEC        Alias          Vlan    Oper    Admin
Type    Asym PFC    Oper Speed
-------------  -------  -------  -----  -----  -------------  --------------  ------  -------
--------------  ----------  ------------
      Ethernet0         2     0M   9100    N/A    Eth1(Port1)        routed     up
up          RJ45       off         10G
      Ethernet1         1     0M   9100    N/A    Eth2(Port2)         trunk     up
up          RJ45       off         10G
      Ethernet2         4     0M   9100    N/A    Eth3(Port3)         trunk     up
up          RJ45       off         10G
      Ethernet3         3     0M   9100    N/A    Eth4(Port4)         trunk     up
up          RJ45       off         10G
      Ethernet4         6     0M   9100    N/A    Eth5(Port5)         trunk     up
up          RJ45       off         10G
      Ethernet5         5     0M   9100    N/A    Eth6(Port6)         trunk     up
up          RJ45       off         10G
      Ethernet6         8     0M   9100    N/A    Eth7(Port7)         trunk     up
up          RJ45       off         10G
      Ethernet7         7     0M   9100    N/A    Eth8(Port8)         trunk     up
up          RJ45       off         10G
```

Step 2: Use the config interface shutdown command to shut down either one interface, or multiple interfaces as shown below.

Example

admin@sonic:~$ sudo config interface shutdown Ethernet1

admin@sonic:~$ sudo config interface shutdown Ethernet2,Ethernet3,Ethernet4,Ethernet5

Step 3: Once interfaces are configured, use the show interfaces status command to check the status.

Example

admin@sonic:~$ show interfaces status Ethernet1,Ethernet2,Ethernet3,Ethernet4,Ethernet5

Interface	Lanes	Speed	MTU	FEC	Alias	Vlan	Oper	Admin	Type	Asym PFC	Oper Speed
Ethernet1	1	0M	9100	N/A	Eth2(Port2)	trunk	down	down	RJ45	off	10G
Ethernet2	4	0M	9100	N/A	Eth3(Port3)	trunk	down	down	RJ45	off	10G
Ethernet3	3	0M	9100	N/A	Eth4(Port4)	trunk	down	down	RJ45	off	10G
Ethernet4	6	0M	9100	N/A	Eth5(Port5)	trunk	down	down	RJ45	off	10G
Ethernet5	5	0M	9100	N/A	Eth6(Port6)	trunk	down	down	RJ45	off	10G

Starting Up the Interfaces

Step 4: Use the config interface startup command to enable either one interface, or multiple interfaces as shown below.

Example

admin@sonic:~$ sudo config interface startup Ethernet1

admin@sonic:~$ sudo config interface startup Ethernet2,Ethernet3,Ethernet4,Ethernet5

Step 5: Once interfaces are configured, use the show interfaces status command again to check the status.

Example

admin@sonic:~$ show interfaces status Ethernet1,Ethernet2,Ethernet3,Ethernet4,Ethernet5

Interface	Lanes	Speed	MTU	FEC	Alias	Vlan	Oper	Admin	Type	Asym PFC	Oper Speed

```
  Ethernet1      1     0M   9100    N/A  Eth2(Port2)  trunk     up      up    RJ45
off         10G
  Ethernet2      4     0M   9100    N/A  Eth3(Port3)  trunk     up      up    RJ45
off         10G
  Ethernet3      3     0M   9100    N/A  Eth4(Port4)  trunk     up      up    RJ45
off         10G
  Ethernet4      6     0M   9100    N/A  Eth5(Port5)  trunk     up      up    RJ45
off         10G
  Ethernet5      5     0M   9100    N/A  Eth6(Port6)  trunk     up      up    RJ45
off         10G
```

For more information, see the (202111) Interfaces section in the CLI reference guide.

Configuring Interface IP Addresses

SONiC can configure an IP address for a physical interface, port-channel, VLAN interface, or loopback interface using the config interface ip add command. To specify a VLAN, the VLAN ID replaces the interface name.

When configuring an IP address for the management interface "eth0", you can also provide the default gateway IP address as an optional parameter.

To remove an IP address from an interface, use the config interface remove command.

Example

admin@sonic:~$ sudo config interface ip add Ethernet63

10.11.12.13/24

admin@sonic:~$ sudo config interface ip add eth0 20.11.12.13/24

20.11.12.254

admin@sonic:~$ sudo config interface ip add vlan100 10.11.12.13/24

admin@sonic:~$ sudo config interface ip remove Ethernet63

10.11.12.13/24

For more information, see the (202111) Interfaces section in the CLI reference guide.

Interface VLAN MAC

VLAN interfaces can be configured with MAC addresses.

Create a VLAN interface and then run the config vlan mac_address command to change the MAC address of the VLAN interface.

example：

```
admin@sonic:~$ sudo config vlan add 200
admin@sonic:~$ sudo config vlan member add 200 -u Ethernet64
admin@sonic:~$ sudo config interface ip add Vlan200 200.1.1.200/24
admin@sonic:~$ sudo config vlan mac_address 200 80:a2:35:88:88:89
admin@sonic:~$ sudo ifconfig Vlan200
Vlan200: flags=4099<UP,BROADCAST,MULTICAST>  mtu 9100
        inet 200.1.1.200  netmask 255.255.255.0  broadcast 200.1.1.255
        ether 80:a2:35:88:88:89  txqueuelen 1000  (Ethernet)
        RX packets 0  bytes 0 (0.0 B)
        RX errors 0  dropped 0  overruns 0  frame 0
        TX packets 0  bytes 0 (0.0 B)
        TX errors 0  dropped 0 overruns 0  carrier 0  collisions 0
```

Run the show vlan brief --mac command to display the MAC addresses configured for the current VLAN interface

example：

```
admin@sonic:~$ show vlan brief --mac
+-----------+---------------+-------------------+
|   VLAN ID | Ports         | VLAN MAC          |
+===========+===============+===================+
|       200 |   Ethernet64  | 80:a2:35:88:88:89 |
+-----------+---------------+-------------------+
```

Link Aggregation Control Protocol (LACP)

Link Aggregation Control Protocol (LACP) is a protocol for dynamic link aggregation. LACP communicates with the peer end through LACPDUs. After LACP is enabled for a port, the interface in the dynamic aggregation group automatically uses LACP. The port sends LACPDUs to notify the peer of its system priority, system

MAC address, port priority, port number, and operation Key. After receiving the information, the peer end compares the information with the information saved on other ports to select the port that can be aggregated. In this way, the peer end can agree on whether to add or withdraw the ports from a dynamic aggregation group. To delete a portchannel, run the config portchannel <portchannel_name> command. The portchannel_name must be in the format of "PortChannelxxxx", where "xxxx" is a 1 to 4 digit value, for example, "PortChannel0002".

config portchannel has the following six options:

- min-links – the minimum number of links required to start an aggregate link.
- fallback - true/false. When LACP fallback is set to true, each portchannel selects a member port as the active port during fallback mode. The LACP fallbac function allows an active port with LACP enabled to LAG before receiving LACPDUs from the peer port.
- lacp-key - auto/number of lacp key. Sets the generated or given value to the lacp key, which defaults to' auto'.
- lacp-system-id - Specifies the LACP system ID in the format of xxxx.xxxx.xxxx. The MAC of the DUT if not specified。
- lacp-system-priority - Configures the LACP system priority. The value ranges from 0 to 65535. The default value is 65535.
- dev-number - The default value is 1.

To view information about portchannel, run the show interface portchannel command.

example：

```
admin@sonic:~$ sudo config portchannel add PortChannel1
admin@sonic:~$ show interfaces portchannel
Flags: A - active, I - inactive, Up - up, Dw - Down, N/A - not available,
       S - selected, D - deselected, * - not synced,
       M - mixed speed
  No.  Team Dev      Protocol     Ports        Oper Key    Admin Key    Fast Rate    System ID    System Priority
Dev Number
-----  ------------  -----------  ------------  ----------  -----------  -----------  -----------  -----------------  ------------
    1  PortChannel1  LACP(A)(Dw)  Ethernet3(D)  11          auto         false                                        1
    2  PortChannel2  LACP(A)(Dw)  N/A           N/A         auto         false                                        1
```

Configuring Port Mirroring

Port mirroring allows you to mirror the traffic from any switch source port to a destination port for real-time analysis. SONiC supports two types of mirroring:

- SPAN: Switch Port Analyzer (SPAN) is the local mirroring of traffic from one or more source ports on a switch to the destination port on the same switch.
- ERSPAN: Encapsulated Remote SPAN (ERSPAN) supports the mirroring of traffic from source ports on multiple switches over a Layer 3 network. ERSPAN uses Generic Routing Encapsulation (GRE) to tunnel the mirrored traffic across the network to the switch that contains the destination port.

To add or remove a SPAN or ERSPAN mirror session in SONiC, use the config mirror_session command. Using this command, each mirror session is identified by a session name. In addition, both SPAN and ERSPAN support ACL-based mirroring and can be used in ACL configurations.

To configure a SPAN mirror session, specify the source ports/LAG and destination port, and then select whether you want to mirror ingress/egress traffic or both.

To configure an ERSPAN mirror session, users need to configure the following fields that are used to forward the mirrored packets:

- Source IP address
- Destination IP address
- DSCP (QoS) value with which mirrored packets are forwarded
- TTL value
- Optional - GRE type in the GRE tunnel packet. The range is 0...0xFFFF, although a decimal format is also accepted. The default value is 0x88BE.
- Optional - Queue in which packets shall be sent out of the device. This parameter accepts 0 only in the current implementation.
- Optional - List of source ports that can include both Ethernet and LAG ports.
- Optional - Direction: The mirror session direction when configured along with the source port (Supports rx/tx/both. The default direction is both).
- Optional - Policer that is used to control the rate at which frames are mirrored.

Example

```
admin@sonic:~# config mirror_session span add port0 Ethernet0 Ethernet4,PortChannel10,Ethernet8

admin@sonic:~# config mirror_session erspan add mrr_port 1.2.3.4 20.21.22.23 8 100 0x6558 0 Ethernet0

admin@sonic:~# show mirror_session
ERSPAN Sessions
Name      Status  SRC IP    DST IP      GRE    DSCP  TTL  Queue  Policer  Monitor Port  SRC Port  Direction
```

```
mrr_port   inactive 1.2.3.4   20.21.22.23 0x6558       8 100        0
Ethernet0   both

SPAN Sessions
Name   Status   DST Port   SRC Port                        Direction   Queue   Policer

port0   active   Ethernet0   Ethernet4,PortChannel10,Ethernet8 both

admin@sonic:~#
```

For more information on port mirroring, see the (202111) Mirroring section in the CLI reference guide.

Configure Interface Speed, Breakout Mode and Auto-Negotiation

- Configure Interface Speed
- Configure Interface Breakout Mode
- Configure Interface Auto-Negotiation

Configure Interface Speed

To configure an interface speed, you need to know the speeds supported by the interface, and then configure the interface with a valid value. The value of 10000, 40000, 100000, and 400000 indicates 10G, 40G, 100G and 400G respectively.

Use the CLI command config interface speed to specify an interface speed, as shown below.

Example (configure interface speed with command)

```
admin@sonic:~$ sudo config interface speed Ethernet10 40000
```

Or, you can modify the interface speed in the configuration file config_db.json.

Example (configure interface speed in configuration file)

```
"PORT": {
    "Ethernet10": {
        "alias": "Eth1/1",
        "description": "Eth1/1",
        "index": "1",
        "lanes": "2",
        "mtu": "9100",
```

```
        "parent_port": "Ethernet0",
        "pfc_asym": "off",
        "speed": "40000"
    },
```

Use the show interfaces status command to check the configuration.

Example (show interface status)

```
admin@sonic:~$ show interfaces status Ethernet10
Interface          Lanes    Speed    MTU    FEC          Alias    Oper    Admin
-------------  -----------  -------  -----  -----  ---------------  ------  -------
Ethernet10       101,102     40G   9100     rs   fortyGigE1/1/1     up      up
```

Note：

An interface will not be linked up if the value in the configuration file is not a supported value.

Configure Interface Breakout Mode

The port breakout feature allows an interface to extend switch ports by splitting a port into two or four lower-speed ports using breakout cable. For example, to split one 100G port (1x100G) into four 25G interfaces (4x25G), use the show interfaces breakout command to see the available breakout modes for the interface.

Example (show interfaces breakout)

```
admin@lnos-x1-a-fab01:~$ show interfaces  breakout
{
    "Ethernet0": {
        "index": "1,1,1,1",
        "default_brkout_mode": "1x100G[40G]",
        "child ports": "Ethernet0",
        "child port speed": "100G",
        "breakout_modes": "1x100G[40G],2x50G,4x25G[10G]",
        "Current Breakout Mode": "1x100G[40G]",
        "lanes": "65,66,67,68",
        "alias_at_lanes": "Eth1/1, Eth1/2, Eth1/3, Eth1/4"
    ....
    }
```

```
}
```

You can use the config interface breakout command to select the mode you need.

Example

```
admin@sonic:~$ sudo config interface breakout Ethernet0
1x100G[40G] 2x50G
admin@sonic:~$ sudo config interface breakout Ethernet0 2x50G
```

Note：

The breakout mode should match the breakout cable used in the connection.

For example, when selecting the 2x50G mode, you must use a 2x50G breakout cable.

The port dependency configuration should be removed before performing port breakout.

The following error message instructs you to remove the dependency configuration.

Error

```
Dependecies Exist. No further action will be taken
*** Printing dependecies ***
/sonic-acl:sonic-acl/ACL_TABLE/ACL_TABLE_LIST[ACL_TABLE_NAME='DATAACL']/ports[.='Ethernet0']
/sonic-acl:sonic-
acl/ACL_TABLE/ACL_TABLE_LIST[ACL_TABLE_NAME='EVERFLOW']/ports[.='Ethernet0']
/sonic-acl:sonic-
acl/ACL_TABLE/ACL_TABLE_LIST[ACL_TABLE_NAME='EVERFLOWV6']/ports[.='Ethernet0']
/sonic-buffer-pg:sonic-buffer-pg/BUFFER_PG/BUFFER_PG_LIST[port='Ethernet0'][pg_num='0']/port
/sonic-buffer-pg:sonic-buffer-pg/BUFFER_PG/BUFFER_PG_LIST[port='Ethernet0'][pg_num='1']/port
/sonic-buffer-pg:sonic-buffer-pg/BUFFER_PG/BUFFER_PG_LIST[port='Ethernet0'][pg_num='2']/port
/sonic-buffer-pg:sonic-buffer-pg/BUFFER_PG/BUFFER_PG_LIST[port='Ethernet0'][pg_num='3']/port
/sonic-buffer-pg:sonic-buffer-pg/BUFFER_PG/BUFFER_PG_LIST[port='Ethernet0'][pg_num='4']/port
/sonic-buffer-pg:sonic-buffer-pg/BUFFER_PG/BUFFER_PG_LIST[port='Ethernet0'][pg_num='5']/port
/sonic-buffer-pg:sonic-buffer-pg/BUFFER_PG/BUFFER_PG_LIST[port='Ethernet0'][pg_num='6']/port
/sonic-buffer-pg:sonic-buffer-pg/BUFFER_PG/BUFFER_PG_LIST[port='Ethernet0'][pg_num='7']/port
/sonic-buffer-queue:sonic-buffer-
queue/BUFFER_QUEUE/BUFFER_QUEUE_LIST[port='Ethernet0'][qindex='0']/port
/sonic-buffer-queue:sonic-buffer-
queue/BUFFER_QUEUE/BUFFER_QUEUE_LIST[port='Ethernet0'][qindex='1']/port
/sonic-buffer-queue:sonic-buffer-
queue/BUFFER_QUEUE/BUFFER_QUEUE_LIST[port='Ethernet0'][qindex='2']/port
```

```
/sonic-buffer-queue:sonic-buffer-
queue/BUFFER_QUEUE/BUFFER_QUEUE_LIST[port='Ethernet0'][qindex='3']/port
/sonic-buffer-queue:sonic-buffer-
queue/BUFFER_QUEUE/BUFFER_QUEUE_LIST[port='Ethernet0'][qindex='4']/port
/sonic-buffer-queue:sonic-buffer-
queue/BUFFER_QUEUE/BUFFER_QUEUE_LIST[port='Ethernet0'][qindex='5']/port
/sonic-buffer-queue:sonic-buffer-
queue/BUFFER_QUEUE/BUFFER_QUEUE_LIST[port='Ethernet0'][qindex='6']/port
/sonic-buffer-queue:sonic-buffer-
queue/BUFFER_QUEUE/BUFFER_QUEUE_LIST[port='Ethernet0'][qindex='7']/port
/sonic-interface:sonic-interface/INTERFACE/INTERFACE_LIST[name='Ethernet0']/name
/sonic-interface:sonic-interface/INTERFACE/INTERFACE_IPPREFIX_LIST[name='Ethernet0'][ip-prefix='
10.0.0.0/31']
/name
/sonic-interface:sonic-interface/INTERFACE/INTERFACE_IPPREFIX_LIST[name='Ethernet0'][ip-
prefix='fc00::1/126']

/name
```

Use -f to forcibly remove the dependency.

Example

```
admin@sonic:~$ sudo config interface breakout Ethernet0 4x25G[10G] -f
```

If you previously configured "sflow interface" for the interface, you can't use "sflow interface enable" or "sflow interface disable" to remove the port dependency.

Please use -f to remove the dependency.

If the following error message appears while configuring the interface breakout, It may be that a field is missing from the config_db.json.

Error

```
sonic_yang(6):Note: Below table(s) have no YANG models: KDUMP, REST_SERVER, SNMP,
SNMP_COMMUNITY, XCVRD_LOG
libyang[0]: Leafref "/sonic-mgmt_port:sonic-mgmt_port/sonic-mgmt_port:MGMT_PORT/sonic-mgmt_port:
MGMT_PORT_LIST/sonic-mgmt_port:name" of value "eth0" points to a non-existing leaf. (path: /sonic-
mgmt_interface:sonic-
mgmt_interface/MGMT_INTERFACE/MGMT_INTERFACE_LIST[name='eth0'][ip_prefix=' 192.168.1.1
/24']/name)
sonic_yang(3):Data Loading Failed:Leafref "/sonic-mgmt_port:sonic-mgmt_port/sonic-
```

```
mgmt_port:MGMT_PORT/sonic-
mgmt_port:MGMT_PORT_LIST/sonic-mgmt_port:name" of value "eth0" points to a non-existing leaf.
Data Loading Failed
Leafref "/sonic-mgmt_port:sonic-mgmt_port/sonic-mgmt_port:MGMT_PORT/sonic-
mgmt_port:MGMT_PORT_LIST/sonic-
mgmt_port:name" of value "eth0" points to a non-existing leaf.
ConfigMgmt Class creation failed
Failed to break out Port. Error: Failed to load the config. Error: ConfigMgmtDPB Class creation failed
```

The above error indicates that the MGMT_PORT field, which is typically referred to by MGMT INTERFACE is missing. The MGMT_INTERFACE is configured, when user configure IP address in the management interface (eth0).

It can be fixed by adding the following field to config_db.json.

MGMT_PORT

```
"MGMT_PORT": {
    "eth0": {
        "alias": "eth0",
        "admin_status": "up",
        "description": "Management Port"
    }
}
```

Configure Interface Auto-Negotiation

Refer to (202111) Auto-Negotiation in CLI reference guide.

Link-Training

What Link Training does

Enable Link-Training

Show Link-Training

Disble Link-Training

Show Link-Training

LT Admin

LT Oper

Feature support information

Note

What Link Training does

Link training is a process by which the transmitter and receiver on a high-speed serial link communicate with each other to tune the equalization settings. Theoretically, enabling link-training automatically tunes the FIR filter for each channel to achieve the desired bit error rate (BER). The IEEE 802.3 standard defines a set of link training protocols for various mediums. In this section, we focus on IEEE clause 72 and 93, dynamically improving the link quality over the SFP coppers and backplanes and Link Training behavior when auto-negotiation is disabled.

The configuration of Link Training and port auto-negotiation are independent. User may enable (or disable) Link Training and/or auto- negotiation individually. The operations of Link Training and auto-negotiation maybe depend on the ASIC design. Link Training command is applied on interface, config interface link-training, to enable (or disable) link-training on a physical interface.

Enable Link-Training

Step1: Use "show interfaces link-training status" command to check the link-training configuration and status.

Display link-training configuration and status

```
admin@sonic:~$ show interfaces link-training status
  Interface    LT Oper    LT Admin    Oper    Admin
-----------  ---------  ----------  ------  -------
  Ethernet0        off         N/A      up       up
  Ethernet4        off         off      up       up
  Ethernet8        off          on    down       up
```

Step 2: Use "config interface link-training <interface_name> on" command to enable link training.

Enable link-training at Ethernet4

```
admin@sonic:~$ sudo config interface link-training Ethernet4 on
```

Show Link-Training

Step 3: Once configured, use " show interfaces link-training status" command to check the link - training configuration and status.

Display link-training configuration and status

```
admin@sonic:~$ show interfaces link-training status
  Interface    LT Oper    LT Admin    Oper    Admin
```

Ethernet0	off	N/A	up	up
Ethernet4	off	on	down	up
Ethernet8	off	on	down	up

Disble Link-Training

Step 4: Use "config interface link-training <interface_name> off" command to disable link - training.

Disable link-training at Ethernet4

```
admin@sonic:~$ sudo config interface link-training Ethernet4 off
```

Show Link-Training

Step 5: Once configured, use " show interfaces link-training status" command to check the link - training configuration and status.

Display link-training configuration and status

```
admin@sonic:~$ show interfaces link-training status
  Interface    LT Oper    LT Admin    Oper    Admin

  Ethernet0      off          N/A       up       up
  Ethernet4      off          off       up       up
  Ethernet8      off           on       down     up
```

LT Admin

LT Admin	Link-Training configuration
N/A	Not configured, default configuration
on	Link-Training is enabled by user configuration
off	Link-Training is disabled by user configuration

LT Oper

LT Oper	Description
off	Disabled
on	Enabled, while the further operational status is not yet available
trained	Enabled, while the pre-emphasis is tuned successfully
not_trained	Enabled, while the pre-emphasis is not yet tuned, no further failure is available

frame_lock	Enabled, while the training frame delineation is detected
snr_low	Enabled, while the SNR low threshold is detected
timeout	Enabled, while the training process is timed out

Feature support information

Chip Family	DUT	Support or not
BCM	N9200-64DC	√
BCM	N9500-128QC	√
BCM	N9500-64OC	√

Note

1. The above configuration works when port auto-negotiation is disabled.
2. If the port auto-negotiation is enabled, the link-training configuration shall be auto-enabled.
3. If link-training configuration is enabled on port auto-negotiation enabled, the configuration shall be saved and reapplied when the port auto-negotiation configuration switch to disabled.

Configure Laser Frequency and TX Power for ZR Transceiver

- Configure transceiver laser frequency
- Configure transceiver TX power
- Configure transceiver laser frequency

You should set both frequency and grid space in this CLI command. Now the supported grid space are only 75GHz and 100GHz.

Example (configure transceiver laser frequency with command)
admin@sonic:~$ sudo config interface transceiver frequency Ethernet0 193200 100 Setting laser frequency to 193200 GHz and grid space to 100 GHz on port Ethernet0

It needs some time to modify transceiver frequency due to the data-path is needed to re-initialize. You can check the initialization state by following command.

Example (show transceiver states with command)
admin@sonic:~$ show logging CMIS Nov 1 03:46:56.709089 sonic NOTICE pmon#xcvrd: CMIS: Ethernet0: 400G, lanemask=0xff,

state=INSERTED,

retries=0

Nov 1 03:46:57.304001 sonic NOTICE pmon#xcvrd: CMIS: Ethernet0: force Datapath reinit

Nov 1 03:46:57.380135 sonic NOTICE pmon#xcvrd: CMIS: Ethernet0: 400G, lanemask=0xff, state=DP_DEINIT,

retries=0

Nov 1 03:46:57.509658 sonic NOTICE pmon#xcvrd: CMIS: Ethernet0: unable to enter low-power mode

Nov 1 03:46:57.672149 sonic NOTICE pmon#xcvrd: CMIS: Ethernet0: 400G, lanemask=0xff, state=DP_DEINIT,

retries=1

Nov 1 03:46:59.945924 sonic NOTICE pmon#xcvrd: CMIS: Ethernet0: 400G, lanemask=0xff, state=AP_CONFIGURED, retries=1

Nov 1 03:47:49.136637 sonic NOTICE pmon#xcvrd: message repeated 48 times: [CMIS: Ethernet0: 400G,

lanemask=0xff, state=AP_CONFIGURED, retries=1]

Nov 1 03:47:49.136637 sonic NOTICE pmon#xcvrd: CMIS: Ethernet0 configured laser frequency 193200 GHz grid space 100 GHz

Nov 1 03:47:50.293814 sonic NOTICE pmon#xcvrd: CMIS: Ethernet0: 400G, lanemask=0xff, state=DP_INIT, retries=1

Nov 1 03:47:51.398751 sonic NOTICE pmon#xcvrd: CMIS: Ethernet0: 400G, lanemask=0xff, state=DP_TXON, retries=1

Nov 1 03:48:09.655975 sonic NOTICE pmon#xcvrd: message repeated 17 times: [CMIS: Ethernet0: 400G,

lanemask=0xff, state=DP_TXON, retries=1]

Nov 1 03:48:09.655975 sonic NOTICE pmon#xcvrd: CMIS: Ethernet0: 400G, lanemask=0xff, state=DP_ACTIVATION, retries=1

Nov 1 03:48:09.667160 sonic NOTICE pmon#xcvrd: CMIS: Ethernet0: READY

When the data-path is ready you can use the show command to check the active frequency and grid space.

Example (show transceiver laser frequency with command)

```
admin@sonic:~$ sudo show interfaces transceiver frequency Ethernet0
_________ ___________ ___________
Port      Frequency   Grid space

Ethernet0 193200GHz   100GHz
```

Configure transceiver TX power

The tx power can be configured by following CLI command. If the parameter is a negative, it needs to add "--" before the negative number.

Example (configure transceiver tx power with command)

```
admin@sonic:~$ sudo config interface transceiver tx_power Ethernet0 -- -10
Setting target Tx output power to -10.0 dBm on port Ethernet0
```

It doesn't need to re-initialize the data-path for TX power modified.

Example (show transceiver state with command)

```
admin@sonic:~$ sudo show logging CMIS
Nov  1 04:00:53.672734 sonic NOTICE pmon#xcvrd: CMIS: Ethernet0: 400G, lanemask=0xff,
state=INSERTED,
retries=0
Nov  1 04:00:54.816412 sonic NOTICE pmon#xcvrd: CMIS: Ethernet0 Successfully configured Tx power =
-9.0
Nov  1 04:00:55.183279 sonic NOTICE pmon#xcvrd: CMIS: Ethernet0: READY
```

You can check the TXPOWER in transceiver eeprom digital optical monitoring (DOM) data.

Example (show transceiver tx power with command)

```
admin@sonic:~$ sudo show interfaces transceiver eeprom Ethernet0 --dom
Ethernet0: SFP EEPROM detected
        Active App Selection Host Lane 1: 1
        Active App Selection Host Lane 2: 1
        Active App Selection Host Lane 3: 1
        Active App Selection Host Lane 4: 1
        Active App Selection Host Lane 5: 1
        Active App Selection Host Lane 6: 1
        Active App Selection Host Lane 7: 1
        Active App Selection Host Lane 8: 1
        Active Firmware Version: 1.9
        Application Advertisement:
                1: 400GAUI-8 C2M (Annex 120E) |  400ZR, DWDM, amplified
        CMIS Revision: 5.0
        Connector: LC
        Encoding: N/A
        Extended Identifier: Power Class 8 (19.0W Max)
        Extended RateSelect Compliance: N/A
```

Hardware Revision: 0.0

Host Electrical Interface: 400GAUI-8 C2M (Annex 120E)

Host Lane Assignment Options: 1

Host Lane Count: 8

Identifier: QSFP-DD Double Density 8X Pluggable Transceiver

Inactive Firmware Version: 1.9

Length Cable Assembly(m): 0.0

Media Interface Code: 400ZR, DWDM, amplified

Media Interface Technology: C-band tunable laser

Media Lane Assignment Options: 1

Media Lane Count: 1

Nominal Bit Rate(100Mbs): 0

Specification compliance: sm_media_interface

Supported Max Laser Frequency: 196100GHz

Supported Max TX Power: -8.0dBm

Supported Min Laser Frequency: 191300GHz

Supported Min TX Power: -15.0dBm

Vendor Date Code(YYYY-MM-DD Lot): 2022-05-23 00

Vendor Name: NeoPhotonics

Vendor OUI: 00-15-06

Vendor PN: QDDZR400100C1000

Vendor Rev: 00

Vendor SN: EVT2122019

ChannelMonitorValues:

RX1Power: -11.57390760389438dBm

RX2Power: -infdBm

RX3Power: -infdBm

RX4Power: -infdBm

RX5Power: -infdBm

RX6Power: -infdBm

RX7Power: -infdBm

RX8Power: -infdBm

TX1Bias: 283.6mA

TX1Power: -8.917733436250714dBm

TX2Bias: 0.0mA

TX2Power: -infdBm

TX3Bias: 0.0mA

TX3Power: -infdBm

TX4Bias: 0.0mA

TX4Power: -infdBm

TX5Bias: 0.0mA

TX5Power: -infdBm

TX6Bias: 0.0mA

TX6Power: -infdBm

TX7Bias: 0.0mA

TX7Power: -infdBm

TX8Bias: 0.0mA

TX8Power: -infdBm

ChannelThresholdValues:

RxPowerHighAlarm : 4.0dBm

RxPowerHighWarning: 2.0dBm

RxPowerLowAlarm : -20.0dBm

RxPowerLowWarning : -18.013dBm

TxBiasHighAlarm : 470.0mA

TxBiasHighWarning : 400.0mA

TxBiasLowAlarm : 20.0mA

TxBiasLowWarning : 50.0mA

TxPowerHighAlarm : -4.0dBm

TxPowerHighWarning: -6.002dBm

TxPowerLowAlarm : -19.031dBm

TxPowerLowWarning : -17.011dBm

ModuleMonitorValues:

Temperature: 70.086C

Vcc: 3.155Volts

ModuleThresholdValues:

TempHighAlarm : 80.0C

TempHighWarning: 78.0C

TempLowAlarm : -3.0C

TempLowWarning : 0.0C
VccHighAlarm : 3.5Volts
VccHighWarning : 3.45Volts
VccLowAlarm : 3.085Volts
VccLowWarning : 3.135Volts

Auto-Negotiation

Capability
General Guide
Configure Interface Auto-Negotiation
CL28/CL37
CL28 Limitation
CL73
Default FEC ability rule

Capability Support

DEVICE	CL73	UserGuide
N9200-64DC	√	AutoNego User Guide for N9200-64DC
N9500-128QC	√	AutoNego User Guide for N9500-128QC
N9500-64QC	√	AutoNego User Guide for N9500-64QC

General Guide

Configure Interface Auto-Negotiation

To enable auto-negotiation, you can specify a set of port speeds using the config interface advertised-speeds command that is referred to when auto-negotiating with the peer interface, or "all" speeds supported by the (port) interface if you do not configure the "advertised speeds". The command sequence can be interchanged, but the latest "advertised speed" command will replace the former "advertised speeds" in the sequence.

For example, in the following sequence, the final active advertised speed is (100M, 1G).

(1) configure advertised speeds (1G, 10G)

(2) enable autonego

(3) configure advertised speed (100M, 1G).

Similar behavior can also be applied to the config interface advertised-types command in CLI reference guide.

CL73

1. Enable Auto-Negotiation.
2. Config interface advertised-speeds.
3. Config interface advertised-types.
4. Note that the XCVR type shall be configured by the config interface advertised-types command, or the function may not work and the port may not link up.

Example of commands
admin@sonic:~$ sudo config interface autoneg Ethernet0 enabled admin@sonic:~$ sudo config interface advertised-speeds Ethernet0 100000,40000 admin@sonic:~$ sudo config interface advertised-types Ethernet0 CR4,KR4 admin@sonic:~$ sudo config interface advertised-types Ethernet0 all

Default FEC ability rule

Rule for default FEC ability for port advertised-speeds 100G/40G/25G/10G
RS-FEC is advertised if port speeds 25G or 100G are advertised. BASE-R FEC is advertised if port speeds 10G or 40G are advertised. No FEC is advertised if port speeds 40G and 100G are advertised. No FEC is advertised if port speeds 10G and 25G are advertised.

AutoNego User Guide for N9200-64DC

Capability

Breakout mode	Support adv-speeds	Support adv-types
1x400G	400000	CR8, KR8, LR8, SR8
2x200G	200000	CR4, KR4, LR4, SR4
4x100G	100000	CR2, KR2, SR2

Limitation：

The AutoNego is not supported at the 10G interface(Ethernet512, Ethernet513).

IP Configuration

IPv4/IPv6 Interfaces and Routes

You can specify IPv4/IPv6 subnets on a switch by manually or dynamically assigning IPv4/IPv6 addresses to port or VLAN interfaces. To specify IP subnets not directly connected to a switch, you can either configure static routes, or use dynamic routing protocols to identify routes that lead to other subnets by exchanging protocol messages with other routers on the network.

To display IPv4 or IPv6 routes that are currently in the routing table, use the show ip route or show ipv6 route commands.

Example

```
admin@sonic:~$ show ip route
Codes: K - kernel route, C - connected, S - static, R - RIP,
       O - OSPF, I - IS-IS, B - BGP, E - EIGRP, N - NHRP,
       T - Table, v - VNC, V - VNC-Direct, A - Babel, D - SHARP,
       F - PBR, f - OpenFabric,
       > - selected route, * - FIB route, q - queued route, r - rejected route

S>* 0.0.0.0/0 [200/0] via 10.11.162.254, eth0
C>* 1.1.0.0/16 is directly connected, Vlan100
C>* 10.1.1.0/31 is directly connected, Ethernet112
C>* 10.1.1.2/31 is directly connected, Ethernet116
C>* 10.11.162.0/24 is directly connected, eth0

admin@sonic:~$ show ipv6 route
Codes: K - kernel route, C - connected, S - static, R - RIPng,
       O - OSPFv3, I - IS-IS, B - BGP, N - NHRP, T - Table,
       v - VNC, V - VNC-Direct, A - Babel, D - SHARP, F - PBR,
       f - OpenFabric,
       > - selected route, * - FIB route, q - queued route, r - rejected route

C>* 2018:2001::/126 is directly connected, Ethernet112
C>* 2018:2002::/126 is directly connected, Ethernet116
C>* fc00:2::102/128 is directly connected, eth0
C * fe80::/64 is directly connected, Vlan100
```

```
C * fe80::/64 is directly connected, Ethernet112
C * fe80::/64 is directly connected, Ethernet116
C * fe80::/64 is directly connected, Bridge
C * fe80::/64 is directly connected, PortChannel0011
C>* fe80::/64 is directly connected, eth0
```

To display details about all the Layer3 interfaces on a switch for which IPv4/IPv6 addresses have been assigned, use the show ip interfaces or show ipv6 interfaces command.

The type of interfaces that can be assigned IPv4/IPv6 addresses include the following.

- Front panel physical ports
- PortChannel interfaces
- VLAN interfaces
- Loopback interfaces
- docker interface (IP only)
- management interface

Example

