



Extreme ONE OS Switching v22.2.1.0 Command Reference

Comprehensive CLI Commands and Syntax

9039419-00 Rev AA
December 2025



Copyright © 2025 Extreme Networks, Inc. All rights reserved.

Legal Notice

Extreme Networks, Inc. reserves the right to make changes in specifications and other information contained in this document and its website without prior notice. The reader should in all cases consult representatives of Extreme Networks to determine whether any such changes have been made.

The hardware, firmware, software or any specifications described or referred to in this document are subject to change without notice.

Trademarks

Extreme Networks and the Extreme Networks logo are trademarks or registered trademarks of Extreme Networks, Inc. in the United States and/or other countries.

All other names (including any product names) mentioned in this document are the property of their respective owners and may be trademarks or registered trademarks of their respective companies/owners.

For additional information on Extreme Networks trademarks, see: <https://www.extremenetworks.com/about-extreme-networks/company/legal/trademarks>

Open Source Declarations

Some software files have been licensed under certain open source or third-party licenses.

End-user license agreements and open source declarations can be found at: <https://www.extremenetworks.com/support/policies/open-source-declaration/>

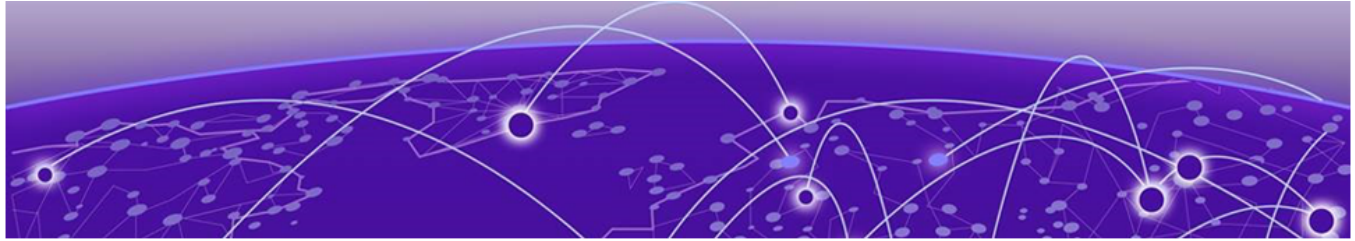


Table of Contents

Abstract.....	xvi
Preface.....	xvii
Text Conventions.....	xvii
Documentation and Training.....	xix
Open Source Declarations.....	xix
Training.....	xix
Help and Support.....	xix
Subscribe to Product Announcements.....	xx
Send Feedback.....	xx
About This Document.....	21
What's New in This Document.....	21
New Commands.....	21
Modified Commands.....	21
Supported Platforms.....	22
Using the CLI.....	23
User Accounts.....	23
Accessing the CLI.....	24
Command Modes.....	24
EXEC (#) Mode.....	24
Global Configuration Mode.....	24
Using the do Command as a Shortcut.....	25
Displaying the Running Configuration and the Running State.....	25
Displaying CLI Commands and Command Syntax.....	26
Completing CLI Commands.....	28
CLI Keyboard Shortcuts.....	29
Using CLI Command Output Modifiers.....	29
Unsupported Input Characters.....	30
System Diagnostic Commands.....	30
Extreme ONE OS Commands.....	31
aaa (Extreme ONE OS)	36
accounting (Extreme ONE OS)	38
accounting-method.....	40
authentication-method.....	42
admin-user.....	44
authentication (Extreme ONE OS).....	45
base-dn.....	47
bmc.....	49
bmc factory-reset.....	50
breakout.....	51
certificate-id.....	53

certificate-manager.....	55
certificate-manager delete.....	57
certificate-manager export ca-certificate	58
certificate-manager generate ssl-profile-id.....	60
certificate-manager import app-certificate.....	61
certificate manager import ca-certificate.....	63
certificate-manager import-pkcs.....	65
cipher.....	67
clear counters interface management	69
clear tech-support.....	70
clear trace.....	72
cli (Extreme ONE OS).....	73
client-alive.....	75
clock set.....	77
clock timezone.....	78
community (SNMP server system configuration).....	80
connector (Extreme ONE OS).....	82
console (logging system configuration).....	84
contact (Extreme ONE OS).....	85
copy backup-config file.....	86
copy default-config running-config.....	88
copy file backup-config.....	89
copy file running-config.....	91
copy file file.....	93
copy running-config backup-config.....	95
copy running-config file.....	97
crypto.....	99
dns (Extreme ONE OS).....	100
domain-name (Extreme ONE OS).....	101
enable (gRPC server configuration).....	103
enable (SSH server system configuration).....	105
enable (NTP system configuration).....	107
enable (SNMP server VRF system configuration).....	109
enable (Telnet server system configuration).....	111
event (AAA accounting system configuration).....	112
expiry-alert (AAA authentication password attributes configuration).....	114
expiry-alert (certificate manager configuration).....	116
facility.....	118
factory-reset.....	120
grpc-server.....	123
grub.....	125
hardware (Extreme ONE OS).....	127
host (Extreme ONE OS).....	128
hostname (Extreme ONE OS).....	130
interface extMgmt.....	131
interface management (Extreme ONE OS)	133
ipv4 address (BMC interface configuration).....	134
ipv4 gateway (BMC interface configuration).....	136
key.....	138

keychain.....	140
key-exchange.....	141
key-id.....	143
ldaps.....	144
license eula advanced_core.....	146
license eula advanced_features.....	148
location (Extreme ONE OS).....	149
logging (Extreme ONE OS).....	150
login-banner.....	151
ls (Extreme ONE OS)	153
mac.....	156
map-role group.....	158
min-version (TLS service-level system configuration).....	160
min-version (TLS system configuration).....	162
mode-transport.....	164
monitor.....	166
monitor (logging system configuration).....	168
motd-banner.....	169
mutual-tls.....	171
name-server (Extreme ONE OS).....	173
ntp (Extreme ONE OS).....	175
password-attributes.....	177
peer (NTP system configuration).....	181
port (GRPC server configuration).....	183
python (Extreme ONE OS).....	185
radsec.....	188
receive-lifetime.....	190
reload.....	192
remote-port.....	195
remote-server.....	197
rm backup-config.....	199
secret-key (keychain).....	200
secret-key (RADIUS server system configuration).....	201
secret-key (TACACS+ server system configuration).....	203
secure-forwarding tls.....	205
send-lifetime.....	207
server-group.....	209
server (LDAP server system configuration).....	211
server (NTP system configuration).....	213
server (RADIUS server system configuration).....	215
server (TACACS+ server system configuration).....	217
service (TLS system configuration).....	219
show backup-config.....	221
show backup-config replay-status.....	223
show bmc-status (Extreme ONE OS).....	225
show certificate-manager app-certificate.....	226
show certificate-manager ca-certificate.....	228
show cli history.....	230
show clock.....	231

show counters interface management	232
show firmware (Extreme ONE OS).....	235
show firmware history (Extreme ONE OS).....	236
show grpc-server gnmi capabilities.....	237
show grpc-server gnmi statistics.....	239
show interface.....	241
show interface brief.....	247
show interface management.....	249
show license.....	251
show logging audit.....	253
show logging raslogs.....	256
show ntp association (Extreme ONE OS).....	257
show ntp status (Extreme ONE OS).....	259
show running-config system snmp-server.....	260
show sysinfo all.....	262
show sysinfo chassis.....	264
show sysinfo fan.....	267
show sysinfo inventory all.....	268
show sysinfo inventory fan.....	269
show sysinfo inventory power-supply.....	271
show sysinfo led.....	272
show sysinfo media detected.....	273
show sysinfo media interface ethernet.....	275
show sysinfo media interface extMgmt.....	276
show sysinfo media supported.....	278
show sysinfo power-supply.....	280
show sysinfo sensor all.....	281
show sysinfo sensor cpu.....	282
show system contact.....	283
show system diagnostics.....	284
show system dns.....	286
show system health (Extreme ONE OS).....	287
show system internal apigw.....	292
show system internal cdb.....	293
show system internal cli service.....	295
show system internal k3s-cert-validity.....	297
show system internal msg-svc.....	298
show system internal sdb.....	305
show system location.....	307
show system logging facility.....	308
show system logging remote-server.....	310
show system mac-address-table.....	312
show system service (Extreme ONE OS).....	313
show tech-support.....	315
show tls min-version.....	316
show trace.....	317
show usb (Extreme ONE OS).....	319
show users (Extreme ONE OS).....	320
show version (Extreme ONE OS).....	321

show ztp dhcp logs.....	323
show ztp dhcp status.....	325
shutdown (BMC interface configuration).....	327
snmp-server (Extreme ONE OS).....	329
source-interface.....	331
ssh-server.....	334
ssh-server restart.....	336
ssl-profile-id (AAA LDAP server group system configuration).....	337
ssl-profile-id (AAA RADIUS server group system configuration).....	339
ssl-profile-id (AAA token validator system configuration).....	341
start-shell (Extreme ONE OS).....	343
system (Extreme ONE OS).....	344
system diagnostics run extended.....	345
system diagnostics run normal.....	347
system diagnostics update.....	349
system firmware commit (Extreme ONE OS).....	351
system firmware fullinstall (Extreme ONE OS)	353
system firmware rollback (Extreme ONE OS).....	355
system firmware uninstall (Extreme ONE OS).....	357
system firmware update (Extreme ONE OS)	358
system internal tdpa.....	360
system programmable update (Extreme ONE OS).....	362
system tech-support.....	364
telnet-server.....	368
terminal length (Extreme ONE OS).....	370
terminal timeout.....	371
tls.....	372
tls-profile-id.....	373
token-validator.....	375
tolerance.....	377
udp-port.....	379
unhide-secrets.....	381
unlock username.....	383
usb enable (Extreme ONE OS).....	384
user (Extreme ONE OS SNMP server system configuration).....	385
user (AAA authentication system configuration).....	387
username (GRUB system configuration).....	390
version (SNMP server VRF host system configuration).....	392
vrf (GRPC server configuration).....	394
vrf (logging remote server system configuration).....	396
vrf (NTP system configuration).....	398
vrf (Extreme ONE OS).....	400
ztp dhcp cancel.....	403
Extreme ONE OS Switching Commands.....	405
access-list	415
access-list (control plane configuration).....	417
access-list (interface configuration).....	419
actions (routing policy map statement configuration).....	422
activate (BGP address family configuration).....	424

activate (BGP peer group address family configuration).....	426
activate-network.....	428
add-paths receive.....	430
address-family (BGP configuration).....	433
address-family (BGP peer group configuration).....	435
address-type (MLAG peer configuration).....	437
advertisement (IPv6 router advertisement prefix configuration).....	439
agetime (ARP).....	441
agetime (neighbor discovery).....	442
allow-own-as.....	443
arp-proxy.....	445
arp-snooping.....	446
as-notation.....	447
as-path-set.....	449
auth-password.....	451
auto-offset.....	453
auto-boot.....	455
autoconfiguration (IPv6 router advertisement prefix configuration).....	457
auto-negotiate (Extreme ONE OS Switching).....	459
bfd.....	462
bgp-actions (routing policy statement actions configuration).....	463
bgp-conditions (routing policy statement conditions configuration).....	465
bgp-defined-sets.....	467
bridge-domain.....	469
bringup-delay (MLAG peer configuration).....	470
bus.....	471
channel-group.....	473
clear bgp neighbor	475
clear bgp peer-group.....	477
clear bgp vrf routes l2vpn-evpn nvo	479
clear bgp vrf routes l2vpn-evpn nvo route-distinguisher	481
clear bgp vrf routes l2vpn-evpn nvo route-distinguisher route-type	483
clear bgp vrf routes l2vpn-evpn nvo route-type	485
clear bgp vrf routes l2vpn-evpn nvo route-type arp vlan ipv4-prefix mac	487
clear bgp vrf routes l2vpn-evpn nvo route-type imr vlan originator	489
clear bgp vrf routes l2vpn-evpn nvo route-type mac vlan mac	491
clear bgp vrf routes l2vpn-evpn nvo route-type nd vlan ipv6-prefix mac	493
clear bgp vrf routes l2vpn-evpn nvo route-type prefix vlan ipv4-prefix	495
clear bgp vrf routes l2vpn-evpn nvo route-type prefix vlan ipv6-prefix	497
clear bgp vrf routes mac-vrf bridge-domain	499
clear bgp vrf routes mac-vrf bridge-domain route-type arp	500
clear bgp vrf routes mac-vrf bridge-domain route-type imr	502
clear bgp vrf routes mac-vrf bridge-domain route-type mac	504
clear bgp vrf routes mac-vrf bridge-domain route-type nd	506
clear counters access-list	508
clear counters bfd.....	510
clear counters bridge-domain	512
clear counters interface ethernet	513
clear counters interface port-channel	514

clear counters interface tunnel	515
clear counters lacp.....	516
clear counters lldp	517
clear counters mlag.....	519
clear counters qos interface control-plane queue	520
clear counters qos interface ethernet queue	521
clear ipv4 neighbor interface (Extreme ONE OS Switching)	523
clear ipv4 neighbor vrf.....	525
clear ipv6 neighbor interface.....	528
clear ipv6 neighbor vrf.....	530
clear l2 mac-move-detection shut-list.....	533
clear ipv4 route vrf (IP routing).....	535
clear ipv6 route vrf (IP routing).....	536
clear mac-address-table bridge-domain.....	537
cluster-id.....	539
community-set.....	541
conditions (routing policy map statement configuration).....	543
connect-retry.....	545
cos traffic-class.....	547
cpus.....	549
decapsulation (QoS NVO configuration).....	551
default-route-originate (BGP peer group address family configuration).....	553
default-traffic-class.....	555
deny (IPv4 ACL configuration).....	556
deny (IPv6 ACL configuration).....	561
deny (MAC ACL configuration).....	566
deploy (IAH VM configuration).....	569
description (BGP peer group configuration).....	571
description (bridge domain configuration).....	573
description (Ethernet interface configuration).....	574
description (IAH VM configuration).....	575
description (mirror configuration).....	577
destination interface (mirror configuration).....	578
destination (MLAG peer configuration).....	580
destination (tunnel interface configuration).....	582
disk (IAH VM configuration).....	584
dns (IAH VM configuration).....	586
dscp traffic-class.....	588
duplex (Ethernet interface configuration).....	590
ebgp-multihop	592
ecmp-max-path (global system configuration).....	595
ecmp-max-path (VRF instance configuration).....	596
enable (IPv6 router advertisement configuration).....	598
enable-bfd.....	600
enable-on-reload.....	603
encapsulation (QoS NVO configuration).....	605
evpn-instance (VRF configuration).....	607
export-policy (BGP peer group address family configuration).....	608
ext-community-set.....	610

extend-bd (MLAG configuration).....	612
fast-external-failover	615
fec (Ethernet interface configuration).....	617
file (IAH VM disk configuration).....	620
format (IAH VM disk configuration).....	622
graceful-restart (BGP address family configuration).....	624
graceful-restart (BGP configuration).....	626
graceful-restart (BGP peer group address family configuration).....	628
graceful-restart (BGP peer group configuration).....	630
hello-timer	633
helper-only (BGP graceful restart configuration).....	634
hold-time (BGP peer group timer configuration).....	636
holdtime (LLDP configuration).....	638
hop-limit	639
hostname (IAH VM configuration).....	641
iah fileinfo	643
iah (global configuration).....	645
iah vm	646
iah vm console	648
import-policy (BGP peer group address family configuration).....	649
iah vm import	651
iah vm upgrade	654
input (interface QoS configuration).....	656
instance-type	657
interface ethernet	659
interface (IAH VM network configuration).....	660
interface management ipv4 (IAH VM configuration).....	662
interface management ipv6 (IAH VM configuration).....	664
interface (MLAG configuration).....	666
interface port-channel (LAG configuration).....	668
interface (QoS configuration).....	669
interface tunnel (Extreme ONE OS Switching).....	671
interval (BFD profile configuration).....	673
interval (IPv6 router advertisement configuration).....	675
ip mtu (Extreme ONE OS Switching)	677
ip mtu (Extreme ONE OS Switching interface configuration).....	679
ipv4 address anycast-gateway (Virtual Ethernet interface configuration).....	681
ipv4 address dhcp (Extreme ONE OS Switching).....	683
ipv4 address (interface configuration).....	685
ipv4 anycast-gateway-mac	687
ipv4 neighbor (interface configuration).....	688
ipv4 neighbor reachable-time (interface configuration).....	690
ipv6 address anycast-gateway (Virtual Ethernet interface configuration).....	692
ipv6 address auto-config (Extreme ONE OS Switching).....	694
ipv6 address dhcp (Extreme ONE OS Switching)	695
ipv6 address (interface configuration).....	697
ipv6 anycast-gateway-mac	699
ipv6 neighbor (interface configuration).....	700
ipv6 neighbor reachable-time (interface configuration).....	702

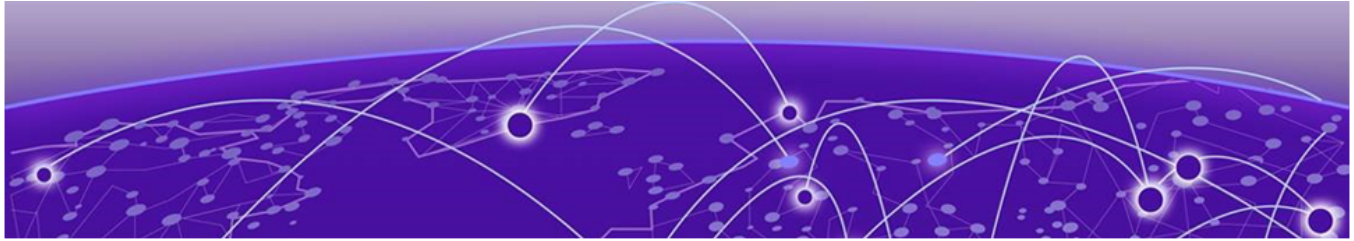
ipv6-router-advertisement.....	704
keepalive-interval.....	706
keepalive (MLAG configuration).....	708
keepalive (MLAG peer configuration).....	710
l3vni (VRF instance type configuration).....	712
l3vni (VRF instance type NVO configuration).....	714
lACP min-links.....	716
lACP port-priority.....	718
lACP interval.....	720
lACP system-mac.....	722
lACP system-priority.....	724
ldap cacert.....	726
ldap host.....	728
lifetime.....	730
link-error-disable.....	732
link-fault-signaling.....	735
listen-range (BGP peer group configuration).....	738
lldp.....	740
load-balance (Extreme ONE OS Switching global configuration).....	741
local-as-forced.....	743
local-as.....	745
local-pref-eq (routing policy statement bgp conditions configuration).....	747
loopback-mode.....	749
l2 mac-move-detection.....	750
mac-address-table aging-time.....	752
mac (MLAG configuration).....	754
maintenance-mode (system configuration).....	755
managed-config-flag.....	757
management-address.....	759
match-as-path-set (routing policy statement bgp conditions configuration).....	761
match-community-set (routing policy statement bgp conditions configuration).....	763
match-ext-community-set (routing policy statement bgp conditions configuration).....	765
match-prefix-set (routing policy map statement conditions configuration).....	767
match-set-options (routing policy BGP community set configuration).....	769
match-set-options (routing policy BGP extended community set configuration).....	771
med-eq (routing policy statement bgp conditions configuration).....	773
member.....	775
member nve.....	777
member (routing policy BGP AS path set configuration).....	779
member (routing policy BGP community set configuration).....	781
member (routing policy BGP extended community set configuration).....	783
member (BFD configuration).....	785
member (single tagged).....	787
member (tagged).....	789
member (untagged).....	791
member vni-domain.....	793
member (VRF configuration).....	795
mirror (global configuration).....	797
mirror (interface configuration).....	799

mlag	801
mode disable-unsolicited-ra	802
mtu (Extreme ONE OS Switching interface configuration).....	804
mtu (Extreme ONE OS Switching global configuration).....	807
nd-proxy	809
nd-snooping	810
neighbor (BGP peer group configuration).....	811
network (BGP address family configuration).....	813
network (IAH VM configuration).....	815
next-hop-enable-default (BGP address family configuration).....	817
next-hop-in (routing policy statement bgp conditions configuration).....	819
next-hop-recursion (BGP address family configuration).....	821
nexthop-self (BGP peer group address family configuration).....	823
nexthop-unchanged (BGP peer group address family configuration).....	825
ntp (IAH VM configuration).....	827
nve (EVPN instance configuration).....	829
nve (overlay configuration).....	831
nvo (BGP peer group configuration).....	833
nvo (QoS configuration).....	835
nvo	836
nvo (VRF instance type configuration).....	838
onlink (IPv6 router advertisement prefix configuration).....	840
other-config	842
output (interface QoS configuration).....	844
overlay-networks	845
password (IAH VM configuration).....	846
peer-group	847
peer (MLAG configuration).....	849
permit (IPv4 ACL configuration).....	851
permit (IPv6 ACL configuration).....	856
permit (MAC ACL configuration).....	861
ping	864
policy-result (routing policy statement actions configuration).....	867
policy (routing policy configuration).....	869
port ethernet	871
preferred-lifetime (IPv6 router advertisement prefix configuration).....	873
prefix (IPv6 router advertisement configuration).....	875
prefix-independent-convergence (Extreme ONE OS Switching)	877
prefix-set	879
prefix	881
profile (BFD configuration).....	883
protocol lacp	885
protocol lldp	886
qos (Extreme ONE OS Switching).....	888
ram	889
receive (LLDP configuration).....	891
remark-map (interface QoS configuration).....	893
remark-map (QoS configuration).....	895
remark (QoS configuration).....	897

remote-as (BGP peer group configuration).....	899
resilient-hash (VRF configuration).....	901
restart-time (BGP graceful restart configuration).....	902
retain-route-target-all (BGP L2VPN EVPN address family configuration).....	904
role (MLAG configuration).....	906
route-distinguisher (EVPN configuration).....	908
route-distinguisher (VRF configuration).....	910
route-policy	912
route-reflector-client	913
route-target (Extreme ONE OS Switching).....	915
router bgp	917
router-id (BGP configuration).....	919
send-default-route (BGP address family configuration).....	921
set-as-path-prepend (routing policy statement BGP actions configuration).....	923
set-community (routing policy statement BGP actions configuration).....	925
set-ext-community (routing policy statement BGP actions configuration).....	927
set-local-pref (routing policy statement BGP actions configuration).....	929
set-med (routing policy statement BGP actions configuration).....	931
set-next-hop (routing policy statement BGP actions configuration).....	933
set-route-origin (routing policy statement BGP actions configuration).....	935
show access-list	937
show anycast-gateway (IPv4 and IPv6).....	939
show bfd neighbors application (Extreme ONE OS Switching).....	941
show bfd neighbors dest-ip (Extreme ONE OS Switching).....	945
show bfd neighbors interface (Extreme ONE OS Switching).....	947
show bfd neighbors	948
show bfd profile	951
show bfd	953
show bgp vrf l2vpn-evpn	955
show bgp vrf neighbor	957
show bgp vrf neighbors	962
show bgp vrf routes	964
show bgp vrf summary	971
show bridge-domain	973
show counters bridge-domain	975
show counters interface ethernet	978
show counters interface port-channel	983
show counters interface tunnel	986
show counters lacp	987
show counters link-fault-signaling	989
show counters lldp interface	991
show counters lldp summary	993
show counters mlag peer	995
show counters qos interface control-plane queue	997
show counters qos interface ethernet queue	1000
show iah host	1002
show iah snapshot	1005
show iah vm	1006
show interface ethernet	1010

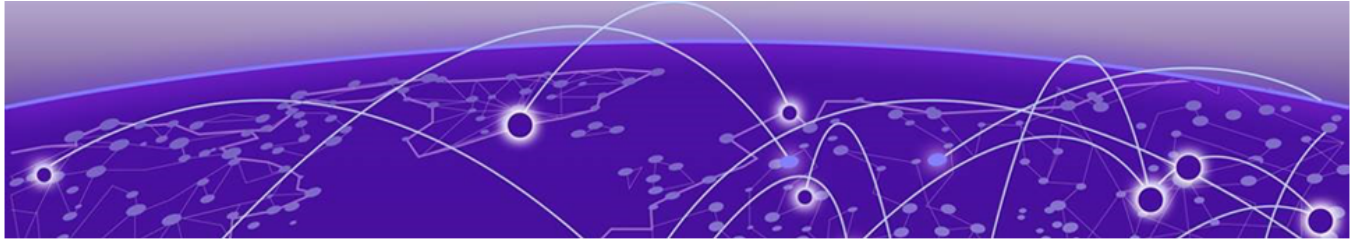
show interface ethernet link-error-disable.....	1012
show interface loopback-mode.....	1013
show interface subinterface.....	1014
show interface tunnel (Extreme ONE OS Switching).....	1016
show ipv4 interface.....	1018
show ipv6 interface.....	1020
show ipv4 neighbor interface.....	1022
show ipv4 neighbor vrf.....	1024
show ipv6 neighbor interface.....	1028
show ipv6 neighbor vrf.....	1030
show ipv4 route vrf (IP routing).....	1033
show ipv6 route vrf (IP routing).....	1038
show lacp interface.....	1041
show lacp system-identifier.....	1044
show link-fault-signaling	1045
show lldp neighbors interface	1046
show lldp neighbors	1049
show lldp	1051
show l2 mac-move-detection	1054
show l2 mac-move-detection shut-list	1055
show mac-address-table bridge-domain.....	1056
show mirror.....	1059
show mlag.....	1061
show mlag interfaces.....	1063
show mlag peer.....	1066
show mlag peer bfd.....	1069
show overlay-networks.....	1071
show qos interface.....	1072
show qos nvo.....	1075
show qos remark-map.....	1076
show qos traffic-class-map.....	1078
show running-config access-list	1080
show running-config lldp	1087
show system mac-address-table aging-time (Extreme ONE OS Switching).....	1088
show system maintenance-mode.....	1089
show system redundant-management.....	1090
shutdown (BGP peer group configuration).....	1091
shutdown (Ethernet interface configuration).....	1093
shutdown (subinterface configuration).....	1095
shutdown (tunnel interface configuration).....	1097
size (IAH VM disk configuration).....	1099
source-interface (MLAG peer keepalive configuration).....	1101
source-interface (NVE configuration).....	1103
source nve.....	1105
speed (Ethernet interface configuration).....	1107
speed (port channel interface configuration).....	1109
split-horizon.....	1111
src-protocol.....	1113
stale-route-time (BGP graceful restart configuration).....	1115

statement (routing policy configuration).....	1117
static-mac-address	1119
static-route	1122
static-route enable-bfd	1127
statistics (bridge domain configuration).....	1131
subinterface (tagged).....	1132
subinterface (untagged strict).....	1133
subinterface vlan (untagged).....	1134
suppress-arp	1136
suppress-nd	1137
suppress-tlv	1138
system maintenance-mode	1140
table-connections	1142
threshold-monitor	1143
threshold-monitor (CPU and memory).....	1145
timers (BGP peer group configuration).....	1147
timezone (IAH VM configuration).....	1149
traceroute	1151
traffic-class cos (QoS remark map configuration).....	1153
traffic-class dscp (QoS remark map configuration).....	1155
traffic-class-map (interface QoS input configuration).....	1157
traffic-class-map (QoS configuration).....	1159
transmit (LLDP configuration).....	1161
trust (interface QoS input configuration).....	1162
trusted-peer (IAH VM configuration).....	1164
tunnel-mode	1166
type (MLAG peer keepalive configuration).....	1168
type (tunnel interface configuration).....	1170
update-source (BGP peer group configuration)	1172
uplink-track interface (MLAG configuration).....	1176
use-multiple-paths (BGP address family configuration).....	1178
use-multiple-paths	1180
valid-lifetime (IPv6 router advertisement prefix configuration).....	1182
vlan-id	1185
vm (IAH configuration).....	1186
vni-domain (bridge domain configuration).....	1188
vni-domain (overlay configuration).....	1189
vrf (Extreme ONE OS Switching).....	1191



Abstract

The *Extreme ONE OS Switching Command Reference Guide* version 22.2.1.0 provides advanced technical guidance for configuring, managing, and troubleshooting enterprise-grade network switching and routing platforms. This guide details hierarchical CLI command structures. The reference is intended for advanced technical audiences requiring precise command syntax, configuration examples, and operational best practices for high-availability, secure, and scalable enterprise switching and routing environments.



Preface

Read the following topics to learn about:

- The meanings of text formats used in this document.
- Where you can find additional information and help.
- How to reach us with questions and comments.

Text Conventions

Unless otherwise noted, information in this document applies to all supported environments for the products in question. Exceptions, like command keywords associated with a specific software version, are identified in the text.

When a feature, function, or operation pertains to a specific hardware product, the product name is used. When features, functions, and operations are the same across an entire product family, such as Extreme Networks switches, the product is referred to as *the switch*.

Table 1: Notes and warnings

Icon	Notice type	Alerts you to...
	Tip	Helpful tips and notices for using the product
	Note	Useful information or instructions
	Important	Important features or instructions
	Caution	Risk of personal injury, system damage, or loss of data
	Warning	Risk of severe personal injury

Table 2: Text

Convention	Description
screen displays	This typeface indicates command syntax, or represents information as it is displayed on the screen.
The words <i>enter</i> and <i>type</i>	When you see the word <i>enter</i> in this guide, you must type something, and then press the Return or Enter key. Do not press the Return or Enter key when an instruction simply says <i>type</i> .
Key names	Key names are written in boldface, for example Ctrl or Esc . If you must press two or more keys simultaneously, the key names are linked with a plus sign (+). Example: Press Ctrl+Alt+Del
<i>Words in italicized type</i>	Italics emphasize a point or denote new terms at the place where they are defined in the text. Italics are also used when referring to publication titles.
NEW!	New information. In a PDF, this is searchable text.

Table 3: Command syntax

Convention	Description
bold text	Bold text indicates command names, keywords, and command options.
<i>italic</i> text	Italic text indicates variable content.
[]	Syntax components displayed within square brackets are optional. Default responses to system prompts are enclosed in square brackets.
{ x y z }	A choice of required parameters is enclosed in curly brackets separated by vertical bars. You must select one of the options.
x y	A vertical bar separates mutually exclusive elements.
< >	Nonprinting characters, such as passwords, are enclosed in angle brackets.
...	Repeat the previous element, for example, <i>member[member...]</i> .
\	In command examples, the backslash indicates a “soft” line break. When a backslash separates two lines of a command input, enter the entire command at the prompt without the backslash.

Documentation and Training

Find Extreme Networks product information at the following locations:

[Current Product Documentation](#)

[Release Notes](#)

[Hardware and Software Compatibility](#) for Extreme Networks products

[Extreme Optics Compatibility](#)

[Other Resources](#) such as articles, white papers, and case studies

Open Source Declarations

Some software files have been licensed under certain open source licenses. Information is available on the [Open Source Declaration](#) page.

Training

Extreme Networks offers product training courses, both online and in person, as well as specialized certifications. For details, visit the [Extreme Networks Training](#) page.

Help and Support

If you require assistance, contact Extreme Networks using one of the following methods:

[Extreme Portal](#)

Search the GTAC (Global Technical Assistance Center) knowledge base; manage support cases and service contracts; download software; and obtain product licensing, training, and certifications.

[The Hub](#)

A forum for Extreme Networks customers to connect with one another, answer questions, and share ideas and feedback. This community is monitored by Extreme Networks employees, but is not intended to replace specific guidance from GTAC.

[Call GTAC](#)

For immediate support: (800) 998 2408 (toll-free in U.S. and Canada) or 1 (408) 579 2800. For the support phone number in your country, visit www.extremenetworks.com/support/contact.

Before contacting Extreme Networks for technical support, have the following information ready:

- Your Extreme Networks service contract number, or serial numbers for all involved Extreme Networks products
- A description of the failure
- A description of any actions already taken to resolve the problem
- A description of your network environment (such as layout, cable type, other relevant environmental information)
- Network load at the time of trouble (if known)

- The device history (for example, if you have returned the device before, or if this is a recurring problem)
- Any related RMA (Return Material Authorization) numbers

Subscribe to Product Announcements

You can subscribe to email notifications for product and software release announcements, Field Notices, and Vulnerability Notices.

1. Go to [The Hub](#).
2. In the list of categories, expand the **Product Announcements** list.
3. Select a product for which you would like to receive notifications.
4. Select **Subscribe**.
5. To select additional products, return to the **Product Announcements** list and repeat steps 3 and 4.

You can modify your product selections or unsubscribe at any time.

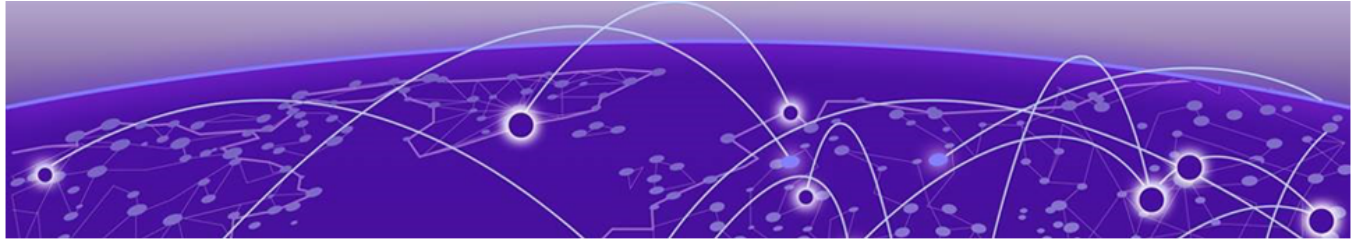
Send Feedback

The User Enablement team at Extreme Networks has made every effort to ensure that this document is accurate, complete, and easy to use. We strive to improve our documentation to help you in your work, so we want to hear from you. We welcome all feedback, but we especially want to know about:

- Content errors, or confusing or conflicting information.
- Improvements that would help you find relevant information.
- Broken links or usability issues.

To send feedback, email us at Product-Documentation@extremenetworks.com.

Provide as much detail as possible including the publication title, topic heading, and page number (if applicable), along with your comments and suggestions for improvement.



About This Document

[What's New in This Document](#) on page 21

[Supported Platforms](#) on page 22

What's New in This Document

New Commands

Extreme ONE OS 22.2.1.0 introduces the following commands:

- [clear bgp peer-group](#) on page 477
- [copy backup-config file](#) on page 86
- [copy file backup-config](#) on page 89
- [copy running-config backup-config](#) on page 95
- [license eula advanced_core](#) on page 146
- [license eula advanced_features](#) on page 148
- [min-version \(TLS service-level system configuration\)](#) on page 160
- [min-version \(TLS system configuration\)](#) on page 162
- [rm backup-config](#) on page 199
- [service \(TLS system configuration\)](#) on page 219
- [show backup-config](#) on page 221
- [show license](#) on page 251
- [show tls min-version](#) on page 316
- [speed \(port channel interface configuration\)](#) on page 1109
- [tls](#) on page 372

Modified Commands

None.

Supported Platforms

Extreme ONE OS Switching 22.2.1.0 supports Extreme 8520, Extreme 8720, Extreme 8730, and Extreme 8820 hardware platforms.

**Note**

Although many software and hardware configurations are tested and supported for this release, all possible configurations and scenarios are beyond this document's scope.

For information about other releases, see the documentation for those releases.



Using the CLI

[User Accounts](#) on page 23
[Accessing the CLI](#) on page 24
[Command Modes](#) on page 24
[Displaying the Running Configuration and the Running State](#) on page 25
[Displaying CLI Commands and Command Syntax](#) on page 26
[Completing CLI Commands](#) on page 28
[CLI Keyboard Shortcuts](#) on page 29
[Using CLI Command Output Modifiers](#) on page 29
[Unsupported Input Characters](#) on page 30
[System Diagnostic Commands](#) on page 30

The command line provides a powerful means for configuring, managing, and monitoring the device.

The following topics describe accessing and using the command-line interface (CLI), including syntax, command completion, shortcuts, and other helpful subjects.

User Accounts

A user account specifies that user's level of access to the device CLI.

The software uses role-based access control (RBAC) as the authorization mechanism. A *role* is a container for rules, which specify which commands can be executed and with which permissions. When you create a user account, you need to specify a role for that account. In general, *user* (as opposed to *user-level*) refers to any account to which the 'user' or 'admin' role can be assigned.

The software is provisioned with one default account. This is an administrator user named "admin." You must change the default password of 'admin' user upon first login.

Accounts with admin permissions can execute all commands supported on the device.

Accounts with user-level permissions can execute all **show** commands supported on the device. User-level accounts can also execute the following operational commands: **exit**, **ping**, and **traceroute**.

For more information on user accounts and roles, refer to the *Extreme ONE OS Security Command Reference*.

Accessing the CLI

After an IP address is assigned to the device, you can access the CLI through a serial console connection to the Ethernet management port or a Telnet or SSH session using the device management IP address.

For more information on a serial console connection, see the relevant *Hardware Installation Guide*. For information on a session connection, see the relevant *Management Configuration Guide*.

The procedure to access the CLI is the same through either the console interface or through a Telnet or SSH session; both access methods bring you to the login prompt. The following example shows the admin role logging into the device:

```
device login: admin
Password:*****
device#
```



Note

Multiple users can open sessions on the device and issue commands. The device supports a maximum of 32 CLI sessions.

Command Modes

The CLI uses an industry-standard hierarchical shell familiar to networking administrators.

EXEC (#) Mode

EXEC (#) mode supports all **clear** and **show** commands. In addition, you can perform some admin privileged actions, such as firmware upgrade, that can change the state of system. The following example shows the EXEC (#) prompt. At this prompt, you issue the **configure terminal** command to enter global configuration mode:

```
device# configure terminal
device(config)#
```

Global Configuration Mode

Global configuration mode supports commands that can change the device configuration. The system configuration is restored upon reboot. The global configuration mode provides access to subconfiguration modes for individual interfaces, VLANs, routing protocols, and other configuration areas. The following example shows how you access the interface subconfiguration mode by issuing the **interface** command with a specified interface:

```
device(config)# interface ethernet 0/1
device(config-if-eth-0/1)#
```


Using the do Command as a Shortcut

You can use the **do** command to save time when you are working in any configuration mode and you want to run a command in EXEC (#) mode.

For example, if you are configuring an Ethernet interface and you want to execute an EXEC (#) mode command, such as the **ls** command, you would first have to exit interface configuration mode. By using the **do** command with the **ls** command, you can ignore the need to change configuration modes, as shown in the following example:

```
device(config-if-eth-0/2)# do ls

config-file  :
pcap-file   :
-rw-r--r--  0  2025-03-07 04:32:14  README.md
tech-support :
firmware    :
coredumps   :
iah         :
device(config-if-eth-0/2)#
```



Note

The **ls** command also works in any configuration mode without the **do** command, but command autocomplete (using the **Tab** key) is disabled.

Displaying the Running Configuration and the Running State

You can display the system configuration using either of two commands: **show running-config** or **show running-state**.

show running-config displays the configuration accepted by the system. This shows the user intent saved in the configuration.

show running-state displays the configuration that is currently applied in the system. This refers to the current operational state or status of the system.

You can optionally use one of a variety of keywords following the **show running-config** command:

```
device# show running-config ?

<cr>
access-list      Access-list information
bfd              BFD commands
bgp              Show BGP commands
bridge-domain    Bridge Domain
control-plane    Control plane configurations
hardware         hardware commands
iah              interface
interface        Interface
ip               The internet protocol information
ipv4             The internet protocol (ipv4) information
ipv6             The internet protocol (ipv6) information
keychain         keychain configurations
lACP             LACP commands
lldp             LLDP commands
```

```

load-balance    load-balance commands
mac             MAC
mirror          Mirror
mlag            MLAG commands
mtu             MTU Config
overlay-networks Display overlay configuration
qos             QoS
route-policy    Routing Policy configurations
system          System information
threshold-monitor threshold-monitor information
vrf             Vrf
|
device#

```

You can optionally use one of the same set of keywords following the **show running-state** command:

```

device# show running-state ?

<cr>
access-list      Access-list information
bfd              BFD commands
bgp              Show BGP commands
bridge-domain    Bridge Domain
control-plane    Control plane configurations
hardware         hardware commands
iah              iah
interface        Interface
ip               The internet protocol information
ipv4             The internet protocol (ipv4) information
ipv6             The internet protocol (ipv6) information
keychain         keychain configurations
lacp             LACP commands
lldp             LLDP commands
load-balance     load-balance commands
mac             MAC
mirror          Mirror
mlag            MLAG commands
mtu             MTU Config
overlay-networks Display overlay configuration
qos             QoS
route-policy    Routing Policy configurations
system          System information
threshold-monitor threshold-monitor information
vrf             Vrf
|
device#

```

Displaying CLI Commands and Command Syntax

You can display commands and syntax information in any mode and from any point in the command hierarchy.

Enter a question mark (?) in any command mode to display the list of commands available in that mode:

```

device#
cat          Display contents of file
certificate-manager Certificate management operations
clear        Clear operational data
clock        Clock settings
configure    Configuration mode

```

```

copy          Copy files or configurations
efa           Extreme Fabric Automation (EFA) settings
exit          Exit current session
factory-reset Reset device to factory default settings
iah           Integrated Application Hosting (IAH) operations
list          List all supported operational commands
ls            List disk and USB contents
measured-boot Measured boot operations
monitor       System monitoring operations
no            Negate a command or set its defaults
ping          Ping destination address or hostname
python        Start Python interpreter or execute Python script
quit          Exit current session
reload        Reboot the device
rm            Remove file
show          Show system information
ssh-server    SSH server operations
start-shell   Switch to host shell
system        System operational commands
terminal      Terminal settings
traceroute    Trace route to remote IP address or hostname
unlock        Unlock user account
usb           Enable USB storage access
ztp           Zero Touch Provisioning (ZTP) operations
device#

```

To display a list of commands that start with the same characters, type the characters followed by a question mark (?):

```

device# c?
cat          Display contents of file
certificate-manager Certificate management operations
clear        Clear operational data
clock        Clock settings
configure    Configuration mode
copy         Copy files or configurations
device#

```

To display the keywords and arguments associated with a command, enter the keyword followed by a space and then a question mark (?):

```

device# terminal ?
length      Set number of lines on a screen
timeout     Sets the interval that command interpreter wait for user input
device#

```

If the question mark (?) is typed within an incomplete keyword, but the keyword matches several keywords, the CLI displays help for all the matching keywords:

```

device# show b?
bfd          Display Bidirectional Forwarding Detection (BFD) information
bgp          Display Border Gateway Protocol (BGP) information
bridge-domain Display Bridge Domain (BD) information
device#

```

The CLI accepts abbreviations for commands. This example is the abbreviation for the **configure terminal** command:

```

device# co t
device(config)#

```

This example is valid only when an abbreviation represents only one possible command.

If the device does not recognize a command after you press **Enter**, an error message displays:

```
device# hookup
-----^
%Error: Unexpected token 'hookup'.
device#
```

If you enter an incomplete command, an error message displays:

```
device# show
-----^
%Error: Command incomplete: 'show'
device#
```

Completing CLI Commands

To complete the spelling of a command or keyword automatically, begin typing the command or keyword and then press **Tab**. For example, at the CLI command prompt, type `configure t` and press **Tab**:

```
device# configure t
```

The CLI displays the following command:

```
device# configure terminal
```

If there is more than one command or keyword associated with the characters typed, the CLI displays all choices. For example, at the CLI command prompt, type `system s` and press **Tab**:

```
device# show b<tab>
```

The CLI displays the following command:

```
bfd          Display Bidirectional Forwarding Detection (BFD) information
bgp          Display Border Gateway Protocol (BGP) information
bridge-domain Display Bridge Domain (BD) information
device# show b
```

CLI Keyboard Shortcuts

The following table lists CLI keyboard shortcuts.

Table 4: CLI keyboard shortcuts

Keystroke	Description
Ctrl+A	Moves the cursor to the beginning of the command line.
Ctrl+B (or the left arrow key)	Moves the cursor back one character.
Ctrl+C	Escapes and terminates command prompts and ongoing tasks (such as lengthy displays), and displays a fresh command prompt.
Ctrl+E	Moves the cursor to the end of the command line.
Ctrl+F (or the right arrow key)	Moves the cursor forward one character.
Ctrl+N (or the down arrow key)	Displays commands in the history buffer with the most recent command displayed last.
Ctrl+P (or the up arrow key)	Displays commands in the history buffer with the most recent command displayed first.
Ctrl+U or Ctrl+X	Deletes all characters from the cursor to the beginning of the command line.
Ctrl+W	Deletes the last word you typed.
Esc B	Moves the cursor back one word.
Esc F	Moves the cursor forward one word.



Note

In EXEC (#) mode, use the **show cli history** command to list the commands most recently entered. The device retains the history of the last 1000 commands entered for the current session.

Using CLI Command Output Modifiers

You can filter the output of the CLI **show** commands by using the output modifiers described below.

Table 5: CLI command output modifiers

Output modifier	Description
append	Appends output text to a file.
begin	Begin with the line that matches the specified expression.
count	Counts the number of lines in the output.
exclude	Exclude lines that match the specified expression.
include	Include lines that match the specified expression.

Table 5: CLI command output modifiers (continued)

Output modifier	Description
last	Select last n lines of output. Number of lines to list.
linnum	Enumerates lines in the output.
more	Paginate output.
nomore	Suppress pagination of the output.
save	Save output text to a file.
until	End with the line that matches the specified expression.

Unsupported Input Characters

Characters dependent on combinations of the **AltGr** key and another key are never supported.

**Note**

The **AltGr** key is the **Alt** key to the right of the spacebar.

System Diagnostic Commands

System diagnostic commands, such as "show system internal", are developed and intended for specialized troubleshooting.

You should work closely with Extreme Networks technical support in executing such commands and interpreting their results.

**Note**

Not all diagnostic commands are documented.



Extreme ONE OS Commands

[aaa \(Extreme ONE OS\)](#) on page 36
[accounting \(Extreme ONE OS\)](#) on page 38
[accounting-method](#) on page 40
[authentication-method](#) on page 42
[admin-user](#) on page 44
[authentication \(Extreme ONE OS\)](#) on page 45
[base-dn](#) on page 47
[bmc](#) on page 49
[bmc factory-reset](#) on page 50
[breakout](#) on page 51
[certificate-id](#) on page 53
[certificate-manager](#) on page 55
[certificate-manager delete](#) on page 57
[certificate-manager export ca-certificate](#) on page 58
[certificate-manager generate ssl-profile-id](#) on page 60
[certificate-manager import app-certificate](#) on page 61
[certificate manager import ca-certificate](#) on page 63
[certificate-manager import-pkcs](#) on page 65
[cipher](#) on page 67
[clear counters interface management](#) on page 69
[clear tech-support](#) on page 70
[clear trace](#) on page 72
[cli \(Extreme ONE OS\)](#) on page 73
[client-alive](#) on page 75
[clock set](#) on page 77
[clock timezone](#) on page 78
[community \(SNMP server system configuration\)](#) on page 80
[connector \(Extreme ONE OS\)](#) on page 82
[console \(logging system configuration\)](#) on page 84
[contact \(Extreme ONE OS\)](#) on page 85
[copy backup-config file](#) on page 86
[copy default-config running-config](#) on page 88
[copy file backup-config](#) on page 89
[copy file running-config](#) on page 91

[copy file file](#) on page 93
[copy running-config backup-config](#) on page 95
[copy running-config file](#) on page 97
[crypto](#) on page 99
[dns \(Extreme ONE OS\)](#) on page 100
[domain-name \(Extreme ONE OS\)](#) on page 101
[enable \(gRPC server configuration\)](#) on page 103
[enable \(SSH server system configuration\)](#) on page 105
[enable \(NTP system configuration\)](#) on page 107
[enable \(SNMP server VRF system configuration\)](#) on page 109
[enable \(Telnet server system configuration\)](#) on page 111
[event \(AAA accounting system configuration\)](#) on page 112
[expiry-alert \(AAA authentication password attributes configuration\)](#) on page 114
[expiry-alert \(certificate manager configuration\)](#) on page 116
[facility](#) on page 118
[factory-reset](#) on page 120
[grpc-server](#) on page 123
[grub](#) on page 125
[hardware \(Extreme ONE OS\)](#) on page 127
[host \(Extreme ONE OS\)](#) on page 128
[hostname \(Extreme ONE OS\)](#) on page 130
[interface extMgmt](#) on page 131
[interface management \(Extreme ONE OS\)](#) on page 133
[ipv4 address \(BMC interface configuration\)](#) on page 134
[ipv4 gateway \(BMC interface configuration\)](#) on page 136
[key](#) on page 138
[keychain](#) on page 140
[key-exchange](#) on page 141
[key-id](#) on page 143
[ldaps](#) on page 144
[license eula advanced_core](#) on page 146
[license eula advanced_features](#) on page 148
[location \(Extreme ONE OS\)](#) on page 149
[logging \(Extreme ONE OS\)](#) on page 150
[login-banner](#) on page 151
[ls \(Extreme ONE OS\)](#) on page 153
[mac](#) on page 156
[map-role group](#) on page 158
[min-version \(TLS service-level system configuration\)](#) on page 160
[min-version \(TLS system configuration\)](#) on page 162
[mode-transport](#) on page 164
[monitor](#) on page 166

[monitor \(logging system configuration\)](#) on page 168
[motd-banner](#) on page 169
[mutual-tls](#) on page 171
[name-server \(Extreme ONE OS\)](#) on page 173
[ntp \(Extreme ONE OS\)](#) on page 175
[password-attributes](#) on page 177
[peer \(NTP system configuration\)](#) on page 181
[port \(GRPC server configuration\)](#) on page 183
[python \(Extreme ONE OS\)](#) on page 185
[radsec](#) on page 188
[receive-lifetime](#) on page 190
[reload](#) on page 192
[remote-port](#) on page 195
[remote-server](#) on page 197
[rm backup-config](#) on page 199
[secret-key \(keychain\)](#) on page 200
[secret-key \(RADIUS server system configuration\)](#) on page 201
[secret-key \(TACACS+ server system configuration\)](#) on page 203
[secure-forwarding tls](#) on page 205
[send-lifetime](#) on page 207
[server-group](#) on page 209
[server \(LDAP server system configuration\)](#) on page 211
[server \(NTP system configuration\)](#) on page 213
[server \(RADIUS server system configuration\)](#) on page 215
[server \(TACACS+ server system configuration\)](#) on page 217
[service \(TLS system configuration\)](#) on page 219
[show backup-config](#) on page 221
[show backup-config replay-status](#) on page 223
[show bmc-status \(Extreme ONE OS\)](#) on page 225
[show certificate-manager app-certificate](#) on page 226
[show certificate-manager ca-certificate](#) on page 228
[show cli history](#) on page 230
[show clock](#) on page 231
[show counters interface management](#) on page 232
[show firmware \(Extreme ONE OS\)](#) on page 235
[show firmware history \(Extreme ONE OS\)](#) on page 236
[show grpc-server gnmi capabilities](#) on page 237
[show grpc-server gnmi statistics](#) on page 239
[show interface](#) on page 241
[show interface brief](#) on page 247
[show interface management](#) on page 249
[show license](#) on page 251

[show logging audit](#) on page 253
[show logging raslogs](#) on page 256
[show ntp association \(Extreme ONE OS\)](#) on page 257
[show ntp status \(Extreme ONE OS\)](#) on page 259
[show running-config system snmp-server](#) on page 260
[show sysinfo all](#) on page 262
[show sysinfo chassis](#) on page 264
[show sysinfo fan](#) on page 267
[show sysinfo inventory all](#) on page 268
[show sysinfo inventory fan](#) on page 269
[show sysinfo inventory power-supply](#) on page 271
[show sysinfo led](#) on page 272
[show sysinfo media detected](#) on page 273
[show sysinfo media interface ethernet](#) on page 275
[show sysinfo media interface extMgmt](#) on page 276
[show sysinfo media supported](#) on page 278
[show sysinfo power-supply](#) on page 280
[show sysinfo sensor all](#) on page 281
[show sysinfo sensor cpu](#) on page 282
[show system contact](#) on page 283
[show system diagnostics](#) on page 284
[show system dns](#) on page 286
[show system health \(Extreme ONE OS\)](#) on page 287
[show system internal apigw](#) on page 292
[show system internal cdb](#) on page 293
[show system internal cli service](#) on page 295
[show system internal k3s-cert-validity](#) on page 297
[show system internal msg-svc](#) on page 298
[show system internal sdb](#) on page 305
[show system location](#) on page 307
[show system logging facility](#) on page 308
[show system logging remote-server](#) on page 310
[show system mac-address-table](#) on page 312
[show system service \(Extreme ONE OS\)](#) on page 313
[show tech-support](#) on page 315
[show tls min-version](#) on page 316
[show trace](#) on page 317
[show usb \(Extreme ONE OS\)](#) on page 319
[show users \(Extreme ONE OS\)](#) on page 320
[show version \(Extreme ONE OS\)](#) on page 321
[show ztp dhcp logs](#) on page 323
[show ztp dhcp status](#) on page 325

[shutdown \(BMC interface configuration\)](#) on page 327

[snmp-server \(Extreme ONE OS\)](#) on page 329

[source-interface](#) on page 331

[ssh-server](#) on page 334

[ssh-server restart](#) on page 336

[ssl-profile-id \(AAA LDAP server group system configuration\)](#) on page 337

[ssl-profile-id \(AAA RADIUS server group system configuration\)](#) on page 339

[ssl-profile-id \(AAA token validator system configuration\)](#) on page 341

[start-shell \(Extreme ONE OS\)](#) on page 343

[system \(Extreme ONE OS\)](#) on page 344

[system diagnostics run extended](#) on page 345

[system diagnostics run normal](#) on page 347

[system diagnostics update](#) on page 349

[system firmware commit \(Extreme ONE OS\)](#) on page 351

[system firmware fullinstall \(Extreme ONE OS\)](#) on page 353

[system firmware rollback \(Extreme ONE OS\)](#) on page 355

[system firmware uninstall \(Extreme ONE OS\)](#) on page 357

[system firmware update \(Extreme ONE OS\)](#) on page 358

[system internal tdpa](#) on page 360

[system programmable update \(Extreme ONE OS\)](#) on page 362

[system tech-support](#) on page 364

[telnet-server](#) on page 368

[terminal length \(Extreme ONE OS\)](#) on page 370

[terminal timeout](#) on page 371

[tls](#) on page 372

[tls-profile-id](#) on page 373

[token-validator](#) on page 375

[tolerance](#) on page 377

[udp-port](#) on page 379

[unhide-secrets](#) on page 381

[unlock username](#) on page 383

[usb enable \(Extreme ONE OS\)](#) on page 384

[user \(Extreme ONE OS SNMP server system configuration\)](#) on page 385

[user \(AAA authentication system configuration\)](#) on page 387

[username \(GRUB system configuration\)](#) on page 390

[version \(SNMP server VRF host system configuration\)](#) on page 392

[vrf \(GRPC server configuration\)](#) on page 394

[vrf \(logging remote server system configuration\)](#) on page 396

[vrf \(NTP system configuration\)](#) on page 398

[vrf \(Extreme ONE OS\)](#) on page 400

[ztp dhcp cancel](#) on page 403

aaa (Extreme ONE OS)

Use the **aaa** command to enter AAA system configuration (config-system-aaa) mode.

Syntax

aaa

Command Default

Authentication, Authorization, and Accounting (AAA) is not configured on the device.

Parameters

This command has no keywords or variables.

Modes

System configuration (config-system)

Usage Guidelines

This command enters the mode required for configuring the AAA settings on a device. They include settings for the accounting method (TACACS+ or RADIUS), event logging, authentication method (such as LDAP or RADIUS), credentials and roles, password requirements, and server configuration.

Examples

The following example enters AAA system configuration (config-system-aaa) mode:

```
device# configure terminal
device(config)# system
device(config-system)# aaa
device(config-system-aaa)#
```

The following example displays the AAA configuration for the TACACS+ server group that is running currently on the device:

```
device# show running-config system aaa

system
aaa
  accounting
    accounting-method tacacs+
    event login
    event command
  authentication
    admin-user admin password-hashed
    $6$6xVuJKNl$cvFtWdM70QRFZpCIxXOGbmWjIz1Vh0iFYnRNhjuCvcJHcee.8tLV9lg8LnD58M2hakM.RbVkkfWcXT
    .GowuD//
    authentication-method tacacs+ local
    password-attributes
    min-length 10
```

```
min-uppercase 2
min-lowercase 3
min-special 2
min-numeric 2
max-password-age 10
max-retry 4
lockout-duration 5
server-group tacacs+
server 2.2.2.2
    secret-key-hashed SeH06jSmdLQslwsXZKVHcifR/80uY9fcwSOEXU80w+KEmAxy1wb7FYiJYY/
6DZfxPUAlwYnkTOOfRO7jWiDzdA==
    source-interface ethernet 0/2.20
    vrf tenant-1
!
!
!
device#
```

accounting (Extreme ONE OS)

Use the **accounting** command to enter AAA accounting system configuration (config-system-aaa-accounting) mode.

Syntax

accounting

Command Default

No accounting settings for Authentication, Authorization, and Accounting (AAA) are configured on the device.

Parameters

This command has no keywords or variables.

Modes

AAA system configuration (config-system-aaa)

Usage Guidelines

This command enters the mode required for configuring the AAA accounting settings on a device. They include settings for the accounting method (TACACS+ or RADIUS) and event logging.

Examples

The following example enters AAA accounting system configuration (config-system-aaa-accounting) mode and configures AAA on the device. This example includes the settings for AAA accounting using TACACS+:

```
device# configure terminal
device(config)# system
device(config-system)# aaa
device(config-system-aaa)# accounting
device(config-system-aaa-accounting)# accounting-method tacacs+
device(config-system-aaa-accounting)# event login
device(config-system-aaa-accounting)# event command
device(config-system-aaa-accounting)# exit
device(config-system-aaa)#
```

The following example displays the AAA configuration for the TACACS+ server group that is running currently on the device. The output includes the settings for AAA accounting using TACACS+:

```
device# show running-config system aaa

system
aaa
```

```
authentication
  admin-user admin password-hashed $6$u83YhQlRKjz6Tc6H$xUzntTLBemkSKNzTKxdlaw3G/
  bTim9mJlgdD95FGG80c5DvmfOloUF1la3b5QbPb68MAspDw8aDhvRGCPsYog.
  !
  accounting
    accounting-method tacacs+
    event login
    event command
  !
!
device#
```

accounting-method

Use the **accounting-method** command to specify that TACACS+ or RADIUS external servers are used to log AAA accounting information.

Syntax

```
accounting-method { tacacs+ | radius }  
no accounting-method
```

Command Default

The AAA accounting method is not specified.

Parameters

tacacs+

Specifies that a TACACS+ server is used for accounting.

radius

Specifies that a RADIUS server is used for accounting.

Modes

AAA accounting system configuration (config-system-aaa-accounting)

Usage Guidelines

The LDAP server does not support accounting.

gNMI activities are not logged to a RADIUS server if accounting is enabled.

Use the **no** form of this command to disable the logging of AAA accounting information to an external server.

Examples

The following example configures AAA on the device. This example specifies that a TACACS+ external server is used to log AAA accounting information:

```
device# configure terminal  
device(config)# system  
device(config-system)# aaa  
device(config-system-aaa)# accounting  
device(config-system-aaa-accounting)# accounting-method tacacs+  
device(config-system-aaa-accounting)# event login  
device(config-system-aaa-accounting)# event command  
device(config-system-aaa-accounting)# exit  
device(config-system-aaa)#
```


The following example displays the AAA configuration for the TACACS+ server group that is running currently on the device. The output shows that a TACACS+ external server is used to log AAA accounting information:

```
device# show running-config system aaa

system
  aaa
    authentication
      admin-user admin password-hashed $6$u83YhQlRKjz6Tc6H$xUzntTLBemkSKNzTKxdlaw3G/
bTim9mJlgdD95FGG80c5DVmf0loUF11a3b5QbPb68MAspDw8aDhvRGCPsYog.
    !
    accounting
      accounting-method tacacs+
      event login
      event command
    !
  !
!
device#
```

authentication-method

Use the **authentication-method** command to specify the AAA authentication protocol to be used.

Syntax

```
authentication-method { ldap | tacacs+ | radius } local  
no authentication-method { ldap | tacacs+ | radius } local
```

Command Default

The device performs Authentication, Accounting, and Authorization (AAA) via the local database on the device.

Parameters

ldap

Uses Lightweight Directory Access Protocol (LDAP) servers.

tacacs+

Uses Terminal Access Controller Access-Control System Plus (TACACS+) servers.

radius

Uses Remote Authentication Dial-In User Service (RADIUS) servers.

local

Uses the local device database if prior authentication methods are not active or if authentication fails. This is a required keyword.

Modes

AAA authentication system configuration (config-system-aaa-authentication)

Usage Guidelines

This command configures the AAA authentication method. You can use LDAP, TACACS+, or RADIUS remote authentication as the primary method. Locally configured authentication is always the secondary method.

Examples

The following example configures AAA on the device. This example specifies that the authentication method will use TACACS+ servers:

```
device# configure terminal  
device(config)# system  
device(config-system)# aaa  
device(config-system-aaa)# authentication
```

```
device(config-system-aaa-authentication)# authentication-method tacacs+ local
device(config-system-aaa-authentication)#
```

The following example displays the AAA configuration for the TACACS+ server group that is running currently on the device. In this example, the AAA login sequence uses TACACS+ servers for authentication; if prior authentication methods are not active or if TACACS+ authentication fails, the device uses the local device database for authentication:

```
device# show running-config system aaa

system
  aaa
    authentication
      admin-user admin password-hashed $6$u83YhQlRKjz6Tc6H$xUzntTLBemkSKNzTKxdlaw3G/
bTim9mJlgdD95FGG80c5DVmfOloUF1la3b5QbPb68MAspDw8aDhvRGCPsYog.
      authentication-method tacacs+ local
    !
  !
!
device#
```

admin-user

Use the **admin-user** command to modify the username or credentials for the default admin account that is used to configure AAA authentication.

Syntax

```
admin-user username { password | password-hashed } password
```

Command Default

The default admin user uses default credentials for Authentication, Authorization, and Accounting (AAA) on the device. The default username is "admin".

Parameters

username

Username of the admin user. Use a unique name. Do not use a name that you already specified with AAA **user** command.

password

Specifies that the password of the admin user is in plain text.

password-hashed

Specifies that the password of the admin user is hashed.

password

Password of the admin user.

Modes

AAA authentication system configuration (config-system-aaa-authentication)

Usage Guidelines

This command modifies the username and password for the default admin user that is used for configuring AAA authentication.

You cannot remove the default admin user account altogether.

Examples

The following example configures AAA on the device. In this example, the password for the default admin user is changed to a nondefault value:

```
device# configure terminal
device(config)# system
device(config-system)# aaa
device(config-system-aaa)# authentication
device(config-system-aaa-authentication)# admin-user admin password !Pass@123!
device(config-system-aaa-authentication)#
```

authentication (Extreme ONE OS)

Use the **authentication** command to enter AAA authentication system configuration (config-system-aaa-authentication) mode.

Syntax

authentication

Command Default

No authentication settings for Authentication, Authorization, and Accounting (AAA) are configured on the device.

Parameters

This command has no keywords or variables.

Modes

AAA system configuration (config-system-aaa)

Usage Guidelines

This command enters the mode required for configuring the AAA authentication settings on a device. They include the authentication method (such as TACACS+ or RADIUS), password requirements, user roles and passwords. and credentials for the default admin user.

Examples

The following example enters AAA authentication system configuration (config-system-aaa-authentication) mode and configures AAA authentication on the device:

```
device# configure terminal
device(config)# system
device(config-system)# aaa
device(config-system-aaa)# authentication
device(config-system-aaa-authentication)# admin-user admin password !Pass@123!
device(config-system-aaa-authentication)# authentication-method tacacs+ local
device(config-system-aaa-authentication)# user test1 role user password #Pass@456!
inactivity-expiry-period 20 inactivity-warning-period 15
device(config-system-aaa-authentication)# user test2 role user password $Pass@789!
inactivity-expiry-period 15 inactivity-warning-period 10
device(config-system-aaa-authentication)#
```

The following example displays the AAA configuration for the TACACS+ server group that is running currently on the device. In this example, AAA authentication is configured:

```
device# show running-config system aaa
```

```
system
aaa
  authentication
    admin-user admin password-hashed $6$36501360$KbvljM9Oxu6mVXzGDv/
fKyBkcQ8V9FvaPMKLmHJNYFgV.IMwd4wGQfqdqCh3s36AETpB5U6BNoezjY2x3.DEY/
    user test1 role user password-hashed
$6$211308932$3QX3w8sC9JKPvaiC6slu1R2RReTcaqwKkT44vgRkCeqKgnpWlpnFIaaCdhZNrT3NXGbKpshqnHBKL
/Fuva2jp/ inactivity-expiry-period
20 inactivity-warning-period 15
    user test2 role user password-hashed
$6$420427520$wAD4wSswD4kI48YHdgWtxZmuZaEpVFNTBQKIwQ9y.wbDYgLNHMTTb4Lsr9WNsa8CLb5A.1VFs/
zccCb2ZD8ji/ inactivity-expiry-period
15 inactivity-warning-period 10
    authentication-method tacacs+ local
  !
!
!
device#
```

base-dn

Use the **base-dn** command to configure the base domain name for a Lightweight Directory Access Protocol (LDAP) server in an LDAP server group.

Syntax

base-dn *domain-name*

no base-dn

Command Default

The current LDAP server host has no base domain name.

Parameters

base-dn *domain-name*

Base domain name. Use 1 to 255 characters.

Modes

AAA LDAP server group server system configuration (config-system-aaa-server-group-ldap-server-address)

Usage Guidelines

You can use the base domain name to perform search operations in the active directory tree.

Use the **no** form of this command to remove the configured base domain name.

Examples

The following example adds a server with IPv4 address 3.3.3.3 to the LDAP server group on the device and configures a base domain name for the server:

```
device# configure terminal
device(config)# system
device(config-system)# aaa
device(config-system-aaa)# server-group ldap
device(config-system-aaa-server-group-ldap)# server 3.3.3.3
device(config-system-aaa-server-group-ldap-server-3.3.3.3)# base-dn example.com
device(config-system-aaa-server-group-ldap-server-3.3.3.3)# end
device#
```

The following example displays the LDAP server group configuration that is running currently on the device. In this example, the LDAP server group is configured with one server host, which uses example.com as its base domain name:

```
device# show running-config system aaa server-group ldap
```

```
system
aaa
  server-group ldap
    server 3.3.3.3
      base-dn example.com
    !
  !
!
device#
```

The following example displays the LDAP server group state that is running currently on the device. In this example, the LDAP server group is configured with one server host, which uses example.com as its base domain name:

```
device# show running-state system aaa server-group ldap

system
aaa
  server-group ldap
    server 3.3.3.3
      base-dn example.com
    !
  !
!
device#
```

bmc

Use the **bmc** command to enter BMC configuration (bmc) mode.

Syntax

bmc

Parameters

This command has no keywords or variables.

Modes

Global configuration (config)

Usage Guidelines

Use this command to enter the mode for accessing the internal baseboard management controller (BMC) LAN settings. The BMC is a microcontroller and the major component of the Intelligent Platform Management Interface (IPMI) subsystem, which assists in remote hardware monitoring, management, and troubleshooting of the device. These settings include the IP address source (DHCP or a static address), IP address, subnet mask, MAC address, and default gateway IP address.

To display details about the internal BMC LAN configuration on the device, see [show bmc-status \(Extreme ONE OS\)](#) on page 225.

Examples

The following example enters BMC configuration (bmc) mode:

```
device# configure terminal
device(config)# bmc
device(bmc)#
```

bmc factory-reset

Use the **bmc factory-reset** command to perform a cold reboot of the internal baseboard management controller (BMC).

Syntax

bmc factory-reset

Parameters

This command has no keywords or variables.

Modes

EXEC (#)

Usage Guidelines

A cold reboot of the BMC removes all BMC configuration and initializes the BMC with the factory default settings. The BMC is a microcontroller and the major component of the Intelligent Platform Management Interface (IPMI) subsystem, which assists in remote hardware monitoring, management, and troubleshooting of the device. The settings that are reset include the LAN settings such as the IP address source (DHCP or a static address), IP address, subnet mask, MAC address, and default gateway IP address.

To display details about the internal BMC LAN configuration on the device, see [show bmc-status \(Extreme ONE OS\)](#) on page 225.

Examples

The following example performs a cold reboot of the BMC on the device:

```
device# bmc factory-reset
%Info: WARNING: BMC and System will be rebooted sequentially after bmc factory reset! Do
you want to continue? [y/n]:
```

breakout

Use the **breakout** command to configure a breakout mode for a connector.

Syntax

```
breakout { 4x10g | 4x25g | 4x100g }
```

Command Default

Breakout mode is not configured on the specified connector.

Parameters

4x10g

Configures four 10g breakout interfaces on the port.

4x25g

Configures four 25g breakout interfaces on the port.

4x100g

Configures four 100g breakout interfaces on the port.

Modes

Connector configuration (config-hardware)

Usage Guidelines

Use this command to configure breakout mode for the supported connectors.



Note

Enabling breakout on an interface is a disruptive action and will make the port unavailable.

You do not need to reboot a device for the breakout configuration change to take effect.

When you configure a breakout mode on an Ethernet port, the original interface is deleted and replaced with four new ports with the breakout speed. The new breakout interfaces in the default configuration are created automatically. These new interfaces are identified with the name of the original interface followed by a suffix. For example, the breakout interfaces of connector 0/25 have the following naming convention:

```
interface ethernet 0/25:1
interface ethernet 0/25:2
interface ethernet 0/25:3
interface ethernet 0/25:4
```

The existing port must not be part of a port channel.

You must shut down the port before using this command.

To configure a connector, see [connector \(Extreme ONE OS\)](#) on page 82.

Use the **no** form of this command to reset a port to non-breakout mode. This deconfigures and deletes the breakout interfaces. The original Ethernet interface in the default configuration is created automatically.

Examples

The following example configures a connector on the device and a breakout mode for the connector. In this example, port 0/1 is deleted from the configuration and replaced with ports 1/1:1, 1/1:2, 1/1:3, and 1/1:4 with the default port configuration:

```
device# configure terminal
device(config)# hardware
device(config-hardware)# connector 0/1
device(config-connector-0/1)# breakout 4x10g
device(config-connector-0/1)#
```

The following example displays the hardware configuration that is running currently on the device:

```
device# show running-config

! Firmware: 10.2.1.0, Application(s): 22.2.1.0
! Date and Time: Tue, 2025-12-09 15:16:13 EST

...

device(config)# show running-config hardware

hardware
  connector 0/1
    breakout 4x10g
!
device#
```

certificate-id

Use the **certificate-id** command to associate an SSL profile containing a gRPC server certificate with the gRPC server instance.

Syntax

```
certificate-id certificate-name  
no certificate-id
```

Command Default

The gRPC server instance on the device has no associated certificate.

Parameters

certificate-name
Name of the certificate. Use 1 to 64 characters.

Modes

gRPC server system configuration (config-system-grpc-server-*name*)

Usage Guidelines

Use this command to associate an SSL profile (containing the gRPC server certificate) with gRPC for a TLS secured connection. The SSL profile must exist in the system before you run this command. To create one, see [certificate-manager import app-certificate](#) on page 61.

A gRPC server instance must be associated with an SSL profile through the **certificate-id** command. This profile should contain the gRPC server certificate.

For mutual authentication, a certificate authority (CA) certificate must also be imported to the same profile. For example:

```
device# certificate-manager generate ssl-profile-id ssl-reserved-generated certificate-  
extension san 1.1.1.1  
device# certificate-manager import ssl-profile-id ssl-reserved-generated ca-certificate  
protocol scp host 1.1.1.1 certificate /tmp/cert.crt user user1 password pass1 vrf mgmt-vrf  
device#
```

You must disable the gRPC server instance before using this command (or any other command) to alter its configuration. Then, you must re-enable the instance to restore its functionality. To disable or enable a gRPC server instance, see [enable \(gRPC server configuration\)](#) on page 103.

Use the **no** form of this command to disassociate the profile from the gRPC server instance.

Examples

The following example configures a gRPC server instance. In this example, an SSL profile named test_cert_id is associated with the instance:

```
device# configure terminal
device(config)# system
device(config-system)# grpc-server mygrpcserver
device(config-system-grpc-server-mygrpcserver)# certificate-id test_cert_id
device(config-system-grpc-server-mygrpcserver)# port 9340
device(config-system-grpc-server-mygrpcserver)# mutual-tls
device(config-system-grpc-server-mygrpcserver)# vrf vrf-blue
device(config-system-grpc-server-mygrpcserver)# enable
device(config-system-grpc-server-mygrpcserver)#
```

The following example displays the gRPC server instance configuration that is running currently on the device. In this example, an SSL profile named test-cert-id is associated with the gRPC service:

```
device# show running-config system grpc-server

system
  grpc-server mygrpcserver
    certificate-id test_cert_id
    port 9340
    mutual-tls
    vrf vrf-blue
    enable
!
device#
```

certificate-manager

Use the **certificate-manager** command to enter Certificate manager system configuration (config-system-cert-mgr) mode.

Syntax

certificate-manager

Command Default

The subcommands for this command and this container all have default values that do not appear in the running configuration unless you change them. The show command in the example below will show that a certificate manager is configured with the default values.

Parameters

This command has no keywords or variables.

Modes

System configuration (config-system)

Usage Guidelines

Use this command to enter the mode required for configuring certificate manager settings such as the expiry alert.

Table 6: Configuration Options for Certificate Manager

Configuration Option	Description
end	End current mode and change to enable mode
exit	Exit current mode and down to parent mode
expiry-alert	Certificate expiry alert period
list	List all supported configuration commands
pwd	Display current mode

Examples

The following example shows how to enter Certificate manager system configuration (config-system-cert-mgr) mode and configure the certificate manager expiry alert:

```
device# configure terminal
device(config)# system
```

```
device(config-system)# certificate-manager
device(config-system-cert-mgr)# expiry-alert
device(config-system-cert-mgr-exp)# critical 30
device(config-system-cert-mgr-exp)# major 60
device(config-system-cert-mgr-exp)# minor 80
device(config-system-cert-mgr-exp)# info 90
device(config-system-cert-mgr-exp)#
```

The following example displays the certificate manager configuration that is running currently on the device:

```
device# show running-config system certificate-manager

system
  certificate-manager
    expiry-alert
      critical 30
      major 60
      minor 80
      info 90
    !
  !
!
device#
```


certificate-manager delete

Use the **certificate-manager delete** command to delete application certificates or CA certificates from the system certificates store.

Syntax

```
certificate-manager delete { ssl-profile-id profile-name | all }
```

Parameters

ssl-profile-id *profile-name*

Name of an SSL profile containing the imported application certificate or certificate authority (CA) certificate to delete. Use 1 to 64 characters. You must restart the services that were using the deleted certificate for the changes to take effect.

all

Deletes all application certificates and CA certificates from the certificate store. You must restart the services that were using these deleted certificates for the changes to take effect.

Modes

EXEC (#)

Examples

The following example deletes all application certificates and CA certificates from the certificate store:

```
device# certificate-manager delete all
Deleted all profiles successfully.
device#
```

The following example deletes the application certificate or CA certificate contained in an SSL profile named sp1 from the certificate store:

```
device# certificate-manager delete ssl-profile-id sp1
Deleted ssl-profile-id sp1 successfully.
device#
```

certificate-manager export ca-certificate

Use the **certificate-manager export ca-certificate** command to copy system default trusted certificate authority (CA) certificates to an external remote server for use in establishing gNMI and gNOI connections.

Syntax

```
certificate-manager export ca-certificate default protocol { scp |  
  sftp } remote-server remote-ip remote-file file-name user remote-user  
  password remote-user-password [ source-ip address ] [ vrf vrf-name ]
```

Parameters

default

System default trusted CA certificate.

protocol { scp | sftp }

Protocol to use for retrieval.

remote-server remote-ip

IPv4 or IPv6 address of the remote host server.

remote-file file-name

File on the remote server where the CA certificate is copied. Use 1 to 512 characters.

user remote-user

User name on the remote server host. Use 1 to 64 characters.

password remote-user-password

Remote server host password. Use 1 to 64 characters.

source-ip address

(Optional) Source IPv4 or IPv6 address to be used to export the certificate. Only unicast address are supported.

- For an IPv4 address, use IPv4 notation (*A.B.C.D*). For example, 10.0.0.0.
- For an IPv6 address, use the *x::x::x::x* format. For example, 1001::1.

vrf vrf-name

(Optional) VRF instance to be used to export the certificate.

Modes

EXEC (#)

Usage Guidelines

The gRPC Network Management Interface (gNMI) is a gRPC-based protocol used to modify and retrieve configuration from a target device and control and generate

telemetry streams from a target device to a data collection system. The gNMI service defines operations for configuration management, operational state retrieval, and bulk data collection through streaming telemetry to provide a powerful way to manage devices.

The gRPC Network Operations Interface (gNOI) defines a set of gRPC-based microservices for running operational commands on network devices. The gNOI system service lets you perform system operations on a target network device, including device reboots, software upgrades, and network troubleshooting. You can execute supported “cert” service remote procedure calls (RPCs) to manage certificates on the network device. Using gNOI operations allows a single set of microservices to manage large-scale multivendor networks efficiently.

Use this command to export a CA certificate to an external remote server to make it available for importing later into a system certificates store.

PEM is the only supported certificate format.

X.509 is the only supported certificate type.

Examples

The following example copies the system default trusted CA certificates to an external remote server:

```
device# certificate-manager export ca-certificate default protocol scp remote-server
1.1.1.1 remote-file /tmp/cert.pem user user1 password **** vrf mgmt-vrf
Exported switch 'default' CA certificate successfully
device#
```

certificate-manager generate ssl-profile-id

Use the **certificate-manager generate ssl-profile-id** command to generate an application certificate and a private key and add them to a reserved SSL profile in the system certificates store.

Syntax

```
certificate-manager generate ssl-profile-id ssl-reserved-generated  
certificate-extension san san-entry-1 [ san-entry-2 ... ]
```

Parameters

san *san-entry-1* [*san-entry-2...*]

Subject Alternative Name (SAN) entries for the certificate. These are additional hosts to be secured by the same certificate. Use an IPv4 or IPv6 address or a domain name. You can specify up to five space-separated SAN entries.

- For an IPv4 address, use IPv4 notation (*A.B.C.D*). For example, 10.0.0.0.
- For an IPv6 address, use the *x::x::x::x* format. For example, 1001::1.

Modes

EXEC (#)

Usage Guidelines

This command generates an application certificate and a private key. These are then copied to a reserved SSL profile for use by an application that requires a certificate for TLS connectivity.

To use this certificate and private key, the application instance must be associated with this SSL profile. For a gRPC server, use the following command to associate the SSL profile:

```
device(config-system)# grpc-server  
device(config-system-grpc-server-DEFAULT)# certificate-id ssl-reserved-generated
```

Examples

The following example generate a certificate and a private key and adds them to the system certificates store:

```
device# certificate-manager generate ssl-profile-id ssl-reserved-generated certificate-  
extension san 1.1.1.1  
Generated app certificate successfully for ssl-profile-id ssl-reserved-generated  
device#
```

certificate-manager import app-certificate

Use the **certificate-manager import app-certificate** command to copy an application certificate and (optionally) a private key from an external remote server to the system certificates store.

Syntax

```
certificate-manager import ssl-profile-id profile-name app-certificate  
protocol { scp | sftp } host remote-ip certificate certificate-  
file [ key key-file ] user remote-user password remote-user-password  
[ source-ip address ] [ vrf vrf-name ]
```

Parameters

ssl-profile-id *profile-name*

Name of the SSL profile to which the certificate is imported. Use 1 to 64 characters.

protocol { **scp** | **sftp** }

Protocol to use for retrieval.

host *remote-ip*

IPv4 or IPv6 address of the remote server host.

certificate *certificate-file*

Certificate file in PEM format. Use 1 to 512 characters.

key *key-file*

(Optional) Key file in PEM format. Use 1 to 512 characters.

user *remote-user*

User name on the remote server host. Use 1 to 64 characters.

password *remote-user-password*

Password on the remote server host. Use 1 to 64 characters.

source-ip *address*

(Optional) Source IPv4 or IPv6 address to be used to import the CA certificate. Only unicast addresses are supported.

- For an IPv4 address, use IPv4 notation (*A.B.C.D*). For example, 10.0.0.0.
- For an IPv6 address, use the *x::x::x* format. For example, 1001::1.

vrf *vrf-name*

(Optional) VRF instance to be used to import the CA certificate.

Modes

EXEC (#)

Usage Guidelines

Use this command to copy a certificate and (optionally) a private key from an external remote server to the system certificates store. If a private key is omitted, the imported certificate can be used only for token validation (but not for TLS handshake).

An SSL profile can be shared across applications.

PEM is the only supported certificate format.

X.509 is the only supported certificate type.

If a gRPC server instance using this certificate ID is already enabled, you must disable it before using this command and then re-enable it afterward.

Examples

The following example imports application certificates from an external remote server:

```
device# certificate-manager import ssl-profile-id sp1 app-certificate protocol scp host
1.1.1.1 certificate /tmp/cert.pem key /tmp/key.pem user user1 password **** vrf mgmt-vrf
Imported app certificate successfully to ssl-profile-id sp1
device#
```

The following example imports application certificates (without a key) from an external remote server:

```
device# certificate-manager import ssl-profile-id sp1 app-certificate protocol scp host
1.1.1.1 certificate /tmp/cert.pem user user1 password **** vrf mgmt-vrf
Warning: Importing app-cert without key. This certificate cannot be used for tls
handshake, it can be only used for token validation
Imported app certificate successfully to ssl-profile-id sp1
device#
```

certificate manager import ca-certificate

Use the **certificate manager import ca-certificate** command to copy trusted certificate authority (CA) certificates from an external remote server to the trusted certificates list of the system.

Syntax

```
certificate manager import ssl-profile-id profile-name ca-certificate  
protocol { scp | sftp } host remote-ip certificate ca-certificate-  
file user remote-user password remote-user-password [ source-ip  
address ] [ vrf vrf-name ]
```

Parameters

ssl-profile-id *profile-name*

Name of the SSL profile to which the certificate is imported. Use 1 to 64 characters.

protocol { **scp** | **sftp** }

Protocol to use for retrieval.

host *remote-ip*

IPv4 or IPv6 address of the remote server host.

certificate *ca-certificate-file*

CA certificate file in PEM format. Use 1 to 512 characters.

user *remote-user*

User name on the remote server host. Use 1 to 64 characters.

password *remote-user-password*

Remote server host password. Use 1 to 64 characters.

source-ip *address*

(Optional) Source IPv4 or IPv6 address to be used to import the certificate. Only unicast addresses are supported.

- For an IPv4 address, use IPv4 notation (*A.B.C.D*). For example, 10.0.0.0.
- For an IPv6 address, use the *x::x::x::x* format. For example, 1001::1.

vrf *vrf-name*

(Optional) VRF instance to be used to import the certificate.

Modes

EXEC (#)

Usage Guidelines

For an application requiring a CA certificate for verifying a peer certificate for TLS connectivity, use this command to copy the CA certificate to the specified SSL profile.

This command imports a CA certificate from trusted certificate authority to use for validating peers.

An SSL profile can be shared across applications.

To use a CA certificate, an application instance must be associated with the SSL profile. System logging (syslog), LDAP over SSL/TLS (LDAPS), and RADIUS over TLS (RADSEC) each require a CA certificate for verifying the peer certificate for TLS connectivity. The following commands provide examples of how to associate an application instance with an SSL profile for these applications respectively:

```
kvm-x86-64(config-system-logging-remote-server-1.1.1.1)# tls-profile-id sp1
kvm-x86-64(config-system-aaa-server-group-ldap-server-1.1.1.1)# ssl-profile-id sp1
kvm-x86-64(config-system-aaa-server-group-radius-server-1.1.1.1)# ssl-profile-id sp1
```

Examples

The following example imports trusted CA certificates from an external remote server:

```
device# certificate-manager import ssl-profile-id sp1 ca-certificate protocol scp host
1.1.1.1 certificate /tmp/cert.crt user vikram password **** vrf mgmt-vrf
Imported CA certificate successfully to ssl-profile-id sp1
device#
```


certificate-manager import-pkcs

Use the **certificate-manager import-pkcs** command to copy a PKCS certificate key bundle from an external remote server to the system certificates store.

Syntax

```
certificate-manager import-pkcs ssl-profile-id profile-name app-  
certificate protocol { scp | sftp } host remote-ip file pkcs-file-  
name passphrase file-passphrase user remote-user password remote-  
user-password [ source-ip address ] [ vrf vrf-name ]
```

Parameters

ssl-profile-id *profile-name*

Name of the SSL profile to which the certificate is imported. Use 1 to 64 characters.

protocol { **scp** | **sftp** }

Protocol to use for retrieval.

host *remote-ip*

IPv4 or IPv6 address of the remote host server.

file *pkcs-file-name*

Certificate file in Public Key Cryptography Standards (PKCS) format. Use 1 to 512 characters.

passphrase *file-passphrase*

Passphrase to unlock the PKCS-formatted certificate file. Use 1 to 64 characters.

user *remote-user*

Remote host server user name. Use 1 to 64 characters.

password *remote-user-password*

Remote host server password. Use 1 to 64 characters.

source-ip *address*

(Optional) Source IPv4 or IPv6 address to be used to import the PKCS certificate. Only unicast addresses are supported.

- For an IPv4 address, use IPv4 notation (*A.B.C.D*). For example, 10.0.0.0.
- For an IPv6 address, use the *x::x::x::x* format. For example, 1001::1.

vrf *vrf-name*

(Optional) VRF instance to be used to import the PKCS certificate.

Modes

EXEC (#)

Usage Guidelines

Use this command to copy a PKCS certificate key bundle to the reserved SSL profile for an application requiring a certificate for TLS connectivity. To use this certificate, the application instance must be associated with this SSL profile.

For a gRPC server, use the following command to associate an SSL profile:

```
device(config-system)# grpc-server
device(config-system-grpc-server-DEFAULT)# certificate-id sp1
device#
```

X.509 is the only supported certificate type.

If a gRPC server instance using this SSL profile ID is already enabled, you must disable it before using this command and then re-enable it afterward.

Examples

The following example imports a PKCS certificate key bundle from an external remote server:

```
device# certificate-manager import-pkcs ssl-profile-id sp1 app-cert protocol scp host
1.1.1.1 file /tmp/cert.pkcs12 passphrase **** user user1 password **** vrf mgmt-vrf
Imported app certificate successfully to ssl-profile-id sp1
device#
```

cipher

Use the **cipher** command to configure the SSH server to use a set of ciphers in an order of priority that you specify.

Syntax

```
cipher { [ aes128-cbc ] [ aes192-cbc ] [ aes256-cbc ] [ aes128-ctr ]  
          [ aes192-ctr ] [ aes256-ctr ] [ aes128-gcm-openssh ] [ aes256-gcm-  
          openssh ] [ chacha20-poly1305-openssh ] }  
  
no cipher
```

Command Default

The following ciphers are configured in the following priority order (first is highest):

```
aes256-gcm-openssh  
aes256-ctr  
aes256-cbc  
chacha20-poly1305-openssh  
aes192-ctr  
aes192-cbc  
aes128-gcm-openssh  
aes128-ctr  
aes128-cbc
```

Parameters

```
{ [ aes128-cbc ] [ aes192-cbc ] [ aes256-cbc ] [ aes128-ctr ]  
  [ aes192-ctr ] [ aes256-ctr ] [ aes128-gcm-openssh ] [ aes256-gcm-  
  openssh ] [ chacha20-poly1305-openssh ] }
```

Type of cypher. Each name indicates the algorithm and cryptographic parameters that are used. Enter any combination of one or more ciphers in your preferred order of priority (first is highest).

Modes

SSH server system configuration (config-system-ssh-server-*name*)

Usage Guidelines

Ciphers protect the data transported over an SSH connection. Each cipher is an algorithm that encrypts the link.

Only the ciphers that you specify are configured. The rest are ignored.

After using this command, you must restart the SSH server to apply the cipher configuration. For details, see [ssh-server restart](#) on page 336.

Use the **no** form of this command to restore the default order of the configured ciphers.

Examples

The following example configures the SSH server with nine ciphers in descending priority with aes256-ctr as the highest priority:

```
device# configure terminal
device(config)# system
device(config-system)# ssh-server h2
device(config-system-ssh-server-h2)# cipher aes256-ctr aes256-gcm-openssh aes256-cbc
aes192-ctr aes192-cbc aes128-gcm-openssh aes128-ctr aes128-cbc
device(config-system-ssh-server-h2)# client-alive 20
device(config-system-ssh-server-h2)# key rsa 2048
device(config-system-ssh-server-h2)# key-exchange ecdh-sha2-nistp256 curve25519-sha256
diffie-hellman-group-exchange-sha256
device(config-system-ssh-server-h2)# mac hmac-sha2-256 hmac-sha2-512
device(config-system-ssh-server-h2)# vrf vrf-red
device(config-system-ssh-server-h2)# enable
device(config-system-ssh-server-h2)#
```

The following example displays the SSH server configuration that is running currently on the device. In this example, the server uses nine ciphers in descending priority with aes256-ctr as the highest priority:

```
device# show running-config system ssh-server

system
  ssh-server h2
    cipher aes256-ctr aes256-gcm-openssh aes256-cbc aes192-ctr aes192-cbc aes128-gcm-
openssh aes128-ctr aes128-cbc
    client-alive 20
    key rsa 2048
    key-exchange ecdh-sha2-nistp256 curve25519-sha256 diffie-hellman-group-exchange-sha256
    mac hmac-sha2-256 hmac-sha2-512
    vrf vrf-red
    enable
  !
!
device#
```

clear counters interface management

Use the **clear counters interface management** command to clear statistics for the management interface.

Syntax

clear counters interface management *interface-name*

Command Default

Any existing management interface statistics remain uncleared.

Parameters

interface-name

Management interface name. The only supported value is 0 in this release.

Modes

EXEC (#)

Usage Guidelines

Statistics for data for the interfaces are four fields (1 second, 1 minute, 5 minutes, and all time) per direction for bits, bytes, and Megabits per second (Mbps). This data is a "rolling frame" and is not stored.

You must clear the counters if you want to start at zero for more recent statistics.

To view the counters before clearing them, see [show counters interface management](#) on page 232.

Examples

The following example clears statistics for the management interface:

```
device# clear counters interface management 0
device#
```

clear tech-support

Use the **clear tech-support** command to clear all tech support files or a specific tech support file on a device.

Syntax

```
clear tech-support [ file-name ]
```

Command Default

Any existing tech support files remain uncleared.

Parameters

file-name

(Optional) Name of a specific file to clear. For example, TS-420ax-250922T180231.tar.gz or TS-420ax-250922T180231.log. Omitting this keyword clears all available files.

Modes

EXEC (#)

Usage Guidelines

Tech support files are a collection of the output of various **show** commands, log files, and configuration files that you can provide as a system snapshot when requesting help with debugging the device. You create these files via the **system tech-support** command. The command creates two files: a compress TAR file (such as TS-duta-250206T0299.tar.gz) and a similarly named log file (such as TS-duta-250206T0299.log).

If you want to see a list of the existing tech support files, enter the **show tech-support** command.

If you want to generate new tech support files, see [system tech-support](#) on page 364.

Examples

The following example displays a list of the existing tech support files on the device:

```
device# show tech-support
File Name                               Size (bytes)  Created
TS-dutb-251028T160410.log              8037         2025-10-28 16:04:54
TS-dutb-251028T160410.tar.gz          12298874      2025-10-28 16:04:54
device#
```

The following example clears a tech support file named TS-dutb-251028T152455.log on the device:

```
device# clear tech-support TS-dutb-251028T152455.log
Cleared the Tech-Support file: TS-420ax-250922T180231.log
device#
```

The following example clears all available tech support files on the device:

```
device# clear tech-support
WARNING: All the available Techsupport files will be cleared. Do you want to continue?
[y/n]: y
Tech-Support Files cleared
device#
```

clear trace

Use the **clear trace** command to clear old trace logs on a device or old trace logs for a specific microservice on a device.

Syntax

```
clear trace [ microservice-name ]
```

Command Default

Old trace logs remain uncleared.

Parameters

trace *microservice-name*

(Optional) Name of the microservice for which to clear old trace logs. This variable clears the old trace logs for the microservice from the `/tmp/tierra/trace/microservice-name` file. To view the list of microservices on the device, use the **show version** command.

Modes

EXEC (#)

Usage Guidelines

This command clears old trace logs only and does not delete other files such as older .gz files from the directory.

Examples

The following examples clear old trace logs for the chassis-mgr microservice:

```
device# start-shell
[device]# ls -l /tmp/tierra/trace/chassis-mgr/
total 268
-rw----- 1 root root 1791 Mar  7 04:20 chassis-mgr-RASTrace-DError.log
-rw----- 1 root root 269472 Mar  7 04:20 chassis-mgr-RASTrace.log
[admin@device]# exit
exit
device# clear trace chassis-mgr
device# start-shell
[admin@device]# ls -l /tmp/tierra/trace/chassis-mgr/
total 32
-rw----- 1 root root 14398 Mar  7 04:20 chassis-mgr-
RASTrace-2025-03-07T09-20-34.568.log.gz
-rw----- 1 root root 1791 Mar  7 04:20 chassis-mgr-RASTrace-DError.log
-rw----- 1 root root 10216 Mar  7 04:20 chassis-mgr-RASTrace.log
[admin@device]#
```


cli (Extreme ONE OS)

Use the **cli** command to enter CLI system configuration (config-system-cli) mode.

Syntax

cli

Command Default

No CLI system attributes are configured on the device.

Parameters

This command has no keywords or variables.

Modes

System configuration (config-system)

Usage Guidelines

Use this command to enter the mode for entering the commands to configure CLI attributes. For example, whether to unhide RADIUS secret keys, TACACS+ secret keys, and secrets for SNMP communities in **running-config** or **running-state** command output.

This command does not appear in the running configuration unless there are CLI attributes already configured.

Examples

The following example enters CLI system configuration (config-system-cli) mode and configures secrets to be displayed in plain text in **running-config** or **running-state** command output for all modules:

```
device# configure terminal
device(config)# system
device(config-system)# cli
device(config-system-cli)# unhide-secrets all
device(config-system-cli)#
```

The following example displays an example system configuration that is running currently on the device. In this example, the community string for the default SNMP server appears in plain text because secrets are unhidden:

```
device# show running-config system snmp-server

system
  snmp-server DEFAULT
    vrf mgmt-vrf
    enable
```

```
!
  community extremenet
!
!device#
```

client-alive

Use the **client-alive** command to set the interval for sending null packets to the SSH client connection.

Syntax

```
client-alive seconds  
no client-alive
```

Command Default

The interval is zero seconds (disabled).

Parameters

minutes

Duration of the interval. Use an integer from 1 to 14400.

Modes

SSH server system configuration (config-system-ssh-server-*name*)

Usage Guidelines

If the SSH client is not reachable for the specified interval, the connection is terminated gracefully.

To restore the interval to the default value, the **no** form of this command.

Examples

The following example configures and enables a SSH server instance named h2 and sets the interval to 20 minutes:

```
device# configure terminal  
device(config)# system  
device(config-system)# ssh-server h2  
device(config-system-ssh-server-h2)# cipher aes128-gcm-openssh aes256-gcm-openssh  
chacha20-poly1305-openssh  
device(config-system-ssh-server-h2)# client-alive 20  
device(config-system-ssh-server-h2)# key rsa 2048  
device(config-system-ssh-server-h2)# key-exchange ecdh-sha2-nistp256 curve25519-sha256  
diffie-hellman-group-exchange-sha256  
device(config-system-ssh-server-h2)# mac hmac-sha2-256 hmac-sha2-512  
device(config-system-ssh-server-h2)# vrf vrf-red  
device(config-system-ssh-server-h2)# enable  
device(config-system-ssh-server-h2)#
```

The following example displays the SSH server configuration that is running currently on the device:

```
device# show running-config system ssh-server

system
  ssh-server h2
    cipher aes128-gcm-openssh aes256-gcm-openssh chacha20-poly1305-openssh
    client-alive 20
    key rsa 2048
    key-exchange ecdh-sha2-nistp256 curve25519-sha256 diffie-hellman-group-exchange-sha256
    mac hmac-sha2-256 hmac-sha2-512
    vrf vrf-red
    enable
  !
!
device#
```

clock set

Use the **clock set** command to set the system clock date and time on the device.

Syntax

clock set *date-and-time*

Command Default

The clock is set to Coordinated Universal Time (UTC).

Parameters

date-and-time

Date-time string in CCYY-MM-DDTHH:MM:SS format. For example, 2025-11-21T23:00:00.

Modes

EXEC (#)

Usage Guidelines

To display the current system date, time, and time zone setting on the device, see [show clock](#) on page 231.

To set the system time zone on the device, see [clock timezone](#) on page 78.

Examples

The following example configures the clock date and time on the device:

```
device# clock set 2025-11-21T23:00:00
device# show clock
2025-11-21 23:00:00 UTC +0000
device#
```

clock timezone

Use the **clock timezone** command to configure the system time zone on the device.

Syntax

```
clock timezone region/city  
no clock timezone
```

Command Default

The system time zone is set to Coordinated Universal Time (UTC).

Parameters

```
timezone region/city
```

Supported time zone. You must include the slash (/). For example, enter America/Anchorage or Africa/Algiers. Enter the **clock timezone** command without keywords or variables to see the available values.

Modes

System configuration (config-system)

Usage Guidelines

You must reboot the device after changing the time zone to make the change take effect.

To set the system time on the device, see [clock set](#) on page 77.

Use the **no** form of this command to reset the time zone to the default value.

Examples

The following example displays the available time zones:

```
device(config-system)# clock timezone  
  
TIMEZONE  Time zone  
Africa/Abidjan  
Africa/Accra  
Africa/Addis_Ababa  
Africa/Algiers  
Africa/Asmara  
Africa/Bamako  
Africa/Bangui  
Africa/Banjul  
Africa/Bissau  
Africa/Blantyre  
Africa/Brazzaville
```

```
Africa/Bujumbura
Africa/Cairo
Africa/Casablanca
Africa/Ceuta
Africa/Conakry
Africa/Dakar
Africa/Dar_es_Salaam
Africa/Djibouti
Africa/Douala
Africa/El_Aaiun
Africa/Freetown
Africa/Gaborone

...

device#
```

The following example configures the system clock time zone:

```
device# configure terminal
device(config)# system
device(config-system)# clock timezone Asia/Kolkata
device(config-system)#
```

community (SNMP server system configuration)

Use the **community** command to configure the SNMP server community string to be used for SNMPv1 and SNMPv2c protocol access.

Syntax

community *community-string*

no community *community-string*

Command Default

The Simple Network Management Protocol (SNMP) server on the device has no community string configured.

Parameters

community *community-string*

String that is needed to gain access to SNMP. Use 2 to 16 characters.

Modes

SNMP server system configuration (config-system-snmp-server)

Usage Guidelines

The SNMP community string is similar to a user ID or password that allows access to device statistics. SNMP community strings are used only by devices that support the SNMPv1 and SNMPv2c protocols.



Note

If you want to use SNMPv3, you must use the [user \(Extreme ONE OS SNMP server system configuration\)](#) on page 385 command (instead of the **community** command) to configure authentication. SNMPv3 uses username and password authentication along with an encryption key.

All the communities configured will have READ all permission. No community will have WRITE permission.

Use the **no** form of this command to remove the SNMP server community string.

Examples

The following example configures an SNMP agent with the trap destination information with an SNMP server community string attached to it. In this example,

the trap host uses Ethernet interface 0/1 as the source interface, which determines the source IP address to be used while sending traps to the trap host:

```
device# configure terminal
device(config)# system
device(config-system)# snmp-server DEFAULT
device(config-system-snmp-server)# community extremero
device(config-system-snmp-server)# vrf mgmt-vrf
device(config-system-snmp-server-vrf-mgmt-vrf)# enable
device(config-system-snmp-server-vrf-mgmt-vrf)# host 10.10.10.1 v3user
device(config-system-snmp-server-vrf-mgmt-vrf-host-10.10.10.1-****)# udp-port 200
device(config-system-snmp-server-vrf-mgmt-vrf-host-10.10.10.1-****)# source-interface
ethernet 0/1
device(config-system-snmp-server-vrf-mgmt-vrf-host-10.10.10.1-****)#
```

The following example displays the SNMP server configuration that is running currently on the device:

```
device# show running-config system snmp-server

system
  snmp-server DEFAULT
  vrf mgmt-vrf
  enable
  host 10.10.10.1 /LMRbduZHP
  udp-port 200
  source-interface ethernet 0/1.0
  !
  !
  community-hashed dXsYmo05bQ
  !
!
device#
```



Tip

The SNMP server community string displays as a hash in the **show running-config system snmp-server** command output. If you want to unhide the SNMP server community string so that it displays as plain text, see [unhide-secrets](#) on page 381.

connector (Extreme ONE OS)

Use the **connector** command to configure a connector and enter Connector configuration (config-hardware) mode.

Syntax

connector *interface-name*

Command Default

A connector is configured with interface 0/1.

Parameters

interface-name

Port on the device that supports breakout mode. Use the *slot/port* format. For example, 0/1.

Modes

Hardware configuration (config-hardware)

Usage Guidelines

Use this command to enter the configuration mode for specifying a breakout mode for any port that supports the breakout feature. In this mode, you can break out the port into four 10G or 25G breakout interfaces.

To specify a breakout mode, see [breakout](#) on page 51.

Examples

The following example configures a connector on the device and a breakout mode for the connector:

```
device# configure terminal
device(config)# hardware
device(config-hardware)# connector 0/1
device(config-connector-0/1)# breakout 4x10g
device(config-connector-0/1)#
```

The following example displays the hardware configuration that is running currently on the device:

```
device# show running-config

! Firmware: 10.2.1.0, Application(s): 22.2.1.0
! Date and Time: Tue, 2025-12-09 15:16:13 EST
...
```

```
device(config)# show running-config hardware
```

```
hardware
  connector 0/1
  breakout 4x10g
!
device#
```

console (logging system configuration)

Use the **console** command to display system logs in CLI sessions directly to the console.

Syntax

```
console [ severity severity-name ]  
no console [ severity severity-name ]
```

Command Default

System logging directly to the console is enabled. The severity level is set to “info.”

Parameters

severity-name

(Optional) Level of severity. Enter one of the following values: emergency, alert, critical, error, warning, notice, info, or debug.

Modes

Logging system configuration (config-system-logging)

Usage Guidelines

Use this command to display system logs in CLI sessions to the console. Console logging is what displays if you connect to the CLI directly via the physical console port.

If the “error” level is configured, only the logs with a level of “error” or a higher level such as “critical” or “alert” are displayed.

To display system logs in CLI sessions indirectly using Telnet or SSH, see [monitor \(logging system configuration\)](#) on page 168.

Use the **no** form of this command to restore the default behavior.

Examples

The following example configures the system log to display in a CLI session when you are connected directly to the console:

```
device# configure terminal  
device(config)# system  
device(config-system)# logging  
device(config-system-logging)# console severity error  
device(config-system-logging)#
```

contact (Extreme ONE OS)

Use the **contact** command to configure an administrative contact for a deployed device.

Syntax

```
contact contact-info  
no contact
```

Command Default

The device has no administrative contact configured.

Parameters

contact-info
Contact string. Use 1 to 255 alphanumeric characters.

Modes

System configuration (config-system)

Usage Guidelines

Use this command to configure the network administrator's contact details. The network administrator is responsible for addressing problems impacting the network.

To delete the contact details, use the **no** form of this command.

Examples

The following example configures the network administrator's contact details:

```
device# configure terminal  
device(config)# system  
device(config-system)# contact Hyderabad  
device(config-system)#
```

The following example displays the network administrator's contact configuration that is running currently on the device:

```
device# show running-config system contact  
  
system  
    contact Hyderabad  
!  
device#
```

copy backup-config file

Use the **copy backup-config file** command to copy a device's backup configuration to a local or remote file.

Syntax

```
copy backup-config file { disk-file-path | usb-file-path | remote-file-path } [ source-ip ip-address ] [ vrf vrf-name ]
```

Parameters

disk-file-path

Disk file path in format `disk://config-file/file-name`.

usb-file-path

USB file path in format `usb://file-name`.

remote-file-path

Remote server file path in format `scp://username[:password]@host:[port]/filepath`.

Only valid unicast IPv4 and IPv6 addresses are supported. Broadcast IPv4 addresses and multicast IP addresses are not supported.

source-ip *ip-address*

(Optional) Source IPv4 or IPv6 address. Applies only to remote server file paths.

vrf *vrf-name*

(Optional) VRF name. Applies only to remote server file paths.

Modes

EXEC (#)

Usage Guidelines

You can copy the backup configuration to a file. You can use this file later to restore the backup configuration, which you can then use to return the device to its normal state. The backup configuration is useful for troubleshooting or system recovery if the device is reset unintentionally or the configuration database becomes corrupted.

The device maintains only one backup configuration at a time. When you create a backup configuration, any existing backup configuration is overwritten. This ensures that only the most recent user-created backup configuration is preserved.

A backup configuration is stored in a plain-text ASCII file that resembles the output of the **show running-config** command. If you want to view the current file, see [show backup-config](#) on page 221.

You can save the backup configuration to a file on the device's internal disk, a USB device attached to the device, or a Secure Copy (SCP) server:

- When the file destination is a USB drive, make sure that access to the drive is enabled on the device. For details, see [usb enable \(Extreme ONE OS\)](#) on page 384.
- When the file destination is a disk drive, check the disk types via the **ls** command (the wrong disk type produces an error). For details, see [ls \(Extreme ONE OS\)](#) on page 153.
- When the file destination is a remote location, make sure that the IP address is reachable and that you are using the proper login credentials.

Examples

The following example copies the backup configuration to the specified file on a disk drive:

```
device# copy backup-config file disk://config-file/runcfg.txt
Copy backup-config to file is successful
device#
```

The following example copies the backup configuration to the specified file on a USB drive:

```
device# copy backup-config file usb://abc.conf
Copy backup-config to file is successful
device#
```

The following example copies the backup configuration to the specified remote file:

```
device# copy backup-config file scp://exampleuser@1.2.3.4:/home/example source-ip 5.6.7.8
vrf red
Copy backup-config to file is successful
device#
```

copy default-config running-config

Use the **copy default-config running-config** command to clear the running configuration.

Syntax

copy default-config running-config

Parameters

This command has no keywords or variables.

Modes

EXEC (#)

Usage Guidelines

After you run this command, the system reboots with the management interface configuration.

Examples

The following example replaces the running configuration with the default configuration:

```
device# copy default-config running-config
This operation will modify your running configuration.
WARN: system will be rebooted to have configuration changes to take effect!
Do you want to continue? [y/n]: y
device#
device# Connection to 10.10.0.1 closed by remote host.
Connection to 10.37.35.1 closed.
```


copy file backup-config

Use the **copy file backup-config** command to copy the contents of a configuration file from a local or remote location to the backup configuration.

Syntax

```
copy file { disk-file-path | usb-file-path | remote-file-path } backup-  
config [ apply-on-next-reboot ]
```

Command Default

No backup configuration exists on the device.

Parameters

disk-file-path

Disk file path in format `disk://config-file/file-name`.

usb-file-path

USB file path in format `usb://file-name`.

remote-file-path

Remote server file path in format `scp://username[:password]@host:[port]/filepath`.

Only valid unicast IPv4 and IPv6 addresses are supported. Broadcast IPv4 addresses and multicast IP addresses are not supported.

apply-on-next-reboot

(Optional) Applies the backup configuration at the next reboot.

Modes

EXEC (#)

Usage Guidelines

You can copy a configuration file to the backup configuration and then use the backup configuration to restore the device to its normal state later. The backup configuration is useful for troubleshooting or system recovery if the device is reset unintentionally or the configuration database becomes corrupted.

This command has an option to apply the backup configuration to the running configuration at the next reboot regardless of the reboot reason.

The device maintains only one backup configuration at a time. When you create a new backup configuration, any existing backup configuration is overwritten. This ensures that only the most recent user-created backup configuration is preserved.

A backup configuration is stored in a plain-text ASCII file that resembles the output of the **show running-config** command. If you want to view the current file, see [show backup-config](#) on page 221.

You can copy a configuration file from the device's internal disk, from a USB drive attached to the device, or from a Secure Copy (SCP) server:

- When the file is on a USB drive, make sure that access to the drive is enabled on the device. For details, see [usb enable \(Extreme ONE OS\)](#) on page 384.
- When the file is on a disk drive, check the disk types via the **ls** command (the wrong disk type produces an error). For details, see [ls \(Extreme ONE OS\)](#) on page 153.
- When the file is at a remote location, make sure that the IP address is reachable and that you are using the proper login credentials.

If you used the **apply-on-next-reboot** keyword, you can monitor the status or progress of the automated backup configuration replay after the device reboots. For details, see [show backup-config replay-status](#) on page 223.

Examples

The following example copies a configuration file from a disk drive to the backup configuration:

```
device# copy file disk://config-file/abc.conf backup-config
Backup-config created successfully
device#
```

The following example copies a configuration file from a USB drive to the backup configuration:

```
device# copy file usb://abc.conf backup-config
Backup-config created successfully
device#
```

The following example copies a configuration file from a remote location to the backup configuration:

```
device# copy file scp://exampleuser@1.1.1.1:/home/bob backup-config
exampleuser@1.1.1.1's password: *****
Backup-config created successfully
device#
```

The following example copies a configuration file from a remote location to the backup configuration and also applies the backup configuration to the running configuration at the next reboot:

```
device# copy file scp://exampleuser@1.1.1.1:/home/bob backup-config apply-on-next-reboot
exampleuser@1.1.1.1's password: *****
Backup-config created successfully and will be applied on next reboot
device#
```

copy file running-config

Use the **copy file running-config** command to copy contents of a configuration file from the specified location to the running configuration.

Syntax

```
copy file { disk-file-path | usb-file-path | remote-file-path } running-  
config [ source-ip ip-address ] [ vrf vrf-name ]
```

Parameters

disk-file-path

Disk file path in format `disk://config-file/file-name`.

usb-file-path

USB file path in format `usb://file-name`.

remote-file-path

Remote server file path in format `scp://username[:password]@host:[port]/filepath`.

Only valid unicast IPv4 and IPv6 addresses are supported. Broadcast IPv4 addresses and multicast IP addresses are not supported.

source-ip *ip-address*

(Optional) Source IPv4 or IPv6 address. Applies only to remote server file paths.

vrf *vrf-name*

(Optional) VRF name. Applies only to remote server file paths.

Modes

EXEC (#)

Usage Guidelines

You can copy a configuration file from the device's internal disk, from a USB device attached to the device, or from a Secure Copy (SCP) server:

- When the file source is a USB drive, make sure that access to the drive is enabled on the device. For details, see [usb enable \(Extreme ONE OS\)](#) on page 384.
- When the file source is a disk drive, check the disk types via the **ls** command (the wrong disk type produces an error). For details, see [ls \(Extreme ONE OS\)](#) on page 153.
- When the file source is a remote location, make sure that the IP address is reachable and that you are using the proper login credentials.

Examples

The following example shows how to copy a configuration from a disk drive to the current running configuration:

```
device# copy file disk://config-file/runncfg.txt running-config
Applying management interface IP address configuration
..
device#
```

copy file file

Use the **copy file file** command to copy a disk or USB file to a remote server or vice versa.

Syntax

Syntax

```
copy file { disk-file-path | usb-file-path } file { disk-file-path | usb-file-path } [ source-ip ip-address ] [ vrf vrf-name ]
```

Parameters

disk-file-path

Disk file path in format `disk://config-file/file-name`.

usb-file-path

USB file path in format `usb://file-name`.

remote-file-path

Remote server file path in format `scp://username[:password]@host:[port]/filepath`.

Only valid unicast IPv4 and IPv6 addresses are supported. Broadcast IPv4 addresses and multicast IP addresses are not supported.

source-ip *ip-address*

(Optional) IPv4 address or IPv6 address. Applies only to remote server file paths.

vrf *vrf-name*

(Optional) Name of the VRF. Applies only to remote server file paths.

Modes

EXEC (#)

Usage Guidelines

When the file is from USB, make sure that USB is enabled on the device.

When the file is from disk, check the disk types from the **dir** command (the wrong disk types will produce an error).

When the file is remote, make sure that the IP address is reachable and that you are using the proper credentials.

Copying a file from one remote server to another remote server is not supported.

Examples

The following example copies a file from the USB to a remote location:

```
device# copy file usb://config-file/test file disk://config-file/newtest
Copy file to a file is successful
device#
```

copy running-config backup-config

Use the **copy running-config backup-config** command to copy the configuration that is running currently on the device to the backup configuration.

Syntax

```
copy running-config backup-config [ apply-on-next-reboot ]
```

Parameters

apply-on-next-reboot

(Optional) Applies the backup configuration at the next reboot.

Modes

EXEC (#)

Usage Guidelines

You can copy the running configuration to the backup configuration and then use the backup configuration to restore the device to its normal state later. The backup configuration is useful for troubleshooting or system recovery if the device is reset unintentionally or the configuration database becomes corrupted.

This command has an option to apply the backup configuration to the running configuration at the next reboot regardless of the reboot reason.

The device maintains only one backup configuration at a time. When you create a backup configuration, any existing backup configuration is overwritten. This ensures that only the most recent user-created backup configuration is preserved.

The backup configuration is stored in a plain-text ASCII file that resembles the output of the **show running-config** command. If you want to view the current file, see [show backup-config](#) on page 221.

If you used the `apply-on-next-reboot` keyword, you can monitor the status or progress of the automated backup configuration replay after the device reboots. For details, see [show backup-config replay-status](#) on page 223.

Examples

The following example copies the configuration that is running currently on the device to the backup configuration:

```
device# copy running-config backup-config
Backup-config created successfully
device#
```

The following example copies the configuration that is running currently on the device to the backup configuration and also applies the backup configuration to the running configuration at the next reboot:

```
device# copy running-config backup-config apply-on-next-reboot
Backup-config created successfully and will be applied on next reboot
device#
```


copy running-config file

Use the **copy running-config file** command to copy the running configuration to the specified file to create a backup.

Syntax

```
copy running-config file { disk-file-path | usb-file-path | remote-file-path } [ source-ip ip-address ] [ vrf vrf-name ]
```

Parameters

disk-file-path

Disk file path in format `disk://config-file/file-name`.

usb-file-path

USB file path in format `usb://file-name`.

remote-file-path

Remote server file path in format `scp://username[:password]@host:[port]/filepath`.

Only valid unicast IPv4 and IPv6 addresses are supported. Broadcast IPv4 addresses and multicast IP addresses are not supported.

source-ip *ip-address*

(Optional) Source IPv4 or IPv6 address. Applies only to remote server file paths.

vrf *vrf-name*

(Optional) VRF name. Applies only to remote server file paths.

Modes

EXEC (#)

Usage Guidelines

You can copy the running configuration to the device's internal disk, to a USB device attached to the device, or to a Secure Copy (SCP) server:

- When the file destination is a USB drive, make sure that access to the drive is enabled on the device. For details, see [usb enable \(Extreme ONE OS\)](#) on page 384.
- When the file destination is a disk drive, check the disk types via the **ls** command (the wrong disk type produces an error). For details, see [ls \(Extreme ONE OS\)](#) on page 153.
- When the file destination is a remote location, make sure that the IP address is reachable and that you are using the proper login credentials.

Examples

The following example copies the running configuration to the specified disk file:

```
device# copy running-config file disk://config-file/runcfg.txt
Copy running-config to file is successful
device#
```

crypto

Use the **crypto** command to configure the cryptographic algorithm for a key in the keychain.

Syntax

```
crypto { aes_128_cmac_96 | crypto_none | hmac_md5 | hmac_sha_1 |  
        hmac_sha_1_12 | hmac_sha_1_20 | hmac_sha_1_96 | hmac_sha_256 | md5 |  
        sha_1 }
```

Command Default

No cryptographic algorithm for a key in the keychain is configured.

Parameters

```
aes_128_cmac_96 | crypto_none | hmac_md5 | hmac_sha_1 |  
hmac_sha_1_12 | hmac_sha_1_20 | hmac_sha_1_96 | hmac_sha_256 | md5  
| sha_1
```

Hash-based message authentication code (HMAC) algorithm for the key.

Modes

Keychain key ID configuration (config-key-*key-id*)

Examples

The following example configures the cryptographic algorithm for a key with key ID 33:

```
device# configure terminal  
device(config)# keychain key1  
device(config-keychain-key1)# key-id 33  
device(config-keychain-key1-key-33)# crypto md5  
device(config-keychain-key1-key-33)#
```

The following example displays the keychain configuration that is running currently on the device. In this example, the keychain uses a single key, which uses the MD5 cryptographic algorithm:

```
device# show running-config keychain  
  
keychain key1  
    key-id 33  
    crypto md5  
    !  
    !  
device#
```

dns (Extreme ONE OS)

Use the **dns** command to enter DNS configuration (config-system-dns) mode.

Syntax

dns

Parameters

This command has no keywords or variables.

Modes

System configuration (config-system)

Usage Guidelines

Use this command to enter the mode for configuring Domain Name System (DNS) server settings such as the domain name of the device and the IP address of the domain name server. Configuring a domain name and a domain name server for your device makes contacting other devices on your network easier.

This command does not appear in the running configuration unless there are submode parameters configured.

Examples

The following example enters DNS configuration (config-system-dns) mode and configures domain names and domain name servers on the device:

```
device# configure terminal
device(config)# system
device(config-system)# dns
device(config-system-dns)# domain-name mydevice1.com
device(config-system-dns)# domain-name mydevice2.com
device(config-system-dns)# name-server 1.1.1.1
device(config-system-dns)# name-server 2.2.2.2
device(config-system-dns)#
```

The following example displays the DNS configuration that is running currently on the device:

```
device# show running-config system dns

system
  dns
    domain-name mydevice1.com
    domain-name mydevice2.com
    name-server 1.1.1.1
    name-server 2.2.2.2
  !
!
device#
```

domain-name (Extreme ONE OS)

Use the **dns** command to set a Domain Name System (DNS) domain name of the device.

Syntax

domain-name *domain-name*

no domain-name *domain-name*

Parameters

domain-name

Domain name. Use 1 to 253 alphanumeric characters.

Modes

DNS configuration (config-system-dns)

Usage Guidelines

Use this command to configure the DNS setting that specifies a domain name of the device. This is a default domain name that the device uses to complete unqualified hostnames (names without a dotted-decimal domain name).

Configuring a domain name and a domain name server for your device makes contacting other devices on your network easier.

You can specify one or more domain names.

Use the **no** form of this command to remove a domain name from the device.

Examples

The following example specifies domains named mydevice1.com and mydevice2.com for the device:

```
device# configure terminal
device(config)# system
device(config-system)# dns
device(config-system-dns)# domain-name mydevice1.com
device(config-system-dns)# domain-name mydevice2.com
device(config-system-dns)# name-server 1.1.1.1
device(config-system-dns)# name-server 2.2.2.2
device(config-system-dns)#
```

The following example displays the DNS configuration that is running currently on the device. In this example, the device uses two domain names:

```
device# show running-config system dns

system
```

```
dns
  domain-name mydevice1.com
  domain-name mydevice2.com
  name-server 1.1.1.1
  name-server 2.2.2.2
!
device#
```

enable (gRPC server configuration)

Use the **enable** command to enable the gRPC server instance and allow northbound access to the gNMI/gNOI RPCs.

Syntax

enable
no enable

Command Default

The gRPC server instance is disabled.

Parameters

This command has no keywords or variables.

Modes

gRPC server system configuration (config-system-grpc-server-*name*)

Usage Guidelines

You must disable the gRPC server instance before using this command (or any other command) to alter its configuration. Then, you must re-enable the instance to restore its functionality. To disable or enable a gRPC server instance, see [enable \(gRPC server configuration\)](#) on page 103.

If you disable the gRPC server instance, ongoing subscriptions are disrupted (depending on the client side timeout configuration).

The combination of port and VRF should be unique across gRPC server instances.

A TCP listening port must be configured and valid for the gRPC server instance.

The certificate authority file ca-list.pem cannot be empty.

The certificate must exist in the system before you run this command. To do so, see [certificate-manager import app-certificate](#) on page 61.

To disable the gRPC server instance, use the **no** form of this command.

Examples

The following example configures a gRPC server instance. In this example, the instance is enabled after all configuration:

```
device# configure terminal
device(config)# system
```

```
device(config-system)# grpc-server mygrpcserver
device(config-system-grpc-server-mygrpcserver)# certificate-id test_cert_id
device(config-system-grpc-server-mygrpcserver)# port 9340
device(config-system-grpc-server-mygrpcserver)# mutual-tls
device(config-system-grpc-server-mygrpcserver)# vrf vrf-blue
device(config-system-grpc-server-mygrpcserver)# enable
device(config-system-grpc-server-mygrpcserver)#
```

The following example displays the gRPC server instance configuration that is running currently on the device. In this example, the instance is enabled:

```
device# show running-config system grpc-server

system
  grpc-server mygrpcserver
    certificate-id test_cert_id
    port 9340
    mutual-tls
    vrf vrf-blue
    enable
!
device#
```


enable (SSH server system configuration)

Use the **enable** command to enable or disable an SSH server instance.

Syntax

```
enable  
no enable
```

Command Default

SSH is running on the mgmt-vrf by default.

Parameters

This command has no keywords or variables.

Modes

SSH server system configuration (config-system-ssh-server-*name*)

Usage Guidelines

All existing connections with the device are lost when you shut down the SSH server instance.

To disable the SSH server instance, the **no** form of this command.

Examples

The following example enables a SSH server instance named h2:

```
device# configure terminal  
device(config)# system  
device(config-system)# ssh-server h2  
device(config-system-ssh-server-h2)# enable  
device(config-system-ssh-server-h2)#
```

The following example displays the SSH server configuration that is running currently on the device:

```
device# show running-config system ssh-server  
  
system  
  ssh-server h2  
    cipher aes256-ctr aes256-gcm-openssh aes256-cbc aes192-ctr aes192-cbc aes128-gcm-  
openssh aes128-ctr aes128-cbc  
    client-alive 20  
    key rsa 2048  
    key-exchange ecdh-sha2-nistp256 curve25519-sha256 diffie-hellman-group-exchange-sha256  
    mac hmac-sha2-256 hmac-sha2-512  
    vrf vrf-red  
    enable
```

```
!  
!  
device#
```

enable (NTP system configuration)

Use the **enable** command to enable Network Time Protocol (NTP).

Syntax

```
enable  
no enable
```

Command Default

NTP is disabled.

Parameters

This command has no keywords or variables.

Modes

NTP system configuration (config-system-ntp)

Usage Guidelines

Use this command to enable NTP on the device. You use NTP to synchronize the clocks over your network.

The NTP service is a global network of public and private servers that provides precise time synchronization to devices worldwide. It works by exchanging time data packets among client and server devices.

NTP maintains uniform time across all devices in a network. The NTP commands let you configure an external time server to synchronize all local clocks in a network.

To keep the time in your network current, each device should have its time synchronized with at least one external NTP server.

Examples

The following example enables NTP and then configures NTP to synchronize the system date and time on the device. In this example, a VRF named red-vrf is used to route NTP communications, IPv4 address 2.2.2.2 points to an NTP server to acquire all system time information, and IPv4 address 3.3.3.3 is a peer with which the local device synchronizes its system time:

```
device# configure terminal  
device(config)# system  
device(config-system)# ntp  
device(config-system-ntp)# enable  
device(config-system-ntp)# vrf red-vrf
```

```
device(config-system-ntp)# server 2.2.2.2
device(config-system-ntp)# peer 3.3.3.3
device(config-system-ntp)#
```

The following example displays the NTP configuration that is running currently on the device:

```
device# show running-config system ntp

system
  ntp
    server 2.2.2.2
    peer 3.3.3.3
    vrf red-vrf
    enable
  !
!
device#
```

The following example displays the NTP configuration that is running currently on the device. This is an example of a more complex configuration that uses multiple servers and multiple peers:

```
device# show running-config system ntp

system
  ntp
    enable
    server 0.in.pool.ntp.org
    server 1.in.pool.ntp.org
    server 2.in.pool.ntp.org
    server 1.pool.ntp.org
    server 3.in.pool.ntp.org
    server 2.pool.ntp.org
    server 2.2.2.2
    peer 10.in.pool.ntp.org
    peer 11.in.pool.ntp.org
    peer 12.in.pool.ntp.org
    peer 21.pool.ntp.org
    peer 23.in.pool.ntp.org
    peer 22.pool.ntp.org
    peer 3.3.3.3
  !
!
device#
```

enable (SNMP server VRF system configuration)

Use the **enable** command to enable the SNMP server on the device.

Syntax

enable
no enable

Command Default

The Simple Network Management Protocol (SNMP) server on the device is disabled.

Parameters

This command has no keywords or variables.

Modes

SNMP server VRF system configuration (config-system-snmp-server-vrf-*vrf-name*)

Usage Guidelines

Use the **no** form of this command to disable the SNMP server.

Examples

The following example configures an SNMP server community string and enables the SNMP server:

```
device# configure terminal
device(config)# system
device(config-system)# snmp-server DEFAULT
device(config-system-snmp-server)# community extremero
device(config-system-snmp-server)# vrf mgmt-vrf
device(config-system-snmp-server-vrf-mgmt-vrf)# enable
device(config-system-snmp-server-vrf-mgmt-vrf)#
```

The user can also enable it in user-defined vrfs.

The following example displays the SNMP server configuration that is running currently on the device. In this example, the SNMP server is enabled:

```
device# show running-config system snmp-server

system
  snmp-server DEFAULT
  vrf mgmt-vrf
    enable
  !
  community-hashed dXsYmo05bQ
  !
```

```
!  
device#
```

**Tip**

The SNMP server community string displays as a hash in the **show running-config system snmp-server** command output. If you want to unhide the SNMP server community string so that it displays as plain text, see [unhide-secrets](#) on page 381.

enable (Telnet server system configuration)

Use the **enable** command to enable or disable a Telnet server instance.

Syntax

```
enable  
no enable
```

Command Default

The specified Telnet server instance is disabled.

Parameters

This command has no keywords or variables.

Modes

Telnet server system configuration (config-system-telnet-server-*name*)

Examples

The following example enables a Telnet server instance named telnet1:

```
device# configure terminal  
device(config)# system  
device(config-system)# telnet-server telnet1  
device(config-system-telnet-server-telnet1)# vrf red-vrf  
device(config-system-telnet-server-telnet1)# enable  
device(config-system-telnet-server-telnet1)#
```

The following example displays the Telnet server configuration that is running currently on the device. In this example, a Telnet server instance named telnet1 is enabled:

```
device# show running-config system telnet-server  
  
system  
  telnet-server telnet1  
    vrf red-vrf  
    enable  
  !  
!  
device#
```

event (AAA accounting system configuration)

Use the **event** command to configure the event logging for AAA accounting on a device.

Syntax

```
event { login | command }  
no event { login | command }
```

Command Default

No accounting event logging for Authentication, Authorization, and Accounting (AAA) is configured on the device.

Parameters

login

Logs all EXEC login sessions (console, Telnet, SSH, and gNMI) on the device.

command

Logs the

Logs the EXEC (#) mode CLI commands that users issue on the device.

Modes

AAA accounting system configuration (config-system-aaa-accounting)

Usage Guidelines

AAA accounting capture the actions performed while users are accessing the network. This command specifies the event types that will be part of the AAA accounting information on a device.

Examples

The following example configures AAA on the device. This example enables the collection of accounting data for login events and command events:

```
device# configure terminal  
device(config)# system  
device(config-system)# aaa  
device(config-system-aaa)# accounting  
device(config-system-aaa-accounting)# accounting-method tacacs+  
device(config-system-aaa-accounting)# event login  
device(config-system-aaa-accounting)# event command  
device(config-system-aaa-accounting)#
```


The following example displays the AAA configuration for the TACACS+ server group that is running currently on the device. The output includes the settings for logging AAA accounting events using TACACS+:

```
show running-config system aaa accounting

system
  aaa
    accounting
      accounting-method tacacs+
      event command
      event login
    !
  !
!
device#
```

expiry-alert (AAA authentication password attributes configuration)

Use the **expiry-alert** command to enter AAA authentication password attributes expiry system configuration (config-system-aaa-authentication-password-attributes-expiry) mode.

Syntax

```
expiry-alert  
no expiry-alert
```

Command Default

The device has no AAA authentication password expiry (expiration) alert periods configured.

Parameters

Use this command to enter the mode that is needed for configuring the password expiry alert periods for AAA authentication.

In this mode, use the following subcommands to configure the user password expiry alert periods:

info *days*

Info severity log expiry alert period. The range is 1 to 90.

Use the **no info** command to remove this expiry alert period.

minor *days*

Minor severity log expiry alert period. The range is 1 to 90.

Use the **no minor** command to remove this expiry alert period.

major *days*

Major severity log expiry alert period. The range is 1 to 90.

Use the **no major** command to remove this expiry alert period.

critical *days*

Critical severity log expiry alert period. The range is 1 to 90.

Use the **no critical** command to remove this expiry alert period.

Modes

AAA authentication password attributes system configuration (config-system-aaa-authentication-password-attributes)

Usage Guidelines

The info severity alert period must be longer than the minor, major, and critical alerts periods. The valid order for these periods is info > minor > major > critical.

This command does not appear in the running configuration unless there are password expiry alert periods configured.

Use the **no** form of this command to remove all password expiry alert periods.

Examples

The following example enters AAA authentication password attributes expiry system configuration (config-system-aaa-authentication-password-attributes-expiry) mode and configures the password expiry alert periods:

```
device# configure terminal
device(config)# system
device(config-system)# aaa
device(config-system-aaa)# authentication
device(config-system-aaa-authentication)# password-attributes
device(config-system-aaa-authentication-password-attributes)# expiry-alert
device(config-system-aaa-authentication-password-attributes-expiry)# info 15
device(config-system-aaa-authentication-password-attributes-expiry)# minor 10
device(config-system-aaa-authentication-password-attributes-expiry)# major 5
device(config-system-aaa-authentication-password-attributes-expiry)# critical 3
device(config-system-aaa-authentication-password-attributes-expiry)#
```

The following example displays the AAA configuration that is running currently on the device. In this example, all four password expiry alert periods are set:

```
device# show running-config system aaa

system
  aaa
    authentication
      admin-user admin password-hashed
      $6$10FtgLVtzOEi7jhl$NALdarg9FfowUfWTZUaOnyXG4mehd3cdg3jdGWBFE6uCCSFShPxstb6Mpcd3UJoA16Up19
      6GZ1pn5ilGURdte.
      password-attributes
        expiry-alert
          info 15
          minor 10
          major 5
          critical 3
    !
  !
!
!
device#
```

expiry-alert (certificate manager configuration)

Use the **expiry-alert** command to enter Certificate manager expiry alert system configuration (config-system-cert-mgr-exp) mode.

Syntax

expiry-alert

Command Default

The device has no certificate expiry (expiration) alert periods configured.

Parameters

Use this command to enter the necessary mode for configuring the certificate expiry alert periods, which print the log during the specified severity days.

This command does not appear in the running configuration unless there are certificate expiry alert periods configured.

In this mode, use the following subcommands to configure the certificate expiry alert periods:

info *days*

Info severity log expiry alert period. The range is 1 to 90.

Use the **no info** command to remove this expiry alert period.

minor *days*

Minor severity log expiry alert period. The range is 1 to 90.

Use the **no minor** command to remove this expiry alert period.

major *days*

Major severity log expiry alert period. The range is 1 to 90.

Use the **no major** command to remove this expiry alert period.

critical *days*

Critical severity log expiry alert period. The range is 1 to 90.

Use the **no critical** command to remove this expiry alert period.

Modes

Certificate manager system configuration (config-system-cert-mgr)

Usage Guidelines

Certificate expiry alerts are generated for application certificates and certificate authority (CA) certificates.

The info severity alert period must be longer than the minor, major, and critical alert periods. The valid alert order is info > minor > major > critical.

Examples

The following example enters Certificate manager expiry alert system configuration (config-system-cert-mgr-exp) mode and configures all four certificate expiry alert periods:

```
device# configure terminal
device(config)# system
device(config-system)# certificate-manager
device(config-system-cert-mgr)# expiry-alert
device(config-system-cert-mgr-exp)# critical 30
device(config-system-cert-mgr-exp)# major 60
device(config-system-cert-mgr-exp)# minor 80
device(config-system-cert-mgr-exp)# info 90
device(config-system-cert-mgr-exp)#
```

The following example displays the certificate manager configuration that is running currently on the device. In this example, all four certificate expiry alert periods are configured:

```
device# show running-config system certificate-manager

system
  certificate-manager
    expiry-alert
      critical 30
      major 60
      minor 80
      info 90
    !
  !
!
device#
```

facility

Use the **facility** command to configure the severity level with respect to facilities on a device.

Syntax

```
facility facility-name severity severity-name  
no facility
```

Command Default

The severity level for each facility is set to “info.”

Parameters

facility-name

(Optional) Name of the facility (microservice). This can be a base Extreme ONE OS service or a service for an application such as Extreme ONE OS Switching. For example, interface-mgr or mlag.

The base Extreme ONE OS services are: api-gw, chassis-mgr, config-db, ingress-gw, interface-mgr, msg-bus, security, snmp-agent, state-db, base-svc, and monitor-svc.

The Extreme ONE OS Switching services are: arpnd, bfd, bgp, classifiers, fwd-hal, lacp, lldp, mftm, mlag, sr-api-gw, telegraf, and uftm.

severity-name

Level of severity. Enter one of the following values: emergency, alert, critical, error, warning, notice, info, or debug.

Modes

Logging system configuration (config-system-logging)

Usage Guidelines

If the “error” level is configured, only the logs with a level of “error” or a higher level such as “critical” or “alert” are displayed.

Use the **no** form of this command to restore the default behavior.

Examples

The following example configures logging so that “error” level (or higher) logs are displayed for the chassis-mgr service on CLI sessions through Telnet or SSH:

```
device# configure terminal  
device(config)# system
```

```
device(config-system)# logging
device(config-system-logging)# facility
device(config-system-logging-facility)# service api-gw severity error
device(config-system-logging-facility)#
```

factory-reset

Use the **factory-reset** command to reset the device to factory settings.

Syntax

factory-reset

Parameters

This command has no keywords or variables.

Modes

EXEC (#)

Usage Guidelines

This command resets the device to its default factory settings by removing all current configurations, previous images, log files and so on. After completing the reset, the device reboots with the default factory configuration.