```
admin@sonic:~$ show ip interfaces
Interface         Master         IPv4 address/mask    Admin/Oper    BGP Neighbor    Neighbor IP    Flags
-------
Loopback0                        1.0.0.1/32           up/up         N/A             N/A
Loopback11        Vrf-red        11.1.1.1/32          up/up         N/A             N/A
Loopback100       Vrf-blue       100.0.0.1/32         up/up         N/A             N/A
PortChannel01                    10.0.0.56/31         up/down       DEVICE1         10.0.0.57
PortChannel02                    10.0.0.58/31         up/down       DEVICE2         10.0.0.59
PortChannel03                    10.0.0.60/31         up/down       DEVICE3         10.0.0.61
PortChannel04                    10.0.0.62/31         up/down       DEVICE4         10.0.0.63
Vlan100           Vrf-red        1001.1.1/24          up/up         N/A             N/A
Vlan1000                         192.168.0.1/27       up/up         N/A             N/A
docker0                          240.127.1.1/24       up/down       N/A             N/A
eth0                             10.3.147.252/23      up/up         N/A             N/A
lo                               127.0.0.1/8          up/up         N/A             N/A

admin@sonic:~$ show ipv6 interfaces
Interface         Master     IPv6 address/mask                     Admin/Oper    BGP Neighbor
Neighbor IP
                  --------   ------------------------------------   ------------  --------------
```

```
-------------
Bridge                fe80::7c45:1dff:fe08:cdd%Bridge/64       up/up       N/A          N/A
Loopback11      Vrf-red   1100::1/128                          up/up
PortChannel01           fc00::71/126                         up/down     DEVICE1      fc00::72
PortChannel02           fc00::75/126                         up/down     DEVICE2      fc00::76
PortChannel03           fc00::79/126                         up/down     DEVICE3      fc00::7a
PortChannel04           fc00::7d/126                         up/down     DEVICE4      fc00::7e
Vlan100         Vrf-red   100::1/112                         up/up       N/A          N/A
                                fe80::eef4:bbff:fefe:880a%Vlan100/64
eth0                fe80::eef4:bbff:fefe:880a%eth0/64        up/up       N/A          N/A
lo                  fc00:1::32/128                           up/up       N/A          N/A
```

For more information on IPv4/IPv6 interfaces and routes, see the (202111) IP/IPv6 section in the CLI reference guide.

Neighbor Discovery Protocol (NDP)

Neighbor Discovery Protocol (NDP) is one of the protocols used in IPv6 to manage local area networks, which helps nodes discover nodes near them, as well as manage network address resolution (ND) and status lookup. NDP is mainly used for address resolution, router discovery, prefix discovery, and parameter automatic configuration in IPv6.

Configuring the ND Agent

Configuring the ARP proxy also enables you to configure the NDP proxy. Currently, you cannot configure the ND proxy separately. You can configure the NDP proxy only by configuring the ARP proxy.config vlan proxy_arp vid enabled config vlan proxy_arp vid disabled ARP proxy is disabled by default and can be configured only on vlan interfaces.

example：

```
admin@sonic:~$ sudo config vlan proxy_arp 10 enabled
admin@sonic:~$ sudo config vlan proxy_arp 10 disabled
```

You can run the show vlan brief command to view the ARP proxy configuration.

example：

```
admin@sonic:~$ show vlan brief
+-----------+----------------+-------------+-----------+---------+--------------+---------------------------+
|  VLAN ID | IP Address     | Ports       | Port      | Proxy   | DHCP Helper  | DHCP Relay Configuration  |
|          |                |             | Tagging   | ARP     | Address      |                           |
```

```
+==========+================+=============+==========+=========+==============
+============================+
|       10 | 10.10.10.254/24 | PortChannel1 | untagged  | enabled |             | Source Interface:         |
|          |                 | PortChannel2 | untagged  |         |             | Link Selection:           |
|          |                 |              |           |         |             | Server Vrf:               |
|          |                 |              |           |         |             | Server ID Override:       |
+----------+-----------------+--------------+-----------+---------+-------------+---------------------------+
```

Configuring ND Aging

config interface nd aging

The default ND aging time is 300 to 86400. If you have learned the ND, setting the aging time does not take effect immediately. The aging time takes effect only after the ND ages.

example：

```
admin@sonic:~$ sudo config interface nd aging Vlan10 10000
```

You can run the show interfaces nd {interface} command to view ND configurations on interfaces

example：

```
admin@sonic:~$ show interfaces nd
Interface     Aging(sec)   ND Host route
-----------  ------------  --------------
Ethernet0        1800      disabled
Vlan10           1800      enabled
admin@sonic:~$ show interfaces nd Vlan10
Interface     Aging(sec)   ND Host route
-----------  ------------  --------------
Vlan10           1800      enabled
```

The ND host route is configured

config vlan nd_host_route vid enabled

config vlan nd_host_route vid disabled

The ND host route is disabled by default and can be configured only on vlan interfaces. ND to host routes convert ND to K routes.

example：

```
admin@sonic:~$ sudo config vlan nd_host_route 10 enabled
admin@sonic:~$ sudo config vlan nd_host_route 10 disabled
```

You can run the show interfaces nd {interface} command to view ND configurations on interfaces.

example：

```
admin@sonic:~$ show interfaces nd
Interface     Aging(sec)   ND Host route
-----------  ------------  ---------------
Ethernet0        1800      disabled
Vlan10           1800      enabled
admin@sonic:~$ show interfaces nd Vlan10
Interface     Aging(sec)   ND Host route
-----------  ------------  ---------------
Vlan10           1800      enabled
```

For more information on NDP, see the (202111) NDP section in the CLI reference guide.

Static Anycast Gateway (SAG)

The SONiC Static Anycast Gateway (SAG) feature is a method of moving a subnet default gateway close to endpoints in VXLAN networks.

In a typical folded-Clos data center network, using a static anycast gateway means that all the leaf switches (VXLAN VTEPs) are configured with the same IPv4/IPv6 address and function as the gateway for all their attached host servers. An active SAG eliminates the need for some protocol traffic across the network fabric and enables host workload flexibility and efficiency.

When implementing SAG, it is important to note that host server VLANs are mapped to each VXLAN by the VNI number. A leaf switch only advertises its gateway IP address and MAC address to the attached host server VLANs, never to the network fabric. By this method, all the anycast gateway IPv4/IPv6 addresses, and MAC addresses can remain consistent across all leaf switch VLANs.

To configure SAG on a switch, use the config sag mac_address add command to configure the SAG MAC address (only one allowed). Then configure the SAG IPv4/IPv6 address on a specific VLAN interface and use the config interface sag ip add command. Use the config sag ipv4 enable or config sag

ipv6 enable command to enable the IPv4/IPv6 address on an interface. To display SAG interface settings, use the show sag , show sag ip , and show sag ipv6 commands.

Example

```
admin@sonic:~# sudo config sag mac_address add 00:11:22:33:44:55

admin@sonic:~# sudo config interface sag ip add Vlan100 192.168.1.1/24

admin@sonic:~# sudo config sag ipv4 enable

--------------------  --------------------
admin@sonic~#show sag ip
Vlan Interface Name    IPv4 address/mask

Vlan100              192.168.1.1/24
-----------------  ------  ------
admin@sonic:~#show sag
Static Anycast Gateway Information

MacAddress         IPv4    IPv6

00:11:22:33:44:55  enable  enable
```

For more information, see the (202111) SAG section in the CLI reference guide.

Layer 2

ARP

The Address Resolution Protocol (ARP) is a TCP/IP protocol that obtains physical addresses based on IP addresses. When the host sends the message, it broadcasts the ARP request containing the target IP address to all hosts on the LAN and receives the return message to determine the physical address of the target. After receiving the return message, the IP address and physical address are stored in the local ARP cache and reserved for a certain period of time. The ARP cache is directly queried to save resources on the next request. The address resolution protocol is based on the mutual trust of hosts on the network. Hosts on the LAN can automatically send ARP reply messages. When receiving reply packets, other hosts record them in the local ARP cache without checking the authenticity of the packets.

- Configuring Static ARP
- Config interface arp add
- Config interface arp remove

example：

```
admin@sonic:~$ sudo config interface arp add Vlan10 10.10.10.10 a8:eb:d3:36:a5:82
admin@sonic:~$ sudo config interface arp remove Vlan10 10.10.10.10
```

The show arp {-if interface} command displays all dynamic and static ARP entries.

example：

```
admin@sonic:~$ show arp
Address         MacAddress         Iface         Vlan
--------------  -----------------  ------------  ------
10.10.10.1      e8:eb:d3:36:a5:87  PortChannel1  10
10.10.10.2      e8:eb:d3:36:c9:47  PortChannel2  10
30.30.30.2      68:21:5f:ff:67:80  Ethernet0     -
172.21.170.254  c4:a5:03:37:2f:96  eth0          -
Total number of entries 4
admin@sonic:~$ show arp -if Vlan10
Address     MacAddress         Iface         Vlan
----------  -----------------  ------------  ------
10.10.10.1  e8:eb:d3:36:a5:87  PortChannel1  10
```

```
10.10.10.2  e8:eb:d3:36:c9:47  PortChannel2      10
Total number of entries 2
```

Configuring ARP Aging

Config interface arp aging

The default ARP aging time is 300 to 86400. If you have learned the ARP, setting the ARP aging time does not take effect immediately. The aging time takes effect only after ARP aging.

example：

```
admin@sonic:~$ sudo config interface arp aging Vlan10 10000
```

You can run the show interfaces arp {interface} command to view ARP configurations on interfaces.

example：

```
admin@sonic:~$ show interfaces arp
Interface    Aging(sec)   Gratuitous ARP    ARP Host Route
-----------  ------------  ----------------  ----------------
Ethernet0       1800      disabled          disabled
Vlan10          1800      disabled          disabled
admin@sonic:~$ show interfaces arp Vlan10
Interface    Aging(sec)   Gratuitous ARP    ARP Host Route
-----------  ------------  ----------------  ----------------
Vlan10          1800      disabled          disabled
```

Configuring gratuitous ARP

Config interface arp gratuitous enable

Config interface arp gratuitous disable

By default, gratuite-based ARP is disbale. After gratuite-based ARP is enabled, all received gratuite-based ARP packets can be added to the ARP cache.

example：

```
admin@sonic:~$ sudo config interface arp gratuitous enable Vlan10
admin@sonic:~$ sudo config interface arp gratuitous disable Vlan10
```

You can run the show interfaces arp {interface} command to view ARP configurations on interfaces.

example：

```
admin@sonic:~$ show interfaces arp
Interface    Aging(sec)   Gratuitous ARP    ARP Host Route
-----------  ------------  ----------------  ----------------
```

```
Ethernet0       1800     disabled         disabled
Vlan10          1800     disabled         disabled
admin@sonic:~$ show interfaces arp Vlan10
Interface     Aging(sec)   Gratuitous ARP    ARP Host Route
-----------  ------------  ----------------  ----------------
Vlan10          1800     disabled         disabled
```

Configure ARP host routes

config vlan arp_host_route vid enabled

config vlan arp_host_route vid disabled

By default, ARP host routes are disabled and can be configured only on vlan interfaces. ARP to host routes convert ARP to K routes.

example：

```
admin@sonic:~$ sudo config vlan arp_host_route 10 enabled
admin@sonic:~$ sudo config vlan arp_host_route 10 disabled
```

You can run the show interfaces arp {interface} command to view ARP configurations on interfaces.

example：

```
admin@sonic:~$ show interfaces arp
Interface     Aging(sec)   Gratuitous ARP    ARP Host Route
-----------  ------------  ----------------  ----------------
Ethernet0       1800     disabled         disabled
Vlan10          1800     disabled         disabled
admin@sonic:~$ show interfaces arp Vlan10
Interface     Aging(sec)   Gratuitous ARP    ARP Host Route
-----------  ------------  ----------------  ----------------
Vlan10          1800     disabled         disabled
```

Configuring ARP Proxy

Config vlan proxy_arp vid enabled

Config vlan proxy_arp vid disabled

ARP proxy is disabled by default and can be configured only on vlan interfaces.

example：

```
admin@sonic:~$ sudo config vlan proxy_arp 10 enabled
admin@sonic:~$ sudo config vlan proxy_arp 10 disabled
```

You can run the show vlan brief command to view the ARP proxy configuration.

example：

```
admin@sonic:~$ show vlan brief
+-----------+----------------+-------------+-----------+---------+--------------+---------------------------+
|  VLAN ID | IP Address      | Ports       | Port      | Proxy   | DHCP Helper  | DHCP Relay Configuration  |
|          |                 |             | Tagging   | ARP     | Address      |                           |
+===========+=================+=============+===========+=========+==============
+============================+
|       10 | 10.10.10.254/24 | PortChannel1 | untagged  | enabled |              | Source Interface:         |
|          |                 | PortChannel2 | untagged  |         |              | Link Selection:           |
|          |                 |              |           |         |              | Server Vrf:               |
|          |                 |              |           |         |              | Server ID Override:       |
+-----------+----------------+-------------+-----------+---------+--------------+---------------------------+
```

Configuring DHCP Relay

In some network environments, the DHCP server is located in another segment that is outside of the broadcast segment. In that case, the network administrator needs to configure the DHCP relay to forward DHCP requests to the DHCP server in the remote segment, and to re-direct DHCP offers to DHCP clients.

To configure the DHCP relay, the network administrator needs to select an interface (VLAN) from which DHCP requests are received, and the server that services those DHCP requests on the interface (VLAN). The following is an example of configuring a DHCP relay on an interface (VLAN 1000) and specifying the DHCP server address 7.7.7.7. For detailed information, refer to the (202111) DHCP Relay section in the CLI reference guide.

NOTE: The sub port interface does not support use with DHCP relay service.

Example
admin@sonic:~$ sudo config vlan dhcp_relay add 1000 7.7.7.7 Added DHCP relay destination address 7.7.7.7 to Vlan1000 Restarting DHCP relay service.. admin@sonic:~$ sudo config vlan dhcp_relay del 1000 7.7.7.7 Removed DHCP relay destination address 7.7.7.7 from Vlan1000 Restarting DHCP relay service... Running command: systemctl restart dhcp_relay

DHCP relay over different VRFs

The DHCP server will determine the destination IP of the DHCP offer based on the Relay agent IP address field(IP address of the client interface) of the DHCP discover packet. But the socket cannot receive the DHCP offer in server-VRF because the destination IP of the DHCP offer is not included in server-VRF. Therefore, need to change the Relay agent IP address to server-VRF through the source interface command and enable the link-selection sub-option.

The link-selection sub-option of the Agent information option for the DHCP is used by any DHCP relay agent that desires to specify a subnet/link for a DHCP client request that it is relaying but needs the subnet/link specification to be different from the IP address the DHCP server should use when communicating with the relay agent. Therefore, the link-selection sub-option specifies an IP address that determines a subnet on which the DHCP client is located, and the relay agent IP address field can be used to communicate with the relay agent.

The source interface is used to specify the Relay agent IP address field and the source IP address of the DHCP discover relayed packet.

Before enabling the link selection, need to use the source interface command to specify the Relay agent IP address and the source IP address of DHCP discover that belongs to server-VRF.

Example

```
admin@sonic:~$ sudo config vlan dhcp_relay src_intf add 1000 PortChannel0001
Added DHCP relay source interface PortChannel0001 for Vlan1000
Restarting DHCP relay service...

admin@sonic:~$ sudo config vlan dhcp_relay link_selection add 1000
Enable DHCP relay link selection for Vlan1000
Restarting DHCP relay service...

admin@sonic:~$ show ip interfaces
Interface       Master    IPv4 address/mask    Admin/Oper    BGP Neighbor    Neighbor IP
--------------  --------  -------------------  ------------  --------------  -------------
Loopback0       Vrf1      10.1.0.32/32         up/up         N/A             N/A
PortChannel0001           10.0.0.56/31         up/up         N/A             10.0.0.57
Vlan1000                  192.168.0.1/21       up/up         ARISTA01T1      N/A
docker0                   240.127.1.1/24       up/down       N/A             N/A
eth0                      10.250.0.196/16      up/up         N/A             N/A
lo                        127.0.0.1/16         up/up         N/A             N/A
```

```
admin@sonic:~$ show vlan brief
+-----------+----------------+------------+---------------+--------------+----------------+-------------
+-------------+
|  VLAN ID | IP Address      | Ports      | Port Tagging  | DHCP Helper  | DHCP Source    | DHCP
Link
| Proxy ARP   |
|          |                 |            |               | Address      | Interface      | Selection
|          |
+===========+=================+============+===============+==============+================+===
==============+=============+
==============+
|       1000 |  192.168.0.1/21  | Ethernet4  | untagged      | 10.0.0.57    | PortChannel0001 |  enabled
| disabled    |
+-----------+----------------+------------+---------------+--------------+----------------+-------------
+-------------+
```

Configuring interface TPID

TPID is used to identify whether the frame received is a tagged or un tagged frame. Default TPID is 0x8100.

The primarily usage of SONIC TPID configuration is to configure HW ASIC to treat non-matching TPID VLAN tags as customer payload and preserve the VLAN TAG, the control plane traffic will not be handled.

The TPID value only allows 0x8100, 0x9100, 0x9200, and 0x88A8.

To change the TPID value for a physical interface or port-channel interface, use the config interface tpid command.

Example

```
admin@sonic:~$ sudo config interface tpid Ethernet0 0x88a8
```

Once TPID is configured, use show interfaces tpid command to check the configuration.

Example

```
admin@sonic:~$ show interfaces tpid
  Interface        Alias    Oper    Admin    TPID
```

```
-----------  -------------  ------  -------  ------
  Ethernet0    Eth1(Port1)      up       up  0x88A8
```

For more information, see the (202111) Interfaces (Updated) section in the CLI reference guide.

Note：

1. Currently Linux kernel only supports default 802.1Q TPID 0x8100, SONiC TPID configuration does NOT change Linux kernel's TPID value.
2. Dynamic port channel is formed by LACP protocol and the LACP packet is handled by Linux kernel, changing TPID for the dynamic port channel will make the LACP packet not recognized and cause the port channel interface to down.

Control Plane Policing (CoPP)

Control Plane Policing (CoPP) allows users to manage and rate limit control plane packets that go to the route processor. The CoPP configuration file is composed of one or more trap groups. Each trap group includes one or more predefined trap IDs that are mapped to known protocols (STP, LACP, EAPOL, etc.), and a meter (if required) decides how to process the specific packets (forward, trap to CPU, etc.).

The following are the supported trap IDs:

stp
lacp
eapol
lldp
pvrst
igmp_query
sample_packet
udld
switch_cust_range
arp_req
arp_resp
dhcp
ospf
pim
dhcpv6
neigh_discovery
src_nat_miss

dest_nat_miss
isis
arp_suppression
nd_suppression
ip2me
ssh
snmp
bgp
bgpv6
bfd
bfdv6
l3_mtu_error
ttl_error

The following shows an example of a CoPP configuration file:

Example

```
{
    "COPP_TABLE:trap.group.nat": {
        "trap_ids": "src_nat_miss,dest_nat_miss",
        "trap_action": "trap",
        "trap_priority": "1",
        "queue": "1",
        "meter_type": "packets",
        "mode": "sr_tcm",
        "cir": "600",
        "cbs": "600",
        "red_action": "drop"
    },
    "OP": "SET"
}
```

You can update the CoPP configuration by updating the configuration file, which is /usr/share/sonic/templates/copp.json.j2 located in the swss container, and then use config reload to apply the new configuration.

Example

```
admin@sonic:~$ docker exec -it swss bash
root@sonic:/# vi /usr/share/sonic/templates/copp.json.j2
root@sonic:/# exit
exit
admin@sonic:~$ config reload -y
```

For more information, see the (202111) CoPP section in the CLI reference guide.

Creating VLANs

In large networks, you can isolate traffic into separate broadcast domains by grouping network nodes into Layer 2 VLANs. VLANs provide greater network efficiency by reducing broadcast traffic, allowing you to make network changes without updating IP addresses or IP subnets. VLANs inherently provide a high level of network security since traffic must pass through a configured Layer 3 link to reach a different VLAN.

To create a VLAN, first, configure a VLAN ID using the config vlan add command, and then add port members using the config vlan member add command.

Example

```
admin@sonic:~$ sudo config vlan add 100

admin@sonic:~$ sudo config vlan member add 100 Ethernet4
```

To display information about configured VLANs, use the show vlan brief command or the show vlan config command.

Example

```
admin@sonic:~$ show vlan brief
+-----------+----------------+------------+---------------+-------------+---------------+---------------
+-------------+
|  VLAN ID | IP Address      | Ports      | Port Tagging  | Proxy ARP   | DHCP Helper   | DHCP
Source    |
DHCP Link   |
|         |                 |            |               |             | Address       | Interface     |
Selection   |
+===========+=================+============+===============+=============+=====
==========+===============+==
===========+
```

```
|        100   1.1.2.2/16         Ethernet0    tagged           disabled     | 192.0.0.1
|              |                                                                |
|                                 Ethernet4    tagged                        | 192.0.0.2
|              |                                                                |
|                                                                               | 192.0.0.3
|              |                                                                |
|                                                                               |
+----------+----------------+-----------+---------------+------------+--------------+--------------
+------------+

admin@sonic:~$ show vlan config
Name       VID  Member      Mode

Vlan100     100  Ethernet0    tagged
Vlan100     100  Ethernet4    tagged
```

Config vlan mtu

Restriction:

1. Range of MTU config:
    a. Maximum of MTU is 9216
    b. Minimum of MTU is 1500
2. sag interface and vrrp interface is created from vlan interface. The mtu of sag interface and vrrp interface would also be changed as the same value as vlan mtu if v lan mtu is changed.
3. The config of vlan mtu would not change mtu of its member ports. When user config vlan mtu, sonic would not provide warning message if v lan mtu is larger than mtu of member port. User needs to check that the size of vlan mtu should be smaller or equal to all of its member ports.

    Otherwise the network may not be reachable.

    a. For example, the topology is configured as follows Figure1.

        - In DUT1, mtu of Vlan100 is 9100 and mtu of Ethernet52(vlan member of Vlan100) is 1500. The ip of Vlan100 is 10.10.100.38
        - In DUT2, mtu of Vlan100 is 9100 and mtu of Ethernet56(vlan member of Vlan100) is 1500. The ip of Vlan100 is 10.10.100.39
        - If ping from DUT1 with packet size 2000 to DUT2, the ping result would be failed

```
admin@DUT1:~$ ping 10.10.100.39 -s 2000 -c 10
PING 10.10.100.39 (10.10.100.39) 2000(2028) bytes of data.
--- 10.10.100.39 ping statistics ---
10 packets transmitted, 0 received, 100% packet loss, time 9211ms
```

- But if changed mtu of Vlan100 in both DUT1 and DUT2 to 1500, the ping can work

```
admin@DUT1:~$ ping 10.10.100.39 -s 2000 -c 10
PING 10.10.100.39 (10.10.100.39) 2000(2028) bytes of data.
2008 bytes from 10.10.100.39: icmp_seq=1 ttl=64 time=0.500 ms
2008 bytes from 10.10.100.39: icmp_seq=2 ttl=64 time=0.837 ms
2008 bytes from 10.10.100.39: icmp_seq=3 ttl=64 time=0.448 ms
2008 bytes from 10.10.100.39: icmp_seq=4 ttl=64 time=0.443 ms
2008 bytes from 10.10.100.39: icmp_seq=5 ttl=64 time=0.432 ms
2008 bytes from 10.10.100.39: icmp_seq=6 ttl=64 time=0.403 ms
2008 bytes from 10.10.100.39: icmp_seq=7 ttl=64 time=0.391 ms
2008 bytes from 10.10.100.39: icmp_seq=8 ttl=64 time=0.458 ms
2008 bytes from 10.10.100.39: icmp_seq=9 ttl=64 time=0.464 ms
2008 bytes from 10.10.100.39: icmp_seq=10 ttl=64 time=0.471 ms

--- 10.10.100.39 ping statistics ---
10 packets transmitted, 10 received, 0% packet loss, time 9205ms
rtt min/avg/max/mdev = 0.391/0.484/0.837/0.121 ms
```

	MTU
Vlan100	9100
Ethernet52	1500

	MTU
Vlan100	9100
Ethernet56	1500

Figure 1

Default setting:

- MTU is 9100 by default

Procedure ：

Suppose that Ethernet52 is vlan member of Vlan100. The following example is shown how to config port mtu and vlan mtu.

Step 1. Configure MTU 1500 on Ethernet52.

```
sudo config interface mtu Ethernet52 1500
```

Step 2. Config MTU 1500 on Vlan100

```
sudo config vlan mtu 100 1500
```

Step 3. Check MTU

```
admin@sonic:~$ show vlan brief --mtu
+-----------+------------+------------+------------+
|   VLAN ID |    VLAN MTU  | Ports      |    Port MTU  |
+===========+============+============+============+
|        100 |        1500  | Ethernet52 |        1500  |
+-----------+------------+------------+------------+
```

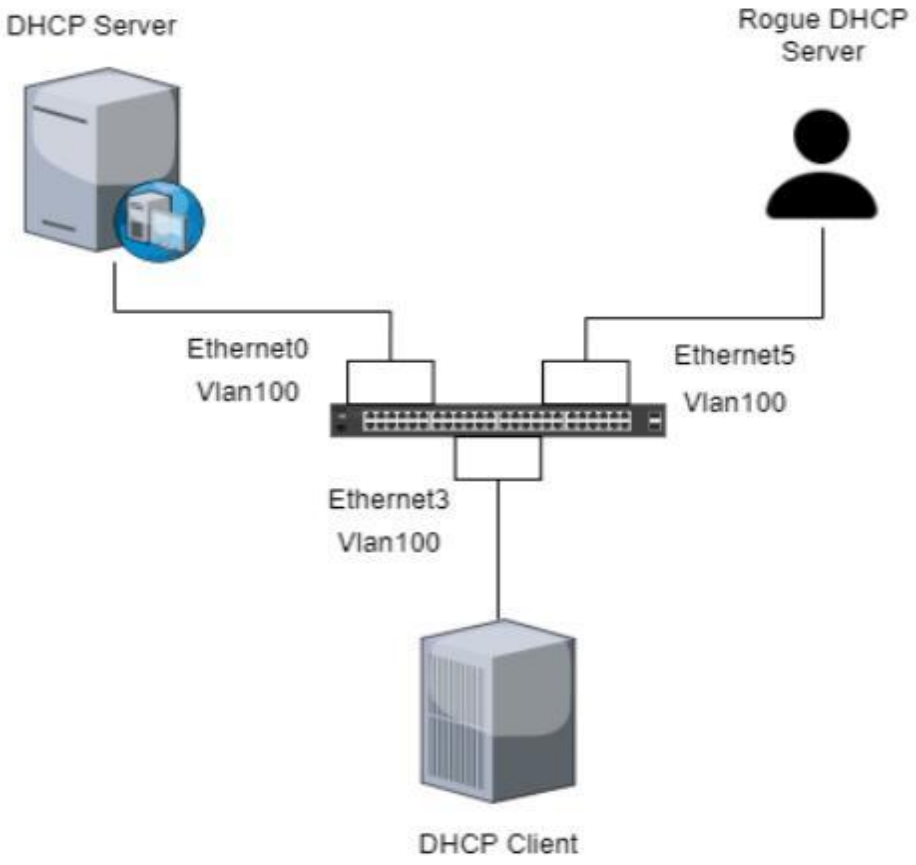
For more information, see the (202111) VLAN (Update) section in the CLI reference guide.

DHCP Snooping User Guide

Introduction

DHCP Snooping allows a switch to protect a network from rogue DHCP servers and its a DHCP security feature that provides security by filtering un-trusted DHCP packets. This user guide provides brief instructions on how to configure and use DHCP snooping in SONiC system.

Scenario



1. Enable dhcp snooping feature

Step 1.
admin@sonic:~# sudo config feature state dhcp_snoop enabled

2. Enable ipv4 for dhcp snooping

Step 2.
admin@sonic:~# sudo config dhcp-snoop ipv4 enable

3. Enable vlan for dhcp snooping

Step 3.
admin@sonic:~# sudo config dhcp-snoop vlan enable 100

4. add trusted port for dhcp snooping

Step 4.
admin@sonic:~# sudo config dhcp-snoop trusted add Ethernet0

Limitation

- DHCP snooping only support interface which is the member of vlan.

- DHCP snooping only support dhcp ipv4.
- DHCP snooping does not support in MCLAG environment.
- DHCP snooping does not support the management interface.
- DHCP snooping does not support VxLAN tunnel.

Forwarding Database (FDB)

Switches store the addresses for all known devices. This information is used to forward traffic directly between the inbound and outbound ports. All the addresses learned by monitoring traffic are stored in the dynamic address table, or forwarding database (FDB).

To display the MAC (FDB) entries either in full or partial (by VLAN or port), use the show mac command. To clear the FDB table of all entries, use the sonic-clear fdb all command.

Example

```
admin@sonic:~$ show mac
No.  Vlan  MacAddress         Port          Type
---  ----  -----------------  ------------  -------
  1  1000  E2:8C:56:85:4A:CD  Ethernet192  Dynamic
  2  1000  A0:1B:5E:47:C9:76  Ethernet192  Dynamic
  3  1000  AA:54:EF:2C:EE:30  Ethernet192  Dynamic
  4  1000  A4:3F:F2:17:A3:FC  Ethernet192  Dynamic
  5  1000  0C:FC:01:72:29:91  Ethernet192  Dynamic
  6  1000  48:6D:01:7E:C9:FD  Ethernet192  Dynamic
  7  1000  1C:6B:7E:34:5F:A6  Ethernet192  Dynamic
  8  1000  EE:81:D9:7B:93:A9  Ethernet192  Dynamic
  9  1000  CC:F8:8D:BB:85:E2  Ethernet192  Dynamic
 10  1000  0A:52:B3:9C:FB:6C  Ethernet192  Dynamic
 11  1000  C6:E2:72:02:D1:23  Ethernet192  Dynamic
 12  1000  8A:C9:5C:25:E9:28  Ethernet192  Dynamic
 13  1000  5E:CD:34:E4:94:18  Ethernet192  Dynamic
 14  1000  7E:49:1F:B5:91:B5  Ethernet192  Dynamic
 15  1000  AE:DD:67:F3:09:5A  Ethernet192  Dynamic
 16  1000  DC:2F:D1:08:4B:DE  Ethernet192  Dynamic
 17  1000  50:96:23:AD:F1:65  Ethernet192  Dynamic
                              Dynamic
```

```
18      1000        C6:C9:5E:AE:24:42    Ethernet192
Total number of entries 18

admin@sonic:~$ sonic-clear fdb all
FDB entries are cleared.
```

Static FDB Guidelines

Static FDB CLI commands allow users to manage static FDB entries through the configuration of the FDB table. A typical FDB entry includes the MAC address, VLAN ID, and interface information associated with it. The example shown below illustrates the way to add and delete a static MAC address.

Prerequisites

The interface should be configured as a member in one of VLANs on the network devices.

Static FDB Example

Execute the command below to add a static FDB entry with mac address "00:10:3a:2b:05:67" with VLAN 1000 on port "Ethernet0".

Example command to add static mac

```
root@sonic:/# config static-mac add 00:10:3a:2b:05:67 100 Ethernet0
```

After the static FDB entry is added, the entry can be found from the output of "show mac".

Example command for show mac

```
admin@sonic:~$ show mac
No.  Vlan  MacAddress        Port  Type

1    100   00:10:3a:2b:05:67  Ethernet0  Static
Total number of entries 1
```

If the static FDB entry needs to be removed, execute the command below.

Example command to delete static mac

```
root@sonic:/# config static-mac del 00:10:3a:2b:05:67 100
```

Notice

- Some features such as EVPN and MCLAG will program FDB entries. The FDB entries configured by EVPN and MCLAG will be replaced by configuring the static FDB entry with the same (MAC, VLAN). It is expected that the protocol might run into problems under this scenario. So it is strongly not recommended to configure static FDB entries when the features like EVPN and MCLAG are running.

- The maximum number of the user-configured static FDB entries is 1024.

For more information, see the (202111) FDB section in the CLI reference guide.

MAC Aging Time

MAC aging is enabled by default with a default time of 600 seconds. When the aging time is configured as 0, MAC aging is disabled. The configurable range is from 0 to 1000000 seconds. Globally, there is only one aging time for dynamic MAC address table entries, and subsequent executions of the mac aging-time command take the last configuration as authoritative.

Config aging time
Configure the MAC aging time to 300 seconds.
Example command to set aging time
root@sonic:/# config mac aging-time 300
Disable MAC address aging.
Example command to disable aging time
root@sonic:/# config mac aging-time 0
Show MAC address aging.
Example command to show aging time
root@sonic:/# show mac aging-time

IGMP Snooping User Guide

Introduction
Preparation and Prerequisites
Getting Started with IGMP Snooping
Enabling IGMP Snooping
Enabling IGMP Snooping on a VLAN
Active the IGMP Snooping on a VLAN
Add a port to a multicast group
Limitations

Introduction

IGMP Snooping is a feature that allows a network switch to listen in on the IGMP conversation between hosts and routers. The switch can then determine which ports to send multicast traffic to, based on the IGMP messages it receives.

This allows the switch to send multicast traffic only to the ports that have requested it, instead of flooding all ports with the traffic.

Preparation and Prerequisites

This guide assumes that you understand networking concepts and protocols, including the Internet Group Management Protocol (IGMP).

Getting Started with IGMP Snooping

Enabling IGMP Snooping

To enable IGMP Snooping in SONiC, you must first enable the IGMP Snooping daemon. To do this, run the following command:

```
admin@sonic:~$ sudo config igmp snooping enable
```

This will enable the IGMP Snooping globally and start make IGMP packets to be delivered to the daemon.Enabling IGMP Snooping on a VLAN

To enable IGMP Snooping on a VLAN, run the following command:

```
sudo config igmp snooping vlan enable 100
```

This will enable IGMP Snooping on VLAN 100, and the default version is 2.

Active the IGMP Snooping on a VLAN

The IGMP Snooping service will not process the IGMP packets until the VLAN is activated. To activate the VLAN, there are 2 ways:

1. Enable the IGMP Snoping Querier globally, and the VLAN will be activated automatically. To do this, run the following command:

   ```
   admin@sonic:~$ sudo config igmp snooping querier enable
   ```

2. Be sure the multicast router port is present and link up on the VLAN. There are 2 ways to add the multicast router port to the VLAN:
    a. Receive the IGMP Query packet from a port on the VLAN. Then the port will act as the multicast router port automatically.
    b. Add the multicast router port to the VLAN. To do this, run the following command:

```
admin@sonic:~$ sudo config igmp snooping vlan mrouter-port add 100 Ethernet0
```

To check the status of IGMP Snooping on a VLAN, run the following command:

```
show igmp snooping
```

The IGMP Snooping Running Status will show the status of IGMP Snooping on a VLAN.

Add a port to a multicast group

After the IGMP Snooping is active on a VLAN, the IGMP reports will be processed. The IGMP Snooping service will learn the IGMP reports from the hosts and add the host to the multicast group.

The multicast group will be added to the VLAN and the multicast router port will be added to the group automatically.

To check the IGMP Snooping group member, run the following command:

```
admin@sonic:~$ show igmp snooping groups
```

While the IP mulitcast traffic is received in the VLAN, it will forward the traffic to the group members. The multicast traffic will be forwarded to the multicast router port if there is no group member on the VLAN.

Limitations

1. IGMPv3 protocol support is limited. It currently only supports IGMPv3 ASM mode with supported record types including MODE_IS_EXCLUDE, CHANGE_TO_INCLUDE_MODE, and CHANGE_TO_EXCLUDE_MODE.
2. The protocol does not support VxLAN overlay
3. The protocol does not support VLAN stacking (QinQ)
4. The protocol does not support MCLAG. The MCLAG peer switch cannot learn IGMP information from the ICCP TLV.

Link Aggregation (LAG)

LAG (Link Aggregation) combine the bandwidth of multiple Ethernet ports into a single logical link. A link Aggregation Group (LAG) is also referred to as a port channel.

To create a port channel, first configure a port channel name using the config portchannel add command, and then add members using the config portchannel member add command.

The port channel name should have prefix 'PortChannel' and suffix '<0-9999>'. For example, "PortChannel200".

Example

```
admin@sonic:~$ sudo config portchannel add PortChannel200

admin@sonic:~$ sudo config portchannel member add PortChannel200 Ethernet4
```

To display information about configured port channel, use the show interfaces portchannel command.

Example

```
admin@sonic:~$ show interfaces portchannel
Flags: A - active, I - inactive, Up - up, Dw - Down, N/A - not available,
       S - selected, D - deselected, * - not synced,
       M - mixed speed

  No.  Team Dev         Protocol     Ports         Oper Key    Admin Key    Fast Rate

  200  PortChannel200   LACP(A)(Dw)  Ethernet4(D)  1200        auto         false
```

For more information, see the (202111) Port Channels section in the CLI reference guide.

Link Layer Discovery Protocol (LLDP)

The Link Layer Discovery Protocol (LLDP) is used to discover basic information about neighboring devices in the local broadcast domain. LLDP is a Layer 2 protocol that uses periodic broadcasts to advertise information about the sending device. Advertised information is represented in Type Length Value (TLV) format according to the IEEE 802.1AB standard, and can include details such as device identification, capabilities, and configuration settings. LLDP also defines how to store and maintain information gathered about the neighboring network nodes it discovers. LLDP information can be used by applications to simplify troubleshooting, enhance network management, and maintain an accurate network topology.

You can use the show lldp table command to display a summary of discovered LLDP neighbors, or the show lldp neighbors command to display detailed information about LLDP neighbors.

Example

```
admin@sonic:~$ show lldp table
Capability codes: (R) Router, (B) Bridge, (O) Other
LocalPort    RemoteDevice    RemotePortID      Capability    RemotePortDescr
-----------  --------------  ----------------  ------------  ---------------
Ethernet112  ARISTA01T1       Ethernet1         BR
Ethernet116  ARISTA02T1       Ethernet1         BR
eth0                         00:e0:0c:00:01:05 BR            Ethernet Port on unit 1, port 8
Total entries displayed:  3
```

For more information, see the (202111) LLDP (old. backup 2111.7) section in the CLI reference guide.

Multi-Chassis Link Aggregation Group (MC-LAG)

What is MC-LAG (Multi-Chassis)

A Multi-Chassis Link Aggregation Group (MC-LAG) is a pair of links that terminates on two cooperating switches and appears as an ordinary link aggregation group (LAG). The cooperating switches are MC-LAG peer switches and communicate through an interface called a peer link. While the peer link’s primary purpose is to exchange MC-LAG control information between peer switches, it also carries data traffic from devices that are attached to only one MC-LAG peer and have no alternative path. An MC-LAG domain consists of the peer switches and the control links that connect these switches.

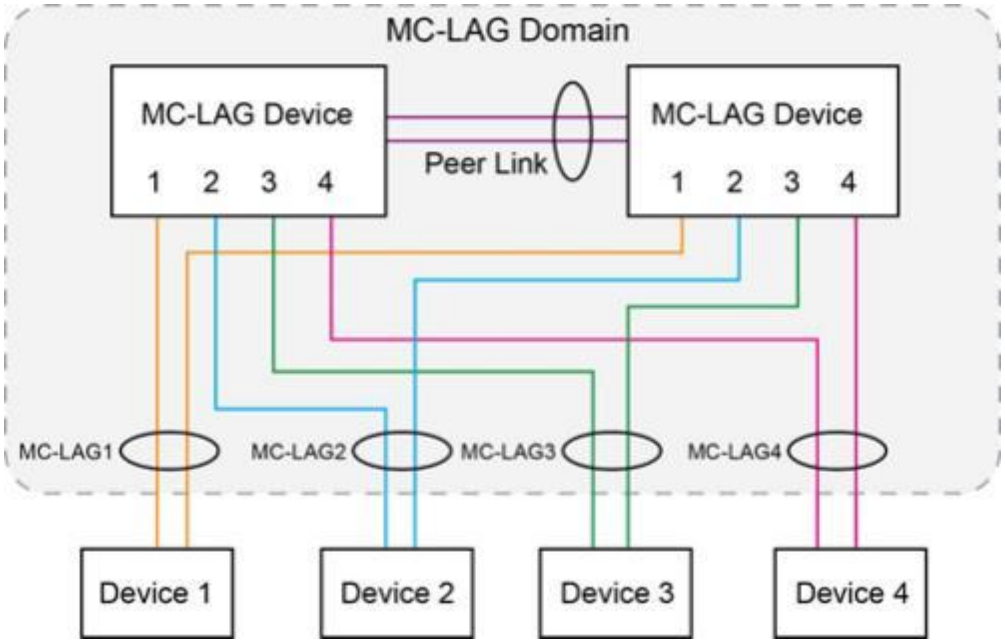


Figure 1: MC-LAG example

Note：

Peer Link: A link between the cooperating switches (or MC-LAG device)

Control Link : A link between peer switches, exchanging control information. It could be in a peer link if there is no extra link between MC-LAG devices.

Host device : Device which connects MC-LAG device with port-channel. eg Device 1, Device 2. Device 4.

Configure MC-LAG domain

Currently, there can be only one MC-LAG domain configured on a system and only configured port channels can be MC-LAG interfaces. To set up an MC-LAG domain, you need to configure two switch systems and their Peer Link connection.

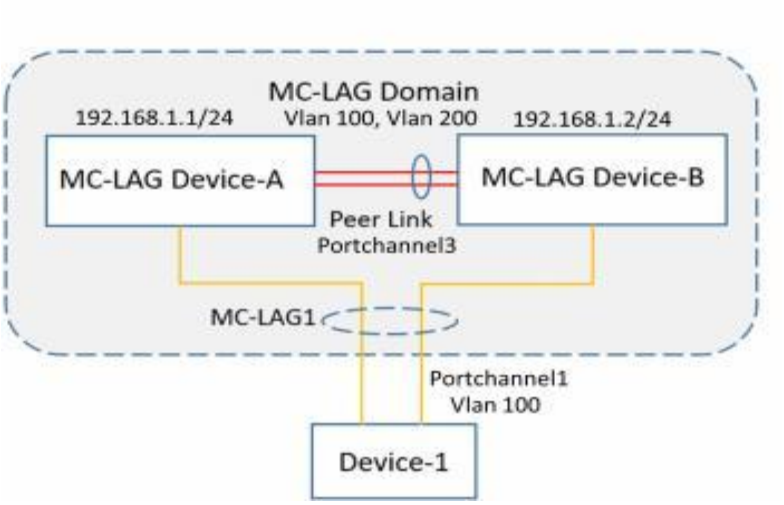


Figure 2: MC-LAG topology for configuration

Follow these basic steps for each MC-LAG switch:

Step 1: Create port channels for the host device (data forwarding) connections and the MC-LAG Peer Link connection between the switches.

Step 2: Create two VLANs, one for the MC-LAG peer keepalive link and the other for the data forwarding port channels.

- These two VLANs should be different.

Step 3: Add an IP address for the MC-LAG peer keepalive link.

Step 4: Create the MC-LAG domain by specifying the MC-LAG peer keepalive link IP addresses (local and peer device) and the peer port-channel interface name.

- If the keepalive link is set on the VLAN interface and a peer link port is added to this VLAN, need to enable unique IP for this VLAN.

Step 5: Add the data forwarding port channels as the MC-LAG members.

- Use the config mclag add command to configure the MC-LAG domain.
- Add the host device port-channel members using the config mc lag member add command.
- To display the MC-LAG configuration and status, use the show mc lag brief command.

An example of how to configure the MC-LAG domain referring to Figure 2 is as follows:

Example		
admin@sonic:~$	sudo	config portchannel add PortChannel1
admin@sonic:~$	sudo	config portchannel add PortChannel3
admin@sonic:~$	sudo	config vlan add 100
admin@sonic:~$	sudo	config vlan add 200
admin@sonic:~$	sudo	config vlan member add 100 PortChannel1
admin@sonic:~$	sudo	config vlan member add 100 PortChannel3
admin@sonic:~$	sudo	config vlan member add 200 PortChannel3
admin@sonic:~$	sudo	config mclag unique-ip add Vlan200
admin@sonic:~$	sudo	config interface ip add Vlan200 192.168.1.1/24
admin@sonic:~$	sudo	config mclag add 1 192.168.1.1 192.168.1.2 PortChannel3
admin@sonic:~$	sudo	config mclag member add 1 PortChannel1
admin@sonic:~$	sudo	mclag brief
admin@sonic:~$	show	

```
    Domain ID                    : 1
    Role                         : Active
    Session Status               : Up
    Peer Link Status             : Up
    Source Address               : 192.168.1.1
    Peer Address                 : 192.168.1.2
    Peer Link                    : PortChannel3
    Keepalive Interval           : 1 secs
    Session Timeout              : 15 secs
    System MAC                   : b8:6a:97:73:6c:96
    Number of MCLAG Interfaces   : 1
    MCLAG Interface              Local/Remote Status
    PortChannel1                 Up/Up
    PortChannel2                 Up/Up

admin@sonic:~$ show mclag unique-ip
        Unique IP:          : Vlan200

admin@sonic:~$ show vlan brief
+-----------+---------------+----------------+---------------+---------------+--------------
+-------------+-------------+
|   VLAN ID | IP Address    | Ports          | Port Tagging  | DHCP Helper   |  DHCP Source   | DHCP
Link   | Proxy ARP   |
                            |                |               | Address       |  Interface     |
|         |
                          |
Selection   |
+===========+================+================+===============+===============
+===============+============
=+==============+
|         100 |                | PortChannel1   | tagged        |                |
|               | disabled     |
                                                | tagged        |                |
|             |                | PortChannel3
|               |              |
```

```
+----------+---------------+----------------+---------------+--------------+--------------
+------------+------------+
|      200 | 192.168.1.1/24 | PortChannel3     | tagged         |                |
|            | disabled    |
+----------+---------------+----------------+---------------+--------------+--------------
+------------+------------+
```

Note：

- Peer-link PortChannel3, and MC-LAG enabled interface PortChannel1 are both members of Vlan100. MC-LAG peer keepalive link must belong to another Vlan200.
- Configure unique-ip on VLAN200 due to peer link PortChannel3 being added to this VLAN. To allow separate IP configuration on the VLAN200 interface for keepalive link.
- Both active and standby nodes require the same Unique IP configuration on the VLAN interface.

For more information, see the (202111) MC LAG section in the CLI reference guide.

Flush MAC behavior vs host moving over the port-channel.

In the current behavior, MAC addresses learned from peer MCLAG devices will be treated as static entries on the local MCLAG device, allowing them to move.

When a dynamic MAC address on a peer MCLAG interface gets aged out, the same static MAC will be changed to a dynamic type if traffic from the MAC is received on a local MCLAG device. Otherwise, the static MAC will also be deleted.

For details, please see the following example

Case 1: Flush MAC will change the status from dynamic to static and allow port change. (MAC learned from MCLAG)

Step 1: The data flows from Device-1 to Device-A, and the MAC of Device-1 is learned in Device-A (dynamic) and synced to Device-B (static).

Step 2: Device-A flushes the dynamic MAC and sends a delete message to Device-B.

Step 3: Device-B deletes the static MAC (learned from MCLAG), but if there is traffic flowing through Device-B at this time, it will change the status to dynamic MAC and notify Device-A.

Step 4: Device-A adds it as a static MAC. (learned from MCLAG).

Device-A

```
admin@leaf-1:~$ show mac
  No.    Vlan  MacAddress         Port           Type
    1     201  00:10:94:00:00:02  PortChannel10  Dynamic
Total number of entries 1
admin@leaf-1:~$
admin@leaf-1:~$ sudo sonic-clear fdb all
FDB entries are cleared.

admin@leaf-1:~$
admin@leaf-1:~$ show mac
  No.    Vlan  MacAddress         Port           Type
    1     201  00:10:94:00:00:02  PortChannel10  Static
Total number of entries 1
```

Case 2: Flush can clear the dynamic mac

Step 1: The data flows from Device-1 to Device-A, and the MAC of Device-1 is learned in Device-A (dynamic) and synced to Device-B (static).

Step 2: Device-A flushes the dynamic MAC and sends a delete message to Device-B.

Step 3: Device-B deletes the static MAC (learned from MCLAG), and if there is no traffic flowing through Device-B, Device-B is allowed to delete the static MAC.

Device-A

```
admin@leaf-1:~$ show mac
  No.    Vlan  MacAddress         Port           Type
    1     201  00:10:94:00:00:02  PortChannel10  Dynamic
Total number of entries 1
admin@leaf-1:~$
admin@leaf-1:~$ sudo sonic-clear fdb all
FDB entries are cleared.

admin@leaf-1:~$ show mac
  No.    Vlan  MacAddress         Port           Type

Total number of entries 0
admin@leaf-1:~$
```

PVST and STP

The Spanning Tree Protocol (STP) prevents Layer 2 loops in a network and provides redundant links. In the event of a primary link failure, the backup link takes over and network traffic is unaffected. With numerous pathways between the devices, STP also ensures that the least cost path is taken.

The pvst mode allows for running multiple instances of spanning trees on a per VLAN basis. One advantage of PVST is that it allows for load-balancing of the traffic.

To enable the spanning tree and specify the mode for the device, use the config spanning-tree command.

Example

```
admin@sonic:~$ config spanning-tree enable pvst
```

After creating a VLAN and a port is added to the VLAN, the spanning tree on the VLAN starts running.

Example

```
admin@sonic:~$ config vlan add 100
admin@sonic:~$ config vlan member add 100 Ethernet4
```

You can use the show spanning-tree command to display the running status.

Example

```
admin@sonic:~$ show spanning-tree
Spanning-tree Mode: PVST

VLAN 100 - STP instance 0

STP Bridge Parameters:
Bridge          Bridge Bridge Bridge Hold  LastTopology Topology
Identifier      MaxAge Hello  FwdDly Time  Change       Change
hex             sec    sec    sec    sec   sec          cnt
806480a235321cbb 20    2      15     1     11           1

RootBridge      RootPath  DesignatedBridge  RootPort          Max Hel Fwd
Identifier      Cost      Identifier                Age lo  Dly
hex                       hex                       sec sec sec
806480a235321cbb 0        806480a235321cbb  Root            20  2   15

STP Port Parameters:
Port            Prio Path      Port Uplink State       Designated  Designated      Designated
Name            rity Cost      Fast Fast               Cost        Root            Bridge
```

Ethernet4	128	200	N	N	FORWARDING	0	806480a235321cbb	806480a235321cbb

For more information, see the (202111) PVST (STP) section in the CLI reference guide.

XSTP (STP, RSTP, MSTP)

Introduction

The purpose of Spanning Tree Protocol (STP) is to prevent and eliminate loops in Ethernet networks. Ethernet networks use a tree-like topology, where data flows from a root bridge (also known as the root switch) to the end devices (computers, servers, etc.) connected to the network.

Loops can occur in a network when there are redundant links between switches. These loops can lead to broadcast storms, excessive network traffic, and duplication of frames, resulting in network congestion, degraded performance, and even network outages.

STP addresses these issues by creating a loop-free logical topology within the network. It does this by selecting a root bridge and calculating the shortest path from the root bridge to each switch in the network. STP disables some of the redundant links to eliminate loops while keeping backup links available for failover.

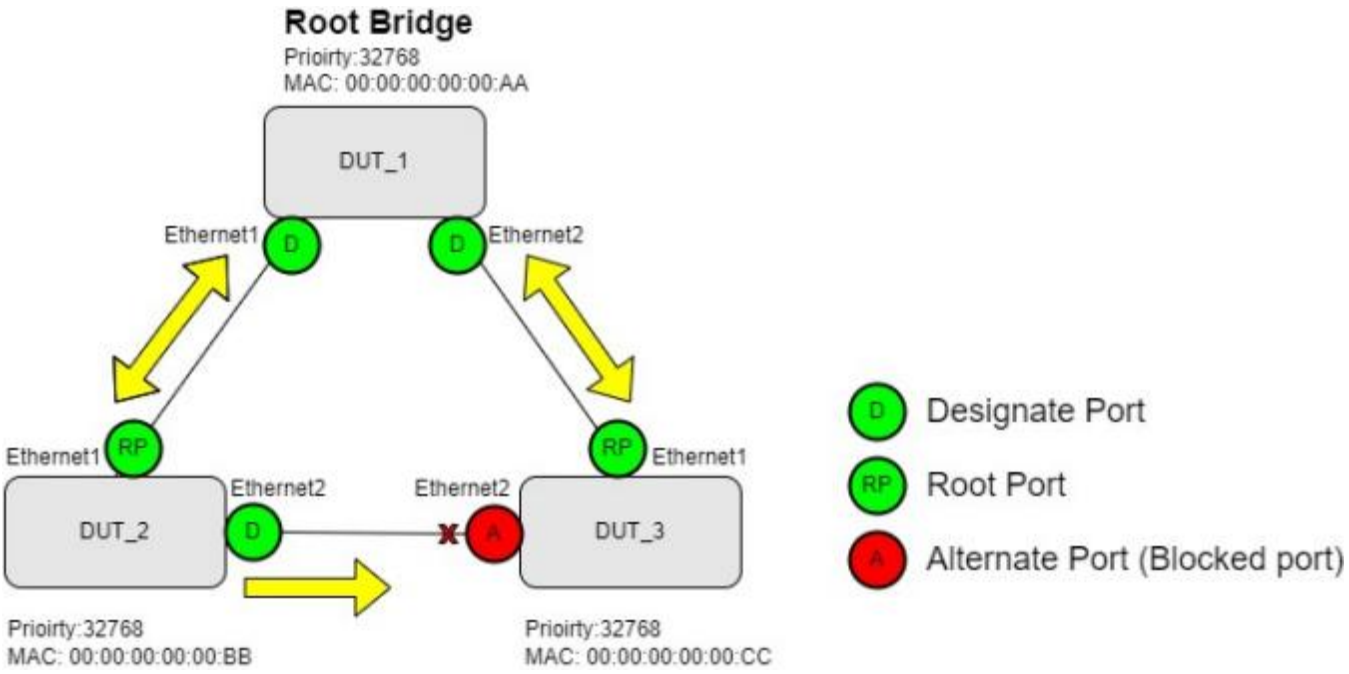
The XSTP feature supports STP, RSTP (Rapid Spanning Tree Protocol), and MSTP (Multiple Spanning Tree Protocol).

The RSTP mode provides faster convergence than STP.

The MSTP mode assigns each VLAN to a specific instance, and each instance can have its own spanning tree. This way, MSTP allows for the independent configuration and optimization of spanning trees for different VLANs, resulting in more efficient network utilization.

MSTP is compatible with RSTP and STP, and RSTP is compatible with STP

Scenario 1 without MSTI



In this scenario, we have 3 DUTs connected to each other. If spanning tree protocol is enabled on all DUTs, thay start sending Bridge Protocol Data Units (BPDU) messages to determine the root bridge.

The Mac address and the priority together make what we called the Bridge ID. The root bridge is determined based on the lowest Bridge

ID. DUT_1 has the lowest Bridge ID, so it becomes the root bridge.

DUT_2 and DUT_3 receive the BPDU messages from DUT_1 and realize that they are not the root bridge. DUT_2 and DUT_3 determine its root port, which is the port providing the shortest path to the root bridge.

In this scenario, DUT_2 and DUT_2 determine their root ports as follows:

- DUT_2: Ethernet1 connected to DUT_1
- DUT_3: Ethernet1 connected to DUT_1

Designate ports are the ports that have been selected as the best path to reach a specific segment. On a root bridge, all ports are capable of forwarding and, therefore, considered designate ports.

In this scenario,

- DUT_1 designates its Ethernet1 port as the designate port for the link to DUT_2.
- DUT_1 designates its Ethernet2 port as the designate port for the link to DUT_3.

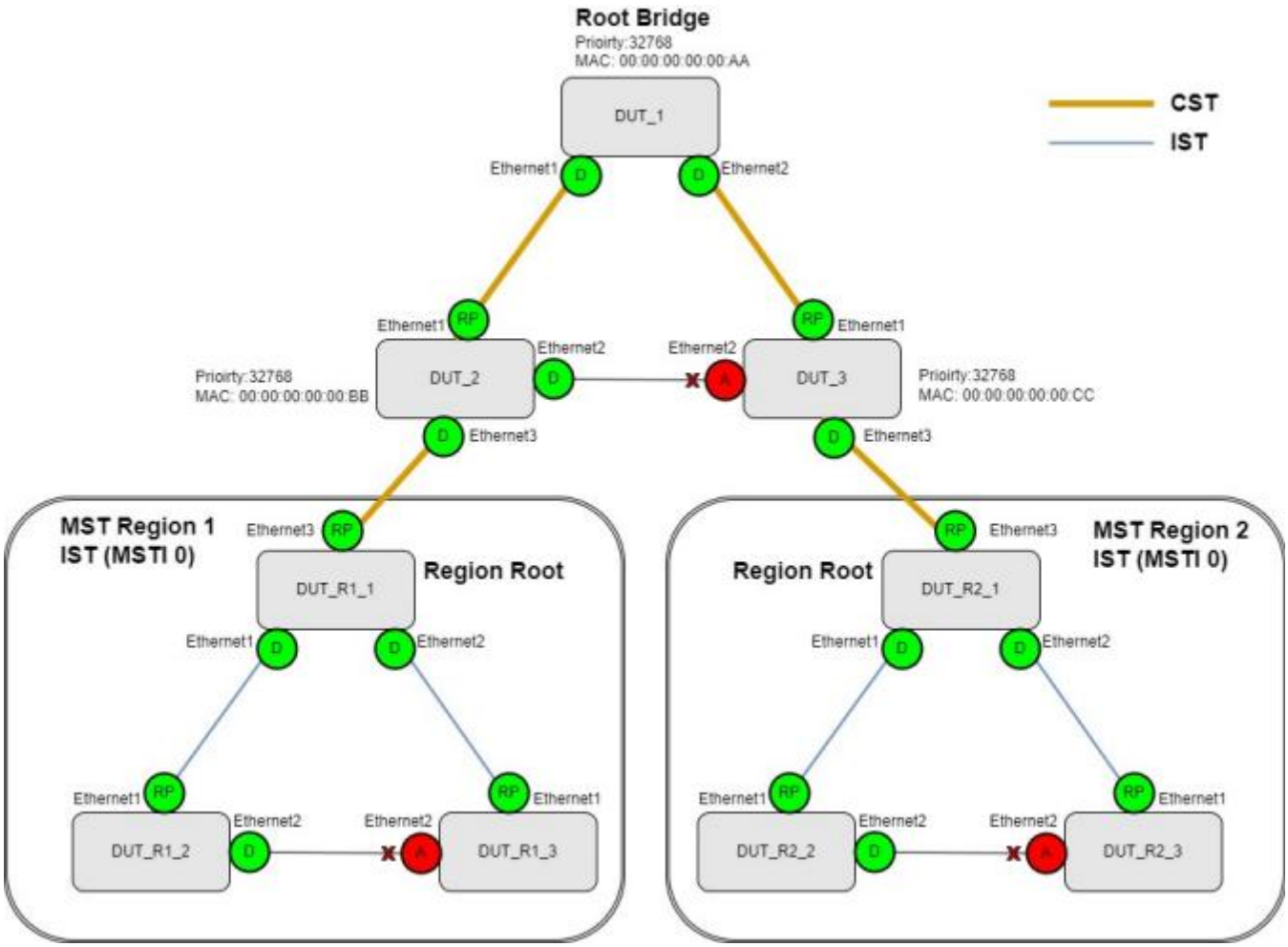
In the last segment, which connects DUT_2 and DUT_3, one of the ports should transition to a discarding state, also known as an alternate port, which prevents the passage of traffic.

Assuming port cost of all links is the same, DUT_2 and DUT_3 will compare their Bridge IDs, Ethernet2 of DUT_2 which has the lower Bridge ID will be designated as the designate port, while the Ethernet2 of DUT_3 will be designated as the alternate port.

In this final spanning tree, DUT_1 is the root bridge, DUT_2 and DUT_3 are designated switches, and the blocked link is shown as a crossed-out line.

Scenario 2 with MSTIs

MSTP divides an entire Layer 2 network into multiple MST regions, each connected by a computed CST (Common Spanning Tree). Within each MST region, multiple spanning trees known as MSTIs (Multiple Spanning Tree Instances) are calculated. Among these MSTIs, MSTI 0 is designated as the internal spanning tree (IST).



In MSTP, each region can have its own independent set of spanning tree instances. Each MST region is identified by a unique region name and revision number. DUTs within the same MST region share the same MST configuration, including the number of MST instances and their respective mappings to VLANs.

All DUTs in same MST region have the following characteristics:

- Same region name.
- Same MSTP revision level.
- Same VLAN-to-MSTI mapping configuration.

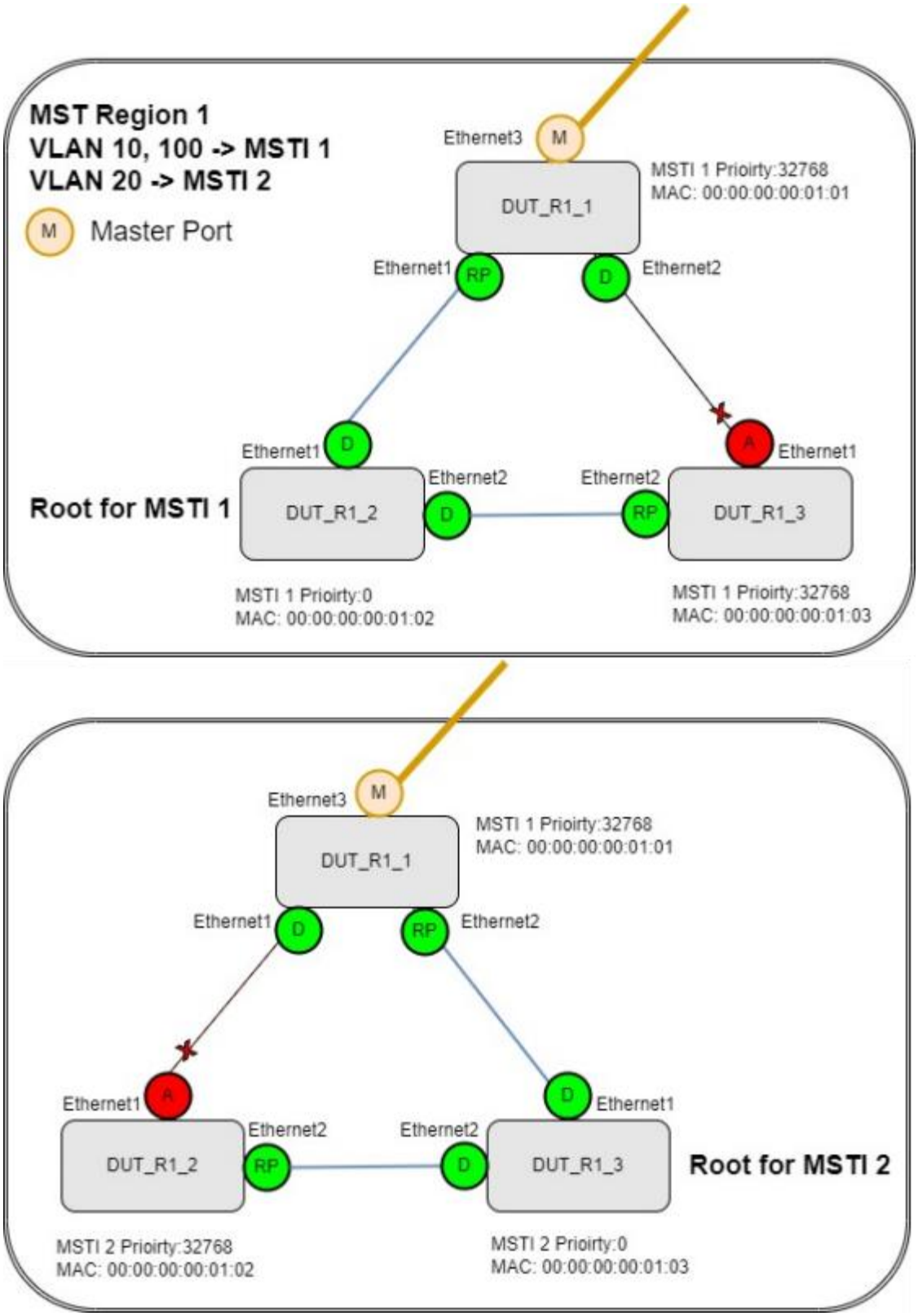
MST Region 1 contains DUT_R1_1, DUT_R1_2, DUT_R1_3.

Suppose that the VLAN-to-MSTI mapping table of MST Region 1 is as follows:

- VLAN 10, 100 are mapped to MSTI 1.
- VLAN 20 is mapped to MSTI 2.

Based on the topology, different spanning trees in an MST region might have different regional roots. In MST Region 1, the regional root of MSTI 1 is DUT_R1_2, the regional root of MSTI 2 is DUT_R1_3, and the regional root of MSTI 0 (also known as the IST) is DUT_R1_1.

A master port is on the shortest path connecting MST regions to the CIST root. Ethernet3 on DUT_R1_1 is the master port because it is the closest port in the region to the CIST root. It is a root port on the IST or CIST and still a master port on the other MSTIs.



How to use it

To enable the spanning tree and specify mode for the device, using the config spanning-tree command.

Example

```
admin@sonic:~$ config spanning-tree enable rstp
```

You can use the show spanning-tree command to display running status.

Example

```
admin@sonic:~$ show spanning-tree
Spanning Tree Information

---------------------------------------------------
Spanning Tree Mode            : RSTP
Spanning Tree Enabled/Disabled  : Enabled
Instance                      : 0
VLANs Configured              : 1-4094
Priority                    : 32768
Bridge Hello Time (sec.)      : 2
Bridge Max. Age (sec.)        : 20
Bridge Forward Delay (sec.)   : 15
Root Hello Time (sec.)        : 2
Root Max. Age (sec.)          : 20
Root Forward Delay (sec.)     : 15
Max. Hops                   : 20
Remaining Hops                : 20
Designated Root             : 32768.14448FBD57C8
Current Root Port             : 0
Current Root Cost             : 0
Number of Topology Changes    : 0
Last Topology Change Time (sec.): 1592
Transmission Limit            : 3
Path Cost Method              : Long
---------------------------------------------------
Ethernet1 Information
---------------------------------------------------
Admin Status                  : Enabled
Role                          : Designate
State                         : Forwarding
Admin Path Cost               : 0
Oper Path Cost                : 400
```

```
Priority                        : 128
Designated Cost                 : 0
Designated Port                 : 128.2
Designated Root                 : 32768.14448FBD57C8
Designated Bridge               : 32768.14448FBD57C8
Designated                      : 1
Forward Transitions             : Auto
Admin Edge Port                 : Enabled
Oper Edge Port                  : Auto
Admin Link Type                 : Point-to-point
Oper Link Type                  : Enabled
Spanning-Tree Status            : Disabled
Root Guard Status               : Disabled
BPDU Guard Status               : Disabled
BPDU Guard Auto Recovery        : Disabled
BPDU Guard Auto Recovery Interval : 300
BPDU Filter Status              : Disabled
TC Propagate Stop               : Disabled
```

The MST region name and revision number are utilized to identify a unique MST region. A bridge, such as this switch, can only be a part of one MST region, and all bridges within the same region must be configured with same MST instances.

Here is a sample configuration for MST on DUT_R1_2:

Example

```
admin@sonic:~$ config spanning-tree enable mstp
admin@sonic:~$ config spanning-tree mst name region1
admin@sonic:~$ config spanning-tree mst revision 1
admin@sonic:~$ config vlan add 10
admin@sonic:~$ config vlan add 100
admin@sonic:~$ config vlan add 20
admin@sonic:~$ config vlan member add 10 Ethernet1
admin@sonic:~$ config vlan member add 10 Ethernet2
admin@sonic:~$ config vlan member add 10 Ethernet3
admin@sonic:~$ config vlan member add 100 Ethernet1
admin@sonic:~$ config vlan member add 100 Ethernet2
admin@sonic:~$ config vlan member add 100 Ethernet3
```

```
admin@sonic:~$ config vlan member add 20 Ethernet1
admin@sonic:~$ config vlan member add 20 Ethernet2
admin@sonic:~$ config vlan member add 20 Ethernet3
admin@sonic:~$ config spanning-tree mst vlan add 1 10
admin@sonic:~$ config spanning-tree mst vlan add 1 100
admin@sonic:~$ config spanning-tree mst vlan add 2 20
admin@sonic:~$ config spanning-tree mst priority 1 0
```

You can use the show spanning-tree mst instance command to display specified MSTI running status.

Example

```
admin@sonic:~$ show spanning-tree mst instance 1
Spanning Tree Information
---------------------------------------------------
  Spanning Tree Mode            : MSTP
  Spanning Tree Enabled/Disabled  : Enabled
  Instance                      : 1
  VLANs Configured              : 10,100
  Priority                      : 0
  Bridge Hello Time (sec.)      : 2
  Bridge Max. Age (sec.)        : 20
  Bridge Forward Delay (sec.)   : 15
  Root Hello Time (sec.)        : 2
  Root Max. Age (sec.)          : 20
  Root Forward Delay (sec.)     : 15
  Max. Hops                     : 20
  Remaining Hops                : 20
  Designated Root               : 0.1.04F8F86AFA91
  Current Root Port             : 0
  Current Root Cost             : 0
  Number of Topology Changes    : 0
  Last Topology Change Time (sec.): 2998
  Transmission Limit            : 3
  Path Cost Method              : Long
---------------------------------------------------
Ethernet1 Information
---------------------------------------------------
  Admin Status                  : Enabled
  Role                          : Designate
  State                         : Forwarding
```

```
External Admin Path Cost                    : 0
Internal Admin Path Cost                    : 0
External Oper Path Cost                     : 400
Internal Oper Path Cost                     : 400
Priority                                    : 128
Designated Cost                             : 0
Designated Port                             : 128.2
Designated Root                             : 0.1.04F8F86AFA91
Designated Bridge                           : 0.1.04F8F86AFA91
Forward Transitions                         : 3
Admin Edge Port                             : Auto
Oper Edge Port                              : Disabled
Admin Link Type                             : Auto
Oper Link Type                              : Point-to-point
Spanning-Tree Status                        : Enabled
Root Guard Status                           : Disabled
BPDU Guard Status                           : Disabled
BPDU Guard Auto Recovery                    :
BPDU Guard Auto Recovery Interval           : 300
BPDU Filter Status                          : Disabled
TC Propagate Stop                           : Disabled
```

You can use the show spanning-tree mst configuration command to display MST name, revision and MSTI-VLAN mapping configuration .

Example