Examples

The following example resets the device to its factory settings:

```
device# factory-reset
WARNING: All configuration, log files and previous images will be wiped out and device
will reboot! Do you want to continue? [y/n]: y

!!! factory-reset command executed. System will reboot immediately !!!

duta# [ 112.393117] reboot: Restarting system
BIOS 3A04 POST starts...
[11][32][15][19][37][4F][3B][60][61][9A][62][78][68][69][6A][70][71][79][90][92][94][94]
[94][94][94][94][94][95][96][91][92][97]

Processor Type: Intel(R) Atom(TM) CPU C3758 @ 2.20GHz
Processor Speed: 2200 MHz
Total Memory: 16GB (DDR4 2400MHz)

USB Devices total: 0 KBDs, 0 MICE, 0 MASS, 1 HUBs

Detected ATA/ATAPI Devices...
SATA PORT0: KINGSTON OM8POS3128B-A0, S.M.A.R.T Supported
Version 2.19.1266. Copyright (C) 2021 American Megatrends, Inc.
BIOS Date: 06/26/2021 00:36:24 Ver: 3A04
Press <DEL> or <F2> to enter setup.
Press <F11> for BBS POPUP menu.
Press <F12> if you want to boot from the network.

Booting...Welcome to GRUB!

GNU GRUB version 2.06

+-----+
```



```

|*Open Network Linux
| ONIE
|
|
|
|
|
|
|
|
|-----+

Use the ^ and v keys to select which entry is highlighted.
Press enter to boot the selected OS, `e' to edit the commands
before booting or `c' for a command-line.
The highlighted entry will be executed automatically in 0s.
Booting `Open Network Linux'

Loading Open Network Linux ...
initrng: Linux Random Number Generator (RNG) early init
modprobe ixgbe
found interface name alias eth0 --> mal
remapping interface eth0 --> mal
INFO:onl-mounts:Running fsck on ONL-IAHDATA [ /dev/sda7 ]...
INFO:onl-mounts:/dev/sda7 [ ONL-IAHDATA ] is clean.
INFO:onl-mounts:Running fsck on ONL-SECDATA [ /dev/sda10 ]...
INFO:onl-mounts:/dev/sda10 [ ONL-SECDATA ] is clean.
INFO:onl-mounts:Running fsck on ONL-APPDATA [ /dev/sda8 ]...
INFO:onl-mounts:/dev/sda8 [ ONL-APPDATA ] is clean.
INFO:onl-mounts:Running fsck on ONL-USRDATA [ /dev/sda6 ]...
INFO:onl-mounts:/dev/sda6 [ ONL-USRDATA ] is clean.
INFO:onl-mounts:Running fsck on ONL-CONFIG [ /dev/sda4 ]...
INFO:onl-mounts:/dev/sda4 [ ONL-CONFIG ] is clean.
INFO:onl-mounts:Running fsck on ONL-BOOT [ /dev/sda3 ]...
INFO:onl-mounts:/dev/sda3 [ ONL-BOOT ] is clean.
INFO:onl-mounts:Running fsck on ONL-DATA [ /dev/sda11 ]...
INFO:onl-mounts:/dev/sda11 [ ONL-DATA ] is clean.
INFO:onl-mounts:Running fsck on ONL-IMAGES [ /dev/sda5 ]...
INFO:onl-mounts:/dev/sda5 [ ONL-IMAGES ] is clean.
INFO:PKI:Using existing private key.
INFO:PKI:Using existing certificate.
2025-02-06 06:08:18 - hwclock -s executed
Letsssss setup device for migration
*****
*
* Open Network Linux Loader
*
*      Version: ONL-main
*      Id: 2025-01-24.07:54-0099757
*
*      Platform: x86-64-extremenetworks-chassis-32c-r0
*      mal: e4:db:ae:5c:b4:01
*
*****
[ boot-config ]
NETDEV=mal
BOOTMODE=INSTALLED
SWI=images::latest
[ Starting Autoboot ]
[ Configuring Interfaces ]
[ BOOTMODE is installed. ]
[ Trying dir:data:/... ]

```

```
creating 1048576k of tmpfs in /tmp/boot-tmpfs-MOSBoo
Mounting dir:data:/
DEBUG:swimount:+ blkid
DEBUG:swimount:+ ubinfo -a
DEBUG:swimount:+ mount -o rw,remount /mnt/onl/data
Switching rootfs
[ 16.078279] Loading IQPKT module...
[ 16.132372] Finished Loading IQPKT module...
[ExtremeOneOS Mgmt IFC Stack]: Success
[ 23.672363] bond0: (slave eth4): invalid new link 3 on slave

This system is for authorized users only. All activity is logged and regularly
checked by systems personal. Individuals using this system without authority or
in excess of their authority are subject to having all their services revoked.
Any illegal activities conducted by user or attempts to take down this system
or its services will be reported to local law enforcement and said user will be
punished to the full extent of the law. Anyone using this system consents to these terms.
duta login: [ 114.876361] bond0: (slave eth3): invalid new link 3 on slave
[ 118.248362] bond0: (slave eth3): invalid new link 3 on slave

duta login: admin
Password:
Last login: Thu Feb 6 01:05:45 EST 2025 on ttyS0

*****
System is ready for all commands
*****
```

grpc-server

Use the **grpc-server** command to specify a gRPC server name and enter gRPC server system configuration (config-system-grpc-server-*name*) mode.

Syntax

```
grpc-server [server-name ]  
no grpc-server [server-name ]
```

Command Default

No gRPC server instance is enabled on the device.

Parameters

server-name

(Optional) Name of the gRPC server instance. Use 1 to 64 characters. If you do not specify a server name, the default instance is assumed.

Modes

System configuration (config-system)

Usage Guidelines

You can create multiple gRPC server instances with a user defined VRF for in-band access. .

You must disable the gRPC server instance before using this command (or any other command) to alter its configuration. Then, you must re-enable the instance to restore its functionality. To disable or enable a gRPC server instance, see [enable \(gRPC server configuration\)](#) on page 103.

When enabled, the gRPC server runs over a secure TLS 1.2 connection.

To enable a GRPC server, you must associate a certificate ID. To do so, see [certificate-id](#) on page 53.

Use the **no** form of this command to remove all configuration for the gRPC server instance.

Examples

The following example configures a gRPC server named mygrpcserver and enters gRPC server system configuration (config-system-grpc-server-*name*) mode:

```
device# configure terminal  
device(config)# system
```

```
device(config-system)# grpc-server mygrpcserver
device(config-system-grpc-server-mygrpcserver)# certificate-id ssl-reserved-generated
device(config-system-grpc-server-mygrpcserver)# port 443
device(config-system-grpc-server-mygrpcserver)# enable
device(config-system-grpc-server-mygrpcserver)#
```

The following example displays the gRPC server configuration that is running currently on the device. In this example, the default gRPC server and a gRPC server named mygrpcserver are configured:

```
device# show running-config system grpc-server mygrpcserver
system
  grpc-server mygrpcserver
    certificate-id ssl-reserved-generated
    port 443
    enable
  !
!
device#
```

grub

Use the **grub** command to enter GRUB system configuration (config-system-grub) mode on a device.

Syntax

grub

Parameters

This command has no keywords or variables.

Modes

System configuration (config-system)

Usage Guidelines

Use this command to enter the mode for configuring the GRand Unified Bootloader (GRUB) settings such as password protection of the bootloader on a device. GRUB is a powerful and widely-used bootloader in many UNIX-like operating systems. A bootloader is the first program that runs when a device boots. It loads the operating system into memory and starts it. Its main role is to manage and boot multiple operating systems from a single device.



Note

The initial username for logging into the GRUB user interface itself is "root". GRUB prompts for a new password during the first login.

Examples

The following example enters GRUB system configuration (config-system-grub) mode on the device:

```
device# configure terminal
device(config)# system
device(config-system)# grub
device(config-system-grub)#
```

The following example displays the GRUB configuration that is running currently on the device:

```
device # show running-config system grub

system
  grub
    username root password-hashed
grub.pbkdf2.sha512.10000.99290C16E44A4613BFAF94D724F081CE970C7AD32D0B08FB38C0AB8CE91BBF4DD
C7482B50E59B9D45BF0A5E9269F50CFDA03F45E59B42A78FE5EB35D589CFFBF.F2DFFFBF15622F841F31EBDF35
5B2CC6DA6820A244E44A54498C220E6C1881DBB63D1484994D6E1DCE7EC5691C6FC6337F49A4DF812CED2CAE0D
599D7E2A6E71
!
```

```
!  
device#
```

hardware (Extreme ONE OS)

Use the **hardware** command to enter Hardware configuration (config-hardware) mode.

Syntax

hardware

Parameters

This command has no keywords or variables.

Modes

Global configuration (config)

Usage Guidelines

Use this command to enter the mode for configuring connectors and breakout modes for those connectors.

Examples

The following example enters Hardware configuration (config-hardware) mode and configures a connector on the device and a breakout mode for the connector:

```
device# configure terminal
device(config)# hardware
device(config-hardware)# connector 0/1
device(config-connector-0/1)# breakout 4x10g
device(config-connector-0/1)#
```

The following example displays the hardware configuration that is running currently on the device:

```
device# show running-config

! Firmware: 10.2.1.0, Application(s): 22.2.1.0
! Date and Time: Tue, 2025-12-09 15:16:13 EST

...

device(config)# show running-config hardware

hardware
  connector 0/1
    breakout 4x10g
!
device#
```

host (Extreme ONE OS)

Use the **host** command to configure the SNMP agent with the TRAP destination information.

Syntax

```
host { ip-address | host-name } { community-string | user-name }  
no host { ip-address | host-name } { community-string | user-name }
```

Command Default

The SNMP server on the device has no TRAP receivers configured.

Parameters

ip-address | *host-name*

Trap receiver IPv4 or IPv6 address or hostname.

community-string | *user-name*

Community string or user.

- (For SNMPv1 or SNMPv2c) Specify a community string using 2 to 16 alphanumeric.
- (For SNMPv3) Specify a username using 1 to 32 alphanumeric.

Modes

SNMP server VRF system configuration (*config-system-snmp-server-vrf-vrf-name*)

Usage Guidelines

Use this command to configure a trap receiver so that all SNMP traps sent by the device arrive at the same SNMP trap receiver or set of receivers (which are usually one or more host devices on the network).

By default, the host is considered a v2c host, unless you enter the host mode and change it using the **version** command. The default port number is 162 unless changed via the **udp-port** command.

Only valid unicast IPv4 and IPv6 addresses are supported. Broadcast IPv4 addresses and multicast IP addresses are not supported.

Use the **no** form of this command to remove the SNMP server trap receiver configuration.

Examples

The following example configures 4.4.4.4 as the SNMP trap receiver:

```
device# configure terminal
device(config)# system
device(config-system)# snmp-server DEFAULT
device(config-system-snmp-server)# community mynet
device(config-system-snmp-server)# vrf mgmt-vrf
device(config-system-snmp-server-vrf-mgmt-vrf)# enable
device(config-system-snmp-server-vrf-mgmt-vrf)# host 4.4.4.4 mynet
device(config-system-snmp-server-vrf-mgmt-vrf-host-4.4.4.4-****)# version 2c
device(config-system-snmp-server-vrf-mgmt-vrf-host-4.4.4.4-****)# udp-port 200
device(config-system-snmp-server-vrf-mgmt-vrf-host-4.4.4.4-****)# source-interface
ethernet 0/1
device(config-system-snmp-server-vrf-mgmt-vrf-host-4.4.4.4-****)#
```

The following example displays the SNMP server configuration that is running currently on the device. In this example, an SNMP trap receiver is specified using an IPv4 address, a community string, a UDP port, and an SNMP version:

```
device# show running-config system snmp-server

system
  snmp-server DEFAULT
    vrf mgmt-vrf
      enable
      host 4.4.4.4 jERww/4GrM
      version 2c
      udp-port 200
      source-interface ethernet 0/1.0
    !
  !
  community-hashed jERww/4GrM
!
device#
```



Tip

The SNMP server community string displays as a hash in the **show running-config system snmp-server** command output. If you want to unhide the SNMP server community string so that it displays as plain text, see [unhide-secrets](#) on page 381.

hostname (Extreme ONE OS)

Use the **hostname** command to configure the hostname for a deployed device.

Syntax

hostname *hostname*

no hostname

Parameters

hostname

hostname string. Use 1 to 32 alphanumeric characters.

Modes

System configuration (config-system)

Usage Guidelines

Use this command to configure the name that identifies the device on the network.

To restore the hostname to the default value, use the **no** form of this command.

Examples

The following example configures the hostname on the device:

```
device# configure terminal
device(config)# system
device(config-system)# hostname sanjose-r1
device(config-system)#
```

The following example displays the hostname configuration that is running currently on the device:

```
device# show running-config system hostname

system
  hostname sanjose-r1
!
device#
```

interface extMgmt

Use the **interface extMgmt** command to configure an external management interface and enter External management interface configuration (config-if-extMgmt-*name*) mode.

Syntax

```
interface extMgmt interface-name
```

Parameters

interface-name

Name of the interface. Use either 1 or 2.

Modes

Global configuration (config)

Usage Guidelines



Note

This command is supported only on the Extreme 8730 platform.

External management ports do not support the interface counter **show** or **clear** commands.

External management ports are Layer 2 interfaces. Therefore, IP configuration is not supported on these ports.

MTU configuration is not supported on external management ports.

Use the **no** form of this command to remove the interface configuration from the device.

Examples

The following example enters External management interface configuration (config-if-extMgmt-*name*) mode and configures extMgmt interface 1:

```
device# configure terminal
device(config)# interface extMgmt 1
device(config-if-extMgmt-1)# description external mgmt interface 1
device(config-if-extMgmt-1)# no shutdown
device(config-if-extMgmt-1)#
```

The following example displays the configuration of extMgmt interface 1 that is running currently on the device:

```
device# show running-config interface extMgmt 1

interface extMgmt 1
  description external mgmt interface 1
  no shutdown
!
device#
```

interface management (Extreme ONE OS)

Use the **interface management** command to configure the management interface and enter Management interface configuration (config-if-management-*name*) mode.

Syntax

interface management *interface-name*

Parameters

interface-name

Name of the management interface. In this release, the only supported value is 0.

Modes

Global configuration (config)

Usage Guidelines

To view details about the management interface on a device, see [show interface management](#) on page 249.

Use the **no** form of this command to remove the management interface configuration from the device.

Examples

The following example enters Management interface configuration (config-if-management-*name*) mode and configures management interface 0:

```
device# configure terminal
device(config)# interface management 0
device(config-if-management-0)# no ipv4 address dhcp
device(config-if-management-0)# ipv4 address 10.24.85.81/20
device(config-if-management-0)# ipv6 address 2001:DB8::69bc:832:e61f:13ff:fe67:4b94/32
device(config-if-management-0)# no shutdown
device(config-if-management-0)#
```

The following example displays the configuration of management interface 0 that is running currently on the device:

```
device# show running-config interface management 0

interface management 0
  ipv4 address 10.24.85.81/20
  ipv6 address 2001:DB8::69bc:832:e61f:13ff:fe67:4b94/32
  no shutdown
!
device#
```

ipv4 address (BMC interface configuration)

Use the **ipv4 address** command to set a static IPv4 address and prefix or specify DHCP on the BMC management interface.

Syntax

```
ipv4 address { ipv4-address | dhcp }  
no ipv4 address dhcp
```

Command Default

The interface obtains an IPv4 address via DHCP.

Parameters

ipv4-address

Static IPv4 address. Use IPv4 CIDR notation (*A.B.C.D/M*), where *M* is the prefix length in the range of 0–32 for IPv4 addresses. For example, use an IPv4 address of 10.0.0.0 and a prefix length of 8 to specify addresses in the range of 10.0.0.0 to 10.255.255.255.

dhcp

Uses DHCP to obtain an IPv4 address.

Modes

BMC interface configuration (interface-*name*)

Usage Guidelines

The BMC is a microcontroller and the major component of the Intelligent Platform Management Interface (IPMI) subsystem, which assists in remote hardware monitoring, management, and troubleshooting of the device.

This command requires up to one minute for an IPv4 address change to take effect on the BMC hardware. This delay also applies to the settings such as the shutdown toggle, subnet mask, default gateway IPv4 address, or reset. This delay is an inherent hardware attribute of a BMC.

If you want to switch from DHCP to a static IPv4 address and prefix on the interface, you must first disable DHCP by entering **no ipv4 address dhcp**.

To display details about the internal BMC LAN configuration on the device, see [show bmc-status \(Extreme ONE OS\)](#) on page 225.

Examples

The following example configures and enables the BMC management interface on the device. In this example, the interface uses a static IPv4 address and prefix:

```
device# configure terminal
device(config)# bmc
device(bmc)# interface 0
device(interface-0)# no ipv4 address dhcp
device(interface-0)# ipv4 address 192.168.1.1/24
device(interface-0)# no shutdown
device(interface-0)#
```

The following example shows the BMC state that is running currently on the device:

```
device# show running-state bmc

!
bmc
  interface 0
    no ipv4 address dhcp
    ipv4 address 192.168.1.1/24
    no shutdown
  !
!
device#
```

ipv4 gateway (BMC interface configuration)

Use the **ipv4 gateway** command to set the default gateway IPv4 address on the BMC management interface.

Syntax

```
ipv4 gateway ipv4-address
```

Command Default

The interface has no default gateway IPv4 address.

Parameters

ipv4-address

IPv4 address. Use IPv4 notation (*A.B.C.D*). For example, use an IPv4 address of 192.0.2.0.

Modes

BMC interface configuration (*interface-name*)

Usage Guidelines

The BMC is a microcontroller and the major component of the Intelligent Platform Management Interface (IPMI) subsystem, which assists in remote hardware monitoring, management, and troubleshooting of the device.

This command requires up to one minute for a default IPv4 gateway change to take effect on the BMC hardware. This delay also applies to the settings such as the IPv4 address, shutdown toggle, subnet mask, default gateway IPv4 address, or reset. This delay is an inherent hardware attribute of a BMC.

To display details about the internal BMC LAN configuration on the device, see [show bmc-status \(Extreme ONE OS\)](#) on page 225.

Examples

The following example configures and enables the BMC management interface on the device. In this example, the interface uses 10.32.83.1 as the default gateway IPv4 address:

```
device# configure terminal
device(config)# bmc
device(bmc)# interface 0
device(interface-0)# ipv4 gateway 192.0.2.0
device(interface-0)# no shutdown
device(interface-0)#
```


The following example shows the BMC status on the device:

```
device# show bmc-status

IP Address Source : Static Address
IP Address : 198.51.100.0
Subnet Mask : 255.255.255.0
MAC Address : b4:a3:bd:0d:88:07
Default Gateway IP : 192.0.2.0
Firmware Revision : 1.11
8730-32d#
```

The following example shows the BMC configuration that is running currently on the device:

```
device# show running-config bmc

!
bmc
interface 0
no ipv4 address dhcp
ipv4 address 198.51.100.0/24
ipv4 gateway 192.0.2.0
no shutdown
!
!
```

The following example shows the BMC state that is running currently on the device:

```
device# show running-state bmc

!
bmc
interface 0
no ipv4 address dhcp
ipv4 address 198.51.100.0/24
ipv4 gateway 192.0.2.0
no shutdown
!
!
```

key

Use the **key** command to generate an SSH host key pair for Elliptic Curve Digital Signature Algorithm (ECDSA) or Rivest-Shamir-Adleman (RSA) public key encryption.

Syntax

```
key { ecdsa curve | rsa key-size }  
no key { ecdsa curve | rsa key-size }
```

Command Default

The SSH server on the device has host key pairs already generated on first boot (because SSH is enabled by default). (This command allows additional SSH server instances for a specific VRF to have unique keys from the mgmt-vrf generated.)

Parameters

curve

Curve size in bits for generating an ECDSA host-key pair. The only curve size supported in this release is 256.

key-size

Key size in bits for generating an RSA host-key pair. Use either 2048 or 4096.

Modes

SSH server system configuration (config-system-ssh-server-*name*)

Examples

The following example creates the SSH host key pair for RSA:

```
device# configure terminal  
device(config)# system  
device(config-system)# ssh-server h2  
device(config-system-ssh-server-h2)# cipher aes128-gcm-openssh aes256-gcm-openssh  
chacha20-poly1305-openssh  
device(config-system-ssh-server-h2)# key rsa 2048  
device(config-system-ssh-server-h2)#
```

The following example displays the SSH server configuration that is running currently on the device:

```
device# show running-config system ssh-server h2  
  
system  
  ssh-server h2  
    cipher aes128-gcm-openssh aes256-gcm-openssh chacha20-poly1305-openssh  
    key rsa 2048  
    !
```

```
!  
device#
```

keychain

Use the **keychain** command to configure a keychain and enter keychain configuration mode.

Syntax

keychain *keychain-name*

no keychain *keychain-name*

Command Default

The device has no keychains configured.

Parameters

keychain-name

Name of the keychain. Use 4 to 32 characters.

Modes

Global configuration (config)

Usage Guidelines

A maximum of 128 keychains is supported.

Use the no form of this command to remove a keychain (including all of its keys). Before you remove a keychain, ensure that no feature uses it. If a feature is configured to use a keychain that you remove, that feature is likely to fail to communicate with other devices.

Examples

The following example configures a keychain and enters keychain configuration mode (*config-keychain-name*):

```
device# configure terminal
device(config)# keychain keychain1
device(config-keychain-keychain1)#
```

The following example displays the keychain configuration that is running currently on the device. In this example, a keychain named keychain1 is configured:

```
device# show running-config keychain

keychain keychain1
device#
```

key-exchange

Use the **key-exchange** command to configure SSH to use a set of key exchange algorithms in a priority order that you specify.

Syntax

```
key-exchange { [ curve25519-sha256 ] [ curve25519-sha256-libssh ]
  [ diffie-hellman-group-exchange-sha1 ] [ diffie-hellman-group-exchange-sha256 ]
  [ diffie-hellman-group14-sha1 ] [ diffie-hellman-group14-sha256 ]
  [ diffie-hellman-group16-sha512 ] [ diffie-hellman-group18-sha512 ]
  [ ecdh-sha2-nistp256 ] [ ecdh-sha2-nistp384 ]
  [ ecdh-sha2-nistp521 ] }
```

no key-exchange

Command Default

The following set of key exchange algorithms is used in the following priority order (first is highest):

```
curve25519-sha256
curve25519-sha256-libssh
diffie-hellman-group-exchange-sha1
diffie-hellman-group-exchange-sha256
diffie-hellman-group14-sha1
diffie-hellman-group14-sha256
diffie-hellman-group16-sha512
diffie-hellman-group18-sha512
ecdh-sha2-nistp256
ecdh-sha2-nistp384
ecdh-sha2-nistp521
```

```
{ [ curve25519-sha256 ] [ curve25519-sha256-libssh ] [ diffie-hellman-group-exchange-sha1 ]
  [ diffie-hellman-group-exchange-sha256 ] [ diffie-hellman-group14-sha1 ] [ diffie-hellman-group14-sha256 ]
  [ diffie-hellman-group16-sha512 ] [ diffie-hellman-group18-sha512 ] [ ecdh-sha2-nistp256 ]
  [ ecdh-sha2-nistp384 ] [ ecdh-sha2-nistp521 ] }
```

Type of key exchange algorithm. Each name indicates the algorithm and cryptographic parameters that are used. Enter any combination of one or more algorithms in your preferred order of priority (first is highest).

Modes

SSH server system configuration (config-system-ssh-server-*name*)

Usage Guidelines

Key exchange algorithms are used to securely exchange a shared session key with a peer.

Each keyword represents an algorithm that is used to distribute a shared key to prevent external interference, manipulation, or recovery.

Only user selected key exchange algorithms are configured. The rest are ignored.

Use the **no** form of this command to restore the default key exchange algorithms.

Examples

The following example configures a set of three key exchange algorithms in order of priority (with ecdh-sha2-nistp256 as the highest priority):

```
device# configure terminal
device(config)# system
device(config-system)# ssh-server h2
device(config-system-ssh-server-h2)# aes128-gcm-openssh aes256-gcm-openssh chacha20-
poly1305-openssh
device(config-system-ssh-server-h2)# client-alive 20
device(config-system-ssh-server-h2)# key rsa 2048
device(config-system-ssh-server-h2)# key-exchange ecdh-sha2-nistp256 curve25519-sha256
diffie-hellman-group-exchange-sha256
device(config-system-ssh-server-h2)# mac hmac-sha2-256 hmac-sha2-512
device(config-system-ssh-server-h2)# vrf vrf-red
device(config-system-ssh-server-h2)# enable
device(config-system-ssh-server-h2)#
```

The following example displays the SSH server configuration that is running currently on the device. In this example, SSH uses a set of three key exchange algorithms in order of priority (with ecdh-sha2-nistp256 as the highest priority):

```
device# show running-config system ssh-server h2

system
  ssh-server h2
    cipher aes128-gcm-openssh aes256-gcm-openssh chacha20-poly1305-openssh
    client-alive 20
    key rsa 2048
    key-exchange ecdh-sha2-nistp256 curve25519-sha256 diffie-hellman-group-exchange-sha256
    mac hmac-sha2-256 hmac-sha2-512
    vrf vrf-red
    enable
  !
!
device#
```

key-id

Use the **key-id** command to configure key IDs for a keychain.

Syntax

key-id *value*

no key-id *value*

Command Default

The specified keychain has no key IDs.

Parameters

value

Unique key ID value. Unique key ID in the integer range 1–255 or hex string 0x01–0xFF.

Modes

Keychain configuration (config-*keychain-name*)

Usage Guidelines

A maximum of eight key IDs per keychain is supported.

Use the no form of this command to remove a key ID from the current keychain.

Examples

The following example configures a keychain named keychain1 with two keys that use 4 and 215 as their respective key IDs:

```
device# configure terminal
device(config)# keychain keychain1
device(config-keychain-keychain1)# key-id 4
device(config-keychain-keychain1-key-4)# exit
device(config-keychain-keychain1)# key-id 215
device(config-keychain-keychain1-key-215)#
```

The following example displays the keychain configuration that is running currently on the device. In this example, a keychain named keychain1 is configured with two keys that have 4 and 215 as their respective key IDs:

```
device# show running-config keychain

keychain keychain1
  key-id 4
  key-id 215
device#
```

ldaps

Use the **ldaps** command to enable LDAP Secure (LDAPS) on an LDAP server in an LDAP server group.

Syntax

ldaps

no ldaps

Command Default

LDAPS on the current LDAP server host is disabled.

Parameters

This command has no keywords or variables.

Modes

AAA LDAP server group server system configuration (config-system-aaa-server-group-ldap-server-address)

Usage Guidelines

LDAPS encrypts LDAP data (including user credentials) in transit during any communication with the LDAP server. LDAPS in Extreme ONE OS uses Transport Layer Security (TLS) as the cryptographic protocol.



Note

Before configuring LDAPS, make sure that you copy trusted CA certificates from an external remote server to the system trust certificates list.

Use the **no** form of this command to disable LDAPS for the current LDAP server host.

Examples

The following example adds a server with IPv4 address 3.3.3.3 to the LDAP server group on the device. In this example, LDAPS is enabled on the server:

```
device# configure terminal
device(config)# system
device(config-system)# aaa
device(config-system-aaa)# server-group ldap
device(config-system-aaa-server-group-ldap)# server 3.3.3.3
device(config-system-aaa-server-group-ldap-server-3.3.3.3)# ldaps
device(config-system-aaa-server-group-ldap-server-3.3.3.3)#
```


The following example displays the LDAP server group configuration that is running currently on the device. In this example, the LDAP server group is configured with one LDAP server. LDAPS is enabled on the server:

```
device# show running-config system aaa server-group ldap

system
  aaa
    server-group ldap
      server 3.3.3.3
      ldaps
    !
  !
!
device#
```

license eula advanced_core

Use the **license eula advanced_core** command to accept or decline the End User License Agreement (EULA) for advanced core features on the Extreme 8730 platform.

Syntax

```
license eula [ accept | decline ] advanced_core
```

Command Default

The license EULA for advanced core features is in the denied state.

Parameters

accept

Enables the use of features covered under the advanced_core license.

decline

Disables the use of features covered under the advanced_core license.

Modes

EXEC (#)

Usage Guidelines

This command supports advanced core features such as integrated application hosting (IAH). This license is valid only on the Extreme 8730 platform. For a licensed feature, configuration and gNMI set for any leaf/list under the respective YANG paths are blocked unless the EULA is accepted.

To accept or decline licensed features such as IAH on Extreme ONE OS platforms except the Extreme 8730 or BGP EVPN, see [license eula advanced_features](#) on page 148.

To display all licenses added and their status, or to display the current EULA statement for features, see [show license](#) on page 251.

Examples

The following example accepts the EULA to enable advanced core features on the device:

```
device# license eula accept advanced_core
EULA accepted successfully for feature ADVANCED_CORE
Use of the features requires a license to be purchased within 30 days. By accepting the
EULA you indicate that you have read and accept the Extreme End User License Agreement
found at the following URL.
[https://learn.extremenetworks.com/rs/641-VMV-602/images/Extreme-Networks-End-User-
```

```
License-Agreement.pdf].  
device#
```

license eula advanced_features

Use the **license eula advanced_features** command to accept or decline the End User License Agreement (EULA) for advanced features on a device.

Syntax

```
license eula [ accept | decline ] advanced_features
```

Command Default

The license EULA for advanced features is in the denied state.

Parameters

accept

Enables the use of features covered under the advanced_features license.

decline

Disables the use of features covered under the advanced_features license.

Modes

EXEC (#)

Usage Guidelines

This command supports licensed features such as integrated application hosting (IAH) on all Extreme ONE OS platforms except the Extreme 8730 or BGP EVPN. For a licensed feature, configuration and gNMI set for any leaf/list under the respective YANG paths are blocked unless the EULA is accepted.

To accept or decline the license EULA for IAH on the Extreme 8730 platform, see [license eula advanced_core](#) on page 146.

To display all licenses added and their status, or to display the current EULA statement for features, see [show license](#) on page 251.

Examples

The following example accepts the EULA to enable advanced features on the device:

```
device# license eula accept advanced_features
EULA accepted successfully for feature ADVANCED_FEATURES
Use of the features requires a license to be purchased within 30 days. By accepting the
EULA you indicate that you have
read and accept the Extreme End User License Agreement found at the following URL.
[https://learn.extremenetworks.com/rs
/641-VMV-602/images/Extreme-Networks-End-User-License-Agreement.pdf].
device#
```

location (Extreme ONE OS)

Use the **location** command to configure an administrative location for a deployed device.

Syntax

```
location location-info  
no location
```

Command Default

The device has no administrative location details configured.

Parameters

```
location-info  
Location string. Use 1 to 255 alphanumeric characters in a single string.
```

Modes

System configuration (config-system)

Usage Guidelines

Use this command to configure the device's location details.

To delete the location, use the **no** form of this command.

Examples

The following example configures the device's location details:

```
device# configure terminal  
device(config)# system  
device(config-system)# location Bangalore device(config-system)#
```

The following example displays the location details configuration that is running currently on the device:

```
device# show running-config system location  
  
system  
location Bangalore  
!  
device#
```

logging (Extreme ONE OS)

Use the **logging** command to enter Logging system configuration (config-system-logging) mode.

Syntax

logging

Command Default

System logging is disabled.

Parameters

This command has no keywords or variables.

Modes

System configuration (config-system)

Usage Guidelines

Use this command to enter the mode for configuring system logging settings. This includes the settings for the remote server, whether to enable log messages on the console and when connected via Telnet or SSH. and severity levels for which to display log messages.

Examples

The following example enters Logging system configuration (config-system-logging) mode and configures system logging:

```
device# configure terminal
device(config)# system
device(config-system)# logging
device(config-system-logging)#
```

login-banner

Use the **login-banner** command to customize the banner message that the device displays during login authentication.

Syntax

login-banner *message*

no login-banner

Command Default

The device displays the following login banner message: This system is for authorized users only. All activity is logged and regularly checked by systems personal. Individuals using this system without authority or in excess of their authority are subject to having all their services revoked. Any illegal activities conducted by user or attempts to take down this system or its services will be reported to local law enforcement and said user will be punished to the full extent of the law. Anyone using this system consents to these terms.

Parameters

message

Login banner message string. The valid range is 1 to 2048 characters.

Modes

System configuration (config-system)

Usage Guidelines

Use this command to replace the default login banner message.

The login banner displays a message when a user tries to log in to the device. You should use the login banner to display legal notices such as security warnings and more permanent messages to users.

To configure the banner message to display *after* a user logs in to the device, see [motd-banner](#) on page 169.

Use the **no** form of this command to restore the default login banner message.

Examples

The following example configures the banner login message:

```
device# configure terminal
device(config)# system
```

```
device(config-system)# login-banner UNAUTHORIZED ACCESS TO THIS DEVICE IS NOT ALLOWED.  
Activities performed on this device are monitored and recorded. Do not access or  
configure this device without explicitly authorized permission. Unauthorized attempts to  
access or configure this device might result in civil or criminal penalties.  
device(config-system)#
```

The following example displays the banner messages that are running currently on the device. In this example, a custom login banner message is configured (but the default MOTD banner message has not been customized and is, therefore, not displayed):

```
device# show running-config system  
  
system  
    login-banner UNAUTHORIZED ACCESS TO THIS DEVICE IS NOT ALLOWED. Activities performed  
on this device are monitored and recorded. Do not access or configure this device  
without explicitly authorized permission. Unauthorized attempts to access or configure  
this device might result in civil or criminal penalties.  
!  
device#
```


ls (Extreme ONE OS)

Use the **ls** command to list the contents of a specified directory or all directories on a device.

Syntax

```
ls [ directory ] [ | output-filter ]
```

Parameters

directory

(Optional) Directory for which to display the contents. You can specify one of the following directories:

```
disk://config-file
disk://coredumps
disk://firmware
disk://iah
disk://pcap-file
disk://tech-support
usb://
```

Omitting this variable lists the contents of all of these directories.

| *output-filter*

(Optional) Output filter. The pipe symbol (|) is required. You can use any of the following output filters to tailor or redirect the output of the **ls** command:

Output field	Description
append <i>filename</i>	Appends the output text to a file. Use the following format for the filename: disk://config-file/ <i>filename</i> .
begin <i>regex</i>	Begins the output with the first line encountered that matches a pattern that you specify with a Regular Expression (regex). For example, enter ls disk://tech-support begin \btw* to list all files and folders in disk://tech-support starting with the first one encountered whose name begins with the letter 't' or 'T'. (\b indicates a word boundary.)
count	Counts the number of lines in the output.
exclude <i>regex</i>	Excludes the lines that match a pattern that you specify with a regex. For example, enter ls disk://iah exclude \bt\w* to omit all files and folders in disk://iah whose names begin with the letter 't' or 'T'.

Output field	Description
include <i>regex</i>	Includes the lines that match a pattern that you specify with a regex. For example, enter ls disk://tech-support include \bt\w* to list all files and folders in disk://tech-support whose names begin with the letter 't' or 'T.'
last <i>number</i>	Displays only the last <i>number</i> of lines.
linnum	Adds line numbers to the lines in the output.
more	Paginates the output.
nomore	Suppresses the output pagination.
save <i>filename</i>	Saves the output text to a file. Use the following format for the filename: disk://config-file/<i>filename</i> . Note: If a disk://config-file/<i>filename</i> file already exists, this output filter overwrites it without a warning.
until <i>regex</i>	Ends the output with the first line encountered that matches a pattern specified by a regex. For example, enter ls disk://tech-support until \wz\b* to list all files and folders in disk://tech-support until encountering the first one whose name ends with the letter 'z' or 'Z.'

Modes

EXEC (#)

Examples

The following example lists the contents of all directories on the device:

```
device# ls

config-file :
pcap-file  :
-rw-r--r-- 0 2025-04-02 23:33:16 README.md
tech-support :
-rw----- 6427      2025-04-01 08:54:42 TS-32d-250401T5416.log
-rw-r--r-- 5012653 2025-04-01 08:54:42 TS-32d-250401T5416.tar.gz
-rw----- 6427      2025-04-02 09:13:04 TS-32d-250402T1238.log
-rw-r--r-- 7837214 2025-04-02 09:13:04 TS-32d-250402T1238.tar.gz
firmware   :
coredumps  :
iah        :
drwxr-xr-x 4096      2025-04-03 05:23:52 efaboot
-rw-r--r-- 2599249794 2025-04-01 08:22:34 tpmv-4.7.3-7.amd64.deb
device#
```

The following example lists the contents of the disk://tech-support directory on the device:

```
device# ls disk://tech-support

tech-support  :
-rw-----   6427      2025-04-01 08:54:42  TS-32d-250401T5416.log
-rw-r--r--   5012653   2025-04-01 08:54:42  TS-32d-250401T5416.tar.gz
-rw-----   6427      2025-04-02 09:13:04  TS-32d-250402T1238.log
-rw-r--r--   7837214   2025-04-02 09:13:04  TS-32d-250402T1238.tar.gz
device#
```

The following example lists the directories and folders whose names end with the letter 'z' or 'Z' in the disk://tech-support directory:

```
device# ls disk://tech-support | include \wz\b*

-rw-r--r--   5012653   2025-04-01 08:54:42  TS-32d-250401T5416.tar.gz
-rw-r--r--   7837214   2025-04-02 09:13:04  TS-32d-250402T1238.tar.gz
device#
```

The following example redirects the listing of the contents of the disk://tech-support directory to a file named disk://config-file/files_for_support_call (and *overwrites* a file of the same name if one exists):

```
device# ls disk://tech-support | save disk://config-file/files_for_support_call
device#
```

The following example appends the listing of the contents of the disk://coredumps directory to the end of a file named disk://config-file/files_for_support_call:

```
device# ls disk://coredumps | append disk://config-file/files_for_support_call
device#
```

mac

Use the **mac** command to configure the SSH server to use a set of message authentication code (MAC) algorithms specified in a priority order that you specify.

Syntax

```
mac { [ hmac-sha1 ] [ hmac-sha1-96 ] [ hmac-sha1-etm-openssh ] [ hmac-sha2-256 ] [ hmac-sha2-512 ] [ hmac-sha2-256-etm-openssh ] [ hmac-sha2-512-etm-openssh ] }  
  
no mac
```

Command Default

The following MAC algorithms are configured in the following priority order (first is highest): hmac-sha2-512-etm-openssh, hmac-sha2-256-etm-openssh, hmac-sha2-512, hmac-sha2-256

Parameters

```
[ hmac-sha1 ] [ hmac-sha1-96 ] [ hmac-sha1-etm-openssh ] [ hmac-sha2-256 ] [ hmac-sha2-512 ] [ hmac-sha2-256-etm-openssh ] [ hmac-sha2-512-etm-openssh ]
```

Type of MAC algorithm. Each name indicates the algorithm and cryptographic parameters that are used. Enter any combination of one or more algorithms in your preferred order of priority (first is highest).

Modes

SSH server system configuration (config-system-ssh-server-*name*)

Usage Guidelines

Message authentication codes maintain the integrity of each message sent across an SSH connection.

If you are using an older SSH client, you should either update or configure Extreme ONE OS to include a supported MAC algorithm of the client.

Only the MAC algorithms that you specify are configured. The rest are ignored.

Use the **no** form of this command to remove the configured MAC algorithms.

Examples

The following example specifies two MAC algorithms for the SSH server on the device:

```
device# configure terminal  
device(config)# system
```

```
device(config-system)# ssh-server h2
device(config-system-ssh-server-h2)# cipher aes128-gcm-openssh aes256-gcm-openssh
chacha20-poly1305-openssh
device(config-system-ssh-server-h2)# client-alive 20
device(config-system-ssh-server-h2)# key rsa 2048
device(config-system-ssh-server-h2)# key-exchange ecdh-sha2-nistp256 curve25519-sha256
diffie-hellman-group-exchange-sha256
device(config-system-ssh-server-h2)# mac hmac-sha2-256 hmac-sha2-512
device(config-system-ssh-server-h2)#
```

The following example displays the SSH server configuration that is running currently on the device. In this example, the SSH server uses two MAC algorithms in order of priority (with hmac-sha2-256 as the first priority):

```
device# show running-config system ssh-server

system
  ssh-server h2
    cipher aes128-gcm-openssh aes256-gcm-openssh chacha20-poly1305-openssh
    client-alive 20
    key rsa 2048
    key-exchange ecdh-sha2-nistp256 curve25519-sha256 diffie-hellman-group-exchange-sha256
    mac hmac-sha2-256 hmac-sha2-512
    vrf vrf-red
    enable
  !
!
device#
```

map-role group

Use the **map-role group** command to configure a role mapping to a group that is configured on the LDAP server.

Syntax

```
map-role group group role { admin | user }  
no map-role group group
```

Command Default

The Lightweight Directory Access Protocol (LDAP) server has no role mappings configured.

Parameters

group *group*
Group name on the LDAP server. Use 1 to 64 characters.

role { **admin** | **user** }
User role.

Modes

AAA LDAP server group system configuration (config-system-aaa-server-group-ldap)

Usage Guidelines

Use the **no** form of this command to remove the specified user role.

Examples

The following example configures AAA on the device. This example configures the admin role for a group named group2 on the LDAP server:

```
device# configure terminal  
device(config)# system  
device(config-system)# aaa  
device(config-system-aaa)# server-group ldap  
device(config-system-aaa-server-group-ldap)# map-role group group2 role admin  
device(config-system-aaa-server-group-ldap)#
```

The following example displays the AAA configuration that is running currently on the device. In this example, the admin role for a group named group2 on the LDAP server is configured:

```
device# show running-config system aaa  
  
system  
aaa
```

```
server-group ldap
  map-role group group2 role admin
!
device#
```

min-version (TLS service-level system configuration)

Use the **min-version** command to set the TLS required minimum version for a specific TLS-enabled application on a device.

Syntax

```
min-version version  
no min-version
```

Command Default

The minimum required Transport Layer Security (TLS) setting for the application on the device uses the global TLS required minimum version setting (if configured).

Parameters

version
Minimum version to be required. Use 1.2 or 1.3.

Modes

TLS service-level system configuration (config-system-tls-service-*name*)

Usage Guidelines

This feature provides the flexibility to let you exclude specific services from the system-wide minimum required TLS version. This feature enhances security and administrative control by letting you use the CLI to enforce a minimum version (either TLSv1.2 or TLSv1.3) for specific TLS-enabled client/service applications. This ensures compatibility with legacy systems or third party integrations that might not support newer TLS versions.

Use the **no** form of this command to remove the service-level minimum version configurations.

To enforce a system-wide minimum TLS version across all TLS-enabled services, see [min-version \(TLS system configuration\)](#) on page 162.

Examples

The following example sets the TLS required minimum version for the Remote Procedure Calls (gRPC) server service on the device to TLSv1.2. This version overrides the global setting (if set):

```
device# configure terminal  
device(config)# system  
device(config-system)# tls  
device(config-system-tls)# service grpc-server
```



```
device(config-system-tls-service-grpc-server)# min-version 1.2
device(config-system-tls-service-grpc-server)#
```

The following example sets the TLS required minimum version for the RADIUS service on the device to TLSv1.2. This version overrides the global setting (if set):

```
device# configure terminal
device(config)# system
device(config-system)# tls
device(config-system-tls)# service radius-client
device(config-system-tls-service-radius-client)# min-version 1.2
device(config-system-tls-service-radius-client)#
```

The following example sets the TLS required minimum version for the LDAP service on the device to TLSv1.2. This version overrides the global setting (if set):

```
device# configure terminal
device(config)# system
device(config-system)# tls
device(config-system-tls)# service ldap-client
device(config-system-tls-service-ldap-client)# min-version 1.2
device(config-system-tls-service-ldap-client)#
```

The following example sets the TLS required minimum version for the system logging (syslog) service on the device to TLSv1.2. This version overrides the global setting (if set):

```
device# configure terminal
device(config)# system
device(config-system)# tls
device(config-system-tls)# service syslog-client
device(config-system-tls-service-syslog-client)# min-version 1.2
device(config-system-tls-service-syslog-client)#
```

The following example sets the TLS required minimum version for the HTTPS service on the device to TLSv1.2. This version overrides the global setting (if set):

```
device# configure terminal
device(config)# system
device(config-system)# tls
device(config-system-tls)# service https-client
device(config-system-tls-service-https-client)# min-version 1.2
device(config-system-tls-service-https-client)#
```

The following example displays the TLS configuration that is running currently on the device. In this example, the required minimum version for the gRPC server service is set to TLSv1.2:

```
device# show running-config system tls

system
  tls
    service grpc-server
      min-version 1.2
    !
  !
!
device#
```

min-version (TLS system configuration)

Use the **min-version** command to set the global TLS minimum version for all TLS-enabled services on a device.

Syntax

```
min-version version  
no min-version
```

Command Default

The global minimum Transport Layer Security (TLS) setting on the device is TLSv1.2.

Parameters

version

Minimum version to be accepted. You must specify 1.3 in this release.

Modes

TLS system configuration (config-system-tls)

Usage Guidelines

This command provides a system-wide minimum TLS version configuration feature all applications on the device. This feature enhances security and administrative control by letting you use the CLI to enforce a minimum version (TLSv1.3 in this release) across all TLS-enabled client/service applications.

To restore the global required minimum TLS version to the default setting, use the **no** form of this command.

You can also exclude specific services from this enforcement to ensure compatibility with legacy systems or third-party integrations that might not support newer TLS versions. For details, see [min-version \(TLS service-level system configuration\)](#) on page 160.

Examples

The following example sets the global TLS minimum version on the device to TLSv1.3:

```
device# configure terminal  
device(config)# system  
device(config-system)# tls  
device(config-system-tls)# min-version 1.3  
device(config-system-tls)#
```

The following example displays the TLS configuration that is running currently on the device. In this example, the global required minimum version is set to TLSv1.3:

```
device# show running-config system tls

system
  tls
    min-version 1.3
  !
!
device#
```

mode-transport

Use the **mode-transport** command to enable the RELP or TCP transport mode for the remote server for logging on a device.

Syntax

```
mode-transport { relp | tcp }  
no mode-transport { relp | tcp }
```

Command Default

UDP mode is enabled.

Parameters

relp

RELP transport mode.

tcp

TCP transport mode.

Modes

Logging remote server system configuration (*config-system-logging-remote-server-addr-or-domain*)

Usage Guidelines

This command enables the device to forward remote system logging messages (rsyslogs) to a remote syslog server via TCP or the Reliable Event Logging Protocol (RELP) (this protocol runs on top of TCP).

Use the **no** form of this command to restore the default behavior.

Examples

The following example configures logging on a device. In this example, TCP transport mode is enabled for a remote syslog server named with IP address 1.1.1.1:

```
device# configure terminal  
device(config)# system  
device(config-system)# logging  
device(config-system-logging)# remote-server 1.1.1.1  
device(config-system-logging-remote-server-1.1.1.1)# mode-transport tcp  
Warning: Existing Host configuration changed  
device(config-system-logging-remote-server-1.1.1.1)#
```

The following example configures logging on a device. In this example, RELP transport mode is enabled for a remote syslog server named with IP address 2.2.2.2:

```
device# configure terminal
device(config)# system
device(config-system)# logging
device(config-system-logging)# remote-server 2.2.2.2
device(config-system-logging-remote-server-2.2.2.2)# mode-transport relp
Warning: Existing Host configuration changed
device(config-system-logging-remote-server-2.2.2.2)#
```

The following example displays the logging configuration that is running currently on a device. In this example, TCP transport mode is enabled for a remote syslog server with IP address 1.1.1.1:

```
device# show running-config system logging remote-server

system
  logging
    remote-server 1.1.1.1
      mode-transport tcp
    !
  !
!
device#
```

The following example displays details about the remote syslog server that is configured on a device. In this example, TCP transport mode is enabled for a remote syslog server with IP address 1.1.1.1:

```
device# show system logging remote-server
```

HOST	PORT	TRANSPORT	SECURE-FORWARDING	VRF-NAME
1.1.1.1	514	TCP	TLS	test1

```
device#
```

monitor

Use the **monitor** command to start or stop monitor logging.

Syntax

```
monitor { start | stop }
```

Command Default

Monitor logging is disabled.

Parameters

start

Starts the monitor logging.

stop

Stops the monitor logging.

Modes

EXEC (#)

Examples

The following example configures system monitor logging and then starts and stops system monitor logging:

```
device# configure terminal
device(config)# system
device(config-system)# logging
device(config-system-logging)# monitor severity info
device(config-system-logging)# end
device# monitor start
device#
device(config)# interface ethernet 0/1
device(config-if-eth-0/1)# shutdown
device(config-if-eth-0/1)# 2025-03-07 21:13:33.7861 fwd-hal[68]: Level:info LogID:0
Topic:1 Msg:Updating gsfp User adminState: false
2025-03-07 21:13:33.7907 interface-mgr[7]: Level:info LogID:25016 Msg:Interface ethernet
0/1 Admin DOWN
2025-03-07 21:13:33.7910 interface-mgr[7]: Level:info LogID:25014 Msg:Interface ethernet
0/1 Operationally DOWN
2025-03-07 21:13:33.7926 lldp[54]: Level:info LogID:14009 Msg:LLDP Neighbor deleted from
the interface ethernet 0/1

device(config-if-eth-0/1)# no shutdown
device(config-if-eth-0/1)# 2025-03-07 21:13:37.0580 fwd-hal[68]: Level:info LogID:0
Topic:1 Msg:Updating gsfp User adminState: true
2025-03-07 21:13:37.0620 interface-mgr[7]: Level:info LogID:25015 Msg:Interface ethernet
0/1 Admin UP
2025-03-07 21:13:37.3775 interface-mgr[7]: Level:info LogID:25013 Msg:Interface ethernet
0/1 Operationally UP
```

```
device# monitor stop  
device#
```

monitor (logging system configuration)

Use the **monitor** command to display system logs in CLI sessions when connected indirectly using Telnet or SSH.

Syntax

```
monitor [ severity severity-name ]  
no monitor [ severity severity-name ]
```

Command Default

System logging when connected indirectly using Telnet or SSH is disabled. The severity level is set to “info” when you use this command without specifying a severity level.

Parameters

severity-name

(Optional) Level of severity. Enter one of the following values: emergency, alert, critical, error, warning, notice, info, or debug.

Modes

Logging system configuration (config-system-logging)

Usage Guidelines

Use this command to display system logs in CLI sessions through Telnet or SSH. Monitor logging is what displays when you are connected to the CLI via Telnet or SSH (meaning not through the console port).

If the “error” level is configured, only the logs with a level of “error” or a higher level such as “critical” or “alert” are displayed.

To display system logs in CLI sessions directly to the console, see [console \(logging system configuration\)](#) on page 84.

Use the **no** form of this command to restore the default behavior.

Examples

The following example configures the system log to display in a CLI session when you are connected through Telnet or SSH:

```
device# configure terminal  
device(config)# system  
device(config-system)# logging  
device(config-system-logging)# monitor severity error  
device(config-system-logging)#
```


motd-banner

Use the **motd-banner** command to configure the message of the day (MOTD) banner message that the device displays after authenticating the user.

Syntax

```
motd-banner message  
no motd-banner
```

Command Default

The device does not display an MOTD banner message.

Parameters

message

MOTD banner message string. The valid range is 1 to 2048 characters.

Modes

System configuration (config-system)

Usage Guidelines

Use this command to configure the banner message to display *after* a user logs in to the device successfully. You should use an MOTD banner message for helpful information (such as impending system shutdowns) that applies to all users.

To customize the banner message that displays *during* login authentication, see [login-banner](#) on page 151.

Use the **no** form of this command to restore the default behavior.

Examples

The following example configures the MOTD banner message.

```
device# configure terminal  
device(config)# system  
device(config-system)# motd-banner Welcome to Extreme ONE OS. All devices will be  
rebooted on Sunday at 10 A.M.  
device(config-system)#
```

The following example displays the banner messages that are running currently on the device. In this example, a custom MOTD banner message is configured (but the default login banner message has not been customized and is, therefore, not displayed):

```
device# show running-config system banner  
  
system
```

```
motd-banner Welcome to Extreme ONE OS. All devices will be rebooted on Sunday at 10 A.M.  
!  
device#
```

mutual-tls

Use the **mutual-tls** command to enable mutual TLS (mTLS) client authentication on the gRPC server instance.

Syntax

mutual-tls

no mutual-tls

Command Default

mTLS client authentication on the gRPC server instance is disabled.

Parameters

gRPC server system configuration (config-system-grpc-server-*name*)

Modes

This command has no keywords or variables.

Usage Guidelines

You must disable the gRPC server instance before using this command (or any other command) to alter its configuration. Then, you must re-enable the instance to restore its functionality. To disable or enable a gRPC server instance, see [enable \(gRPC server configuration\)](#) on page 103.

Extreme ONE OS mTLS client authentication on the gRPC server instance uses an X.509 certificate.

To disable mTLS client authentication on the gRPC server instance, use the **no** form of this command.

Examples

The following example configures a gRPC server instance. In this example, mTLS client authentication is enabled on the instance:

```
device# configure terminal
device(config)# system
device(config-system)# grpc-server mygrpcserver
device(config-system-grpc-server-mygrpcserver)# certificate-id test_cert_id
device(config-system-grpc-server-mygrpcserver)# port 9340
device(config-system-grpc-server-mygrpcserver)# mutual-tls
device(config-system-grpc-server-mygrpcserver)# vrf vrf-blue
device(config-system-grpc-server-mygrpcserver)# enable
device(config-system-grpc-server-mygrpcserver)#
```

The following example displays the gRPC server instance configuration that is running currently on the device. In this example, mTLS client authentication on the instance is enabled:

```
device# show running-config system grpc-server

system
  grpc-server mygrpcserver
  certificate-id test_cert_id
  port 9340
  mutual-tls
  vrf vrf-blue
  enable
!
device#
```

name-server (Extreme ONE OS)

Use the **name-server** command to set a DNS domain name server to be used by the device.

Syntax

name-server *ip-address*

no name-server *ip-address*

Parameters

ip-address

IPv4 address or IPv6 address. Only unicast addresses are supported.

- For an IPv4 address, use IPv4 notation (*A.B.C.D*). For example, 10.0.0.0.
- For an IPv6 address, use the *x:x::x:x* format. For example, 1001::1.

Modes

DNS configuration (config-system-dns)

Usage Guidelines

Use this command to configure the Domain Name System (DNS) setting that specifies the IP address of a domain name server that the device will use. Domain name servers are used for name and address resolution. If no domain name server is configured, the default domain name server is 255.255.255.255 so that the device can broadcast the DNS lookup to the local network segment.

Configuring a domain name and a domain name server for your device makes contacting other devices on your network easier.

A domain name server will respond if one exists in the local network. If not, a server might be present that can forward the DNS request to the correct DNS server.

You can specify one or more domain name servers. You can enter only one domain name server IP address per command usage.

Use the **no** form of this command to remove a domain name server from the device.

Examples

The following example specifies that domain name servers 1.1.1.1 and 2.2.2.2 will be used by the device:

```
device# configure terminal
device(config)# system
device(config-system)# dns
device(config-system-dns)# domain-name mydevice1.com
```

```
device(config-system-dns)# domain-name mydevice2.com
device(config-system-dns)# name-server 1.1.1.1
device(config-system-dns)# name-server 2.2.2.2
device(config-system-dns)#
```

The following example displays the DNS configuration that is running currently on the device. In this example, the device uses two domain name servers:

```
device# show running-config system dns

system
  dns
    domain-name mydevice1.com
    domain-name mydevice2.com
    name-server 1.1.1.1
    name-server 2.2.2.2
  !
!
device#
```

ntp (Extreme ONE OS)

Use the **ntp** command to enter NTP system configuration (config-system-ntp) mode.

Syntax

ntp

Parameters

This command has no keywords or variables.

Modes

System configuration (config-system)

Usage Guidelines

Use this command to enter the mode for enabling Network Time Protocol (NTP) and configuring NTP settings such as the NTP servers and peers. You use NTP to synchronize the clocks over your network.

The NTP service is a global network of public and private servers that provides precise time synchronization to devices worldwide. It works by exchanging time data packets among client and server devices.

This command does not appear in the running configuration unless there are submode parameters configured.

Examples

The following example enters NTP system configuration (config-system-ntp) mode and configures NTP to synchronize the system date and time on the device. In this example, a VRF named red-vrf is used to route NTP communications, IPv4 address 2.2.2.2 points to an NTP server to acquire all system time information, and IPv4 address 3.3.3.3 is a peer with which the local device synchronizes its system time:

```
device# configure terminal
device(config)# system
device(config-system)# ntp
device(config-system-ntp)# enable
device(config-system-ntp)# vrf red-vrf
device(config-system-ntp)# server 2.2.2.2
device(config-system-ntp)# peer 3.3.3.3
device(config-system-ntp)#
```

The following example displays the NTP configuration that is running currently on the device:

```
device# show running-config system ntp

system
ntp
```

```
server 2.2.2.2
peer 3.3.3.3
vrf red-vrf
enable
!
!
device#
```

The following example displays the NTP configuration that is running currently on the device. This is an example of a more complex configuration that uses multiple servers and multiple peers:

```
device# show running-config system ntp

system
 ntp
  enable
  server 0.in.pool.ntp.org
  server 1.in.pool.ntp.org
  server 2.in.pool.ntp.org
  server 1.pool.ntp.org
  server 3.in.pool.ntp.org
  server 2.pool.ntp.org
  server 2.2.2.2
  peer 10.in.pool.ntp.org
  peer 11.in.pool.ntp.org
  peer 12.in.pool.ntp.org
  peer 21.pool.ntp.org
  peer 23.in.pool.ntp.org
  peer 22.pool.ntp.org
  peer 3.3.3.3
!
!
device#
```


password-attributes

Use the **password-attributes** command to enter AAA authentication password attributes system configuration (config-system-aaa-authentication-password-attributes) mode.

Syntax

password-attributes

no password-attributes

Command Default

The password attributes use the default values described in the following section.

Parameters

The following list shows the subcommands that are available in AAA authentication password attributes system configuration (config-system-aaa-authentication-password-attributes) mode. Use these subcommands to set the attributes for the password that you create when you create a new user account:

expiry-alert

(Optional) Enters AAA authentication password attributes expiry system configuration (config-system-aaa-authentication-password-attributes-expiry) mode. **expiry-alert** periods are calculated according to the configured 'max-password-age'. (This subcommand does not appear in the running configuration unless it has submode parameters configured.) In this mode, use the following subcommands to configure the user password expiry (expiration) alert:

- **info** *days*

(Optional) Days before password expiry logging for informational messages. Use an integer from 1 to 90. This value must exceed the critical, major, and minor expiry alert periods.

- **minor** *days*

(Optional) Days before password expiry logging for minor severity messages. Use an integer from 1 to 90. This value must exceed the critical and major expiry alert periods and must be lower than the info expiry alert period.

- **major** *days*

(Optional) Days before password expiry logging for major severity messages. Use an integer from 1 to 90. This value must be lower than the minor and Info expiry alert periods and must exceed the critical expiry alert period.

- **critical** *days*

(Optional) Days before password expiry logging for critical severity messages. Use an integer from 1 to 90. This value must be lower than the major, minor, and info expiry alert periods.

min-length *num-of-chars*

(Optional) Minimum number of characters. The range is 8 to 32. The default value is 8.

min-lowercase *num-of-lowercase-alpha-chars*

(Optional) Minimum number of lowercase alphabetic characters. The range is 0 to 32. The default value is 1.

min-uppercase *num-of-uppercase-alpha-chars*

(Optional) Minimum number of uppercase alphabetic characters. The range is 0 to 32. The default value is 1.

min-numeric *num-of-numeric-chars*

(Optional) Minimum number of numeric characters. The range is 0 to 32. The default value is 1.

min-special *num-of-special-chars*

(Optional) Minimum number of special characters. The range is 0 to 32. The default value is 1.

max-password-age *days*

(Optional) Maximum number of days before a password must be changed. The range is 0 to 999. The default value is 0 (the password never ages).

lockout-duration *minutes*

(Optional) Lockout duration. The valid range is 0 to 99999. The default value is 0 (if a maximum number of password retries is set and then a user exceeds that value, the account remains locked until an administrator unlocks it manually).

max-retry *retries*

(Optional) Maximum number of login retries before the system locks the user account. The valid range is 0 to 16. The default value is 0 (the lockout mechanism is disabled).

Modes

AAA authentication system configuration (config-system-aaa-authentication)

Usage Guidelines

Use this command to enter the mode required for setting the attributes that the { **password** | **password-hashed** } *password* parameter of the **user** command as well as the **username** (GRUB system configuration) command must satisfy. For details, see [user \(AAA authentication system configuration\)](#) on page 387 and [username \(GRUB system configuration\)](#) on page 390 respectively.

This command affects new passwords (not existing passwords).

Use the **no** form of this command to restore all password attributes to their default values.

Use the **no** form of a subcommand to restore the subcommand's password attribute to its default value.

Examples

The following example enters AAA authentication password attributes system configuration (config-system-aaa-authentication-password-attributes) mode and configures user password attributes:

```
device# configure terminal
device(config)# system
device(config-system)# aaa
device(config-system-aaa)# authentication
device(config-system-aaa-authentication)# password-attributes
device(config-system-aaa-authentication-password-attributes)# expiry-alert
device(config-system-aaa-authentication-password-attributes-expiry)# info 15
device(config-system-aaa-authentication-password-attributes-expiry)# minor 10
device(config-system-aaa-authentication-password-attributes-expiry)# major 5
device(config-system-aaa-authentication-password-attributes-expiry)# critical 3
device(config-system-aaa-authentication-password-attributes-expiry)# exit
device(config-system-aaa-authentication-password-attributes)# min-length 10
device(config-system-aaa-authentication-password-attributes)# min-uppercase 2
device(config-system-aaa-authentication-password-attributes)# min-lowercase 3
device(config-system-aaa-authentication-password-attributes)# min-special 2
device(config-system-aaa-authentication-password-attributes)# min-numeric 2
device(config-system-aaa-authentication-password-attributes)# max-password-age 10
device(config-system-aaa-authentication-password-attributes)# max-retry 4
device(config-system-aaa-authentication-password-attributes)# lockout-duration 5
device(config-system-aaa-authentication-password-attributes)# exit
device(config-system-aaa-authentication)#
```

The following example displays the AAA configuration that is running currently on the device. In this example, all of the password attributes are set to nondefault values:

```
device# show running-config system aaa

system
aaa
  authentication
    admin-user admin password-hashed
    $6$10FtgLVtzOEi7jhl$NALdarg9FfowUfWTZUaOnyXG4mehd3cdg3jdGWBFE6uCCSFShPxstb6Mpcd3UJoA16Up19
    6GZ1pn5ilGURdte.
    user user1 role admin password-hashed $6$518667019$y/
    mszgxTh6NWktOfDi7IkWXTTVuHIBWZxKfcWYZbfJMam8Rx4uUyQCQPMXomVLVK7Ojglfkj5u5S.nyq5ev1
    user user2 role user password-hashed $6$388289609$hhtu7WjjWi8RBaNYM2IcI/WWIo.ct/
    Ci56VqA2Yt13DgFotVpoIQxmKX4wWbY2skSLGzMvnBYky0Ywlhs0RuG1
    password-attributes
      min-length 10
      min-lowercase 3
      min-uppercase 2
      min-numeric 2
      min-special 2
      max-password-age 10
      max-retry 4
      lockout-duration 5
      expiry-alert
        info 15
        minor 10
```

```
major 5
critical 3
!
```

peer (NTP system configuration)

Use the **peer** command to specify a peer with which the local device will synchronize its Network Time Protocol (NTP) data.

Syntax

peer *ip-address*

no peer *ip-address*

Command Default

No NTP peer is configured.

Parameters

ip-address

IPv4 address, IPv6 address, or domain name of an NTP peer.

- For an IPv4 address, use IPv4 notation (*A.B.C.D*). For example, use an IPv4 address of 10.0.0.0.
- For an IPv6 address, use the *x::x::x::x* format. For example, 1001::1.

Modes

NTP system configuration (config-system-ntp)

Usage Guidelines

Use this command to specify the IP address or domain name of a peer with which the local device synchronizes its system time on the device. You use NTP to synchronize the clocks over your network.

The NTP service is a global network of public and private servers that provides precise time synchronization to devices worldwide. It works by exchanging time data packets among client and server devices.

You can specify up to eight NTP peers.

The configured peer time is synchronized only after you enable NTP.

Examples

The following example configures NTP to synchronize the system date and time on the device. In this example, a VRF named red-vrf is used to route NTP communications,

IPv4 address 2.2.2.2 points to an NTP server to acquire all system time information, and IPv4 address 3.3.3.3 is a peer with which the local device synchronizes its system time:

```
device# configure terminal
device(config)# system
device(config-system)# ntp
device(config-system-ntp)# enable
device(config-system-ntp)# vrf red-vrf
device(config-system-ntp)# server 2.2.2.2
device(config-system-ntp)# peer 3.3.3.3
device(config-system-ntp)#
```

The following example displays the NTP configuration that is running currently on the device:

```
device# show running-config system ntp

system
  ntp
    enable
    vrf red-vrf
    server 2.2.2.2
    peer 3.3.3.3
  !
!
device#
```

The following example displays the NTP configuration that is running currently on the device. This is an example of a more complex configuration that uses multiple servers and multiple peers:

```
device# show running-config system ntp

system
  ntp
    enable
    server 0.in.pool.ntp.org
    server 1.in.pool.ntp.org
    server 2.in.pool.ntp.org
    server 1.pool.ntp.org
    server 3.in.pool.ntp.org
    server 2.pool.ntp.org
    server 2.2.2.2
    peer 10.in.pool.ntp.org
    peer 11.in.pool.ntp.org
    peer 12.in.pool.ntp.org
    peer 21.pool.ntp.org
    peer 23.in.pool.ntp.org
    peer 22.pool.ntp.org
    peer 3.3.3.3
  !
!
device#
```

port (gRPC server configuration)

Use the **port** command to configure the TCP port number on which the gRPC server instance should listen.

Syntax

port *port-number*

no port

Command Default

The gRPC server instance on the device has no TCP listening port number assigned.

Parameters

port-number

Port number. Use an integer from 1 to 65535.

Modes

gRPC server system configuration (config-system-grpc-server-*name*)

Usage Guidelines

You must disable the gRPC server instance before using this command (or any other command) to alter its configuration. Then, you must re-enable the instance to restore its functionality. To disable or enable a gRPC server instance, see [enable \(gRPC server configuration\)](#) on page 103.

To restore this setting to the default value, use the **no** form of this command.

Examples

The following example configures a gRPC server instance. In this example, the TCP port on which the instance should listen is set to 9340:

```
device# configure terminal
device(config)# system
device(config-system)# grpc-server mygrpcserver
device(config-system-grpc-server-mygrpcserver)# certificate-id test_cert_id
device(config-system-grpc-server-mygrpcserver)# port 9340
device(config-system-grpc-server-mygrpcserver)# mutual-tls
device(config-system-grpc-server-mygrpcserver)# vrf vrf-blue
device(config-system-grpc-server-mygrpcserver)# enable
device(config-system-grpc-server-mygrpcserver)#
```

The following example displays the gRPC server instance configuration that is running currently on the device. In this example, the TCP listening port is set to 9340:

```
device# show running-config system grpc-server

system
  ntp
    server 2.2.2.2
    peer 3.3.3.3
    vrf red-vrf
    enable
  !
!
device#
```


python (Extreme ONE OS)

Use the **python** command to launch an interactive Python3 shell with an option to run a Python script.

Syntax

```
python [ python-statement | python-script-filename ] [ python-script-arguments ]
```

Parameters

python-statement

(Optional) Valid Python interpreter argument. This launches an interactive Python shell and runs this statement. For example, entering `python -h` invokes the Python shell and displays Python options and arguments.

python-script-filename

(Optional) Name of a Python script file. This launches an interactive Python shell and executes the file. Use 4 to 32 characters (including the **.py** extension). The first character must be alphabetic.

python-script-arguments

(Optional) Passes one or more arguments that are defined in the script.

Modes

EXEC (#)

Usage Guidelines

The command uses the privileges of the logged-in user.

Entering **python** with no additional parameters launches the interactive Python shell. To exit the Python environment and return to the CLI, enter **exit()** or **Ctrl-D**.

Python syntax is case sensitive.

To create a Python script, use the Vi IMproved (Vim) editor as follows:

1. Enter the **start-shell** command to enter the interactive Bash shell.
2. Enter the **vi script.py** command to invoke the editor and create the script file.
3. Use the editor to write the Python script.
4. Save the file and exit the editor.
5. Enter **exit** to return to the operating system prompt.

To use the **copy** command to copy a remote Python script to the device, enter the following commands in EXEC (#) mode:

```
copy file scp://username [ :password ] @host: [ port ] filepath/script.py
file disk://dir/script.py
```

To use the **python** command to run a Python script, specify the full path by entering the following command in start-shell:

```
python /var/data/disk/dir/script.py
```

Examples

The following example displays the version of Python that is installed on the device:

```
device# python --version
Python 3.10.14
device#
```

The following example uses the **-m module-name** Python interface option to display the installed version of the Python package manager module (also called pip):

```
device# python -m pip --version
pip 22.0.3 from /usr/lib/python3.10/site-packages/pip (python 3.10)
device#
```

The following example invokes the interactive Python shell and then runs several commands within the shell:

```
device# python
Python 3.10.14 (main, Mar 19 2024, 21:46:16) [GCC 11.4.0] on linux
Type "help", "copyright", "credits" or "license" for more information.
>>> import os
>>> print("Inside python3 interactive shell")
Inside python3 interactive shell
>>> exit()
device#
```

The following example runs a Python script by specifying its full path:

```
device# python /home/admin/sample_python_script.py
device#
```

The following example invokes the Python shell, then assigns an Extreme ONE OS CLI command to a Python variable, and then runs that command:

```
device# python
Python 3.10.14 (main, Mar 19 2024, 21:46:16) [GCC 11.4.0] on linux
Type "help", "copyright", "credits" or "license" for more information.
>>>
>>> import sys
>>> sys.path.insert(1, '/usr/local/tierra/bin/scripts')
>>> from CLI import CLI
>>>
>>> CLI('show users')
!Command: show users
!Time: Fri Mar 7 16:19:45 2025

Active user sessions:
Username                               Role      Host IP
Device      Time Logged In
=====
=====
```

```
admin                               admin      10.37.34.205
CLI                               Mar 7
Total number of active user sessions : 1

Locked users:
Username
=====
Total number of locked users : 0
```

radsec

Use the **radsec** command to enable RADIUS Security (RadSec) on a RADIUS server host.

Syntax

radsec
no radsec

Command Default

Remote Authentication Dial-In User Service (RADIUS) Security is disabled on the specified RADIUS server host.

Parameters

This command has no keywords or variables.

Modes

AAA RADIUS server group server system configuration (config-system-aaa-server-group-radius-server-*address*)

Usage Guidelines

RadSec encrypts RADIUS data (including user credentials) in transit during any communication with the RADIUS server. RADIUS in Extreme ONE OS uses Transport Layer Security (TLS) as the cryptographic protocol.

Use the **no** form of the command to disable RadSec for the current RADIUS server host.

Examples

The following example configures RADIUS server 2.2.2.2 and enables RadSec on the device:

```
device#  
device# configure terminal  
device(config)# system  
device(config-system)# aaa  
device(config-system-aaa)# server-group radius  
device(config-system-aaa-server-group-radius)# server 2.2.2.2  
device(config-system-aaa-server-group-radius-server-2.2.2.2)# radsec  
device(config-system-aaa-server-group-radius-server-2.2.2.2)#
```

The following example displays the RADIUS server configuration with RadSec enabled on the device:

```
device# show running-config system aaa server-group radius
```

```
system
aaa
  server-group radius
  server 2.2.2.2
  radsec
  !
!
!
!
device#
```

receive-lifetime

Use the **receive-lifetime** command to configure the receive lifetime for the key.

Syntax

```
receive-lifetime start-time start-date-and-time [ end-time end-date-and-time ]  
no receive-lifetime
```

Command Default

Received keys are always valid.

Parameters

start-time *start-date-and-time*

Time (expressed in CCYY-MM-DDTHH:MM:SS) after which the key becomes valid to use.

end-time *end-date-and-time*

(Optional) Time (expressed in CCYY-MM-DDTHH:MM:SS) after which the key can no longer be used.

Modes

Keychain key ID configuration (config-key-*key-id*)

Usage Guidelines

UNIX time is the number of elapsed seconds since the UNIX epoch (January 1, 1970 at midnight UTC/GMT), not counting leap seconds (in ISO 8601: 1970-01-01T00:00:00Z). For details about how to determine the current UNIX time or to convert a human-readable time and date to UNIX time, see <https://www.epochconverter.com/>

Use the no form of this command to remove the configured receive lifetime for the key.

Examples

The following example configures the receive lifetime for a key with ID 55:

```
device# configure terminal  
device(config)# keychain key2  
device(config-keychain-key2)# key-id 55  
device(config-keychain-key2-key-55)# receive-lifetime start-time 2022-05-30T20:20:00 end-time 2022-05-31T11:00:00  
device(config-keychain-key2-key-55)#
```

The following example displays the configuration for a keychain named key2 that is running currently on the device. In this example, the validity of the received key with ID

55 is set to start at 1651226048 seconds (in UNIX time) and end at 1651336048 seconds (in UNIX time):

```
device# show running-config keychain key2

keychain key2
  key-id 55
    receive-lifetime start-time 2022-05-30T20:20:00 end-time 2022-05-31T11:00:00
  !
!
device#
```

reload

Use the **reload** command to reboot the device.

Syntax

reload

Parameters

This command has no keywords or variables.

Modes

EXEC (#)

Usage Guidelines

This command restarts the system and reloads the operating system.

Examples

The following example restarts the device and reloads the operating system:

```
device# reload

WARNING: system will be rebooted! Do you want to continue? [y/n]: y
Reloading.... please wait
[51267.077353] _iqpkt_sock_release: Coldboot case, deinit knet DB, set isFwdHalRunning to
false..
duta# [51292.130957] reboot: Restarting system
BIOS 3A04 POST starts...
[11][32][15][19][37][4F][3B][60][61][9A][62][78][68][69][6A][70][71][79][90][92][94][94]
[94][94][94][94][94][95][96][91][92][97]

Processor Type: Intel(R) Atom(TM) CPU C3758 @ 2.20GHz
Processor Speed: 2200 MHz
Total Memory: 16GB (DDR4 2400MHz)

USB Devices total: 0 KBDs, 0 MICE, 0 MASS, 1 HUBs

Detected ATA/ATAPI Devices...
SATA PORT0: KINGSTON OM8POS3128B-A0, S.M.A.R.T Supported
Version 2.19.1266. Copyright (C) 2021 American Megatrends, Inc.
BIOS Date: 06/26/2021 00:36:24 Ver: 3A04
Press <DEL> or <F2> to enter setup.
Press <F11> for BBS POPUP menu.
Press <F12> if you want to boot from the network.

Bootting...Welcome to GRUB!

GNU GRUB version 2.06

+-----+
|*Open Network Linux|
| ONIE|
```



```

Use the ^ and v keys to select which entry is highlighted.
Press enter to boot the selected OS, `e' to edit the commands
before booting or `c' for a command-line.
The highlighted entry will be executed automatically in 0s.
Booting `Open Network Linux'

Loading Open Network Linux ...
initrng: Linux Random Number Generator (RNG) early init
modprobe ixgbe
found interface name alias eth0 --> mal
remapping interface eth0 --> mal
INFO:onl-mounts:Running fsck on ONL-BOOT [ /dev/sda3 ]...
INFO:onl-mounts:/dev/sda3 [ ONL-BOOT ] is clean.
INFO:onl-mounts:Running fsck on ONL-APPDATA [ /dev/sda8 ]...
INFO:onl-mounts:/dev/sda8 [ ONL-APPDATA ] is clean.
INFO:onl-mounts:Running fsck on ONL-SECDATA [ /dev/sda10 ]...
INFO:onl-mounts:/dev/sda10 [ ONL-SECDATA ] is clean.
INFO:onl-mounts:Running fsck on ONL-DATA [ /dev/sda11 ]...
INFO:onl-mounts:/dev/sda11 [ ONL-DATA ] is clean.
INFO:onl-mounts:Running fsck on ONL-IMAGES [ /dev/sda5 ]...
INFO:onl-mounts:/dev/sda5 [ ONL-IMAGES ] is clean.
INFO:onl-mounts:Running fsck on ONL-IAHDATA [ /dev/sda7 ]...
INFO:onl-mounts:/dev/sda7 [ ONL-IAHDATA ] is clean.
INFO:onl-mounts:Running fsck on ONL-USRDATA [ /dev/sda6 ]...
INFO:onl-mounts:/dev/sda6 [ ONL-USRDATA ] is clean.
INFO:onl-mounts:Running fsck on ONL-CONFIG [ /dev/sda4 ]...
INFO:onl-mounts:/dev/sda4 [ ONL-CONFIG ] is clean.
INFO:PKI:Using existing private key.
INFO:PKI:Using existing certificate.
2025-02-06 06:05:14 - hwclock -s executed
Letsssss setup device for migration
*****
*
* Open Network Linux Loader
*
*      Version: ONL-main
*      Id: 2025-01-24.07:54-0099757
*
*      Platform: x86-64-extremenetworks-chassis-8720-32c-r0
*      mal: e4:db:ae:5c:b4:01
*
*****
[ boot-config ]
NETDEV=mal
BOOTMODE=INSTALLED
[ Starting Autoboot ]
[ Configuring Interfaces ]
[ BOOTMODE is installed. ]
[ Trying dir:data:/... ]
creating 1048576k of tmpfs in /tmp/boot-tmpfs-EldtmD
Mounting dir:data:/
DEBUG:swimount:+ blkid

```

```
DEBUG:swimount:+ ubinfo -a
DEBUG:swimount:+ mount -o rw,remount /mnt/onl/data
Switching rootfs
[ 15.649116] Loading IQPKT module...
[ 15.700355] Finished Loading IQPKT module...
[ExtremeOneOS Mgmt IFC Stack]: Success
[ 23.212359] bond0: (slave eth4): invalid new link 3 on slave
```

This system is for authorized users only. All activity is logged and regularly checked by systems personal. Individuals using this system without authority or in excess of their authority are subject to having all their services revoked. Any illegal activities conducted by user or attempts to take down this system or its services will be reported to local law enforcement and said user will be punished to the full extent of the law. Anyone using this system consents to these terms.

duta login: admin

Password:

Last login: Thu Feb 6 01:02:33 EST 2025 on ttyS0

System is ready for all commands

remote-port

Use the **remote-port** command to configure the port number for the remote syslog server for logging on a device.

Syntax

remote-port *port-number*

no remote-port

Command Default

The remote syslog server uses port 514.

Parameters

port-number

Port number on the remote syslog server. Use an integer from 514 to 65535.

Modes

Logging remote server system configuration (config-system-logging-remote-server-*addr-or-domain*)

Usage Guidelines

Use the **no** form of this command to restore the default port number.

Examples

The following example configures logging on a device. In this example, port 525 is specified for a remote syslog server with IP address 1.1.1.1:

```
kvm-x86-64# configure terminal
kvm-x86-64(config)# system
kvm-x86-64(config-system)# logging
kvm-x86-64(config-system-logging)# remote-server 1.1.1.1
kvm-x86-64(config-system-logging-remote-server-1.1.1.1)# remote-port 525
Warning: Existing Host configuration changed
kvm-x86-64(config-system-logging-remote-server-1.1.1.1)# secure-forwarding tls
Warning: Existing Host configuration changed
kvm-x86-64(config-system-logging-remote-server-1.1.1.1)# vrf test1
Warning: Existing Host configuration changed
kvm-x86-64(config-system-logging-remote-server-1.1.1.1)# do show system logging remote-server
```

HOST	PORT	TRANSPORT	SECURE-FORWARDING	VRF-NAME
1.1.1.1	525	UDP	TLS	test1

The following example configures logging on a device. In this example, no remote port is specified for a remote syslog server with IP address 1.1.1.1. Therefore, port 514 (the default) is used:

```
device(config-system-logging-remote-server-1.1.1.1)# no remote-port
Warning: Existing Host configuration changed
device(config-system-logging-remote-server-1.1.1.1)# do show system logging remote-server
```

HOST	PORT	TRANSPORT	SECURE-FORWARDING	VRF-NAME
2.2.2.2	514	UDP	TLS	test1

```
device(config-system-logging-remote-server-1.1.1.1)#
```

remote-server

Use the **remote-server** command to configure the remote server for logging on a device.

Syntax

remote-server *addr-or-domain*

no remote-server *addr-or-domain*

Command Default

No logging remote server is configured.

Parameters

addr-or-domain

IPv4 address, IPv6 address, or fully qualified domain name of the remote server.

- For an IPv4 address, use IPv4 notation (*A.B.C.D*). For example, 10.0.0.0.
- For an IPv6 address, use the *x:x::x:x* format. For example, 1001::1.
- For a domain name, use the hostname, a dot, and the top-level domain (such as myremoteserver.com).

Modes

Logging remote server system configuration (config-system-logging-remote-server-*addr-or-domain*)

Logging system configuration (config-system-logging)

Usage Guidelines

Use the **no** form of this command to restore the default behavior.

Examples

The following example configures logging on a device. In this example, a remote server with IP address 192.0.2.0 is specified:

```
device# configure terminal
device(config)# system
device(config-system)# logging
device(config-system-logging)# remote-server 192.0.2.0
device(config-system-logging-remote-server-192.0.2.0)#
```

The following example shows details about the remote server configuration on the device. In this example, a remote server with IP address 192.0.2.0 is configured:

```
device# show system logging remote-server
```

HOST	PORT	TRANSPORT	SECURE-FORWARDING	VRF-NAME
192.0.2.0	514	UDP	NONE	NONE

```
device#
```

The following example configures logging on a device. In this example, a remote server with IP address 192.0.2.0 is deleted:

```
device(config-system-logging-remote-server-192.0.2.0)# exit
device(config-system-logging)# no remote-server 192.0.2.0
device(config-system-logging)# do show system logging remote-server
No System Logging Hosts found
device(config-system-logging)#
```

rm backup-config

Use the **rm backup-config** command to remove the backup configuration from the device.

Syntax

rm backup-config

Parameters

This command has no keywords or variables.

Modes

EXEC (#)

Usage Guidelines

You can use this command to cancel an active request that would apply the backup configuration upon reboot (done via the **copy file backup-config apply-on-next-reboot** command) by removing the backup configuration. This action disables the pending request to prevent the backup configuration from being applied on the next reboot.

To display the current backup configuration, see [show backup-config](#) on page 221.

Examples

The following example removes the backup configuration from the device:

```
device# rm backup-config
Backup-config removed successfully
device#
```

secret-key (keychain)

Use the **secret-key** (**keychain**) command to configure the secret for the key to authenticate packets.

Syntax

secret-key *secret*

no secret-key

Parameters

secret-key *secret*

Secret key string. The valid range is 1 to 128 characters.

Modes

Keychain key ID configuration (config-key-*key-id*)

Usage Guidelines

When you use the **show running-config** command or the **show running-config keychain** command, the value of the secret displays as a hash.

Use the no form of this command to remove the configured secret for the key.

Examples

The following example configures the secret for a key with ID 33 to authenticate packets:

```
device# configure terminal
device(config)# keychain key1
device(config-keychain-key1)# key-id 33
device(config-keychain-key1-key-33)# secret-key mysecret
device(config-keychain-key1-key-33)#
```

The following example displays the keychain configuration that is running currently on the device. In this example, a secret for packet authentication is configured for a key with ID 33. The secret is displayed as a hash:

```
device# show running-config keychain key1

keychain key1
  key-id 33
  secret-key-hashed
e292kK4qXs32az2yrfkTQKaA2hq4JWF5a4UE25QkdpZzaYFKo1BQ3YaDiEjBunA0UPL14hsKjkz/aQuFWuW9jA==
!
!
device#
```


secret-key (RADIUS server system configuration)

Use the **secret-key** command to configure the shared secret for authentication on a RADIUS server host.

Syntax

secret-key *secret*

no secret-key

Parameters

secret

Secret key string. Use 1 to 40 characters. This string must match the string that is configured on the Remote Authentication Dial-In User Service (RADIUS) server.

Modes

AAA RADIUS server group server system configuration (config-system-aaa-server-group-radius-server-address)

Usage Guidelines

Use the **no** form of this command to remove the configured secret from the key.

Examples

The following example configures RADIUS server 2.2.2.2 with a shared secret key for authentication:

```
device# configure terminal
device(config)# system
device(config-system)# aaa
device(config-system-aaa)# server-group radius
device(config-system-aaa-server-group-radius)# server 2.2.2.2
device(config-system-aaa-server-group-radius-server-2.2.2.2)# secret-key sharedsecret
device(config-system-aaa-server-group-radius-server-2.2.2.2)
```

The following example displays the RADIUS server configuration with a secret key configured on the device:

```
device# show running-config system aaa server-group radius

system
  aaa
    server-group radius
      server 2.2.2.2
        secret-key-hashed
vZhG7vy09QiPuOzlqBY7hAKEU6p28k2Y9OYbXuKKSaXBBLGn5cGS526d55tecsopUWM1OEWu4xPLo2JOverGEg==
!
!
!
```

```
!  
device#
```

**Tip**

The RADIUS secret key displays as a hash in the **show running-config system aaa server-group radius** command output. If you want to unhide the RADIUS secret key so that it displays as plain text, see [unhide-secrets](#) on page 381.

secret-key (TACACS+ server system configuration)

Use the **secret-key** command to configure the shared secret for authentication on a TACACS+ server host.

Syntax

secret-key *secret*

no secret-key

Parameters

secret-key *secret*

Secret key string. The valid range is 1 to 40 characters. This string must match the string that is configured on the Terminal Access Controller Access-Control System Plus (TACACS+) server.

Modes

AAA TACACS+ server group server system configuration (config-system-aaa-server-group-tacacs+-server-address)

Usage Guidelines

Use the **no** form of this command to remove the configured secret from the key.

Examples

The following example configures the TACACS+ server 2.2.2.2 with a shared secret key for authentication:

```
device# configure terminal
device(config)# system
device(config-system)# aaa
device(config-system-aaa)# server-group tacacs+
device(config-system-aaa-server-group-tacacs+)# server 2.2.2.2
device(config-system-aaa-server-group-tacacs+-server-2.2.2.2)# secret-key sharedsecret
device(config-system-aaa-server-group-tacacs+-server-2.2.2.2)#
```

The following example displays the TACACS+ server configuration with a secret key configured on the device:

```
device# show running-config system aaa server-group tacacs+

system
  aaa
    server-group tacacs+
      server 2.2.2.2
        secret-key-hashed
vZhG7vy09QiPuOz1qBY7hAKEU6p28k2Y9OYbXuKKSaXBBLGn5cGS526d55tecsopUWM1OEWu4xPLo2JOwerGEg==
!
!
!
```

```
!  
device#
```

**Tip**

The TACACS+ secret key displays as a hash in the **show running-config system aaa server-group tacacs+** command output. If you want to unhide the TACACS+ secret key so that it displays as plain text, see [unhide-secrets](#) on page 381.

secure-forwarding tls

Use the **secure-forwarding tls** command to enable TLS encryption for secure forwarding of syslogs to a remote syslog server for logging on a device.

Syntax

```
secure-forwarding tls  
no secure-forwarding tls
```

Command Default

Secure forwarding to the current remote syslog server instance is disabled.

Parameters

This command has no keywords or variables.

Modes

Logging remote server system configuration (config-system-logging-remote-server-*addr-or-domain*)

Usage Guidelines

Before using this command, you must install syslog trusted certificate authority (CA) certificates. To do so, see [certificate manager import ca-certificate](#) on page 63.

Use the **no** form of this command to restore the default behavior.

Examples

The following example configures logging on a device. In this example, TLS encryption for secure forwarding of syslogs to remote syslog server 1.1.1.1 is enabled:

```
device# configure terminal  
device(config)# system  
device(config-system)# logging  
device(config-system-logging)# remote-server 1.1.1.1  
device(config-system-logging-remote-server-1.1.1.1)# secure-forwarding tls  
Warning: Existing Host configuration changed  
device(config-system-logging-remote-server-1.1.1.1)# tls-profile-id spl  
Warning: Existing Host configuration changed  
device(config-system-logging-remote-server-1.1.1.1)#
```

The following example configures logging on a device. In this example, TLS encryption for secure forwarding of syslogs to remote syslog server 1.1.1.1 is enabled:

```
device# show running-config system logging remote-server  
  
system  
logging
```

```
remote-server 1.1.1.1
  secure-forwarding tls
  tls-profile-id sp1
!
!
!
```

The following example restores the logging configuration to the default value (disabled) on a device. In this example, TLS encryption for secure forwarding of syslogs to remote syslog server 1.1.1.1 is disabled:

```
device(config-system-logging-remote-server-1.1.1.1)# no secure-forwarding tls
Warning: Existing Host configuration changed
device(config-system-logging-remote-server-1.1.1.1)# do show system logging remote-server
HOST          PORT  TRANSPORT  SECURE-FORWARDING  VRF-NAME
=====
1.1.1.1       514   UDP        NONE               NONE
device(config-system-logging-remote-server-1.1.1.1)#
```

send-lifetime

Use the **send-lifetime** command to configure the send lifetime for the key.

Syntax

```
send-lifetime start-time start-time [ end-time end-time ] [ send-receive { true | false } ]  
no send-lifetime
```

Command Default

Sent keys are always valid.

Parameters

start-time *start-time*

Date and time (CCYY-MM-DDTHH:MM:SS) at which the key becomes valid to use.

end-time *end-time*

(Optional) Date and time (CCYY-MM-DDTHH:MM:SS) after which the key becomes invalid to use. The end-time value must be after the start-time value.

send-receive { **true** | **false** }

(Optional) When send-receive is set to true, the send lifetime can be used in the receive direction. When set to false, the configuration becomes asymmetric. The default value is true.

Modes

Keychain key ID configuration (config-key-*key-id*)

Usage Guidelines

Use the no form of this command to remove the configured send lifetime for the key.

Examples

The following example configures the send lifetime for the key:

```
device# configure terminal  
device(config)# keychain key1  
device(config-keychain-key1)# key-id 22  
device(config-keychain-key1-key-22)# send-lifetime start-time 2022-05-31T11:00:00 end-time 2022-05-31T11:00:10 send-receive true  
device(config-key-22)#
```

The following example displays the configuration for a keychain named key1 that is running currently on the device:

```
device#show running-config keychain key1

keychain key1
  key-id 22
    send-lifetime start-time 2022-05-31T11:00:00 end-time 2022-05-31T11:00:10 send-
receive true
  !
!
device#
```


server-group

Use the **server-group** command to enter one of the command modes for configuring a AAA server group.

Syntax

```
server-group { ldap | radius | tacacs+ }
```

Command Default

The device has no Authentication, Authorization, and Accounting (AAA) server group configured on the device.

Parameters

ldap

Enters the mode for configuring a Lightweight Directory Access Protocol (LDAP) server group.

radius

Enters the mode for configuring a Remote Authentication Dial-In User Service (RADIUS) server group.

tacacs+

Enters the mode for configuring a Terminal Access Controller Access-Control System plus (TACACS+) server group.

Modes

AAA system configuration (config-system-aaa)

Usage Guidelines

This command enters the mode that is required for configuring the servers in a AAA server group on a device. Configuration of a server includes the following settings:

- IPv4 address or IPv6 address of the server host
- Source interface
- VRF instance within which the server is listening
- (LDAP only) Role mapping, base domain name, and LDAP over SSL/TLS (LDAPS)
- (RADIUS and TACACS+) Secret key for authentication on the server host
- (RADIUS only) RADIUS security (also called RadSec or RADIUS over TLS)

Examples

The following example configures AAA on the device. The example uses the **server-group** command with the **tacacs+** keyword to enter AAA TACACS+ server group

system configuration (config-system-aaa-server-group-tacacs+) mode and configure a TACACS+ server group with one server (which has IPv4 address 2.2.2.2):

```
device# configure terminal
device(config)# system
device(config-system)# aaa
device(config-system-aaa)# server-group tacacs+
device(config-system-aaa-server-group-tacacs+)# server 2.2.2.2
device(config-system-aaa-server-group-tacacs+-server-2.2.2.2)# secret-key sharedsecret
device(config-system-aaa-server-group-tacacs+-server-2.2.2.2)#
```

server (LDAP server system configuration)

Use the **server** command to specify the IPv4 or IPv6 address of an LDAP server in a LDAP server group and enter AAA LDAP server group server system configuration (config-system-aaa-server-group-ldap-server-address) mode.

Syntax

server *ip-address*

no server *ip-address*

Command Default

No Lightweight Directory Access Protocol (LDAP) server is specified for the server group.

Parameters

ip-address

IPv4 or IPv6 address of the LDAP server. Only valid unicast IPv4 and IPv6 addresses are supported. Broadcast IPv4 addresses and multicast IP addresses are not supported.

Modes

AAA LDAP server group system configuration (config-system-aaa-server-group-ldap)

Usage Guidelines

Use the **no** form of this command to remove the LDAP server configuration.

Examples

The following example adds a server with IPv4 address 3.3.3.3 to the LDAP server group on the device and enters AAA LDAP server group server system configuration (config-system-aaa-server-group-ldap-server-address) mode for configuring the server:

```
device# configure terminal
device(config)# system
device(config-system)# aaa
device(config-system-aaa)# server-group ldap
device(config-system-aaa-server-group-ldap)# server 3.3.3.3
device(config-system-aaa-server-group-ldap-server-3.3.3.3)# base-dn example.com
device(config-system-aaa-server-group-ldap-server-3.3.3.3)# ldaps
device(config-system-aaa-server-group-ldap-server-3.3.3.3)#
```

The following example displays the LDAP server group configuration that is running currently on the device. In this example, the server group is configured with one server:

```
device# show running-config system aaa server-group ldap

system
  aaa
    server-group ldap
      server 3.3.3.3
        base-dn example.com
        ldaps
      !
    !
  !
!
device#
```

server (NTP system configuration)

Use the **server** command to specify an external server from which to acquire all Network Time Protocol (NTP) system time data.

Syntax

server *address*

no server *address*

Command Default

No external server is specified.

Parameters

address

IPv4 address, IPv6 address, or domain name of an NTP server.

- For an IPv4 address, use IPv4 notation (*A.B.C.D*). For example, use an IPv4 address of 10.0.0.0.
- For an IPv6 address, use the *x::x::x::x* format. For example, 1001::1.

Modes

NTP system configuration (config-system-ntp)

Usage Guidelines

Use this command to specify the IP address or domain name of an NTP server from which to acquire all system time information for the device. You use NTP to synchronize the clocks over your network.

The NTP service is a global network of public and private servers that provides precise time synchronization to devices worldwide. It works by exchanging time data packets among client and server devices.

You can specify up to eight NTP servers.

The configured server time is synchronized only after you enable NTP.

Examples

The following example configures NTP to synchronize the system date and time on the device. In this example, a VRF named red-vrf is used to route NTP communications,

IPv4 address 2.2.2.2 points to an NTP server to acquire all system time information, and IPv6 address 3.3.3.3 is a peer with which the local device synchronizes its system time:

```
device# configure terminal
device(config)# system
device(config-system)# ntp
device(config-system-ntp)# enable
device(config-system-ntp)# vrf red-vrf
device(config-system-ntp)# server 2.2.2.2
device(config-system-ntp)# peer 3.3.3.3
device(config-system-ntp)#
```

The following example displays the NTP configuration that is running currently on the device:

```
device# show running-config system ntp

system
 ntp
  server 2.2.2.2
  peer 3.3.3.3
  vrf red-vrf
  enable
!
!
device#
```

The following example displays the NTP configuration that is running currently on the device. This is an example of a more complex configuration that uses multiple servers and multiple peers:

```
device# show running-config system ntp

system
 ntp
  enable
  server 0.in.pool.ntp.org
  server 1.in.pool.ntp.org
  server 2.in.pool.ntp.org
  server 1.pool.ntp.org
  server 3.in.pool.ntp.org
  server 2.pool.ntp.org
  server 2.2.2.2
  peer 10.in.pool.ntp.org
  peer 11.in.pool.ntp.org
  peer 12.in.pool.ntp.org
  peer 21.pool.ntp.org
  peer 23.in.pool.ntp.org
  peer 22.pool.ntp.org
  peer 3.3.3.3
!
!
device#
```

server (RADIUS server system configuration)

Use the **server** command to specify the IPv4 or IPv6 address of a RADIUS server in a RADIUS server group and enter AAA RADIUS server group server system configuration (config-system-aaa-server-group-radius-server-*address*) mode.

Syntax

server *ip-address*

Command Default

No Remote Authentication Dial-In User Service (RADIUS) server host is specified for the server group.

Parameters

ip-address

IPv4 or IPv6 address of the RADIUS server. Only valid unicast IPv4 and IPv6 addresses are supported. Broadcast IPv4 addresses and multicast IP addresses are not supported.

Modes

AAA RADIUS server group system configuration (config-system-aaa-server-group-radius)

Usage Guidelines

Use the **no** form of this command to remove the RADIUS server configuration.

Examples

The following example adds a server with IPv4 address 2.2.2.2 to the RADIUS server group on the device and enters AAA RADIUS server group server system configuration (config-system-aaa-server-group-radius-server-*address*) mode for configuring the server:

```
device# configure terminal
device(config)# system
device(config-system)# aaa
device(config-system-aaa)# server-group radius
device(config-system-aaa-server-group-radius)# server 2.2.2.2
device(config-system-aaa-server-group-radius-server-2.2.2.2)#
```

The following example displays the RADIUS server group configuration that is running currently on the device. In this example, the server group is configured with one server:

```
device# show running-config system aaa server-group radius
```

```
system
  aaa
    server-group radius
    server 2.2.2.2
  !
!
!
device#
```


server (TACACS+ server system configuration)

Use the **server** command to specify the IPv4 or IPv6 address of a TACACS+ server in a TACACS+ server group and enter AAA TACACS+ server group server system configuration (config-system-aaa-server-group-tacacs+-server-address) mode.

Syntax

```
server ip-address  
no server ip-address
```

Command Default

No Terminal Access Controller Access-Control System Plus (TACACS+) server is specified for the server group.

Parameters

ip-address

IPv4 or IPv6 address of the TACACS+ server. Only valid unicast IPv4 and IPv6 addresses are supported. Broadcast IPv4 addresses and multicast IP addresses are not supported.

Modes

AAA TACACS+ server group system configuration (config-system-aaa-server-group-tacacs+)

Usage Guidelines

Use the **no** form of this command to remove the TACACS+ server configuration.

Examples

The following example adds a server with IPv4 address 2.2.2.2 to the TACACS+ server group on the device and enters AAA TACACS+ server group server system configuration (config-system-aaa-server-group-tacacs+-server-address) mode for configuring the server:

```
device# configure terminal  
device(config)# system  
device(config-system)# aaa  
device(config-system-aaa)# server-group tacacs+  
device(config-system-aaa-server-group-tacacs+)# server 2.2.2.2  
device(config-system-aaa-server-group-tacacs+-server-2.2.2.2)#
```

The following example displays the TACACS+ server group configuration that is running currently on the device. In this example, the server group is configured with one server:

```
device# show running-config system aaa server-group tacacs+

system
  aaa
    server-group tacacs+
      server 2.2.2.2
    !
  !
!
device#
```

service (TLS system configuration)

Use the **service** command to specify a TLS-enabled client/server service and enter TLS service-level system configuration (`config-system-tls-service-name`) mode.

Syntax

service *service-name*

no service *service-name*

Command Default

Service-level Transport Layer Security (TLS) settings are not configured on the device.

Parameters

service-name

Service name. Use one of the following values:

- **syslog-client**—The System Logging Service (Syslog) client, when configured with the secure-forwarding TLS option, uses TLS.
- **ldap-client**—The LDAP client, when configured with the **ldaps** option, uses TLS. When a TLS minimum version is configured, new connections will follow that version. LDAPS authentication is invoked during SSH login and gNMI user authentication. It uses the configured minimum version during such attempts.
- **radius-client**—The RADIUS client, when configured with the **radsec** option, uses TLS. When a TLS minimum version is configured, new connections will follow that version. RADSEC authentication is invoked during SSH login and gNMI user authentication. It uses the configured minimum version during such attempts.
- **grpc-server**—The gRPC server using TLS will get configured with the TLS minimum version only after an explicit enable/disable of the instance. The existing connections are not affected, and new connections (after gRPC server enable/disable) will follow the configured TLS minimum version.
- **https-client**—When a firmware download is triggered via an HTTPS URL, it is over TLS. The configured TLS minimum version is used for the firmware download. A certificate authority (CA) certificate should be imported using `ssl-reserved-https` as the SSL profile ID. This SSL profile stores the CA certificate necessary to perform firmware updates using HTTPS. For details, see [certificate manager import ca-certificate](#) on page 63.

Modes

TLS system configuration (`config-system-tls`)

Usage Guidelines

This command enters the mode required for configuring the TLS required minimum version for a specific service that uses TLS.

To restore the default behavior, use the **no** form of this command.

This command does not appear in the running configuration unless there are CLI attributes already configured.

Examples

The following example enters the TLS service-level system configuration (config-system-tls-service-*name*) mode for configuring TLS for the gRPC server service on the device:

```
device# configure terminal
device(config)# system
device(config-system)# tls
device(config-system-tls)# service grpc-server
device(config-system-tls-service-grpc-server)#
```

show backup-config

Use the **show backup-config** command to display the backup configuration on a device.

Syntax

```
show backup-config
```

Parameters

This command has no keywords or variables.

Usage Guidelines

This command displays the configuration that will result when you use the backup configuration to restore the device to its normal state. This configuration is useful for troubleshooting or system recovery if the device is reset unintentionally or the configuration database becomes corrupted.

The device maintains at most one backup configuration at a time. When you create a backup configuration, any existing backup configuration is overwritten. This ensures that only the most recent user-created backup configuration is preserved.

A backup configuration is stored in a plain-text ASCII file that resembles the output of the **show running-config** command.



Note

To show the replay status in which the system is using the current backup-config file to restore the running configuration automatically, see [show backup-config replay-status](#) on page 223.

Examples

The following example displays the backup configuration on the device:

```
device# show backup-config

system
  grub
    username root password-hashed
grub.pbkdf2.sha512.10000.AA8C315154428264D5474C322A7F187F120BFB57EDBACB78CBC130B9915E0E9EC
CB615D045F3CD05C39DF7767C7645A279A19
5928ABD9BE5422E096563435D0C.AFBBFB72FC886A16C850D22181913B7E9A100EA71A45583EABA0E7C8A41904
A3B553B42363AFABB919621093C1B46AA0E54133F47E60729C242649F2B8074880
!
aaa
  authentication
    admin-user admin password-hashed
$6$D5Bb9D3t8sKdxflb$rZIwfTkhh33dEvbiMSyqff.zAk.Z2xSYz3dmaA.GQooCbBeDU4Iy0CXZMaXwflyTpK1Q.M
GHmUz.F1/IjdHa..
!
!
```

```
grpc-server DEFAULT
ssh-server DEFAULT
  key rsa 4096
  vrf mgmt-vrf
  enable
!
!
interface management 0
  ipv4 address dhcp
  no shutdown
!
vrf mgmt-vrf
  member management 0
!
device#
```

show backup-config replay-status

Use the **show backup-config replay-status** command to display the status or progress of the automated backup configuration replay on a device.

Syntax

```
show backup-config replay-status
```

Parameters

This command has no keywords or variables.

Usage Guidelines

This command shows the replay status in which the system is using the current backup-config file to restore the running configuration automatically. An automated backup configuration replay occurs when the system reboots following usage of the apply-on-next-reboot keyword of either the **copy file backup-config** command or the **copy running-config backup-config** command. For details, see [copy file backup-config](#) on page 89 or [copy running-config backup-config](#) on page 95 respectively.

The backup configuration is used to restore the device to its normal state. This configuration is useful for troubleshooting or system recovery if the device is reset unintentionally or the configuration database becomes corrupted.

The device maintains at most one backup configuration at a time. When you create a backup configuration, any existing backup configuration is overwritten. This ensures that only the most recent user-created backup configuration is preserved.

A backup configuration is stored in a plain-text ASCII file that resembles the output of the **show running-config** command.



Note

This command does *not* show the current status or progress of a manually triggered backup-config replay.

To see the actual configuration that will result if the backup configuration is used to restore the device to its normal state (either automatically or manually triggered), see [show backup-config](#) on page 221.

Examples

The following example displays the backup configuration replay status on the device. In this example, there is no active replay:

```
device# show backup-config replay-status
```

```
No backup config replay in progress
device#
```

The following example displays the backup configuration replay status on the device. In this example, the backup configuration replay is triggered but has not run long enough to provide a detailed status or progress:

```
device# show backup-configuration replay-status

Backup config replay is initializing, detailed status not yet available
device#
```

The following example displays the backup configuration replay status on the device. In this example, the replay is ten percent complete and will finish in an estimated 135 seconds:

```
device# show backup-configuration replay-status

Replay in progress: 10.00% (125/1250 lines) completed
Elapsed time: 15 seconds
Estimated time remaining: 135 seconds
device#
```


show bmc-status (Extreme ONE OS)

Use the **show bmc-status** command to display details about the internal baseboard management controller (BMC) LAN configuration on a device.

Syntax

```
show bmc-status
```

Parameters

This command has no keywords or variables.

Modes

EXEC (#)

Usage Guidelines

The BMC is a microcontroller and the major component of the Intelligent Platform Management Interface (IPMI) subsystem, which assists in remote hardware monitoring, management, and troubleshooting of the device. Use this command to display details about the current internal BMC Ethernet port configuration such as the IP address source, IP address, subnet mask, MAC address, and default gateway IP address.

This command does not display output if a BMC configuration change is in progress. The settings for the IP address, subnet mask, default gateway IP address, shutdown state, or reset require up to one minute each for the change to take effect on the BMC hardware. This delay is an inherent hardware attribute of a BMC.

Examples

The following example displays the details about the internal BMC LAN configuration on the device:

```
device# show bmc-status

IP Address Source      : DHCP Address
IP Address             : 0.0.0.0
Subnet Mask            : 0.0.0.0
MAC Address            : b4:a3:bd:0d:88:07
Default Gateway IP     : 0.0.0.0
Firmware Revision      : 1.11
device#
```

show certificate-manager app-certificate

Use the **show certificate-manager app-certificate** command to display details about the application certificates in the system certificates store.

Syntax

```
show certificate-manager app-certificate { ssl-profile-id profile-name | all }
```

Parameters

ssl-profile-id *profile-name*

Name of an SSL profile containing the imported application certificate. Use 1 to 64 characters.

all

Displays details for all application certificates in the certificate store.

Modes

EXEC (#)

Usage Guidelines

Certificates in use on the device are monitored for their validity. These include application certificates (and certificate authority (CA) certificates). The device tracks the SSL profiles in use by various applications (such as the gRPC server, remote authentication clients (such as LDAP and RADIUS), logging clients (syslog), and the token validator (for validating JWT token included in gNMI requests). This command shows these endpoints when it displays the certificates.

This command displays information about each application certificate installed on the device as well as its status (the time period during which it is valid). This information includes the certificate ID, country name (C), state or province name (ST), locality (L), organization (O), organizational unit (OU), and common name (CN) for both the certificate issuer and the subject (the entity that is being secured, such as a website or an individual).

Examples

The following example shows details about a specific application certificate SSL profile named `sp1` in the system certificates store:

```
device# show certificate-manager app-certificate ssl-profile-id sp1

App level certificates:
certificate-id: sp1
Endpoints using this certificate-id:[type:EP_DAEMON  endpoint:"grpc-server DEFAULT"]
sha256
Fingerprint=AF:21:61:D0:15:A3:32:07:7D:11:C2:D8:E6:85:61:8B:43:A2:52:C1:95:54:BE:5B:0F:CC:
```

```

93:9D:DE:E3:23:BC
subject=C=US, ST=CA, O=Extreme Networks, OU=Extreme ONE OS switching and Routing,
CN=extremenetworks.com
issuer=C=US, ST=CA, O=Extreme Networks, OU=Extreme ONE OS switching and Routing,
CN=extremenetworks.com
notBefore=Mar 19 06:12:42 2025 GMT
notAfter=Mar 17 06:12:42 2035 GMT
device#

```

The following example shows details about all imported and default application certificate SSL profiles in the system certificates store:

```

device# show certificate-manager app-certificate all

App level certificates:
certificate-id:sp1
Endpoints using this certificate-id:[type:EP_DAEMON  endpoint:"grpc-server DEFAULT"]
sha256
Fingerprint=AF:21:61:D0:15:A3:32:07:7D:11:C2:D8:E6:85:61:8B:43:A2:52:C1:95:54:BE:5B:0F:CC:
93:9D:DE:E3:23:BC
subject=C=US, ST=CA, O=Extreme Networks, OU=Extreme ONE OS switching and Routing,
CN=extremenetworks.com
issuer=C=US, ST=CA, O=Extreme Networks, OU=Extreme ONE OS switching and Routing,
CN=extremenetworks.com
notBefore=Mar 19 06:12:42 2025 GMT
notAfter=Mar 17 06:12:42 2035 GMT

certificate-id:sp2
Endpoints using this certificate-id:[type:EP_DAEMON  endpoint:"token-validator DEFAULT"]
sha256
Fingerprint=AF:21:61:D0:15:A3:32:07:7D:11:C2:D8:E6:85:61:8B:43:A2:52:C1:95:54:BE:5B:0F:CC:
93:9D:DE:E3:23:BC
subject=C=US, ST=CA, O=Extreme Networks, OU=Extreme ONE OS switching and Routing,
CN=extremenetworks.com
issuer=C=US, ST=CA, O=Extreme Networks, OU=Extreme ONE OS switching and Routing,
CN=extremenetworks.com
notBefore=Mar 19 06:12:42 2025 GMT
notAfter=Mar 17 06:12:42 2035 GMT

certificate-id:ssl-reserved-generated
Endpoints using this certificate-id:[type:EP_DAEMON  endpoint:"grpc-server DEFAULT"]
sha256
Fingerprint=58:D7:AF:A2:2E:B3:10:4B:16:75:C2:7C:2A:64:49:05:44:B6:51:07:DD:14:79:01:D6:B3:
44:1E:2B:E9:86:AA
X509v3 Subject Alternative Name:
  IP Address:127.0.0.1, IP Address:1.1.1.1
subject=CN=extreme.extremenetworks.com
issuer=C=US, ST=CA, L=SJ, O=Extreme Networks, OU=Extreme Networks Pvt Ltd,
CN=extreme.extremenetworks.com, emailAddress=support@extremenetworks.com intermediate
notBefore=Jun 20 11:30:54 2025 GMT
notAfter=Jun 19 11:30:54 2028 GMT
device#

```

show certificate-manager ca-certificate

Use the **show certificate-manager ca-certificate** command to display details about the CA certificates in the system certificates store.

Syntax

```
show certificate-manager ca-certificate { ssl-profile-id profile-name | all }
```

Parameters

ssl-profile-id *profile-name*

Name of an SSL profile containing an imported certificate authority (CA) certificate. Use 1 to 64 characters.

all

Displays details for all CA certificates in the certificate store.

Modes

EXEC (#)

Usage Guidelines

Certificates in use on the device are monitored for their validity. These include certificate authority (CA) certificates (and application certificates). The device tracks the SSL profiles in use by various applications (such as the gRPC server, remote authentication clients (such as LDAP and RADIUS), logging clients (syslog), and the token validator (for validating JWT token included in gNMI requests). This command shows these endpoints when it displays the certificates.

This command displays information about each CA certificate installed on the device as well as its status (time period during which it is valid). This information includes the certificate ID, country name (C), state or province name (ST), locality (L), organization (O), organizational unit (OU), and common name (CN) for both the certificate issuer and the subject (the entity that is being secured, such as a website or an individual).

Examples

The following example shows details about a specific CA certificate SSL profile named sp2 in the system certificates store:

```
device# show certificate-manager ca-certificate ssl-profile-id sp2

CA certificates:
certificate-id: sp2
Endpoints using this certificate-id:[type:EP_DAEMON  endpoint:"token-validator DEFAULT,
server-group radius 1.1.1.1"]
sha256
FingerPrint=AF:21:61:D0:15:A3:32:07:7D:11:C2:D8:E6:85:61:8B:43:A2:52:C1:95:54:BE:5B:0F:CC:
```

```

93:9D:DE:E3:23:BC
subject=CN=extremenetworks.com,OU=Extreme ONE OS switching and Routing,O=Extreme
Networks,ST=CA,C=US
issuer=CN=extremenetworks.com,OU=Extreme ONE OS switching and Routing,O=Extreme
Networks,ST=CA,C=US
notBefore=Mar 19 06:12:42 2025 UTC
notAfter=Mar 17 06:12:42 2035 UTC
device#

```

The following example shows details about all imported and default CA certificate SSL profiles in the system certificates store:

```

device# show certificate-manager ca-certificate all

CA certificates:
certificate-id:sp1
Endpoints using this certificate-id:[type:EP_DAEMON endpoint:"grpc-server DEFAULT"]
sha256
FingerPrint=AF:21:61:D0:15:A3:32:07:7D:11:C2:D8:E6:85:61:8B:43:A2:52:C1:95:54:BE:5B:0F:CC:
93:9D:DE:E3:23:BC
subject=CN=extremenetworks.com,OU=Extreme ONE OS switching and Routing,O=Extreme
Networks,ST=CA,C=US
issuer=CN=extremenetworks.com,OU=Extreme ONE OS switching and Routing,O=Extreme
Networks,ST=CA,C=US
notBefore=Mar 19 06:12:42 2025 UTC
notAfter=Mar 17 06:12:42 2035 UTC

certificate-id:sp2
Endpoints using this certificate-id:[type:EP_DAEMON endpoint:"token-validator DEFAULT,
server-group radius 1.1.1.1"]
sha256
FingerPrint=AF:21:61:D0:15:A3:32:07:7D:11:C2:D8:E6:85:61:8B:43:A2:52:C1:95:54:BE:5B:0F:CC:
93:9D:DE:E3:23:BC
subject=CN=extremenetworks.com,OU=Extreme ONE OS switching and Routing,O=Extreme
Networks,ST=CA,C=US
issuer=CN=extremenetworks.com,OU=Extreme ONE OS switching and Routing,O=Extreme
Networks,ST=CA,C=US
notBefore=Mar 19 06:12:42 2025 UTC
notAfter=Mar 17 06:12:42 2035 UTC

certificate-id:tls
Endpoints using this certificate-id:[type:EP_DAEMON endpoint:"NONE"]
sha256
FingerPrint=7B:90:00:77:E7:92:76:01:9D:02:29:DD:96:4A:A6:40:13:D5:EE:FB:59:61:66:57:0D:AB:
09:AF:B3:2F:5E:DB
subject=CN=CA-Cert,O=extreme
issuer=CN=CA-Cert,O=extreme
notBefore=May 2 07:39:35 2025 UTC
notAfter=May 2 07:39:35 2026 UTC

certificate-id:mtlscert
Endpoints using this certificate-id:[type:EP_DAEMON endpoint:"NONE"]
sha256
FingerPrint=E9:5D:5F:9A:D9:BB:27:CA:63:AA:76:DD:3C:53:6F:43:3A:3C:E4:6A:71:C0:E0:EC:F3:5E:
71:F3:2C:56:E7:C4
subject=CN=extremenetworks-CA
issuer=CN=extremenetworks-CA
notBefore=Jun 20 09:01:56 2025 UTC
notAfter=Jun 20 09:01:56 2026 UTC
device#

```

show cli history

Use the **show cli history** command to display a list of the commands that you have entered. The show cli history command is restricted to admin role users only. This command displays the complete CLI history of a user. The history is presented as a paginated list, starting from the oldest commands and progressing to the latest commands. The most recent commands will appear at the end of the list

Syntax

show cli history

Parameters

This command has no keywords or variables.

Modes

EXEC (#)

Usage Guidelines

Use this command to recall and reuse previous commands quickly. This is helpful for tasks such as troubleshooting or configuring similar devices.

The number of commands that is retained in the command history is unlimited. The output is paginated for a command history that exceeds one page.

Examples

The following example displays the output of the **show cli history** command:

```
device# show cli history

sh running-config
sh cli history
conf
interface ethernet 0/1
description this is a configuration
subinterface vlan 1
no shutdown
exit
device#
```

show clock

Use the **show clock** command to display the current system date, time, and timezone on the device.

Syntax

```
show clock
```

Parameters

This command has no keywords or variables.

Modes

EXEC (#)

Usage Guidelines

To set the system time on the device, see [clock set](#) on page 77. To set the system timezone on the device, see [clock timezone](#) on page 78.

Examples

The following example shows the current time:

```
device# show clock  
  
2025-11-07 21:51:00 IST +0530  
device#
```

show counters interface management

Use the **show counters interface management** command to display statistics for the management interface.

Syntax

```
show counters interface management interface-name
```

Parameters

interface-name

Management interface name. The only supported value is 0 in this release.

Modes

EXEC (#)

Usage Guidelines

You must clear the counters if you want to start at zero for more recent statistics. For details, see [clear counters interface management](#) on page 69.

Output

The **show counters interface management** command displays the following information:

Output field	Description
LastClear	Time elapsed since the counters that measure cumulative statistics (such as number of bytes transmitted and received) shown in this report were last reset to zero (such as when the device was rebooted). Note that variables that might affect routing (for example, load and reliability) are not cleared when the counters are cleared.
Input	Statistics for data received on the interface: • Total pkts—Total number of packets of all types received by the interface. • Broadcast pkts—Total number of broadcast packets received by the interface. • Discard pkts—Number of input packets that were discarded silently (possibly because of buffer overflow). • Errors pkts—Total number of no buffer, runs, giants, CRC, frame, overrun, ignored, and abort counts. Other input-related errors can also increment the count, so that this sum might not balance with the other counts. • CRC Errors—Number of times that the cyclic redundancy checksum (CRC) generated by the originating LAN station or remote device does not match the checksum calculated from the data received. On a LAN, this usually means noise or transmission problems on the LAN interface or the LAN bus. A high number is usually from collisions or a station transmitting bad data. On a serial link, CRCs usually indicate noise, gain hits, or other transmission problems on the data link. • MCast pkts—Total number of multicast packets received by the interface. • Octets—The number of octets (bytes) received by the interface.

Output field	Description
Out	Counts of packets sent by the interface: • Total pkts—Total number of packets of all types sent by the interface. • Discard pkts—Number of output packets that were silently discarded (possibly because of buffer overflow). • Errors pkts—Total number of no buffer, runs, giants, CRC, frame, overrun, ignored, and abort counts. Other input-related errors can also increment the count, so that this sum might not balance with the other counts. • MCast pkts—Total number of multicast packets sent by the interface. • Octets—The number of octets ((bytes) sent by the interface.
Rate Info	The rate of traffic or packets on that interface that are ingressing or egressing versus the theoretical maximum that the interface speed can support: • Input—For example, 0.006331 Mb/s, 7 pkts/sec 0.00% of the line rate is how much traffic is incoming to the interface. • Output—For example, 0.003645 Mb/s, 3 pkts/sec 0.00% of the line rate is how much traffic is leaving the device on the interface.

Examples

The following example displays statistics for the management interface:

```

device# show counters interface management 0

Statistics
    LastClear: 0s
Input:
    Total pkts: 2709182
    Discard pkts: 10
    Errors pkts: 0
    CRC Errors: 0
    MCast pkts: 2672357
    Octets: 151348889
Out:
    Total pkts: 17504
    Discard pkts: 0
    Errors pkts: 0
    MCast pkts: 0
    Octets: 2351000
Rate Info:
    Input: 0.003457 Mb/s, 6 pkts/sec 1000000000 speed 0.00% of line-
rate
    Output: 0.000651 Mb/s, 1 pkts/sec 1000000000 speed 0.00% of line-
rate
device#

```

show firmware (Extreme ONE OS)

Use the **show firmware** command to display details about the firmware versions and commit status on a device.

Syntax

```
show firmware
```

Parameters

This command has no keywords or variables.

Modes

EXEC (#)

Usage Guidelines

This command displays the following information about the firmware on the device:

- Extreme ONE OS base OS) current and rollback versions
- Complex Programmable Logic Device (CPLD) versions. The number of CPLDs varies with the platform
- Firmware commit status
- Current and rollback versions of installed applications (such as Extreme ONE OS Switching)

Examples

The following example displays the details about the firmware versions and commit status on the device:

```
device# show firmware

Base OS:
Current Version:          10.2.1.0
Rollback Version:         None
Is Current Version Committed: True

Switching Routing (SR):
Current Version:          22.2.1.0
Rollback Version:         None
Is Current Version Committed: True

System CPLD Version:      Port-CPLD#0:06, Port-CPLD#1:06, Power-CPLD:09
device#
```

show firmware history (Extreme ONE OS)

Use the **show firmware history** command to display the history of the recent firmware versions running on a device.

Syntax

```
show firmware history
```

Parameters

This command has no keywords or variables.

Modes

EXEC (#)

Usage Guidelines

This command displays the version and installation date and time of the last five firmware versions running on a device.

Examples

The following example displays the history of the recent firmware versions running on the device:

```
device# show firmware history

Firmware Version                Install Date
-----
ExtremeOneBase-10.1.6.0         Fri, 07 Mar 2025 21:47:16 IST
ExtremeOneBase-10.1.6.0-008     Fri, 07 Mar 2025 20:53:46 IST
device#
```

show grpc-server gnmi capabilities

Use the **show grpc-server gnmi capabilities** command to display the details of supported YANG data models, gNMI version, and encoding formats for the gNMI service.

Syntax

```
show grpc-server gnmi capabilities
```

Parameters

This command has no keywords or variables.

Modes

EXEC (#)

Usage Guidelines

The gRPC Network Management Interface (gNMI) is a gRPC based protocol used to modify and retrieve configuration from a target device and control and generate telemetry streams from a target device to a data collection system. The gNMI service defines operations for configuration management, operational state retrieval, and bulk data collection.

The gRPC Network Operations Interface (gNOI) defines a set of gRPC based microservices for running operational commands on network devices. The gNOI system service lets you perform system operations on a target network device, including device reboots, software upgrades, and network troubleshooting. You can execute supported "cert" service remote procedure calls (RPCs) to manage certificates on the network device. Using gNOI operations lets you use a single set of microservices to manage large-scale multivendor networks efficiently.

Examples

The following example displays the output of the **show grpc-server gnmi capabilities** command:

```
device# show grpc-server gnmi capabilities

gNMI version: 0.9.1
Supported YANG modules:
Module Name
Organization                                     Version
-----
-----
extreme-aaa-ext                               Extreme Networks,
Inc.                                           1.0.0
extreme-acl-ext                               Extreme Networks,
Inc.                                           1.0.0
extreme-acl-ipv4-ext                           Extreme Networks,
```

```
Inc. 1.0.0
extreme-acl-ipv6-ext Extreme Networks,
Inc. 1.0.0

---

Supported Encoding:
PROTO
device#
```

show grpc-server gnmi statistics

Use the **show grpc-server gnmi statistics** command to display details of active stream subscriptions, including client information, subscription mode, start time, and path details for the gNMI service.

Syntax

```
show grpc-server gnmi statistics
```

Parameters

This command has no keywords or variables.

Modes

EXEC (#)

Usage Guidelines

The gRPC Network Management Interface (gNMI) is a gRPC based protocol used to modify and retrieve configuration from a target device and control and generate telemetry streams from a target device to a data collection system. The gNMI service defines operations for configuration management, operational state retrieval, and bulk data collection.

The gRPC Network Operations Interface (gNOI) defines a set of gRPC based microservices for running operational commands on network devices. The gNOI system service lets you perform system operations on a target network device, including device reboots, software upgrades, and network troubleshooting. You can execute supported "cert" service remote procedure calls (RPCs) to manage certificates on the network device. Using gNOI operations lets you use a single set of microservices to manage large-scale multivendor networks efficiently.

Examples

The following example displays the output of the **show grpc-server gnmi statistics** command:

```
device# show grpc-server gnmi statistics

Number of active external subscriptions: 1
External Subscription Details:

Client:10.37.32.22      Mode:STREAM      Subscribed Path Count:1
Start Time: Wed Nov 27 09:47:43.574 UTC 2024
Keypath
      Subscription Mode   Interval
-----
/interfaces/interface[name=ethernet 0/1]/
config/mtu                ON_CHANGE                0s
```

```
-----  
-----  
device#
```


show interface

Use the **show interface** command to display the status of an Ethernet interface or a LAG interface, which includes any applied access control lists (ACLs).

Syntax

```
show interface { ethernet interface-name | internal interface-name |  
  loopback-mode | management 0 | port-channel interface-name | tunnel  
  interface-name } [ brief ]
```

Parameters

ethernet *interface-name*

Name of the Ethernet interface or range of interfaces. Use *slot/port* or *slot/port:breakout* format (*slot/port:channel-range*). For example, 1/1 or 1/1-3,5,2/7-9,10:1-4. The slashes (/) are required.

internal *interface-name*

Internal interface name. Use either 0 or 1.

loopback-mode

Displays details about the loopback interface.

management 0

Displays details about the management interface.

port-channel *interface-name*

Name of the link aggregation group (LAG) interface (1–255) or range of interfaces. For example, 1 or 1-3,5,7-9.

tunnel *interface-name*

Tunnel interface name. Use an integer or one or more comma-separated ranges of integers from 1 to 100000. For example, 4 or 4-5,5,7-9.

[**brief**]

Displays brief information about the interfaces on a device.

Modes

EXEC (#)

Usage Guidelines

The ethernet keyword of the **show interface** command displays the following information:

Output field	Description
ethernet <i>name</i>	Name of the Ethernet interface.
Admin state	Administrative state of the interface. This indicates whether the interface is forced administratively down indefinitely.
Operational state	Indicates whether the interface is up, down, or in a test mode.
Interface index is	Number that uniquely identifies the interface. This number displays in decimal format as well as its hexadecimal equivalent (with a "0x" before the index number). For details about interface index numbers, see The Interface Group MIB, RFC 2863 .
MTU	Maximum transmission unit of the interface. This is the largest packet that can be sent through the interface before the packet is fragmented.
Hardware is	Identifies the hardware interface.
mac address	LAN hardware address (data link address).
Current Speed	Current speed.
FEC Mode	Indicates whether forward error correction (FEC) is disabled or enabled. FEC is a method of obtaining error control in data transmission where the source (transmitter) sends redundant data, and the destination (receiver) recognizes only the part of the data that contains no apparent errors. The possible values are disabled or enabled.
Link Fault Signaling	Indicates whether link fault signaling is enabled. Link Fault Signaling (LFS) is a physical layer protocol that indicates the health of the link. LFS is part of the IEEE 802.3 standard that describes the ability of the Ethernet physical layer to determine remote or local link faults. It operates between the remote Ethernet device Reconciliation Sublayer (RS) and the local Ethernet device RS.
Link Fault Status	Link fault flag that is set.
<i>type</i> access-list <i>name</i> is <i>direction</i>	Access control list (ACL) that is attached to the interface: <i>type</i>—Type of ACL. The possible values are mac, ip, or ipv6 for MAC, IPv4, or IPv6 ACLs respectively. <i>name</i>—Name of the ACL. <i>direction</i>—Direction in which the ACL is attached. The possible values are in (ingress) or out (egress, which applies to Ethernet and VE interfaces only).

Output field	Description
Statistics	Non-packet-count statistics: - Carrier Transitions—Number of times the interface has "flapped" (completed a cycle of failing and then recovering). This number increases when the cable is unplugged, the remote system is powered down and then up, or another problem occurred. If this number increments quickly (such as every 10 seconds), this means that the cable, the remote system, or the PIC or PIM are malfunctioning. - LastClear—Time elapsed since the counters that measure cumulative statistics (such as number of bytes transmitted and received) shown in this report were last reset to zero (such as when the device was rebooted). Note that variables that might affect routing (for example, load and reliability) are not cleared when the counters are cleared.
Input	Counts of packets received by the interface: - Total packets—Total number of packets of all types received by the interface. - Broadcast packets—Total number of broadcast packets received by the interface. - Discard packets—Number of input packets that were silently discarded (possibly because of buffer overflow). - Errors packets—Total number of no buffer, runs, giants, CRC, frame, overrun, ignored, and abort counts. Other input-related errors can also increment the count, so that this sum might not balance with the other counts. - FCS Errors—Total number of errors where a legal sized frame with a bad frame check sequence (CRC error). An FCS error can be caused by a duplex mismatch, faulty NIC or driver, cabling, hub, or induced noise. - MCast packets—Total number of multicast packets received by the interface. - Octets—The number of octets (bytes) received by the interface. - UCast packets—Total number of unicast packets received by the interface. - Runt packets—Number of packets that are discarded because they are smaller than the minimum packet size of the medium. - CRC Errors—Number of times that the cyclic redundancy checksum (CRC) generated by the originating LAN station or remote device does not match the checksum calculated from the data received. On a LAN, this usually means noise or transmission problems on the LAN interface or the LAN bus. A high number is usually from collisions or a station transmitting bad data. On a serial link, CRCs usually indicate noise, gain hits, or other transmission problems on the data link.

Output field	Description
Input Distribution	Numbers of packets of all types received by the interface (grouped by size). Following are the size groups: • 64 or fewer bytes • 65 to 127 bytes • 128 to 255 bytes • 256 to 511 bytes • 512 to 1,023 bytes • 1,024 to 1,518 bytes • Jumbo packets—1,519 or more bytes (larger than the standard Ethernet IEEE 802.3 frame size of 1,518 bytes)
Out	Counts of packets sent by the interface: • Total pkts—Total number of packets of all types sent by the interface. • Broadcast pkts—Total number of broadcast packets sent by the interface. • Discard pkts—Number of output packets that were silently discarded (possibly because of buffer overflow). • Errors pkts—Total number of no buffer, runts, giants, CRC, frame, overrun, ignored, and abort counts. Other input-related errors can also increment the count, so that this sum might not balance with the other counts. • MCast pkts—Total number of multicast packets sent by the interface. • Octets—The number of octets ((bytes) sent by the interface. • UCast pkts—Total number of unicast packets received by the interface.
Rate Info	Rate information. • Input—Input rate information. For example, 0.000000 Mbits/sec, 0 pkts/sec 0.00% of line-rate. • Output—Output rate information. For example, 0.000000 Mbits/sec, 0 pkts/sec 0.00% of line-rate.

The port-channel keyword of the **show interface** command displays the following information:

Output field	Description
port-channel <i>name</i> is	Name and status of the LAG interface.
MTU	Maximum transmission unit of the interface. This is the largest packet that can be sent through the interface before the packet is fragmented.
IfIndex	Interface index number, which uniquely identifies the interface. This number displays in decimal format and its hexadecimal equivalent (with a "0x" before the index number). For details about interface index numbers, see The Interface Group MIB, RFC 2863 .

Output field	Description
Mac address is	LAN hardware address (data link address). For example, d8:c4:97:80:f0:73.
Port mode is	Duplex mode for the interface. This is either half-duplex, full-duplex, or autonegotiation. For example, Full Duplex, 0G.
MinLinks is	Minimum number of ports that must be linked up and bundled in the port channel. This applies only if Link Aggregation Control Protocol (LACP) is enabled on the interface.
LagType is	Type of link aggregation group (LAG) used by the interface. The two primary types of LAG are static (manual) and Link Aggregation Control Protocol (LACP). Dynamic LAGs use LACP to negotiate settings between the two connected devices.
Active Members in this channel	Number of Ethernet interfaces that are currently active (not down).
Members in this channel	Number of Ethernet interfaces that are currently inactive.
<i>type</i> access-list <i>name</i> is <i>direction</i>	Access control list (ACL) that is attached to the interface: <i>type</i>—Type of ACL. The possible values are mac, ip, or ipv6 for MAC, IPv4, or IPv6 ACLs respectively. <i>name</i>—Name of the ACL. <i>direction</i>—Direction in which the ACL is attached. The possible values are in (ingress) or out (egress, which applies to Ethernet and VE interfaces only).

Examples

The following example shows details about Ethernet interface 0/8 on the device:

```
device# show interface ethernet 0/8

ethernet 0/8 Admin state UP, Oper state UP
Interface index is 16777472 (0x1000100)
MTU 9216 bytes
Hardware is Ethernet mac address 50:0a:9c:97:8c:22
Current Speed 400G
Description: Ethernet 0/8
FEC:
    Mode: RS-FEC
    Uncorrected Blocks: 0
    Corrected Blocks: 703977063
Link Fault Signaling: ON
Link Fault Status: No Fault
Auto Negotiation: OFF
Carrier Transitions: 9
    LastClear: 0s
Input:
    Total pkts: 0
    Broadcast pkts: 0
    Discard pkts: 0
    Errors pkts: 0
    FCS Errors: 0
    MCast pkts: 0
```

```

        Octets: 0
        UCast pkts: 0
        Runt pkts: 0
        CRC Errors: 0
Input Distribution:
    64 byte pkts: 0
    65-127 byte pkts: 0
    128-255 byte pkts: 0
    256-511 byte pkts: 0
    512-1023 byte pkts: 0
    1024-1518 byte pkts: 0
    Jumbo pkts: 0
Out:
    Total pkts: 2655
    Broadcast pkts: 0
    Discard pkts: 0
    Errors pkts: 0
    MCast pkts: 2655
        Octets: 429921
    UCast pkts: 0
Rate Info:
    Input: 0.000000 Mbits/sec, 0 pkts/sec 0.00% of line-rate
    Output: 0.000000 Mbits/sec, 0 pkts/sec 0.00% of line-rate
device#
```

The following example shows the status of LAG interface 1 on the device:

```
device# device# show interface port-channel 1

port-channel 1 is down
MTU 9216 Bytes
IfIndex 0x4000001
Mac address is d8:c4:97:80:f0:ac
Port mode is Full Duplex, 0G
MinLinks is 1
LagType is LACP
Active Members in this channel: Nil
Members in this channel: Nil
mac access-list macl in
ip access-list v4acl in
ipv6 access-list v6acl in
device#
```

show interface brief

Use the **show interface brief** command to display brief information about the interfaces on a device.

Syntax

```
show interface brief
```

Parameters

This command has no keywords or variables.

Modes

EXEC (#)

Output

The **show interface brief** command displays the following information:

Output field	Description
Port	Name of the interface.
Mtu	Maximum transmission unit of the interface. This is the largest packet that can be sent through the interface before the packet is fragmented.
Admin-State	Administrative state of the interface. This indicates whether the interface is up (UP) or forced administratively down indefinitely (DOWN).
Oper-State	Indicates whether the interface is up, down, or in a test mode.
Speed	Nominal speed of the port. For example, 10G or 100G.
Description	Protocol and port number of the interface.

Examples

The following example shows brief information about the interfaces on the device:

```
32d# show interface brief

Flags:  M - Redundant Management  P - Performance-Path
Number of interfaces 37
Port      Mtu      Admin-State Oper-State Speed   Ifindex  Description
-----
ExtMgmt 1 9216      DOWN      DOWN    1G      0x41000020  ExtMgmt 1
ExtMgmt 2 9216      DOWN      DOWN    10G     0x41000040  ExtMgmt 2
Int 0     9216      UP        DOWN    20G     0x21000000  Internal 0
Int 1     9216      UP        DOWN    20G     0x21000020  Internal 1
Eth 0/1   9216      DOWN      DOWN    400G    0x10000020  Ethernet 0/1
```

Eth 0/2 device#	9216	DOWN	DOWN	400G	0x1000040	Ethernet 0/2

show interface management

Use the **show interface management** command to display details about the management interface on a device.

Syntax

```
show interface management interface-name
```

Parameters

interface-name

Name of the interface.

Modes

EXEC (#)

Usage Guidelines



Note

This command applies only to the Extreme 8730 platform.

This command shows the details of the management interface (such as the IP address that is configured on it) as well as statistics for the interface.

Examples

The following example shows details about the management interface on the device:

```
device# show interface management 0

management 0 Admin state UP      Operational state UP
MTU 1514 bytes
Hardware is Ethernet  mac address b4:a3:bd:0e:00:00
Current Speed  1G, Duplex: Full
Description: Management 0
DHCPv4 Enabled [ 198.51.100.0/24 ]
DHCPv6 Disabled

Redundancy Mode: (Dual OOB)
  active-path: extMgmt 1
  primary-path: extMgmt 1
  all-paths: extMgmt 1, extMgmt 2

Statistics
  LastClear: 0s
Input:
  Total pkts: 1038
  Discard pkts: 414
  Errors pkts: 0
  CRC Errors: 0
  MCast pkts: 992
```

```
Octets: 52080
device#
```

The following example shows details about the extMgmt 1 interface on the device:

```
device# show interface extMgmt 1

extMgmt 1 Admin state UP, Oper state UP
  Interface index is 1090519072 (0x41000020)
  MTU 9216 bytes
  Hardware is Ethernet  mac address b4:a3:bd:0e:00:19
  Current Speed  1G
  Description: External Management 1
device#
```

show license

Use the **show license** command to display details about the licenses and the EULA agreement for features on a device.