```
admin@sonic:~$ show spanning-tree mst configuration
MSTP Configuration Information
-------------------------------------------------
Configuration Name      : region1
Revision  Level         : 1

Instance  VLANs
-------------------------------------------------
    0    1-9,11-19,21-99,101-4094
    1    10,100
    2    20
```

Limitation

1. Spanning tree protocol does not support across MLAGs.
2. The STP feature loads the docker-stp image, which is responsible for handling PVST and sets the spanning tree to be disabled by default. If a user decides to enable STP/RSTP/MSTP, the system needs to switch to the docker-xstp image. Please note that the system will automatically restart after the user agrees to save the configuration.

Example

```
admin@sonic:~$ show spanning-tree
Spanning-tree is not configured
admin@sonic:~$ docker ps
CONTAINER ID   IMAGE                  COMMAND               CREATED
STATUS         PORTS     NAMES
181b49372ab0   docker-stp:latest      "/usr/local/bin/supe… "   11 minutes ago   Up 11
minutes            stp

admin@sonic:~$ config spanning-tree enable rstp
The current configurations will be saved, and the device will restart. Continue? [y/N]: y
backuping configuration...
Running command: cp /etc/sonic/config_db.json /etc/sonic/config_db_2023-05-25_03:06:16.json
saving configuration...
Running command: /usr/local/bin/sonic-cfggen -d --print-data > /etc/sonic/config_db.json
Running command: config reload -y
Running command: rm -rf /tmp/dropstat-*
Disabling container monitoring ...
Stopping SONiC target ...
Running command: /usr/local/bin/sonic-cfggen  -j /etc/sonic/init_cfg.json  -j /etc/sonic/config_db.
json  --write-to-db
Running command: /usr/local/bin/db_migrator.py -o migrate
Running command: /usr/local/bin/sonic-cfggen -d -y /etc/sonic/sonic_version.yml -t /usr/share/sonic
/templates/sonic-environment.j2,/etc/sonic/sonic-environment
Restarting SONiC target ...
Enabling container monitoring ...
Updating hostname ...
Reloading Monit configuration ...
Reinitializing monit daemon
```

```
admin@sonic:~$ docker ps
CONTAINER ID   IMAGE                      COMMAND                  CREATED
STATUS       PORTS     NAMES
4942fe1c23a8 hours       docker-xstp:latest stp                  "/usr/local/bin/supe… "   3 hours ago   Up 3
```

The originally supported STP will be replaced by the STP mode of XSTP.

For more information, see the (202111) XSTP section in the CLI reference guide.

Dynamic Load Balancing (DLB)

Dynamic Load Balancing (DLB) is a technique that improves the performance of network traffic by dynamically selecting the optimal paths on the current status of links. Unlike traditional methods such as Static Load Balancing (SLB) assigns path using hash functions, which can lead to inefficient bandwidth usage and congestion due to their lack of awareness of path status. DLB operates by assigning paths based on packet, which attempts to maximize link utilization, but may bring reorder problem; flowlets, which are bursts of packets spaced within a flow to avoid reordering problem. Path selection is based on port and switch resource utilization metrics such port load and queue size to improve the bandwidth usage and avoid congestion.

DLB supports the metrics that determine which of the members of an ECMP group are the most qualified. Three mode is supported:

- spary: assign path per packet.
- eligible: assign path based on flowlet. The flowlet is split by a user-defined interval, default 1ms. If packets arrive smaller than this interval, they are considered part of the same flowlet.
- fixed: assigning a path per packet based on packet patterns.

DLB is only supported on the following models:

- N9200-64DC
- N9500-128QC
- N9500-64OC

Only DLB ECMP is supported; DLB LAG is not supported.

CLI command

DLB Show Commands

show dlb ecmp config
This command is used to display DLB configuration.

Example
admin@sonic:~$ show dlb ecmp config
DLB ECMP Status : Enabled
Mode : Eligible
Flowlet Gap Time : 1000 us

DLB Config Commands

The following command is used to enable DLB for ECMP. Enable DLB, if not specify the DLB mode the default mode is eligible. The eligible mode is flowlet mode, and it requires a mandatory parameter, flowlet gap time, when running in this mode. config dlb ecmp enable {eligible | fixed | spray} [--flowlet-gap-time]

- eligible: Assign path based on flowlet. The flowlet is split by a user-defined interval, default 1ms. If packets arrive smaller than this interval, they are considered part of the same flowlet.
    - < flowlet-gap-time >: The flowlet gap time. The unit is microseconds (μs). Minimum: 16, Maximum: 32,767. The default value is 1,000
- fixed: Assigning a path per packet based on packet patterns.
- spary: Assigning a path for each packet individually.

config dlb ecmp disable

The following example shows how to enable DLB spray mode
Example
admin@sonic:~$ sudo config dlb ecmp enable spray
The following example shows how to enable DLB eligible mode and set flowlet gap time to 1 ms
Example
admin@sonic:~$ sudo config dlb ecmp enable eligible --flowlet-gap-time 1000

Layer 3

SONiC Routing Stack

SONiC uses the Free Range Routing (FRRouting or FRR) IP routing suite for the configuration of routing protocols. FRR supports a number of common routing protocols, such as BGP, OSPF, RIP etc, all of which can be configured through the FRR VTY shell “vtysh.”

Enable Routing Protocols

BGP protocol is always enabled. Other routing protocols, including OSPF, ISIS, BFD etc, can be enabled by setting "frr_mgmt_framework_config" field to "true" in config_db's DEVICE_META table.

CONFIG_DB


```
"DEVICE_METADATA": {
    "localhost": {
        "bgp_asn": "65100",
        "buffer_model": "traditional",
        "cloudtype": "None",
        "default_bgp_status": "up",
        "default_pfcwd_status": "disable",
        "deployment_id": "1",
        "docker_routing_config_mode": "split",
        "frr_mgmt_framework_config": "true",
        "hostname": "n9200-64dc",
        "hwsku": "naddod-N9200-64DC",
        "mac": "68:21:5f:d2:f1:33",
        "platform": "x86_64-naddod_n9200_64dc-r0",
        "region": "None",
        "synchronous_mode": "enable",
        "type": "ToRRouter"
    }
},
```

Configuration of Routing Protocols

The startup configuration files in FRR is controlled by the setting of "docker_routing_config_mode" in config_db's DEVICE_META table.

There are three modes which can be set: "separated", "unified" and "split". Among them, "separated" mode is the default routing config mode.

The different between these three routing config modes are as follow:

docker_routing_config_mode	Config files for FRR's daemons	Config files generation during docker startup
separated	individual configuration file for each daemons (bgpd.conf, ospfd.conf, staticd.conf, ... etc)	generates and overwrites configuration files using config_db
unified	one unified configuration file (frr.conf)	generates and overwrites configuration files using config_db
split	one unified configuration file (frr.conf)	won't modify any configuration file

Note: If add route by CLI in split mode, it needs to execute "copy running-config startup-config" in v tysh to preserve the rout setting. Because it won't generates configuration files from config_db.

Full details of FRR routing support and configuration can be found at:
http://docs.frrouting.org/en/latest/index.html

Enter the v tysh command to access the FRR VTY shell.

Example

```
admin@sonic:~$ vtysh

Hello, this is FRRouting (version 7.2.1-sonic).
Copyright 1996-2005 Kunihiro Ishiguro, et al.

sonic#
sonic# show route-map
ZEBRA:
route-map: RM_SET_SRC Invoked: 82595
 permit, sequence 10 Invoked 82595
   Match clauses:
```

Set clauses:

src 10.1.0.32

Call clause:

Action:

Exit routemap

Border Gateway Protocol (BGP)

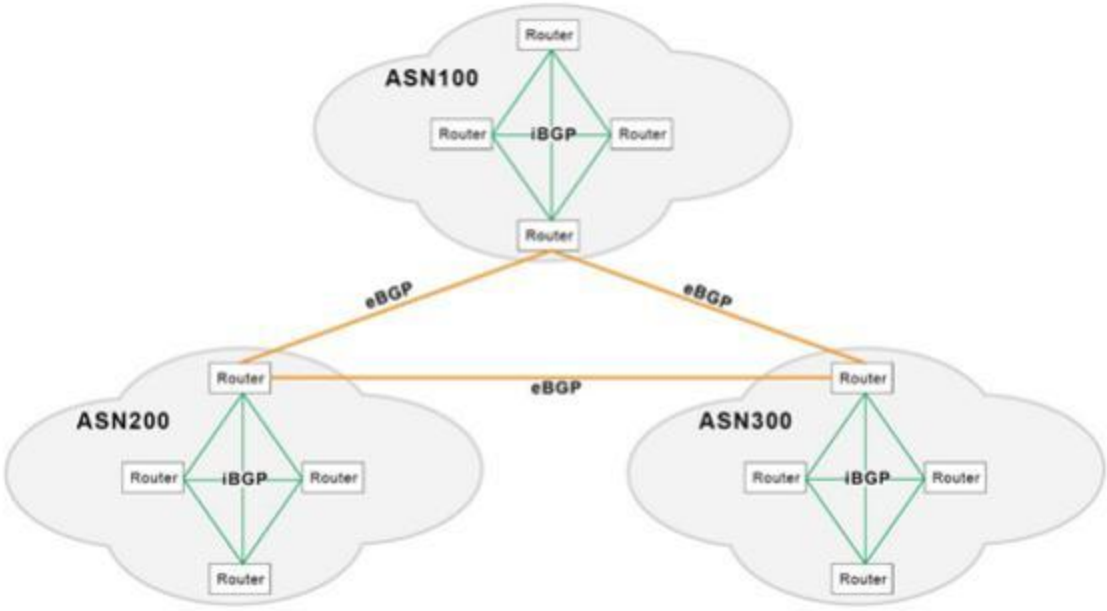
BGP stands for Border Gateway Protocol. It is a routing protocol that is used to exchange routing information between autonomous systems (ASs) on the Internet. An autonomous system (AS) functions as a separate routing domain under one administrative authority, which implements its own routing policies. AS exchanges routing information within its boundaries using Interior Gateway Protocols (IGPs) such as RIP or OSPF and connects to external organizations or the Internet using an Exterior Gateway Protocol (EGP). BGP version 4 is the primary EGP deployed on the Internet today.

A communication session must be maintained between bordering ASs to support the periodic exchange of routing information. One of the major design choices for BGP is the use of a TCP connection to exchange routing information between peers. Exchanging connectivity information over a reliable transport mechanism effectively delegates all error control functions to TCP.

The other major innovation for BGP is the use of path vectors that carry the full list of transit networks, or ASs traversed between the source and destination. Loops are prevented simply by checking the path vector to see if the same AS is listed twice. This approach solves many of the scalability problems encountered when applying distance-vector or link-state methods to make routing decisions in complex topologies.

External and Internal BGP

When connecting to the Internet, external BGP (eBGP) is used. Although BGP is widely used as an exterior gateway protocol (EGP), it is also used in many organizations with complex internal networks. Internal networks can be simplified by exchanging routing information among BGP peers within the same organization through internal BGP (iBGP) peering sessions.



External BGP – eBGP interconnects different ASs through border routers or eBGP peers. These peering routers are commonly connected over a WAN link using a single physical path. Alternatively, multiple eBGP peer connections may be used to provide redundancy or load balancing. Distinct BGP sessions are used between redundancy peers to ensure that if one session fails, another will take over.

BGP uses the AS path attribute to record the ASs that must be followed to reach the prefix for a network aggregate. When a prefix is announced to an eBGP peer, the local AS number (ASN) is prepended to the AS path. This prevents routing loops by rejecting any prefix announcements that include the local ASN in the AS path. These announcements are also used by eBGP in the best path selection process. eBGP speakers can communicate with other external peers or with iBGP peers. A BGP speaker can determine if it is communicating with an external or internal peer by comparing the ASN sent in OPEN messages by a peer with that of its own internal value. If it matches, then this neighbor is an iBGP speaker, and if it does not, then it is an eBGP speaker. An eBGP speaker can advertise prefixes it has learned from another eBGP speaker to a neighboring iBGP speaker, and it can also advertise prefixes it has learned from an iBGP speaker to an eBGP speaker.

Internal BGP – In contrast to eBGP peers which have different ASNs, iBGP peers are configured with the same ASN. All iBGP peers within the same AS should be connected in a full-mesh connection (except when using route reflection). When a prefix is announced from one iBGP peer to another, the AS path is not changed. Since all iBGP peers are fully meshed, they will have the same information in their BGP table, unless routing policies have been modified for some of the peers.

When an iBGP peer receives a prefix announcement, it uses the best path selection algorithm to see if the received announcement is the best path for that prefix. If it is, the peer inserts this route into its routing table and announces it to all of its peers, both iBGP and eBGP. If it is not the best available path, the peer keeps

a copy of it in its routing table so that if path information for that prefix changes (such as if the current best available path is withdrawn), it can be used to calculate a new best available path.

BGP cannot detect routes and provide reachability information. To ensure that each iBGP peer knows how to reach others, each peer must run some sort of Interior Gateway Protocol (such as static routes, direct routes, RIP, or OSPF) which provides neighbor IP addresses. To avoid routing loops, an iBGP speaker cannot advertise prefixes it has learned from one iGBP peer to another neighboring iBGP peer.

Configuring SONiC BGP

SONiC uses the Free Range Routing (FRRouting or FRR) IP routing suite for the configuration of BGP and other routing protocols. The FRR protocol daemons can all be configured through the FRR VTY shell "vtysh." Full details of FRR BGP configuration can be found at: http://docs.frrouting.org/en/latest/bgp.html

Enter the v tysh command to access the FRR VTY shell.

Example
admin@sonic:~$ vtysh Hello, this is FRRouting (version 7.2.1-sonic). Copyright 1996-2005 Kunihiro Ishiguro, et al. sonic#

The commands available for BGP configuration in SONiC are limited and listed below.

To shutdown :

- For all BGP IPv4 and IPv6 sessions, use config bgp shutdown all command
- For a specific BGP session with a neighbor, use config bgp shutdown neighbor command

Similarly, to startup:

- For all neighbors, use config bgp startup all command
- For a specific neighbor, use config bgp startup neighbor command

You can also remove a specific IPv4 or IPv6 BGP neighbor using the config bgp remove neighbor command.

Example
admin@sonic:~$ sudo config bgp shutdown all Shutting down BGP session with neighbor 10.0.0.57... Shutting down BGP session with neighbor 10.0.0.59... Shutting down BGP session with neighbor 10.0.0.61... Shutting down BGP session with neighbor 10.0.0.63...

```
admin@sonic:~$ sudo config bgp startup all
Starting up BGP session with neighbor 10.0.0.57...
Starting up BGP session with neighbor 10.0.0.59...
Starting up BGP session with neighbor 10.0.0.61...
Starting up BGP session with neighbor 10.0.0.63...

admin@sonic:~$ sudo config bgp remove neighbor 10.0.0.57

Removed configuration of BGP neighbor 10.0.0.57
```

Note：

These SONiC bgp commands (shutdown/startup/remove) will only apply to the neighbor entries included in config_db.json (redis db).

Otherwise, it will show an error message stating that it was unable to find the neighbor and would have no effect.

An example entry in config_db.json

```
    "BGP_NEIGHBOR": {
        "10.0.0.57": {
            "admin_status": "up",
            "asn": "64600",
            "holdtime": "10",
            "keepalive": "3",
            "local_addr": "10.0.0.56",
            "name": "ARISTA01T1",
            "nhopself": "0",
            "rrclient": "0"
        },
```

Error

```
admin@sonic:~$ sudo config bgp shutdown neighbor 10.1.1.1
Usage: config bgp shutdown neighbor [OPTIONS] <ipaddr_or_hostname>
Try "config bgp shutdown neighbor -h" for help.

Error: Could not locate neighbor ' 10.1.1.1'
```

For FRR 'split' mode, please use below commands in the FRR VTY shell.

Example

```
admin@sonic:~$ vtysh

Hello, this is FRRouting (version 8.1).
Copyright 1996-2005 Kunihiro Ishiguro, et al.

sonic# config
sonic(config)# router bgp 65100
sonic(config-router)# neighbor 10.0.0.61 shutdown (shutdown)
sonic(config-router)# no neighbor 10.0.0.61 shutdown (startup)
sonic(config-router)# no neighbor 10.0.0.61 (remove)
sonic(config-router)# bgp shutdown (shutdown all)
sonic(config-router)# no bgp shutdown (startup all)
```

To display the BGP configuration in SONiC, use the show bgp summary command.

Example

```
admin@sonic:~$ show bgp summary

IPv4 Unicast Summary:
BGP router identifier 10.1.0.32, local AS number 65100 vrf-id 0
BGP table version 6465
RIB entries 12807, using 2001 KiB of memory
Peers 4, using 83 KiB of memory
Peer groups 2, using 128 bytes of memory

Neighbor        V         AS MsgRcvd MsgSent   TblVer  InQ OutQ  Up/Down State/PfxRcd
10.0.0.57       4      64600    3995    4001        0    0    0 00:39:32         6400
10.0.0.59       4      64600    3995    3998        0    0    0 00:39:32         6400
10.0.0.61       4      64600    3995    4001        0    0    0 00:39:32         6400
10.0.0.63       4      64600    3995    3998        0    0    0 00:39:32         6400

Total number of neighbors 4

IPv6 Unicast Summary:
BGP router identifier 10.1.0.32, local AS number 65100 vrf-id 0
BGP table version 12803
RIB entries 12805, using 2001 KiB of memory
Peers 4, using 83 KiB of memory
Peer groups 2, using 128 bytes of memory

Neighbor        V         AS MsgRcvd MsgSent   TblVer  InQ OutQ  Up/Down State/PfxRcd
```

fc00::72	4	64600	3995	5208	0	0	0 00:39:30	6400
fc00::76	4	64600	3994	5208	0	0	0 00:39:30	6400
fc00::7a	4	64600	3993	5208	0	0	0 00:39:30	6400
fc00::7e	4	64600	3993	5208	0	0	0 00:39:30	6400

Total number of neighbors 4

For more information on SONiC BGP commands, see the (202111) BGP section in the CLI reference guide.

Bidirectional Forwarding Detection (BFD)

Bidirectional Forwarding Detection (BFD) is an IETF protocol (RFC5880) that is designed to rapidly detect failures in communications between systems. BFD functions independently of other routing protocols by checking if the next-hop router on a forwarding path is alive. Where routing protocols normally use "hello" methods to detect link failures that take seconds, BDF is able to detect forwarding path failures in milliseconds. The BDF protocol operates between two systems, sending unicast packets that can be carried in any encapsulation that might be implemented in a network.

In the SONiC system, BFD is included as part of the FRR BGP container. To configure a BFD peer, use the peer command in FRR BFD daemon configuration mode.

Example

```
admin@sonic:~$ vtysh

Hello, this is FRRouting (version 7.2.1-sonic).
Copyright 1996-2005 Kunihiro Ishiguro, et al.

sonic# config
sonic(config)# bfd
sonic(config-bfd)# peer 192.169.1.11
sonic(config-bfd-peer)# end
sonic# show bfd peer 192.169.1.11
BFD Peer:
        peer 192.169.1.11 vrf default
                ID: 1991080464
                Remote ID: 3120344281
                Status: up
                Uptime: 4 hour(s), 29 minute(s), 12 second(s)
                Diagnostics: ok
```

```
Remote diagnostics: ok
Local timers:
        Receive interval: 1000ms
        Transmission interval: 1000ms
        Echo transmission interval: 100ms
Remote timers:
        Receive interval: 300ms
        Transmission interval: 300ms
        Echo transmission interval: 50ms
```

For more information on BFD configuration and BGP BFD commands, see the (202111) BFD (Basic) section in the CLI reference guide.

Virtual Routing and Forwarding (VRF)

Virtual Routing and Forwarding (VRF) is the support for multiple independent routing tables working simultaneously on the same switch. VRF-aware routing protocols, such as FRR BGP, are used to maintain separate VRF tables for each configured VRF instance (identified by a specified name). Once you have configured a VRF instance name, you can bind Layer 3 interfaces (VLANs, ports, and port channels) to the VRF. By default, all Layer 3 interfaces are bound to the default VRF.

Use the config vrf add command to create a VRF instance with a specified name. Note that VRF names should always start with “Vrf.” Note that the name "mgmt" is reserved for the management VRF and should not be used.

To bind a Layer 3 interface to a non-default VRF, use the config interface vrf bind command. Use the config interface vrf unbind command to remove an interface from a non-default VRF.

Use the show vrf command to display all VRFs configured on the system.

Example

```
admin@sonic:~$ sudo config vrf add Vrf01

admin@sonic:~$ config interface vrf bind PortChannel0001 Vrf01

admin@sonic:~$ show vrf
VRF    Interfaces

Vrf01  PortChannel0001
```

For more information on VRF commands, see the (202111) VRF Configuration and (202111) Interfaces (Update) sections in the CLI reference guide.

Management VRF Configuration

Management Virtual Routing and Forwarding (VRF) supports a separate routing table for management traffic that is forwarded by out-of- band management ports on a switch. This enables management traffic to be routed across a management network and still remain secure and separate from network data traffic.

Use the config vrf add mgmt command to enable the management VRF. This binds the management interface "eth0" to the “mgmt” VRF.

Use the show mgmt-vrf command to display details about the management VRF, or the show mgmt-vrf routes to display the management VRF routing table.

Example

```
admin@sonic:~$ sudo config vrf add mgmt
admin@sonic:~$ show mgmt-vrf

  ManagementVRF : Enabled

  Management VRF interfaces in Linux:
  348: mgmt: <NOARP,MASTER,UP,LOWER_UP> mtu 65536 qdisc noqueue state UP mode DEFAULT
  group default qlen 1000
      link/ether f2:2a:d9:bc:e8:f0 brd ff:ff:ff:ff:ff:ff
  2: eth0: <BROADCAST,MULTICAST,UP,LOWER_UP> mtu 1500 qdisc mq master mgmt state UP mode
  DEFAULT group
default qlen 1000
      link/ether 4c:76:25:f4:f9:f3 brd ff:ff:ff:ff:ff:ff
  350: lo-m: <BROADCAST,NOARP,UP,LOWER_UP> mtu 1500 qdisc noqueue master mgmt state
  UNKNOWN mode DEFAULT
group default qlen 1000
      link/ether b2:4c:c6:f3:e9:92 brd ff:ff:ff:ff:ff:ff
admin@sonic:~$ show mgmt-vrf routes

Routes in Management VRF Routing Table:
default via 10.16.210.254 dev eth0 metric 201
broadcast 10.16.210.0 dev eth0 proto kernel scope link src 10.16.210.75
10.16.210.0/24 dev eth0 proto kernel scope link src 10.16.210.75
```

```
local 10.16.210.75 dev eth0 proto kernel scope host src 10.16.210.75
broadcast 10.16.210.255 dev eth0 proto kernel scope link src 10.16.210.75
broadcast 127.0.0.0 dev lo-m proto kernel scope link src 127.0.0.1
127.0.0.0/8 dev lo-m proto kernel scope link src 127.0.0.1
local 127.0.0.1 dev lo-m proto kernel scope host src 127.0.0.1

broadcast 127.255.255.255 dev lo-m proto kernel scope link src 127.0.0.1
```

For more information, see the (202111) Management VRF section in the CLI reference guide.

Leaking VRF Routes

Virtual Routing and Forwarding (VRF) supports multiple independent routing tables on the same switch. Although each VRF is essentially isolated from other VRFs, there are cases where it is a benefit for a VRF to share or “leak” information to another VRF. This VRF route leaking is where routes and next-hops of connected destinations in one VRF are made available to another VRF.

SONiC supports both static and dynamic VRF route leaking through the Free Range Routing (FRR) IP routing suite. For static VRF route leaking, the FRR CLI can be used to specify a next-hop IP that is reachable through a next-hop VRF. With dynamic VRF route leaking, route maps are used to automatically import routes from other VRFs. Prefix lists within route maps are used to match route prefixes in the source VRF and various actions can be applied. If a route map action is permitted, the matched routes will be imported into the destination VRF. The leaked routes are automatically deleted when the corresponding routes are deleted in the source VRF.

As an example of configuring a static VRF route leak, consider the following that leaks a next-hop IP route from a “Vrf1” into a “Vrf2.”

Example

```
admin@sonic:~# vtysh
sonic# configure
sonic(config)# vrf Vrf1
sonic(config-vrf)# ip route 2.2.2.2/32 10.0.0.59 nexthop-vrf Vrf2
sonic(config-vrf)# end
sonic# exit
admin@sonic:~#
admin@sonic:~# show ip route vrf Vrf1
Codes: K - kernel route, C - connected, S - static, R - RIP,
       O - OSPF, I - IS-IS, B - BGP, E - EIGRP, N - NHRP,
```

```
        T - Table, v - VNC, V - VNC-Direct, A - Babel, D - SHARP,
        F - PBR, f - OpenFabric,
        > - selected route, * - FIB route, q - queued route, r - rejected route
VRF Vrf1:
S>* 2.2.2.2/32 [1/0] via 10.0.0.59, PortChannel0002(vrf Vrf2), 01:34:15
C>* 10.0.0.56/31 is directly connected, PortChannel0001, 01:40:04
```

To configure a dynamic VRF route leak, consider the following example that imports BGP routes from the default VRF into “Vrf1.”

Example

```
admin@sonic:~# vtysh
sonic# configure
sonic(config)# router bgp 65200 vrf Vrf1
sonic(config-router)# address-family ipv4 unicast
sonic(config-router-af)# import vrf default
sonic(config-router-af)# end
sonic# exit
admin@sonic:~#
admin@sonic:~# show ip route vrf Vrf1 193.11.248.128
Routing entry for 193.11.248.128/25
  Known via "bgp", distance 200, metric 0, vrf Vrf1, best
  Last update 00:03:47 ago
  * 10.0.0.59, via PortChannel0002(vrf default)
  * 10.0.0.61, via PortChannel0003(vrf default)
  * 10.0.0.63, via PortChannel0004(vrf default)
```

For more information, see the (202111) VRF Route Leaking section in the CLI reference guide, or refer to details in the FRR manual at: http://docs.frrouting.org/en/latest/index.html

Displaying ARP Entries

The Address Resolution Protocol (ARP) is used to map IP addresses to MAC addresses. When an IP frame is received by a router, it first looks up the MAC address corresponding to the destination IP address in the ARP cache. If the address is found, the router writes the MAC address into the appropriate field in the frame header and forwards the frame onto the next hop. IP traffic passes along the path to its destination in this way, with each routing device mapping the destination IP address to the MAC address of the next hop toward the recipient, until the packet is delivered to the destination.

To display entries in the ARP cache, use the show arp command to either list all entries, or list for a specific IP address or interface.

Example

```
admin@sonic:~$ show arp
 Address         MacAddress           Iface        Vlan
-------------    -----------------    -------      ------
192.168.1.183    88:5a:92:fb:bf:41    Ethernet44    -
192.168.1.175    88:5a:92:fc:95:81    Ethernet28    -
192.168.1.181    e4:c7:22:c1:07:7c    Ethernet40    -
192.168.1.179    88:5a:92:de:a8:bc    Ethernet36    -
192.168.1.118    00:1c:73:3c:de:43    Ethernet64    -
192.168.1.11     00:1c:73:3c:e1:38    Ethernet88    -
192.168.1.161    24:e9:b3:71:3a:01    Ethernet0     -
192.168.1.189    24:e9:b3:9d:57:41    Ethernet56    -
192.168.1.187    74:26:ac:8b:8f:c1    Ethernet52    -
192.168.1.165    88:5a:92:de:a0:7c    Ethernet8     -

Total number of entries 10
```

For more information, see the (202111) ARP section in the CLI reference guide.

In Band Management VRF

Management Virtual Routing and Forwarding (VRF) supports a separate routing table for management traffic that is forwarded by in-band L3 interfaces on a switch.

Use the config vrf add mgmt command to create the management VRF and use config in-band-mgmt enable to enable in-band management VRF.

Example

```
admin@sonic:~$ sudo config vrf add mgmt
admin@sonic:~$
admin@sonic:~$ sudo config in-band-mgmt enable
```

To bind a Layer 3 interface to management VRF, use the config interface vrf bind command. Use the config interface vrf unbind command to remove an interface from management VRF.

Example

```
admin@sonic:~$ sudo config interface vrf bind Ethernet4 mgmt
```

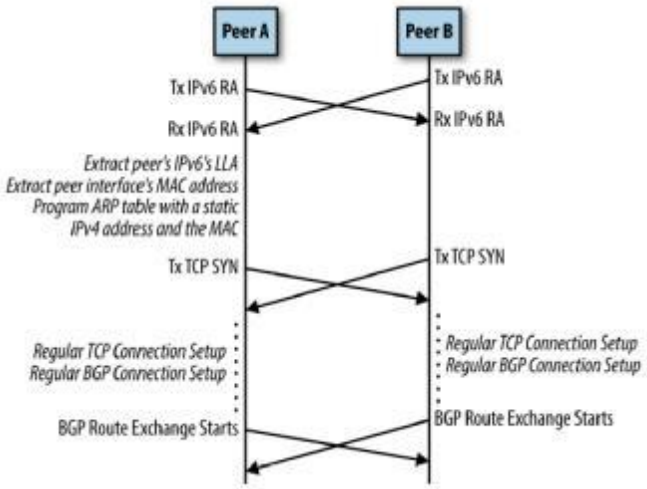
Use the show mgmt-vrf command to display details about the management VRF.

Example

```
admin@sonic:~$ show mgmt-vrf

ManagementVRF : Enabled

In-band ManagementVRF : Enabled

Management VRF interfaces in Linux:
44: mgmt: <NOARP,MASTER,UP,LOWER_UP> mtu 65575 qdisc noqueue state UP mode DEFAULT
group default qlen 1000
    link/ether fe:d2:e8:7e:16:03 brd ff:ff:ff:ff:ff:ff promiscuity 0 minmtu 1280 maxmtu 65575
    vrf table 5000 addrgenmode eui64 numtxqueues 1 numrxqueues 1 gso_max_size 65536
    gso_max_segs 65535
2: eth0: <BROADCAST,MULTICAST,UP,LOWER_UP> mtu 1500 qdisc mq master mgmt state UP mode
DEFAULT group default
qlen 1000
    link/ether 80:a2:35:46:08:99 brd ff:ff:ff:ff:ff:ff
10: Ethernet4: <BROADCAST,MULTICAST,UP,LOWER_UP> mtu 9100 qdisc pfifo_fast master mgmt
state UP mode DEFAULT
group default qlen 1000
    link/ether 80:a2:35:46:08:99 brd ff:ff:ff:ff:ff:ff
45: lo-m: <BROADCAST,NOARP,UP,LOWER_UP> mtu 1500 qdisc noqueue master mgmt state
UNKNOWN mode DEFAULT group
default qlen 1000
    link/ether 3e:06:93:45:7d:f8 brd ff:ff:ff:ff:ff:ff
```

BGP Unnumbered

In the original BGP configuration, it needs to specify a peer IPv4 address to establish peer connections. For CLOS deployment in the data center, the IP interfaces grow dramatically due to the full connection between spines and LEAFs. (each peer connection needs IPv4 addresses on it). To eliminate this situation, we need to use BGP unnumbered to automatically establish peer connections by protocol itself.



This Layer-3 link is capable of running IPv6, so the system will use IPv6 link-local addresses that are automatically generated by each IPv6 interface of the local and remote peer. These addresses will be used to establish the BGP TCP session. The ASN number is ignored during the BGP session establishment.

Once IPv6 BGP session is established, the system is able to exchange IPv4 NLRIs (prefixes) over IPv6 BGP session using IPv6 link-local neighbor address as a next hop. The system associates the IPv6 link local address with that neighbor so that the neighbor will be used as a next hop for the routes.

BGP unnumbered uses 169.254.0.1 as the unnumbered next hop. As such, while using BGP unnumbered, do not use this address in your topology in the following usages:

1. The interface's IPv4 addresses
2. The prefix or next hop of static routes
3. The ARP neighbor address

IBGP is not supported for BGP unnumbered.

Limitation:

One neighbor can only have one unnumbered interface to be connected. In current FRR implementation, it can't identify correct next hop through multiple unnumbered interface. The route table might have a problem when more than one unnumbered interface is connected to the same peer.

BGP Graceful Restart

How graceful restart work

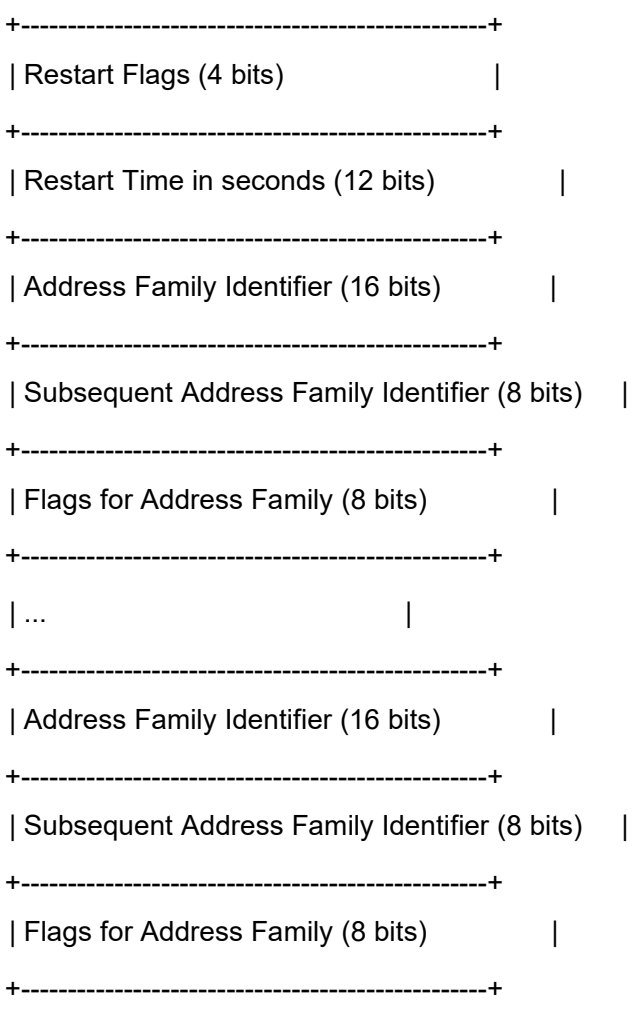
- End-of-RIB marker
  - a special BGP Update Message, no reachable Network Layer Reachability Information (NLRI) and empty withdrawn NLRI.

- not only used for graceful restart, but also help improve route update converage time. indicating the route update is finished, bgp can start routing path calculation.

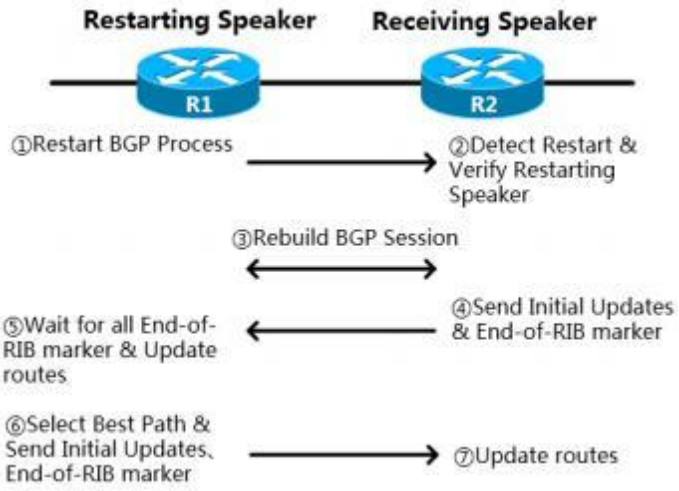
Gracful Restart Capability

- a bgp capability negotiate during open message.
- it's a declearation that during the device restarting process, the data plane still can forward packets. 。the AFI and SAFI support data plane preseve forward state should indicate inside the capability message

Graceful Restart Capability packet format



- if it is no any AFI and SAFI specified, which indicate it the bgp speaker support only for act as restart helper, also imply the bgp speaker will sending End-of-RIB marker when route update finished.
- Working process



The device role

Restaring Speaker : the device execute restart bgp instance or doing warm restart.

Receiving Speaker (know as the bgp helper, used in later description) : the device execute restart bgp instance or doing warm restart.

1. Restarting Speaker restarts BGP process

- Restarting speaker: when restarting speaker restart bgp process, it doesn't remove the routing rule in local RIB (chip) but mark as stale.
- bgp helper: mark route as the stale but preserve it inside the chip, the route still working.

2. Receiving Speaker detects restart &verifies Restarting Speaker

when receiving speaker restart, when it send tcp connection or send Open message, the bgp helper should the rule to determinate whether bgp help should preserve the route:

a.restarting speaker send Open message contains Graceful Restart Capability

b.Graceful Restart Capability specified the related AF

c.AF ForwardingState bit is enable

3. Rebuilding BGP Session

a.restarting speaker should set RestateState bit to 1 in Open message to info bgp helper how long should wait restart peer come back .

b.the bgp helper should send RestartState bit with 0 in Open message

c.if retarting speaker expire the restart time and not re-establish connection with bgp helper, the bgp helper should remove all marked as staled route.

4. Receiving Speaker sends Initial Updates& End-of-RIB marker

receiving speaker should send End-of-RIB marker after sending all route update after rebuild connection with restarting speacker , even though there is no any route update it should send End-of-RIB

5. Restarting Speaker waits for all End-of-RIBmarker & Update routes

   restarting speaker should start path selection until receive all bgp helper send End-of-RIB marker, except following two condition:

   a.the bgp helper doesn't support Graceful Restart Capability.

   b.if bgp helper also restarted , Restart State bit should be set as 1 in Graceful Restart Capability.

6. Restarting Speaker selects Best Path &sends Initial Updates、End-of-RIB marker

   starting path calculation after received all bgp helper sending End-of-RIB, Restarting speaker will remove all stale mark on route and also send End-of-RIB to all it's peers

7. Receiving Speaker updates routes
   bgp helper should remeove stale flag also after receiving all route update from restarting speaker .the route marked as stale and never updated after this process should be removed .

When using docker_routing_config_mode = split in section of metadata.localhost in the sonic_config.json

it have to mauna l indicating the router will support preserve forwarding state while the control plane restart (bgp process restart, or device warm restart), please read document about warm-restart support section about bgp part.