Syntax

```
show license [ eula ]
```

Parameters

eula

(Optional) Displays the EULA statement for the features.

Modes

EXEC (#)

Usage Guidelines

This command displays license information for each feature on a device. This includes the feature name, license description, EULA acceptance date and time, and (optionally) the deadline for purchasing a license and the URL location of the license agreement text. For a licensed feature, configuration and gNMI set for any leaf/list under the respective YANG paths are blocked unless the EULA is accepted.

To accept or decline licensed features such as integrated application hosting (IAH) on all Extreme ONE OS platforms except the Extreme 8730 platform or BGP EVPN, see [license eula advanced_features](#) on page 148.

To accept or decline the license EULA for IAH on the Extreme 8730 platform, see [license eula advanced_core](#) on page 146.

Examples

The following example shows the details about the licenses on the device:

```
device# show license

Feature name: ADVANCED_FEATURES
Description: ADVANCED_FEATURES license is trust based
Accepted on: Fri, 26 Sep 2025 05:51:07 UTC
device#
```

The following example shows the details about the EULA agreement on the device:

```
device# show license eula

Use of the features requires a license to be purchased within 30 days. By accepting
the EULA you indicate that you have read and accept the Extreme End User License
Agreement found at the following URL. [https://learn.extremenetworks.com/rs/641-VMV-602/
```

```
images/Extreme-Networks-End-User-License-Agreement.pdf] .  
device#
```

show logging audit

Use the **show logging audit** command to display entries from the audit log files on a device.

Syntax

```
show logging audit { config | firmware | security }
```

Parameters

config

Displays configuration related logs.

firmware

Displays firmware related logs.

security

Displays security related logs.

Modes

EXEC (#)

Usage Guidelines

To display the general log entries on the device, see [show logging raslogs](#) on page 256.

To display the details of a specific log entry on the device, see [show logging id](#).

Examples

The following example shows the configuration related logs on the device:

```
device# show logging audit config

2025-08-23 09:12:16.206 UTC +0000 LogID:7001 info Msg: none/none/none/http/grpc Method:/
xvm.xvmService/UpdateAppToken Status:OK
Request:token:"eyJhbGciOiJSUzI1NiIsInR5cCI6IkpXVCJ9.eyJyb2xlIjoiYWRTaW4iLCJvcnVzIjoiJFhRyZW
1lIE51dHdvcmtzIiwidmVyaW91IjoibGkiLCJpc3MiOiJOUeIqVG9rZW4g
U2VydmljZSIsInN1YiI6ImFkbWluIiwiaWF0IjoiYnZyY2MGM3Mi00MTk1LTExZWUtYmE4MC0zYTNoOG
FiMGQxZjgifQ.MFLNAz-
Cnr7bVVgo9NS_kdDrGNOyrvL6oAJya0M4fGv8uajL0g5i2AYeDKIDAJLRem6Zhdgm27MGoFCJi288jNDcq0bQbvpEB
R-sdkc05hq2RY1a0w9PP-KNCMQWipiOxnM8dCJgUuMd6DJ36AusgiQH0HBXa_MI-
l8perRIni8OKwGErBdmfI3G6o14t1p40BDG96jSRQRpIbHzJrqqYB3r_SHahdPJQOVIs48EVpcwWC-
Xr5ZsFH9qDaXEN2O7zccPsb8DGQbt4ziBopA4qpVNzYKnQgxIjedEAUEGJks7ddM-
acx9nOCU3iuONd8rr1oWc2GQ_GGTFRQhc32aPg" username:"admin" role:"admin" authmethod:"LOCAL"

2025-08-23 09:12:16.216 UTC +0000 LogID:7001 info Msg: none/none/none/http/grpc Method:/
auth.Auth/IsUserLocked Status:OK Request:username:"admin"

2025-08-23 09:12:16.217 UTC +0000 LogID:7001 info Msg: none/none/none/http/grpc Method:/
auth.Auth/UpdateLastLoginTime Status:OK Request:username:"admin"
```

```
2025-08-23 09:12:16.227 UTC +0000 LogID:7001 info Msg: none/none/none/http/grpc Method:/
aaa.Accounting/UserLogged Status:OK Request:username:"admin"

2025-08-23 09:22:41.558 UTC +0000 LogID:7001 info Msg: none/none/none/http/grpc Method:/
xvm.xvmService/UpdateAppToken Status:OK
Request:token:"eyJhbGciOiJSUzI1NiIsInR5cCI6IkpXVCJ9.eyJyb2xlIjoiYWRTaW4iLCJvc2ciOiJFeHRYZW
1lIE5ldHdvcmVtZiIiwidmVlIjoiMS4wIiwiaWQiOiIiLCJyZXF1ZXM0b3IiOiJjbGkiLCJpc3MiOiJOUeIgVG9rZW4g
U2VydmVlZSIiInN1YiI6ImFkbWluIiwiaXhwIjowLCJqdGkiOiIyNzY2MGZ3Mi00MTk1LTExZWUyYmE4M00zYTlkOG
FiMGQxZjg1fQ.MFLNAz-
Cnr7bVVgo9NS_kdDrGN0yrvL6oAJya0M4fGv8uajL0g5i2AYeDKIDAJLRem6Zhdgm27MGoFCJi288jNDcq0bQbvpEB
R-sdkc05hq2RY1a0w9PP-KNCMQWipiOxnM8dCJgUuMd6DJ36AusgiQH0HBXa_MI-
l8perRIni8OKWGERBdmfI3G6o14t1p40BDG96jSRQRpIbHzJrqqYB3r_ShahdPJQOVIS48EVpcwWC-
Xr5ZsFH9qDaXEN2O7zccPsb8DGOBt4ziBopA4qpVNzYKnQgxIjedEAUEGJks7ddM-
acx9nOCU3iuONd8rr1oWc2GQ_GGTFRQhc32aPg" username:"admin" role:"admin" authmethod:"LOCAL"

2025-08-23 09:29:20.219 UTC +0000 LogID:7001 info Msg: none/none/none/http/grpc Method:/
xvm.xvmService/UpdateAppToken Status:OK
Request:token:"eyJhbGciOiJSUzI1NiIsInR5cCI6IkpXVCJ9.eyJyb2xlIjoiYWRTaW4iLCJvc2ciOiJFeHRYZW
1lIE5ldHdvcmVtZiIiwidmVlIjoiMS4wIiwiaWQiOiIiLCJyZXF1ZXM0b3IiOiJjbGkiLCJpc3MiOiJOUeIgVG9rZW4g
U2VydmVlZSIiInN1YiI6ImFkbWluIiwiaXhwIjowLCJqdGkiOiI0OWMyMTM0NC00MTk3LTExZWUyYjYyNi0zYTlkOG
FiMGQxZjg1fQ.o4ztSnKk9PX7_qiSmGWHWvukWqgyUHZEEMAVIZQ3vJYQnZiQPmGayb5NFWPhtZk2ulkyJWiXgf8dlj
GcfjwHm_-
ujnDg3M_gtRqBSQlufJL1chzoiX62J39t2wz3tGTuwKaFLuD_ggDifqUJCry289bxRfTLrpnR7iFnkNk3wg6Ci_wO6
TQlUjgXyDHZIGvVBmuZ1OeA-
IH6dPchDgyXjzNMmrJ9E8vh88LethwZU8wmgmMlcXiLv3L80Gi8YKBqOw5SQP_45sIp6TnabgSZvxXBSwCwCJASlNjI-
59AQXVlruuFSA4tEZ40NFOKDLHHzcnyoB5V8VFQSCaa3Nscz4e7Q" username:"admin" role:"admin"
authmethod:"LOCAL"

2025-08-23 09:29:20.220 UTC +0000 LogID:7001 info Msg: none/none/none/http/grpc Method:/
auth.Auth/IsUserLocked Status:OK Request:username:"admin"

2025-08-23 09:29:20.221 UTC +0000 LogID:7001 info Msg: none/none/none/http/grpc Method:/
auth.Auth/UpdateLastLoginTime Status:OK Request:username:"admin"

2025-08-23 09:29:20.222 UTC +0000 LogID:7001 info Msg: none/none/none/http/grpc Method:/
aaa.Accounting/UserLogged Statu
device#
```

The following example shows the firmware related audit logs on the device.

```
device# show logging audit firmware

2025-03-28 13:47:22.528 UTC +0000 LogID:5021 info Msg: Firmware change successful.
Current Firmware Version is ExtremeOneBase-10.1.6.0-127
2025-03-28 13:47:54.180 UTC +0000 LogID:5021 info Msg: App SR Installed successful.
2025-04-11 04:29:29.899 UTC +0000 LogID:8002 info Msg: Firmware update file /var/data/
disk/firmware/ExtremeOneSR-22.2.1.0.bin downloaded
2025-04-11 04:29:29.900 UTC +0000 LogID:8004 info Msg: Activating firmware
ExtremeOneSR-22.2.1.0.bin. System will reboot if successful
2025-04-11 04:30:33.125 UTC +0000 LogID:5021 info Msg: App SR Uninstalled.
2025-04-11 04:32:52.366 UTC +0000 LogID:5021 info Msg: Firmware change successful.
Current Firmware Version is ExtremeOneBase-10.1.6.0-157
2025-04-11 04:33:33.803 UTC +0000 LogID:5021 info Msg: App SR Installed successful.
2025-04-11 04:59:23.981 UTC +0000 LogID:8011 info Msg: Successfully committed current
Base firmware image
2025-04-11 05:01:15.894 UTC +0000 LogID:5021 info Msg: App SR Installed successful.
2025-04-11 05:08:29.547 UTC +0000 LogID:5021 info Msg: App SR Installed successful.
device#
```

The following example shows the security related logs on the device:

```
device# show logging audit security | include testuser1
```

```
2025-02-01 17:30:00.001 IST +0530 LogID:7011 warn Msg: User inactivity warning: user
```

```
testuser1 will expire in 1 days
2025-02-02 17:30:00.000 IST +0530 LogID:7012 error Msg: User account expired: user
testuser1 expired 0 days ago due to inactivity
2025-02-02 17:30:00.000 IST +0530 LogID:7018 warn Msg: User testuser1 is locked due to
inactivity expiry period

device# show logging audit security | include testuser2

2025-02-01 17:30:00.001 IST +0530 LogID:7011 warn Msg: User inactivity warning: user
testuser2 will expire in 1 days
2025-02-02 17:30:00.000 IST +0530 LogID:7012 error Msg: User account expired: user
testuser2 expired 0 days ago due to inactivity
2025-02-02 17:30:00.000 IST +0530 LogID:7018 warn Msg: User testuser2 is locked due to
inactivity expiry period
device#
```

show logging raslogs

Use the **show logging raslogs** command to display the general log entries on a device.

Syntax

```
show logging raslogs
```

Parameters

This command has no keywords or variables.

Modes

EXEC (#)

Usage Guidelines

General log entries are from the Reliability, Availability and Serviceability Log (RASLog) file on the device.

To display entries from the audit log files on the device, see [show logging audit](#) on page 253.

To display the details of a specific log entry on the device, see [show logging id](#).

Examples

The following example shows the general log entries on the device:

```
# show logging raslogs
2025-03-10 06:16:42.1415 chassis-mgr[7]:
{"Level":"info","LogID":5300,"Msg":"fanEnable(6): Status: Up, RPM: 25200, RPM Percent:
102\n"}
2025-03-10 06:16:45.9544 monitor-svc[5568]: dumping service info for service chassis-mgr
2025-03-10 06:16:45.9556 monitor-svc[5568]: {"Level":"info","LogID":27003,"Msg":"Service
chassis-mgr is ready. current version chassis-mgr:1.2.0 prev version chassis-mgr:1.2.0
restart count 0"}
2025-03-10 06:16:46.3017 chassis-mgr[7]: {"Level":"info","Msg":"After SR App
Installation, all services in ready state"}
2025-03-10 06:16:46.3019 chassis-mgr[7]: {"Level":"info","Msg":"Cli Readiness Re-login"}
2025-03-10 06:16:46.3493 base-svc[5596]: {"Level":"error","Msg":"failed to kill cli
Sessions to force logout: exit status 1"}
2025-03-10 06:17:00.3926 chassis-mgr[7]: {"Level":"info","LogID":5016,"Msg":"System
initialization is complete. All services are UP"}
2025-03-10 06:17:50.7884 mlag[49]: {"Level":"info","LogID":17002,"Msg":"Timer BringUp
Delay Expired"}
2025-03-10 08:19:31.5031 security[45]:
{"Level":"info","Audit":"Security","LogID":7002,"Msg":"User authentication is successful
for user: admin "}
2025-03-10 08:34:38.0911 security[45]:
{"Level":"info","Audit":"Security","LogID":7002,"Msg":"User authentication is successful
for user: admin "}
```


show ntp association (Extreme ONE OS)

Use the **show ntp association** command to display a list of the associated Network Time Protocol (NTP) servers and peers on a device.

Syntax

```
show ntp association [ detail ]
```

Parameters

detail

(Optional) Displays details about the associated NTP servers and peers on the device.

Modes

EXEC (#)

Usage Guidelines

To view the current synchronization status of the NTP service, see [show ntp status \(Extreme ONE OS\)](#) on page 259.

Examples

The following example displays a list of the associated NTP servers and peers on the device:

```
device# show ntp association

      remote      refid      st t when poll reach  delay  offset  jitter
=====
+162.159.200.1    10.209.8.4        3 u  14   64  177   10.038  +2.113  10.889
95.216.192.15    .STEP.            16 u   -   64   0    0.000  +0.000  0.000
*5.161.213.25    59.61.64.227      2 u   9   64  377   74.703  -0.613  14.551

* synced, # selected, + candidate, - outlayer, x falseticker, ~ configured
device#
```

The following example displays details about the associated NTP servers and peers on the device:

```
device# show ntp association detail

10.32.94.124:123 configured, best candidate, stratum 2,
ref ID 35.73.197.144, time ec127589.4608e8e8 Fri, Jul  4 2025 15:45:13.273,
our mode client, peer mode server, our poll intvl 6, peer poll intvl 6,
root delay 68.466 msec, root disp 27.100, reach 001, jitter 0.842,
delay 0.991 msec, offset +0.009 msec, dispersion 187.527,
precision is 2**-23, version 4
filter delay =      0.99    4.72    1.03    0.66    1.34    1.38    0.00    0.00,
filter offset =    +0.01   -1.86   +0.00   -0.10   +0.19   -0.11   +0.00   +0.00,
```

```
filter disp = 0.00 0.03 0.06 0.09 0.12 0.15 16000.0 16000.0
device#
```

show ntp status (Extreme ONE OS)

Use the **show ntp status** command to display the current synchronization status of the NTP service on a device.

Syntax

```
show ntp status
```

Parameters

This command has no keywords or variables.

Modes

EXEC (#)

Usage Guidelines

To view a list of the associated Network Time Protocol (NTP) servers and peers on the device, see [show ntp association \(Extreme ONE OS\)](#) on page 257.

Examples

The following example displays the current synchronization status of the NTP service on the device:

```
device# show ntp status

Clock is synchronized,  stratum 3, reference clock is 198.51.100.0,
precision is -23,
reference time is eb72b61a.4f9fbefd Wed, Mar  5 2025 11:38:02.311,
clock offset is +0.000000, root delay is 64.599,
root dispersion is 33.446, peer dispersion is 60510,
NTP client mode is enabled (VRF: mgmt-vrf)
device#
```

show running-config system snmp-server

Use the **show running-config system snmp-server** command to display the SNMP server configuration that is running currently on the device.

Syntax

```
show running-config system snmp-server
```

Parameters

This command has no keywords or variables.

Modes

EXEC (#)

Output

The **show running-config system snmp-server** command displays the following information:

Output field	Description
community	(SNMPv1 and SNMPv2c only) SNMP community string, which is a way to access statistics stored in the device. This is a type of shared password between the SNMP management station and the device, which is used to authenticate the SNMP management station. All the communities configured have read-all permission. No communities have write permission.
enable	Indicates whether the SNMP server is enabled.
host	IP address or hostname of the trap receiver. All SNMP traps from the device go to the same SNMP trap receiver or set of receivers (which are usually one or more host devices on the network).
snmp-server	Name of the SNMP server instance.
source-interface	Source interface for sending traps to the trap host for the SNMP server.
udp-port	UDP port number for sending traps to the trap host for the SNMP server.
user	(SNMPv3 only) One or more SNMPv3 users. All users have read-all permission. No users have write permission.
version	SNMP version for sending traps to the trap host for the SNMP server. This is either 1, 2c, or 3.
vrf	VRF instance that is associated to the SNMP server.

Examples

The following example displays the SNMP server configuration that is running currently on the device:

```
device# show running-config system snmp-server

system
  snmp-server DEFAULT
    user user3 auth noauth
    vrf mgmt-vrf
    enable
    host 192.168.1.1 user3
    version 3
    udp-port 200
    source-interface ethernet 0/1.0
  !
!
!
!
device#
```

show sysinfo all

Use the **show sysinfo all** command to display general health information about system hardware components such as the fan, power supply unit (PSU), sensors, slots, and LEDs.

Syntax

```
show sysinfo all
```

Parameters

This command has no keywords or variables.

Modes

EXEC (#)

Examples

The following example shows the output of the **show sysinfo all** command:

```
device# show sysinfo all

Fan Information
Id      Status    RPM      Percentage SpeedLevel    Direction
-----
1       Up         4480     18         LOW           FAN_DIR_F2B
2       Up         4480     18         LOW           FAN_DIR_F2B
3       Up         4480     18         LOW           FAN_DIR_F2B
4       Up         4480     18         LOW           FAN_DIR_F2B
5       Up         4480     18         LOW           FAN_DIR_F2B
6       Up         4640     19         LOW           FAN_DIR_F2B

FAN_DIR_F2B - Fan Airflow Direction is FrontToBack

FanSpeedLevel - <40%[LOW],40-70%[MEDIUM],>70%[HIGH]

Led Information
Id      State    Color    Description
-----
led-0   Solid    Green    Power Supply Unit 1
led-1   Off       OFF      Power Supply Unit 2
led-2   Solid    Green    Fan
led-3   Blink    Amber    System Status

PSU Information
Id      Status    Type    C[in]  C[out]  P[in]  P[out]  V[in]  V[out]
-----
1       Up        AC      0.540  6       105    80      208    12

**C - Current in Amps ,**P - Power in Watts ,**V - Voltage in Volts

Sensor Information
Id Name          Current(°C/Volt)  Warning(°C/Volt)  Critical(°C/Volt)  Shutdown(°C/Volt)
-----
```

```
1 CPU Core 43.00 88.00 90.00 93.00
2 DIMM0 37.00 80.00 84.00 0.00
3 DIMM1 35.00 80.00 84.00 0.00
4 AMBIENT0 34.00 60.00 63.00 65.00
5 AMBIENT1 36.00 61.00 64.00 66.00
6 AMBIENT2 32.00 62.00 65.00 67.00
7 AMBIENT3 34.00 61.00 64.00 66.00
8 AMBIENT4 31.00 63.00 66.00 68.00
9 AMBIENT5 0.00 0.00 0.00 0.00
10 PSU1 Temp1 38.00 109.00 110.00 125.00
11 PSU1 Temp2 30.00 109.00 110.00 125.00
12 PSU1 Temp3 0.00 64.00 65.00 70.00
13 PSU2 Temp1 0.00 109.00 110.00 125.00
14 PSU2 Temp2 0.00 109.00 110.00 125.00
15 PSU2 Temp3 0.00 0.00 0.00 0.00
device#
```

show sysinfo chassis

Use the **show sysinfo chassis** command to display the chassis information such as the product name, platform name, part number, hardware revision, serial number, MAC address, operating system version, and system uptime.

Syntax

```
show sysinfo chassis
```

Parameters

This command has no keywords or variables.

Modes

EXEC (#)

Output

The **show sysinfo chassis** command displays the following information:

Output field	Description
Product Name	Full product name, which includes the manufacturer and the series number. For example, Extreme 8520-48Y.
Platform Name	String that indicates the full platform details such as the processor type and full product name. For example, x86_64-extremenetworks_8520-48y-r0.
Hardware Rev	Unique numeric identifier assigned by Extreme Networks to the specific iteration of the device hardware. This number is used for providing technical support and ensuring compatibility with software and firmware updates. This number might also be used to identify issues or defects in specific hardware versions and to determine which hardware revisions are eligible for warranty or repair services.
Manufacture Date	Date of production assigned to the device by Extreme Networks.
Manufacturer	Manufacturer of the device. For devices running Extreme ONE OS, this field displays Extreme Networks, Inc.
Part Number	Unique alphanumeric identifier assigned by Extreme Networks to all devices that are identical to the current device.
Serial Number	Unique alphanumeric identifier assigned by Extreme Networks to the current device. By checking the serial number against the manufacturer's database, you can confirm whether the device is genuine or a counterfeit.
Vendor	

Output field	Description
System Status	Operational status of the device.
Reboot Reason	Reason for the most recent reboot.
System Description	Full product name, Linux kernel version that is included in the Yocto Project build, and Extreme ONE OS operating system version.
System Uptime	Length of time that the device has been running continuously without unexpected shutdowns, crashes, or other downtime events.
Base MAC Address	MAC address assigned by Extreme Networks to the device, which is used as the default source MAC address for frames generated by the device. This address is also used by the device for its management interface and protocols such as Link Aggregation Control Protocol (LACP).
MAC Range	Group of MAC addresses that are reserved for use by Extreme Networks.
LC Slots	Number of slots available for Lucent Connector (LC) cards or modules.
Fan Count	Number of fans that cool the device and its components.
LED Count	Number of LEDs mounted on the device that indicate status and activity information.
PSU Count	Number of power supply units (PSUs) that provide power to the device and its components.
Thermal Sensor Count	Number of thermal sensors that monitor the temperature of the device and its components.

Examples

The following example shows the output of the **show sysinfo chassis** command:

```
device# show sysinfo chassis

    Product Name: Extreme 8730-32D
    Platform Name: x86_64-extremenetworks-8730-32d-r0
    Hardware Rev: 1.0
    Manufacture Date: 01/14/2025 03:36:23
    Manufacturer: Extreme Networks, Inc.
    Part Number: 801157-00-03
    Serial Number: AD012451G-00026
    Vendor: Extreme Networks Inc.
    System Status: Online
    Reboot Reason: Config Change
    System Description: Extreme 8730-32D, 6.6.54-yocto-standard, Version
ExtremeOneBase-10.1.3.0-075
    System Uptime: 39h39m1s
    Base MAC Address: 50:0a:9c:97:8c:00
        Mac Range: 1024
        LC Slots: 1
        Fan Count: 7
        LED Count: 5
        PSU Count: 2
    Thermal Sensor Count: 4
```

```
Airflow Direction: Front to back  
device#
```

show sysinfo fan

Use the **show sysinfo fan** command to display health information about all fan hardware components such as their status, speed level, airflow direction, and RPM.

Syntax

```
show sysinfo fan
```

Parameters

This command has no keywords or variables.

Modes

EXEC (#)

Examples

The following example shows the output of the **show sysinfo fan** command:

```
device# show sysinfo fan

Fan Information
Id      Status    RPM      Percentage SpeedLevel  Direction
-----
1       Up         4480     18         LOW         FAN_DIR_F2B
2       Up         4640     19         LOW         FAN_DIR_F2B
3       Up         4480     18         LOW         FAN_DIR_F2B
4       Up         4480     18         LOW         FAN_DIR_F2B
5       Up         4480     18         LOW         FAN_DIR_F2B
6       Up         4480     18         LOW         FAN_DIR_F2B

FAN_DIR_F2B - Fan Airflow Direction is FrontToBack

FanSpeedLevel - <40%[LOW],40-70%[MEDIUM],>70%[HIGH]
device#
```

show sysinfo inventory all

Use the **show sysinfo inventory all** command to display inventory information about all linecard and PSU modules on a device.

Syntax

```
show sysinfo inventory all
```

Parameters

This command has no keywords or variables.

Modes

EXEC (#)

Examples

The following example shows the output of the **show sysinfo inventory all** command:

```
device# show sysinfo inventory all

      Module      : PSU-0
      Model       : 700-013684-0100
      SystemNo    : 700-013684-0100
      SerialNo    : L2141900A906P
      Version     : SV1.0

      Module      : PSU-1
      Model       : 700-013684-0100
      SystemNo    : 700-013684-0100
      SerialNo    : L21419008A06P
      Version     : SV1.0
device#
```

show sysinfo inventory fan

Use the **show sysinfo inventory fan** command to display inventory information such as model and serial numbers for fan modules on a device.

Syntax

```
show sysinfo inventory fan
```

Parameters

This command has no keywords or variables.

Modes

EXEC (#)

Usage Guidelines

This command supports only the Extreme 8730 platform.

Examples

The following example shows the output of the **show sysinfo inventory fan** command:

```
device# show sysinfo inventory fan

      Module      : Fan-0
      Model       : 801166-00-01
Serial Number    : F0032312G-00039

      Module      : Fan-1
      Model       : 801166-00-01
Serial Number    : F0032312G-00034

      Module      : Fan-2
      Model       : 801166-00-01
Serial Number    : F0032312G-00031

      Module      : Fan-3
      Model       : 801166-00-01
Serial Number    : F0032312G-00020

      Module      : Fan-4
      Model       : 801166-00-01
Serial Number    : F0032312G-00009

      Module      : Fan-5
      Model       : 801166-00-01
Serial Number    : F0032312G-00033

      Module      : Fan-6
      Model       : 801166-00-01
```

```
Serial Number      : F0032312G-00003  
device#
```

show sysinfo inventory power-supply

Use the **show sysinfo inventory power-supply** command to display inventory information for power supply unit (PSU) modules on a device.

Syntax

```
show sysinfo inventory power-supply
```

Parameters

This command has no keywords or variables.

Modes

EXEC (#)

Examples

The following example shows the output of the **show sysinfo inventory power-supply** command:

```
device# show sysinfo inventory power-supply

  Module      : PSU-0
  Model       : 700-013684-0100
  SystemNo    : 700-013684-0100
  SerialNo    : L2141900A906P
  Version     : SV1.0

  Module      : PSU-1
  Model       : 700-013684-0100
  SystemNo    : 700-013684-0100
  SerialNo    : L21419008A06P
  Version     : SV1.0
device#
```

show sysinfo led

Use the **show sysinfo led** command to display the values of the system health LEDs on the front panel.

Syntax

```
show sysinfo led
```

Parameters

This command has no keywords or variables.

Modes

EXEC (#)

Output

The **show sysinfo led** command displays the following information:

Output field	Description
Id	ID of the specific LED for the component.
State	Overall health of the component. Solid indicates a healthy state. Blink indicates a fault. Off indicates that the device is turned off.
Color	Overall health of the component. Green indicates a healthy status. Amber indicates a fault and requires attention (this is visible even during reboot).
Description	Name of the component. The System Status LED indicates the software state of the device and if the device is up and running (whether it has power and if any power rails have dropped below specification). The Power Supply Unit and Fan LEDs indicate if any of these modules has a warning or error.

Examples

The following example shows the output of the **show sysinfo led** command:

```
device# show sysinfo led

Led Information
Id      State   Color   Description
-----
led-0   Solid    Green   Power Supply
led-2   Solid    Green   Fan
led-3   Solid    Green   System Status
device#
```


show sysinfo media detected

Use the **show sysinfo media detected** command to display information about the media detected in the chassis such as the part number, serial number, vendor, and description.

Syntax

```
show sysinfo media detected
```

Parameters

This command has no keywords or variables.

Modes

EXEC (#)

Examples

The following example shows the output of the **show sysinfo media detected** command:

```
device# show sysinfo media detected
```

S/C Vendor	Qual	Optical Description	Type	PartNum	SerialNum	

0/1 NETWORKS	Yes 100G	Yes SR4 QSFP28 100m	QSFP28	EQPT1H4SR4UCM100	2218H-50292	EXTREME
0/3 NETWORKS	Yes 40G	No Passive DAC QSFP+ 1m	QSFP_PLUS	EQPC40PC010C0100	2246X-50051	EXTREME
0/4 NETWORKS	Yes 100G	No Passive DAC QSFP28 1m	QSFP28	EQPC1HPC010C0100	2214A-53627	EXTREME
0/5 NETWORKS	Yes 400G-DACP	No -QSFPDD2M	QSFP-DD	EQDC4HPC020C0100	T0062319A-53729	EXTREME
0/7 NETWORKS	Yes 400G-FR4	Yes -QSFPDD2KM	QSFP-DD	EQDT4H8FR4FCL100	T0192337E-50030	EXTREME
0/9 NETWORKS	Yes 100G	No Passive DAC QSFP28 1m	QSFP28	EQPC1HPC010C0100	2114A-56882	EXTREME
0/10 NETWORKS	Yes 40G	No Passive DAC QSFP+ 1m	QSFP_PLUS	EQPC40PC010C0100	2203X-40362	EXTREME
0/11 NETWORKS	Yes 400G-DACP	No -QSFPDD1M	QSFP-DD	EQDC4HPC010C0100	T0052319A-53726	EXTREME
0/12 NETWORKS	Yes 100G	Yes DR QSFP28 500M	QSFP28	EQPT1H8DR1DCL100	T0012404E-50004	EXTREME

0/13 NETWORKS	Yes	Yes	QSFP_PLUS	EQPT404SR4VCM100	2109H-50069	EXTREME
	40G-SR4-QSFP150M, 850nm					
0/14 NETWORKS	Yes	No	QSFP28	EQPC1HPC010C0100	2231A-53265	EXTREME
	100G Passive DAC QSFP28 1m					
0/15 NETWORKS	Yes	Yes	QSFP-DD	EQDC4HAC050C0100	T0112313E-50002	EXTREME
	400G-AOC-QSFPDD5M					
0/16 NETWORKS	Yes	Yes	QSFP28	EQPT1H4WM4ECL100	2238E-50070	EXTREME
	100G 4WDM-40 QSFP28 40km					
0/17 NETWORKS	Yes	Yes	QSFP28	EQPT1H4SR4UCM100	2218H-50396	EXTREME
	100G SR4 QSFP28 100m					
0/18 NETWORKS	Yes	Yes	QSFP-DD	EQDT4H8FR4FCL100	T0192343H-50016	EXTREME
	400G-FR4-QSFPDD2KM					
0/21 NETWORKS	Yes	Yes	QSFP28	EQPT1H4WM4WCL100	2215E-50007	EXTREME
	100G 4WDM-20 QSFP28 20km					
0/22 NETWORKS	Yes	Yes	QSFP28	EQPT1H4LR4LCL100	2235P-00033	EXTREME
	100G LR4 QSFP28 10km					
0/23 NETWORKS	Yes	Yes	QSFP_PLUS	EQPT404BD2VCL100	2023L-31062	EXTREME
	40G-BDSR-QSFP150M, 850/900nm					
0/24 NETWORKS	Yes	Yes	QSFP28	EQPT1H4CW4FCL100	2225H-50015	EXTREME
	100G CWDM4 QSFP28 2km					
0/25 NETWORKS	Yes	Yes	QSFP28	EQPT1H4LR4LCL100	22256-20003	EXTREME
	100G LR4 QSFP28 10km					
0/26 NETWORKS	Yes	Yes	QSFP28	EQPT1H4WM4LCL100	2211E-50002	EXTREME
	100G 4WDM-10 QSFP28 10km					
0/27 NETWORKS	Yes	Yes	QSFP28	EQPT1H4ER4ECL1LT	2128E-50031	EXTREME
	100G ER4-lite QSFP28 40km					
0/28 NETWORKS	Yes	Yes	QSFP_PLUS	EQPT404SR4SCM1NT	2236H-50406	EXTREME
	40G-ESR4-QSFP400M-NT					
----- Management Media						

S/C Vendor	Qual	Optical Description	Type	PartNum	SerialNum	

extMgmt 1 CORP.	Yes	No	SFP	FCLF8520P2BTL-EX	2248P-00730	FINISAR
	1G BASE-T SFP 100m					
extMgmt 2 CORP.	Yes	No	SFP	FCLF8520P2BTL-EX	1720P-00638	FINISAR
	1G BASE-T SFP 100m					
device#						

show sysinfo media interface ethernet

Use the **show sysinfo media interface ethernet** command to display the media information associated with the Ethernet physical interfaces on a device.

Syntax

```
show sysinfo media interface ethernet [ interface-name ]
```

Parameters

interface-name

(Optional) Ethernet interface name. Use the *slot/port* format. For example, 0/11.

Modes

EXEC (#)

Examples

The following example shows the output of the **show sysinfo media interface ethernet** command for Ethernet interface 0/14:

```
device# show sysinfo media interface ethernet 0/14

Media Information:
  Interface: ethernet 0/14
    Cage: 14
    Slot: 0
  Qualified: Yes
  Optical: No
  State: Inserted
  Module Type: QSFP28
  Part Number: EQPC1HPCZ50C0100
  Serial Number: 2343X-50499
  Vendor: EXTREME NETWORKS
  Supports Breakout: No
  Description: 100G Passive DAC QSFP28 0.5m
  Channels: 0
  Datecode: 2023-10-24
  BaudRate: 255 (units 100 megabaud)
  LengthSmf: 0 (units km)
  LengthOm3: 0 (units 2 meters)
  LengthOm2: 0 (units 1 meter)
  LengthOm1: 0 (units 1 meter)
  Wavelength: 516 (units of 0.05 nm)
  LengthCopper: 1 (units 1 meter)
  VendorRev: AA
  Options: 000000
  BrMax: 0
  BrMin: 0

device#
```

show sysinfo media interface extMgmt

Use the **show sysinfo media interface extMgmt** command to display the media information associated with an external management interface.

Syntax

```
show sysinfo media interface extMgmt [ interface-name ]
```

Parameters

interface-name

(Optional) External management interface name. The range is 1 to 2.

Modes

EXEC (#)

Usage Guidelines

This command applies only to the Extreme 8730 platform.

Examples

The following example shows the output of the **show sysinfo media interface extMgmt** command for external management interface 1:

```
device# show sysinfo media interface extMgmt 1

Media Information:
    Interface: extMgmt 1
    Cage: 33
    Slot: 0
    Qualified: Yes
    Optical: No
    State: Inserted
    Module Type: SFP
    Part Number: FCLF8522P2BTL-EX
    Serial Number: NBHC437
    Vendor: FINISAR CORP.
    Supports Breakout: No
    Description: 1G BASE-T SFP 100m
    Channels: 0
    Datecode: 2024-05-09
    BaudRate: 12 (units 100 megabaud)
    LengthSmfKm: 0 (units km)
    LengthSmfM: 0 (units 100 meters)
    Length50UmM: 0 (units 10 meters)
    Length625UmM: 0 (units 10 meters)
    Wavelength: 0 (units nm)
    LengthCopper: 100 (units 1 meter)
    VendorRev: A
    Options: 0012
    BrMax: 0
```

```
device#          BrMin: 0
```

show sysinfo media supported

Use the **show sysinfo media supported** command to display the list of media supported along with part numbers, vendor names, and descriptions.

Syntax

```
show sysinfo media supported
```

Parameters

This command has no keywords or variables.

Modes

EXEC (#)

Examples

The following example shows the output of the **show sysinfo media supported** command:

```
device# show sysinfo media supported
```

Type	PartNum	Vendor	Description

qsfp	57-1000129-01	BROCADE	40GBase-SR4 QSFP
qsfp	57-1000263-01	BROCADE	40G QSFP-LR4 10KM
qsfp	58-0000033-01	BROCADE	40G-QSFP-QSFP 1m cable passive
qsfp	58-0000034-01	BROCADE	40G-QSFP-QSFP 3m cable passive
qsfp	58-0000035-01	BROCADE	40G-QSFP-QSFP 5m cable passive
qsfp	58-0000041-01	BROCADE	40G-QSFP-QSFP 1m Active Copper
qsfp	58-0000042-01	BROCADE	40G-QSFP-QSFP 3m Active Copper
qsfp	58-0000043-01	BROCADE	40G-QSFP-QSFP 5m Active Copper
qsfp	57-1000325-01	BROCADE	40G-QSFP+ LM4
qsfp	57-1000306-01	BROCADE	40G QSFP to QSFP cable 10m AOC
qsfp	57-1000339-01	BROCADE	40G QSFP BIDI Optic
qsfp	AFBR-79EBPZ	AVAGO	40G QSFP BIDI Optic
qsfp	AFBR-79EBRZ	AVAGO	40G QSFP BIDI Receiver Optic
qsfp	58-0000053-01	BROCADE	4x10G QSFP 5m Active Copper Cable

qsfp device#	58-0000052-01	BROCADE	4x10G QSFP 3m Active Copper Cable

show sysinfo power-supply

Use the **show sysinfo power-supply** command to display health information about the power capacity of each power supply unit (PSU) and its state along with the readings for voltage and current in and out.

Syntax

```
show sysinfo power-supply
```

Parameters

This command has no keywords or variables.

Modes

EXEC (#)

Examples

The following example shows the output of the **show sysinfo power-supply** command:

```
device# show sysinfo power-supply

PSU Information
Id      Status      Type    C[in]  C[out] P[in]   P[out] V[in]   V[out]
-----
1       Up           AC      0.540  6      105     80     208     12

**C - Current in Amps ,**P - Power in Watts ,**V - Voltage in Volts
device#
```


show sysinfo sensor all

Use the **show sysinfo sensor all** command to display sensor information related to the CPUs and power supply units (PSUs) in a device.

Syntax

```
show sysinfo sensor all
```

Parameters

This command has no keywords or variables.

Modes

EXEC (#)

Examples

The following example shows the output of the **show sysinfo sensor all** command:

```
device# show sysinfo sensor all

Sensor Information
Id  Name                Current (°C/Volt)  Warning (°C/Volt)  Critical (°C/Volt)  Shutdown (°C/
Volt)
-----
1   CPU Core            49.00             76.00              80.00              84.00
2   Sensor 1 (AFO)      42.00             54.00              60.00              66.00
3   Sensor 2 (Hot Spot) 45.00             76.00              80.00              84.00
4   Sensor 3 (AFI)      27.00             54.00              60.00              66.00
device#
```

show sysinfo sensor cpu

Use the **show sysinfo sensor cpu** command to display sensor information related to the CPUs in a device.

Syntax

```
show sysinfo sensor cpu
```

Parameters

This command has no keywords or variables.

Modes

EXEC (#)

Examples

The following example shows the output of the **show sysinfo sensor cpu** command:

```
device# show sysinfo sensor cpu

Sensor Information
Id  Name      Current(°C/Volt)  Warning(°C/Volt)  Critical(°C/Volt)  Shutdown(°C/Volt)
-----
1   CPU Core  43.00             88.00             90.00             93.00
2   DIMM0     37.00             80.00             84.00             0.00
3   DIMM1     34.00             80.00             84.00             0.00
4   AMBIENT0  34.00             60.00             63.00             65.00
5   AMBIENT1  36.00             61.00             64.00             66.00
6   AMBIENT2  31.00             62.00             65.00             67.00
device#
```

show system contact

Use the **show system contact** command to display the administrative contact details that are configured for a deployed device.

Syntax

```
show system contact
```

Parameters

This command has no keywords or variables.

Modes

EXEC (#)

Usage Guidelines

Use this command to display the information for contacting the network administrator of the device. The network administrator is responsible for addressing problems impacting the network.

To configure the network administrator's contact details for the device, see [contact \(Extreme ONE OS\)](#) on page 85.

Examples

The following example displays the network administrator's contact details:

```
device# show system contact

contact: Hyderabad
device#
```

The following example displays the configuration of the network administrator's contact details that is running currently on the device:

```
device# show running-config system contact

system
  contact Hyderabad
!
device#
```

show system diagnostics

Use the **show system diagnostics** command to display the results of an operational system diagnostics run.

Syntax

```
show system diagnostics
```

Parameters

This command has no keywords or variables.

Modes

EXEC (#)

Usage Guidelines

Use this command to display the results from either of the **system diagnostics run** commands. You can perform a normal diagnostics run or an extended diagnostics run. For details, see [system diagnostics run normal](#) on page 347 or [system diagnostics run extended](#) on page 345 respectively.

Examples

The following example performs a normal system diagnostics run on the device:

```
device# system diagnostics run normal

WARNING: system will be rebooted! Do you want to continue? [y/n]: y
Reloading.... please wait
device#
```

The following example shows the results of a normal system diagnostics run on the device:

```
device# show system diagnostics

Operational diagnostics version: 1.1
Date: Mar-10-2025 06:09:22
Version: 1.1
Summary: Diagnostics Pass
Mode: Normal

8730-32D:                801157-00-02 AD012350G-00014
PSU-1: Present           801163-00-00 P0152342A-C0072
PSU-2: Present           801163-00-00 P0152342A-C0052
FAN-1: Present           801167-00-02 F0042349G-00133
FAN-2: Empty
FAN-3: Empty
FAN-4: Present           801167-00-02 F0042349G-00082
FAN-5: Present           801167-00-02 F0042349G-00093
FAN-6: Present           801167-00-02 F0042349G-00085
FAN-7: Present           801167-00-02 F0042349G-00204
```

```
Test                                Result
=====
i2c environment                     PASS
fantray                             PASS
internal flash                      PASS
memory                             PASS
management interface               PASS
side_band interface                 PASS
loopback pci                       PASS
loopback mac interface             PASS
loopback phy fiber                 PASS
snake interface                    PASS
asic0 reg                          PASS
asic0 mem                          SKIP
device#
```

show system dns

Use the **show system dns** command to display the Domain Name System (DNS) server configurations.

Syntax

```
show system dns
```

Parameters

This command has no keywords or variables.

Modes

EXEC (#)

Usage Guidelines

Use this command to display the DNS IP domain name and IP name server configurations.

Examples

The following example displays the DNS server configurations on the device:

```
device# show system dns

ip dns domain name ip dns name-server
=====
openstacklocal      10.37.2.1
device#
```

show system health (Extreme ONE OS)

Use the **show system health** command to monitor critical system resources on a device.

Syntax

```
show system health
```

Parameters

This command has no keywords or variables.

Modes

EXEC (#)

Usage Guidelines

Use this command to display the current system state parameters and health data. This includes data about the BIOS, memory utilization and availability, CPU, disk utilization, Peripheral Component Interconnect (PCI) devices, SSDs, chassis, power supply unit (PSU), fans, and instantaneous system performance.

Examples

The following example displays the system health of the device:

```
device# show system health | nomore

BIOS Vendor: Alpha Networks
BIOS Version: 0.20.0
BIOS Date: 01/20/2025
Kernel Version: 6.6.54-yocto-standard
Kernel Arch: x86_64
CPU Procs: 24
OS: linux
Platform: alpine
Platform Version: 3.21.3
Memory Used: 2.94GB
Memory Used (%): 9.00%
Memory Total: 30.44GB
Memory Free: 26.90GB
CPU Model: Intel Atom(R) P5362 processor
Cpu Load: 3
Primary Rootfs Disk Total: 24.06GB
Primary Rootfs Disk Free: 18.56GB
Primary Rootfs Disk Used: 4.26GB
Secondary Rootfs Disk Total: 25.00GB
System Uptime: 3h51m40s
RAM Buffers(kB): 87260
RAM Caches(kB): 2294720
RAM Free(kB): 26373136
Ram Used(kB): 3160284
Cpu System time: 0
```

```

Cpu IOWait time: 0
Cpu User time: 2
Cpu SoftIRQ time: 0
APPDATA Partition Used: 3.00%
Disk Written(kB): 1144520
CONFIG Partition Used: 1.00%
IAH Partition Used: 1.00%
Secondary Rootfs Disk Used: 4.30GB
Secondary Rootfs Disk Free: 19.00GB
USRDATA Partition Used: 2.00%
Disk Read(kB): 2110711

```

SERVICE	CPU(cores)	Memory(bytes)
api-gw	26m	111Mi
chassis-mgr	31m	23Mi
config-db	4m	4Mi
ingress-gw	1m	10Mi
interface-mgr	7m	30Mi
msg-bus	13m	117Mi
security	1m	13Mi
snmp-agent	2m	36Mi
state-db	6m	4Mi
ztp-base	1m	10Mi
arpnd	1m	17Mi
bfd	1m	15Mi
bgp	6m	17Mi
classifiers	1m	15Mi
fwd-hal	70m	454Mi
iah	1m	15Mi
lACP	4m	18Mi
lldp	2m	16Mi
mftm	1m	12Mi
mlag	4m	18Mi
sr-api-gw	1m	26Mi
telegraf	2m	21Mi
uftm	1m	16Mi

PCI-INFO

```

pci52: eb:00.2 System peripheral [0880]: Intel Corporation Ice Lake RAS [8086:09a3] (rev 02)
pci75: fe:0b.1 System peripheral [0880]: Intel Corporation Device [8086:3448]
pci89: ff:0a.4 System peripheral [0880]: Intel Corporation Device [8086:344d]
pci26: 00:1a.1 Serial controller [0700]: Intel Corporation Device [8086:18d8] (rev 11)
pci21: 00:0e.0 SATA controller [0106]: Intel Corporation Atom Processor P5xxx Series SATA Controller [8086:18f3] (rev 11)
pci45: e3:00.1 System peripheral [0880]: Intel Corporation Ice Lake Mesh 2 PCIe [8086:09a4] (rev 02)
pci20: 00:0b.0 PCI bridge [0604]: Intel Corporation Device [8086:18a6] (rev 11)
pci57: ec:00.1 Ethernet controller [0200]: Intel Corporation Ethernet Connection E822-C for backplane [8086:1890] (rev 20)
pci37: 14:00.0 System peripheral [0880]: Intel Corporation Ice Lake Memory Map/VT-d [8086:09a2] (rev 02)
pci47: e3:00.3 System peripheral [0880]: Intel Corporation Device [8086:09a5] (rev 02)
pci86: ff:0a.1 System peripheral [0880]: Intel Corporation Device [8086:344d]
pci34: 00:1f.7 Non-Essential Instrumentation [1300]: Intel Corporation Atom Processor P5xxx Series Trace Hub [8086:18e1] (rev 11)
pci83: ff:00.4 System peripheral [0880]: Intel Corporation Device [8086:344c]
pci11: 00:01.5 System peripheral [0880]: Intel Corporation Ice Lake CBDMA [QuickData Technology] [8086:0b00]
pci8: 00:01.2 System peripheral [0880]: Intel Corporation Ice Lake CBDMA [QuickData Technology] [8086:0b00]
pci95: ff:1e.2 System peripheral [0880]: Intel Corporation Device [8086:345a] (rev 07)

```



```

pci100: ff:1e.7 System peripheral [0880]: Intel Corporation Device [8086:345f] (rev 07)
pci30: 00:1e.0 USB controller [0c03]: Intel Corporation Atom Processor P5xxx Series USB
xHCI Controller [8086:18d0] (rev 11)
pci79: ff:00.0 System peripheral [0880]: Intel Corporation Device [8086:344c]
pci16: 00:02.2 System peripheral [0880]: Intel Corporation Device [8086:09a8]
pci23: 00:18.0 Communication controller [0780]: Intel Corporation Atom Processor P5xxx
Series MEI Controller [8086:18d3] (rev 11)
pci3: 00:00.2 System peripheral [0880]: Intel Corporation Ice Lake RAS [8086:09a3] (rev
02)
pci13: 00:01.7 System peripheral [0880]: Intel Corporation Ice Lake CBDMA [QuickData
Technology] [8086:0b00]
pci2: 00:00.1 System peripheral [0880]: Intel Corporation Ice Lake Mesh 2 PCIe
[8086:09a4] (rev 02)
pci70: fe:00.1 System peripheral [0880]: Intel Corporation Device [8086:3451]
pci18: 00:07.0 SATA controller [0106]: Intel Corporation Atom Processor P5xxx Series SATA
Controller [8086:18b3] (rev 11)
pci28: 00:1a.3 Unassigned class [ff00]: Intel Corporation Device [8086:18d9] (rev 11)
pci31: 00:1f.0 ISA bridge [0601]: Intel Corporation Device [8086:18dc] (rev 11)
pci82: ff:00.3 System peripheral [0880]: Intel Corporation Device [8086:344c]
pci10: 00:01.4 System peripheral [0880]: Intel Corporation Ice Lake CBDMA [QuickData
Technology] [8086:0b00]
pci44: e3:00.0 System peripheral [0880]: Intel Corporation Ice Lake Memory Map/VT-d
[8086:09a2] (rev 02)
pci77: fe:0c.0 Performance counters [1101]: Intel Corporation Device [8086:344a]
pci15: 00:02.1 System peripheral [0880]: Intel Corporation Ice Lake PMON MSM [8086:09a7]
pci39: 14:00.2 System peripheral [0880]: Intel Corporation Ice Lake RAS [8086:09a3] (rev
02)
pci56: ec:00.0 Ethernet controller [0200]: Intel Corporation Ethernet Connection E822-C
for backplane [8086:1890] (rev 20)
pci48: e3:00.4 Host bridge [0600]: Intel Corporation Ice Lake IEH [8086:0998]
pci96: ff:1e.3 System peripheral [0880]: Intel Corporation Device [8086:345b] (rev 07)
pci1: 00:00.0 System peripheral [0880]: Intel Corporation Ice Lake Memory Map/VT-d
[8086:09a2] (rev 02)
pci17: 00:02.4 Non-Essential Instrumentation [1300]: Intel Corporation Ice Lake NorthPeak
[8086:3456] (rev 01)
pci73: fe:00.5 System peripheral [0880]: Intel Corporation Device [8086:3455]
pci49: e7:00.0 Co-processor [0b40]: Intel Corporation Device [8086:270b]
pci88: ff:0a.3 System peripheral [0880]: Intel Corporation Device [8086:344d]
pci63: f3:00.1 System peripheral [0880]: Intel Corporation Ice Lake Mesh 2 PCIe
[8086:09a4] (rev 02)
pci19: 00:09.0 PCI bridge [0604]: Intel Corporation Device [8086:18a4] (rev 11)
pci7: 00:01.1 System peripheral [0880]: Intel Corporation Ice Lake CBDMA [QuickData
Technology] [8086:0b00]
pci41: 14:00.4 Host bridge [0600]: Intel Corporation Ice Lake IEH [8086:0998]
pci59: ec:00.5 Ethernet controller [0200]: Intel Corporation Ethernet Connection E822-C
for backplane [8086:1890] (rev 20)
pci69: fe:00.0 System peripheral [0880]: Intel Corporation Device [8086:3460]
pci54: eb:00.4 Host bridge [0600]: Intel Corporation Ice Lake IEH [8086:0998]
pci99: ff:1e.6 System peripheral [0880]: Intel Corporation Device [8086:345e] (rev 07)
pci98: ff:1e.5 System peripheral [0880]: Intel Corporation Device [8086:345d] (rev 07)
pci51: eb:00.1 System peripheral [0880]: Intel Corporation Ice Lake Mesh 2 PCIe
[8086:09a4] (rev 02)
pci61: ec:00.7 Ethernet controller [0200]: Intel Corporation Ethernet Connection E822-C
for backplane [8086:1890] (rev 20)
pci12: 00:01.6 System peripheral [0880]: Intel Corporation Ice Lake CBDMA [QuickData
Technology] [8086:0b00]
pci5: 00:00.4 Host bridge [0600]: Intel Corporation Ice Lake IEH [8086:0998]
pci24: 00:18.4 Communication controller [0780]: Intel Corporation Atom Processor P5xxx
Series MEI Controller [8086:18d6] (rev 11)
pci46: e3:00.2 System peripheral [0880]: Intel Corporation Ice Lake RAS [8086:09a3] (rev
02)
pci78: fe:1a.0 Performance counters [1101]: Intel Corporation Device [8086:2880]
pci50: eb:00.0 System peripheral [0880]: Intel Corporation Ice Lake Memory Map/VT-d
[8086:09a2] (rev 02)

```

```

pci66: f3:00.4 Host bridge [0600]: Intel Corporation Ice Lake IEH [8086:0998]
pci40: 14:00.3 System peripheral [0880]: Intel Corporation Device [8086:09a5] (rev 02)
pci42: 14:04.0 PCI bridge [0604]: Intel Corporation Device [8086:334a]
pci60: ec:00.6 Ethernet controller [0200]: Intel Corporation Ethernet Connection E822-C
for backplane [8086:1890] (rev 20)
pci33: 00:1f.5 Serial bus controller [0c80]: Intel Corporation Atom Processor P5xxx
Series SPI Controller [8086:18e0] (rev 11)
pci91: ff:1d.0 System peripheral [0880]: Intel Corporation Device [8086:344f]
pci38: 14:00.1 System peripheral [0880]: Intel Corporation Ice Lake Mesh 2 PCIe
[8086:09a4] (rev 02)
pci55: eb:04.0 PCI bridge [0604]: Intel Corporation Device [8086:18d1] (rev 20)
pci81: ff:00.2 System peripheral [0880]: Intel Corporation Device [8086:344c]
pci43: 15:00.0 Ethernet controller [0200]: Broadcom Inc. and subsidiaries Device
[14e4:b891] (rev 01)
pci84: ff:00.5 System peripheral [0880]: Intel Corporation Device [8086:344c]
pci93: ff:1e.0 System peripheral [0880]: Intel Corporation Device [8086:3458] (rev 07)
pci80: ff:00.1 System peripheral [0880]: Intel Corporation Device [8086:344c]
pci9: 00:01.3 System peripheral [0880]: Intel Corporation Ice Lake CBDMA [QuickData
Technology] [8086:0b00]
pci72: fe:00.3 Host bridge [0600]: Intel Corporation Ice Lake IEH [8086:0998]
pci6: 00:01.0 System peripheral [0880]: Intel Corporation Ice Lake CBDMA [QuickData
Technology] [8086:0b00]
pci68: f4:00.0 Co-processor [0b40]: Intel Corporation C4xxx Series QAT [8086:18a0] (rev
20)
pci85: ff:0a.0 System peripheral [0880]: Intel Corporation Device [8086:344d]
pci14: 00:02.0 System peripheral [0880]: Intel Corporation Ice Lake MSM [8086:09a6]
pci62: f3:00.0 System peripheral [0880]: Intel Corporation Ice Lake Memory Map/VT-d
[8086:09a2] (rev 02)
pci97: ff:1e.4 System peripheral [0880]: Intel Corporation Device [8086:345c] (rev 07)
pci74: fe:0b.0 System peripheral [0880]: Intel Corporation Device [8086:3448]
pci22: 00:0f.0 System peripheral [0880]: Intel Corporation Device [8086:18ac] (rev 11)
pci29: 00:1d.0 Host bridge [0600]: Intel Corporation Ice Lake IEH [8086:0998]
pci94: ff:1e.1 System peripheral [0880]: Intel Corporation Device [8086:3459] (rev 07)
pci53: eb:00.3 System peripheral [0880]: Intel Corporation Device [8086:09a5] (rev 02)
pci27: 00:1a.2 Serial controller [0700]: Intel Corporation Device [8086:18d8] (rev 11)
pci25: 00:1a.0 Serial controller [0700]: Intel Corporation Device [8086:18d8] (rev 11)
pci64: f3:00.2 System peripheral [0880]: Intel Corporation Ice Lake RAS [8086:09a3] (rev
02)
pci76: fe:0b.2 System peripheral [0880]: Intel Corporation Device [8086:344b]
pci36: 02:00.0 Non-Volatile memory controller [0108]: Transcend Information, Inc. NVMe
PCIe SSD 110S/112S/120S/MTE300S/MTE400S/MTE652T2 (DRAM-less) [1d79:2263] (rev 03)
pci71: fe:00.2 System peripheral [0880]: Intel Corporation Device [8086:3452]
pci92: ff:1d.1 System peripheral [0880]: Intel Corporation Device [8086:3457]
pci4: 00:00.3 System peripheral [0880]: Intel Corporation Device [8086:09a5] (rev 02)
pci32: 00:1f.4 SMBus [0c05]: Intel Corporation Atom Processor P5xxx Series SMBus
Controller [8086:18df] (rev 11)
pci35: 01:00.0 Non-Volatile memory controller [0108]: Transcend Information, Inc. NVMe
PCIe SSD 110S/112S/120S/MTE300S/MTE400S/MTE652T2 (DRAM-less) [1d79:2263] (rev 03)
pci90: ff:0a.5 System peripheral [0880]: Intel Corporation Device [8086:344d]
pci67: f3:05.0 PCI bridge [0604]: Intel Corporation Device [8086:18da] (rev 20)
pci87: ff:0a.2 System peripheral [0880]: Intel Corporation Device [8086:344d]
pci58: ec:00.4 Ethernet controller [0200]: Intel Corporation Ethernet Connection E822-C
for backplane [8086:1890] (rev 20)
pci65: f3:00.3 System peripheral [0880]: Intel Corporation Device [8086:09a5] (rev 02)

-----
SSD-INFO
-----
Firmware Version: U1006A3
Sector Size: 256GB
Serial Number: I310230082
LU WWN Device Id: N/A
Rotation Rate: N/A
SATA Version: N/A

```

```
ATA Version: N/A
SSD Type: NVMe
Device Model: TS256GMTE670T
```

```
-----
CHASSIS-INFO (BRIEF)
-----
```

```
Product Name: Extreme 8730-32D
System Status: Online
Serial Number: AD012350G-00014
-----
```

```
PSU-INFO
-----
```

```
Module      : PSU-0
Model       : FSJ004EAAG
SystemNo    : 801163-00-00
SerialNo    : P0152342A-C0072
Version     : SV1.0
Type       : AC
Status      : Up
```

```
Module      : PSU-1
Model       : FSJ004EAAG
SystemNo    : 801163-00-00
SerialNo    : P0152342A-C0052
Version     : SV1.0
Type       : AC
Status      : Present
```

```
-----
FAN-INFO
-----
```

```
fan-0      Failed
fan-3      Up
fan-4      Up
fan-5      Up
fan-6      Up
Overall Fan Status : BAD - 4 out of 7 fans online
-----
```

```
SYSTEM-PERFORMANCE-SNAPSHOT
-----
```

	PID	USER	PR	NI	VIRT	RES	SHR	%CPU	%MEM	TIME+	COMMAND
T5:	15852	root	20	0	5847076	554564	292608 S	7.1	1.7	2:34.00	bcmtmC+
T6:	1	root	20	0	21200	13056	8064 S	0.0	0.0	0:49.12	systemd
T3:	15814	root	-91	0	5847076	554564	292608 S	7.1	1.7	0:39.86	bcmaSys+
T9:	4	root	0	-20	0	0	0 I	0.0	0.0	0:00.00	kworker+
T1:	6026	root	20	0	1831328	565472	138432 S	7.1	1.8	0:37.14	k3s-ser+
T2:	6215	root	20	0	1499372	186632	112704 S	7.1	0.6	0:03.19	contain+
T10:	5	root	0	-20	0	0	0 I	0.0	0.0	0:00.00	kworker+
T11:	6	root	0	-20	0	0	0 I	0.0	0.0	0:00.00	kworker+
T0:	526161	root	20	0	8248	5568	2304 R	35.7	0.0	0:00.08	top
T4:	15851	root	20	0	5847076	554564	292608 R	7.1	1.7	5:34.16	bcmtmC+
T8:	3	root	20	0	0	0	0 S	0.0	0.0	0:00.00	pool_wo+
T7:	2	root	20	0	0	0	0 S	0.0	0.0	0:00.02	kthreadd

device#

```
show system internal apigw
```

Use the **show system internal apigw** command to display details of internal active stream subscriptions, including client information, subscription mode, and start time.

Syntax

```
show system internal apigw gnmi subscriptions
```

Parameters

gnmi

Displays GNMI subscriptions information.

subscriptions

Displays active subscriptions.

Modes

EXEC (#)

Examples

The following example shows details of internal active stream subscriptions, including client information, subscription mode, and start time on the device:

```
device# show system internal apigw gnmi subscriptions

Number of active subscriptions: 1
ClientID           pathCount Subscribed
Keypath
Subscription Mode Int
erval Start Time
=====
=====
===== ===
=====
=====
10.42.0.37:42367 1 /network-instances/network-instance[name=]/state/network-
instance-id ON_CHANGE 0s
Tue Nov 26 06:09:35.923 UTC 2024
```

show system internal cdb

Use the **show system internal cdb** command to display the internal configuration database records or the timestamp, user, and IP address of the last update for a specific data path.

Syntax

```
show system internal cdb { path path | timestamp topics }
```

Parameters

path *path*

Displays the configuration database (CDB) records for the specified data path.

timestamp

Displays the timestamp, user, and IP address of the last update of the specified data path.

topics

Displays the message topics for the specified data path.

Modes

EXEC (#)

Usage Guidelines

To display the internal state database records or the timestamp, user, and IP address of the last update for a specific data path on the device, see [show system internal sdb](#) on page 305.

Examples

The following example shows the configuration database records of the /keychains data path on the device:

```
device# show system internal cdb path /keychains

key /keychains
{
  "keychain": [
    {
      "name": "testKey1",
      "config": {
        "name": "testKey1"
      }
    }
  ]
}
device#
```

The following example shows the latest timestamp, user, and IP address of the last update of the /keychains data path on the device:

```
show system internal cdb timestamp path /keychains
```

Key		User	Address	Updated
time				
=====				
=====				
/keychains/keychain[name=testKey1]				2023-06-07
09:47:13 +0000 UTC			10.42.0.1	
device#				

show system internal cli service

Use the **show system internal cli service** command to display details of all CLI commands installed for services.

Syntax

```
show system internal cli service { all | service-name }
```

Parameters

all

Displays the details of the CLI commands that are installed for all services.

service-name

Displays the details of the CLI commands that are installed for the specified service.

Modes

EXEC (#)

Usage Guidelines

Use this command to show information about the CLI commands that are installed for all services or a specified service on the device. These details include the application (such as Extreme ONE OS Switching, library, and mode where the command is available).

Examples

The following example shows details about the CLI commands that are installed for the BFD service on the device. In this example, SR indicates that the BFD service is part of the Extreme ONE OS Switching application:

```
device# show system internal cli service bfd
```

App	Service	Library	Mode	Command
SR	bfd	bfd	%s#	show
<running-config running-state> bfd				
SR	bfd	bfd	%s#	show bfd
[neighbors [vrf <NAME all>] [details]]				
SR	bfd	bfd	%s#	show bfd
neighbors application <bgp mlag ospf ospf3 static-ip tunnel> details				
SR	bfd	bfd	%s#	show bfd
neighbors dest-ip <A.B.C.D X:X::X:X> details				
SR	bfd	bfd	%s#	show bfd
neighbors dest-ip <A.B.C.D X:X::X:X> vrf <NAME all> details				
SR	bfd	bfd	%s#	show bfd
neighbors dest-ip <A.B.C.D X:X::X:X> interface <ethernet IFNAME loopback IFNAME port-channel PORANGE tunnel IFNAME>				

```

| ve IFNAME> details
SR      bfd      bfd      %s#      show bfd
neighbors interface <ethernet IFNAME | loopback IFNAME | port-channel PORANGE | tunnel
IFNAME | ve IFNAME> details
SR      bfd      bfd      %s#      show bfd
profile <NAME | all>
SR      bfd      bfd      %s#      clear
counters bfd interface <<ethernet <IFNAME | all>> | <loopback <IFNAME | all>> | <port-
channel <PORANGE | all>> | <tunnel <IFN
AME | all>> | <ve <IFNAME | all>>>
SR      bfd      bfd      %s#      clear
counters bfd dest-ip <A.B.C.D | X:X::X:X>
SR      bfd      bfd      %s#      clear
counters bfd dest-ip <A.B.C.D | X:X::X:X> vrf <NAME | all>
SR      bfd      bfd      %s(config)#      bfd
SR      bfd      bfd      %s(config-bfd)#      member
<ethernet IFNAME | loopback IFNAME | port-channel PORANGE | tunnel IFNAME | ve IFNAME>
<profile PROFNAME | shutdown>
SR      bfd      bfd      %s(config-bfd)#      no member
<ethernet IFNAME | loopback IFNAME | port-channel PORANGE | tunnel IFNAME | ve IFNAME>
<profile | shutdown>
SR      bfd      bfd      %s(config-bfd)#      [no]
profile NAME
SR      bfd      bfd      %s(config-bfd-profile-%s)#      <interval
(10-30000) min-rx (10-30000) multiplier (3-50) | no interval>
device#

```


show system internal k3s-cert-validity

Use the **show system internal k3s-cert-validity** command to display validity details of the k3s server certificate.

Syntax

```
show system internal k3s-cert-validity
```

Parameters

This command has no keywords or variables.

Modes

EXEC (#)

Usage Guidelines

This command provides details of the date and timestamp for validity of the k3s server certificate, the date and timestamp when the certificate was last generated, and the date and timestamp when certificate validity was last checked.

Examples

The following example shows the dates and timestamps for the k3s server certificate validity, the last generation of the certificate, and the last check of the certificate validity on the device:

```
device# show system internal k3s-cert-validity

Validity not after: Nov 22 10:12:17 2025 GMT
Generated:         Nov 22 10:12:17 2024 GMT
Last checked:      Nov 25 09:32:15 2024 GMT
device#
```

show system internal msg-svc

Use the **show system internal msg-svc** command to display information about the consumers, topics, and connected client applications for the internal NATS messaging service.

Syntax

```
show system internal msg-svc { consumers | topics | clients [ detail
  [ client-name client-name ] ] }
```

Parameters

consumers

Displays the messaging service consumers information.

topics

Displays the messaging service topics information.

clients [**detail** [**client-name** *client-name*]]

Displays the messaging subscription information for connected client applications. The **detail** keyword provides a consolidated view of subscription information of all the messaging service's connected client applications or a specified connected client application.

Modes

EXEC (#)

Usage Guidelines

Neural Autonomic Transport System (NATS) is an open-source infrastructure that allows data exchange among applications and services. This data is segmented in the form of messages. Extreme ONE OS uses a NATS client library to publish, subscribe, request, and reply among instances of an application or among separate applications. These applications are called *client applications* or sometimes just *clients* (because the NATS server treats them as clients).