```
bgp graceful-restart preserve-fw-state
```

Refer: https://docs.frrouting.org/en/latest/bgp.html#clicmd-bgp-graceful-restart-preserve-fw-state

Reference

https://www.rfc-editor.org/rfc/rfc4724.html

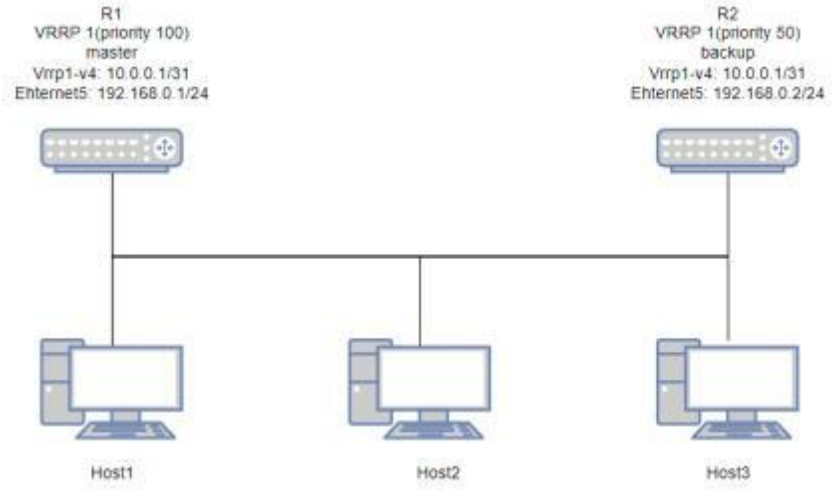
Virtual Router Redundancy Protocol (VRRP)

- Introduction
- Scenario
- How to use it
- How to check it
- Note

Introduction

Virtual Router Redundancy Protocol (VRRP) is a well-known backup router mechanism to prevent traffic broken suddenly when a router fails. The election happens between these routers which participate in the VRRP group. The highest priority router is elected as the master responsible for forwarding traffic, and another act as backup.

Scenario



Assume that there are two routers participating in the VRRP group 1. The following is the related definition

- left router R1: VRRP group's priority is high.
- right router R2: VRRP group's priority is low.
- Vrrp1-v4: It is a VRRP interface for IPv4. its IP to be as the gateway.
- Ethernet5: It is the parent interface. VRRP protocol receives/sends messages through it.

After configuring the VRRP group and interface completely, the VRRP protocol will do the following things

- Election role: Once the VRRP group configuration is completed, advertisement packets are sent to all routers within the group to determine the master and backup roles. The router with the highest priority within the VRRP group is elected as the master. If the priorities are equal, the router with the highest IP address on the LAN interface becomes the master. In that scenario, R1 will become master because its priority is high.

- Gratuitous ARP: After the VRRP group becomes the master, gratuitous ARP packets are sent to the hosts to let them know how to reach the gateway.

- Failover: The backup router will automatically take over the master and start handling the network traffic If the master router fails or is unreachable.

How to use it

To ensure proper operation of VRRP, the following three components need to be configured:

1. VRRP group:

R1

```
admin@sonic:~$ vtysh
sonic# configure
sonic(config)# interface                    Ethernet5
sonic(config-if)# vrrp 1                    version 3
sonic(config-if)# vrrp 1                    priority 100
sonic(config-if)# vrrp 1                    ip 10.0.0.1
```

R2

```
admin@sonic:~$ vtysh
sonic# configure
sonic(config)# interface                    Ethernet5
sonic(config-if)# vrrp 1                    version 3
sonic(config-if)# vrrp 1                    priority 50
sonic(config-if)# vrrp 1                    ip 10.0.0.1
```

Note: There are more setting can be configured, such as advertisement-interval and preempt.

additional configurations

```
sonic(config-if)# vrrp 1 advertisement-interval 1500
 (10-40950) This is the interval at which VRRP advertisements will be sent. Values are given in
 milliseconds,
but must be multiples of 10

sonic(config-if)# vrrp 1 preempt
When enabled, preemption allows Backup routers with higher priority to take over Master status from the
existing Master. Enabled by default.
```

Note: The version of the VRRP group for R1 and R2 needs to be the same.

2. VRRP interface:

The configurations of R1 and R2 are identical.

R1 & R2

```
admin@sonic:~$ sudo config interface vrrp add Vrrp1-v4 Ethernet5
admin@sonic:~$ sudo config interface vrrp ip add Vrrp1-v4 10.0.0.1/31
```

3. Parent interface:

R1

```
admin@sonic:~$ sudo config interface ip add Ethernet5 192.168.0.1/24
```

Note: VRRP and parent interface need to be bound to the same VRF.

How to check it

We can use the following command to check out the brief information about VRRP group.

R1

```
admin@sonic:~$ show vrrp summary

 Interface   VRID   Priority   IPv4   IPv6   State (v4)   State (v6)

 Ethernet5   1      100        1      0      Master       Backup
```

R2

```
admin@sonic:~$ show vrrp summary

 Interface   VRID   Priority   IPv4   IPv6   State (v4)   State (v6)

 Ethernet5   1      50         1      0      Backup       Backup
```

Note:

Assume a special situation where R1 and R2 have the same priority, but R1 becomes the master faster than R2. If R2 receives the advertisement packet while in the backup role, it never has a chance to become the master, even if R2's IP is the highest, unless the master is broken. This is a known issue for FRR.

Network Monitoring

Critical Resource Monitoring (CRM)

SONiC supports the monitoring of critical switch ASIC resources by polling Switch Abstraction Interface (SAI) attributes and setting threshold values.

You can query the current usage or availability of various resources on a switch. You can also set thresholds for these resources, which result in messages being sent. The default high threshold for a resource is 85%, and the default low threshold is 70%. When a high threshold is exceeded, a Syslog Warning message is sent, and when the resource falls back below the low threshold, a Clear message is sent. The CRM feature suppresses the Syslog messages after sending ten times.

The CRM feature monitors the currently used and available number of entries for the following resources:

- IPv4 and IPv6 routes
- IPv4 and IPv6 Nexthops
- IPv4 and IPv6 Neighbors
- Next-hop group members and objects
- ACLs: Table, Group, Entries, and Counters/Statistics
- FDB Entries
- IPMC entries
- SNAT entries
- DNAT entries
- SRV6 MY_SID entries and nexthop

You can use the following commands to display information on CRM resources:

- crm show resources all - CRM information for all resources
- crm show resources acl - CRM information for ACL resources
  support : (broadcom),(intel)x1_profile|x2_profile|y1_profile|y2_profile|y3_profile
  unsupport : None
- crm show resources dnat - CRM information for DNAT resource
  support : (intel)x2_profile|y2_profile
  unsupport : (broadcom) , (intel)x1_profile|y1_profile|y3_profile
- crm show resources fdb - CRM information for FDB resources
  support : (broadcom),(intel)x1_profile|x2_profile|y1_profile|y2_profile|y3_profile
  unsupport : None

- crm show resources ipmc - CRM resources IPMC address family
  support : (broadcom)
  unsupport : (intel)x1_profile|x2_profile|y1_profile|y2_profile|y3_profile
- crm show resources ipv4 - CRM resources IPv4 address family
  support : (broadcom),(intel)x1_profile|x2_profile|y1_profile|y2_profile|y3_profile
  unsupport : None
- crm show resources ipv6 - CRM resources IPv6 address family
  support : (broadcom),(intel)x1_profile|x2_profile|y1_profile|y2_profile|y3_profile
  unsupport : None
- crm show resources mpls - CRM resources MPLS address family
  support : (intel)y3_profile
  unsupport : (broadcom),(intel)x1_profile|x2_profile|y1_profile|y2_profile
- crm show resources nexthop - CRM information for nexthop resources
  support : (broadcom),(intel)x1_profile|x2_profile|y1_profile|y2_profile|y3_profile
  unsupport : None
- crm show resources snat - CRM information for SNAT resources
  support : (broadcom),(intel)x2_profile|y2_profile
  unsupport : (intel)x1_profile|y1_profile|y3_profile
- crm show resources srv6-my-sid-entry - CRM information for SRV6 MY_SID entry
  support : (intel)y3_profile
  unsupport : (broadcom),(intel)x1_profile|x2_profile|y1_profile|y2_profile
- crm show resources srv6-nexthop - CRM information for SRV6 Nexthop
  support : (intel)y3_profile
  unsupport : (broadcom),(intel)x1_profile|x2_profile|y1_profile|y2_profile

Example

```
admin@sonic:~$ crm show resources all

Resource Name          Used Count    Available Count

ipv4_route                  33            32719
ipv6_route                  35            10206
ipv4_nexthop                 0            65534
ipv6_nexthop                 0            65534
ipv4_neighbor               33            16283
ipv6_neighbor               33             8141
nexthop_group_member         0            16384
nexthop_group                0              256
```

```
fdb_entry                0          32767
ipmc_entry               0          16383
snat_entry               0          1023

Stage    Bind Point    Resource Name      Used Count    Available Count
-------  ------------  -----------------  ------------  ----------------
INGRESS  PORT          acl_group          32            224
INGRESS  PORT          acl_table          3             2
INGRESS  LAG           acl_group          0             224
INGRESS  LAG           acl_table          0             2
INGRESS  VLAN          acl_group          0             224
INGRESS  VLAN          acl_table          0             5
INGRESS  RIF           acl_group          0             224
INGRESS  RIF           acl_table          0             5
INGRESS  SWITCH        acl_group          0             224
INGRESS  SWITCH        acl_table          0             5
EGRESS   PORT          acl_group          0             224
EGRESS   PORT          acl_table          0             2
EGRESS   LAG           acl_group          0             224
EGRESS   LAG           acl_table          0             2
EGRESS   VLAN          acl_group          0             224
EGRESS   VLAN          acl_table          0             2
EGRESS   RIF           acl_group          0             224
EGRESS   RIF           acl_table          0             2
EGRESS   SWITCH        acl_group          0             224
EGRESS   SWITCH        acl_table          0             2

Table ID         Resource Name     Used Count    Available Count

0x7000000000a20  acl_entry         2             6142
0x7000000000a20  acl_counter       2             6142
```

To set the switch ASIC SAI polling interval, use the crm config polling interval command.

To set resource thresholds, use one of the following commands for the specific resource:

crm config thresholds acl - CRM configuration for ACL resources

crm config thresholds dnat - CRM configuration for Destination NAT resources

crm config thresholds fdb - CRM configuration for FDB resources
crm config thresholds ipmc - CRM configuration for IPMC resource
crm config thresholds ipv4 - CRM resource IPv4 address-family
crm config thresholds ipv6 - CRM resource IPv6 address-family
crm config thresholds mpls - CRM resource MPLS address-family
crm config thresholds nexthop - CRM configuration for nexthop resources
crm config thresholds snat - CRM configuration for Source NAT resource
crm config thresholds srv6-my-sid-entry - CRM configuration for SRV6 MY_SID resource
crm config thresholds srv6-nexthop - CRM configuration for SRV6 Nexthop resource

Example
admin@sonic:~$ crm config polling interval 400
admin@sonic:~$ crm config thresholds acl group high 90
admin@sonic:~$ crm config thresholds acl group low 10

For more information, see the (202111) Critical Resource Monitoring section in the CLI reference guide.

Flow Sampling (sFlow)

Flow sampling (sFlow) can be used with a remote sFlow Collector to provide an accurate, detailed and real-time overview of the types and levels of traffic present on the network. The sFlow Agent samples 1 out of n packets from all data traversing a switch, re-encapsulates the samples as sFlow datagrams and transmits them to the sFlow Collector. This sampling occurs at the internal hardware level where all traffic is seen, whereas traditional probes only have a partial view of traffic as it is sampled at the monitored interface. Moreover, the processor and memory load imposed by the sFlow agent is minimal since local analysis does not take place.

As the Collector receives streams from the various sFlow agents (other switches or routers) throughout the network, a timely, network- wide picture of utilization and traffic flows is created. Analysis of the sFlow stream(s) can reveal trends and information that can be leveraged to detect, diagnose, and fix network problems.sFlow is supported on all Broadcom switches.

To configure sFlow in SONiC, first define a collector using the config sflow collector add command (up to two collectors can be defined) to specify the IPv4 or IPv6 address. By default, sFlow is disabled globally on a switch, but enabled on all interfaces. To enable sFlow globally, use the config sflow command to start sampling will start on all interfaces.

To display the global sFlow configuration, use the show sflow command.

Example
admin@sonic:~# sudo config sflow collector add collector_A 10.11.46.2 admin@sonic:~# sudo config sflow enable admin@sonic:~# show sflow sFlow Global Information: sFlow Admin State: up sFlow Polling Interval: default sFlow AgentID: lo 2 Collectors configured: Name: collector_A IP addr: 10.11.46.2 UDP port: 6343 Name: collector_lo IP addr: 127.0.0.1 UDP port: 6343

For more information on sFlow and how to configure the polling interval and sampling rates, see the (202111) sFlow section in the CLI reference guide.

Monitor Link

Monitor Link is an interface linkage scheme. It monitors the upstream interface and triggers the change of the up/down status of the downstream interface according to the change of its up/down status. In this way, the topology protocol on the downstream device switches links. Monitor Link monitors the Link where the upstream interface resides and synchronizes the downstream interface. The upstream and downstream interfaces that work together form a Monitor Link group. In some Layer 2 topology networking, when the uplink is faulty, the device on the downlink cannot sense that the link is faulty. As a result, the topology protocol cannot perform link switchover. By monitoring the uplink, Monitor Link synchronizes the downlink Settings. In this way, the uplink fault is quickly reported to the downlink, and the topology protocol on the downstream device is triggered to switch links, preventing traffic loss caused by an uplink fault for a long time.

To create the monitor link uplink interfaces and downlink interfaces use config monitor-link group member add <group_id> <types (downlink or uplink)> <interface_name>command;

delete monitor link uplink interfaces and downlink interfaces use config monitor-link group member del <group_id> <types (downlink or uplink> <interface_name>command;

config downlink up-delay use config monitor-link group downlink-up-delay set <group_id> <downlink_up_delay>command；

delete downlink up-delay use config monitor-link group downlink-up-delay set <group_id>command；

- The group id ranges from 1 to 16；

• downlink-up-delay ranges from 1 to 300s. The default value is 0.

To view information about a monitor link group, run the show monitor-link group command.

example：

```
admin@sonic:~$ sudo config monitor-link group member add 1 uplink Ethernet0
admin@sonic:~$ sudo config monitor-link group member add 1 downlink Ethernet1
admin@sonic:~$ sudo config monitor-link group downlink-up-delay set 1 10
admin@sonic:~$ show monitor-link group
------- Monitor Link Configuration -------
Group ID             : 1
Downlink Up Delay    : 10
Uplink Interface     : Ethernet0
Downlink Interface   : Ethernet1

Interface   Type      Oper Status   Shutdown Reason
----------  --------  ------------  -----------------
Ethernet0   uplink    down
Ethernet1   downlink  down          monitorlink

admin@sonic:~$
```

Configuring Drop Counters

SONiC includes a feature that monitors packet drops to help troubleshoot and debug issues in a network. The feature provides counters that are configurable for specific drop reasons and therefore can be easily applied to various network priorities.

Before configuring the packet drop counters, use the show dropcounters capabilities command to display the counter capabilities (drop counters and drop reasons) that are available on a switch. Then use the config dropcounters install command to initialize a new drop counter. You must specify a name for the counter, indicate the available counter type, and include an initial list of drop reasons.

Example

```
admin@sonic:~$ show dropcounters capabilities
   Counter Type      Total
PORT_INGRESS_DROPS          3
SWITCH_EGRESS_DROPS         2
PORT_INGRESS_DROPS:
L2 ANY
SMAC_MULTICAST
SMAC_EQUALS_DMAC
INGRESS VLAN FILTER
_   _
EXCEEDS L2 MTU
_ _
SIP CLASS E
_    _
SIP LINK LOCAL
_   _
DIP LINK LOCAL
_   _
UNRESOLVED_NEXT_HOP
DECAP_ERROR
SWITCH_EGRESS_DROPS:
L2 ANY
L3 ANY
A CUSTOM REASON
_     _
admin@sonic:~$ sudo config dropcounters install DEBUG_2 PORT_INGRESS_DROPS
\[EXCEEDS_L2_MTU,DECAP_ERROR\] -d
"More port ingress drops" -g BAD -a BAD_DROPS
```

To display the current configured drop counters, use the show dropcounters configuration command. To display the collected statistics for the configured drop counters, use the show dropcounters count command.

Example

```
admin@sonic:~$ show dropcounters configuration
Counter   Alias     Group  Type              Reasons              Description
DEBUG_0   RX_LEGIT  LEGIT  PORT_INGRESS_DROPS  SMAC_EQUALS_DMAC    Legitimate
port-level RX pipeline drops
```

```
INGRESS_VLAN_FILTER
DEBUG_1 TX_LEGIT None SWITCH_EGRESS_DROPS EGRESS_VLAN_FILTER Legitimate
switch-level TX pipeline
drops
admin@sonic:~$ show dropcounters configuration -g LEGIT
Counter Alias Group Type Reasons Description
DEBUG_0 RX_LEGIT LEGIT PORT_INGRESS_DROPS SMAC_EQUALS_DMAC Legitimate
port-level RX pipeline drops
INGRESS VLAN FILTER
_ _
admin@sonic:~$ show dropcounters counts -t SWITCH EGRESS DROPS
_ _
DEVICE TX LEGIT
_
------ --------
sonic 1000
```

For more information, see the (202111) Drop Counters section in the CLI reference guide.

Setting the Watermark Telemetry Interval

Network telemetry monitors network health and helps identify problems and latency issues so they can be resolved quickly. SONiC supports a telemetry system whereby devices in the network periodically stream information to data collectors in the management system.

The interval for the telemetry can be set using the config watermark telemetry interval command. This parameter leverages the Linux timer and is set to a default 120 seconds. In the following example, the telemetry interval is set to 999 seconds.

Example
admin@sonic:~$ sudo config watermark telemetry interval 999

For more information, see the (202111) Watermark section in the CLI reference guide.

Inband Network Telemetry (INT)

INT (In band Telemetry) is a network monitoring technology that collects data from devices. Devices equipped with INT function will actively send collected data to the collector, providing real-time and high-speed data

collection function. The collector analyzes the collected data to achieve the purpose of monitoring the performance and network operation of network devices.

INT is only supported on the following models:

- N9200-64DC
- N9500-128QC
- N9500-64OC

INT only supports transit role.

To configure INT, using the config telemetry ifa command.

- enable or disable INT
  config telemetry ifa enable|disable

- add or delete device id
  config telemetry ifa device-id set|unset <address>

- configure or unconfigure transit interface
  config telemetry ifa interface transit-role enable|disable <interface_name>

- show INT configuration
  show telemetry ifa

Example1: device-id

The following example shows how to config int device id, using config telemetry ifa device-id command.

Example

config telemetry ifa device-id set 10.0.0.1

Example2: int status

The following example shows how to enable or disable int status, using config telemetry int enable|disable command.

Example

config telemetry ifa enable

config telemetry ifa disable

Example3: trasit-role interface

The following example shows how to configure int transit-role interface, using config telemetry ifa transit-role command. Specify the transit-role interface to Ethernet0 or remove the Ethernet0 from transit role interface.

Example

config telemetry ifa interface transit-role enable Ethernet0

config telemetry ifa interface transit-role disable Ethernet0

Example4: show int information

The following example shows int configuration information, using show telemetry ifa command.

Example

show telemetry ifa

Mirror On Drop (MOD)

The MOD (Mirror On Drop) function can detect packet loss during the internal forwarding process of packets within the device. When packet loss occurs, the device will send the reason for packet loss and the characteristics of the discarded message to the collector.

MOD supports users to configure packet drop reasons through CLI, and currently supports the following 8 packet drop reasons:

- --invalid-vlan
- --next-hop-drop
- --ingress-fp-drop
- --l3-hdr-error
- --l3-ttl-error
- --mc-drop
- --ipv6-hdr-error
- --ipv4-hdr-error

MOD is only supported on the following models:

- N9200-64DC

MOD only supports UDP protocol when sending packets to the collector; MOD does not support mirror to trunk port.

To configure MOD, using the config telemetry mod command.

- add or delete device id
  config telemetry mod device-id set|unset <address>

- config reason list
  config telemetry mod reason-list set [OPTIONS] <name>
  config telemetry mod reason-list unset <name>

- add or remove collector
  config telemetry mod collector add <collector_name> <src_ip_addr> <dst_ip_addr> <src_port> <dst_port> [vid]
  config telemetry mod collector del <collector_name>

- show MOD configuration
  show telemetry mod

Example1: device-id

The following example shows how to config mod device id, using config telemetry mod device-id command.

Example

config telemetry mod device-id set 10.0.0.1
config telemetry mod device-id unset 10.0.0.1

Example2: reason-list

The following example shows how to configure mod reason-list, using config telemetry mod reason-list command. Specify the reason-list name to test, drop reason to invalid vlan or remove the reason-list test.

Example

config telemetry mod reason-list set test --invalid-vlan
config telemetry mod reason-list unset test

Example3: collector

The following example shows how to configure mod collector, using config telemetry mod collector command. Specify the collector name to test, src ip to 192.168.1.1, dst ip to 192.168.1.2, src port to 10, dst port to 20, the optional parameter vlan to 100 or remove the collector test.

Example

config telemetry mod collector add test 192.168.1.1 192.168.1.2 10 20 100
config telemetry mod collector del test

Example4: show mod information

The following example shows mod configuration information, using show telemetry mod command.

Example

show telemetry mod

Counter Space Design

"Counter Space" refers to the counter design on Sonic, where an independent counter view is created for each login account.

Therefore, the execution of "clear counter" will only affect the counter view of the current login account. That is, the counter values seen by different login account might be different.

Example for "clear counter":

Let's start with the example below.

Example
admin@sonic:~$ show int count |grep Ethernet0 Ethernet0 U 74 0.00 B/s 0.00% 0 21 0 79 0.27 B/s 0.00% 0 0 0 admin@sonic:~$ sudo config interface shutdown Ethernet0 admin@sonic:~$ sudo sonic-clear counter Cleared counters admin@sonic:~$ show int count | grep Ethernet0 Ethernet0 X 74 0.00 B/s 0.00% 0 21 0 81 0.00 B/s 0.00% 0 0 0

At the first command, it executes "show int counter" with "grep Ethernet0" to filter the counter output that contains "Ethernet0", which will show the interface counter of Ethernet0.

At the second command, it runs "sudo config interface shutdown Ethernet0" to prevent any traffic running in the background interfering with the result of clear counter.

At the third command, it executes "sudo" command with the CLI command to clear the counter. Note that the effect of "sudo" will change the effective login account as "root" and execute the commands behind "sudo".

At the last command, we executes the command to show the counter of Ethernet0 again and find that the counter has not been cleared.

This is because the current login user account is "admin". The execution of "show int counter" displays the counter value belongs to admin's counter space. The counter that we cleared at the second command takes effect on root's counter space. Therefore, the counter status in admin's counter space is not affected by "sudo sonic-clear counter".

The Example of the Correct Way to Use Counter Commands

Example	
admin@sonic:~$ show int count | grep Ethernet0 Ethernet0 X 74 0.00 B/s 0.00% 0.00% 0 0 0	0 21 0 81 0.00 B/s
admin@sonic:~$ sudo config interface shutdown Ethernet0 admin@sonic:~$ sonic-clear counter Cleared counters admin@sonic:~$ show int count | grep Ethernet0 Ethernet0 X 0 0.00 B/s 0.00% 0.00% 0 0 0	0 0 0 0 0.00 B/s

It only makes sense to use the same account to execute the counter related CLI commands so that it takes effect on the same counter space.

QoS Configuration

Class of Service (CoS)

QoS Classification is used to classify the packets to assign a Traffic Class (TC) value based on service requirements. TC is the internal priority in switch (8 possible value) using for packet buffering and scheduling. TC-to-priority-group (PG) and TC-to-Queue maps depend on the TC maps to the ingress buffer (priority-group) and egress queue, respectively.

The TC is determined based on the priority of the incoming packet and the trust mode of a port. There have two trust mode values: dot1 , and dscp .

- dot1p:
  If the packet is VLAN tagged, the Dot1p is used to map TC.
  Otherwise, if the packet is un tagged, the port default priority (0, and can not be configured) is used as Dot1p.
- dscp:
  If the packet is IP packet, the DSCP is used to map TC.
  Otherwise, if the packet is VLAN tagged, the Dot1p is used.
  Otherwise, if the packet is un tagged, the port default priority is used.

To determine the trust mode of a port, use the following rules:

The default trust mode of a port is dot1p.
If a port be configured with a DSCP-to-TC mapping, the trust mode of the port is dscp.
Default mapping tables are used to determine the TC when Dot1p-to-TC is not configured on a port.
Default mapping tables vary by switch platform.

Table 1: Default Dot1p-to-TC mapping on Broadcom switches.

Dot1p	0	1	2	3	4	5	6	7
TC	0	1	2	3	4	5	6	7

Table 2: Default DSCP-to-TC mapping.

DSCP	0 ~ 63
TC	0

Dot1p-to-TC Mapping

Assign the TC based on PCP in the VLAN tag.

The mapping can be configured per ingress physical port.

DSCP-to-TC Mapping

Assign the TC based on DSCP field in the IP header.

The mapping can be configured per ingress physical port.

TC-to-queue Mapping

Assign the queue ID based on TC.

The mapping can be configured per ingress physical port.

The default mapping tables vary by switch platform.

Table 6: Default TC-to-Queue mapping on Broadcom switches.

TC	0	1	2	3	4	5	6	7
Queue	0	1	2	3	4	5	6	7

To configure a Dot1p-to-TC profile, using config qos dot1p-tc command. To configure Dot1p-to-TC on an interface, using config interface qos dot1p-tc command.

To configure a DSCP-to-TC profile, using config qos dscp-tc command. To configure DSCP-to-TC on an interface, using config interface qos dscp-tc command.

To configure a TC-to-queue profile, using config qos tc-queue command. To configure TC-to queue on an interface, using config interface qos tc-queue command.

Apply a Dot1p-to-TC Table to a Port

Apply a DSCP-to-TC Table to a Port

Apply a TC-to-Queue Table to a Port

Apply a Dot1p-to-TC Table to a Port

1. Create a Dot1p-to-TC profile, using config qos dot1p-tc command. The following example is to create a Dot1p-to-TC profile dot1p-tc - prof and adds a mapping dot1p(0) to TC(3) when creating.

```
admin@sonic:~$ sudo config qos dot1p-tc add dot1p-tc-prof --dot1p 0 --tc 3
```

2. Update the Dot1p-to-TC profile for adding/removing mapping, using config qos dot1p-tc update command. The following example is to add a new mapping dot1p(1) to TC(4).

```
admin@sonic:~$ sudo config qos dot1p-tc update dot1p-tc-prof --dot1p 1 --tc 4
```

3. Apply the Dot1p-to-TC profile to a port, using config interface qos dot1p-tc bind command. The following example is to apply the Dot1p-to-TC profile dot1p-tc-prof to Ethernet0 .

```
admin@sonic:~$ sudo config interface qos dot1p-tc bind Ethernet0 dot1p-tc-prof
```

Apply a DSCP-to-TC Table to a Port

1. Create a DSCP-to-TC profile, using config qos dscp-tc command. The following example is to create a DSCP-to-TC profile dscp-tc-prof and adds a mapping DSCP(8) to TC(3) when creating.

```
admin@sonic:~$ sudo config qos dscp-tc add dscp-tc-prof --dscp 8 --tc 3
```

2. Update the DSCP-to-TC profile for adding/removing mapping, using config qos dscp-tc update command. The following example is to add a new mapping DSCP(32) to TC(4).

```
admin@sonic:~$ sudo config qos dscp-tc update dscp-tc-prof --dscp 32 --tc 4
```

3. Apply the DSCP-to-TC profile to a port, using config interface qos dscp-tc bind command. The following example is to apply the DSCP-to-TC profile dscp-tc-prof to Ethernet0 .

```
admin@sonic:~$ sudo config interface qos dscp-tc bind Ethernet0 dscp-tc-prof
```

Apply a TC-to-Queue Table to a Port

1. Create a TC-to-queue profile, using config qos tc-queue command. The following example is to create a TC-queue profile tc-queue - prof and adds a mapping TC(3) to queue(5) when creating.

```
admin@sonic:~$ sudo config qos tc-queue add tc-queue-prof --tc 3 --queue 5
```

2. Update the TC-queue profile for adding/removing mapping, using config qos tc-queue update command. The following example is to add a new mapping TC(4) to queue(6).

```
admin@sonic:~$ sudo config qos tc-queue update tc-queue-prof --tc 4 --queue 6
```

3. Apply the TC-queue profile to a port, using config interface qos tc-queue bind command. The following example is to apply the TC- queue profile tc-queue-prof to Ethernet0 .

```
admin@sonic:~$ sudo config interface qos tc-queue bind Ethernet0 tc-queue-prof
```

For more information, see the (202111) QoS section in the CLI reference guide.

QoS Marking

Once the packets have been classified by QoS Classification, QoS Marking allows to set or change the priority value in the packets before sending them. The new priority value is brought to the next switches for fine-tuning or uniform the class of the traffic in the network.

QoS Marking enables to rewrite the Dot1p or DSCP of incoming packets based on Traffic Class (TC) value.

Enable Dot1p rewrite by configuring a TC-to-Dot1p mapping on an egress port. On N9200-64DC switch, the Dot1p rewrite is always enabled.

Enable DSCP rewrite by configuring a TC-to-DSCP mapping on an egress port. The DSCP rewrite is disabled by default on all ports.

Table 1: Default TC-to-Dot1p mapping.

TC	0	1	2	3	4	5	6	7
Dot1p	0	1	2	3	4	5	6	7

Table 2: Default TC-to-DSCP mapping.

TC	0	1	2	3	4	5	6	7
DSCP	0	0	0	0	0	0	0	0

TC-to-Dot1p Mapping

Rewrite the PCP in the VLAN tag based on the TC-to-Dot1p mapping table.

Any TC value that is not specified in the configuration is assigned to a default Dot1p value for rewriting. Refer to Table 1 for the default mapping.

The mapping can be configured per egress physical port.

TC-to-DSCP Mapping

Rewrite the DSCP field in the IP header based on the TC-to-DSCP mapping table.

Any TC value that is not specified in the configuration is assigned to a default DSCP value for rewriting. Refer to Table 2 for the default mapping.

The mapping can be configured per egress physical port.

To configure a Dot1p remarking profile, using config qos remark dot1p command. To configure Dot1p remarking on an interface, using con fig interface qos remark dot1p command.

To configure a DSCP remarking profile, using config qos remark dscp command. To configure DSCP remarking on an interface, using config interface qos remark dscp command.

Configuring Marking PCP

1. To create a Dot1p-TC profile, using config qos dot1p-tc command. For example, create a Dot1p-TC profile "dot1p_tc" to map the packet Dot1p priority of 0,2 to switch priority 0,2, respectively.

Example

```
sudo config qos dot1p-tc add dot1p_tc --dot1p 0 --tc 0
sudo config qos dot1p-tc update dot1p_tc --dot1p 2 --tc 2
```

2. To configure a Dot1p-TC profile on the ingress port, using config interface qos dot1p-tc command. For example, configure the Dot1p- TC profile "dot1p_tc" on Ethernet0 as the following command.

Example

```
sudo config interface qos dot1p-tc bind Ethernet0 dot1p_tc
```

3. To create a Dot1p remarking profile, using config qos remark dot1p command. For example, create a Dot1p remarking profile "remark_pcp" to map the switch priority 0,2 to PCP value 1,3 for rewriting.

Example

```
sudo config qos remark dot1p add remark_pcp --tc 0 --dot1p 1
sudo config qos remark dot1p update remark_pcp --tc 2 --dot1p 3
```

4. To enable Dot1p remaking on the egress port, using config interface qos remark dot1p bind command. For example, enable Dot1p remarking with the mapping "remark_pcp" on Ethernet2 as the following command.

Example

```
sudo config interface qos remark dot1p bind Ethernet2 remark_pcp
```

5. Verify the configuration, using show interface qos command.

Example

```
admin@sonic:~$ show interface qos
Ethernet0:
  Dot1p to TC: dot1p_tc

Ethernet2:
  Dot1p remark: remark_pcp
```

Configuring marking DSCP

1. To create a DSCP-TC profile, using config qos dscp-tc command.

For example, create a DSCP-TC profile "dscp_tc" to map the DSCP 0,16 to switch priority 0,2, respectively.

Example

```
sudo config qos dscp-tc add dscp_tc --dscp 0 --tc 0
sudo config qos dscp-tc update dscp_tc --dscp 16 --tc 2
```

2. To configure a DSCP-TC profile on the ingress port, using config interface qos dscp-tc command.

For example, configure the DSCP-TC profile "dscp_tc" on Ethernet0 as the following command.

Example

```
sudo config interface qos dscp-tc bind Ethernet0 dscp_tc
```

3. To create a DSCP remarking profile, using config qos remark dscp command.

For example, create a DSCP remarking profile "remark_dscp" to map the switch priority 0,2 to DSCP value 1,17 for rewriting.

Example

```
sudo config qos remark dscp add remark_dscp --tc 0 --dscp 1
sudo config qos remark dscp update remark_dscp --tc 2 --dscp 17
```

4. To enable DSCP remarking on the egress port, using config interface qos remark dscp bind command.

For example, enable DSCP remarking with the mapping "remark_dscp" on Ethernet2 as the following command.

Example

```
sudo config interface qos remark dscp bind Ethernet2 remark_dscp
```

5. Verify the configuration, using show interface qos command.

Example

```
admin@sonic:~$ show interface qos
Ethernet0:
  DSCP to TC: dscp_tc

Ethernet2:
  IP DSCP remark: remark_dscp
```

For more information, see the (202111) QoS (new) section in the CLI reference guide.

Explicit Congestion Notification (ECN)

Explicit congestion notification (ECN) enables traffic congestion control between two ECN-enabled devices. When the congestion is detected by the switch, the ECN field is set in the IP header of a packet before forwarding it. If the receiver device received the packets with ECN marked, it sends the congestion notification packets (CNP) to the sender device to reduce the transmit packet rate. The CNP packets continue to send to the sender until the congestion is released.

ECN uses the last two bits, ECT and CE, of the DS field of the IP header as the following:

ECT	EC	Combinations Indicate
0	0	Non ECN-Capable Transport, Non-ECT
1	0	ECN Capable Transport, ECT(0)
0	1	ECN Capable Transport, ECT(1)
1	1	Congestion Encountered, CE

ECN uses the WRED thresholds being applied to an egress queue. When ECN enabled, process the packets based on following rules:

- When the average queue length is below the minimum threshold , no packet is marked.
- When the average queue length is between minimum threshold and maximum threshold , one of the following scenarios can occur:
- If received the IP packet with ECT(0) or ECT(1) codepoints, changes the ECT and EC bits to 1 based drop probability and forwards the packet.
- If received the IP packet with CE codepoint '11', the packet is transmitted. No further marking is required.
- If received the IP packet with not-ECT codepoint '00', the packet is transmitted.
- When the average queue length exceeds the maximum threshold , all the incoming IP packets with the ECT(0) and ECT(1) codepoints are marked.

For switches using Broadcom ASIC, when ECN is enabled, WRED is also enabled at the same time. WRED is applied to all non-IP packets and IP packets with the Non-ECN capable (00) codepoint.

RDMA over Converged Ethernet is a network protocol that allows Remote Direct Memory Access (RDMA) over an Ethernet network. RoCEv2 (RoCE version 2) enhances RoCEv1 with a UDP/IP and using the well -known UDP port 4791. It defines congestion control mechanism based on ECN marking. The RoCEv2 Congestion Management defines the three points:

Reaction Point (RP): reduces the transmission rate, according to the received CNPs. PR is the sender device.

Congestion Point (CP): detects congestion and marks EC bit. CP is a switch.

Notification Point (NP): reacts to the ECN marked packets by sending congestion notification packets (CNPs). NP is the receiver device.

The following processes show how ECN works using RoCEv2 as an example.