Examples

The following example shows details about the messaging service consumers on the device:

```
device# show system internal msg-svc consumers | nomore
```

Consumer Name	Topic
Name	Ack Pending Redelivered Waiting Pending Delivered
Stream Seq# Delivered Consumer Seq# Last Delivered time	Acknowledged
Topic Seq# Acknowledged Consumer Seq# Last Acknowledged time	
=====	
=====	=====

```

=====
=====
api-gw-state$openconfig-platform-Consumer      state$openconfig-
platform      0      0      0      0      2781404
2781404      2025-02-28 17:27:54.269817861 +0000 UTC 2781404
2781404      2025-02-28 17:27:54.270670498 +0000 UTC
chassis-mgr-state$openconfig-platform-Consumer      state$openconfig-
platform      0      0      0      0      2781404
2781404      2025-02-28 17:27:54.269836328 +0000 UTC 2781404
2781404      2025-02-28 17:27:54.270597352 +0000 UTC
mlag-state$openconfig-platform-Consumer      state$openconfig-
platform      0      0      0      0      2781404
2781404      2025-02-28 17:27:54.269851856 +0000 UTC 2781404
2781404      2025-02-28 17:27:54.270631695 +0000 UTC
interface-mgr-state$openconfig-platform-Consumer      state$openconfig-
platform      0      0      0      0      2781404
2781404      2025-02-28 17:27:54.269876186 +0000 UTC 2781404
2781404      2025-02-28 17:27:54.270618625 +0000 UTC
lacp-state$openconfig-platform-Consumer      state$openconfig-
platform      0      0      0      0      2781404
2781404      2025-02-28 17:27:54.269939322 +0000 UTC 2781404
2781404      2025-02-28 17:27:54.270657723 +0000 UTC
fwd-hal-state$openconfig-platform-Consumer      state$openconfig-
platform      0      0      0      0      2781404
2781404      2025-02-28 17:27:54.26991012 +0000 UTC 2781404
2781404      2025-02-28 17:27:54.270662734 +0000 UTC
api-gw-state$openconfig-lldp-Consumer      state$openconfig-
lldp      0      0      0      0      157452
157452      2025-02-28 17:27:53.438422653 +0000 UTC 157452
157452      2025-02-28 17:27:53.4387347 +0000 UTC
interface-mgr-state$openconfig-platform-SERVICE-Consumer      state$openconfig-platform-
SERVICE      0      0      0      0      213
215      2025-02-28 12:00:08.912840738 +0000 UTC 213
215      2025-02-28 12:00:08.914720141 +0000 UTC
api-gw-state$openconfig-platform-SERVICE-Consumer      state$openconfig-platform-
SERVICE      0      0      0      0      213
213      2025-02-28 12:00:08.912891764 +0000 UTC 213
213      2025-02-28 12:00:08.914691558 +0000 UTC
bgp-state$openconfig-platform-SERVICE-Consumer      state$openconfig-platform-
SERVICE      0      0      0      0      213
213      2025-02-28 12:00:08.912928597 +0000 UTC 213
213      2025-02-28 12:00:08.91359103 +0000 UTC
arpnd-state$openconfig-platform-SERVICE-Consumer      state$openconfig-platform-
SERVICE      0      0      0      0      213
215      2025-02-28 12:00:08.912966876 +0000 UTC 213
215      2025-02-28 12:00:08.913568381 +0000 UTC
uftm-state$openconfig-platform-SERVICE-Consumer      state$openconfig-platform-
SERVICE      0      0      0      0      213
215      2025-02-28 12:00:08.913001151 +0000 UTC 213
215      2025-02-28 12:00:08.913703462 +0000 UTC
bfd-state$openconfig-platform-SERVICE-Consumer      state$openconfig-platform-
SERVICE      0      0      0      0      213
215      2025-02-28 12:00:08.913034752 +0000 UTC 213
215      2025-02-28 12:00:08.913721627 +0000 UTC
mlag-state$openconfig-platform-SERVICE-Consumer      state$openconfig-platform-
SERVICE      0      0      0      0      213
215      2025-02-28 12:00:08.913073526 +0000 UTC 213
215      2025-02-28 12:00:08.914672429 +0000 UTC
vpp-sr-state$openconfig-platform-SERVICE-Consumer      state$openconfig-platform-
SERVICE      0      0      0      0      213
213      2025-02-28 12:00:08.913111855 +0000 UTC 213
213      2025-02-28 12:00:08.91354825 +0000 UTC
mftm-state$openconfig-platform-SERVICE-Consumer      state$openconfig-platform-
SERVICE      0      0      0      0      213

```

```

213                2025-02-28 12:00:08.913152822 +0000 UTC 213
                213                2025-02-28 12:00:08.914783365 +0000 UTC
sr-api-gw-state$openconfig-platform-SERVICE-Consumer      state$openconfig-platform-
SERVICE            0            0            0            0            213
213                2025-02-28 12:00:08.913196927 +0000 UTC 213
                213                2025-02-28 12:00:08.914765516 +0000 UTC
fwd-hal-state$openconfig-platform-SERVICE-Consumer      state$openconfig-platform-
SERVICE            0            0            0            0            213
215                2025-02-28 12:00:08.913243079 +0000 UTC 213
                215                2025-02-28 12:00:08.914744415 +0000 UTC
laccp-state$openconfig-platform-SERVICE-Consumer      state$openconfig-platform-
SERVICE            0            0            0            0            213
213                2025-02-28 12:00:08.91327496 +0000 UTC 213
                213                2025-02-28 12:00:08.914648634 +0000 UTC

...

device#

```

The following example shows details about the messaging service topics on the device:

```

device# show system internal msg-svc topics | nomore

Topic Name                               Created
Retained Messages   Bytes   First sequence Last sequence Last message
time                               Consumers
=====
state$openconfig-platform                2025-02-10 12:11:22.354154538 +0000
UTC 212                                61804      2782465      2782676      2025-02-28
17:39:54.276486763 +0000 UTC 6
state$openconfig-lldp                    2025-02-10 12:11:54.162722036 +0000
UTC 12                                6348      157513      157524      2025-02-28
17:39:53.466497571 +0000 UTC 1
state$openconfig-platform-SERVICE        2025-02-10 12:11:22.543459284 +0000
UTC 0                                0         214         213         2025-02-28
12:00:08.912664354 +0000 UTC 13
state$openconfig-interfaces               2025-02-10 12:11:33.549386922 +0000
UTC 0                                0         119         118         2025-02-28
12:00:06.924189526 +0000 UTC 11
state$openconfig-system                   2025-02-10 12:11:21.836952706 +0000
UTC 0                                0         94          93         2025-02-28
12:00:09.450555644 +0000 UTC 4
state$openconfig-qos                      2025-02-10 12:11:54.147962959 +0000
UTC 0                                0         65          64         2025-02-28
12:00:09.083719581 +0000 UTC 1
sysconf$openconfig-interfaces             2025-02-10 12:11:33.506952474 +0000
UTC 0                                0         62          61         2025-02-28
12:00:10.557458207 +0000 UTC 1
sysconf$openconfig-platform              2025-02-10 12:11:33.542007951 +0000
UTC 0                                0         61          60         2025-02-28
12:00:10.899015146 +0000 UTC 1
state$openconfig-platform-APP             2025-02-10 12:11:33.547450164 +0000
UTC 0                                0         53          52         2025-02-28
12:00:08.570014748 +0000 UTC 5
state$extreme-overlay-networks            2025-02-10 12:11:54.145145644 +0000
UTC 0                                0         48          47         2025-02-28
12:00:06.032962001 +0000 UTC 5
state$openconfig-network-instance         2025-02-10 12:11:33.551639176 +0000
UTC 0                                0         42          41         2025-02-28
12:00:07.693729864 +0000 UTC 11
sysconf$openconfig-network-instance       2025-02-10 12:11:33.521066811 +0000
UTC 0                                0         40          39         2025-02-28

```

```

12:00:10.726203239 +0000 UTC 1
state$extreme-load-balance 2025-02-10 12:11:54.153045773 +0000
UTC 0 0 40 39 2025-02-28
12:00:05.504074409 +0000 UTC 1
state$openconfig-network-instance-STATIC 2025-02-10 12:11:33.545333041 +0000
UTC 0 0 40 39 2025-02-28
12:00:08.047065821 +0000 UTC 1
config$openconfig-lldp 2025-02-10 12:12:03.882066367 +0000
UTC 0 0 40 39 2025-02-28
12:00:02.997059795 +0000 UTC 1
state$openconfig-platform-CAPABILITY 2025-02-10 12:11:22.899917095 +0000
UTC 0 0 39 38 2025-02-28
12:00:08.739565014 +0000 UTC 2
sysconf$openconfig-system 2025-02-10 12:11:33.553765164 +0000
UTC 0 0 39 38 2025-02-28
12:00:11.239886185 +0000 UTC 1
state$openconfig-bfd 2025-02-10 12:11:54.15887351 +0000
UTC 0 0 39 38 2025-02-28
12:00:06.555755918 +0000 UTC 2
sysconf$openconfig-bfd 2025-02-10 12:11:54.181015984 +0000
UTC 0 0 39 38 2025-02-28
12:00:10.383990087 +0000 UTC 1
sysconf$extreme-overlay-networks 2025-02-10 12:11:54.173156246 +0000
UTC 0 0 39 38 2025-02-28
12:00:10.020168317 +0000 UTC 1
config$openconfig-system 2025-02-10 12:11:21.327885182 +0000
UTC 0 0 38 37 2025-02-28
12:00:04.455747771 +0000 UTC 21
state$extreme-mlag 2025-02-10 12:11:54.160804215 +0000
UTC 0 0 38 37 2025-02-28
12:00:05.859715658 +0000 UTC 3
config$openconfig-routing-policy 2025-02-10 12:12:13.173811082 +0000
UTC 0 0 37 36 2025-02-28
12:00:04.282264773 +0000 UTC 1
internalstate$extreme-internal-pipeline 2025-02-10 12:11:33.523320913 +0000
UTC 0 0 37 36 2025-02-28
12:00:04.971345238 +0000 UTC 1
config$openconfig-lacp 2025-02-10 12:12:08.008559154 +0000
UTC 0 0 37 36 2025-02-28
12:00:02.827425904 +0000 UTC 1
config$openconfig-networking-instance 2025-02-10 12:12:13.176537634 +0000
UTC 0 0 37 36 2025-02-28
12:00:03.764510432 +0000 UTC 1
state$openconfig-network-instance-BGP 2025-02-10 12:11:33.502156422 +0000
UTC 0 0 37 36 2025-02-28
12:00:07.873833055 +0000 UTC 1
internalstate$extreme-internal-sap 2025-02-10 12:11:33.50961715 +0000
UTC 0 0 37 36 2025-02-28
12:00:05.144400466 +0000 UTC 1
config$openconfig-network-instance 2025-02-10 12:11:56.207145488 +0000
UTC 0 0 37 36 2025-02-28
12:00:03.172462383 +0000 UTC 4
state$openconfig-keychain 2025-02-10 12:11:33.532673439 +0000
UTC 0 0 37 36 2025-02-28
12:00:07.111849131 +0000 UTC 2
config$openconfig-qos 2025-02-10 12:12:09.164396611 +0000
UTC 0 0 37 36 2025-02-28
12:00:04.107820407 +0000 UTC 1
...
device#

```

The following example shows a list of the messaging service client applications on the device:

```
device# show system internal msg-svc clients | nomore
```

Client Name	Client IP	Port	Uptime
chassis-mgr	10.42.0.35	36955	18d5h30m41s
api-gw	10.42.0.32	52689	18d5h30m37s
interface-mgr	10.42.0.1	48571	18d5h30m41s
mlag	10.42.0.1	25346	18d5h30m7s
fwd-hal	10.42.0.1	63984	18d5h30m7s
lACP	10.42.0.42	39444	18d5h29m55s
lldp	10.42.0.1	36610	18d5h30m7s
arpnd	10.42.0.1	4082	18d5h30m7s
uftp	10.42.0.1	7732	18d5h30m7s
bfd	10.42.0.1	40582	18d5h30m7s
bgp	10.42.0.1	57258	18d5h30m7s
classifiers	10.42.0.1	10431	18d5h30m6s
mftm	10.42.0.41	55949	18d5h30m6s
iah	10.42.0.34	39741	18d5h30m7s
telegraf	10.42.0.38	59133	18d5h30m7s
sr-api-gw	10.42.0.33	53009	18d5h30m17s
vpp-sr	10.42.0.1	60050	18d5h30m5s
lACP	10.42.0.42	48575	18d5h30m6s
security	10.42.0.1	36086	18d5h30m18s
monitor-svc	10.42.0.1	14275	18d5h30m40s
snmp-agent	10.42.0.40	60077	18d5h30m17s
ingress-gw	10.42.0.1	42687	18d5h30m41s
onlp-svc	10.42.0.1	54647	18d5h30m41s

```
device#
```

The following example shows details about the messaging service client applications on the device:

```
device# show system internal msg-svc clients detail | nomore
```

```
Client Name: chassis-mgr Client IP: 10.42.0.35 Port: 36955
Uptime: 18d5h25m19s
```

Consumer Name	Topic
Name	Ack Pending Redelivered Waiting Pending Delivered Topic
Seq# Delivered Consumer Seq# Last Delivered time	Acknowledged Topic
Seq# Acknowledged Consumer Seq# Last Acknowledged time	
=====	
=====	
=====	
=====	
chassis-mgr-state\$openconfig-platform-Consumer	state\$openconfig-
platform 0 0	
0 0 2782312 2782312	2025-02-28
17:36:34.302747866 +0000 UTC 2782312	
2782312 2025-02-28 17:36:34.303855553 +0000 UTC	
chassis-mgr-state\$openconfig-platform-APP-Consumer	state\$openconfig-platform-
APP 0 0	
0 0 52 52	2025-02-28
12:00:08.570264351 +0000 UTC 52	
52 2025-02-28 12:00:08.57069889 +0000 UTC	
chassis-mgr-state\$openconfig-platform-CAPABILITY-Consumer	state\$openconfig-platform-
CAPABILITY 0 0	
0 0 38 38	2025-02-28
12:00:08.739674992 +0000 UTC 38	
38 2025-02-28 12:00:08.742554009 +0000 UTC	
chassis-mgr-config\$openconfig-system-Consumer	config\$openconfig-
system 0 0	

```

0      0      37      37      2025-02-28
12:00:04.456500388 +0000 UTC 37
      37      2025-02-28 12:00:04.457554696 +0000 UTC
chassis-mgr-config$openconfig-platform-Consumer      config$openconfig-
platform      0      0
0      0      36      36      2025-02-28
12:00:03.938769497 +0000 UTC 36
      36      2025-02-28 12:00:03.939133426 +0000 UTC

Client Name: api-gw Client IP: 10.42.0.32 Port: 52689
Uptime: 18d5h25m15s
Consumer Name      Topic
Name      Ack Pending Redelivered Waiting Pending Delivered
Topic Seq# Delivered Consumer Seq# Last Delivered time      Acknowledged
Topic Seq# Acknowledged Consumer Seq# Last Acknowledged time
=====
=====
=====
=====
api-gw-state$openconfig-platform-Consumer      state$openconfig-
platform      0      0      0      0      2782312
2782312      2025-02-28 17:36:34.30288704 +0000 UTC 2782312
      2782312      2025-02-28 17:36:34.30356179 +0000 UTC
api-gw-state$openconfig-lldp-Consumer      state$openconfig-
lldp      0      0      0      0      157504
157504      2025-02-28 17:36:33.45884963 +0000 UTC 157504
      157504      2025-02-28 17:36:33.459292972 +0000 UTC
api-gw-state$openconfig-platform-SERVICE-Consumer      state$openconfig-platform-
SERVICE      0      0      0      0      213
213      2025-02-28 12:00:08.912891764 +0000 UTC 213
      213      2025-02-28 12:00:08.914691558 +0000 UTC
api-gw-state$openconfig-interfaces-Consumer      state$openconfig-
interfaces      0      0      0      0      118
118      2025-02-28 12:00:06.924540486 +0000 UTC 118
      118      2025-02-28 12:00:06.926387205 +0000 UTC
api-gw-state$openconfig-system-Consumer      state$openconfig-
system      0      0      0      0      93
93      2025-02-28 12:00:09.450761507 +0000 UTC 93
      93      2025-02-28 12:00:09.450950678 +0000 UTC

...

device#

```

The following example shows details about the security client application on the device:

```

device# show system internal msg-svc clients detail client-name security | nomore

Client Name: security Client IP: 10.42.0.1 Port: 36086
Uptime: 18d5h31m26s
Consumer Name      Topic Name
Ack Pending Redelivered Waiting Pending Delivered Topic Seq# Delivered Consumer Seq# Last
Delivered time      Acknowledged Topic Seq# Acknowledged Consumer Seq#
Last Acknowledged time
=====
=====
=====
=====
security-state$openconfig-network-instance-Consumer state$openconfig-network-instance
0      0      0      0      41      41
2025-02-28 12:00:07.693963009 +0000 UTC 41
41      2025-02-28 12:00:07.698179146 +0000 UTC
security-config$openconfig-system-Consumer      config$openconfig-system

```

```
0          0          0          0          37          37
2025-02-28 12:00:04.456120603 +0000 UTC 37
37          2025-02-28 12:00:04.456551175 +0000 UTC
security-config$openconfig-keychain-Consumer          config$openconfig-keychain
0          0          0          0          36          36
2025-02-28 12:00:02.654439387 +0000 UTC 36
36          2025-02-28 12:00:02.654725306 +0000 UTC
device#
```


show system internal sdb

Use the **show system internal sdb** command to display the internal state database records or the timestamp, user, and IP address of the last update for a specific data path.

Syntax

```
show system internal sdb { path path | timestamp topics }
```

Parameters

path *path*

Displays the state database (SDB) records for the specified data path.

timestamp

Displays the timestamp, user, and IP address of the last update of the specified data path.

topics

Displays the message topics for the specified data path.

Modes

EXEC (#)

Usage Guidelines

To display the internal configuration database records or the timestamp, user, and IP address of the last update for a specific data path on the device, see [show system internal cdb](#) on page 293.

Examples

The following example shows the state database records of the /keychains data path on the device:

```
device# show system internal sdb path /keychains

key /keychains
{
  "keychain": [
    {
      "name": "testKey1",
      "state": {
        "name": "testKey1"
      }
    }
  ]
}
device#
```

The following example shows the latest timestamp, user, and IP address of the last update of the /keychains data path on the device:

```
show system internal sdb timestamp path /keychains
```

Key	Updated time	User	Address
=====			
=====			
/keychains/keychain[name=testKey1]	2023-06-14 12:48:51 +0000 UTC		
device#			

show system location

Use the **show system location** command to display the administrative location details that are configured for a deployed device.

Syntax

```
show system location
```

Parameters

This command has no keywords or variables.

Modes

EXEC (#)

Usage Guidelines

Use this command to display the device's administrative location details.

To configure the administrative location details for the device, see [location \(Extreme ONE OS\)](#) on page 149.

Examples

The following example displays the device's administrative location details:

```
device# show system location

location: Bangalore
device#
```

The following example displays the configuration of the administrative location details that is running currently on the device:

```
device# show running-config system location

system
  location Bangalore
!
device#
```

show system logging facility

Use the **show system logging facility** command to display the severity level with respect to facilities on a device.

Syntax

```
show system logging facility [ facility-name ]
```

Parameters

facility-name

(Optional) Name of the facility (microservice). This can be a base Extreme ONE OS service or a service for an application such as Extreme ONE OS Switching. Use any combination of 1 to 64 alphanumeric characters, space, underscores, hyphens, or periods. For example, interface-mgr or mlag.

The base Extreme ONE OS services are: api-gw, chassis-mgr, config-db, ingress-gw, interface-mgr, msg-bus, security, snmp-agent, state-db, base-svc, and monitor-svc.

The Extreme ONE OS Switching application services are: arpnd, bfd, bgp, classifiers, fwd-hal, lacp, lldp, mftm, mlag, sr-api-gw, telegraf, and uftm.

Modes

EXEC (#)

Usage Guidelines

To configure the severity level with respect to facilities on the device, see [facility](#) on page 118.

Output

The **show system logging facility** command displays the following information:

Output field	Description
Service	Name of the facility.
Severity	Severity level. The possible levels are emergency, alert, critical, error, warning, notice, info, or debug.

Examples

The following example shows the severity levels with respect to all facilities on the device:

```
device# show system logging facility
```

```
Service          Severity
=====
mlog              INFORMATIONAL
arpnd            DEBUG
device#
```

The following example shows the severity level with respect to the mlog facility on the device:

```
device# show system logging facility mlog

Service          Severity
=====
mlog              INFORMATIONAL
device#
```

show system logging remote-server

Use the **show system logging remote-server** command to display details about the remote server configuration on a device.

Syntax

```
show system logging remote-server
```

Parameters

This command has no keywords or variables.

Modes

EXEC (#)

Usage Guidelines

To configure the severity level with respect to facilities on the device, see [facility](#) on page 118.

Output

The **show system logging remote-server** command displays the following information:

Output field	Description
REMOTE SERVER	Name of the remote system logging server.
ADDRESS	IP address of the remote system logging server.
PORT	Number of the destination port on the remote system logging server.
TRANSPORT	Protocol that is used to send system logs (UDP or TCP).
SECURE-FORWARDING	Indicates whether secure forwarding is enabled. This is either NONE or TLS.
VRF-NAME	Name of the VRF used to route system logging communications.

Examples

The following example shows details about the remote server configuration on the device:

```
device# show system logging remote-server

HOST          PORT  TRANSPORT  SECURE-FORWARDING  VRF-NAME
=====

```

```
1.1.1.1      514    UDP    TLS      test1
device#
```

show system mac-address-table

Use the **show system mac-address-table** command to display the MAC forwarding table aging time.

Syntax

```
show system mac-address-table aging-time
```

Parameters

aging-time

Displays aging time (in seconds).

Modes

EXEC (#)

Usage Guidelines

Use this command to display the device's MAC forwarding table aging time. This means that after the specified period of inactivity, a MAC address is deleted from the table.

Examples

The following example displays the MAC forwarding table aging time:

```
device# show system mac-address-table aging-time  
  
Mac Aging Time: 1800  
device#
```


show system service (Extreme ONE OS)

Use the **show system service** command to display the running microservices attributes for a deployed device.

Syntax

```
show system service
```

Parameters

This command has no keywords or variables.

Modes

EXEC (#)

Usage Guidelines

Use this command to display details about each service that is running on the device. These details include the service name, restart count, current version, rollback version, ready status, and current state.

Examples

The following example displays the running microservices attributes on the device:

```
device# show system service | nomore
```

BASE Services:					
SERVICE	CURRENT VERSION	ROLLBACK VERSION	READY	STATE	RESTARTS

api-gw	1.1.0	None	true	Running	0
chassis-mgr	1.2.0	None	true	Running	0
config-db	1.1.0	None	true	Running	0
ingress-gw	1.0.0	None	true	Running	0
interface-mgr	1.2.0	None	true	Running	0
msg-bus	1.1.0	None	true	Running	0
security	1.1.0	None	true	Running	0
snmp-agent	1.1.0	None	true	Running	0
state-db	1.1.0	None	true	Running	0
ztp-base	1.0.0	None	true	Running	0
SR Services:					
SERVICE	CURRENT VERSION	ROLLBACK VERSION	READY	STATE	RESTARTS

arpnd	0.1.0	None	true	Running	0
bfd	0.1.0	None	true	Running	0
bgp	0.1.0	None	true	Running	0
classifiers	0.1.0	None	true	Running	0
fwd-hal	0.1.0	None	true	Running	0
iah	0.1.0	None	true	Running	0
lACP	1.0.0	None	true	Running	0

```

lldp          1.0.0      None      true      Running    0
mftm          0.1.0      None      true      Running    0
mlag          0.1.0      None      true      Running    0
sr-api-gw     1.0.0      None      true      Running    0
telegraf      1.0.0      None      true      Running    0
uftm          0.1.0      None      true      Running    0
vpp-sr        0.1.0      None      true      Running    0
Reconciliation Status info:
SERVICE      RECONCILIATION      RECONCILIATION FAILURE
              STATUS      REASON
-----
chassis-mgr   active-state        None
interface-mgr active-state        None
security      active-state        None
snmp-agent    active-state        None
arpnd         active-state        None
bfd           active-state        None
bgp           active-state        None
fwd-hal       active-state        None
iah           active-state        None
lACP          active-state        None
lldp          active-state        None
mlag          active-state        None
uftm          active-state        None
device#

```

show tech-support

Use the **show tech-support** command to display a list of the existing tech support files on the device.

Syntax

```
show tech-support
```

Parameters

This command has no keywords or variables.

Modes

EXEC (#)

Usage Guidelines

This command lists each available tech support file as well as its file size and the date and time of its creation. Tech support files reside in the `/var/data/disk/techsupport` directory.

If you want to clear all tech support files or a specific tech support file, see [clear tech-support](#) on page 70.

If you want to generate new tech support files, see [system tech-support](#) on page 364.

Examples

The following example displays a list of the existing tech support files on the device:

```
device# show tech-support
File Name                               Size (bytes)  Created
TS-dutb-250922T181951.log              7135         2025-09-22 18:19:55
TS-dutb-250922T181951.tar.gz           2620075      2025-09-22 18:19:55
TS-dutb-251028T160410.log              8037         2025-10-28 16:04:54
TS-dutb-251028T160410.tar.gz          12298874     2025-10-28 16:04:54
device#
```

show tls min-version

Use the **show tls min-version** command to display the minimum version of Transport Layer Security (TLS) used by the services that are running on a device.

Syntax

```
show tls min-version
```

Parameters

This command has no keywords or variables.

Modes

EXEC (#)

Usage Guidelines

To set the global TLS minimum version for all TLS-enabled services on a device, see [min-version \(TLS system configuration\)](#) on page 162. To set the TLS required minimum version for a specific TLS-enabled application on a device, see [min-version \(TLS service-level system configuration\)](#) on page 160.

Examples

The following example displays the minimum version of TLS used by services on the device:

```
device# show tls min-version

Service      TLS min-version
-----
grpc-server  1.2
syslog-client 1.2
ldap-client  1.2
radius-client 1.2
https-client 1.2
device#
```

show trace

Use the **show trace** command to display the trace logs on a device or trace logs for a specific microservice on a device.

Syntax

```
show trace [ microservice-name ]
```

Parameters

service-name

(Optional) Name of the facility (microservice). This can be a base Extreme ONE OS service or a service for an application such as Extreme ONE OS Switching. Use any combination of 1 to 64 alphanumeric characters, space, underscores, hyphens, or periods. For example, interface-mgr or mlag.

The base Extreme ONE OS services are: api-gw, chassis-mgr, config-db, ingress-gw, interface-mgr, msg-bus, security, snmp-agent, state-db, base-svc, and monitor-svc.

The Extreme ONE OS Switching application services are: arpnd, bfd, bgp, classifiers, fwd-hal, lacp, lldp, mftm, mlag, sr-api-gw, telegraf, and uftm.

Modes

EXEC (#)

Output

The **show trace** command displays the following information:

Output field	Description
"Level"	Level.
"Service"	Name of the microservice.
"Time"	Date and time that the message was logged.
"Msg"	Message content.

Examples

The following example shows the trace logs for the arpnd microservice:

```
device# show trace arpnd
```

```
{"Level":"trace","Service":"arpnd","Time":"2022-09-22 22:33:03.564 UTC +0000","Msg":"K3S  
Health check: /alive"}  
{"Level":"trace","Service":"arpnd","Time":"2022-09-22 22:33:06.999 UTC +0000","Msg":"K3S  
Health check: /ready"}  
{"Level":"trace","Service":"arpnd","Time":"2022-09-22 22:33:13.565 UTC +0000","Msg":"K3S  
Health check: /alive"}
```

```
{ "Level": "trace", "Service": "arpnd", "Time": "2022-09-22 22:33:16.963 UTC +0000", "Msg": "K3S
Health check: /ready" }
{ "Level": "trace", "Service": "arpnd", "Time": "2022-09-22 22:33:23.561 UTC +0000", "Msg": "K3S
Health check: /alive" }
{ "Level": "trace", "Service": "arpnd", "Time": "2022-09-22 22:33:26.964 UTC +0000", "Msg": "K3S
Health check: /ready" }
{ "Level": "trace", "Service": "arpnd", "Time": "2022-09-22 22:33:33.563 UTC +0000", "Msg": "K3S
Health check: /alive" }
{ "Level": "trace", "Service": "arpnd", "Time": "2022-09-22 22:33:36.964 UTC +0000", "Msg": "K3S
Health check: /ready" }
{ "Level": "trace", "Service": "arpnd", "Time": "2022-09-22 22:33:43.566 UTC +0000", "Msg": "K3S
Health check: /alive" }
{ "Level": "trace", "Service": "arpnd", "Time": "2022-09-22 22:33:46.978 UTC +0000", "Msg": "K3S
Health check: /ready" }
{ "Level": "trace", "Service": "arpnd", "Time": "2022-09-22 22:33:53.562 UTC +0000", "Msg": "K3S
Health check: /alive" }
{ "Level": "trace", "Service": "arpnd", "Time": "2022-09-22 22:33:56.964 UTC +0000", "Msg": "K3S
Health check: /ready" }
{ "Level": "trace", "Service": "arpnd", "Time": "2022-09-22 22:34:03.567 UTC +0000", "Msg": "K3S
Health check: /alive" }
{ "Level": "trace", "Service": "arpnd", "Time": "2022-09-22 22:34:06.976 UTC +0000", "Msg": "K3S
Health check: /ready" }
{ "Level": "trace", "Service": "arpnd", "Time": "2022-09-22 22:34:13.560 UTC +0000", "Msg": "K3S
Health check: /alive" }
{ "Level": "trace", "Service": "arpnd", "Time": "2022-09-22 22:34:16.964 UTC +0000", "Msg": "K3S
Health check: /ready" }
{ "Level": "trace", "Service": "arpnd", "Time": "2022-09-22 22:34:23.560 UTC +0000", "Msg": "K3S
Health check: /alive" }
{ "Level": "trace", "Service": "arpnd", "Time": "2022-09-22 22:34:26.967 UTC +0000", "Msg": "K3S
Health check: /ready" }
device#
```

show usb (Extreme ONE OS)

Use the **show usb** command to display whether USB storage is enabled on a device.

Syntax

show usb

Parameters

This command has no keywords or variables.

Modes

EXEC (#)

Usage Guidelines

To disable or enable USB storage on the device, see [usb enable \(Extreme ONE OS\)](#) on page 384.

Examples

The following example displays whether USB storage is enabled on the device:

```
device# show usb

USB Enabled:  false
device#
```

show users (Extreme ONE OS)

Use the **show users** command to display details about active users and locked users.

Syntax

```
show users
```

Parameters

This command has no keywords or variables.

Modes

EXEC (#)

Usage Guidelines

This command displays the currently logged in users' information as well as information about any locked users on a device.

Examples

The following example displays the details about the active users and the locked users on the device:

```
device# show users

Active user sessions:
Username          Role      Host IP
Device            Time Logged In
=====
admin             admin     -
CONSOLE          Feb 10
admin             admin     198.51.100.0
CLI               Feb 10
Total number of active user sessions : 2

Locked users:
Username
=====
Total number of locked users : 0
device#
```


show version (Extreme ONE OS)

Use the **show version** command to display details about the versions of current and rollback firmware on a device.

Syntax

```
show version
```

Parameters

This command has no keywords or variables.

Modes

EXEC (#)

Usage Guidelines

Use this command to display details about the versions of current and rollback firmware (if any) on the system kernel. This command displays system uptime and commit status of the firmware. It lists the microservices along with their state and version.

This command also displays the versions of the complex programmable logic device (CPLD) modules on system. The number of CPLD modules varies by platform.

Examples

The following example displays the details about the versions of current and rollback firmware that is installed on the device:

```
device# show version

Extreme ONE OS
Copyright (c) 2025 Extreme Networks Inc.

Firmware Info:
Base OS:
Current Version:          10.1.6.0-157
Rollback Version:         None
Is Current Version Committed: True

Switching Routing (SR):
Current Version:          22.2.1.0-157
Rollback Version:         22.2.1.0-127
Is Current Version Committed: False

FPGA Version:
Built Time:
Install Time:
Kernel:                   6.6.54-yocto-standard

System Uptime:            608h54m23s
```

MicroService Info:

BASE Services:

SERVICE	CURRENT VERSION	ROLLBACK VERSION	READY	STATE	RESTARTS

api-gw	1.1.0	None	true	Running	0
chassis-mgr	1.2.0	None	true	Running	0
config-db	1.1.0	None	true	Running	0
ingress-gw	1.0.0	None	true	Running	0
interface-mgr	1.2.0	None	true	Running	0
msg-bus	1.1.0	None	true	Running	0
security	1.1.0	None	true	Running	0
snmp-agent	1.1.0	None	true	Running	0
state-db	1.1.0	None	true	Running	0
ztp-base	1.0.0	None	true	Running	0

SR Services:

SERVICE	CURRENT VERSION	ROLLBACK VERSION	READY	STATE	RESTARTS

arpnd	0.1.0	None	true	Running	0
bfd	0.1.0	None	true	Running	0
bgp	0.1.0	None	true	Running	0
classifiers	0.1.0	None	true	Running	0
fwd-hal	0.1.0	None	true	Running	0
iah	0.1.0	None	true	Running	0
lACP	1.0.0	None	true	Running	0
lldp	1.0.0	None	true	Running	0
mftm	0.1.0	None	true	Running	0
mlag	0.1.0	None	true	Running	0
sr-api-gw	1.0.0	None	true	Running	0
telegraf	1.0.0	None	true	Running	0
uftm	0.1.0	None	true	Running	0
vpp-sr	0.1.0	None	true	Running	0
device#					

show ztp dhcp logs

Use the **show ztp dhcp logs** command to display the log from the ZTP over DHCP process.

Syntax

```
show ztp dhcp logs
```

Parameters

This command has no keywords or variables.

Modes

EXEC (#)

Usage Guidelines

Use this command to show the log files that describe the current state of the zero touch provisioning (ZTP) over DHCP process. This includes each step of downloading files from a remote server and applying the configurations.

ZTP over DHCP is an automated way to provision and configure large numbers of network devices without manual intervention. It uses the DHCP protocol to deliver initial configuration or bootstrap instructions when a device first powers on.

ZTP can also be used to update existing system configurations via scripts.

When you connect a new Extreme device to your network, it loads the appropriate firmware and configurations automatically without manual intervention.

To cancel ZTP over DHCP on a device, see [ztp dhcp cancel](#) on page 403. To view the current status and server details associated with ZTP over DHCP, see [show ztp dhcp status](#) on page 325.

Examples

The following example displays the log files from the ZTP over DHCP process on the device. In this example, the process completed successfully:

```
device# show ztp dhcp logs

2025/11/30 06:34:41.874547 ZTP mode is enabled on device and config terminal is
blocked.Use 'ztp dhcp cancel' or wait for ZTP process to complete.
2025/11/30 06:34:41.880658 Waiting for updates on dhcp option 66 and 67 from dhcp server.
2025/11/30 06:35:08.542569 Received dhcpcoption 66:198.51.100.0 and dhcpcoption 67: config/
files/ztp.conf
2025/11/30 06:35:08.542806 Starting ZTP DHCP Process.
2025/11/30 06:35:08.542925 Received dhcpcoption 43: false.No CA certificate for HTTPS
server from DHCP server.Expect to try download from HTTPS server insecure way.
2025/11/30 06:35:08.545769 Received dhcpcoption 66:198.51.100.0 and dhcpcoption 67: config/
files/ztp.conf
```

```
2025/11/30 06:35:08.545848 Starting ZTP DHCP Process.
2025/11/30 06:36:08.557782 Attempting download from URL : https://198.51.100.0/config/
files/ztp.conf.
2025/11/30 06:36:08.580980 Attempting download from URL : http://198.51.100.0/config/
files/ztp.conf.
2025/11/30 06:36:09.099712 Successfully downloaded ZTP configuration file config/files/
ztp.conf to /var/data/ztp/ztp.conf.
2025/11/30 06:36:09.112303 ZTP Configuration sanity check pass
2025/11/30 06:36:09.117227 Attempting download from URL : https://198.51.100.0//config/
files/script.py.
2025/11/30 06:36:09.123054 Attempting download from URL : http://198.51.100.0//config/
files/script.py.
2025/11/30 06:36:09.647748 Successfully downloaded python script /config/files/script.py
to /var/data/ztp/script.py.
2025/11/30 06:36:09.648020 Starting to download firmware [ /config/files/
ExtremeOneSR-22.2.1.0.bin ].
2025/11/30 06:36:09.651187 Attempting download from URL : https://198.51.100.0//config/
files/ExtremeOneSR-22.2.1.0.bin.
2025/11/30 06:36:09.654625 Attempting download from URL : http://198.51.100.0//config/
files/ExtremeOneSR-22.2.1.0.bin.
2025/11/30 06:36:37.184842 Successfully Downloaded firmware.
2025/11/30 06:36:37.185524 Activate this new firmware and reboot.
2025/11/30 03:45:15.103309 ZTP mode is enabled on device and config terminal is
blocked.Use 'ztp dhcp cancel' or wait for ZTP process to complete.
2025/11/30 03:45:15.104254 Starting ZTP DHCP Process.
2025/11/30 03:45:17.104380 Device booted after ztp firmware upgrade.2025/11/30
06:37:09.539884 Running configuration script : startup.cfg.
2025/11/30 06:37:12.793263 Completed executing script : startup.cfg.Refer /var/
data/ztp/log/configReplay.log for any unsuccessful commands
2025/11/30 06:37:16.506938 Running configuration python script : /config/files/script.py.
2025/11/30 06:37:49.606110 Completed executing python script : /config/files/
script.py.Refer /var/data/ztp/log/pythonZtp.log for any unsuccessful commands
2025/11/30 06:37:49.616706 ZTP DHCP Completed.
device#
```

show ztp dhcp status

Use the **show ztp dhcp status** command to display the current status and server details for ZTP over DHCP.

Syntax

```
show ztp dhcp status
```

Parameters

This command has no keywords or variables.

Modes

EXEC (#)

Usage Guidelines

Use this command to display the current status and server details for the zero touch provisioning (ZTP) over DHCP process. This includes the process status (such as not started, in progress, completed, or canceled) and ZTP DHCP details (such as the network settings, firmware image name on the remote server and its download state, and ZTP-related configuration file names).

ZTP over DHCP is an automated way to provision and configure large numbers of network devices without manual intervention. It uses the DHCP protocol to deliver initial configuration or bootstrap instructions when a device first powers on.

ZTP can also be used to update existing systems via scripts.

When you connect a new Extreme device to your network, it loads the appropriate firmware and configurations automatically without manual intervention.

To cancel ZTP over DHCP on a device, see [ztp dhcp cancel](#) on page 403. To view the log files from the ZTP over DHCP process, see [show ztp dhcp logs](#) on page 323.

Examples

The following example displays the current status and server details for ZTP over DHCP on the device. In this example, the process has completed:

```
device# show ztp dhcp status

ZTP Status: Completed
Client-id: EXTREMENETWORKS##8730-32D##AD012451G-00007
ZTP Interface: mal
Interface IP: 10.32.129.191
Http Server IP Address: 10.64.13.105
Transfer Mode: HTTP
ZTP Configuration File: config/files/ztp.conf
Switch Configuration File: startup.cfg, /config/files/script_device.py
```

```
Firmware Image: /config/files/ExtremeOneSR-22.2.1.0.bin  
Firmware Download State: firmwareInstallCompleted  
device#
```

shutdown (BMC interface configuration)

Use the **shutdown** command to enable or disable the BMC management interface.

Syntax

```
shutdown  
no shutdown
```

Command Default

The BMC management interface is disabled.

Parameters

This command has no keywords or variables.

Modes

BMC interface configuration (interface-*name*)

Usage Guidelines

The BMC is a microcontroller and the major component of the Intelligent Platform Management Interface (IPMI) subsystem, which assists in remote hardware monitoring, management, and troubleshooting of the device.

This command requires up to one minute for the change to take effect on the BMC hardware. This delay also applies to the settings such as the IP address, subnet mask, default gateway IP address, or reset. This delay is an inherent hardware attribute of a BMC.

To display details about the internal BMC LAN configuration on the device, see [show bmc-status \(Extreme ONE OS\)](#) on page 225.

Examples

The following example configures and enables the BMC management interface on the device:

```
device# configure terminal  
device(config)# bmc  
device(bmc)# interface 0  
device(interface-0)# no shutdown  
device(interface-0)#
```

The following example shows the BMC configuration that is running currently on the device. In this example, **no shutdown** indicates that the interface is not shut down:

```
device# show running-config bmc
```

```
!  
bmc  
  interface 0  
    ipv4 address dhcp  
    no shutdown  
  !  
!  
device#
```

The following example shows the BMC state that is running currently on the device:

```
device# show running-state bmc  
  
!  
bmc  
  interface 0  
    ipv4 address dhcp  
    no shutdown  
  !  
!  
device#
```


snmp-server (Extreme ONE OS)

Use the **snmp-server** command to enter SNMP server system configuration (config-system-snmp-server) mode.

Syntax

```
snmp-server server-name
```

Command Default

No Simple Network Management Protocol (SNMP) server instance is configured on the device.

Parameters

server-name

Name of the SNMP server instance to configure. Use a string from 1 to 64 characters. Only an SNMP server instance named DEFAULT is allowed in this release.

Modes

System configuration (config-system)

Usage Guidelines

Use this command to enter the mode for configuring SNMP server settings. These settings include the community string (for SNMPv1 and SNMPv2c), or the server username and password (for SNMPv3), and the settings (such as the trap host, trap host source interface, trap host UDP port, and SNMP version) on the VRF to associate with the SNMP server.

Examples

The following example enters SNMP server system configuration (config-system-snmp-server) mode and configures a Simple Network Management Protocol (SNMP) agent with the trap destination information with the combination of a host and a username attached to it.

```
device# configure terminal
device(config)# system
device(config-system)# snmp-server DEFAULT
device(config-system-snmp-server)# user user3 auth noauth
device(config-system-snmp-server)# vrf mgmt-vrf
device(config-system-snmp-server-vrf-mgmt-vrf)# enable
kvm-x86-64(config-system-snmp-server-vrf-mgmt-vrf)# host 10.23.17.128 user3
kvm-x86-64(config-system-snmp-server-vrf-mgmt-vrf-host-10.23.17.128-****)# version 3
kvm-x86-64(config-system-snmp-server-vrf-mgmt-vrf-host-10.23.17.128-user3)# udp-port 200
kvm-x86-64(config-system-snmp-server-vrf-mgmt-vrf-host-10.23.17.128-user3)# source-
```

```
interface ethernet 0/1
kvm-x86-64 (config-system-snmp-server-vrf-mgmt-vrf-host-10.23.17.128-user3) #
```

The following example displays the SNMP server configuration that is running currently on the device:

```
device# show running-config system snmp-server

system
  snmp-server DEFAULT
  user user3 auth noauth
  vrf mgmt-vrf
    enable
    host 10.23.17.128 user3
    version 3
    udp-port 200
    source-interface ethernet 0/1.0
  !
!
!
device#
```

source-interface

Use the **source-interface** command to specify the source interface for protocol packets for an LDAP, RADIUS, or TACACS+ server or for sending traps to the trap host for an SNMP server.

Syntax

```
source-interface { ethernet interface-name | loopback port-number | port-channel port-range | ve interface-name }  
no source-interface
```

Command Default

No source interface for the specified server is configured on the device.

Parameters

ethernet *interface-name*

Ethernet interface name. Use *slot/port* format. For example, 1/1.

loopback *port-number*

Loopback port number. Use an integer from 1 to 1024.

port-channel *port-range*

Port-channel port range. Use an integer from 1 to 255.

ve *interface-name*

Virtual Ethernet (VE) interface name. Use an integer or one or more comma-separated ranges of integers from 1 to 8192. For example, 2 or 2-3,5,7-9.

Modes

AAA LDAP server group server system configuration (config-system-aaa-server-group-ldap-server-address)

AAA RADIUS server group server system configuration (config-system-aaa-server-group-radius-server-address)

SNMP server VRF host system configuration (config-system-snmp-server-vrf-vrf-name-host-host-name-community-or-user-name)

AAA TACACS+ server group server system configuration (config-system-aaa-server-group-tacacs+-server-address)

Usage Guidelines

To remove the source interface for the specified server, use the **no** form of this command.

Examples

The following example configures a Simple Network Management Protocol (SNMP) agent with the trap destination information with the combination of a host and a username attached to it. In this example, the trap host uses Ethernet interface 0/1 as the source interface, which determines the source IP address to be used while sending traps to the trap host:

```
device# configure terminal
device(config)# system
device(config-system)# snmp-server DEFAULT
device(config-system-snmp-server)# user user3 auth noauth
device(config-system-snmp-server)# vrf mgmt-vrf
device(config-system-snmp-server-vrf-mgmt-vrf)# enable
device(config-system-snmp-server-vrf-mgmt-vrf)# host 192.168.1.1 user3
device(config-system-snmp-server-vrf-mgmt-vrf-host-192.168.1.1-xxxx)# version 3
device(config-system-snmp-server-vrf-mgmt-vrf-host-192.168.1.1-user3)# udp-port 200
device(config-system-snmp-server-vrf-mgmt-vrf-host-192.168.1.1-user3)# source-interface ethernet 0/1
device(config-system-snmp-server-vrf-mgmt-vrf-host-192.168.1.1-user3)
```

The following example displays the SNMP server configuration that is running currently on the device:

```
device# show running-config system snmp-server

system
  snmp-server DEFAULT
  user user3 auth noauth
  vrf mgmt-vrf
  enable
  host 192.168.1.1 user3
  version 3
  udp-port 200
  source-interface ethernet 0/1.0
  !
!
!
!
device#
```

The following example associates a VRF instance named red-vrf with the Terminal Access Controller Access-Control System Plus (TACACS+) server host 6.6.6.6. In this example, the host uses Ethernet interface 0/1.0 as the source interface, which determines the source of TACACS+ protocol packets:

```
device# configure terminal
device(config)# system
device(config-system)# aaa
device(config-system-aaa)# server-group tacacs+
device(config-system-aaa-server-group-tacacs+)# server 6.6.6.6
device(config-system-aaa-server-group-tacacs+-server-6.6.6.6)# secret-key sharedsecret
device(config-system-aaa-server-group-tacacs+-server-6.6.6.6)# source-interface ethernet 0/1.0
device(config-system-aaa-server-group-tacacs+-server-6.6.6.6)# vrf red-vrf
device(config-system-aaa-server-group-tacacs+-server-6.6.6.6)#
device#
```

The following example displays the TACACS+ server configuration that is running currently on the device:

```
device# show running-config system aaa server-group tacacs+

system
  aaa
    server-group tacacs+
      server 6.6.6.6
        secret-key-hashed SeHO6jSmdLQslwsXZKVHcifR/80uY9fcwSOEXU80w+KEmAxy1wb7FYiJYY/
6DZfxPUAlwYnkTOOfR07jWiDzdA==
      source-interface ethernet 0/1.0
      vrf red-vrf
    !
  !
!
```

**Tip**

The TACACS+ secret key displays as a hash in the **show running-config system aaa server-group tacacs+** command output. If you want to unhide the TACACS+ secret key so that it displays as plain text, see [unhide-secrets](#) on page 381.

ssh-server

Use the **ssh-server** command to configure an SSH server instance and enter SSH server system configuration (`config-system-ssh-server-name`) mode.

Syntax

ssh-server *server-name*

Command Default

An SSH server instance named DEFAULT is created. This instance is associated with a VRF named mgmt-vrf.

Parameters

server-name

Name of the SSH server. Use any combination of 1 to 64 alphanumeric characters, space, underscores, hyphens, or periods.

Modes

System configuration (`config-system`)

Usage Guidelines

Use this command to enter the mode for configuring SSH server settings. These settings include the server name, the cipher to use, the idle timeout for the SSH client connection, the host key pair for public key encryption, the key exchange algorithms to use, the message authentication code (MAC) algorithms to use, and the VRF to associate with the SSH server.

Examples

The following example enters SSH server system configuration (`config-system-ssh-server-name`) mode and configures an SSH server named h2 on the device:

```
device# configure terminal
device(config)# system
device(config-system)# ssh-server h2
device(config-system-ssh-server-h2)# cipher aes128-gcm-openssh aes256-gcm-openssh
chacha20-poly1305-openssh
device(config-system-ssh-server-h2)# client-alive 20
device(config-system-ssh-server-h2)# key rsa 2048
device(config-system-ssh-server-h2)# key-exchange ecdh-sha2-nistp256 curve25519-sha256
diffie-hellman-group-exchange-sha256
device(config-system-ssh-server-h2)# mac hmac-sha2-256 hmac-sha2-512
device(config-system-ssh-server-h2)# vrf vrf-red
device(config-system-ssh-server-h2)# enable
device(config-system-ssh-server-h2)#
```

The following example shows the SSH server configuration that is running currently on the device:

```
device# show running-config system ssh-server

system
  ssh-server h2
    cipher aes128-gcm-openssh aes256-gcm-openssh chacha20-poly1305-openssh
    mac hmac-sha2-256 hmac-sha2-512
    key-exchange ecdh-sha2-nistp256 curve25519-sha256
    client-alive 20
    key rsa 2048
    vrf vrf-red
    enable
  !
!
device#
```

ssh-server restart

Use the **ssh-server restart** command to restart the SSH server.

Syntax

```
ssh-server restart
```

Parameters

This command has no keywords or variables.

Modes

EXEC (#)

Usage Guidelines

After performing any configuration of the SSH server parameters, you must restart the SSH server to apply the cipher configuration.

Examples

The following example restarts the SSH server:

```
device# ssh-server restart
device#
```


ssl-profile-id (AAA LDAP server group system configuration)

Use the **ssl-profile-id** command to associate an SSL profile with an LDAP server instance for AAA.

Syntax

```
ssl-profile-id ssl-profile-name
```

Command Default

No SSL profile is associated with the current LDAP server instance.

Parameters

ssl-profile-name

Name of the SSL profile. Use 1 to 64 characters.

Modes

AAA LDAP server group server system configuration (config-system-aaa-server-group-ldap-server-*address*)

Usage Guidelines

This command specifies which SSL profile to associate with the LDAP server instance. This profile must contain the required trusted certificate authority (CA) certificate to validate the LDAP server certificate.

Before using this command, you must import a CA certificate into the SSL profile that you want to use. To do so, see [certificate manager import ca-certificate](#) on page 63.

There is no impact if the SSL profile is updated later. LDAPS authentication is invoked during SSH login and gNMI user authentication. It uses the latest certificate during such attempts.

To remove the SSL profile from the current LDAP server instance, use the **no** form of this command.

Examples

The following example associates an SSL profile named `sp1` with LDAP server instance 1.1.1.1 on the device:

```
device# configure terminal
device(config)# system
device(config-system)# aaa
device(config-system-aaa)# server-group ldap
device(config-system-aaa-server-group-ldap)# server 1.1.1.1
device(config-system-aaa-server-group-ldap-server-1.1.1.1)# base-dn example.com
```

```
device(config-system-aaa-server-group-ldap-server-1.1.1.1)# ldaps
device(config-system-aaa-server-group-ldap-server-1.1.1.1)# ssl-profile-id sp1
device(config-system-aaa-server-group-ldap-server-1.1.1.1)#
```

The following example displays the AAA configuration for an LDAP server group named ldap that is running currently on the device. In this example, the LDAP server group is configured with LDAP server instance 1.1.1.1, which is associated with an SSL profile named sp1:

```
device# show running-config system aaa server-group ldap

system
  aaa
    server-group ldap
      server 1.1.1.1
        base-dn example.com
        ldaps
        ssl-profile-id sp1
      !
    !
  !
!
device#
```

ssl-profile-id (AAA RADIUS server group system configuration)

Use the **ssl-profile-id** command to associate an SSL profile with a RADIUS server instance for AAA.

Syntax

ssl-profile-id *ssl-profile-name*

Command Default

No SSL profile is associated with the current RADIUS server instance.

Parameters

ssl-profile-name

Name of the SSL profile. Use 1 to 64 characters.

Modes

AAA RADIUS server group server system configuration (config-system-aaa-server-group-radius-server-*address*)

Usage Guidelines

This command specifies which SSL profile to associate with the RADIUS server instance. This profile must contain the required certificate authority (CA) certificate to validate the RADIUS server certificate.

Before using this command, you must import the CA certificate into the specified SSL profile. To do so, see [certificate manager import ca-certificate](#) on page 63.

There is no impact if the SSL profile is updated. RADSEC authentication is invoked during SSH login and gNMI user authentication. It uses the latest certificate during these attempts.

To remove the SSL profile from the current RADIUS server instance, use the **no** form of this command.

Examples

The following example associates an SSL profile named sp1 with RADIUS server instance 1.1.1.1 on the device:

```
device# configure terminal
device(config)# system
device(config-system)# aaa
device(config-system-aaa)# server-group radius
device(config-system-aaa-server-group-radius)# server 1.1.1.1
```

```
device(config-system-aaa-server-group-radius-server-1.1.1.1)# ssl-profile-id sp1
device(config-system-aaa-server-group-radius-server-1.1.1.1)#
```

The following example displays the AAA configuration for a RADIUS server group named radius that is running currently on the device. In this example, the RADIUS server group is configured with RADIUS server instance 1.1.1.1, which is associated with an SSL profile named sp1:

```
device# show running-config system aaa server-group radius

system
  aaa
    server-group radius
      server 1.1.1.1
        ssl-profile-id sp1
      !
    !
  !
!
device#
```

ssl-profile-id (AAA token validator system configuration)

Use the **ssl-profile-id** command to associate an SSL profile with a token validator instance for AAA.

Syntax

```
ssl-profile-id ssl-profile-name
```

Command Default

No SSL profile is associated with the current token validator instance.

Parameters

ssl-profile-name

Name of the SSL profile. Use 1 to 64 characters.

Modes

AAA token validator system configuration (*config-system-aaa-token-validator-name*)

Usage Guidelines

This command specifies which SSL profile to associate with the token validator instance.

A token validator instance is used to validate a JSON web token (JWT) that is included in gNMI or gNOI requests.

To remove the SSL profile from the current token validator instance, use the **no** form of this command.

Examples

The following example associates an SSL profile named `sp1` with a token validator instance named `val1` on the device:

```
device# configure terminal
device(config)# system
device(config-system)# aaa
device(config-system-aaa)# token-validator val1
device(config-system-aaa-token-validator-val1)# ssl-profile-id sp1
device(config-system-aaa-token-validator-val1)#
```

The following example displays the AAA configuration that is running currently on the device. In this example, AAA is configured with a token validator instance named `val1`, which is associated with an SSL profile named `sp1`:

```
device# show running-config system aaa token-validator

system
```

```
aaa
  token-validator val1
  ssl-profile-id sp1
!
!
!
device#
```

start-shell (Extreme ONE OS)

Use the **start-shell** command to start an interactive Bash shell session.

Syntax

```
start-shell
```

Parameters

This command has no keywords or variables.

Modes

EXEC (#)

Usage Guidelines

You can elevate permissions by using the **sudo** command.

Examples

The following example starts an interactive Bash shell session on the device and executes Linux commands within the session:

```
device# start-shell
[admin@device]# pwd
/home/admin
[admin@device]# id
uid=1000(admin) gid=1000(admin) groups=1000(admin)
[admin@device]#
```

system (Extreme ONE OS)

Use the **system** command to enter System configuration (config-system) mode.

Syntax

system

Parameters

This command has no keywords or variables.

Modes

System configuration (config-system)

Usage Guidelines

Use this command to enter the mode for configuring system-level settings such as AAA, the system clock, the hostname, the login banner, NTP, SNMP, SSH, and Telnet.

Examples

The following example enters System configuration (config-system) mode:

```
device# configure terminal
device(config)# system
device(config)#
```


system diagnostics run extended

Use the **system diagnostics run extended** command to run operational system diagnostics in the extended mode.

Syntax

```
system diagnostics run extended
```

Command Default

Parameters

This command has no keywords or variables.

Modes

EXEC (#)

Usage Guidelines

This command sets the correct parameters and then runs system diagnostics in the extended mode. The device reboots after completion of the diagnostics run.

If you need to upgrade the system diagnostics firmware before running system diagnostics, see [system diagnostics update](#) on page 349.

To run system diagnostics in the normal (non-extended) mode, see [system diagnostics run normal](#) on page 347.

Diagnostics can also be run directly from the GRand Unified Bootloader (GRUB) menu at boot time. In this case, an additional user authentication is required. The hardcoded user name and password are "root" and "abrcadabra" respectively.

To display the results that were saved during the diagnostics run, see [show system diagnostics](#) on page 284.

Examples

The following example runs system diagnostics in the extended mode on the device:

```
device# system diagnostics run extended

WARNING: system will be rebooted! Do you want to continue? [y/n]: y
Reloading.... please wait
device#
```

The following example shows the results of an extended system diagnostics run on the device:

```
device# show system diagnostics

Operational diagnostics version: 1.0
Date: Nov-14-2024 06:12:26
Version: 1.0
Summary: Diagnostics Fail
Mode: Extended

8730-32D: 801157-00-02 AD012350G-00031
PSU-1: Present 801162-00-00 P0142342A-C0034
PSU-2: Present 801162-00-00 P0142342A-C0044
FAN-1: Present 801166-00-02 F0032349G-00033
FAN-2: Present 801166-00-02 F0032349G-00440
FAN-3: Present 801166-00-02 F0032349G-00044
FAN-4: Present 801166-00-02 F0032349G-00393
FAN-5: Present 801166-00-02 F0032349G-00046
FAN-6: Present 801166-00-02 F0032349G-00031
FAN-7: Present 801166-00-02 F0032349G-00039

Test Result
=====
i2c environment PASS
fantray PASS
internal flash PASS
memory PASS
management interface FAIL
side_band interface FAIL
loopback pci PASS
loopback mac interface PASS
loopback phy fiber PASS
snake interface FAIL
asic0 reg PASS
asic0 mem PASS
device#
```

system diagnostics run normal

Use the **system diagnostics run normal** command to run operational system diagnostics in the normal mode.

Syntax

```
system diagnostics run normal
```

Command Default

Parameters

This command has no keywords or variables.

Modes

EXEC (#)

Usage Guidelines

This command sets the correct parameters and then runs system diagnostics in the normal (non-extended) mode. The device reboots after completion of the diagnostics run.

If you need to upgrade the system diagnostics firmware before running system diagnostics, see [system diagnostics update](#) on page 349.

To run system diagnostics in the extended mode, see [system diagnostics run extended](#) on page 345.

Diagnostics can also be run directly from the GRand Unified Bootloader (GRUB) menu at boot time. In this case, an additional user authentication is required. The hardcoded user name and password are "root" and "abrcadabra" respectively.

To display the results that were saved during the diagnostics run, see [show system diagnostics](#) on page 284.

Examples

The following example runs system diagnostics in the normal mode on the device:

```
device# system diagnostics run normal

WARNING: system will be rebooted! Do you want to continue? [y/n]: y
Reloading.... please wait
device#
```

The following example shows the results of a normal system diagnostic run on the device:

```
device# show system diagnostics

Operational diagnostics version: 1.1
Date: Mar-10-2025 06:09:22
Version: 1.1
Summary: Diagnostics Pass
Mode: Normal

8730-32D:                801157-00-02 AD012350G-00014
PSU-1: Present           801163-00-00 P0152342A-C0072
PSU-2: Present           801163-00-00 P0152342A-C0052
FAN-1: Present           801167-00-02 F0042349G-00133
FAN-2: Empty
FAN-3: Empty
FAN-4: Present           801167-00-02 F0042349G-00082
FAN-5: Present           801167-00-02 F0042349G-00093
FAN-6: Present           801167-00-02 F0042349G-00085
FAN-7: Present           801167-00-02 F0042349G-00204

Test                      Result
=====
i2c environment           PASS
fantray                   PASS
internal flash            PASS
memory                    PASS
management interface      PASS
side_band interface       PASS
loopback pci              PASS
loopback mac interface    PASS
loopback phy fiber        PASS
snake interface           PASS
asic0 reg                 PASS
asic0 mem                 SKIP
device#
```

system diagnostics update

Use the **system diagnostics update** command to update the operational diagnostics firmware on a device.

Syntax

```
system diagnostics update { disk-file-path | usb-file-path | remote-file-path | http-file-path }
```

Parameters

disk-file-path

Disk file path in format `disk://config-file/file-name`.

usb-file-path

USB file path in format `usb://file-name`.

scp-file-path

Remote server file path in format `scp[sftp]://username[:password]@host:[port]/filepath [vrf vrf-name]`.

A domain name, an IPv4 address, or an IPv6 address is supported as the host. Only valid unicast IP addresses are supported. Multicast IP addresses are not supported.

http-file-path

Remote server file path in format `http[https]://[username[:password]@host:[port]/filepath [vrf vrf-name]`.

A domain name, an IPv4 address, or an IPv6 address is supported as the host. Only valid unicast IP addresses are supported. Multicast IP addresses are not supported.

Modes

EXEC (#)

Usage Guidelines

This command applies only to the Extreme 8730 platform. This platform supports system diagnostics using a suite of tests contained in a preinstalled bootable image. System diagnostics comprises several commands to upgrade this image (if needed), then run the test suite, and then display the test results:

- (Optional) The **system diagnostics update** command (this command) upgrades the system diagnostics firmware. For details, see [system diagnostics update](#) on page 349.
- Either of the **system diagnostics run** commands will reboot the device with the diagnostics image and run the tests. Afterward, the results are stored in the

diagnostics partition of the device. The device then reboots with Extreme ONE OS. For details about using these commands, see [system diagnostics run normal](#) on page 347 or [system diagnostics run extended](#) on page 345.

- The **show system diagnostics** command displays the results saved during the diagnostics run. For details, see [show system diagnostics](#) on page 284.

System diagnostics can also be run directly from the GRand Unified Bootloader (GRUB) menu at boot time.

Examples

The following example updates the diagnostics firmware of the system on the device:

```
device# system diagnostics update disk://firmware/summit-aoede.1.2
Operational diagnostics firmware updated
device#
```

system firmware commit (Extreme ONE OS)

Use the **system firmware commit** command to commit the version of an image that is running currently on the device after an update.

Syntax

```
system firmware commit [ app-name ]
```

Command Default

The version of an image that is running currently after an update remains uncommitted until you perform a commit using this command. There is no auto-commit after a firmware update.

Parameters

app-name

(Optional) Name of an application to commit. For example, enter SR to commit the Extreme ONE OS Switching application. If you do not specify an application name, Extreme ONE OS (the base OS) is committed.

Modes

EXEC (#)

Usage Guidelines

A firmware update leaves two firmware images on the device: the previous firmware (as the rollback version) and the new firmware (as the current version that is running on the device). If you are satisfied with the new image, you should execute this command to commit the image.



Note

You cannot roll back to the prior version after committing a new version.

You cannot commit a previously committed version. You do not need to recommit the prior version of an image if you roll back to that version.

Make sure that Extreme ONE OS (the base OS) is committed before you update the firmware for an application such as Extreme ONE OS Switching. You can update the firmware for multiple applications, one after another, without committing each one after its update.

Examples

The following example commits the base Extreme ONE OS firmware on the device:

```
device# system firmware commit
Firmware Commit is in progress, do not reboot or powercycle box
| Committing ... 2024-10-18 16:50:09.4378 mgmtsvc-cli[1236514]: Level:info LogID:8007
Msg:Commit current Base firmware image Audit:Firmware
- Committing ...
Commit state:COMMIT_SUCCESS
device#
```

The following example commits the Extreme ONE OS Switching application firmware on the device:

```
device# system firmware commit SR
Firmware Commit is in progress, do not reboot or powercycle box
| Committing ... 2024-10-18 16:50:30.0184 mgmtsvc-cli[1236514]: Level:info LogID:8007
Msg:Commit current app SR image Audit:Firmware
\ Committing ...
Commit state:COMMIT_SUCCESS
device#
```


system firmware fullinstall (Extreme ONE OS)

Use the **system firmware fullinstall** command to re-install the firmware on a device.

Syntax

```
system firmware fullinstall URL [ no-preserve ]
```

Command Default

No firmware full re-installation has been performed on the device.

Parameters

URL

URL that points to the firmware image .bin file. Specify the URL in one of the following formats:

- **usb://filename**
- **disk://firmware/filename**
- **scp://username:password@host** [*:port*] */filepath*
- **sftp://username:password@host** [*:port*] */filepath*
- **http://** [*username:password@*] *host* [*:port*] */filepath*
- **https://** [*username:password@*] *host* [*:port*] */filepath*

host usage requires that DNS resolution is configured on the device.

[**no-preserve**]

(Optional) Removes the management certificates and keys during the firmware installation.

Modes

EXEC (#)

Usage Guidelines

This command initiates new firmware installation from from the CLI and restores the necessary certificate files (unless you specify the no-preserve keyword) and the SSH configuration file.

This command mimics the **onie-nos-install** command to install the new firmware image .bin file. The **onie-nos-install** command is used in ONIE to install a Network Operating System (NOS) onto a bare metal network device.

This command deletes all partitions from the device and preserves the management certificate files by default.

After the system reboots following a firmware full installation, if you are satisfied with the new image, you should commit the image. For details, see [system firmware commit \(Extreme ONE OS\)](#) on page 351.

Examples

The following example performs a full installation of the Extreme ONE OS Switching application on the device:

```
device# system firmware fullinstall http://<image path>:8081/artifactory/one-os-local-
snapshots/SR/22.2.1.0/build/SR-22.2.1.0/install/SR-22.2.1.0.bin
Preparing Device for Fullinstall...This will take some time

Activation completed successfully in 1m33s
Firmware update complete. Rebooting switch to activate
Reloading.... please wait
device#
```

system firmware rollback (Extreme ONE OS)

Use the **system firmware rollback** command to roll back the version of an image that is running currently on the device after an update.

Syntax

```
system firmware rollback [ app-name ]
```

Command Default

If a version that is running on the device after an update is yet to be committed, the device continues to use that version unless rolled back. (You cannot roll back a version that is already committed.)

Parameters

app-name

(Optional) Name of an application to roll back. For example, enter SR to roll back the Extreme ONE OS Switching application. If you do not specify an application name, Extreme ONE OS (the base OS) is rolled back.

Modes

EXEC (#)

Usage Guidelines

A firmware update leaves two firmware images on the device: the previous firmware (as the rollback version) and the new firmware (as the current version that is running on the device). After a firmware update, if you are unsatisfied with the new image, you should execute this command to roll back the image to the prior version. To determine the prior version, see [show version \(Extreme ONE OS\)](#) on page 321.



Note

You cannot roll back to the prior version after committing the new version.

You do not need to recommit the prior version of an image after you roll back to that version.

Examples

The following example rolls back the base and application Extreme ONE OS firmware on the device:

```
device# system firmware rollback
Firmware Rollback is in progress...
Rollback completed successfully in 10s
```

```
Firmware rollback complete. Rebooting switch to activate
device#
```

The following example rolls back the Extreme ONE OS Switching application firmware on the device

```
device# system firmware rollback SR
Performing App Rollback...
2024-11-19 08:12:14.9940 mftm[15]: Level:info LogID:4008 Msg:Starting Shutdown of
Microservice, received signal:terminated
2024-11-19 08:12:14.9942 mftm[15]: Level:info LogID:4009 Msg:Completed shutting down all
worker threads, exiting
2024-11-19 08:12:15.0893 bfd[30]: Level:info LogID:230001 Topic:1 Msg:msShutdownHandler:
BFD MS Shutdown Ctx:bfd
2024-11-19 08:12:15.0894 bfd[30]: Level:info LogID:4008 Msg:Starting Shutdown of
Microservice, received signal:terminated
2024-11-19 08:12:15.0896 bfd[30]: Level:info LogID:4009 Msg:Completed shutting down all
worker threads, exiting
2024-11-19 08:12:15.0896 bfd[30]: Level:info LogID:23001 Topic:1 Msg:main: BFD MS is
going down
[ 2266.724310] _iqpkt_sock_release: Warmboot case!
2024-11-19 08:12:15.1964 lldp[30]: Level:fatal LogID:14003 Msg:LLDP protocol service
shutting down lldp
2024-11-19 08:12:15.1972 lldp[30]: Level:info LogID:4008 Msg:Starting Shutdown of
Microservice, received signal:terminated
2024-11-19 08:12:15.1992 lldp[30]: Level:info LogID:4006 Msg:[lldp-ms]: Worker Shutdown
started
2024-11-19 08:12:15.4341 sr-api-gw[16]: Level:info LogID:4008 Msg:Starting Shutdown of
Microservice, received signal:terminated
2024-11-19 08:12:15.4342 sr-api-gw[16]: Level:info LogID:4009 Msg:Completed shutting down
all worker threads, exiting
2024-11-19 08:12:15.4589 lacp[15]: Level:info LogID:4008 Msg:Starting Shutdown of
Microservice, received signal:terminated
2024-11-19 08:12:15.4590 lacp[15]: Level:info LogID:4006 Msg:[lacp-ms]: Worker Shutdown
started
2024-11-19 08:12:15.5181 classifiers[30]: Level:info LogID:4008 Msg:Starting Shutdown of
Microservice, received signal:terminated
2024-11-19 08:12:15.5182 classifiers[30]: Level:info LogID:4009 Msg:Completed shutting
down all worker threads, exiting
2024-11-19 08:12:16.2006 lldp[30]: Level:info LogID:4007 Msg:[lldp-ms]: Worker Shutdown
done
2024-11-19 08:12:16.2007 lldp[30]: Level:info LogID:4006 Msg:[lldp-stats]: Worker
Shutdown started
2024-11-19 08:12:16.2008 lldp[30]: Level:info LogID:4007 Msg:[lldp-stats]: Worker
Shutdown done
2024-11-19 08:12:16.2008 lldp[30]: Level:info LogID:4009 Msg:Completed shutting down all
worker threads, exiting
2024-11-19 08:12:16.2008 lldp[30]: Level:fatal LogID:14001 Msg:LLDP protocol service is
shutting down
2024-11-19 08:12:16.2037 lacp[15]: Level:info LogID:4007 Msg:[lacp-ms]: Worker Shutdown
done
2024-11-19 08:12:16.2311 lacp[15]: Level:info LogID:4006 Msg:[lacp-stats]: Worker
Shutdown started
2024-11-19 08:12:16.2320 lacp[15]: Level:info LogID:4007 Msg:[lacp-stats]: Worker
Shutdown done
2024-11-19 08:12:16.2322 lacp[15]: Level:info LogID:4009 Msg:Completed shutting down all
worker threads, exiting

App SR Uninstallation completed. Please re-launch the CLI session.
device#
```

system firmware uninstall (Extreme ONE OS)

Use the **system firmware uninstall** command to uninstall application firmware from the device.

Syntax

```
system firmware uninstall app-name
```

Parameters

app-name

Name of an application to uninstall. Enter SR to uninstall the Extreme ONE OS Switching application.

Modes

EXEC (#)

Usage Guidelines

This command does not uninstall Extreme ONE OS (the base OS) or kernel patches.

This command does not perform a rollback. If you want to roll back the current version of an application that is running on the device to the previous firmware (the rollback version), see [system firmware rollback \(Extreme ONE OS\)](#) on page 355.

Examples

The following example uninstalls the Extreme ONE OS Switching application from the device:

```
device# system firmware uninstall SR
Uninstalling App...
Uninstall for SR : UNINSTALL_SUCCESS
device#
```

system firmware update (Extreme ONE OS)

Use the **system firmware update** command to update firmware on the device.

Syntax

```
system firmware update URL [ vrf vrf-name ] [ force ] [ default-config ]
```

Command Default

No firmware is updated (there are no automatic updates).

Parameters

URL

URL that points to the firmware image file. Use one of the following values:

- **usb:***//filename*
- **disk:***//firmware/filename*
- **scp:***//username:password@host [:port] /filepath [vrf vrf-name]*
- **sftp:***//username:password@host [:port] /filepath [vrf vrf-name]*
- **http:***// [username:password@] host [:port] /filepath [vrf vrf-name]*
- **https:***// [username:password@] host [:port] /filepath [vrf vrf-name]*

vrf *vrf-name* is a VRF instance name (1 to 64 characters).



Note

USB flash drives are compatible with FAT, EXT, and NTFS file systems.

host usage requires that DNS resolution is configured on the device.

vrf *vrf-name*

(Optional) VRF instance name (1 to 64 characters).

force

(Optional) Forces the firmware upgrade operation.

default-config

(Optional) Loads the factory default configuration and discards the current running configuration on reboot after the upgrade.

Modes

EXEC (#)

Usage Guidelines

You can perform three types of firmware update:

- Full update of Extreme ONE OS (the base OS). This update uses a .bin image file. This update reboots the system if there are no errors.
- Extreme ONE OS application update (such as Extreme ONE OS Switching). This update use a .app image file. This update does not reboot the system.
- Kernel patch update. This update uses a .incr image file (which is an archive of a .ko (kernel patch) file). This update does not reboot the system.

All firmware image files use the following versioning format: [Product ID].[Major].[Minor].[Patch].bin. For example, 22.2.1.0.bin.

After the installation, the system generates a log message indicating that installation is complete.

A firmware update leaves two firmware images on the device: the previous firmware (as the rollback version) and the new firmware (as the current version that is running on the device). After the system reboots following a firmware update, if you are satisfied with the new image, you should commit the image. For details, see [system firmware commit \(Extreme ONE OS\)](#) on page 351.

If you are not satisfied with the new version (or if the new firmware does not start properly), you can roll back to the prior version. For details, see [system firmware rollback \(Extreme ONE OS\)](#) on page 355.

To update firmware of the programmable components on the device, see [system programmable update \(Extreme ONE OS\)](#) on page 362.

Examples

The following example updates the Extreme ONE OS Switching application on the device:

```
device# system firmware update disk://firmware/ExtremeOneSR-22.2.1.0.bin
Activating Firmware...This will take some time
App SR Uninstallation completed.
Activation completed successfully in 1m38s
Firmware update complete. Rebooting switch to activate
Reloading.... please wait
device#
```

system internal tdpa

Use the **system internal tdpa** command to initiate a stub that downloads the Trusted Delivery Provisioning Application (TDPA) and provisions the HWROT to set up the device for Measured Boot.

Syntax

```
system internal tdpa { dry-run | provision | unit-test | show-hwrot }  
    server ip-address [ { port port-number | verbose | version | no-  
    remove } ]
```

Parameters

dry-run

Dry run provisioning while keeping the HWROT (Hardware Root of Trust) unlocked.

provision

Provision the device for field use. Rollback is not possible and might require a BIOS change.

unit-test

Unit test for the TDPA application. For technical support use only.

show-hwrot

Show HWROT certificates and counters.

ip-address

IPv4 or IPv6 address of the TDP server where the actual TDPA resides.

port-number

TDP server port.

verbose

Enables verbose logs.

version

Displays the TDPA version.

no-remove

(Dry runs only) Keeps certificates.

Modes

EXEC (#)

Usage Guidelines

This command cannot be executed once the TPM chip is provisioned. This command will return an error message.

Examples

The following example performs a dry run to download the TDPA and provisions the HWROT to set up the device for Measured Boot:

```
device# system internal tdpa dry-run server 134.141.77.75

Tdpa Program....
ip: 134.141.77.75 port:  tdpaCmd:  dry-run Verbose:  false
Namespace(action='dry-run', version=False, verbose=False,
server_ip_address='134.141.77.75', server_port=443, do_not_lock_asp=False,
do_not_remove_dry_run_certs=False)
PATH: /mnt/onl/config/tdpa/logs
FILE: /mnt/onl/config/tdpa/logs/tdpa_dry-run.log
Logging to file /mnt/onl/config/tdpa/logs/tdpa_dry-run.log
=====
Started Trusted Delivery Provisioning App version 1.8.1.0 pid 48781
Start Time: 2024-10-29 06:56:48
=====
Processing cli arguments
Python Version: 3.10.12
Action: dry-run
Lock ASP with Password: False
Set 'asp_locked' Flag (read by UBoot): False
Verbose Logging: Disabled
TDPS Address: 134.141.77.75:443
TDPS Certificate Store: DEVELOPMENT
Boot Version verification not applicable
Create Components
Initialize Components
TDPS 'https://134.141.77.75' is alive; app_version 21.2.23; database OK; internet OK
.
.
.
.
=====
Finished Trusted Delivery Provisioning App version 1.8.1.0 pid 48781
Finish Time: 2024-10-29 06:57:47
Duration: 58.46 seconds
TDPA Action:dry-run, Status:SUCCESS
=====
Log file /mnt/onl/config/tdpa/logs/tdpa_dry-run.log (3519 bytes) has been uploaded to
https://134.141.77.75/api/logs
device#
```

system programmable update (Extreme ONE OS)

Use the **system programmable update** command to update firmware of the programmable components on the device.

Syntax

```
system programmable [ component ] update URL
```

Parameters

component

(Optional) Name of a programmable component. Specify bios, bmc, cpu-cpld, micro-controller, port-cpld, power-cpld, or um-switch.

URL

URL that points to the firmware image file. Use one of the following values:

- **usb://filename**
- **disk://firmware/filename**
- **scp://username:password@host [:port] /filepath**
- **sftp://username:password@host [:port] /filepath**
- **http:// [username:password@] host [:port] /filepath**
- **https:// [username:password@] host [:port] /filepath**

host usage requires that DNS resolution is configured on the device.

Modes

EXEC (#)

Usage Guidelines

The firmware images have different extensions depending on the intended component.

After the installation, the system generates a log message indicating that installation is complete.

Examples

The following example updates the baseboard management controller (BMC) firmware on the device:

```
device# system programmable bmc update scp://exampleuser:examplepassword@198.51.100.0:22/home/henry12/bmc-fw/XXXX-32c/ix7f_cec_signed_v3.06-3907480_Production.ima_enc
WARNING: power must not be interrupted and bmc will be inaccessible during this time! Do
you want to continue? [y/n]: y
%Info: Firmware file downloaded and validated for bmc
```

```
%Info: Flashing Firmware on bmc...This will take some time  
%Info: Flashing firmware on bmc completed in 3m47s  
device#
```

system tech-support

Use the **system tech-support** command to generate a compressed TAR file of the data needed for assistance in monitoring and troubleshooting the device.

Syntax

```
system tech-support [ remote-file-path ] [ source-ip address | vrf vrf-name ]
```

Parameters

remote-file-path

(Optional) Remote server file path in format `scp://username:password@host[:port]/filepath`.

If a path is provided, the generated archive is copied securely to the specified remote server using Secure Copy Protocol (SCP). If no path is specified, the archive is saved locally on the device in the `/var/data/disk/techsupport` directory.

source-ip *address*

(Optional) Source IPv4 or IPv6 address. Only unicast addresses are supported.

- For an IPv4 address, use IPv4 notation (*A.B.C.D*). For example, 10.0.0.0.
- For an IPv6 address, use the *x:x::x:x* format. For example, 1001::1.

vrf *vrf-name*

(Optional) VRF instance.

Modes

EXEC (#)

Usage Guidelines

Use this command to collect the output of various **show** commands, log files, and configuration files that you can provide as a system snapshot when requesting help with debugging the device. Use this command only in conjunction with Extreme Networks Technical Support. The command creates two files: a compress TAR file (such as TS-duta-250206T0299.tar.gz) and a similarly named log file (such as TS-duta-250206T0299.log).

If you want to see a list of the existing tech support files, enter the **show tech-support** command.

If you want to clear all tech support files or a specific tech support file, see [clear tech-support](#) on page 70.

Examples

The following example displays a list of the existing tech support files on the device:

```
device# show tech-support
File Name                               Size (bytes)  Created
TS-dutb-251028T160410.log              8037         2025-10-28 16:04:54
TS-dutb-251028T160410.tar.gz          12298874     2025-10-28 16:04:54
device#
```

The following example generates a compressed TAR file named TS-48xt-250429T073900.tar.gz and a log file named TS-48xt-250429T073900.log containing the data to send to Extreme Networks Technical Support for monitoring and troubleshooting. These files are saved locally at /var/data/disk/techsupport:

```
device# system tech-support
Generating show commands output
Executing base-svc host commands
Executing commands for service chassis-mgr
Executing commands for service state-db
Executing commands for service interface-mgr
Archiving /etc/extreme
Archiving /var/log/messages
Archiving /var/log/messages.1
Archiving /var/log/messages.1.gz
Archiving /var/log/messages.2
Archiving /var/log/messages.2.gz
Archiving /var/log/messages.3
Archiving /var/log/messages.3.gz
Archiving /var/log/messages.4
Archiving /var/log/messages.4.gz
Archiving /var/log/auth.log
Archiving /var/log/debug
Archiving /var/log/debug.1
Archiving /var/log/debug.1.gz
Archiving /var/log/debug.2
Archiving /var/log/debug.2.gz
Archiving /var/log/debug.3
Archiving /var/log/debug.3.gz
Archiving /var/log/debug.4
Archiving /var/log/debug.4.gz
Archiving /var/log/containers
Archiving /var/log/tierra
Archiving /var/log/log_file_journalctl
Archiving /var/log/daemon.log
Archiving /var/log/bmclog.gz
Archiving /var/log/consolelog
Archiving /tmp/tierra
Archiving /mnt/onl/usrdata/coredumps
Archiving /mnt/onl/usrdata/oom
Archiving /mnt/onl/usrdata/pprof
Archiving /run/log/journal
Archiving /var/data/disk/techsupport/ShowOutput.txt
Archiving /var/data/disk/techsupport/.prevJournalLog
Archiving /var/ms-commands
Archiving /mnt/onl/config/auditlog
Archiving /mnt/onl/config/db
Archiving /mnt/onl/config/allHwTables.txt
Archiving /var/data/monitors/k3s_events
Archiving /var/data/monitors/pods_resources_usage.log.gz
Archiving /var/data/monitors/system_resources_usage.log.gz
Archiving /etc/snmp
Archiving /var/log/snmpd.log
Archiving /var/log/ethphyd.bootlog
```

```

Archiving /etc/passwd
Archiving /etc/shadow
Archiving /etc/group
Archiving /etc/ssh
Archiving /etc/tierra
Archiving /etc/ldap.conf
Archiving /etc/ldap.map
Archiving /etc/tacacs.conf
Archiving /etc/radius.conf
Archiving /etc/pam.d
Archiving /var/data/cert-mgmt/certs
Archiving /var/data/cert-mgmt/app-cert
Archiving /var/data/cert-mgmt/ca-trust
Archiving /lib/systemd/system/xinetd*.service
Archiving /lib/systemd/system/sshd*.service
Archiving /var/crash
Archiving /var/log/base-svc-stdoutlogs
Archiving /var/data/ztp
Archiving /var/log/db-migrator.log
Cleaning up temporary files
TechSupport completed and file TS-48xt-250429T073900.tar.gz generated

Done
device#

```

The following example generates a compressed TAR file named TS-32d-250620T084936.tar.gz and a log file named TS-32d-250620T084936.log. These files are copied to the /home/user/techsupport/TS/ directory on remote host 1.2.3.4:

```

device# system tech-support scp://exampleuser:examplepassword@1.2.3.4/home/user/
techsupport/TS/
Generating show commands output
Executing base-svc host commands
Executing commands for service chassis-mgr
Executing commands for service state-db
Executing commands for service interface-mgr
Archiving /etc/extreme
Archiving /var/log/messages
Archiving /var/log/messages.1
Archiving /var/log/messages.1.gz
Archiving /var/log/messages.2
Archiving /var/log/messages.2.gz
Archiving /var/log/messages.3
Archiving /var/log/messages.3.gz
Archiving /var/log/messages.4
Archiving /var/log/messages.4.gz
Archiving /var/log/auth.log
Archiving /var/log/debug
Archiving /var/log/debug.1
Archiving /var/log/debug.1.gz
Archiving /var/log/debug.2
Archiving /var/log/debug.2.gz
Archiving /var/log/debug.3
Archiving /var/log/debug.3.gz
Archiving /var/log/debug.4
Archiving /var/log/debug.4.gz
Archiving /var/log/containers
Archiving /var/log/tierra
Archiving /var/log/log_file_journalctl
Archiving /var/log/daemon.log
Archiving /var/log/bmclog.gz
Archiving /var/log/consolelog
Archiving /tmp/tierra
Archiving /mnt/onl/usrdata/coredumps

```

```

Archiving /mnt/onl/usrdata/oom
Archiving /mnt/onl/usrdata/pprof
Archiving /run/log/journal
Archiving /var/data/disk/techsupport/ShowOutput.txt
Archiving /var/data/disk/techsupport/.prevJournalLog
Archiving /var/ms-commands
Archiving /mnt/onl/config/auditlog
Archiving /mnt/onl/config/db
Archiving /mnt/onl/config/allHwTables.txt
Archiving /var/data/monitors/k3s_events
Archiving /var/data/monitors/pods_resources_usage.log.gz
Archiving /var/data/monitors/system_resources_usage.log.gz
Archiving /etc/snmp
Archiving /var/log/snmpd.log
Archiving /var/log/ethphyd.bootlog
Archiving /etc/passwd
Archiving /etc/shadow
Archiving /etc/group
Archiving /etc/ssh
Archiving /etc/tierra
Archiving /etc/ldap.conf
Archiving /etc/ldap.map
Archiving /etc/tacacs.conf
Archiving /etc/radius.conf
Archiving /etc/pam.d
Archiving /var/data/cert-mgmt/certs
Archiving /var/data/cert-mgmt/app-cert
Archiving /var/data/cert-mgmt/ca-trust
Archiving /lib/systemd/system/xinetd*.service
Archiving /lib/systemd/system/sshd*.service
Archiving /var/crash
Archiving /var/log/base-svc-stdoutlogs
Archiving /var/data/ztp
Archiving /var/log/db-migrator.log
Cleaning up temporary files
TechSupport completed and file TS-32d-250620T084936.tar.gz generated

Done
Copying the techsupport files to remote server...
The Techsupport is successfully copied to the remote server
device#

```

The following example displays a list of the existing tech support files on the device:

```

device# ls disk://tech-support
tech-support :
-rw-----  8037      2025-10-28 15:25:39  TS-48xt-250429T073900.log
-rw-r--r-- 11981288 2025-10-28 15:25:39  TS-48xt-250429T073900.tar.gz
device#

```

telnet-server

Use the **telnet-server** command specify a Telnet server instance name and enter Telnet server system configuration (config-system-telnet-server-*name*) mode.

Syntax

```
telnet-server [ server-name ]
```

Command Default

No Telnet server instance is enabled on the device.

Parameters

server-name

(Optional) Name of the Telnet server instance to configure. Use 1 to 64 characters. If you do not specify a server name, DEFAULT is used as the name.

Modes

System configuration (config-system)

Usage Guidelines

Use this command to enter the mode for configuring Telnet server settings. These settings include whether to enable the Telnet server instance and the VRF to associate with it.

Examples

The following example enters Telnet server system configuration (config-system-telnet-server-*name*) mode and configures a Telnet server instance named telnet1:

```
device# configure terminal
device(config)# system
device(config-system)# telnet-server telnet1
device(config-system-telnet-server-telnet1)# vrf red-vrf
device(config-system-telnet-server-telnet1)# enable
device(config-system-telnet-server-telnet1)#
```

The following example displays the Telnet server configuration that is running currently on the device. In this example, a Telnet server instance named telnet1 is enabled:

```
device# show running-config system telnet-server

system
  telnet-server telnet1
    vrf red-vrf
    enable
!
```



```
!  
device#
```

terminal length (Extreme ONE OS)

Use the **terminal length** command to set the number of lines that the terminal displays on the device.

Syntax

```
terminal length lines
```

Command Default

The screen displays 24 lines of text.

Parameters

lines

Lines per page. Use an integer from 0 (no pagination) to 512.

Modes

EXEC (#)

Usage Guidelines

This command sets the number of lines to be displayed on the screen. Setting the terminal length to 0 removes page breaks for the outputs of all **show** commands so that the display of command output is uninterrupted.

This value applies only to the current terminal where the command is executed. The value is not preserved across reboots.

Examples

The following example configures the length of the terminal to 20 lines on the device:

```
device# terminal length 20
device#
```

terminal timeout

Use the **terminal timeout** command to set the amount of time before the terminal times out after no activity.

Syntax

terminal timeout *seconds*

Command Default

Each session times out after 600 seconds (10 minutes).

Parameters

seconds

Seconds before the session times out. Use an integer from 0 (infinite timeout) to 8192.

Modes

EXEC (#)

Usage Guidelines

This value applies only to the current terminal where the command is executed. The value is not preserved across reboots.

Examples

The following example configures the timeout of the terminal to 300 seconds (five minutes) on the device:

```
device# terminal timeout 300
device#
```

tls

Use the **tls** command to enter TLS system configuration (config-system-tls) mode.

Syntax

tls

Command Default

Global Transport Layer Security (TLS) settings are not configured on the device.

Parameters

This command has no keywords or variables.

Modes

System configuration (config-system)

Usage Guidelines

This command enters the mode required for configuring the TLS settings on a device. They include settings for the global TLS minimum version (applicable to all client/service applications that use TLS) and the TLS minimum versions for specific services that use TLS.

This command does not appear in the running configuration unless there are CLI attributes already configured.

Examples

The following example enters TLS system configuration (config-system-tls) mode:

```
device# configure terminal
device(config)# system
device(config-system)# tls
device(config-system-tls)#
```

tls-profile-id

Use the **tls-profile-id** command to specify an SSL profile for TLS encryption for securely forwarding syslogs to a remote syslog server.

Syntax

```
tls-profile-id tls-profile-name  
no tls-profile-id
```

Command Default

No SSL profile is specified for Transport Layer Security (TLS) encryption for forwarding syslogs to the current remote syslog server instance.

Parameters

tls-profile-name

Name of the TLS profile to use. Use 1 to 64 characters.

Modes

Logging remote server system configuration (config-system-logging-remote-server-*addr-or-domain*)

Usage Guidelines

A syslog client instance must be associated with an SSL profile through the **command** attribute. This profile must contain the certificate authority (CA) certificate to validate the syslog server certificate.

Before using this command, you must copy a CA certificate to the specified SSL profile. To do so, see [certificate manager import ca-certificate](#) on page 63.

Use the **no** form of this command to restore the default behavior.

Examples

The following example configures logging on a device. In this example, an SSL profile named `sp1` is specified for TLS encryption to securely forward syslogs to remote syslog server 1.1.1.1:

```
device# configure terminal  
device(config)# system  
device(config-system)# logging  
device(config-system-logging)# remote-server 1.1.1.1  
device(config-system-logging-remote-server-1.1.1.1)# tls-profile-id sp1  
Warning: Configuration exists  
device(config-system-logging-remote-server-1.1.1.1)# secure-forwarding tls  
Warning: Existing Host configuration changed
```

```

device(config-system-logging-remote-server-1.1.1.1)# mode-transport tcp
Warning: Existing Host configuration changed
device(config-system-logging-remote-server-1.1.1.1)# remote-port 525
Warning: Existing Host configuration changed

```

The following example configures logging on a device. In this example, an SSL profile named `sp1` is specified for TLS encryption to securely forward syslogs to remote syslog server 1.1.1.1:

```

device# show running-config system logging remote-server

system
 logging
  remote-server 1.1.1.1
  secure-forwarding tls
  mode-transport tcp
  remote-port 525
  tls-profile-id sp1
!
!
!

```

The following example restores the logging configuration to the default value (disabled) on a device. In this example, TLS encryption for secure forwarding of syslogs to remote syslog server 1.1.1.1 is disabled:

```

device(config-system-logging-remote-server-1.1.1.1)# no secure-forwarding tls
Warning: Existing Host configuration changed
device(config-system-logging-remote-server-1.1.1.1)# do show system logging remote-server
HOST          PORT  TRANSPORT  SECURE-FORWARDING  VRF-NAME
=====
1.1.1.1        525   TCP        NONE                NONE
device(config-system-logging-remote-server-1.1.1.1)#

```

token-validator

Use the **token-validator** command to specify a token validator instance for AAA and then enter AAA token validator system configuration (`config-system-aaa-token-validator-name`) mode.

Syntax

token-validator *validator-instance-name*

Command Default

The default token validator instance (named DEFAULT) is used.

Parameters

validator-instance-name

Name of the token validator instance. Use 1 to 64 characters. If you do not specify a name, the default instance (named DEFAULT) is used.

Modes

AAA system configuration (`config-system-aaa`)

Usage Guidelines

This command enters the mode required for configuring the token validator settings on a device. They include the the subcommand for specifying the SSL profile with which you want to associate the token validator instance.

A token validator instance is used to validate a JSON web token (JWT) that is included in gNMI or gNOI requests. Token validation supports external users who are not authenticated on the device (either through local PAM or remote authentication via LDAP or RADIUS).

You can configure only one token validator instance at a time.

To restore this setting to the default value, use the **no** form of this command.

Examples

The following example enters AAA token validator system configuration (`config-system-aaa-token-validator-name`) mode and specifies `vall` as the token validator instance on the device:

```
device# configure terminal
device(config)# system
device(config-system)# aaa
device(config-system-aaa)# token-validator vall
device(config-system-aaa-token-validator-vall)#
```

The following example displays the AAA configuration that is running currently on the device. In this example, AAA is configured with a token validator instance named vall:

```
device# show running-config system aaa

system
  aaa
    authentication
      admin-user admin password-hashed
      $6$l0FtgLVtzOEi7jhl$NALdarg9FfowUfWTZUaOnyXG4mehd3cdg3jdGWBFE6uCCSFShPxstb6Mpcd3UJoA16Up19
      6GZ1pn5ilGURdte.
      !
      token-validator vall
    !
  !
device#
```


tolerance

Use the **tolerance** command to extend the validity of the old expired key to ensure smooth key rollover.

Syntax

```
tolerance { seconds | forever }  
no tolerance
```

The tolerance for key expiration is set to zero (no tolerance).

Parameters

seconds

Tolerance time in seconds. The valid range is 0-600.

forever

Sets the tolerance time to infinite so that the old key never expires.

Modes

Keychain configuration (config-*keychain-name*)

Usage Guidelines

This command extends the validity of the old expired key for a period of time so that the receive processing can ensure smooth key rollover.



Note

A longer time period can induce security risks because of an exposed old key.

You can decrease the new key activation time so that packets received can be authenticated using a key that is about to be activated.

Use the no form of this command to decrease the new key activation time so that the packet received can be authenticated using a key which is yet to be activated.

Examples

The following example configures the tolerance values for two keychains:

```
device# configure terminal  
device(config)# keychain keychain1  
device(config-keychain-keychain1)# tolerance 400  
device(config-keychain-keychain1)# exit  
device(config)# keychain keychain2  
device(config-keychain-keychain2)# tolerance forever  
device(config-keychain-keychain2)#
```

The following example displays the keychain configuration that is running currently on the device. In this example, two keychains are configured with a tolerance value:

```
device# show running-config keychain

show running-config keychain
keychain keychain1
    tolerance 400
!
keychain keychain2
    tolerance forever
!
device#
```

udp-port

Use the **udp-port** command to specify the UDP port number for sending traps to the trap host for an SNMP server.

Syntax

udp-port *port-number*

no udp-port

Command Default

UDP port number 162 is used.

Parameters

port-number

UDP port number to use. Use an integer from 1 to 65535.

Modes

SNMP server VRF host system configuration (config-system-snmp-server-vrf-*vrf-name*-host-*host-name-community-or-user-name*)

Usage Guidelines

Use this command to configure the UDP port on which the receiver is listening for SNMP traps.

To restore the UDP port number to the default value for the specified server, use the **no** form of this command.

Examples

The following example configures a Simple Network Management Protocol (SNMP) agent with the trap destination information with the combination of a host and a username attached to it. In this example, the trap host 192.168.1.1 uses UDP port 200:

```
device# configure terminal
device(config)# system
device(config-system)# snmp-server DEFAULT
device(config-system-snmp-server)# user user3 auth noauth
device(config-system-snmp-server)# vrf mgmt-vrf
device(config-system-snmp-server-vrf-mgmt-vrf)# enable
device(config-system-snmp-server-vrf-mgmt-vrf)# host 192.168.1.1 public
device(config-system-snmp-server-vrf-mgmt-vrf-host-192.168.1.1-xxxx)# version 3
device(config-system-snmp-server-vrf-mgmt-vrf-host-192.168.1.1-public)# udp-port 200
device(config-system-snmp-server-vrf-mgmt-vrf-host-192.168.1.1-public)
```

The following example displays the SNMP server configuration that is running currently on the device:

```
device# show running-config system snmp-server

system
  snmp-server DEFAULT
  user user3 auth noauth
  vrf mgmt-vrf
  enable
  host 192.168.1.1 public
  version 3
  udp-port 200
  !
!
!
device#
```

unhide-secrets

Use the **unhide-secrets** command to set the CLI system attribute for displaying secrets in plain text when you view the running configuration or running state of a device.

Syntax

```
unhide-secrets all  
no unhide-secrets
```

Command Default

Secrets appear in hashed form when you view the running configuration or running state of a device.

Parameters

all

Displays secrets for all modules (RADIUS secret keys, TACACS+ secret keys, and secrets for SNMP communities).

Modes

CLI system configuration (config-system-cli)

Usage Guidelines

Use this command to configure the CLI attribute to unhide RADIUS secret keys, TACACS+ secret keys, and secrets for SNMP communities in **running-config** or **running-state** command output.

Use the **no** form of this command to restore the default behavior.

Examples

The following example shows the default behavior for displaying secrets, which converts them to hashed form. In this example, the SNMP server community name appears as a hash:

```
device# show running-config system snmp-server  
  
system  
  snmp-server DEFAULT  
    community-hashed jERww/4GrM  
  vrf mgmt-vrf  
    enable  
  !  
!
```

```
!  
device#
```

The following example configures secrets so that they are displayed in plain text in **running-config** or **running-state** command output for all modules:

```
device# configure terminal  
device(config)# system  
device(config-system)# cli  
device(config-system-cli)# unhide-secrets all  
device(config-system-cli)#
```

The following example shows the SNMP server community name after the device is configured to unhide secrets. In this example, the name appears in plain text form:

```
device# show running-config system snmp-server  
  
system  
  snmp-server DEFAULT  
    community extremero  
  vrf mgmt-vrf  
    enable  
  host 192.168.1.1 user3  
    version 3  
    udp-port 200  
    source-interface ethernet 0/1.0  
  !  
  !  
  !  
  !  
device#
```

unlock username

Use the **unlock username** command to unlock a user account that locked because of account inactivity.

Syntax

unlock username *username*

Command Default

Locked users remain locked.

Parameters

username *username*

Name of the locked user. Use 1 to 32 characters.

Modes

EXEC (#)

Usage Guidelines

You can use the **show users** to display the list of locked users.

Examples

The following example unlocks a locked user account:

```
device# unlock username bsmith
Unlocking the user account is successful.
device#
```

usb enable (Extreme ONE OS)

Use the **usb enable** command to disable or enable USB storage on a device.

Syntax

usb enable

no usb enable

Command Default

USB storage is disabled on the device.

Parameters

This command has no keywords or variables.

Modes

EXEC (#)

Usage Guidelines

To disable USB storage on the device, use the **no** form of this command.

To verify that USB storage is enabled on the device, see [show usb \(Extreme ONE OS\)](#) on page 319.

Examples

The following example enables USB storage on the device:

```
device# usb enable
device#
```

The following example displays whether USB storage is enabled on the device:

```
device# show usb

USB Enabled:  true
device#
```


user (Extreme ONE OS SNMP server system configuration)

Use the **user** command to configure the SNMP server users to be used for SNMPv3 protocol access.

Syntax

```
user username auth { noauth | md5 | sha } { auth-password | auth-password-hashed } auth-key-password priv { nopriv | aes | des } { priv-password | priv-password-hashed } priv-password-phrase  
  
no user username
```

Command Default

The device has no SNMP server users configured to be used for SNMPv3 protocol access.

Parameters

user *username*

Simple Network Management Protocol (SNMP) v3 username.

auth { **noauth** | **md5** | **sha** }

Authentication method to use.

auth-password *auth-password-phrase*

Password for authentication. Use 8 to 42 alphanumeric characters.

priv { **nopriv** | **aes** | **des** }

Encryption method to use.

priv-password *priv-password-phrase*

Password for encryption. Use 8 to 42 alphanumeric characters.

Modes

SNMP server system configuration (config-system-snmp-server)

Usage Guidelines

All the users configured will have READ all permission. No user will ever have WRITE permission.

Use the **no** form of this command to remove an SNMP server user.



Note

If you want to use SNMPv1 and SNMPv2c, you must use the [community \(SNMP server system configuration\)](#) on page 80 command (instead of the **user** command) to configure authentication. SNMPv1 and SNMPv2c use a community string, which is similar to a user ID or password that allows access to device statistics.

Usage Guidelines

The **show running-config** and **show running-config snmp-server** commands display the values for auth-password and priv-password in hashed format.

Examples

The following example configures three SNMPv3 server users:

```
device# configure terminal
device(config)# system
device(config-system)# snmp-server DEFAULT
device(config-system-snmp-server)# user v3user2 auth md5 auth-password Testing@123 priv
aes priv-password Testing@123
device(config-system-snmp-server)# user v3user3 auth noauth
device(config-system-snmp-server)# user v3user8 auth md5 auth-password Testing@123 priv
aes priv-password Testing@123
device(config-system-snmp-server)# vrf mgmt-vrf
device(config-system-snmp-server-vrf-mgmt-vrf)# enable
device(config-system-snmp-server-vrf-mgmt-vrf)# host 192.168.1.1 v3user3
device(config-system-snmp-server-vrf-mgmt-vrf-host-192.168.1.1-xxxx)# version 3
device(config-system-snmp-server-vrf-mgmt-vrf-host-192.168.1.1-v3user3)# udp-port 200
device(config-system-snmp-server-vrf-mgmt-vrf-host-192.168.1.1-v3user3)# source-interface
ethernet 0/1
device(config-system-snmp-server-vrf-mgmt-vrf-host-192.168.1.1-v3user3)
```

The following example shows the SNMPV3 user name after the device is configured to unhide secrets. In this example, the passwords appears in plain text form::

```
device# show running-config system snmp-server

system
  snmp-server DEFAULT
    user v3user2 auth md5 auth-password Testing@123 priv aes priv-password Testing@123
    user v3user3 auth noauth
    user v3user8 auth md5 auth-password Testing@123 priv aes priv-password Testing@123
  vrf red-vrf
    enable
    host 192.168.1.1 v3user3
    version 3
    udp-port 200
    source-interface ethernet 0/1
  !
!
!
device#
```

user (AAA authentication system configuration)

Use the **user** command to configure a user account.

Syntax

```
user username role { admin | user } { password | password-hashed }  
    password [ inactivity-expiry-period days ] [ inactivity-warning-  
    period days ]  
  
no user username
```

Command Default

There are no other default users (other than the "admin" user account, which is provisioned by default and is not removable).

Parameters

user *username*

Username string. Use 1 to 32 characters. Do not use the same name as the default "admin" user.

role { **admin** | **user** }

User role.

{ **password** | **password-hashed** } *password*

Password string. Use 8 to 128 characters. The string must also satisfy the password attributes that are configured in the system. To view these attributes, enter the **show running-config system** command. To adjust these attributes, see [password-attributes](#) on page 177.

inactivity-expiry-period *days*

(Optional) Number of days that the account is allowed to be inactive. Use 0 to 180 days. The default is 0.

After the **inactivity-expiry-period**, the user account is locked, and a syslog is generated.

Setting the **inactivity-expiry-period** to 0 turns off the inactive account disable mechanism.

inactivity-warning-period *days*

(Optional) Number of days after which a warning is issued before disabling the account. Use 1 to 120 days. The default is 0.

After the **inactivity-warning-period**, a warning syslog is generated.

Modes

Global configuration (config)

Usage Guidelines

This command configures a user, assigns a role, and sets other properties for a given user account but not the default "admin" account.

This account cannot be deleted from the system (only the password can be changed using the **admin-user** command). All other users with the role "admin" have the same rights as the default "admin" user (they can also change other "admin" role account passwords), but they are removable.

You can enable alerts and account lockout functionality for the user account by adding the **inactivity-expiry-period** and **inactivity-warning-period** keywords.

You must set the inactivity expiry period before setting the inactivity warning period.

The inactivity warning period must be less than the inactivity expiry period.

To reset the expiry period and the warning period, use the full **user** command and provide the default values.

Use the **no** form of this command to delete the user configuration.

Examples

The following example configures AAA on the device. This example configures two user accounts (test1 and test2):

```
device# configure terminal
device(config)# system
device(config-system)# aaa
device(config-system-aaa)# authentication
device(config-system-aaa-authentication)# user user1 role user password #Pass@456!
inactivity-expiry-period 20 inactivity-warning-period 15
device(config-system-aaa-authentication)# user user2 role user password $Pass@789!
inactivity-expiry-period 20 inactivity-warning-period 15
device(config-system-aaa-authentication)#
```

The following example displays the AAA configuration that is running currently on the device. In this example, two user accounts (test1 and test2) are configured:

```
device# show running-config system aaa authentication

system
aaa
authentication
    admin-user admin password-hashed
    $6$SBA6UhDM8hddYLA8$xAM4PDdtzHkQoj8JCHf7jMKyo8Hjiqjiuxfd5dkN41QuO5TsLnyZZV3LUI2gsZkcoTmA1j
    Uf7TMie7JEVIWom/
    user user1 role user password-hashed
    $6$106681641$1i7apm7M7e5tMakmLvqgzEHIAbRv7nKsF2OnDzjeIE3xLkZfrsnI6IuGmsIB/IqWFe5CTknqC/
    hQTPOEUJ0lu0 inactivity-expiry-period 20 inactivity-warning-period 15
    user user2 role user password-hashed $6$981264507$r1Om7KNzvqrc/
    JKpGmWc99TnFJ7xj84b0nt403swzOElnTKZMks86Jbl8JilUwSCTrAZ2o8PwZxJoL3bjpgG80 inactivity-
    expiry-period 15 inactivity-warning-period 10
    !
    !
    !
```

```
!  
device#
```

username (GRUB system configuration)

Use the **username** command to change the GRUB password on a device.

Syntax

```
username username { password | password-hashed } password
```

Parameters

username

Username in plain text. Use 1 to 32 alphanumeric, underscore, or period characters. The username must begin with an alphanumeric or underscore character.

{ **password** | **password-hashed** } *password*

Plain text password or password hash.

- A plain text password must satisfy the password strength requirements determined via the **password-attributes** command. For details, see [password-attributes](#) on page 177.
- A password hash must be a grub-mkpasswd-pbkdf2 generated password hash of length 282 with a prefix of grub.pbkdf2.sha512.

Modes

GRUB system configuration (config-system-grub)

Usage Guidelines

Use this command to configure the GRand Unified Bootloader (GRUB) credentials on a device. This username and password provide protection for the GRUB bootloader menu when the device boots. Any inputs (attempt to boot from other boot entries, edit a boot entry, or enter the GRUB command line interface) other than the default boot option in the GRUB menu will prompt for authentication.

GRUB is a powerful and widely-used bootloader in many UNIX-like operating systems. A bootloader is the first program that runs when a device boots. It loads the operating system into memory and starts it. Its main role is to manage and boot multiple operating systems from a single device.



Note

The initial username for the GRUB user interface itself is "root". GRUB prompts for a new password during the first login.

Examples

The following example configures the GRUB credentials on the device:

```
device# configure terminal
device(config)# system
device(config-system)# grub
device(config-system-grub)# username root password Rockstar!456
device(config-system-grub)#
```

The following example displays the GRUB configuration that is running currently on the device:

```
device # show running-config system grub

system
  grub
    username root password-hashed
grub.pbkdf2.sha512.10000.99290C16E44A4613BFAF94D724F081CE970C7AD32D0B08FB38C0AB8CE91BBF4DD
C7482B50E59B9D45BF0A5E9269F50CFDA03F45E59B42A78FE5EB35D589CFFBF.F2DFFFBF15622F841F31EBDF35
5B2CC6DA6820A244E44A54498C220E6C1881DBB63D1484994D6E1DCE7EC5691C6FC6337F49A4DF812CED2CAE0D
599D7E2A6E71
!
!
device#
```

version (SNMP server VRF host system configuration)

Use the **version** command to specify the SNMP version for sending traps to the trap host for an SNMP server.

Syntax

```
version snmp-version  
no version
```

Command Default

Simple Network Management Protocol (SNMP) version 2c is used.

Parameters

snmp-version
SNMP version. Specify 1, 2c, or 3. The default is 2c.

Modes

SNMP server VRF host system configuration (config-system-snmp-server-vrf-*vrf-name*-host-*host-name-community-or-user-name*)

Usage Guidelines

To restore the SNMP version to the default value for the specified server, use the **no** form of this command.

Examples

The following example configures an SNMP agent with the trap destination information with the combination of a host and a username attached to it. In this example, the trap host 192.168.1.1 uses SNMPv3:

```
device# configure terminal  
device(config)# system  
device(config-system)# snmp-server DEFAULT  
device(config-system-snmp-server)# user user3 auth noauth  
device(config-system-snmp-server)# vrf mgmt-vrf  
device(config-system-snmp-server-vrf-mgmt-vrf)# enable  
device(config-system-snmp-server-vrf-mgmt-vrf)# host 192.168.1.1 public  
device(config-system-snmp-server-vrf-mgmt-vrf-host-192.168.1.1-xxxx)# version 3  
device(config-system-snmp-server-vrf-mgmt-vrf-host-192.168.1.1-public)# udp-port 200  
device(config-system-snmp-server-vrf-mgmt-vrf-host-192.168.1.1-public)# source-interface  
ethernet 0/1  
device(config-system-snmp-server-vrf-mgmt-vrf-host-192.168.1.1-public)
```


The following example displays the SNMP server configuration that is running currently on the device:

```
device# show running-config system snmp-server

system
  snmp-server DEFAULT
  user user3 auth noauth
  vrf mgmt-vrf
  enable
  host 192.168.1.1 public
  version 3
  udp-port 200
  source-interface ethernet 0/1
  !
!
!
device#
```

vrf (GRPC server configuration)

Use the **vrf** command to create the VRF instance for a gRPC server instance.

Syntax

vrf *vrf-name*

no vrf

Command Default

The gRPC server instance on the device uses the management VRF (which is named mgmt-vrf).

Parameters

vrf-name

VRF instance name. Use a string from 1 to 64 characters.

Modes

gRPC server system configuration (config-system-grpc-server-*name*)

Usage Guidelines

You must disable the gRPC server instance before using this command (or any other command) to alter its configuration. Then, you must re-enable the instance to restore its functionality. To disable or enable a gRPC server instance, see [enable \(gRPC server configuration\)](#) on page 103.

To restore this setting to the default value, use the **no** form of this command.

Examples

The following example configures a gRPC server instance. In this example, a VRF instance named vrf-blue is created for the gRPC server instance:

```
device# configure terminal

device(config-system)# grpc-server mygrpcserver
device(config-system-grpc-server-mygrpcserver)# certificate-id test_cert_id
device(config-system-grpc-server-mygrpcserver)# port 9340
device(config-system-grpc-server-mygrpcserver)# mutual-tls
device(config-system-grpc-server-mygrpcserver)# vrf vrf-blue
device(config-system-grpc-server-mygrpcserver)# enable
device(config-system-grpc-server-mygrpcserver)#
```

The following example displays the gRPC server instance configuration that is running currently on the device. In this example, the instance uses a VRF instance named vrf-blue:

```
device# show running-config system grpc-server

system
  grpc-server mygrpcserver
  certificate-id test_cert_id
  port 9340
  mutual-tls
  vrf vrf-blue
  enable
!
device#
```

vrf (logging remote server system configuration)

Use the **vrf** command to configure the VRF instance for the remote syslog server for logging on a device.

Syntax

```
vrf vrf-name  
no vrf
```

Command Default

The remote syslog server uses the management VRF (which is named mgmt-vrf).

Parameters

vrf-name

VRF instance name. Use a string from 1 to 64 characters.

Modes

Logging remote server system configuration (config-system-logging-remote-server-*addr-or-domain*)

Usage Guidelines

Use this command to configure a VRF name to route syslog communications.

To restore this setting to the default value, use the **no** form of this command.

Examples

The following example configures logging on a device. In this example, a VRF instance named red-vrf is configured for the remote syslog server:

```
device# configure terminal  
device(config)# system  
device(config-system)# logging  
device(config-system-logging)# remote-server 1.1.1.1  
device(config-system-logging-remote-server-1.1.1.1)# mode-transport tcp  
device(config-system-logging-remote-server-1.1.1.1)# remote-port 525  
device(config-system-logging-remote-server-1.1.1.1)# secure-forwarding tls  
device(config-system-logging-remote-server-1.1.1.1)# vrf red-vrf  
device(config-system-logging-remote-server-1.1.1.1)#
```

The following example displays the remote syslog server configuration that is running currently on the device. In this example, the server uses a VRF instance named red-vrf:

```
device# show running-config system logging remote-server  
  
system
```

```
logging
  remote-server 1.1.1.1
  mode-transport tcp
  remote-port 525
  secure-forwarding tls
  vrf red-vrf
!
!
device#
```

vrf (NTP system configuration)

Use the **vrf** command to specify a VRF for routing Network Time Protocol (NTP) communications.

Syntax

```
vrf vrf-name  
no vrf
```

Command Default

No VRF is specified.

Parameters

vrf-name

Name of the VRF instance. This name must start with an alphabetic character or an underscore followed by an arbitrary sequence of alphabetic or numeric characters, underscores, hyphens, or dots. The name cannot exceed 64 characters.

Modes

NTP system configuration (config-system-ntp)

Usage Guidelines

You use NTP to synchronize the clocks over your network. The NTP service is a global network of public and private servers that provides precise time synchronization to devices worldwide. It works by exchanging time data packets among client and server devices.

Examples

The following example configures NTP to synchronize the system date and time on the device. In this example, a VRF named red-vrf is used to route NTP communications, IPv4 address 2.2.2.2 points to an NTP server to acquire all system time information, and IPv4 address 3.3.3.3 is a peer with which the local device synchronizes its system time:

```
device# configure terminal  
device(config)# system  
device(config-system)# ntp  
device(config-system-ntp)# enable  
device(config-system-ntp)# vrf red-vrf  
device(config-system-ntp)# server 2.2.2.2  
device(config-system-ntp)# peer 3.3.3.3  
device(config-system-ntp)#
```

The following example displays the NTP configuration that is running currently on the device:

```
device# show running-config system ntp

system
  ntp
    enable
    vrf red-vrf
    server 2.2.2.2
    peer 3.3.3.3
  !
!
device#
```

The following example displays the NTP configuration that is running currently on the device. This is an example of a more complex configuration that uses multiple servers and multiple peers:

```
device# show running-config system ntp

system
  ntp
    enable
    vrf red-vrf
    server 0.in.pool.ntp.org
    server 1.in.pool.ntp.org
    server 2.in.pool.ntp.org
    server 1.pool.ntp.org
    server 3.in.pool.ntp.org
    server 2.pool.ntp.org
    server 2.2.2.2
    peer 10.in.pool.ntp.org
    peer 11.in.pool.ntp.org
    peer 12.in.pool.ntp.org
    peer 21.pool.ntp.org
    peer 23.in.pool.ntp.org
    peer 22.pool.ntp.org
    peer 3.3.3.3
  !
!
device#
```

vrf (Extreme ONE OS)

Use the **vrf** command to associate a VRF instance to an LDAP, RADIUS, SNMP, SSH, TACACS+, or Telnet server.

Syntax

vrf *vrf-name*

no vrf *vrf-name*

Command Default

The LDAP, RADIUS, SNMP, SSH, TACACS+, and Telnet servers have no associated VRF instances.

Parameters

vrf-name

VRF instance name. Use 1 to 64 characters.

Modes

AAA LDAP server group server system configuration (config-system-aaa-server-group-ldap-server-*address*)

AAA RADIUS server group server system configuration (config-system-aaa-server-group-radius-server-*address*)

SNMP server system configuration (config-system-snmp-server)

SSH server system configuration (config-system-ssh-server-*name*)

AAA TACACS+ server group server system configuration (config-system-aaa-server-group-tacacs+-server-*address*)

Telnet server system configuration (config-system-telnet-server-*name*)

Usage Guidelines

SSH runs on all VRFs by default. Telnet, TACACS+, Radius, RADSEC, LDAP, and LDAPS need to be assigned to a VRF.

A maximum of six VRF instances is supported.

To remove a VRF instance from the specified server, use the **no** form of this command.

Examples

The following example associates a VRF instance named red-vrf with the Lightweight Directory Access Protocol (LDAP) server host 7.7.7.20:

```
device# configure terminal
device(config)# system
device(config-system)# aaa
device(config-system-aaa)# server-group ldap
device(config-system-aaa-server-group-ldap)# server 7.7.7.20
device(config-system-aaa-server-group-ldap-server-7.7.7.20)# vrf red-vrf
device(config-system-aaa-server-group-ldap-server-7.7.7.20)# do show running-config
system aaa server-group ldap
system
  aaa
    server-group ldap
      server 7.7.7.20
        vrf red-vrf
      !
    !
  !
!
device#
```

The following example associates a VRF instance named red-vrf with the Remote Authentication Dial-In User Service (RADIUS) server host 4.4.4.4:

```
device# configure terminal
device(config)# system
device(config-system)# aaa
device(config-system-aaa)# server-group radius
device(config-system-aaa-server-group-radius)# server 4.4.4.4
device(config-system-aaa-server-group-radius-server-4.4.4.4)# vrf red-vrf
device(config-system-aaa-server-group-radius-server-4.4.4.4)# do show running-config
system aaa server-group radius
system
  aaa
    server-group radius
      server 4.4.4.4
        vrf red-vrf
      !
    !
  !
!
device#
```

The following example associates a VRF instance named red-vrf with a Simple Network Management Protocol (SNMP) server named DEFAULT:

```
device# configure terminal
device(config)# system
device(config-system)# snmp-server
device(config-system-snmp-server)# vrf red-vrf
device(config-system-snmp-server-vrf-red-vrf)# do show running-config system snmp-server
system
  snmp-server DEFAULT
    vrf red-vrf
  !
!
device#
```

The following example associates a VRF instance named red-vrf with a Secure Shell (SSH) server named h2:

```
device# configure terminal
device(config)# system
device(config-system)# ssh-server h2
device(config-system-ssh-server-h2)# vrf vrf-red
device(config-system-ssh-server-h2)#
```

The following example associates a VRF instance named red-vrf with the Terminal Access Controller Access-Control System Plus (TACACS+) server host 6.6.6.6:

```
device# configure terminal
device(config)# system
device(config-system)# aaa
device(config-system-aaa)# server-group tacacs+
device(config-system-aaa-server-group-tacacs+)# server 6.6.6.6
device(config-system-aaa-server-group-tacacs+-server-6.6.6.6)# vrf red-vrf
device(config-system-aaa-server-group-tacacs+-server-6.6.6.6)# do show running-config
system aaa server-group tacacs+
system
  aaa
    server-group tacacs+
      server 6.6.6.6
        vrf red-vrf
    !
  !
!
device#
```

The following example associates a VRF instance named red-vrf with a Telnet server named telnet1:

```
device# configure terminal
device(config)# system
device(config-system)# telnet-server telnet1
device(config-system-telnet-server-telnet1)# vrf red-vrf
device(config-system-telnet-server-telnet1)# do show running-config system telnet-server
system
  telnet-server telnet1
    vrf red-vrf
  !
!
device#
```

ztp dhcp cancel

Use the **ztp dhcp cancel** command to cancel the ZTP over DHCP process on a device.

Syntax

```
ztp dhcp cancel
```

Command Default

ZTP over DHCP is enabled on the device.

Parameters

This command has no keywords or variables.

Modes

EXEC (#)

Usage Guidelines

Use this command to cancel the zero touch provisioning (ZTP) process if the device cannot start ZTP over DHCP because of misconfiguration or a network issue. Note that this command has no effect if ZTP over DHCP is currently upgrading the firmware or applying configurations from a remote server.

During the ZTP over DHCP process, configuration commands are not allowed until you use this command.

ZTP over DHCP is an automated way to provision and configure large numbers of network devices without manual intervention. It uses the DHCP protocol to deliver initial configuration or bootstrap instructions when a device first powers on.

ZTP can also be used to update existing system configurations via scripts.

When you connect a new Extreme device to your network, it loads the appropriate firmware and configurations automatically without manual intervention.

To view the current status and server details associated with ZTP over DHCP, see [show ztp dhcp status](#) on page 325. To view the log files from the ZTP over DHCP process, see [show ztp dhcp logs](#) on page 323.

Examples

The following example cancels the ZTP over DHCP process on the device:

```
device# ztp dhcp cancel
Successfully canceled ZTP Process
device#
```

The following example displays the current status and server details for ZTP over DHCP on the device. In this example, the ZTP over DHCP process was canceled:

```
device# show ztp dhcp status

ZTP Status: Canceled
Client-id: EXTREMENETWORKS##8730-32D##AD012451G-00007
ZTP Interface: ma1
Interface IP: 10.32.129.191
Http Server IP Address: 10.64.13.105
Transfer Mode: HTTP
ZTP Configuration File: config/files/ztp.conf
Switch Configuration File: startup.cfg, /config/files/script_device.py
Firmware Image: /config/files/ExtremeOneSR-22.2.1.0.bin
Firmware Download State: FirmwareInstallSkipped
device#
```



Extreme ONE OS Switching Commands

[access-list](#) on page 415
[access-list \(control plane configuration\)](#) on page 417
[access-list \(interface configuration\)](#) on page 419
[actions \(routing policy map statement configuration\)](#) on page 422
[activate \(BGP address family configuration\)](#) on page 424
[activate \(BGP peer group address family configuration\)](#) on page 426
[activate-network](#) on page 428
[add-paths receive](#) on page 430
[address-family \(BGP configuration\)](#) on page 433
[address-family \(BGP peer group configuration\)](#) on page 435
[address-type \(MLAG peer configuration\)](#) on page 437
[advertisement \(IPv6 router advertisement prefix configuration\)](#) on page 439
[agetime \(ARP\)](#) on page 441
[agetime \(neighbor discovery\)](#) on page 442
[allow-own-as](#) on page 443
[arp-proxy](#) on page 445
[arp-snooping](#) on page 446
[as-notation](#) on page 447
[as-path-set](#) on page 449
[auth-password](#) on page 451
[auto-offset](#) on page 453
[auto-boot](#) on page 455
[autoconfiguration \(IPv6 router advertisement prefix configuration\)](#) on page 457
[auto-negotiate \(Extreme ONE OS Switching\)](#) on page 459
[bfd](#) on page 462
[bgp-actions \(routing policy statement actions configuration\)](#) on page 463
[bgp-conditions \(routing policy statement conditions configuration\)](#) on page 465
[bgp-defined-sets](#) on page 467
[bridge-domain](#) on page 469
[bringup-delay \(MLAG peer configuration\)](#) on page 470
[bus](#) on page 471
[channel-group](#) on page 473
[clear bgp neighbor](#) on page 475
[clear bgp peer-group](#) on page 477

[clear bgp vrf routes l2vpn-evpn nvo](#) on page 479
[clear bgp vrf routes l2vpn-evpn nvo route-distinguisher](#) on page 481
[clear bgp vrf routes l2vpn-evpn nvo route-distinguisher route-type](#) on page 483
[clear bgp vrf routes l2vpn-evpn nvo route-type](#) on page 485
[clear bgp vrf routes l2vpn-evpn nvo route-type arp vlan ipv4-prefix mac](#)
on page 487
[clear bgp vrf routes l2vpn-evpn nvo route-type imr vlan originator](#) on page 489
[clear bgp vrf routes l2vpn-evpn nvo route-type mac vlan mac](#) on page 491
[clear bgp vrf routes l2vpn-evpn nvo route-type nd vlan ipv6-prefix mac](#)
on page 493
[clear bgp vrf routes l2vpn-evpn nvo route-type prefix vlan ipv4-prefix](#)
on page 495
[clear bgp vrf routes l2vpn-evpn nvo route-type prefix vlan ipv6-prefix](#)
on page 497
[clear bgp vrf routes mac-vrf bridge-domain](#) on page 499
[clear bgp vrf routes mac-vrf bridge-domain route-type arp](#) on page 500
[clear bgp vrf routes mac-vrf bridge-domain route-type imr](#) on page 502
[clear bgp vrf routes mac-vrf bridge-domain route-type mac](#) on page 504
[clear bgp vrf routes mac-vrf bridge-domain route-type nd](#) on page 506
[clear counters access-list](#) on page 508
[clear counters bfd](#) on page 510
[clear counters bridge-domain](#) on page 512
[clear counters interface ethernet](#) on page 513
[clear counters interface port-channel](#) on page 514
[clear counters interface tunnel](#) on page 515
[clear counters lacp](#) on page 516
[clear counters lldp](#) on page 517
[clear counters mlag](#) on page 519
[clear counters qos interface control-plane queue](#) on page 520
[clear counters qos interface ethernet queue](#) on page 521
[clear ipv4 neighbor interface \(Extreme ONE OS Switching\)](#) on page 523
[clear ipv4 neighbor vrf](#) on page 525
[clear ipv6 neighbor interface](#) on page 528
[clear ipv6 neighbor vrf](#) on page 530
[clear l2 mac-move-detection shut-list](#) on page 533
[clear ipv4 route vrf \(IP routing\)](#) on page 535
[clear ipv6 route vrf \(IP routing\)](#) on page 536
[clear mac-address-table bridge-domain](#) on page 537
[cluster-id](#) on page 539
[community-set](#) on page 541
[conditions \(routing policy map statement configuration\)](#) on page 543
[connect-retry](#) on page 545
[cos traffic-class](#) on page 547

[cpus](#) on page 549

[decapsulation \(QoS NVO configuration\)](#) on page 551

[default-route-originate \(BGP peer group address family configuration\)](#) on page 553

[default-traffic-class](#) on page 555

[deny \(IPv4 ACL configuration\)](#) on page 556

[deny \(IPv6 ACL configuration\)](#) on page 561

[deny \(MAC ACL configuration\)](#) on page 566

[deploy \(IAH VM configuration\)](#) on page 569

[description \(BGP peer group configuration\)](#) on page 571

[description \(bridge domain configuration\)](#) on page 573

[description \(Ethernet interface configuration\)](#) on page 574

[description \(IAH VM configuration\)](#) on page 575

[description \(mirror configuration\)](#) on page 577

[destination interface \(mirror configuration\)](#) on page 578

[destination \(MLAG peer configuration\)](#) on page 580

[destination \(tunnel interface configuration\)](#) on page 582

[disk \(IAH VM configuration\)](#) on page 584

[dns \(IAH VM configuration\)](#) on page 586

[dscp traffic-class](#) on page 588

[duplex \(Ethernet interface configuration\)](#) on page 590

[ebgp-multihop](#) on page 592

[ecmp-max-path \(global system configuration\)](#) on page 595

[ecmp-max-path \(VRF instance configuration\)](#) on page 596

[enable \(IPv6 router advertisement configuration\)](#) on page 598

[enable-bfd](#) on page 600

[enable-on-reload](#) on page 603

[encapsulation \(QoS NVO configuration\)](#) on page 605

[evpn-instance \(VRF configuration\)](#) on page 607

[export-policy \(BGP peer group address family configuration\)](#) on page 608

[ext-community-set](#) on page 610

[extend-bd \(MLAG configuration\)](#) on page 612

[fast-external-failover](#) on page 615

[fec \(Ethernet interface configuration\)](#) on page 617

[file \(IAH VM disk configuration\)](#) on page 620

[format \(IAH VM disk configuration\)](#) on page 622

[graceful-restart \(BGP address family configuration\)](#) on page 624

[graceful-restart \(BGP configuration\)](#) on page 626

[graceful-restart \(BGP peer group address family configuration\)](#) on page 628

[graceful-restart \(BGP peer group configuration\)](#) on page 630

[hello-timer](#) on page 633

[helper-only \(BGP graceful restart configuration\)](#) on page 634

[hold-time \(BGP peer group timer configuration\)](#) on page 636

[holdtime \(LLDP configuration\)](#) on page 638

[hop-limit](#) on page 639

[hostname \(IAH VM configuration\)](#) on page 641

[iah fileinfo](#) on page 643

[iah \(global configuration\)](#) on page 645

[iah vm](#) on page 646

[iah vm console](#) on page 648

[import-policy \(BGP peer group address family configuration\)](#) on page 649

[iah vm import](#) on page 651

[iah vm upgrade](#) on page 654

[input \(interface QoS configuration\)](#) on page 656

[instance-type](#) on page 657

[interface ethernet](#) on page 659

[interface \(IAH VM network configuration\)](#) on page 660

[interface management ipv4 \(IAH VM configuration\)](#) on page 662

[interface management ipv6 \(IAH VM configuration\)](#) on page 664

[interface \(MLAG configuration\)](#) on page 666

[interface port-channel \(LAG configuration\)](#) on page 668

[interface \(QoS configuration\)](#) on page 669

[interface tunnel \(Extreme ONE OS Switching\)](#) on page 671

[interval \(BFD profile configuration\)](#) on page 673

[interval \(IPv6 router advertisement configuration\)](#) on page 675

[ip mtu \(Extreme ONE OS Switching\)](#) on page 677

[ip mtu \(Extreme ONE OS Switching interface configuration\)](#) on page 679

[ipv4 address anycast-gateway \(Virtual Ethernet interface configuration\)](#) on page 681

[ipv4 address dhcp \(Extreme ONE OS Switching\)](#) on page 683

[ipv4 address \(interface configuration\)](#) on page 685

[ipv4 anycast-gateway-mac](#) on page 687

[ipv4 neighbor \(interface configuration\)](#) on page 688

[ipv4 neighbor reachable-time \(interface configuration\)](#) on page 690

[ipv6 address anycast-gateway \(Virtual Ethernet interface configuration\)](#) on page 692

[ipv6 address auto-config \(Extreme ONE OS Switching\)](#) on page 694

[ipv6 address dhcp \(Extreme ONE OS Switching\)](#) on page 695

[ipv6 address \(interface configuration\)](#) on page 697

[ipv6 anycast-gateway-mac](#) on page 699

[ipv6 neighbor \(interface configuration\)](#) on page 700

[ipv6 neighbor reachable-time \(interface configuration\)](#) on page 702

[ipv6-router-advertisement](#) on page 704

[keepalive-interval](#) on page 706

[keepalive \(MLAG configuration\)](#) on page 708

[keepalive \(MLAG peer configuration\)](#) on page 710

[l3vni \(VRF instance type configuration\)](#) on page 712

[l3vni \(VRF instance type NVO configuration\)](#) on page 714

[lacp min-links](#) on page 716

[lacp port-priority](#) on page 718

[lacp interval](#) on page 720

[lacp system-mac](#) on page 722

[lacp system-priority](#) on page 724

[ldap cacert](#) on page 726

[ldap host](#) on page 728

[lifetime](#) on page 730

[link-error-disable](#) on page 732

[link-fault-signaling](#) on page 735

[listen-range \(BGP peer group configuration\)](#) on page 738

[lldp](#) on page 740

[load-balance \(Extreme ONE OS Switching global configuration\)](#) on page 741

[local-as-forced](#) on page 743

[local-as Need to put something in parentheses to differentiate from other 'local-as' commands - what platforms support this command???](#) on page 745

[local-pref-eq \(routing policy statement bgp conditions configuration\)](#) on page 747

[loopback-mode](#) on page 749

[l2 mac-move-detection](#) on page 750

[mac-address-table aging-time](#) on page 752

[mac \(MLAG configuration\)](#) on page 754

[maintenance-mode \(system configuration\)](#) on page 755

[managed-config-flag](#) on page 757

[management-address](#) on page 759

[match-as-path-set \(routing policy statement bgp conditions configuration\)](#) on page 761

[match-community-set \(routing policy statement bgp conditions configuration\)](#) on page 763

[match-ext-community-set \(routing policy statement bgp conditions configuration\)](#) on page 765

[match-prefix-set \(routing policy map statement conditions configuration\)](#) on page 767

[match-set-options \(routing policy BGP community set configuration\)](#) on page 769

[match-set-options \(routing policy BGP extended community set configuration\)](#) on page 771

[med-eq \(routing policy statement bgp conditions configuration\)](#) on page 773

[member](#) on page 775

[member nve](#) on page 777

[member \(routing policy BGP AS path set configuration\)](#) on page 779

[member \(routing policy BGP community set configuration\)](#) on page 781

[member \(routing policy BGP extended community set configuration\)](#) on page 783

[member \(BFD configuration\)](#) on page 785

[member \(single tagged\)](#) on page 787

[member \(tagged\)](#) on page 789

[member \(untagged\)](#) on page 791

[member vni-domain](#) on page 793

[member \(VRF configuration\)](#) on page 795

[mirror \(global configuration\)](#) on page 797

[mirror \(interface configuration\)](#) on page 799

[mlag](#) on page 801

[mode disable-unsolicited-ra](#) on page 802

[mtu \(Extreme ONE OS Switching interface configuration\)](#) on page 804

[mtu \(Extreme ONE OS Switching global configuration\)](#) on page 807

[nd-proxy](#) on page 809

[nd-snooping](#) on page 810

[neighbor \(BGP peer group configuration\)](#) on page 811

[network \(BGP address family configuration\)](#) on page 813

[network \(IAH VM configuration\)](#) on page 815

[next-hop-enable-default \(BGP address family configuration\)](#) on page 817

[next-hop-in \(routing policy statement bgp conditions configuration\)](#) on page 819

[next-hop-recursion \(BGP address family configuration\)](#) on page 821

[nexthop-self \(BGP peer group address family configuration\)](#) on page 823

[nexthop-unchanged \(BGP peer group address family configuration\)](#) on page 825

[ntp \(IAH VM configuration\)](#) on page 827

[nve \(EVPN instance configuration\)](#) on page 829

[nve \(overlay configuration\)](#) on page 831

[nvo \(BGP peer group configuration\)](#) on page 833

[nvo \(QoS configuration\)](#) on page 835

[nvo](#) on page 836

[nvo \(VRF instance type configuration\)](#) on page 838

[onlink \(IPv6 router advertisement prefix configuration\)](#) on page 840

[other-config](#) on page 842

[output \(interface QoS configuration\)](#) on page 844

[overlay-networks](#) on page 845

[password \(IAH VM configuration\)](#) on page 846

[peer-group](#) on page 847

[peer \(MLAG configuration\)](#) on page 849

[permit \(IPv4 ACL configuration\)](#) on page 851

[permit \(IPv6 ACL configuration\)](#) on page 856

[permit \(MAC ACL configuration\)](#) on page 861

[ping](#) on page 864

[policy-result \(routing policy statement actions configuration\)](#) on page 867

[policy \(routing policy configuration\)](#) on page 869

[port ethernet](#) on page 871

[preferred-lifetime \(IPv6 router advertisement prefix configuration\)](#) on page 873

[prefix \(IPv6 router advertisement configuration\)](#) on page 875

[prefix-independent-convergence \(Extreme ONE OS Switching\)](#) on page 877

[prefix-set](#) on page 879

[prefix](#) on page 881

[profile \(BFD configuration\)](#) on page 883

[protocol lacp](#) on page 885

[protocol lldp](#) on page 886

[qos \(Extreme ONE OS Switching\)](#) on page 888

[ram](#) on page 889

[receive \(LLDP configuration\)](#) on page 891

[remark-map \(interface QoS configuration\)](#) on page 893

[remark-map \(QoS configuration\)](#) on page 895

[remark \(QoS configuration\)](#) on page 897

[remote-as \(BGP peer group configuration\)](#) on page 899

[resilient-hash \(VRF configuration\)](#) on page 901

[restart-time \(BGP graceful restart configuration\)](#) on page 902

[retain-route-target-all \(BGP L2VPN EVPN address family configuration\)](#) on page 904

[role \(MLAG configuration\)](#) on page 906

[route-distinguisher \(EVPN configuration\)](#) on page 908

[route-distinguisher \(VRF configuration\)](#) on page 910

[route-policy](#) on page 912

[route-reflector-client](#) on page 913

[route-target \(Extreme ONE OS Switching\)](#) on page 915

[router bgp](#) on page 917

[router-id \(BGP configuration\)](#) on page 919

[send-default-route \(BGP address family configuration\)](#) on page 921

[set-as-path-prepend \(routing policy statement BGP actions configuration\)](#) on page 923

[set-community \(routing policy statement BGP actions configuration\)](#) on page 925

[set-ext-community \(routing policy statement BGP actions configuration\)](#) on page 927

[set-local-pref \(routing policy statement BGP actions configuration\)](#) on page 929

[set-med \(routing policy statement BGP actions configuration\)](#) on page 931

[set-next-hop \(routing policy statement BGP actions configuration\)](#) on page 933

[set-route-origin \(routing policy statement BGP actions configuration\)](#) on page 935

[show access-list](#) on page 937

[show anycast-gateway \(IPv4 and IPv6\)](#) on page 939

[show bfd neighbors application \(Extreme ONE OS Switching\)](#) on page 941

[show bfd neighbors dest-ip \(Extreme ONE OS Switching\)](#) on page 945

[show bfd neighbors interface \(Extreme ONE OS Switching\)](#) on page 947

[show bfd neighbors](#) on page 948

[show bfd profile](#) on page 951

[show bfd](#) on page 953

[show bgp vrf l2vpn-evpn](#) on page 955

[show bgp vrf neighbor](#) on page 957

[show bgp vrf neighbors](#) on page 962

[show bgp vrf routes](#) on page 964

[show bgp vrf summary](#) on page 971

[show bridge-domain](#) on page 973

[show counters bridge-domain](#) on page 975

[show counters interface ethernet](#) on page 978

[show counters interface port-channel](#) on page 983

[show counters interface tunnel](#) on page 986

[show counters lacp](#) on page 987

[show counters link-fault-signaling](#) on page 989

[show counters lldp interface](#) on page 991

[show counters lldp summary](#) on page 993

[show counters mlag peer](#) on page 995

[show counters qos interface control-plane queue](#) on page 997

[show counters qos interface ethernet queue](#) on page 1000

[show iah host](#) on page 1002

[show iah snapshot](#) on page 1005

[show iah vm](#) on page 1006

[show interface ethernet](#) on page 1010

[show interface ethernet link-error-disable](#) on page 1012

[show interface loopback-mode](#) on page 1013

[show interface subinterface](#) on page 1014

[show interface tunnel \(Extreme ONE OS Switching\)](#) on page 1016

[show ipv4 interface](#) on page 1018

[show ipv6 interface](#) on page 1020

[show ipv4 neighbor interface](#) on page 1022

[show ipv4 neighbor vrf](#) on page 1024

[show ipv6 neighbor interface](#) on page 1028

[show ipv6 neighbor vrf](#) on page 1030

[show ipv4 route vrf \(IP routing\)](#) on page 1033

[show ipv6 route vrf \(IP routing\)](#) on page 1038

[show lacp interface](#) on page 1041
[show lacp system-identifier](#) on page 1044
[show link-fault-signaling](#) on page 1045
[show lldp neighbors interface](#) on page 1046
[show lldp neighbors](#) on page 1049
[show lldp](#) on page 1051
[show l2 mac-move-detection](#) on page 1054
[show l2 mac-move-detection shut-list](#) on page 1055
[show mac-address-table bridge-domain](#) on page 1056
[show mirror](#) on page 1059
[show mlag](#) on page 1061
[show mlag interfaces](#) on page 1063
[show mlag peer](#) on page 1066
[show mlag peer bfd](#) on page 1069
[show overlay-networks](#) on page 1071
[show qos interface](#) on page 1072
[show qos nvo](#) on page 1075
[show qos remark-map](#) on page 1076
[show qos traffic-class-map](#) on page 1078
[show running-config access-list](#) on page 1080
[show running-config lldp](#) on page 1087
[show system mac-address-table aging-time \(Extreme ONE OS Switching\)](#) on page 1088
[show system maintenance-mode](#) on page 1089
[show system redundant-management](#) on page 1090
[shutdown \(BGP peer group configuration\)](#) on page 1091
[shutdown \(Ethernet interface configuration\)](#) on page 1093
[shutdown \(subinterface configuration\)](#) on page 1095
[shutdown \(tunnel interface configuration\)](#) on page 1097
[size \(IAH VM disk configuration\)](#) on page 1099
[source-interface \(MLAG peer keepalive configuration\)](#) on page 1101
[source-interface \(NVE configuration\)](#) on page 1103
[source nve](#) on page 1105
[speed \(Ethernet interface configuration\)](#) on page 1107
[speed \(port channel interface configuration\)](#) on page 1109
[split-horizon](#) on page 1111
[src-protocol](#) on page 1113
[stale-route-time \(BGP graceful restart configuration\)](#) on page 1115
[statement \(routing policy configuration\)](#) on page 1117
[static-mac-address](#) on page 1119
[static-route](#) on page 1122
[static-route enable-bfd](#) on page 1127

[statistics \(bridge domain configuration\)](#) on page 1131

[subinterface \(tagged\)](#) on page 1132

[subinterface \(untagged strict\)](#) on page 1133

[subinterface vlan \(untagged\)](#) on page 1134

[suppress-arp](#) on page 1136

[suppress-nd](#) on page 1137

[suppress-tlv](#) on page 1138

[system maintenance-mode](#) on page 1140

[table-connections](#) on page 1142

[threshold-monitor](#) on page 1143

[threshold-monitor \(CPU and memory\)](#) on page 1145

[timers \(BGP peer group configuration\)](#) on page 1147

[timezone \(IAH VM configuration\)](#) on page 1149

[traceroute](#) on page 1151

[traffic-class cos \(QoS remark map configuration\)](#) on page 1153

[traffic-class dscp \(QoS remark map configuration\)](#) on page 1155

[traffic-class-map \(interface QoS input configuration\)](#) on page 1157

[traffic-class-map \(QoS configuration\)](#) on page 1159

[transmit \(LLDP configuration\)](#) on page 1161

[trust \(interface QoS input configuration\)](#) on page 1162

[trusted-peer \(IAH VM configuration\)](#) on page 1164

[tunnel-mode](#) on page 1166

[type \(MLAG peer keepalive configuration\)](#) on page 1168

[type \(tunnel interface configuration\)](#) on page 1170

[update-source \(BGP peer group configuration\)](#) on page 1172

[uplink-track interface \(MLAG configuration\)](#) on page 1176

[use-multiple-paths \(BGP address family configuration\)](#) on page 1178

[use-multiple-paths](#) on page 1180

[valid-lifetime \(IPv6 router advertisement prefix configuration\)](#) on page 1182

[vlan-id](#) on page 1185

[vm \(IAH configuration\)](#) on page 1186

[vni-domain \(bridge domain configuration\)](#) on page 1188

[vni-domain \(overlay configuration\)](#) on page 1189

[vrf \(Extreme ONE OS Switching\)](#) on page 1191

access-list

Use the **access-list** command to define an IP access control list (ACL), IPv6 ACL, or MAC ACL and enter the corresponding configuration mode.