```
RP(sender) ---------- CP(switch) ---------- NP(receiver)
```

RoCEv2 with ECT(0) or ECT(1)

1. ---------->

The RP(sender) send the RoCEv2 packet with ECT(0) or ECT(1) to NP(receiver).

set ECN field to '11'

2. --------->

If the congestion is detected by CP(switch), it sets the ECN field to '11' of the packet.

notify with RoCEv2 CNP packet

3. <----------

When the NP(receiver) receives the packet with ECN field as '11', it sends a RoCEv2 CNP packet to the RP(sender).

4. Change the transmission rate

When the RP(sender) receives the CNP packet, it slows down the transmission rate.

To configure the WRED profile, referring this document Weighted Random Early Detection (WRED) for more information.

The ECN can be configured through CLI command. You need to first configure a WRED profile, using config wred add command, and then bind the profile to the port interface's queues, using config interface wred bind queue command.

The following example is to configure a WRED profile wred-prof. This example uses green max threshold of 800k bytes, green min threshold of 100k bytes, and green drop probability of 100%.

Example

```
admin@sonic:~$ sudo config wred add wred-prof --mode ecn --gmin 100000 --gmax 800000 --gdrop 100
```

Once a profile has been created, it can be applied on the port interface's queues, as follows:

Example

```
admin@sonic:~$ sudo config interface wred bind queue Ethernet0 0 wred-prof
```

Verify the configuration using show wred and show interface wred command.

Example

```
admin@sonic:~$ show wred
Profile: wred-prof
Color    Mode     Min Threshold    Max Threshold    Drop Probability
Green    ECN          100000           800000              100
Yellow   ECN
Red      ECN

admin@sonic:~$ show interfaces wred
Ethernet0
Queue: 0
ECN/WRED: wred-prof
Color    Mode     Min Threshold    Max Threshold    Drop Probability
Green    ECN          100000           800000              100
Yellow   ECN
Red      ECN

admin@sonic:~$
```

The following example is to unbind the profile from the port interface's queues, and delete the profile:

Example

```
admin@sonic:~$ sudo config interface wred unbind queue Ethernet0 0 wred-prof
admin@sonic:~$ sudo config wred del wred-prof
```

For more information, see the (202111) WRED & ECN (updated, add first two sections) section in the CLI reference guide.

Dynamic Explicit Congestion Notification（DECN）

DECN (Dynamic Explicit Congestion Notification) is a dynamic network congestion control protocol. Its aim is to optimize network traffic control, reduce network congestion and enhance network performance by using algorithms to real-time adjust the network congestion notification mechanism.

DECN uses the WRED thresholds being applied to an egress queue. When DECN enabled, process the packets based on following rules:

When the average queue length is below the minimum threshold, no packet is marked.When the average queue length is between minimum threshold and maximum threshold, one of the following scenarios can occur:

If received the IP packet with ECT(0) or ECT(1) codepoints, changes the ECT and EC bits to 1 based drop probability and forwards the packet.

If received the IP packet with CE codepoint '11', the packet is transmitted. No further marking is required.

If received the IP packet with not-ECT codepoint '00', the packet is transmitted.

When the average queue length exceeds the maximum threshold, all the incoming IP packets with the ECT(0) and ECT(1)codepoints are marked.

DECN is only supported on the following models:

- N9200-64DC
- N9500-128QC
- N9500-64OC

DECN and static ECN modes are mutually exclusive and cannot be configured at the same time.

The DECN can be configured through CLI command. You need to first configure a DECN profile, using config decn add command, and then bind the profile to the port interface's queues, using config interface decn bind queue command.

Example

config decn profile add ecn_profile --bufferusage 50 --throughputrate 50 --gmin 5000--gmax 10000--gdrop 100.

Once a profile has been created, it can be applied on the port interface's queues, as follows:

Example

config interface decn bind queue Ethernet0 3 ecn_profile.

Verify the configuration using show decn profile and show interface decn command.

Example

show decn profle

show interface decn config

The following example is to unbind the profile from the port interface's queues, and delete the profile:

Example

config interface decn unbind queue Ethernet0 3

config decn del ecn_profile

Priority Flow Control (PFC)

Priority Flow Control (PFC), which is defined in IEEE 802.1Qbb, provides hop-by-hop layer 2 congestion control. PFC extends the IEEE 802.3x Ethernet PAUSE, which allows pausing the traffic based on its priority (802.1p) rather than pausing all the traffic on a link.

When a congestion happens on the ingress buffer of the downstream switch, it sends the PFC PAUSE frame to the upstream switch. The PFC PAUSE frame contains a 2-octet priority specifying which traffic of the priority class should be paused and a 2-octet time value specifying the pause time. When the upstream switch receives a PFC PAUSE frame, it stops transmitting based on the priorities and the period in the PFC frame. The upstream switch resumes transmitting after the time expires or receives a new PFC PAUSE frame with pause time is zero. All the incoming packets are stored in a central buffer called the Shared Buffer.

The Shared Buffer can be divided into three parts: reserved buffer, shared buffer, and headroom buffer. Each buffer has a maximum limit value and uses counters to record the usage. Each packet is counted in both ingress and egress based on its priority. Packet drop happens if the usage hits the maximum limit. The PFC PAUSE frame is generated if the usage of the ingress buffer reaches a predetermined threshold called the XOFF limit (the sum of the size of the reserved buffer and the shared buffer). The headroom buffer

is used to store packets which are in flight during the time before the PAUSE takes effect. This includes the time for the PFC PAUSE frame to be sent to the upstream switch and the time for the egress queue to stop the transmission. The PFC PAUSE frame is generated with pause time as zero if the usage of the ingress buffer falls below the predetermined XON limit.

This section describes the parameters for the PFC feature:

- PFC is supported to enable per-port per-priority. PFC is disabled by default on all ports.
- PFC is supported to configure the XOFF limit of a priority-group. An 802.3x PAUSE frame is generated when the buffer limit at the ingress port reaches this threshold.
- PFC is supported to configure the XON limit of a priority-group. The XOFF state will be released when the buffer limit at the ingress port falls below this threshold.
- PFC is supported to configure the traffic-class-to-priority-group (TC2PG) mapping on each port that allows mapping the traffic class to a PG.

- PFC is supported to configure the Priority-to-queue (PRIO2Q) mapping on each port that allows mapping the priority(s) in the PFC frame to a queue. The mapping determines which queue will be paused when an 802.3 PAUSE frame with that priority is received.

Traffic-class-to-priority-group (TC2PG) Mapping.

The TC2PG map has two purposes. One is to determine the priority-group (PG) for buffering. The other is to determine the priority in the PFC PAUSE frame. The behavior of the mapping varies by switch platform.

Traffic-class-to-priority-group (TC2PG) Mapping - Broadcom switches.

The TC is mapped to the priority-group (PG) based on the following rules:

If the packet is VLAN tagged, the Dot1p in the packet is used as TC to map the PG.

If the packet is un tagged, the TC, which is determined by QoS classification, is used to map the PG.

The PG range is 0 to 7.

The TCs, which are mapped to the same PG, are used as priorities in the PFC PAUSE frame if the PG is under pressure.

Table 1 illustrates the default TC-to-PG mapping on Broadcom Trident2plus, Trident3, Tomahawk, and Tomahawk2 switches.

Table 2 illustrates the default TC-to-PG mapping on Broadcom Trident4, Tomahawk3, and Tomahawk4 switches.

Table 1: Default TC-to-PG mapping on Broadcom Trident2plus, Trident3, Tomahawk and Tomahawk2 switches.

TC	0 ~ 7
PG	7

Table 2: Default TC-to-PG mapping on Broadcom Trident4, Tomahawk3, and Tomahawk4 switches.

TC	0	1	2	3	4	5	6	7
PG	0	1	2	3	4	5	6	7

Priority-to-queue (PRIO2Q) Mapping

Maps the 802.1p priority(s) in PFC frame to the queue specifying which queue stops the transmission if received the PFC frame with the given priority(s). Only Broadcom switches is supported to change the mapping. Table 3 shows the default mapping.

Table 4: Default Priority-to-Queue mapping .

PFC priority	0	1	2	3	4	5	6	7
queue	0	1	2	3	4	5	6	7

To configure the Shared Buffer refer to Shared Buffer Configuration for more information.
To configure the TC-to-PG mapping, using config qos tc-pg and config interface qos tc-pg commands.

To configure the PFC-priority-to-Queue mapping, using config qos pfc-queue and config interface qos pfc-queue commands.
To enable PFC on an interface, use the config interface pfc priority command.

Configuration Example

1. Set buffer pools and profiles for lossless traffic.

To configure the buffer settings for lossless traffic, using config qos reload command.

Example
admin@sonic:~$ sudo config qos reload

The below buffer pool and profiles will be created after running the above command.

- ingress_lossy_pool - ingress buffer pool for lossy traffic
- ingress_lossless_pool - ingress buffer pool for lossless taffic
- egress_lossy_pool - egress buffer pool for lossy traffic
- egress_lossless_pool - egress buffer pool for lossless traffic
- ingress_lossy_profile - ingress buffer profile that is used to apply the lossy priority-groups. By default, all priority- groups on all ports are applied using this profile.
- ingress_lossless_profile - ingress buffer profile that is used to apply lossless priority-groups.
- egress_lossy_profile - egress buffer profile that is used to apply lossy queues. By default, all queues on all ports are are applied using this profile.
- egress_lossless_profile - egress buffer profile that is used to apply lossless queues.

The buffer pool and profiles are calculated based on the formulas given on Shared Buffer Configuration.
The parameters used per device shown in appendix.

Then. Save the configuration and restart the switch for the change to take effect.

Example
admin@sonic:~$ sudo config save -y admin@sonic:~$ sudo reboot

2. Configure the priority-group to use lossless buffer settings.

To configure a priority-group to apply the PFC thresholds using the predefined profile ingress_lossless_profile , use the config interface buffer command. For example, the following command shows how to apply the profile to priority-group 3:

Example
admin@sonic:~$ sudo config interface buffer bind priority-group all 3 ingress_lossless_profile

Use the ingress_lossless_profile_10g for 10G ports and ingress_lossless_profile_40g for 40G ports. The following example shows how to apply the profile to priority group 3 on a 40G port:

Example
admin@sonic:~$ sudo config interface buffer bind priority-group Ethernet49 3 ingress_lossless_profile_40g

3. Configure the queue to use lossless buffer settings.

To configure the queue for lossless traffic, use the config interface buffer command. egress_lossless_profile is predefined buffer profile. Use queue 3 as example:

Example
admin@sonic:~$ sudo config interface buffer bind queue all 3 egress_lossless_profile

4. Configure the TC-to-PG and TC-to-Queue QoS mapping to specify the PG and queue which configured on step 2 and 3.

To configure the TC-to-PG QoS mapping, it need to create a TC-to-PG profile, using config qos tc-pg add command. Then apply the profie to an interface, using config interface qos tc-pg bind command.

Example
admin@sonic:~$ sudo config qos tc-pg add tc-pg-prof --tc 3 --pg 3 admin@sonic:~$ sudo config interface qos tc-pg bind all tc-pg-prof

To configure the TC-to-Queue QoS mapping, it need to create a TC-to-Queue profile, using config qos tc-queue add command. Then apply the profile to an interface, using config interface qos tc-queue bind command.

Example

```
admin@sonic:~$ sudo config qos tc-queue add tc-queue-prof --tc 3 --queue 3
admin@sonic:~$ sudo config interface qos tc-queue bind all tc-queue-prof
```

NOTE

- When configuring the TC-to-PG and TC-to-Queue QoS maps, you must separate the lossy and lossless traffic by assigning them to different PG/queues. For example, if a PG/queue is configured to store both lossy and lossless traffic, the lossless traffic can react to buffer congestion, but the lossy traffic cannot. This can cause the buffer to fill up with lossy traffic and starve the lossless traffic. Eventually, the lossless traffic will be dropped. For Broadcom Tomahawk4 switches, if you combine the lossy and lossless traffic into the same PG/queue, the buffer settings and QoS maps configuration will be held until the correct configuration is completed.
- During the process of changing the buffer settings and TC-to-PG and TC-to-Queue mappings, all packets will be dropped . Packet forwarding will resume once all the changes have been configured into the hardware.

It is recommended to use a 1:1 mapping for TC-to-PG and TC-to-Queue mapping for PFC. The following example shows the mapping for PFC priority 0 to 7:

TC 0 maps to PG 0 and queue 0
TC 1 maps to PG 1 and queue 1
TC 2 maps to PG 2 and queue 2
TC 3 maps to PG 3 and queue 3
TC 4 maps to PG 4 and queue 4
TC 5 maps to PG 5 and queue 5
TC 6 maps to PG 6 and queue 6
TC 7 maps to PG 7 and queue 7

5. Enable the PFC on an interface.

To enable PFC on an interface, use the config interface pfc priority command and set the priority to "on". When successfully configured, the current lossless priorities on the interface are displayed.

Example

```
admin@sonic:~$ sudo config interface pfc priority Ethernet60 3 on
Interface---Lossless priorities--

Ethernet60    3
```

To configure asymmetric PFC on an interface, use the config interface pfc asymmetri command and set it to “on” . When set to asymmetric PFC, an interface accepts pause frames for all priorities, but only sends pause frames for lossless priorities.

To check the status of asymmetric PFC, use the show interfaces status command.

Example
admin@sonic:~$ sudo config interface pfc asymmetric Ethernet60 on

6. Configure the mapping between the packet priority to Traffic Class (TC).

To configure the packet priority to switch priority mapping. Use Dot1p-to-TC as example, use config qos dot1p-tc add command to create the profile. Then use config interface qos dot1p-tc bind command to apply the profile to an interface.

Example
admin@sonic:~$ sudo config qos dot1p-tc add 1p-tc-prof --dot1p 3 --tc 3 admin@sonic:~$ sudo config interface qos dot1p-tc bind Ethernet60 1p-tc-prof

It is recommended to use a 1:1 mapping for Dot1p-to-TC mapping for PFC. For example:

- 802.1p 0 maps to TC 0
- 802.1p 1 maps to TC 1
- 802.1p 2 maps to TC 2
- 802.1p 3 maps to TC 3
- 802.1p 4 maps to TC 4
- 802.1p 5 maps to TC 5
- 802.1p 6 maps to TC 6
- 802.1p 7 maps to TC 7

Summary of the above example, the QoS mapping is shown in the following table:

Dot1P	TC	PG	Queue
3	3	3	3

If the buffer usage of PG 3 reaches the xoff-threshold, a PFC frame with 3 priority will be sent.

If a PFC frame with 3 priority is received, Queue 3 will pause.

For more information, see the (202111) PFC, (202111) QoS section in the CLI reference guide.

Appendix

This appendix shows detailed information on the buffer pool and profiles per device.

N9200-64DC (using Broadcom Tomahawk4)

N9500-128QC (using Broadcom Tomahawk5)

N9500-64OC (using Broadcom Tomahawk5)

The parameters of the ingress lossy pool named as "ingress_lossy_pool" are:

- Size of 51,989,336 bytes

The parameters of the egress lossy pool named as "egress_lossy_pool" are:

- Size of 51,989,336 bytes

The parameters of the ingress lossy profile named as "ingress_lossy_profile" are:

- Pool of "ingress_lossy_pool"
- Reserved size of 0 bytes
- Shared dynamic mode, alpha of 1 (66%)

The parameters of the egress lossy profile named as "egress_lossy_profile" are:

- Pool of "egress_lossy_pool"
- Reserved size of 0 bytes
- Shared dynamic mode, alpha of 0 (50%)

The parameters of the ingress lossless pool named as "ingress_lossless_pool" are:

- Size of 10,171,826 bytes
- Headroom size of 50,859,134 bytes

The parameters of the egress lossless pool named as "egress_lossless_pool" are:

- Size of 10,171,826 bytes

The parameter of the ingress lossless profile named as "ingress_lossless_profile" are:

- Pool of "ingress_lossless_pool"
- Reserved size of 0 bytes
- Shared dynamic mode, alpha of 1 (66%)
- Headroom size of 971,176 bytes (Note)
- Xon of 3,051,548 bytes
- Xon-offset of 18,432 bytes

The parameter of the egress lossless profile named as "egress_lossless_profile" are:

- Pool of "egress_lossless_pool"

- Reserved size of 0
- Shared dynamic mode, alpha of 1 (66%)

Note: The headroom of the priority-group can be calculated using the following formulas:

headroom size of a PG = 2 × (latency + media delay + cable length × 5) × pps × packets to cells × cell size ÷ 109 packets to cells conversion = ceiling[(packet size + M) ÷ cell size], M is 64 for Broadcom switches

The parameters are used as below:

- Latency of 1,000 nanoseconds for Broadcom switches.
- Media delay of 100 nanoseconds.
- Cable length of 300 meters.
- Packet per second (PPS) of 735,294,118 (using the size of 68 bytes).
- Port speed of 400 Gb/s.
- Cell size of 254 bytes.

The maximum number of the port priority-group is supported to enable PFC without packet loss is 52 .

50,859,134 / 971,176 = 52

The parameters used in above formula:

- 50,859,134: headroom size in ingress lossless pool (ingress_lossless_pool).
- 971,176: headroom size in ingress lossless profile (ingress_lossless_profile).

Starting the PFC Watchdog

Priority Flow Control (PFC) watchdog provides to detect and mitigate the PFC deadlock (storm) problem. PFC deadlock is the one of well-known problems if used the PFC mechanism. PFC deadlock is caused the PFC PAUSE frames are propagated running into a loop. When the deadlock is happening, the buffer resource in each switch is not able to release to lift the XOFF (transmission-off) state. In this situation, no any packet can be transmitted in the whole network. To remove the deadlock, it can temporatily disables PFC to ignore the PFC PAUSE frames. Or remove all packets have been stored in the queue.

- PFC watchdog has three function blocks, i.e. detection, mitigation and restoration.
- PFC deadlock detection - to detect whether the PFC deadlock happened by monitoring the queue PAUSE state. If the queue is in a paused state exceed the detection time, the deadlock is happening.
- PFC deadlock mitigation - once the deadlock is detected, take the one of the actions per queue (priority) level: ◦ forward - temporatily disables PFC to ignore the PFC PAUSE frames.drop - drop the all packets (in / out direction) of the queue.
- log - record the deadlock event to syslog.

- PFC deadlock restoration - monitor the PFC PAUSE frame on the link while taking the mitigation action. If the period of no receiving any PFC PAUSE frame over a configurable time. Re-start the PFC mechanism on the queue.

The PFCWD can enable/disable per port. It is only available for lossless queue. Default is disable for all port.

To configure the PFC refer to Priority Flow Control (PFC) for more information.

To enable the PFC watchdog feature on a specified port or all ports, use the config pfcwd start command.
To disable the PFC watchdog feature on all ports, use the config pfcwd stop command.

Configuration Example

Step1, Prerequisite, configure the lossless buffer setting

Configure the lossless buffer setting.

Example
admin@sonic:~$ sudo config qos reload

Save the configuration and restart the switch for the change to take effect.

Example
admin@sonic:~$ sudo config save -y admin@sonic:~$ sudo reboot

Step2, Configure the QoS mapping

Broadcom switches only

The below table is the QoS mapping is used for enabling PFC on Broadcom switches.

Dot1p	TC	PG	Queue
0	0	0	0
1	1	1	1
2	2	2	2
3	3	3	3
4	4	4	4
5	5	5	5
6	6	6	6
7	7	7	7

The following commands are used to configure the above QoS mappings. Configure the Dot1p-to-TC mapping as:

Example, Broadcom	switches only	
admin@sonic:~$	sudo config	qos dot1p-tc add dot1p-tc-prof --dot1p 0 --tc 0
admin@sonic:~$	sudo config	qos dot1p-tc update dot1p-tc-prof --dot1p 1 --tc 1

```
admin@sonic:~$ sudo config qos dot1p-tc update dot1p-tc-prof --dot1p 2 --tc 2
admin@sonic:~$ sudo config qos dot1p-tc update dot1p-tc-prof --dot1p 3 --tc 3
admin@sonic:~$ sudo config qos dot1p-tc update dot1p-tc-prof --dot1p 4 --tc 4
admin@sonic:~$ sudo config qos dot1p-tc update dot1p-tc-prof --dot1p 5 --tc 5
admin@sonic:~$ sudo config qos dot1p-tc update dot1p-tc-prof --dot1p 6 --tc 6
admin@sonic:~$ sudo config qos dot1p-tc update dot1p-tc-prof --dot1p 7 --tc 7
admin@sonic:~$ sudo config interface qos dot1p-tc bind Ethernet64 dot1p-tc-prof
```

Configure the Tc-to-PG mapping as:

Example, Broadcom switches only

```
admin@sonic:~$ sudo config qos tc-pg add tc-pg-prof --tc 0 --pg 0
admin@sonic:~$ sudo config qos tc-pg update tc-pg-prof --tc 1 --pg 1
admin@sonic:~$ sudo config qos tc-pg update tc-pg-prof --tc 2 --pg 2
admin@sonic:~$ sudo config qos tc-pg update tc-pg-prof --tc 3 --pg 3
admin@sonic:~$ sudo config qos tc-pg update tc-pg-prof --tc 4 --pg 4
admin@sonic:~$ sudo config qos tc-pg update tc-pg-prof --tc 5 --pg 5
admin@sonic:~$ sudo config qos tc-pg update tc-pg-prof --tc 6 --pg 6
admin@sonic:~$ sudo config qos tc-pg update tc-pg-prof --tc 7 --pg 7
admin@sonic:~$ sudo config interface qos tc-pg bind all tc-pg-prof
```

Configure the TC-to-Queue mapping as:

Example, Broadcom switches only

```
admin@sonic:~$ sudo config qos tc-queue add tc-queue-prof --tc 0 --queue 0
admin@sonic:~$ sudo config qos tc-queue update tc-queue-prof --tc 1 --queue 1
admin@sonic:~$ sudo config qos tc-queue update tc-queue-prof --tc 2 --queue 2
admin@sonic:~$ sudo config qos tc-queue update tc-queue-prof --tc 3 --queue 3
admin@sonic:~$ sudo config qos tc-queue update tc-queue-prof --tc 4 --queue 4
admin@sonic:~$ sudo config qos tc-queue update tc-queue-prof --tc 5 --queue 5
admin@sonic:~$ sudo config qos tc-queue update tc-queue-prof --tc 6 --queue 6
admin@sonic:~$ sudo config qos tc-queue update tc-queue-prof --tc 7 --queue 7
admin@sonic:~$ sudo config interface qos tc-queue bind all tc-queue-prof
```

Step3, Apply the lossless buffer profile settings to PGs and queues

Apply the lossless buffer profile settings to PGs and queue which are enabled PFC function.

Apply the lossless buffer profile of ingress_lossless_profile to PG as the example:

Example

```
admin@sonic:~$ sudo config interface buffer bind priority-group all 1 ingress_lossless_profile
```

Apply the lossless buffer profile to egress_lossless_profile to queue as the below example:

Example

```
admin@sonic:~$ sudo config interface buffer bind queue all 1 egress_lossless_profile
```

Step4, Enable the priority of PFC

Enable the 802.1p priority of PFC on the interface. For example, enable priority 1 on Ethernet64 as below:

Example

```
admin@sonic:~$ sudo config interface pfc priority Ethernet64 1 on
```

Step5, Start the PFC WD

The below command shows how to enable PFC watchdog. This example uses the action of the forward, detection time of 400 ms,restoration time of 400 ms, and enabled port of Ethernet64.

Always use the forward action as "forward " for Broadcom switches.

Example

```
admin@sonic:~$ config pfcwd start --action forward --restoration-time 400 ports Ethernet64 detection-time 400
admin@sonic:~$ show pfcwd config
PORT    ACTION   DETECTION TIME   RESTORATION TIME
Ethernet64    drop          400             400
admin@sonic:~$
```

For more information, see the (202111) PFC Watchdog section in the CLI reference guide.

Displaying Queue and Priority-Group Counters

SONiC supports commands that display details of port queue counters and buffer watermarks.

Use the show queue counters command to display packet and byte counters for all queues of all ports or one specific-port.

Example

```
admin@sonic:~$ show queue counters Ethernet72
```

Port	TxQ	Counter/pkts	Counter/bytes	Drop/pkts	Drop/bytes
Ethernet72	UC0	0	0	0	0
Ethernet72	UC1	0	0	0	0
Ethernet72	UC2	0	0	0	0
Ethernet72	UC3	0	0	0	0

Ethernet72	UC4	0	0	0	0
Ethernet72	UC5	0	0	0	0
Ethernet72	UC6	0	0	0	0
Ethernet72	UC7	0	0	0	0
Ethernet72	UC8	0	0	0	0
Ethernet72	UC9	0	0	0	0
Ethernet72	MC0	0	0	0	0
Ethernet72	MC1	0	0	0	0
Ethernet72	MC2	0	0	0	0
Ethernet72	MC3	0	0	0	0
Ethernet72	MC4	0	0	0	0
Ethernet72	MC5	0	0	0	0
Ethernet72	MC6	0	0	0	0
Ethernet72	MC7	0	0	0	0
Ethernet72	MC8	0	0	0	0
Ethernet72	MC9	0	0	0	0

NOTE:

Queue counters can be cleared by using the sonic-clear queue counters command.

To display the user watermark for port queues (egress shared pool occupancy per queue), either for unicast queues or multicast queues, use the show queue watermark command.

Example

admin@sonic:~$ show queue watermark unicast

Egress shared pool occupancy per unicast queue:

Port	UC0	UC1	UC2	UC3	UC4	UC5	UC6	UC7	UC8	UC9
Ethernet0	0	0	0	0	0	0	0	0	0	0
Ethernet4	416	0	0	0	0	0	0	0	0	0
Ethernet8	416	0	0	0	0	0	0	0	0	0
Ethernet12	416	0	0	0	0	0	0	0	0	0

To display the user watermark or persistent-watermark for port ingress "headroom" or "shared pool occupancy" per priority-group for all ports, use the show priority-group command. Or, use the show queue persistent-watermark command to display the user persistent- watermark for either unicast or multicast queues (egress shared pool occupancy per queue) for all ports.

Example

```
admin@sonic:~$ show priority-group watermark shared

Ingress shared pool occupancy per PG:

Port       PG0    PG1    PG2    PG3    PG4    PG5    PG6    PG7
---------- -----  -----  -----  -----  -----  -----  -----  -----
Ethernet0    0      0      0      0      0      0      0      0
Ethernet4    0      0      0      0      0      0      0      0
Ethernet8    0      0      0      0      0      0      0      0
Ethernet12   0      0      0      0      0      0      0      0

admin@sonic:~$ show queue persistent-watermark unicast

Egress shared pool occupancy per unicast queue:

Port       UC0    UC1    UC2    UC3    UC4    UC5    UC6    UC7    UC8    UC9
---------- -----  -----  -----  -----  -----  -----  -----  -----  -----  -----
Ethernet0    0      0      0      0      0      0      0      0      0      0
Ethernet4  416      0      0      0      0      0      0      0      0      0
Ethernet8  416      0      0      0      0      0      0      0      0      0
```

NOTE:

Both the "user watermark" and "persistent watermark" can be cleared using the following commands:

- sonic-clear queue persistent-watermark unicast
- sonic-clear queue persistent-watermark multicast
- sonic-clear priority-group persistent-watermark shared
- sonic-clear priority-group persistent-watermark headroom

For more information, see the (202111) QoS section in the CLI reference guide.

Queue Scheduling

Queue scheduling provides preferential treatment of traffic classes mapped to specific egress queues. SONiC supports SP, WRR, and DWRR scheduling disciplines.

SP - Higher priority egress queues get scheduled for transmission over lower priority queues.

WRR - Egress queues receive bandwidth proportional to the configured weight. The scheduling granularity is per packet which causes large and small packets to be treated the same. Flows with large packets have an advantage over flows with smaller packets.

DWRR - Similar to WRR but uses deficit counter scheduling granularity to account for packet size variations and provide a more accurate proportion of bandwidth.

Queue scheduling is only supported on physical port(s) and unicast queue(s).

The queue scheduling can be configured through CLI command. You need to first configure a profile to port queues. Following steps show how to configurte queue scheduling:

Configuring queue scheduling

1. Configure the scheduler profile, using config scheduler add command. For example, create the scheduler type is set to "WRR", the bandwidth is set to 40 percentage.

Example
admin@sonic:~$ sudo config scheduler add sched-prof-wrr40 --sched_type WRR --weight 40

To create a scheduler profile "sched-prof-wrr20", the scheduler type is set to "WRR", the bandwidth is set to 20 percentage.

Example
admin@sonic:~$ sudo config scheduler add sched-prof-wrr20 --sched_type WRR --weight 20

2. Configure the scheduler profile to queue, using config interface scheduler bind queue command. For example, apply the scheduler profile "sched-prof-wrr40" to queue 3 of Ethernet48.

Example
admin@sonic:~$ sudo config interface scheduler bind queue Ethernet48 3 sched-prof-wrr40
admin@sonic:~$ sudo config interface scheduler bind queue Ethernet48 4 sched-prof-wrr20

Apply the scheduler profile "sched-prof-wrr20" to queue4 of Ethernet48.

Example
admin@sonic:~$ sudo config interface scheduler bind queue Ethernet48 4 sched-prof-wrr20

The queue(s) is not specified the scheduler using the default value.

On Broadcom switches, the default value is WRR with weight 1.

3. Verify the configuration, using show interface scheduler command.

Example

```
admin@sonic:~$ show interfaces scheduler
Ethernet48
Type      Queue  Profile          Scheduling Mode      Weight  Shaper Type      Bandwidth
------  -------  ---------------  -----------------  --------  -------------  -----------
Queue         3  sched-prof-wrr40  WRR                    40  N/A                      0
Queue         4  sched-prof-wrr20  WRR                    20  N/A                      0
```

For more information, see the (202111) Scheduling & Shaping (new) section in the CLI reference guide.

Weighted Random Early Detection (WRED)

Weighted Random Early Detection (WRED) is a variation of the Random Early Detection (RED) algorithm that provides early detection of congestion per traffic class. RED is an Active Queue Management (AQM) mechanism that is implemented for congestion avoidance. WRED/RED addresses the issues caused by Tail-Drop, which responds to congestion too late. Tail-Drop detects congestion only when the queue is already full and has no choice but to drop all incoming packets in this situation.

This causes throughput to degrade because too many packets are dropped. Delay also increases due to the increased mean queue length. All incoming packets are discarded because the congestion notification is sent simultaneously, which leads to global synchronization. Global synchronization leads to the issue.

of unfairness of flows. WRED probabilistically drops packets when the average queue length reaches a defined threshold. With TCP, congestion control is used to slow down the transmission, which keeps the switch queue as low as possible. The WRED drop probability is used to determine whether a packet should be dropped or enqueued according to the current average queue length. The parameters that are used to calculate the drop probability are as follows:

- minimum threshold: minimum threshold to start WRED dropping
- maximum threshold: maximum threshold for WRED dropping
- maximum drop probability: maximum dropping probability

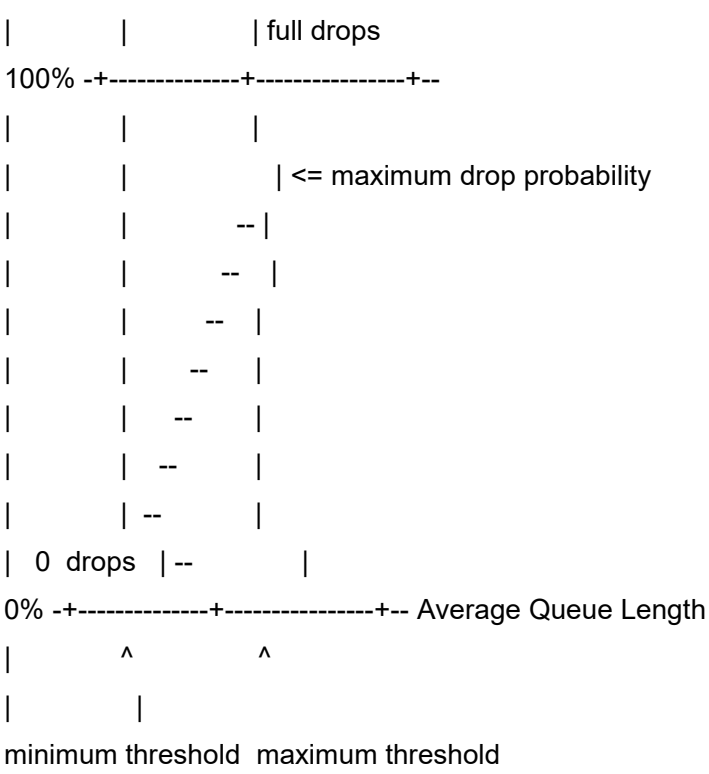
When WRED is enabled, packets are processed according to the following rules:

- When the average queue length is below the minimum threshold , no packet is dropped.
- When the average queue length is between minimum threshold and maximum threshold , incoming packets are dropped according to the drop probability below.

Drop probability = maximum drop probability x (average queue length – minimum threshold) ÷ (maximum threshold – minimum threshold)

- When the average queue length exceeds the maximum threshold , all the incoming packets are dropped.

The below figure visualizes the above behaviors.



The following rules should be applied for setting the WRED parameters:

- The minimum threshold should be set high enough to avoid the WRED mechanism being triggered too early. With network traffic
- mainly being bursty in nature, the minimum threshold should be set as a corresponding high level to make the link utilization at an acceptable level.
- The size between the minimum and maximum threshold should be set large enough to make the connections have enough time to slow down their transmission. If the difference is too small, the behavior is similar to Tail-Drop, which can cause global synchronization.

The WRED can be configured through CLI command. You need to first configure a WRED profile, using config wred add command, and then bind the profile to the port interface's queues, using config interface wred bind queue command.

Configuring WRED

1. Configure the WRED packet drop profile wred-prof. This example uses a minimum threshold of 100k bytes, a maximum threshold of 800k bytes, a maximum drop probability of 100%.

```
admin@sonic:~$ sudo config wred add wred-prof --mode wred --gmin 100000 --gmax 800000 --gdrop 100
```

2. Apply the drop profile wred-prof to queue 0 on Ethernet176.

```
admin@sonic:~$ sudo config interface wred bind queue Ethernet176 0 wred-prof
```

3. Verify the configuration using show interface wred command.

```
admin@sonic:~$ show interface wred
Ethernet176
Queue: 0 ------  ---------------  ---------------  ------------------
ECN/WRED: wred-prof
Color   Mode     Min Threshold   Max Threshold   Drop Probability
Green   WRED          100000         800000              100
Yellow
Red
```

For more information, see the (202111) WRED & ECN (updated, add first two sections) section in the CLI reference guide.

RDMA over Converged Ethernet (RoCE)

Remote Direct Memory Access (RDMA) is a direct memory access technology provided by InfiniBand networks. It enables access to the memory of remote computers like locally DMA. This happens without the involvement of the processor and operating system for all RDMA-enabled computers. Without copying the data between the application, operating system, and NIC, the data transfer rate and latency are significantly improved. The usage of the CPU is nearly zero consumption. RDMA over Converged Ethernet (RoCE) is a network protocol that enables RDMA over an Ethernet network. RoCEv2 (RoCE version 2) enhances RoCEv1 with UDP/IP and using the well-known UDP port 4791. To construct a RoCE network, the congestion control will be the critical challenge that needs to be dealt with. It requires the key features to be provided in the switch.

- Priority Flow Control (PFC) provides hop-by-hop layer 2 congestion control. PFC extends the IEEE 802.3x Ethernet PAUSE, allowing to pause the traffic based on its priority (802.1p) rather than pausing all the traffic on a link.
- Explicit congestion notification (ECN) provides end-to-end layer 3 congestion control. When the congestion is detected by the switch, the ECN field is set in the IP header of a packet before forwarding it. If the receiver device receives the packets

with ECN marked, it sends the congestion notification packets (CNP) to the sender device to reduce the transmit packet rate. The CNP packets continue to be sent to the sender until the congestion is released.

- Enhanced Transmission Selection (ETS) provides minimum guaranteed bandwidth per class basis.

If only PFC is enabled on an RoCE network, it can guarantee no packet loss under congestion. PFC is implemented by NIC resulting in a microsecond-level response time. However, PFC mechanism has numerous well-known problems which reduces the throughput of the uncongested (victim) flows, e.g., Head-of-Line Blocking (HoLB), congestion spreading and deadlock. Using ECN, which provides congestion notification to the source of congestion, can decrease the number of PFC triggering. RoCEv2 Congestion Management (RCM) defines the congestion flows based on the ECN field mark.

To configure PFC you can refer to this document Priority Flow Control (PFC) for more information.

To configure PFC Watchdog you can refer to this document Starting the PFC Watchdog for more information.

To configure ECN you can refer to this document Explicit Congestion Notification (ECN) for more information.

To configure ETS you can refer to this docment Queue Scheduling for more information.

Below provides an example of how to configure RoCE over a DSCP-based lossless network.

There have multiple traffic types on network:

- Lossy traffic for LAN traffic.
- Loss-less traffic for RDMA traffic.

The below table is used for the following example:

Traffic Flow	Loss-less	DSCP	PG	Queue	Bandwidth(%)
LAN flow	No	0, 63	2	2	50%
RoCE, data	Yes	26	3	3	50%
RoCE, CNP	Yes	48	4	4	strict

Configuring Buffer

1. Change the buffer configuration to lossless.

Example
admin@sonic:~$ sudo config qos reload

2. Save the configuration and restart the switch for the change to take effect.

Example
admin@sonic:~$ sudo config save -y admin@sonic:~$ sudo reboot

Configuring ETS: Map DSCP to TCs and allocate bandwitdh

1. Create a DSCP-to-TC profile and add the following mapping.

DSCP	TC
0, 63	0
26	3
48	4

Example

```
admin@sonic:~$ sudo config qos dscp-tc add dscp-tc-prof --dot1p 0,63 --tc 0
admin@sonic:~$ sudo config qos dscp-tc update dscp-tc-prof --dot1p 26 --tc 3
admin@sonic:~$ sudo config qos dscp-tc update dscp-tc-prof --dot1p 48 --tc 4
```

2. Configure the DSCP-to-TC profile on the specified port(s).

Example

```
admin@sonic:~$ sudo config interface qos dot1p-tc bind Ethernet0 dot1p-tc-prof
```

3. Create a TC-Queue profile and add the following mapping.

TC	Queue
0	0
3	3
4	4

Example

```
admin@sonic:~$ sudo config qos tc-queue add tc-q-prof --tc 0 --queue 0
admin@sonic:~$ sudo config qos tc-queue update tc-q-prof --tc 3 --queue 3
admin@sonic:~$ sudo config qos tc-queue update tc-q-prof --tc 4 --queue 4
```

4. Configure the TC-to-Queue profile on the specified port(s).

Example

```
admin@sonic:~$ sudo config interface qos tc-queue bind Ethernet0 tc-q-prof
```

5. Create the following scheduler profiles.

Scheduler Profile	Bandwidth
sched-wrr-50	WRR, 50%
sched-strict	strict

6. Configure the scheduler profiles on the specified queues.

Configure the scheduler profile sched-wrr-50 to queue 0 and 3, sched-strict to queue 4.

Example

```
admin@sonic:~$ sudo config interface scheduler bind queue Ethernet0 0 sched-wrr-50
admin@sonic:~$ sudo config interface scheduler bind queue Ethernet0 3 sched-wrr-50
admin@sonic:~$ sudo config interface scheduler bind queue Ethernet0 4 sched-strict
```

Configuring PFC: Enable PFC for lossless traffic

1. Configure the ingress PG to apply lossless buffer settings.

Example

```
admin@sonic:~$ sudo config interface buffer bind priority-group Ethernet0 3 ingress_lossless_profile
admin@sonic:~$ sudo config interface buffer bind priority-group Ethernet0 4 ingress_lossless_profile
```

2. Configure the egress queue to apply lossless buffer settings.

Example

```
admin@sonic:~$ sudo config interface buffer bind queue Ethernet0 3 egress_lossless_profile
admin@sonic:~$ sudo config interface buffer bind queue Ethernet0 4 egress_lossless_profile
```

3. Enable PFC on the specfied port(s).

Example

```
admin@sonic:~$ sudo config interface pfc priority Ethernet0 3 on
admin@sonic:~$ sudo config interface pfc priority Ethernet0 4 on
```

Configuring ECN: Enable ECN for congestion notification

1. Create a WRED profile.

Example
admin@sonic:~$ sudo config wred add wred-prof --mode ecn --gmin 100000 --gmax 800000 --gdrop 100

2. Configure the WRED profile on the specified queue(s).

Example
admin@sonic:~$ sudo config interface wred bind queue Ethernet0 0 wred-prof admin@sonic:~$ sudo config interface wred bind queue Ethernet0 3 wred-prof

Port and Queue Shaping

Port shaping allows you to control the traffic bandwidth of a port so that it passes less than the full line rate. This feature is important for preventing congestion on certain network links that might drop packets if the maximum bandwidth is exceeded.

Similarly, queue shaping limits the rate at which specific port queues transmit packets. Queue shaping can be useful for preventing high-priority traffic from blocking lower-priority queues.

SONiC supports egress port and queue shaping by two types of metering; either by kilo-bits per second (kbps) or packets per second (pps). Also, you can specify a maximum bandwidth.

Note:

Egress port and queeu shaping is only supported on physical port(s) and unicast queue(s).

The port and queue shaping can be configured through CLI command. You need to first configure a "scheduler profile, and then bind the profile to port interfaces or port queues. Following steps show how to configurte port/queue shaping:

Configuring port shaping

1. Configure the scheduler profile "profile-1", using config scheduler add command. The meter type "bytes" is defined, the maximum bandwidth rate is set to 10 Gbps.

Example
admin@sonic:~$ sudo config scheduler add profile-1 --shaper_type=bytes --bandwidth=10g

2. Apply the scheduler profile "profile-1" to Ethernet0, using config interface scheduler bind port command.

Example

```
admin@sonic:~$ sudo config interface scheduler bind port Ethernet0 profile-1
```

3. Verify the configuration using show interface scheduler command.

Example

```
admin@sonic:~$ show interface scheduler

Ethernet0

Name    Scheduling Type    Weight    Shaper Type   Bandwidth

------  -----------------  --------  -----------  ----------

prof_1  N/A                N/A       bytes         10 Gbps
```

Configuring queue shaping

1. Configure the scheduler profile "profile-1", using config scheduler add command. The meter type "bytes" is defined, the maximum bandwidth rate is set to 10 Gbps.

Example

```
admin@sonic:~$ sudo config scheduler add profile-1 --shaper_type=bytes --bandwidth=10g
```

2. Apply the scheduler profile "profile-1" to queue 0 on Ethernet0, using config interface scheduler bind queue command.

Example

```
admin@sonic:~$ sudo config interface scheduler bind queue Ethernet0 0 profile-1
```

3. Verify the configuration using show interface scheduler command.

Example

```
admin@sonic:~$ show interface scheduler

Ethernet0

Type     Queue  Profile    Scheduling Mode    Weight    Shaper Type    Bandwidth

------  -------  ---------  -----------------  --------  ------------  ----------

Queue        0  profile-1  N/A                N/A       bytes          10 Gbps
```

For more information, see the (202111) Scheduling & Shaping (new) section in the CLI reference guide.

Port Rate Limit (Traffic Policing)

Port rate-limit, allows you to control the traffic bandwidth of a port so that it passes less than the full line rate. This feature is important for preventing congestion on certain network links that might drop packets if the maximum bandwidth is exceeded.

SONiC supports ingress rate limit by two types of metering: kilo-bits per second (kbps) or packets per second (pps). You can also specify a maximum bandwidth and a maximum burst size.

Port Rate Limit is only supported on Broadcom switches.

Port Rate Limit uses the token bucket algorithm to manage the maximum rate of traffic that can pass through on a port.

Port Rate Limit only supports the ingress direction of a physical port.

Port Rate Limit cannot work with ACL at the same time.

The port rate-limit can be configured through CLI command. To enable the port rate-limit on a specified port, use the config interface rate-limit add command. To disable the port rate-limit on a specified port, use the config interface rate-limit del command.

The following example enables port rate limiting on port Ethernet0. The meter type is "bytes", the rate is set to 10 Gbps, and the burst size is set to 1024 bytes.

Example

```
sudo config interface rate-limit add Ethernet0 --meter-type bytes --rate 10g --burst-size 1k
```

The following example enables port rate limit on port Ethernet0. The meter type is "packets", the rate is set to 100 pps.

Example

```
sudo config interface rate-limit add Ethernet0 --meter-type packets --rate 100
```

The following example shows how to verify the configuration by using the show interfaces rate-limit command.

Example

```
admin@sonic:~$ show interface rate-limit
-----------  ------------  ---------  -----------
Interface    Meter Type        Rate   Burst Size
Ethernet0    bytes          10 Gbps        1 Kib
```

The following example is to disable port rate limit on port Ethernet0.

Example
sudo config interface rate-limit del Ethernet0

For more information, see the (202111) Port Rate Limit section in the CLI reference guide.

Shared Buffer Configuration

Shared Buffer Configuration allows users to configure the shared buffer for packet buffering within the device. All received packets are stored in the internal buffer before being transmitted. The internal buffer is also called a "shared buffer" because it is shared by all ingress and egress ports. To avoid starvation and unfairness, the buffer configuration allows users to define the buffer pool, including the maximum amount of shared buffer that can be used. A buffer profile template can be further defined to refer to the buffer pool and applied to ingress priority-groups (PGs) and egress queues on ports.

The default buffer configuration is properly measured for system operation. For any change or modification, users should be aware of the effect on the system and have the confidence to manage it. If the system is in an unexpected state and is out of service, the user may restore the factory default configuration (refer to "sudo config-setup factory" in New Default Configuration)

To configure the buffer pool, use the config buffer pool command. To configure the buffer profile, use the config buffer profile command. To apply the buffer profile to a PG or queue, use the config interface buffer bind priority-group command and config interface buffer bind queue command respectively.

Shared Buffer Configuration Guide

1. Configuring the buffer pool.

The amount of buffer memory available varies by switch platform. The following table shows the buffer amounts per switch platform.

Table 1: Switch buffer size.

Switch	Total ingress buffer size (bytes)	Total egress buffer size (bytes)
N9200-64DC	56,510,148	56,510,148

The buffer can be divided into individual buffer pools . In a buffer pool, you can specify the total shared buffer size for all PGs that use this pool and specify the headroom size if the pool is used for lossless PGs. However, the reserved size (or minimum guaranteed buffer) of a PG or queue is not included in the shared buffer size of the pool. The following equations show the correct configuration for buffer pools and reserved size . The total

configuration sizes must not exceed the total amount per ingress and egress stage. Once the sum of the configuration sizes exceeds the total amount, th different pools will compete for the buffer size and cause packet drops, even if it doesn't reach the pool limit.

Total ingress buffer size (shared buffer size + headroom size for all ingress buffer pools) + (reserved size for all PGs)

Total egress buffer size (shared buffer size for all egress buffer pools) + (reserved size for all queues)

For Broadcom Tomahawk4 switches, the ingress and egress pools mapped by TC2PG and TC2Queue should have the same share size for a given packet .

During the process of changing the buffer settings and TC2PG and TC2Queue mappings, all packets will be dropped. Packet forwarding

will resume once all the changes have been configured into the hardware.

Headroom Size Calculation

Headroom buffer is used to absorb inflight packets while the upstream is responding to the PFC PAUSE . The headroom size requirement depends on the processing and queuing delays of the downstream switch, the transmission time based on the port speed and the cable length, and the response time of the upstream switch . In general, the size can be calculated approximately as below .

headroom size of a PG = 2 × (latency + media delay + cable length × 5) × pps × packets to cells × cell size ÷ 109 packets to cells conversion = ceiling[(packet size + M) ÷ cell size], M is 64 for Broadcom switches

where:

- latency is switch latency for queuing or responding in nanoseconds, using respectively 1,000 (1μs) for Broadcom switches.
- Media delay is PHY latency in nanoseconds .
- Cable length is the cable length in meters .
- 5 is the link propagation delay in nanoseconds calculated by the propagation speed of 200,000,000 meters/sec . . pps is packet per second at the current port speed .
- Cell size, which varies across switch platforms, is shown in Table 2.

For example, to calculate the headroom size with the media delay of 100 ns, cable length of 300 meters, pps of 1,369,713 (using the size of 9,126 bytes and the port speed of 100 Gb/s), and cell size of 256 bytes on Broadcom Trident 3 switch, we get: 65 KB 2 × (1,000 + 100 + 300 × 5) × 1,369,713 × 36 × 256 ÷ 109

The size of headroom pool is the sum of the size of the headroom requirement for all the lossless PGs. E.g., the size for 32 ports with 100Gb/s and 2 priorities enabled per port is 65KB × 32 × 2 = 4,160 KB. All ports and all priorities can be paused simultaneously .

Table 2: Cell size .

Switch	bytes per cell
N9200-64DC	254

The following example configures the ingress buffer pool "ingress_lossless_pool". This example uses a shared buffer size of 2,940,941 bytes and a headroom size of 14,704,704 bytes.

Example

```
admin@sonic:~$ sudo config buffer pool add ingress_lossless_pool --type ingress --size 2940941 --xoff-
size
14704704
```

2. Configuring the buffer profile.

For each buffer profile, there are:

- pool name: The name of the pool which contains the maximum shared size and the maximum headroom size that can be used in this profile.
- reserved size: The minimum guaranteed size for the priority-group or queue on a port.
- shared size: The maximum shared size for the priority-group or queue on a port. When the reserved size is insufficient, the shared size will be used. It's possible to specify an absolute value or a percentage of the buffer pool size.
- (optional) headroom size (xoff): The headroom size for absorbing inflight packets to avoid packet loss for PFC.
- (optional) xon and xon-offset: Used to determine the XON limit, below which the XOFF state will be released when the buffer limit at the ingress port falls. The XON limit is the larger value of (xon) and (the overall PG size - xon-offset).

The following example configures a buffer profile "ingress_lossless_profile". The example uses the buffer pool "ingress_lossless_pool", reserved size of 0 bytes, shared buffer percentage of 66%, headroom size of 14,704,704 bytes, xon of 1,746,919 bytes, and xon-offset of 18,432 bytes.

Example

```
admin@sonic:~$ sudo config buffer profile add ingress_lossless_profile --pool ingress_lossless_pool --
size 0

--shared-percent 66 --xoff 14704704 --xon 1746919 --xon-offset 18432
```

3. Configuring the buffer profile to PG or queue.

For example, to configure the buffer profile "ingress_lossless_profile" to PG3 of Ethernet60:

Example

```
admin@sonic:~$ sudo config interface buffer bind priority-group Ethernet60 3 ingress_lossless_profile
```

For example, to configure the buffer profile "egress_lossless_profile" to queue3 of Ethernet60:

Example
admin@sonic:~$ sudo config interface buffer bind queue Ethernet60 3 egress_lossless_profile

For more information, see the (202111) Shared Buffer (Modified) section in the CLI reference guide.

DiffServ

Differentiated Services (DiffServ) is a traffic management architecture defined in RFC 2475. It is used to prioritize network traffic based on different classes of service. In DiffServ, the Differentiated Services Code Point (DSCP) is a 6-bit field in the IP header that classifies and prioritizes traffic.

Per-Hop Behaviors (PHBs) are defined in network devices to determine how traffic with different DSCP values should be treated. These behaviors guide the handling of packets based on their assigned DSCP values.

DiffServ manages various services such as elastic, continuous media, and real-time applications by applying different policies to control their behavior during congestion. These policies are constructed using metering, marking, scheduling, and shaping functions, which help manage the behavior of received packets as they traverse downstream switches.

Compared to the Quality of Service (QoS) feature, which classifies packets based on the Class of Service (CoS) or DSCP field in the VLAN or IP header, DiffServ provides a more granular classification of network traffic. It achieves this by utilizing Access Control Lists (ACLs) as the classifier, allowing consideration of additional packet fields such as source/destination IP addresses and TCP/UDP ports. DiffServ also enables traffic remarking or changing the egress queue based on a policing algorithm, providing better control over traffic based on the input bandwidth requirements of different services.

The following table illustrates the conversion between DSCP and EF/AF/CS PHB:

DSCP value	PHB	Description
46	ef	Expedited Forwarding PHB
10	af11	Assured Forwarding PHB
12	af12	
14	af13	
18	af21	
20	af22	
22	af23	
26	af31	
28	af32	
30	af33	
34	af41	

36	af42	
38	af43	
8	cs1	Class selector PHB
16	cs2	
24	cs3	
32	cs4	
40	cs5	
48	cs6	
56	cs7	
0	be	Best effort PHB

Diffserv is only supported on Broadcom switches.

The following shows the general steps for configuring DiffServ:

Step1 , Create an ACL table type.

The following shows the fields supported:

- VLAN ID
- CoS (PCP value)
- Source IPv4/IPv6 address
- Destination IPv4/IPv6 address
- DSCP
- L4 source port (TCP/UDP source port)
- L4 destination port (TCP/UDP destination port)

If VLAN ID and CoS are specified to match in the ACL table type, the tagged VLAN of the incoming packet is used for the match. If the incoming packet is un tagged, the VLAN ID is added using the native VLAN of the port and CoS is set to 0.

The ACL table type is only supported to match IP packets (IPv4 or IPv6).

The N9200-64DC switches do not support matching VLAN ID and PCP for the egress stage.

The following shows the actions supported:

- Change CoS
- Change DSCP
- Change TC (Traffic Class) (available at the ingress stage only)

· Apply policer, providing:

Storm-control (Single Rate Two Color)

SrTCM (Single Rate Three Color)

TrTCM (Two Rate Three Color)

Step2 , Create an ACL table. Specify the defined type in Step 1 and the stage and bind point where you want to match the packets.

The PortChannel port is not supported as a bind point at the egress stage.

The N9200-64DC switch supports only a single ACL table that can apply a policer in hardware for the ingress stage. In addition, ingress port rate limiting is also implemented using TCAM and a policer. Therefore, if a port has ingress port rate limiting enabled, it cannot have an ACL table with an ingress stage policer. Similarly, for other switches, ingress port rate limiting cannot be used with an ACL table that has DiffServ actions applied on the same port.

The N9200-64DC switch does not support changing CoS and applying a policer for the egress stage.

Step3 , Configure a policer (meter), if you want to apply the meter function.

Step4 , Add the ACL rule. Specify the match fields and actions. The actions include:

- Change CoS
- Change DSCP
- Change TC (Traffic Class)
- Apply policer

For the change TC action, it only available on ingress stage. If specify the change TC action, you also need to configure the TC2Queue mapping, refer to Class of Service (CoS) for more information.

To configure an ACL table, using config acl add table-type and config acl add table command.

To configure the ACL rules, using config acl add rule command.

To configure a policer (meter), using config policer command.

Example: Configuring DSCP Marking

This example shows how to change the DSCP value using an ACL rule.

1. Create an ACL table type diffserv-type with the capability to match the DSCP field.

Example
admin@sonic:~$ sudo config acl add table-type diffserv-type --match-dscp

2. Create an ACL table diffserv-table using the previously defined type. Specify the ingress stage and Ethernet0 port.

Example

```
admin@sonic:~$ sudo config acl add table diffserv-table diffserv-type -s ingress -p Ethernet0
```

3. Add an ACL rule to the ACL table diffserv-table table. Set the packet action to permit, rule priority to 10,000, match DSCP value of 0, and change the DSCP value to 63.

Example

```
admin@sonic:~$ sudo config acl add rule diffserv-table permit --priority 10000 --dscp 0 --set-dscp 63
```

When a packet matches an ingress ACL rule, the original CoS/DSCP value is used to map to the Dot1pToTC or DSCPToTC QoS maps on the ingress port for TC assignment.

The new CoS/DSCP value specified in the ACL rule will override the value of the TC2Dot1p/TC2DSCP remarking QoS maps on the egress port.

Example: Configuring CoS Marking

This example shows how to change the CoS value using an ACL rule.

1. Create an ACL table type diffserv-type with the capability to match the VLAN ID and CoS field.

Example

```
admin@sonic:~$ sudo config acl add table-type diffserv-type --match-vlan-id --match-cos
```

2. Create an ACL table diffserv-table using the previously defined type. Specify the ingress stage and Ethernet0 port.

Example

```
admin@sonic:~$ sudo config acl add table diffserv-table diffserv-type -s ingress -p Ethernet0
```

3. Add an ACL rule to the ACL table diffserv-table table. Set the packet action to permit, rule priority to 10,000, match VLAN ID and CoS of 10 and 1, and change the CoS value to 6.

Example

```
admin@sonic:~$ sudo config acl add rule diffserv-table permit --priority 10000 --vlan-id 10 --cos 1 --set-cos 6
```

Example: Configuring TC Assignment

This example shows how to assign a traffic class (TC) to change the egress queue. A shaper is also applied to the queue to limit the maximum bandwidth it can use.

1. Create an ACL table type diffserv-type with the capability to match the DSCP field.

Example

```
admin@sonic:~$ sudo config acl add table-type diffserv-type --match-dscp
```

2. Create an ACL table diffserv-table using the previously defined type. Specify the ingress stage and Ethernet0 port.

Example

```
admin@sonic:~$ sudo config acl add table diffserv-table diffserv-type -s ingress -p Ethernet0
```

3. Add an ACL rule to the ACL table diffserv-table table. Set the packet action to permit, rule priority to 10,000, match DSCP value of 0, and change the Traffic Class to 7.

Example

```
admin@sonic:~$ sudo config acl add rule diffserv-table permit --priority 10000 --dscp 0 --set-tc 7
```

4. Create an scheduler shaper-prof with a bandwidth of 10 Gbps.

Example

```
admin@sonic:~$ sudo config scheduler add shaper-prof --shaper_type bytes --bandwidth 10g
```

5. Bind the shaper-prof to egress port Ethernet52 and queue 7.

Example

```
admin@sonic:~$ sudo config interface scheduler bind queue Ethernet52 7 shaper-prof
```

Example: Configuring Traffic Policing

This example shows how to use a policer to change the DSCP based on a policing algorithm.

1. Create an ACL table type diffserv-type with the capability to match the DSCP field and attach the policer action.

Example

```
admin@sonic:~$ sudo config acl add table-type diffserv-type --match-dscp --action-set-policer
```

2. Create an ACL table diffserv-table using the previously defined type. Specify the ingress stage and Ethernet0 port.

Example

```
admin@sonic:~$ sudo config acl add table diffserv-table diffserv-type -s ingress -p Ethernet0
```

3. Create a policer pol with the following actions: changing the DSCP to 10 for green packets, changing the DSCP to 18 for yellow packets, and changing the DSCP to 26 for red packets.

Example

```
admin@sonic:~$ sudo config policer add pol tr_tcm packets blind --cir 100 --cbs 200 --pir 300 --pbs 400 --
```

```
green_set_dscp 10 --yellow_set_dscp 18 --red_set_dscp 26
```

4. Add an ACL rule to the ACL table diffserv-table , specifying the packet action of permit, rule priority to 10,000, match DSCP 0, and attaches the policer pol .

Example

```
admin@sonic:~$ sudo config acl add rule diffserv-table permit --priority 10000 --dscp 0 --policer pol
```

The policer action takes precedence over setting the COS and DSCP if an ACL rule includes both actions.

For more information, see the (202111) ACL and (202111) Policer sections in the CLI reference guide.

Security

AAA and TACACS+

The Authentication, Authorization, and Accounting (AAA) feature provides the main framework for configuring access control on the switch.

- The three security functions can be summarized as follows:
- Authentication — Identifies users that request access to the network.
- Authorization — Determines if users can access specific services.
- Accounting — Provides reports, auditing, and billing for services that users have accessed on the network.

The AAA functions require the use of configured Terminal Access Controller Access Control System Plus (TACACS+) servers in the network. The security servers can be defined as sequential groups that are applied as a method for controlling user access to specified services. For example, when the switch attempts to authenticate a user, a request is sent to the first server in the defined group, if there is no response the second server will be tried, and so on. If a pass or fail is returned at any point, the process stops.

Note:

It is essential to keep in mind that when you modify the AAA configuration directly in the "/etc/sonic/config_db.json" file and use the "config reload" or "reboot" command to load the new configuration from a file, the new AAA configuration will not take effect until the "hostcfgd.service" is ready. The PAM files will retain the prior setting until the service is ready. Depending on the different switches, the wait time for the service to be ready may vary and it typically takes around 3 to 5 minutes.

Use the aaa authentication login command to configure whether AAA should use the local database or a remote TACACS+ server database for user authentication. By default, AAA uses a local database for authentication. Local authentication restricts management access based on user names and passwords manually configured on the switch. New users can be added or deleted using Linux commands (note that configuration using Linux commands is not preserved during reboot).

You can enable remote TACACS+ server authentication by selecting "tacacs+" in the aaa authentication login command. Administrators need to configure the TACACS+ server accordingly and ensure that connectivity to the TACACS+ server is available. Once TACACS+ server remote authentication is selected, all user logins will be authenticated by the TACACS+ server. If the authentication fails, AAA will check the "failthrough" configuration and, if enabled, authenticate the user based on the local database.

Example

```
admin@sonic:~$ sudo config aaa authentication login tacacs+
```

Use the aaa authentication failthrough command to enable the failthrough option.

When failthrough is enabled and multiple TACACS+ servers have been configured (up to a maximum of seven), the authentication process will continue through all configured servers, even when an authentication request to the first server fails. If all remote authentication fails, the process will then proceed to local authentication.

When failthrough is disabled and an authentication request fails on the first server, the authentication process will stop and the login will be disallowed.

Example

```
admin@sonic:~$ sudo config aaa authentication failthrough enable
admin@sonic:~$ show aaa
AAA authentication login local
AAA authentication failthrough False
```

Authentication and User Level

SONiC itself has no concept of user levels, so an additional command is needed to set levels for existing users.

Add the userpri command to set the user level except root and admin.

Usage: userpri <username> <privilege>

userpri show

If the username does not exist, an error is reported, and <username> <privilege> is saved to /etc/user_privilege.

userpri show shows all users except root and admin.

Different user levels correspond to different access rights, that is, locally authorized. The plan is as follows:

Estate	Type	Group	Authority
15	admin	sudo/ docker	Access to all commands
2-14	user	users	You cannot sudo su, nor can you access other configuration commands (such as redis-cli, sonic-cli, vtysh, docker, etc.), other commands are accessible. For example, common linux commands, show, etc

1	Guest	users	Only the show command can be accessed

When a user is created locally, the user level is 7 by default. You can run the userpri show command to view the user level

example:

```
root@sonic:/home/admin# useradd local
root@sonic:/home/admin# passwd local
New password:
Retype new password:
passwd: password updated successfully
root@sonic:/home/admin# userpri show
test 7
local 7
root@sonic:/home/admin#
```

The user level can be modified with userpri <username> <privilege>

example:

```
root@sonic:/home/admin# userpri local 1
root@sonic:/home/admin# userpri show
test 7
local 1
root@sonic:/home/admin#
```

When tacacs authentication is used, as described in (202111.8) AAA and TACACS+, successful authentication brings the level on the tacacs server to the DUT, for example:

The group level is 8. The user name and password are w8/w8

The Tacacs server configuration is as follows:

```
group = p8 {
    default service = permit
    enable = permit
    service = shell {
        default command = permit
        default attribute = permit
        set priv-lvl = 8
    }
}
user = w8 {
    password = clear w8
    member = p8
}
```

After logging in with w8, view the user level, which is the same as on the tacacs server.

example:

```
root@sonic:/home/admin# userpri show
test 7
local 1
w8 8
root@sonic:/home/admin#
```

For more information on AAA, see the (202111) AAA section in the CLI reference guide.

Authorization & Accounting

The authentication, authorization, and accounting (AAA) feature provides the main framework for configuring access control on the switch.

The three security functions can be summarized as follows:

- Authentication — Identifies users that request access to the network.
- Authorization — Determines if users can access specific services.
- Accounting — Provides reports, auditing, and billing for services that users have accessed on the network.

Note : Authorization and accounting are only supported using the TACACS+ server as a remote server.
Authorization.

When aaa authorization is set to local, authorization is based on local permissions. That is, commands can be accessed based on user levels.

example:

```
root@sonic:/home/admin# config aaa authorization local
root@sonic:/home/admin# show aaa
AAA authentication login local,tacacs+
AAA authentication failthrough True
AAA authorization login local
AAA accounting login disable (default)
```

Test level 1 users are as follows:

```
test@sonic:~$
test@sonic:~$ show aaa
AAA authentication login local,tacacs+
AAA authentication failthrough True
AAA authorization login local
AAA accounting login disable (default)


test@sonic:~$ ls
Access Denied
test@sonic:~$ pwd
Access Denied
test@sonic:~$
```

When aaa authorization is tacacs, authorization is performed entirely by the tacacs server

example:

```
root@sonic:/home/admin# config aaa authorization tacacs+
root@sonic:/home/admin# show aaa
AAA authentication login local,tacacs+
AAA authentication failthrough True
AAA authorization login tacacs+
AAA accounting login disable (default)
```

The Tacacs server configuration is as follows:

```
group = test {
        default service = permit
        enable = permit
        service = shell {
            default command = deny
            default attribute = permit
            set priv-lvl = 1
            cmd = /usr/local/bin/show
            {
                deny aaa
                permit .*
            }
        }
}
user = test {
    password = clear test
    member = test
}
```

Log in with this user and test as follows:

```
test@sonic:~$ show aaa
deny by tacacs
test@sonic:~$ show tacacs
TACPLUS global auth_type pap (default)
TACPLUS global timeout 5 (default)
TACPLUS global passkey <EMPTY_STRING> (default)

TACPLUS_SERVER address 172.21.170.242
        passkey ******
        priority 1
        tcp_port 49

test@sonic:~$ ls
deny by tacacs
test@sonic:~$ pwd
/home/test
test@sonic:~$
```

When aaa authorization is tacacs+ local, tacacs authorization is performed first. local authorization is performed only when the tacacs server does not respond.

example:

```
root@sonic:/home/admin# config aaa authorization tacacs+ local
root@sonic:/home/admin# show aaa
AAA authentication login local,tacacs+
AAA authentication failthrough True
AAA authorization login tacacs+,local
AAA accounting login disable (default)
```

Again using the test user example above, if tacacs server can access it, the test results should be exactly the same as if only tacacs authorization were configured.

If the tacacs server is unreachable, the test result should be consistent with that of configuring only local authorization.

To enable authorization, use the config aaa authorization command and specify the authorization method sequence. To enable accounting, use the config aaa accounting command and specify the accounting methods. To disable the accounting, use the config aaa accounting disable command.

Configuring authorization & accounting

1. The following example shows to enable TACACS+ authorization and local accounting.

Example

```
admin@sonic:~$ sudo config tacacs add 192.168.8.122 -k tacacs -o 4999 -a login
admin@sonic:~$ sudo config aaa authentication login tacacs+ local
admin@sonic:~$ sudo config aaa authorization tacacs+
admin@sonic:~$ sudo config aaa accounting local
admin@sonic:~$ show aaa
AAA authentication login tacacs+
AAA authentication failthrough False (default)
AAA authorization login tacacs+
AAA accounting login local
```

2. The following example shows the configuration file of Tacacs+ (tac_plus) server.

Example: tac_plus.cfg

```
key = tacacs
accounting file = /home/admin/tac_plus.acct
group = netadmin {
default service = permit
service = exec {
priv-lvl = 15
}
```

```
}
group = netuser {
default service = permit
service = exec {
priv-lvl = 1
}
}
user = tadmin {
login = cleartext tadmin
member = netadmin
}
user = tuser {
login = cleartext tuser
member = netuser
cmd = /usr/bin/cat {
deny .*
}
cmd = /usr/bin/find {
deny   -exec
permit .*
}
}
```

Start the tac_plus server.

Example

```
admin@spine-2:~$ tac_plus/tacacs-F4.0.4.28/tac_plus -C tac_plus.cfg -G -g -p 4999
```

For more information on AAA, see the (202111) AAA section in the CLI reference guide.

Configuring TACACS+ Server Parameters

When a remote authentication server is used, you must specify the message exchange parameters for the TACACS+ protocol. With multiple servers specified (maximum of seven), some parameters are global and can be applied to all servers, or the parameters can be configured separately for each server.

Use the config tacacs add command to specify a TACACS+ server. Specify the server IP address and optionally other parameters if they are not set at the global level. The authtype, passkey and timeout parameters must be defined, either using the config tacacs add command or by using the global setting commands config tacacs au thtype , config tacacs passkey , and config tacacs timeout .

Example
admin@sonic:~$ sudo config tacacs add 10.11.12.13 -t 10 -k testing789 -a pap -o 50 -p 9

To remove a TACACS+ server, use the config tacacs delete command and just specify the IP address.

Example
admin@sonic:~$ sudo config tacacs delete 10.11.12.13

For more information on configuring TACACS+ servers, see the (202111) TACACS+ section in the CLI reference guide.

Configuring RADIUS Server Parameters

When a remote authentication server is used, you must specify the message exchange parameters for the RADIUS protocol. With multiple servers specified (maximum of 8), some parameters are global and can be applied to all servers, or the parameters can be configured separately for each server.

Use the config radius add command to specify a RADIUS server. Specify the server IP address and optionally other parameters if they are not set at the global level. The authtype, passkey and timeout parameters must be defined, either using the config radius add command or by using the global setting commands config radius authtype , config radius passkey , and config radius timeout .

Example
admin@sonic:~$ sudo config radius add 10.11.12.13 -k test1234

To remove a RADIUS server, use the config radius delete command and just specify the IP address.

Example
admin@sonic:~$ sudo config radius delete 10.11.12.13

Example: configuring FreeRADIUS

To configure the RADIUS server edit the RADIUS configuration file (/etc/raddb/clients.conf). Below shows an example to configure the client IP address of 192.168.0/24, secret passkey of "test1234".

Example

```
client 192.168.8.0/24 {
secret = test1234
}
```

To add user for authentication edit the RADIUS user configuration file (/ect/raddb/users). Below shows an exam le to add a user, name of "ruser", password of "mypassword", the privilege of 7.

Example

```
ruser    Cleartext-Password := "mypassword"
Management-Privilege-Level = 7
```

RADIUS support the privilege level of 1 ~ 15. Privilege level 15 is for the administrator user.

The user authenticated by RADIUS will obtain the privilege level without needing to enable authorization by "aaa authorization" command.

Start FreeRADIUS server:

Example

```
admin@spine-2:~/freeradius$ sbin/radiusd -f
```

For more information on configuring RADIUS servers, see the (202111) RADIUS section in the CLI reference guide.

Access Control Lists (ACL)

An Access Control List (ACL) is a sequential list of matching criteria with permit or deny actions that are applied to IPv4/IPv6 addresses or other specific criteria. SONiC tests ingress or egress packets against the criteria on the ACL, one by one. The packet is accepted if it matches a permit rule, or dropped if it matches a deny rule. If no rules match, the packet is matched by the default rule, which is an implicit DENY ALL rule.

ACL creation then Rules provisioning

Create ACL table

ACL is created with a name, type, and interface which binds the ACL to (Refer config acl add table command). Before creating the ACL table, user needs to have (a) matching criteria which is examined, (b) applied interfaces, (c) direction, ingress or egress.

For example, create an ACL table "DATA_L3", tyoe "L3" (for IPv4), and apply to interfaces "Ethernet0, Ethernet4" in "ingress" stage,

the command is

Example

admin@sonic:~$ sudo config acl add table DATA_L3 L3 --description "L3 table" --ports "Ethernet0,Ethernet4" --

stage "ingress"

Apply rules

Before provisioning ACL rules, user needs to configure the matching criteria and action in configuration file (refer ACL Rule Configuration File), then using config acl update full (or config acl update incremental) command to load (or replace) the rules in the configuration file. There is default rule an implicit DENY ALL rule be inserted at the end.

For example, update those rules specified in configuration file, acl_rule_L3_scrip_dstip.json.

Example

admin@sonic:~$ sudo config acl update incremental acl_rule_L3_scrip_dstip.json

Sample of ACL configuration

acl_rule_L3_scrip_dstip.json

```
{
"acl":
{
"acl-sets":
{
"acl-set":
{
```

```
"DATA_L3":
{
"acl-entries":
{
"acl-entry":
{
"RULE_1":
{
"ip":
{
"config":
{
"destination-ip-address": "4.4.4.4/32",
"source-ip-address": "3.3.3.3/32"
}
},
"config":
{
"sequence-id": 1
},
"actions":
{
"config":
{
"forwarding-action": "ACCEPT"
}
}
}
}
```

```
}
}
}
}
}
}
```

The following shows how to configure a "blacklist" as the below example using ACL configuration file.

```
1) drop     sip=3.3.3.3 and dip=4.4.4.4
2) forward  any
3) drop     any                    (default rule, auto inserted)
```

The second rule in the configuration file is used to disable the default rule.

The configuration file of the example shows as below:

acl_rule_blacklist.json

```
sudo config acl add table DATA_L3 L3
cat > acl_rule.json << EEE
{
"acl":
{
"acl-sets":
{
"acl-set":
{
"DATA_L3":
{
"acl-entries":
{
"acl-entry":
{
```

"1":

{

"ip":

{

"config":

{

"destination-ip-address": "4.4.4.4/32",

"source-ip-address": "3.3.3.3/32"

}

},

"config":

{

"sequence-id": 1

},

"actions":

{

"config":

{

"forwarding-action": "DROP"

}

}

}

"2":

{

"config":

{

"sequence-id": 2

},

"actions":

```
{
"config":
{
"forwarding-action": "ACCEPT"
}
}
}
}
}
}
}
}
}
EEE
```

IP tables rules

The cac lmgrd process control the ACL related rule's update after the related database change, it also initialize the default iptables rules when the process start.

It will check interface configured on Ethernet/PortChannel/VLAN/Management interface add block IP2ME rule into iptables by default.

The following example blocks the IP address (DROP all -- 0.0.0.0/0 192.168.0.0) configured on the management interface (eth0).

acl_rule_blacklist.json

```
admin@leaf-1:~$ show ip in
Interface    Master    IPv4 address/mask    Admin/Oper    BGP Neighbor    Neighbor IP
docker0              240.127.1.1/24       up/down       N/A             N/A
eth0                 192.168.8.111/20     up/up         N/A             N/A
lo                   127.0.0.1/16         up/up         N/A             N/A
admin@leaf-1:~$ sudo iptables -n -L
```

```
Chain INPUT (policy ACCEPT)
target     prot opt source               destination
ACCEPT     all  --  127.0.0.1           0.0.0.0/0
ACCEPT     all  --  0.0.0.0/0           0.0.0.0/0           ctstate RELATED,ESTABLISHED
ACCEPT     icmp --  0.0.0.0/0           0.0.0.0/0           icmptype 8
ACCEPT     icmp --  0.0.0.0/0           0.0.0.0/0           icmptype 0
ACCEPT     icmp --  0.0.0.0/0           0.0.0.0/0           icmptype 3
ACCEPT     icmp --  0.0.0.0/0           0.0.0.0/0           icmptype 11
ACCEPT     udp  --  0.0.0.0/0           0.0.0.0/0           udp dpts:67:68
ACCEPT     udp  --  0.0.0.0/0           0.0.0.0/0           udp dpts:546:547
ACCEPT     udp  --  0.0.0.0/0           0.0.0.0/0           udp dpt:4789
ACCEPT     udp  --  0.0.0.0/0           0.0.0.0/0           udp spts:49152:65535 dpt:3784
ACCEPT     udp  --  0.0.0.0/0           0.0.0.0/0           udp dpt:3785
ACCEPT     udp  --  0.0.0.0/0           0.0.0.0/0           udp spts:49152:65535 dpt:4784
ACCEPT     tcp  --  0.0.0.0/0           0.0.0.0/0           tcp dpt:179
ACCEPT     tcp  --  0.0.0.0/0           0.0.0.0/0           tcp spt:179
DROP       all  --  0.0.0.0/0           192.168.0.0
ACCEPT     all  --  0.0.0.0/0           0.0.0.0/0           TTL match TTL < 2
Chain FORWARD (policy ACCEPT)
target     prot opt source               destination
Chain OUTPUT (policy ACCEPT)
target     prot opt source               destination
```

If user configure IP addresses on other interfaces after system startup, cac lmgrd doesn't reflect the changes automatically.

It needs to trigger it to re-initialize the default iptables rules manually (by sudo systemctl restart cac lmgrd) or ACL related configuration change.

Configuring a Control Plane ACL

A Control Plane ACL is an enhancement of ACLs and is applied to packets that originate from CoPP to filter and protect the main CPU.

In SONiC you can configure CoPP to forward and limit the packet flow to the CPU, but you cannot apply ACLs to filter out some traffic as they do in the data plane. For example, to filter out (or grant) SNMP, NTP, SSH access from some hosts (or IP address). In this enhancement, a reserved keyword "CTRLPLANE" is used for the ACL rules that apply to the management flow (or control plane flow). The current supported services are SNMP, NTP, and SSH.

Here is an example of how to apply Control Plane ACLs to permit packets from address (3.3.3.3) to access management services and deny access from other IP addresses.

First, use ACL commands to creates an ACL table, for example "DATA_CTRL", with the keyword CTRLPLANE, and then specify the applied services, e.g. SNMP and SSH.

create control	**plane acl**	**table**
admin@sonic:	~$ sudo	config acl add table DATA_CTRL CTRLPLANE --description "SSH, SNMP" "CTRLPLANE table" --services

Second, create a configuration file (e.g. service_rules.json) and add the rules that you want to apply for the services. The following example needs to specify:

- packets from (source ip address) 3.3.3.3
- action is accept

configuration file (service_rules.json)

```
{
"acl":
{
"acl-sets":
{
"acl-set":
{
"DATA_CTRL":
{
"acl-entries":
```

```
{
"acl-entry":
{
"1":
{
"ip":
{
"config":
{
"source-ip-address": "3.3.3.3/32"
}
},
"config":
{
"sequence-id": 1
},
"actions":
{
"config":
{
"forwarding-action": "ACCEPT"
}
}
}
}
}
}
}
}
```

```
}
}
```

Third, use the config acl update command to add the rules.

add the rules to ACL

```
admin@sonic:~$ sudo config acl update incremental service_rules.json
```

You can use the commands show acl table or show acl rule o examine the configuration.

ACL Rule Configuration File

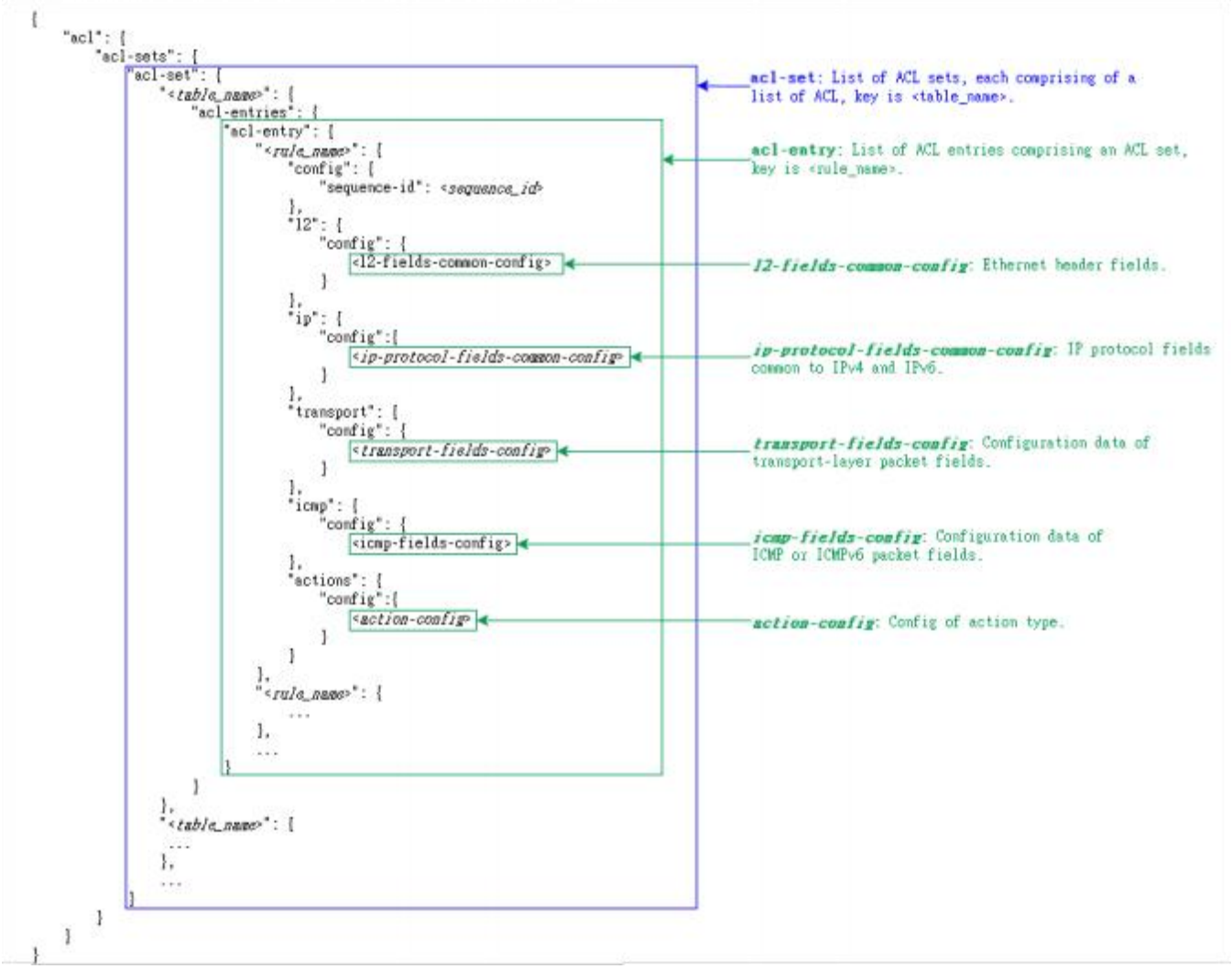
ACL matching rule and action is specified in configuration file and can be applied in data plane (ACL) or control plane ACL,depending the applied field (or type).

In this section, briefs overall structure of configuration, then describes valid options of field, and valid combination of matching criteria and actions.

ACL Rule Configuration File

The ACL rule configuration JSON file uses the data models created by OpenConfig (http://ops.openconfig.net/branches/models/master/docs /openconfig-acl.html) represented as JSON format.

The following shows the file structure for ACL rule configuration.



Parameters:

- acl-set : A collection of ACL table(s), each table is composed of ACL entries.

  - Key is the ACL table name. Each element is an acl-entries object. For example:

{

"DATA_L3": {

"acl-entries": {

"acl-entry": { ... }

}

},

"DATA_L3V6": {

...

}

}

· acl-entries : A collection of access entries.

acl-entry : A collection of rules that are composted of (<rule name>, <configuration>). The key is the <rule name> in the following structure:

{

"rule_name_1": {

<access-list-entries-config>,

<l2-fields-common-config> (optional),

<ip-protocol-fields-common-config> (optional),

<transport-fields-config> (optional),

<action-config>

},

"rule_name_2": {

<access-list-entries-config>,

<l2-fields-common-config> (optional),

<ip-protocol-fields-common-config> (optional),

<transport-fields-config> (optional),

<action-config>

}

}

· <access-list-entries-config>: Access Control list Entries (ACE) config that specifies the matching rule.

sequence-id : The sequence ID determines the order in which ACL entries are applied. The sequence ID must be unique

for each entry in an ACL set. Target devices should apply the ACL entry rules in ascending order determined by

sequence ID (low to high), rather than the relying only on order in the list. Minimum: 1. Maximum: 5000. For example: "sequence-id": 1.

· <l2-fields-common-config>: Ethernet header fields.

vlan-id : VLAN identifier number. Minimum: "1". Maximum: "4095".

· <ip-protocol-fields-common-config>: IP protocol fields common to IPv4 and IPv6.

protocol : The protocol carried in the IP packet, expressed either as its IP protocol number (e.g. "1000"), or by a defined identity:

- "IP_ICMP" : Internet Control Message Protocol (1).
- "IP_IGMP" : Internet Group Membership Protocol (2).
- "IP_TCP" : Transmission Control Protocol (6).
- "IP_UDP" : User Datagram Protocol (17).
- "IP_RSVP" : Resource Reservation Protocol (46).
- "IP_GRE" : Generic Routing Encapsulation (47).
- "IP_AUTH" : Authentication header, e.g., for IPSEC (51).
- "IP_PIM" : Protocol Independent Multicast (103).
- "IP_L2TP" : Layer Two Tunneling Protocol v.3 (115).

For example: "protocol": "IP_ICMP".

source-ip-address : The source IP address of a packet. Valid values are :

- The string of IPv4 prefix (e.g. "192.168.1.1/32").
- The string of IPv6 prefix (e.g. "2001::db:1/128").

destination-ip-address : The destination IP address of a packet. Valid values are :

- The string of IPv4 prefix (e.g. "192.168.1.2/32").
- The string of IPv6 prefix (e.g. "2001::db:2/128").

dscp : The number of the diffserv codepoint. Minimum: 0. Maximum: 63. For example: "dscp": 8. · <icmp-fields-config>: Configure data of ICMP and ICMPv6.

type : ICMP type. Minimum: "0". Maximum: "255".

code : ICMP code. Minimum: "0". Maximum: "255".

· <transport-fields-config>: Configuration data of transport-layer packet fields.

source-port : Source port (e.g. "1000") or range (e.g. "1000..2000"). Minimum: "0". Maximum: "65535".

destination-port : Destination port (e.g. "1000") or range (e.g. "1000..2000"). Minimum: "0". Maximum: "65535". tcp_flags : List of TCP flags to match. Valid values are :

- "TCP_FIN" : TCP FIN flag.
- "TCP_SYN" : TCP SYN flag.
- "TCP_RST" : TCP RST flag.
- "TCP_PSH" : TCP PSH flag.

- "TCP_ACK" : TCP ACK flag.

- "TCP_URG" : TCP URG flag.

For example: "tcp_flags": ["TCP_FIN", "TCP_SYN"].

· <action-config>: Config of the action type.

forwarding-action : Specifies the forwarding action. One forwarding action must be specified for each ACL entry. One of :

- "ACCEPT" : Accept the packet.

- "DROP" or "REJECT" : Drop the packet without sending any ICMP error message. ▪ For example: "forwarding-action": "ACCEPT".

set-traffic-class-action: Specifies the traffic class. Minimum: 0. Maximum: 7.

▪ For example: "set-traffic-class-action": 0.

Matching Criteria and Actions

Data Plane ACL

Due to chipset capability, the (ingress, egress)<stage> cannot apply to all (L3, L3V6, MIRROR, MIRRORV6, MIRROR_DSCP) matching criteria type. Following matrix shows the valid matching criteria <type> and <stage>. After the criteria matched, apply the action.

Matching Criteria

Match criteria	L3	L3	L3V6	L3V6	MIRROR	MIRROR	MIRRORV6	MIRRORV6	MIRROR_DSCP	MIRROR_DSCP
	Ingress	Egress	Ingress	Egress	Ingress	Egress	Ingress	Egress	Ingress	Egress
vlan-id	V	V	V	V	V	X	V	X	X	X
protocol	V	V	V	V	V		V		X	
source-ip-address	V (ipv4-prefix)	V (ipv4-prefix)	V (ipv6-prefix)	V (ipv6-prefix)	V (ipv4-prefix)		V (ipv6-prefix)		X	
destination-ip-address	V (ipv4-prefix)	V (ipv4-prefix)	V (ipv6-prefix)	V (ipv6-prefix)	V (ipv4-prefix)		V (ipv6-prefix)		X	
tcp_flags	V	V	V	V	V		V		X	
source-port	V	V (Not support port range)	V	V (Not support port range)	V		V		X	
destination-port	V	V (Not support	V	V (Not support	V		V		X	

		port range)		port range)				
dscp	X	X	X	X	V	V	V	
(ICMP) type	V	V	V	V	V	V	X	
(ICMP) code	V	V	V	V	V	V	X	

Note

"V" is valid matching critieria

"X" is invalid matching criteria

Allowed actions for matched criteria in data plane

Actions	L3	L3V6	MIRROR	MIRRORV6	MIRROR_DSCP
ACCEPT	V	V	V	V	V
DROP / REJECT	V	V	X	X	X
set-traffic-class-action	V	V	V	V	V

Note:

. Mirroring is a forwarding action, which does not support DROP or REJECT

. "V": valid action to apply when criteria matched, "X": not allowed, due to chipset capacity

Control Plane ACL

Control Plane ACL applies to the packet in software protocol stack, which inspects source IP address and tcp flags, and takes action in (Accept, Drop/Reject).

Matching Criteria

Actions	CTRLPLANE
protocol	X
source-ip-address	V
destination-ip-address	X
tcp_flags	V
source-port	X
destination-port	X
dscp	X
(ICMP) type	X

(ICMP) code	X

Note:

"V" is valid matching critieria

"X" is invalid matching criteria.

Allowed actions for matched criteria in control plane

Actions	CTRLPLANE
ACCEPT	V
DROP / REJECT	V
set-traffic-class-action	X

Port Mac Security

. Prerequisites

- Configure Port MAC Security
- Enable Port Mac Security
- Disable Port Mac Security
- Configure the maximum number of MAC address
- Configure the violation action
- Display the configuration of Port MAC Security

Port MAC Security is a feature that allows network administrators to restrict the use of MAC addresses on network devices, thereby enhancing network security.

This user guide provides brief instructions on how to configure and use Port MAC Security on your network devices with SONiC system.

For detailed information, refer to (202111) Port Mac Security in the CLI reference guide.

Prerequisites

The interface should be configured as a member in one of VLANs on the network devices.

Note: Currently Port Mac Security is only supported to be enabled on the physical port.

Configure Port MAC Security

Enable Port Mac Security

Example
admin@sonic:~$ sudo config interface pms Ethernet0 enabled

Disable Port Mac Security

Example
admin@sonic:~$ sudo config interface pms Ethernet0 disabled

Configure the maximum number of MAC address

Example
admin@sonic:~$ sudo config interface pms Ethernet0 max-mac-count 1000

Configure the violation action

Example
admin@sonic:~$ sudo config interface pms Ethernet0 violation-action protect
admin@sonic:~$ sudo config interface pms Ethernet0 violation-action restrict
admin@sonic:~$ sudo config interface pms Ethernet0 violation-action shutdown

Display the configuration of Port MAC Security

Example
admin@sonic:~$ show interface pms status

PINS

This user guide introduces the fundamental operation to configure P4 target which is supported by PINS(P4 Integrated Network Stack, a SONIC-based switch stack allowing switches to be programmed remotely via P4Runtime).

The content of this document includes using p4runtime shell as a P4 client to configure route entry, verifying the configuration in redis database and compiling P4 source code.

Note: that the configuration of ACL is not supported in this revision and there is a plan to support it in the future release.

· Introduction

· Setup the P4 Client on Your Local Environment

Check if the Docker Image of P4runtime-sh is Available in Your Local Repository

· Start the Connection with P4Runtime Sever and Push a Forwarding Pipeline Configuration with the Shell ·

An Example to Set a Route Entry on P4 Target through P4runtime Shell

- Configure Route Interface Table
- Configure Neighbor Table
- Configure Nexthop Table
- Configure Ipv4 Route Entry Table
- Verify the Table Entry in Redis Database

· Troubleshooting & Limitation

· Appendix A: Compiling P4 Source Code

· Reference

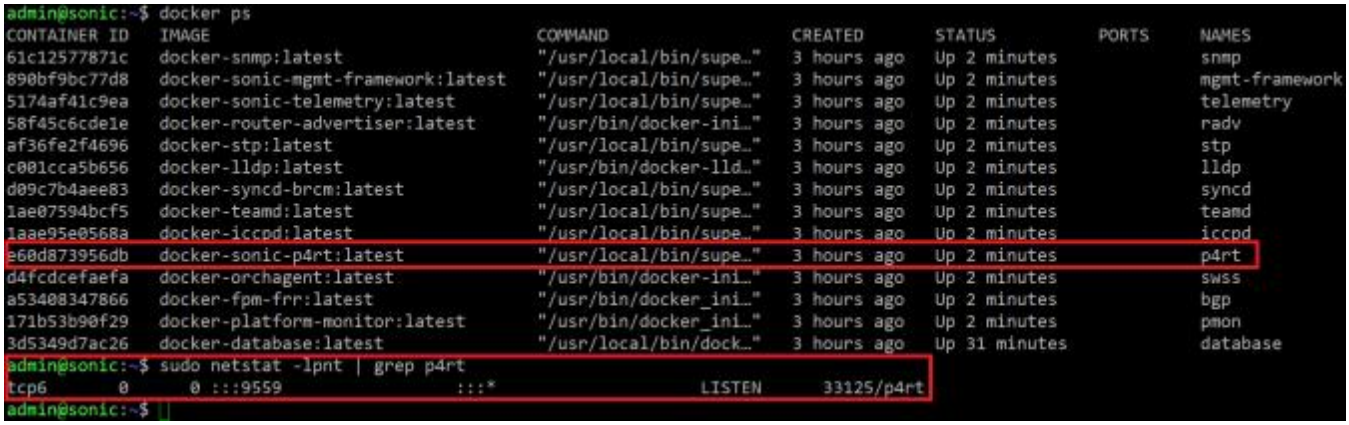
Introduction

PINS revealed that the device which is operated with SONiC can be programable by the remote SDN controllers over P4RT API.

A new container named "p4rt " has been added into SONiC system and a program running as a P4runtime Server is listening on TCP port 955 by default inside this new container.

Therefore, you can check with the following command to ensure that the SONiC image on your device has supported PINS function.

```
$ docker ps

$ sudo netstat -lpnt | grep p4rt
```



Setup the P4 Client on Your Local Environment

There are various tools acted as a P4 client to communicate with P4runtime server in SONiC. For example, "p4runtime-shell", "p4rt- client" and "onos"...etc.

This guide focuses on using "p4runtime-shell " as a P4 client to configure SONiC switch in the following section. And we adopt the method "Run with Docker", which is recommended in the "p4runtime-shell README.md".

Check if the Docker Image of P4runtime-sh is Available in Your Local Repository

A docker image should be located in your repository after following the step in the README.md.

```
$ docker images | grep p4runtime

p4lang/p4runtime-sh          latest          1beb21dedae0   2 months ago   152MB
```

Start the Connection with P4 Runtime Sever and Push a Forwarding Pipeline Configuration with the Shell

Usage
$ docker run -ti -v /tmp/:/tmp/ p4lang/p4runtime-sh --grpc-addr <server IP>:<server port> --device-id 0 --election-id 0,1 --config /tmp/p4info.txt,/tmp/bmv2.json

<server IP> : The IP address of your P4 target's management port.

<server port>: The tcp port of your P4 target, default should be "9559".

p4info.txt : The output of compiling P4 source code, it is also called P4Info file.

bmv2.json : The output of compiling P4 source code.

In the following example, we take the instantiation example(i.e. middleblock.p4) from google in sonic-pins but not activate the ACL related source code(middleblock_without_acl.p4).

Therefore, put the following files(middleblock_without_acl.p4info.txt, middleblock_without_acl.json) which are the output of compiling "middleblock_without_acl.p4" into /tmp/ directory on your server.

Example
$ docker run -ti -v /tmp/:/tmp/ p4lang/p4runtime-sh --grpc-addr 10.250.0.196:9559 --device-id 0 --election-id 0,1 --config /tmp/p4info.txt,/tmp/bmv2.json

An interaction interface will be opened when the connection has been established successfully.

The available commands in the P4Runtime shell can be referenced to the section in "p4runtime-shell README.md".

An Example to Set a Route Entry on P4 Target through P4runtime Shell

The example in this section is going to configure a route entry on P4 target which takes middleblock_without_acl.p4info.txt as its forwarding pipeline configuration step by step.

Configure Route Interface Table

p4runtime shell command
te = table_entry["ingress.routing.router_interface_table"](action="ingress.routing.set_port_and_src_mac") te.match["router_interface_id"]="1000" te.action["port"]="Ethernet0" te.action["src_mac"]="a8:2b:b5:f0:f0:f5" //publish the configuration to P4 target te.insert

The Output Result on P4runtime Shell:

```
P4Runtime sh >>> te = table_entry["ingress.routing.router_interface_table"](action="ingress.routing.set_port_and_src_mac")
             ... te.match["router_interface_id"]="1000"
             ... te.action["port"]="Ethernet0"
             ... te.action["src_mac"]="a8:2b:b5:f0:f0:f5"
field_id: 1
exact {
  value: "1000"
}

param_id: 1
value: "Ethernet0"

param_id: 2
value: "\250+\265\360\360\365"


P4Runtime sh >>> te.insert

P4Runtime sh >>>
```

Configure Neighbor Table

p4runtime shell command
te = table_entry["ingress.routing.neighbor_table"](action="ingress.routing.set_dst_mac") te.match["router_interface_id"]="1000" te.match["neighbor_id"]="fe80::aa2b:b5ff:fef0:ede3" te.action["dst_mac"]="a8:2b:b5:f0:ed:e3" //publish the configuration to P4 target te.insert

The Output Result on P4runtime Shell:

```
P4Runtime sh >>> te = table_entry["ingress.routing.neighbor_table"](action="ingress.routing.set_dst_mac")
            ...: te.match["router_interface_id"]="1000"
            ...: te.match["neighbor_id"]="fe80::aa2b:b5ff:fef0:ede3"
            ...: te.action["dst_mac"]="a8:2b:b5:f0:ed:e3"
field_id: 1
exact {
  value: "1000"
}

field_id: 2
exact {
  value: "fe80::aa2b:b5ff:fef0:ede3"
}

param_id: 1
value: "\250+\265\360\355\343"


P4Runtime sh >>> te.insert

P4Runtime sh >>>
```

Configure Nexthop Table

p4runtime shell command
te = table_entry["ingress.routing.nexthop_table"](action="ingress.routing.set_nexthop")
te.match["nexthop_id"]="nexthop1"
te.action["router_interface_id"]="1000"
te.action["neighbor_id"]="fe80::aa2b:b5ff:fef0:ede3"
//publish the configuration to P4 target
te.insert

The Output Result on P4runtime Shell:

```
P4Runtime sh >>> te = table_entry["ingress.routing.nexthop_table"](action="ingress.routing.set_nexthop")
            ...: te.match["nexthop_id"]="nexthop1"
            ...: te.action["router_interface_id"]="1000"
            ...: te.action["neighbor_id"]="fe80::aa2b:b5ff:fef0:ede3"
field_id: 1
exact {
  value: "nexthop1"
}

param_id: 1
value: "1000"

param_id: 2
value: "fe80::aa2b:b5ff:fef0:ede3"


P4Runtime sh >>> te.insert

P4Runtime sh >>>
```

Configure Ipv4 Route Entry Table

p4runtime shell command

```
te=table_entry["ingress.routing.ipv4_table"](action="ingress.routing.set_nexthop_id")
te.match["vrf_id"]=""
te.match["ipv4_dst"]="10.2.2.0/24"
te.action["nexthop_id"]="nexthop1"
//publish the configuration to P4 target
te.insert
```

Verify the Table Entry in Redis Database

In this section, we assume that the above configuration steps are executed without error. If there is any error occurred during the configuration, please refer to Troubleshooting part for assistance.

In PINS, after the request is handled by the P4runtime server, the relative P4 tables with prefix "P4RT_TABLE" will be added in APP DB.

The tables shown below are the mapping between the table defined in P4Info and the table added in APP DB.

Tables defined in P4Info	Tables defined in APP DB
ingress.routing.router_interface_table	P4RT_TABLE:FIXED_ROUTER_INTERFACE_TABLE
ingress.routing.neighbor_table	P4RT_TABLE:FIXED_NEIGHBOR_TABLE
ingress.routing.nexthop_table	P4RT_TABLE:FIXED_NEXTHOP_TABLE
ingress.routing.ipv4_table	P4RT_TABLE:FIXED_IPV4_TABLE
ingress.routing.ipv6_table	P4RT_TABLE:FIXED_IPV6_TABLE
ingress.routing.wcmp_group_table	P4RT_TABLE:FIXED_WCMP_GROUP_TABLE

The following command can extract the whole table entries configured by PINS. The dumped entries below are the expected output when PINS configuration operations are successful.

```
$ redis-cli KEYS "P4*"
```

```
admin@sonic:~$ redis-cli KEYS "P4*"
1) "P4RT_TABLE:FIXED_IPV4_TABLE:{\"match/ipv4_dst\":\"10.2.2.0/24\",\"match/vrf_id\":\"\"}"
2) "P4RT_TABLE:FIXED_ROUTER_INTERFACE_TABLE:{\"match/router_interface_id\":\"1000\"}"
3) "P4RT_TABLE:FIXED_NEIGHBOR_TABLE:{\"match/neighbor_id\":\"fe80::aa2b:b5ff:fef0:ede3\",\"match/router_interface_id\":\"1000\"}"
4) "P4RT_TABLE:FIXED_NEXTHOP_TABLE:{\"match/nexthop_id\":\"nexthop1\"}"
```

Troubleshooting & Limitation

- A request which is aimed at the specified table like "P4RT_TABLE:FIXED_IPV4_TABLE" is not allowed and will be forbidden by the P4Runtime Server.
- PINS only supports to handle the "read request" which is going to extract the whole table entries in P4 table.
- The P4 Client used in this document(i.e. P4runtime shell) has a limitation on reading the P4 tables entries. It only supports to get a specified table from P4Runtime Server.
- PINS does not support to replace the previous forwarding pipeline configuration during system runtime. Once a forwarding pipeline configuration had been pushed into P4 target, it needs to reboot the device before you push another configuration.

Appendix A: Compiling P4 Source Code

The tool used to compile P4 source code is p4c. It can be installed from package or source code.

Because the file middleblock_without_acl.p4 is based on the header definition in sai_p4, it should be put into the directory(sai_p4/instantiations/google/) in sonic-pins project before starting to compile.

Usage
$ p4c --target <target> --arch <arch> --p4runtime-files=<filename> <source file>

<target> : Specify the target behavior model.

<arch> : Specify the target architecture.

<filename>: The file name of output P4Info file.

<source> : The source file of p4 program.

Example
$ p4c --target bmv2 --arch v1model --p4runtime-files=middleblock_without_acl.p4info.txt middleblock_without_acl.p4