Syntax

```
{ ipv4 | ipv6 | mac } access-list acl-name  
no { ipv4 | ipv6 | mac } access-list acl-name
```

Command Default

No ACL is defined.

Parameters

ipv4

Creates an IPv4 ACL and enters IP ACL configuration mode.

ipv6

Creates an IPv6 ACL and enters IPv6 ACL configuration mode.

mac

Creates a MAC ACL and enters MAC ACL configuration mode.

acl-name

Name of the ACL to be defined. You can use any combination of 1 to 64 uppercase or lowercase alphabetic characters, numerals, underscores, hyphens, or periods. You must use an alphabetic character or an underscore as the first character.

Modes

Global configuration (config)

Usage Guidelines

When you create an ACL rule, you can use the following fields to filter incoming packets:

Field	Description
s-ip	Source IP address
d-ip	Destination IP address
ip-protocol	tcp/udp/icmp/igmp/esp/customProtocolNumber
TcpFlags	Set to true or false
L4Sport	L4 source port

Field	Description
L4DPort	L4 destination port
vlan	VLAN type
dscp	DSCP bit or mapping DSCP value

To remove an IP ACL, IPv6 ACL, or MAC ACL, use the **no** form of this command.

Examples

The following example defines an ACL named v4acl:

```
device# configure terminal
device(config)# ipv4 access-list v4acl
device(config-ipv4-acl)# permit ip 1.1.1.1/32 2.2.2.0 255.255.255.0 count
device(config-ipv4-acl)# exit
device#
```

The following example defines an ACL named v6acl:

```
device# configure terminal
device(config)# ipv6 access-list v6acl
device(config-ipv6-acl)# permit tcp 1::0/64 2::0 ffff::0 sync log
device(config-ipv6-acl)# exit
device#
```

The following example defines an ACL named macacl:

```
device# configure terminal
device(config)# mac access-list macacl
device(config-mac-acl)# permit 11:22:33:44:55:66 ff:ff:ff:ff:ff:ff any vlan 56
device(config-mac-acl)# exit
device#
```

The following example displays the list of all existing ACLs:

```
device# show running-config access-list

ipv4 access-list v4acl
ipv6 access-list v6acl
mac access-list macacl
device#
```


access-list (control plane configuration)

Use the **access-list** command to attach an IPv4 access control list (ACL) or IPv6 ACL to the control plane.

Syntax

```
{ ipv4 | ipv6 } access-list acl-name  
no { ipv4 | ipv6 } access-list acl-name
```

Command Default

No ACLs are attached to the control plane.

Parameters

ipv4

Attaches an IPv4 ACL.

ipv6

Attaches an IPv6 ACL.

acl-name

Name of the ACL to be attached. You can use any combination of 1 to 64 uppercase or lowercase alphabetic characters, numerals, underscores, hyphens, or periods. You must use an alphabetic character or an underscore as the first character.

Modes

Control plane configuration (config-control-plane)

Usage Guidelines

An ACL attached to the control plane filters packets going to the CPU.

Only one ACL per type (IP or IPv6) is allowed.

Attaching MAC ACLs to the control plane is not supported.

To remove an IPv4 ACL or an IPv6 ACL from the control plane, use the **no** form of this command.

Examples

The following example attaches an IPv4 ACL and an IPv6 ACL to the control plane:

```
device# configure terminal  
device(config)# control-plane  
device(config-control-plane)# ipv4 access-list cp-ipv4-acl
```

```
device(config-control-plane)# ipv6 access-list cp-ipv6-acl
device(config-control-plane)#
```

The following example shows the configuration that is running currently on the control plane. In this example, two ACLs are attached:

```
device# show running-config control-plane

control-plane
  ipv4 access-list cp-ipv4-acl
  ipv6 access-list cp-ipv6-acl
device#
```

access-list (interface configuration)

Use the **access-list** command to command to attach an access control list (ACL) to an Ethernet, virtual Ethernet (VE), or link aggregation group (LAG) interface.

Syntax

```
{ ipv4 | ipv6 | mac } access-list acl-name { in | out }  
no { ipv4 | ipv6 | mac } access-list acl-name { in | out }
```

Command Default

No ACLs are defined on the current interface.

Parameters

ipv4

Attaches an IPv4 ACL.

ipv6

Attaches an IPv6 ACL.

mac

Attaches a MAC ACL.

acl-name

Name of the ACL to be attached. You can use any combination of 1 to 64 uppercase or lowercase alphabetic characters, numerals, underscores, hyphens, or periods. You must use an alphabetic character or an underscore as the first character.

in

Attaches the ACL in the ingress direction.

out

(Ethernet and VE interfaces only) Attaches the ACL in the egress direction.

Modes

Ethernet interface configuration (*config-if-eth-name*)

Port Channel interface configuration (*config-if-po-name*)

Virtual Ethernet interface configuration (*config-if-ve-name*)

Usage Guidelines

On Ethernet and VE interfaces, only one ACL per type is allowed in one direction.

On LAG interfaces, ACLs are allowed only in the ingress direction. and only one ACL per type is allowed per interface.

To remove an ACL from an Ethernet, VE, or LAG interface, use the **no** form of this command.

Examples

The following example configures an ACL of each type and in both directions on Ethernet interface 0/1:

```
device# configure terminal
device(config)# interface ethernet 0/1
device(config-if-eth-0/1)# mac access-list macacl in
device(config-if-eth-0/1)# mac access-list macacl out
device(config-if-eth-0/1)# ip access-list v4acl in
device(config-if-eth-0/1)# ip access-list v4acl out
device(config-if-eth-0/1)# ipv6 access-list v6acl in
device(config-if-eth-0/1)# ipv6 access-list v6acl out
device(config-if-eth-0/1)#
```

The following example configures an ACL of each type and in both directions on virtual Ethernet interface 1:

```
device# configure terminal
device(config)# interface ve 1
device(config-if-ve-1)# mac access-list macacl in
device(config-if-ve-1)# mac access-list macacl out
device(config-if-ve-1)# ip access-list v4acl in
device(config-if-ve-1)# ip access-list v4acl out
device(config-if-ve-1)# ipv6 access-list v6acl in
device(config-if-ve-1)# ipv6 access-list v6acl out
device(config-if-ve-1)#
```

The following example configures an ACL of each type and in the ingress direction on LAG interface 1:

```
device# configure terminal
device(config)# interface port-channel 1
device(config-if-po-1)# mac access-list macacl in
device(config-if-po-1)# ip access-list v4acl in
device(config-if-po-1)# ipv6 access-list v6acl in
device(config-if-po-1)#
```

The following example displays the configuration that is running currently on Ethernet interface 0/1. In this example, the interface is using an ACL of all three types in the ingress direction as well as the egress direction:

```
device# show running-config interface ethernet 0/1

interface ethernet 0/1
shutdown
mac access-list macacl in
ip access-list v4acl in
ipv6 access-list v6acl in
mac access-list macacl out
ip access-list v4acl out
ipv6 access-list v6acl out
device#
```

The following example displays the configuration that is running currently on virtual Ethernet interface 1. In this example, the interface is using an ACL of all three types in the ingress direction as well as the egress direction:

```
device# show running-config interface ve 1

interface ve 1
mac access-list macacl in
ip access-list v4acl in
ipv6 access-list v6acl in
mac access-list macacl out
ip access-list v4acl out
ipv6 access-list v6acl out
device#
```

The following example displays the configuration that is running currently on LAG interface 1. In this example, the interface is using an ACL of all three types in the ingress direction:

```
device# show running-config interface port-channel 1

interface port-channel 1
mac access-list macacl in
ip access-list v4acl in
ipv6 access-list v6acl in
device#
```

actions (routing policy map statement configuration)

Use the **actions** command to enter Routing policy statement actions configuration (config-route-policy-policy-*name*-stmt-*number*-actions) mode.

Syntax

actions
no actions

Command Default

The specified routing policy statement contains no actions.

Parameters

This command has no keywords or variables.

Modes

Routing policy statement configuration (config-route-policy-policy-*name*-stmt-*number*)

Usage Guidelines

This command enters the mode for adding actions to the specified routing policy statement. Each statement in a routing policy can have only one BGP actions container. This container can contain multiple actions.

To remove all match actions from a routing policy statement, use the **no** form of this command.

Examples

The following example creates a routing policy named map1 that contains a policy statement with sequence number 10 and enters Routing policy statement actions configuration (config-route-policy-policy-*name*-stmt-*number*-actions) mode:

```
device# configure terminal
device(config)# route-policy
device(config-route-policy)# policy map1
device(config-route-policy-policy-map1)# statement 10
device(config-route-policy-policy-map1-stmt-10)# actions
device(config-route-policy-policy-map1-stmt-10-actions)# policy-result permit
device(config-route-policy-policy-map1-stmt-10-actions)# bgp-actions
device(config-route-policy-policy-map1-stmt-10-actions-bgp)# set-med 100
```

The following example displays the routing policy configuration that is running currently on the device. In this example, a routing policy named map1 contains a policy statement with sequence number 10 with several actions configured:

```
device# show running-config route-policy

route-policy
  policy map1
    statement 10
      actions
        policy-result permit
        bgp-actions
          set-med 100
device#
```

activate (BGP address family configuration)

Use the **activate** command to activate the routing table for a specific BGP address family within a routing process in a VRF instance.

Syntax

activate

no activate

Command Default

Existing BGP address families in the routing process remain deactivated.

Parameters

This command has no keywords or variables.

Modes

VRF BGP address family configuration (*config-vrf-bgp-address-family-type*)

Usage Guidelines

This command resets all established sessions to negotiate the new capability for the specified address family.

Use the **no** form of this command to deactivate the specified address family.



Caution

The **no** form of this command deletes all address family specific routes. Improper usage of this command might disrupt service.

Examples

The following example configures a routing process under a VRF instance named violet. This example creates and activates three address families:

```
device# configure terminal
device(config)# vrf violet
device(config-vrf-violet)# router bgp
device(config-vrf-bgp)# address-family ipv4 unicast
device(config-vrf-bgp-ipv4u)# activate
device(config-vrf-bgp-ipv4u)# exit
device(config-vrf-bgp)# address-family ipv6 unicast
device(config-vrf-bgp-ipv6u)# activate
device(config-vrf-bgp-ipv6u)# exit
device(config-vrf-bgp)# address-family l2vpn evpn
device(config-vrf-bgp-l2vpn-evpn)# activate
device(config-vrf-bgp-l2vpn-evpn)#
```


The following example displays the configuration of a VRF instance named violet that is running currently on the device. In this example, the instance contains a routing process with three activated address families:

```
device# show running-config vrf violet

vrf violet
  router bgp
    address-family ipv4 unicast
      activate
    !
    address-family ipv6 unicast
      activate
    !
    address-family l2vpn evpn
      activate
    !
  !
!
```

activate (BGP peer group address family configuration)

Use the **activate** command to activate a table for a specific address family to hold the routes for a BGP peer group within a VRF instance.

Syntax

activate

no activate

Command Default

Existing address families under the specified BGP peer group remain deactivated.

Parameters

This command has no keywords or variables.

Modes

VRF BGP peer group address family configuration (*config-vrf-bgp-pg-address-family-type*)

Usage Guidelines

This command resets all established sessions to negotiate the new capability for the specified address family.

Use the **no** form of this command to deactivate the specified address family.



Caution

The **no** form of this command tears down all established peer group members and deletes all address family specific routes. Improper usage of this command might disrupt service.

Examples

The following example configures two BGP peer groups (named group1 and group2) under a VRF instance named violet:

```
device# configure terminal
device(config)# vrf violet
device(config-vrf-violet)# router bgp
device(config-vrf-bgp)# peer-group group1
device(config-vrf-bgp-pg)# address-family ipv4 unicast
device(config-vrf-bgp-pg-ipv4u)# activate
device(config-vrf-bgp-pg-ipv4u)# exit
device(config-vrf-bgp-pg)# address-family l2vpn evpn
device(config-vrf-bgp-pg-l2vpn-evpn)# activate
device(config-vrf-bgp-pg-l2vpn-evpn)# exit
```

```
device(config-vrf-bgp-pg) # exit
device(config-vrf-bgp) # peer-group group2
device(config-vrf-bgp-pg) # address-family ipv6 unicast
device(config-vrf-bgp-pg) # address-family l2vpn evpn
device(config-vrf-bgp-pg-l2vpn-evpn) # activate
device(config-vrf-bgp-pg-l2vpn-evpn) #
```

In this example:

- Two address families are activated under group 1: One that uses IPv4 unicast prefixes and another that uses L2VPN EVPN address prefixes.
- Two address families are activated under group 2: One that uses IPv6 unicast prefixes and another that uses L2VPN EVPN address prefixes.

The following example displays the configuration of a VRF instance named violet that is running currently on the device. In this example, the instance contains two BGP peer groups (group1 and group2), and two address families are activated under each peer group:

```
device# show running-config vrf

vrf violet
!
  router bgp
    peer-group group1
      address-family ipv4 unicast
        activate
      address-family l2vpn evpn
        activate
    peer-group group2
      address-family ipv6 unicast
        activate
      address-family l2vpn evpn
        activate
  !
device#
```

activate-network

Use the **activate-network** command to enable the network interfaces on a virtual machine (VM).

Syntax

activate-network

Command Default

Network interfaces are disabled on the current VM.

Parameters

This command has no keywords or variables.

Modes

IAH VM configuration (config-iah-vm-*name*)

Usage Guidelines

For security reasons, network interfaces are not enabled on (attached to) a VM during its first deployment. Before using this command, you should log into the VM console and take the necessary security measures (such as changing the password) before deploying the VM again. After the VM is secure, there is no need to detach the network interfaces (a **no** version of this command is not implemented in this release).

If the network interfaces are enabled, before deploying the VM again, make sure that its disk is already secured.

If the VM is already deployed, this command attaches the network interfaces to the VM. If the VM is not deployed, the network interfaces are attached at its next deployment.

Note that configuration is not allowed when the VM is in the deployed state.

To display details about a specific VM, see [show iah vm](#) on page 1006.

Examples

The following example enables the network interfaces for a VM named Ubuntu1 on the device:

```
device# configure terminal
device(config)# iah
device(config-iah)# vm Ubuntu1
device(config-iah-vm-Ubuntu1)# activate-network
device(config-iah-vm-Ubuntu1)#
```

The following example displays the IAH configuration that is running currently on the device. In this example, the network interfaces are enabled for a VM named Ubuntu1:

```
device# show running-config iah

iah
vm Ubuntu1
  description Ubuntu demo vm
  auto-boot true
  activate-network
  cpus 1
  ram 1024
  disk sda
    size 5000
    format qcow2
    bus sata
    file disk://iah/disk1.qcow2
  !
  network nic2
    interface management 0
  !
!
device#
```

add-paths receive

Use the **add-paths receive** command to enable peers in a BGP address family within a routing process in a VRF instance to receive additional paths.

Syntax

```
add-paths receive  
no add-paths receive
```

Command Default

Peer devices configured under the specified BGP address family cannot receive additional paths.

Parameters

This command has no keywords or variables.

Modes

VRF BGP address family configuration (*config-vrf-bgp-address-family-type*)

VRF BGP peer group address family configuration (*config-vrf-bgp-pg-address-family-type*)

Usage Guidelines

This command enables BGP to negotiate the additional paths capability with its neighbors and receive additional paths for the same prefix or the same prefix under this peer group. This command enables the reception of multiple paths without new paths implicitly replacing the previous paths, which achieves path diversity rather than path hiding. Peer devices exchange and negotiate this capability during session establishment.



Note

Changes to the additional paths capability for peers in a BGP address family take effect only after the BGP session is restarted.



Note

An additional paths capability configured at the peer group level takes precedence over any additional paths capability configured at the BGP address family level.

Additional paths can be enabled as receive only.

Use the **no** form of this command to deactivate the additional paths receive functionality in the specified address family.

Examples

The following example configures a routing process under a VRF instance named violet. This example creates and activates two address families. In this example, additional paths receive functionality is enabled for each address family:

```
device# configure terminal
device(config)# vrf violet
device(config-vrf-violet)# router bgp
device(config-vrf-bgp)# address-family ipv4 unicast
device(config-vrf-bgp-ipv4u)# add-paths receive
device(config-vrf-bgp-ipv4u)# activate
device(config-vrf-bgp-ipv4u)# exit
device(config-vrf-bgp)# address-family ipv6 unicast
device(config-vrf-bgp-ipv6u)# add-paths receive
device(config-vrf-bgp-ipv6u)# activate
device(config-vrf-bgp-ipv6u)#
```

The following example displays the configuration of a VRF instance named violet that is running currently on the device. In this example, the instance contains a routing process with two activated address families. Each address family uses the additional paths receive functionality:

```
device# show running-config vrf violet

vrf violet
  router bgp
    address-family ipv4 unicast
      add-paths receive
      activate
    !
    address-family ipv6 unicast
      add-paths receive
      activate
    !
  !
!
device#
```

The following example configures a routing process under a VRF instance named violet. This example creates and activates one address family under a peer group named pg1. In this example, additional paths receive functionality is enabled for the address family:

```
device# configure terminal
device(config)# vrf violet
device(config-vrf-violet)# router bgp
device(config-vrf-bgp)# peer-group pg1
device(config-vrf-bgp-pg)# address-family ipv4 unicast
device(config-vrf-bgp-pg-ipv4u)# add-paths receive
device(config-vrf-bgp-pg-ipv4u)#
```

The following example displays the configuration of a VRF instance named violet that is running currently on the device. In this example, the instance contains a routing process with one activated address family under a peer group named pg1. The address family uses the additional paths receive functionality:

```
device# show running-config vrf violet

vrf violet
  router bgp
```

```
    peer-group pg1
      address-family ipv4 unicast
      add-paths receive
    !
  !
device#
```


address-family (BGP configuration)

Use the **address-family** command to configure an address family routing table for the BGP routing process within a virtual routing and forwarding (VRF) instance and enter BGP address family configuration mode.

Syntax

```
address-family { ipv4 | ipv6 | l2vpn } { unicast | multicast | evpn }  
no address-family { ipv4 | ipv6 | l2vpn } { unicast | multicast | evpn }
```

Command Default

The BGP routing process within the current VRF instance has no address family routing tables.

Parameters

ipv4

Specifies the IPv4 address family.

ipv6

Specifies the IPv6 address family.

l2vpn

Specifies the Layer 2 VPN (L2VPN) address family.

unicast

Specifies only unicast address prefixes.

multicast

Specifies only multicast address prefixes.

evpn

Specifies only Ethernet VPN (EVPN) address prefixes.

Modes

VRF BGP configuration (config-vrf-bgp)

Usage Guidelines

This command defines a specific behavior of BGP regarding many supported Layer 3 protocols, such as IPv4 networks for unicast, IPv6 networks for unicast, and L2VPN networks for EVPN. BGP treats these behaviors separately as if it is configured in a separate instance for each one. These instances do not interact with each other (even if they are of the same basic type). This lets you configure a single BGP routing process to maintain several databases of different networks.

BGP, as a multiprotocol routing protocol, can carry different types of routing information in a single BGP session. But for each of these network types, you must create an address family to tell BGP which neighbors also support the same protocol type, which routes to inject, how the individual networks will be filtered or their attributes modified, and so on.

Use the **no** form of this command to bring down all established sessions and delete the table for the specified address family.

Examples

The following example configures three address families under a VRF named violet:

```
device# configure terminal
device(config)# vrf violet
device(config-vrf-violet)# router bgp
device(config-vrf-bgp)# address-family ipv4 unicast
device(config-vrf-bgp-ipv4u)# exit
device(config-vrf-bgp)# address-family ipv6 unicast
device(config-vrf-bgp-ipv6u)# exit
device(config-vrf-bgp)# address-family l2vpn evpn
device(config-vrf-bgp-l2vpn-evpn)#
```

The following example displays the configuration of a VRF instance named violet that is running currently on the device. In this example, the instance contains a BGP routing process that is configured with three address families:

```
device# show running-config vrf violet

vrf violet
  router bgp
    address-family ipv4 unicast
    address-family ipv6 unicast
    address-family l2vpn evpn
  !
!
device#
```

address-family (BGP peer group configuration)

Use the **address-family** command to configure a routing session for a BGP peer group within a virtual routing and forwarding (VRF) instance and enter BGP peer group address family configuration mode.

Syntax

```
address-family { ipv4 | ipv6 | l2vpn } { unicast | multicast | evpn }  
no address-family { ipv4 | ipv6 | l2vpn } { unicast | multicast | evpn }
```

Command Default

The specified BGP peer group has no address families.

Parameters

ipv4

Specifies the IPv4 address family.

ipv6

Specifies the IPv6 address family.

l2vpn

Specifies the Layer 2 VPN (L2VPN) address family.

unicast

Specifies only unicast address prefixes.

multicast

Specifies only multicast address prefixes.

evpn

Specifies only Ethernet VPN (EVPN) address prefixes.

Modes

VRF BGP peer group configuration (config-vrf-bgp-pg)

Usage Guidelines

This command defines a specific behavior of BGP regarding many supported Layer 3 protocols, such as IPv4 networks for unicast, IPv6 networks for unicast, and L2VPN networks for EVPN. BGP treats these behaviors separately as if it is configured in a separate instance for each one. These instances do not interact with each other (even if they are of the same basic type). This lets you configure a single BGP process to maintain several databases of different networks.

BGP, as a multiprotocol routing protocol, can carry different types of routing information in a single BGP session. But for each of these network types, you must

create an address family to tell BGP which neighbors also support the same protocol type, which routes to inject, how the individual networks will be filtered or their attributes modified, and so on.

Use the **no** form of this command to bring down all established sessions and delete the table for the specified address family.

Examples

The following example configures two BGP peer groups (named group1 and group2) under a VRF named violet:

```
device# configure terminal
device(config)# vrf violet
device(config-vrf-violet)# router bgp
device(config-vrf-bgp)# peer-group group1
device(config-vrf-bgp-pg)# address-family ipv4 unicast
device(config-vrf-bgp-pg-ipv4u)# exit
device(config-vrf-bgp-pg)# address-family l2vpn evpn
device(config-vrf-bgp-pg-l2vpn-evpn)# exit
device(config-vrf-bgp-pg)# exit
device(config-vrf-bgp)# peer-group group2
device(config-vrf-bgp-pg)# address-family ipv6 unicast
device(config-vrf-bgp-pg-ipv6u)# exit
device(config-vrf-bgp-pg)# address-family l2vpn evpn
device(config-vrf-bgp-pg-l2vpn-evpn)# exit
device(config-vrf-bgp-pg-l2vpn-evpn-ipv6u)# exit
```

In this example:

- group1 is configured with two address families: One that uses IPv4 unicast prefixes and another that uses L2VPN EVPN address prefixes.
- group2 is configured with two address families: One that uses IPv6 unicast prefixes and another that uses L2VPN EVPN address prefixes.

The following example displays the configuration of a VRF instance named violet that is running currently on the device. In this example, the instance contains two BGP peer groups (group1 and group2), and each group is configured with two address families:

```
device# show running-config vrf

vrf violet
!
  router bgp
    peer-group group1
      address-family ipv4 unicast
      address-family l2vpn evpn
    peer-group group2
      address-family ipv6 unicast
      address-family l2vpn evpn
!
device#
```

address-type (MLAG peer configuration)

Use the **address-type** command to specify the address type for multichassis link aggregation (MLAG) peer keepalive mode.

Syntax

```
address-type { auto | manual }  
no address-type
```

Command Default

In an MLAG peer, the default address-type for a primary keepalive is manual, and the default address-type for a secondary keepalive is auto.

Parameters

auto

Automatic keepalive address type.

manual

Manual keepalive address type.

Modes

MLAG peer keepalive configuration (config-mlag-peer-keepalive-*name*)

Usage Guidelines

Using **auto** for a backup keepalive automatically chooses the management interface as the source interface and the peer's management IP address as the destination IP address (which is learned via an MLAG sync message). Using **auto** for a primary keepalive has no effect.

To disable the address type for MLAG peer keepalive mode, use the **no** form of this command.

Examples

The following example displays the MLAG session details on an MLAG device. In this example, the primary keepalive on the MLAG peer uses address-type manual mode, and the secondary keepalive uses address-type auto mode:

```
device# configure terminal  
device(config)# mlag  
device(config-mlag)# peer peer01  
device(config-mlag-peer-peer01)# keepalive keepalive01  
device(config-mlag-peer-keepalive-keepalive01)# address-type auto  
device(config-mlag-peer-keepalive-keepalive01)#
```

The following example displays the MLAG configuration that is running currently on a device. In this example, the primary keepalive is configured to manual mode, and the secondary keepalive is configured to auto mode:

```
device# show mlag peer

Peer dut4
=====
Peer State           : UP
MCP State            : UP
Role                 : PRIMARY
Elected MAC         : 02:00:22:33:44:55
Extend Bridge Count  : 46
Peer Exception       : None

    Keepalive kp-primary
    -----
    Destination Address: 11.1.2.2
    State               : UP
    Type                : PRIMARY
    Address Type        : MANUAL
    Source Interface    : port-channel 20.0 ( 11.1.2.1 )
    First Rx            : 2025-03-05 14:01:30
    Last Rx             : 2025-03-05 16:17:35

    Keepalive kp-secondary
    -----
    Destination Address: 10.38.65.30
    State               : UP
    Type                : SECONDARY
    Address Type        : AUTO
    Source Interface    : management 0.0 ( 10.38.65.31 )
    First Rx            : 2025-03-05 14:01:37
    Last Rx             : 2025-03-06 12:06:17
device#
```

The following example displays the MLAG configuration that is running currently on a device. In this example, the primary keepalive is configured to manual mode, and the secondary keepalive is configured to auto mode:

```
device# show running-config mlag

mlag
  peer dut4
    keepalive kp-primary
      type primary
      address-type manual
      destination 11.1.2.2
      source-interface port-channel 20
    keepalive kp-secondary
      type secondary
      address-type auto
  !
!
!
device#
```

advertisement (IPv6 router advertisement prefix configuration)

Use the **prefix** command to enable advertisement of a prefix for IPv6 router advertisement on an interface.

Syntax

```
advertisement  
no advertisement
```

Command Default

Advertisement within the current prefix is enabled.

Parameters

This command has no keywords or variables.

Modes

IPv6 router advertisement configuration (config-if-*name*-ipv6-router-advertisement)

Usage Guidelines

Router advertisements are ICMPv6 messages sent by an IPv6 enabled router that are used by hosts to detect routers on an attached link. This functionality is similar to the IPv4 ICMP Router Discovery protocol.

Neighbor discovery defines five ICMP packet types: router solicitation and advertisement, neighbor solicitation and advertisement, and redirect messages.

Router advertisements carry information such as the router link layer address, prefixes supported in the network, and the MTU setting. Router advertisements are periodic messages or are also sent in response to router solicitations. Because of the nature of its functionality, router advertisement is considered as a separate feature from neighbor discovery. Two messages are supported as part of this functionality: router solicitation and router advertisement.

Each configured interface sends a router advertisement message periodically to the all-nodes link-local multicast address (FF02::1). To avoid synchronization of advertisements from other routers, the time interval is randomized to send these messages between the configured maximum and minimum time periods.

Router advertisement includes the following information:

- Default router—Whether the sender can be a default router and, if so, the lifetime for the router
- List of prefixes for on-link addresses

- Auto-configuration information
- Network information—Hop limit and MTU setting to use

To disable advertisement within a prefix for IPv6 router advertisement on an interface, use the **no** form of this command. When advertisement is disabled within a prefix, that prefix is not advertised within router advertisement messages that are sent as a result of router solicitation messages.

Examples

The following example configures a prefix of 100::1/64 for IPv6 router advertisement on Ethernet interface 0/1 and enables advertisement within that prefix:

```
device# configure terminal
device(config)# interface ethernet 0/1
device(config-if-eth-0/1)# ipv6-router-advertisement
device(config-if-eth-0/1-ipv6-router-advertisement)# prefix 100::1/64
device(config-if-ipv6-router-advertisement-prefix-100::1/64)# advertisement
device(config-if-ipv6-router-advertisement-prefix-100::1/64)#
```

The following example displays the configuration of Ethernet interface 0/1 that is running currently on the device. In this example, advertisement is enabled within prefix 100::1/64 on the interface:

```
device# show running-config interface ethernet 0/1

!
interface ethernet 0/1
  ipv6 address 1111::10/64
  ipv6 neighbor 1111::1 00:04:96:eb:c4:51
  ipv6-router-advertisement
    hop-limit 133
    lifetime 200
    managed-config-flag
    mode disable-unsolicited-ra
    other-config
    enable
    interval 100 min 50
    prefix 100::1/63
    prefix 100::1/64
      advertisement
        valid-lifetime 200
  !
  no shutdown
!
device#
```


agetime (ARP)

Use the agetime command to configure the address resolution protocol (ARP) agetime on an ARP proxy in a bridge domain.

Syntax

agetime *seconds*

no agetime

Command Default

The agetime on an ARP proxy is set to 1200 seconds.

Parameters

seconds

Number of seconds.

Modes

Bridge domain VLAN ARP proxy configuration (config-vlan-bd-*name*-arp-proxy)

Usage Guidelines

To return the ARP agetime setting to the default value, use the **no** form of this command.

Examples

The following example configures an agetime of 900 seconds on the ARP proxy in a bridge domain with 100 as its ID:

```
device# configure terminal
device(config)# bridge-domain 100 mode vlan
device(config-vlan-bd-100)# arp-proxy
device(config-vlan-bd-100-arp-proxy)# agetime 900
device(config-vlan-bd-100-arp-proxy)#
```

The following example displays the configuration for bridge domain 100 that is running currently on the device. In this example, an agetime of 900 seconds is configured on the ARP proxy in the bridge domain:

```
device# show running-config bridge-domain 100

bridge-domain 100 mode vlan
  arp-proxy
    agetime 900
  !
device#
```

agetime (neighbor discovery)

Use the `agetime` command to configure the neighbor discovery protocol (NDP) `agetime` on an NDP proxy in a bridge domain.

Syntax

agetime *seconds*

no agetime

Command Default

The `agetime` on an NDP proxy is set to 1200 seconds.

Parameters

seconds

Number of seconds.

Modes

Bridge domain VLAN neighbor discovery proxy configuration (config-vlan-bd-*name*-nd-proxy)

Usage Guidelines

To return the NDP `agetime` setting to the default value, use the **no** form of this command.

Examples

The following example configures an `agetime` of 900 seconds on the NDP proxy in a bridge domain with 100 as its ID:

```
device# configure terminal
device(config)# bridge-domain 100 mode vlan
device(config-vlan-bd-100)# nd-proxy
device(config-vlan-bd-100-nd-proxy)# agetime 900
device(config-vlan-bd-100-nd-proxy)#
```

The following example displays the configuration for bridge domain 100 that is running currently on the device. In this example, an `agetime` of 900 seconds is configured on the NDP proxy in the bridge domain:

```
device# show running-config bridge-domain 100

bridge-domain 100 mode vlan
    nd-proxy
        agetime 900
    !
device#
```

allow-own-as

Use the **allow-own-as** command to set how many times the router's own AS number (local AS) can appear in the AS path of a BGP update from peers of this peer group before being rejected or marked as invalid.

Syntax

allow-own-as *number-of-occurrences*

no allow-own-as *number-of-occurrences*

Command Default

A BGP update received with a local AS appearing at least once in the AS path of that update is marked as invalid.

Parameters

number-of-occurrences

Number of allowed occurrences. Specify an integer from 1 to 10.

Modes

VRF BGP peer group configuration (config-vrf-bgp-pg)

Usage Guidelines

This command lets you specify how many times the router's own AS number can appear in the AS path of a received BGP update from a peer before that update is rejected or marked as invalid.

Use the **no** form of this command to reset the *allow-own-as* number to 0 (the command default) for the specified address family.

Examples

The following example configures a BGP peer group named group1 under a Virtual Routing and Forwarding (VRF) instance named violet. A BGP session within the peer group is reset when the local AS number appears in that session for the 11th time:

```
device# configure terminal
device(config)# vrf violet
device(config-vrf-violet)# router bgp
device(config-vrf-bgp)# peer-group group1
device(config-vrf-bgp-pg)# allow-own-as 10
device(config-vrf-bgp-pg)#
```

The following example displays the configuration of a VRF instance named violet that is running currently on the device. In this example, any BGP session in the peer group can encounter the local AS number up to 10 times before being reset:

```
device# show running-config vrf violet

vrf violet
!
  router bgp
    peer-group group1
      allow-own-as 10
!
device#
```

arp-proxy

Use the **arp-proxy** command to enter Bridge domain VLAN ARP proxy configuration (config-vlan-bd-*name*-arp-proxy) mode.

Syntax

arp-proxy

Parameters

This command has no keywords or variables.

Modes

Bridge domain VLAN configuration (config-vlan-bd-*name*)

Usage Guidelines

This command enters the mode for Address Resolution Protocol (ARP) proxy configuration in a bridge domain. This configuration includes agetime, snooping, and ARP suppression.

Examples

The following example enters Bridge domain VLAN ARP proxy configuration (config-vlan-bd-*name*-arp-proxy) mode:

```
device# configure terminal
device(config)# bridge-domain 100 mode vlan
device(config-vlan-bd-100)# arp-proxy
device(config-vlan-bd-100-arp-proxy)#
```

The following example displays the configuration for bridge domain 100 that is running currently on the device. In this example, ARP snooping is enabled on the ARP proxy:

```
device# show running-config bridge-domain 100

bridge-domain 100 mode vlan
    arp-proxy
        arp-snooping
    !
device#
```

arp-snooping

Use the **arp-snooping** command to enable address resolution protocol (ARP) snooping on an ARP proxy in a bridge domain.

Syntax

```
arp-snooping  
no arp-snooping
```

Parameters

This command has no keywords or variables.

Modes

Bridge domain VLAN ARP proxy configuration (config-vlan-bd-*name*-arp-proxy)

Usage Guidelines

To disable ARP snooping on an ARP proxy in a bridge domain, use the **no** form of this command.

Examples

The following example enables ARP snooping on the ARP proxy in a bridge domain with 100 as its ID:

```
device# configure terminal  
device(config)# bridge-domain 100 mode vlan  
device(config-vlan-bd-100)# arp-proxy  
device(config-vlan-bd-100-arp-proxy)# arp-snooping  
device(config-vlan-bd-100-arp-proxy)#
```

The following example displays the configuration for bridge domain 100 that is running currently on the device. In this example, ARP snooping is enabled on the ARP proxy in the bridge domain:

```
device# show running-config bridge-domain 100  
  
bridge-domain 100 mode vlan  
    arp-proxy  
        arp-snooping  
    !  
device#
```

as-notation

Use the **as-notation** command to configure the display format of BGP autonomous system (AS) numbers for the BGP routing process within a virtual routing and forwarding (VRF) instance.

Syntax

```
as-notation { as-plain | as-dot | as-dot-plus }  
no as-notation { as-plain | as-dot | as-dot-plus }
```

Command Default

The BGP routing process within the current VRF instance uses as-dot notation to display AS numbers.

Parameters

as-plain

Displays both 2-byte AS numbers and 4-byte AS numbers in asplain (decimal) notation. For example, 65526 and 234567 respectively.

as-dot

Displays 2-byte AS numbers in asplain (decimal) notation and 4-byte AS numbers in asdot+ (dot) notation. For example, 65526 and 1.169031 (234567 in decimal notation) respectively.

as-dot-plus

Displays both 2-byte AS numbers and 4-byte AS numbers in asdot+ (dot) notation. For example, 0.65526 (65526 in decimal notation) and 1.169031 (234567 in decimal notation) respectively.

Modes

VRF BGP configuration (config-vrf-bgp)

Usage Guidelines

This command changes the format in which AS numbers are displayed in BGP related **show** commands. For more information about AS number display formats, refer to *RFC 5396, Textual Representation of Autonomous System (AS) Numbers* at <https://www.rfc-editor.org/info/rfc5396>.

Use the **no** form of this command to restore the display format to the default setting.

Examples

The following example configures a BGP routing process under a VRF named violet and changes its AS number display format to asdot+:

```
device# configure terminal
device(config)# vrf violet
device(config-vrf-violet)# router bgp
device(config-vrf-bgp)# as-notation as-dot-plus
device(config-vrf-bgp)#
```

The following example displays the configuration of a VRF instance named violet that is running currently on the device. In this example, the instance contains a BGP routing process where the AS notation is set to asdot+:

```
device# show running-config vrf violet

vrf violet
  member ethernet 0/15
  address-family ipv4 unicast
  address-family ipv6 unicast
  router bgp
    local-as 0.100
    as-notation as-dot-plus
  !
!
device#
```


as-path-set

Use the **as-path-set** command to configure an BGP Autonomous System (AS) path set and enter Routing policy BGP AS path set configuration (config-route-policy-bgp-set-as-path-set-*name*) mode.

Syntax

as-path-set *set-name*

no as-path-set *set-name*

Command Default

No BGP AS path sets are configured on the device.

Parameters

set-name

Name of the BGP AS path set to configure. Use any combination of uppercase or lowercase alphabetic characters, numerals, underscores, or hyphens. You must use an alphabetic character as the first character.

Modes

Routing policy configuration (config-route-policy)

Usage Guidelines

Use this command to create a named BGP AS path set and enter the mode for configuring the AS path set.

To remove a BGP AS path set from the device, use the **no** form of this command.

Examples

The following example creates a BGP AS path set named aspath1:

```
device# configure terminal
device(config)# route-policy
device(config-route-policy)# bgp-defined-sets
device(config-route-policy-bgp-set)# as-path-set aspath1
device(config-route-policy-bgp-set-as-path-set-aspath1)# member 100
device(config-route-policy-bgp-set-as-path-set-aspath1)#
```

The following example displays the routing policy configuration that is running currently on the device. In this example, the device is configured with one BGP defined set, which is an AS path set named aspath1:

```
device# show running-config route-policy
```

```
route-policy
  bgp-defined-sets
    as-path-set aspath1
      member 100
device#
```

auth-password

Use the **auth-password** command to specify a password for MD5 authentication to be used by all members of the specified BGP peer group within a Virtual Routing and Forwarding (VRF) instance.

Syntax

auth-password *password-string*

no auth-password *password-string*

Command Default

No MD5 authentication is enabled for members of the specified BGP peer group.

Parameters

password-string

Password string for MD5 authentication. Use 1 to 80 characters.

Modes

VRF BGP peer group configuration (config-vrf-bgp-pg)

Usage Guidelines

Use the **no** form of this command to disable MD5 password authentication for all peer group members.

Examples

The following example configures a BGP peer group named `group1` under a VRF instance named `violet`. The **auth-password** command sets an MD5 password for the members of the `group1` peer group:

```
device# configure terminal
device(config)# vrf violet
device(config-vrf-violet)# router bgp
device(config-vrf-bgp)# peer-group group1
device(config-vrf-bgp-pg)# auth-password MyPassword1
device(config-vrf-bgp-pg)#
```

The following example displays the configuration of a VRF instance named `violet` that is running currently on the device. In this example, MD5 authentication is enabled, and an MD5 password is specified for all `group1` peer group members:

```
device# show running-config vrf violet

vrf violet
!
router bgp
```

```
peer-group group1
  auth-password 9f86d081884c7d659a2feaa0c55ad015a3bf4f1b2b0b822cd15d6c15b0f00a08
!
device#
```

auto-offset

Use the **auto-offset** command to set the automatic offset for a specific Virtual Network Identifier (VNI) domain to a non-default value.

Syntax

auto-offset *offset*

no auto-offset

Command Default

The automatic offset for the VNI domain is preset to 0.

Parameters

offset

Automatic offset value. The range is an integer from 0 to 16744447.

Modes

Overlay network VNI domain configuration (config-vni-domain-name)

Usage Guidelines

Use the no form of this command to reset the automatic offset to the default value.

Examples

The following example sets the automatic offset for a VNI named DG1 to 500,000:

```
device# configure terminal
device(config)# overlay-networks
device(config-overlay)# vni-domain DG1
device(config-vni-DG1)# auto-offset 500000
device(config-vni-DG1)#
```

The following example shows the Network Virtualization Overlay (NVO) instances and VNI domains with auto-offset configured on the device:

```
device# show running-config
overlay-networks
  default-nve base
  default-vni-offset 0
  nvo Domain2
    member nve domain2
  !
  nvo base
    member nve base
  !
  nve domain2
    source-interface loopback 1
    vni-domain DG1
```

```
!  
nve base  
    source-interface loopback 1  
    vni-domain base  
!  
vni-domain base  
vni-domain DG1  
    auto-offset 500000  
!  
!  
device#
```

auto-boot

Use the **auto-boot** command to start a virtual machine (VM) automatically after the host reboots.

Syntax

auto-boot

no auto-boot

Command Default

Auto-boot is disabled on the current VM.

Parameters

This command has no keywords or variables.

Modes

IAH VM configuration (config-iah-vm-*name*)

Usage Guidelines

Use the **no** form of this command to disable auto-boot for a VM. Note that configuration is not allowed when the VM is in the deployed state.

To display details about a specific VM, see [show iah vm](#) on page 1006.

Examples

The following example enables auto-boot for a VM named Ubuntu1 on the device:

```
device# configure terminal
device(config)# iah
device(config-iah)# vm Ubuntu1
device(config-iah-vm-Ubuntu1)# auto-boot
device(config-iah-vm-Ubuntu1)#
```

The following example displays the IAH configuration that is running currently on the device. In this example, auto-boot is enabled for a VM named Ubuntu1:

```
device# show running-config iah

iah
 vm Ubuntu1
  description Ubuntu demo vm
  auto-boot true
  cpus 1
  ram 1024
  disk sda
    size 5000
```

```
format qcow2
bus sata
file disk://iah/disk1.qcow2
!
!
!
device#
```


autoconfiguration (IPv6 router advertisement prefix configuration)

Use the **autoconfiguration** command to enable autoconfiguration within a prefix for IPv6 router advertisement on an interface.

Syntax

```
autoconfiguration  
no autoconfiguration
```

Command Default

Autoconfiguration within the current prefix is enabled.

Parameters

This command has no keywords or variables.

Modes

IPv6 router advertisement configuration (config-if-*name*-ipv6-router-advertisement)

Usage Guidelines

Router advertisements are ICMPv6 messages sent by an IPv6 enabled router that are used by hosts to detect routers on an attached link. This functionality is similar to the IPv4 ICMP Router Discovery protocol.

Neighbor discovery defines five ICMP packet types: router solicitation and advertisement, neighbor solicitation and advertisement, and redirect messages.

Router advertisements carry information such as the router link layer address, prefixes supported in the network, and the MTU setting. Router advertisements are periodic messages or are also sent in response to router solicitations. Because of the nature of its functionality, router advertisement is considered as a separate feature from neighbor discovery. Two messages are supported as part of this functionality: router solicitation and router advertisement.

Each configured interface sends a router advertisement message periodically to the all-nodes link-local multicast address (FF02::1). To avoid synchronization of advertisements from other routers, the time interval is randomized to send these messages between the configured maximum and minimum time periods.

Router advertisement includes the following information:

- Default router—Whether the sender can be a default router and, if so, the lifetime for the router
- List of prefixes for on-link addresses

- Auto-configuration information
- Network information—Hop limit and MTU setting to use

To disable autoconfiguration within a prefix for IPv6 router advertisement on an interface, use the **no** form of this command. When autoconfiguration is disabled, the prefix will not be used for stateless address configuration. This is achieved by setting the autonomous address configuration bit for the prefix.

Examples

The following example configures a prefix of 100::1/64 for IPv6 router advertisement on Ethernet interface 0/1 and enables autoconfiguration within that prefix:

```
device# configure terminal
device(config)# interface ethernet 0/1
device(config-if-eth-0/1)# ipv6-router-advertisement
device(config-if-eth-0/1-ipv6-router-advertisement)# prefix 100::1/64
device(config-if-ipv6-router-advertisement-prefix-100::1/64)# autoconfiguration
device(config-if-ipv6-router-advertisement-prefix-100::1/64)#
```

The following example displays the configuration of Ethernet interface 0/1 that is running currently on the device. In this example, autoconfiguration is enabled within prefix 100::1/64 on the interface:

```
device# show running-config interface ethernet 0/1

!
interface ethernet 0/1
  ipv6 address 1111::10/64
  ipv6 neighbor 1111::1 00:04:96:eb:c4:51
  ipv6-router-advertisement
    hop-limit 133
    lifetime 200
    managed-config-flag
    mode disable-unsolicited-ra
    other-config
    enable
    interval 100 min 50
    prefix 100::1/63
    prefix 100::1/64
      advertisement
      autoconfiguration
      valid-lifetime 200
  !
  no shutdown
!
device#
```

auto-negotiate (Extreme ONE OS Switching)

Use the **auto-negotiate** command to enable an interface to negotiate transmission parameters automatically with its peer interface on copper links.

Syntax

```
auto-negotiate  
no auto-negotiate
```

Command Default

The specified interface auto-negotiates optimal transmission parameters with its peer interface.

Parameters

This command has no keywords or variables.

Modes

Ethernet interface configuration (config-if-eth-*name*)

Internal interface configuration (config-if-int-*name*)

Usage Guidelines

Auto-negotiation automatically tunes the Serializer/Deserializer (SerDes) transmit and receive parameters on an interface, which lets it negotiate transmission parameters with its peer interface. With auto-negotiation disabled, the transmission parameters use static (manual) configuration.

. Auto-negotiation always advertises forwarding error correction (FEC) as part of the auto-negotiation protocol exchange. Both peers agree on the best available mode.

Speed and duplex are auto-negotiated depending on the context. Auto-negotiation applies to the 1Gb copper management interface as well, as it negotiates between 10Mb, 100Mb, 1Gb, and half- or full-duplex; this is handled in the speed command for that interface. However, this command applies only to 25Gb, 100Gb, and 400Gb interfaces.

Devices that allow multiple transmission rates, multiple duplex modes (half duplex and full duplex), and several transmission standards at the same time can employ auto-negotiation (though in practice only one standard at each speed is widely supported). Each device announces its technological capabilities, or potential modes of operation, during auto-negotiation. The optimal common mode is selected, with full duplex preferred over half duplex at the same speed and faster speed preferred over lower speed.

Auto-negotiation must be enabled on both sides to take effect. Otherwise, the link will not come up, or the link will flap (meaning that it will fail and then recover).

Use the **no** form of this command to disable auto-negotiation.

Examples

The following example enables auto-negotiation on Ethernet interface 0/1:

```
device# configure terminal
device(config)# interface ethernet 0/1
device(config-if-eth 0/1)# no shutdown
device(config-if-eth 0/1)# auto-negotiate
device#
```

The following example displays the statistics for Ethernet interface 0/2. In this example, the Auto Negotiation field in the output shows that auto-negotiation is enabled on the interface:

```
device# show interface ethernet 0/32

ethernet 0/32 Admin state UP, Oper state DOWN
Interface index is 16778240 (0x1000400)
MTU 9216 bytes
Hardware is Ethernet  mac address 50:0a:9c:97:30:3c
Current Speed  400G
Description: Ethernet 0/32
FEC:
    Mode: RS-FEC
    Uncorrected Blocks: 18
    Corrected Blocks: 55475033
Link Fault Signaling: ON
Link Fault Status: No Fault
Auto Negotiation: ON
Carrier Transitions: 8
    LastClear: 0s
Input:
    Total pkts: 88
    Broadcast pkts: 0
    Discard pkts: 0
    Errors pkts: 0
    FCS Errors: 0
    MCast pkts: 88
    Octets: 15312
    UCast pkts: 0
    Runt pkts: 0
    CRC Errors: 0
Input Distribution:
    64 byte pkts: 0
    65-127 byte pkts: 0
    128-255 byte pkts: 88
    256-511 byte pkts: 0
    512-1023 byte pkts: 0
    1024-1518 byte pkts: 0
    Jumbo pkts: 0
Out:
    Total pkts: 94
    Broadcast pkts: 0
    Discard pkts: 0
    Errors pkts: 0
    MCast pkts: 94
```

```
Octets: 15134
UCast pkts: 0
Rate Info:
      Input: 0.000000 Mbits/sec, 0 pkts/sec 0.00% of line-rate
      Output: 0.000000 Mbits/sec, 0 pkts/sec 0.00% of line-rate
device#
```

The following example displays the configuration of Ethernet interface 0/3 that is running currently on the device. In this example, the **auto-negotiate** command is absent, which shows that auto-negotiation is disabled on the interface:

```
device# show running-config interface ethernet 0/3

interface ethernet 0/3
  no shutdown
!
device#
```

The following example displays the configuration of Ethernet interface 0/3 that is running currently on the device after auto-negotiation is enabled on the interface:

```
device# show running-config interface ethernet 0/3

interface ethernet 0/3
  auto-negotiation
  no shutdown
!
device#
```

bfd

Use the **bfd** command to enter BFD configuration (config-bfd) mode.

Syntax

bfd

Parameters

This command has no keywords or variables.

Modes

Global configuration (config)

Usage Guidelines

BFD is a protocol for quickly detecting failures between two adjacent forwarding engines, including the interfaces and data link(s). After it is set up, you can configure ("register") protocols (such as OSPF or BGP) or static routes with BFD, which makes them BFD clients (meaning that they use BFD for link failure detection instead of their own mechanisms). When the link fails, BFD informs the client application or static route, which then takes action (such as failover to a different route) on the failure.

This mode lets you access the commands for creating BFD profiles or configuring member interfaces.

Examples

The following example enters BFD configuration mode:

```
device# configure terminal
device(config)# bfd
device(config-bfd)#
```

bgp-actions (routing policy statement actions configuration)

Use the **bgp-actions** command to enter Routing policy statement BGP actions configuration (config-route-policy-policy-*name*-stmt-*number*-actions-bgp-actions) mode.

Syntax

```
bgp-actions  
no bgp-actions
```

Command Default

The specified routing policy statement contains no BGP actions.

Parameters

This command has no keywords or variables.

Modes

Routing policy statement actions configuration (config-route-policy-policy-*name*-stmt-*number*-actions)

Usage Guidelines

This command enters the mode for adding BGP actions to a routing policy statement. You can create BGP actions for AS path prepending, communities, local preferences, and Multi-Exit Discriminator (MED) values.

Each statement in a routing policy can have only one BGP actions container. This container can contain multiple actions.

To remove all BGP actions from a routing policy statement, use the **no** form of this command.

Examples

The following example creates a routing policy named map1 that contains a policy statement with sequence number 10 and enters Routing policy statement BGP actions configuration (config-route-policy-policy-*name*-stmt-*number*-actions-bgp-actions) mode:

```
device# configure terminal  
device(config)# route-policy  
device(config-route-policy)# policy map1  
device(config-route-policy-policy-map1)# statement 10  
device(config-route-policy-policy-map1-stmt-10)# actions
```

```
device(config-route-policy-policy-map1-stmt-10-actions)# bgp-actions
device(config-route-policy-policy-map1-stmt-10-actions-bgp-actions)#
```

The following example displays the routing policy configuration that is running currently on the device. In this example, a routing policy named map1 contains a policy statement with sequence number 10 containing a BGP action to prepend local AS number 110 five times to the AS path:

```
device# show running-config route-policy

route-policy
  policy map1
    statement 10
      actions
        bgp-actions
          set-as-path-prepend 110 5
device#
```


bgp-conditions (routing policy statement conditions configuration)

Use the **bgp-conditions** command to enter Routing policy statement BGP conditions configuration (config-route-policy-policy-*name*-stmt-*number*-conditions-bgp-conditions) mode.

Syntax

bgp-conditions

no bgp-conditions

Command Default

The specified routing policy statement contains no BGP match conditions.

Parameters

This command has no keywords or variables.

Modes

Routing policy statement conditions configuration (config-route-policy-policy-*name*-stmt-*number*-conditions)

Usage Guidelines

This command enters the mode for adding a BGP match condition to a routing policy statement. You can create BGP match conditions against local preference values, AS path sets, community sets, extended community sets, and Multi-Exit Discriminator (MED) values. Each statement in a routing policy can have only one BGP actions container. This container can contain multiple actions.

To remove all BGP match conditions from a routing policy statement, use the **no** form of this command.

Examples

The following example creates a routing policy named `map1` that contains a policy statement with sequence number 10 and enters Routing policy statement BGP conditions configuration (config-route-policy-policy-*name*-stmt-*number*-conditions-bgp-conditions) mode:

```
device# configure terminal
device(config)# route-policy
device(config-route-policy)# policy map1
device(config-route-policy-policy-map1)# statement 10
device(config-route-policy-policy-map1-stmt-10)# conditions
device(config-route-policy-policy-map1-stmt-10-conditions)# bgp-conditions
device(config-route-policy-policy-map1-stmt-10-conditions-bgp-conditions)#
```

The following example displays the routing policy configuration that is running currently on the device. In this example, a routing policy named map1 has a policy statement with sequence number 10 containing a condition that matches an AS path set named aspath1:

```
device# show running-config route-policy

route-policy
  policy map1
    statement 10
      conditions
        bgp-conditions
          match-as-path-set aspath1 any
device#
```

bgp-defined-sets

Use the **bgp-defined-sets** command to enter Routing policy BGP defined sets configuration (config-route-policy-bgp-set) mode.

Syntax

```
bgp-defined-sets  
no bgp-defined-sets
```

Command Default

No BGP defined sets are configured on the device.

Parameters

This command has no keywords or variables.

Modes

Routing policy configuration (config-route-policy)

Usage Guidelines

This command enters the mode for accessing the commands used to configure BGP Autonomous System (AS) path sets, BGP community sets, and BGP extended community sets.

To remove the BGP defined sets configuration from the device, use the **no** form of this command.

Examples

The following example enters Routing policy BGP defined sets configuration (config-route-policy-bgp-set) mode and creates an AS path set named aspath1:

```
device# configure terminal  
device(config)# route-policy  
device(config-route-policy)# bgp-defined-sets  
device(config-route-policy-bgp-set)# as-path-set aspath1  
device(config-route-policy-bgp-set-as-path-set-aspath1)# member 100  
device(config-route-policy-bgp-set-as-path-set-aspath1)#
```

The following example displays the routing policy configuration that is running currently on the device. In this example, the device is configured with one BGP defined set, which is an AS path set named aspath1:

```
device# show running-config route-policy  
  
route-policy  
    bgp-defined-sets
```

```
as-path-set aspath1
  member 100
device#
```

bridge-domain

Use the **bridge-domain** command to create the bridge domain with the specified mode.

Syntax

```
bridge-domain ID [ mode vlan ]
```

```
no bridge-domain ID
```

Command Default

No bridge domains are configured on the device.

Parameters

bridge-domain *ID*

Specifies the bridge domain ID. Valid range is 1-8192.

mode *vlan*

(Optional) Specifies the VLAN bridge domain configuration mode. If you do not use this keyword and variable, the default bridge domain configuration mode is used.

Modes

Global configuration (config)

Usage Guidelines

To remove the bridge domain configuration, use the no form of this command.

Examples

The following examples create a default bridge domain and a VLAN-mode bridge domain:

```
device# config terminal
device(config)#

device(config)# bridge-domain 100
device(config-bd-100)#

device(config)# bridge-domain 200 mode vlan
device(config-vlan-bd-200)#

device(config)# show running-config

bridge-domain 100
bridge-domain 200 mode vlan
device(config)#
```

bringup-delay (MLAG peer configuration)

Use the **bringup-delay** command to set the multichassis link aggregation (MLAG) bringup delay.

Syntax

```
bringup-delay [ seconds ]  
no bringup-delay
```

Command Default

The bringup delay is set to 90 seconds.

Parameters

seconds

(Optional) Delay period in seconds. Use an integer from 0 to 1000.

Modes

MLAG configuration (config-mlag)

Usage Guidelines

You can use the **show running-config mlag** command to display the MLAG configuration that is running currently on a device.

To restore the default bringup delay, use the no form of this command.

Examples

The following example sets an MLAG bringup delay of 100 seconds:

```
device# configure terminal  
device(config)# mlag  
device(config-mlag)# bringup-delay 100  
device(config-mlag)#
```

The following example displays the MLAG configuration that is running currently on a device. In this example, the MLAG bringup delay is set to 100 seconds:

```
device# show running-config mlag  
  
mlag  
    bringup-delay 100  
device#
```

bus

Use the **bus** command to configure a bus type for a disk within a virtual machine (VM).

Syntax

```
bus { sata | ide | scsi | virtio }  
no bus
```

Command Default

The disk on the specified VM uses a virtualization I/O (virtio) bus.

Parameters

```
{ sata | ide | scsi | virtio }
```

Bus type.

Modes

IAH VM disk configuration (config-iah-vm-disk-*name*)

Usage Guidelines

Use the **no** form of this command to restore the bus type to the default value. Note that configuration is not allowed when the VM is in the deployed state.

To display details about a specific VM, see [show iah vm](#) on page 1006.

Examples

The following example specifies an SATA bus type for a disk named sda within a VM named Ubuntu1:

```
device# configure terminal  
device(config)# iah  
device(config-iah)# vm Ubuntu1  
device(config-iah-vm-Ubuntu1)# disk sda  
device(config-iah-vm-disk-sda)# bus sata  
device(config-iah-vm-disk-sda)#
```

The following example displays the IAH configuration that is running currently on the device. In this example, a VM named Ubuntu2 uses a SCSI bus on a disk named sdb:

```
device# show running-config iah  
  
iah  
vm Ubuntu2  
description Ubuntu demo vm  
auto-boot true  
cpus 4  
ram 1024
```

```
disk sdb
  size 5000
  format qcow2
  bus scsi
  file disk://iah/disk1.qcow2
!
!
!
device#
```


channel-group

Use the **channel-group** command to enable link aggregation on an interface and add it to a link aggregation group (LAG).

Syntax

```
channel-group port-channel-number mode { active | passive | on }  
no channel-group
```

Command Default

The specified interface is not a member of a LAG.

Parameters

port-channel-number

Port channel (LAG) number to which the interface is added. The port channel number is in the range of 1 to 255.

mode

LAG mode of the interface.

active

Enables Link Aggregation Control Protocol (LACP) unconditionally. This mode places a port into an active negotiating state where the port initiates negotiations with other ports by sending LACP packets.

passive

Enables LACP only when an LACP device is detected. This mode places a port into a passive negotiating state where the port responds to LACP packets that it receives but does not initiate protocol negotiation.

on

Enables static link aggregation on the interface. A port in this mode is not running LACP and cannot form a port channel with another port that is in active or passive mode.

Modes

Ethernet interface configuration (*config-if-eth-name*)

Usage Guidelines

Before using this command, you must enable LACP globally. For details, see [protocol lacp](#) on page 885.

To remove an interface from a LAG, use the no form of this command.

Examples

The following example adds Ethernet interface 0/29 to a LAG that has a port channel number of 10 and specifies active LACP mode on the interface:

```
device# configure terminal
device(config)# interface ethernet 0/29
(config-if-eth-0/29)# channel-group 10 mode active
(config-if-eth-0/29)#
```

The following example displays the LAG configuration that is running currently on the device. In this example, LACP is enabled globally, a LAG with a port channel number of 10 is configured, and Ethernet interface 0/29 is added to the LAG in active mode:

```
device# show running-config lacp

protocol lacp
interface port-channel 10
    no shutdown
!
interface ethernet 0/29
    channel-group 10 mode active
    no shutdown
!
device#
```

clear bgp neighbor

Use the **clear bgp neighbor** command to reset BGP neighbor connections using soft or hard reconfiguration.

Syntax

```
clear bgp vrf vrf-name { ipv4 | ipv6 } { unicast | multicast } neighbor  
    { neighbor-ipv4-addr | neighbor-ipv6-addr } [soft-in | soft-out ]
```

Command Default

BGP neighbor connections are not reset, and BGP routes received from or advertised to a neighbor remain unrefreshed.

Parameters

vrf *vrf-name*

Filters routes by VRF instance name. This name must start with an alphabetic character or an underscore followed by an arbitrary sequence of alphabetic or numeric characters, underscores, hyphens, or dots. The name cannot exceed 64 characters. Do not use the profile reserved keyword.

{ **ipv4** | **ipv6** }

Specifies an IPv4 or IPv6 neighbor address.

{ **unicast** | **multicast** }

Clears all routes with either unicast or multicast address prefixes.

neighbor

Clears all routes received from or advertised to neighbors.

{ *neighbor-ipv4-addr* | *neighbor-ipv6-addr* }

IPv4 or IPv6 address of the neighbor.

- For an IPv4 address, use the *a.b.c.d* format. For example, 10.1.1.1.
- For an IPv6 address, use the *x:x::x:x* format. For example, 1001::1.

[**soft-in** | **soft-out**]

(Optional) Initiates a soft reset by dynamically refreshing received routes or sent routes respectively without tearing down (hard resetting) the neighbor session. If you do not specify either option, both inbound and outbound neighbor sessions are torn down, and their received routes and sent routes are deleted.

Modes

EXEC (#)

Usage Guidelines

When soft-in or soft-out are used, this command clears BGP routes received from or advertised to a neighbor and requests a dynamic refresh of BGP connections or routes from a neighbor. Otherwise, sessions are hard reset.

Examples

The following example clears all received IPv4 routes from neighbor 1.1.1.1 within a VRF instance named red but does not tear down the session:

```
device# clear bgp ipv4 unicast neighbor 1.1.1.1 soft-in vrf red
device#
```

The following example clears all sent IPv6 routes advertised to neighbor 1001::1 within a VRF instance named blue but does not tear down the session:

```
device# clear bgp ipv6 multicast neighbor 1001::1 soft-out vrf blue
device#
```

clear bgp peer-group

Use the **clear bgp peer-group** command to clear BGP routes that are received from or advertised to the neighbors belong to a specific peer group for the BGP routing process within a virtual routing and forwarding (VRF) instance.

Syntax

```
clear bgp vrf vrf-name { ipv4 | ipv6 } { unicast | multicast } peer-group peer-group-name [soft-in | soft-out ]
```

Command Default

BGP routes that are received from or advertised to the peer group within the VRF instance remain uncleared.

Parameters

vrf-name

Filters routes by VRF instance name. This name must start with an alphabetic character or an underscore followed by an arbitrary sequence of alphabetic or numeric characters, underscores, hyphens, or dots. The name cannot exceed 64 characters. Do not use the **profile** reserved keyword.

{ **ipv4** | **ipv6** }

Specifies an IPv4 or IPv6 neighbor address belonging to the specified peer group.

{ **unicast** | **multicast** }

Clears all routes with either unicast or multicast address prefixes.

peer-group *peer-group-name*

Name of the peer group for which to clear received or advertised BGP routes.

[**soft-in** | **soft-out**]

(Optional) Initiates a soft reset by dynamically refreshing received routes or sent routes respectively without tearing down (hard resetting) the neighbor session belonging to the specified peer group. If you do not specify either option, both inbound and outbound neighbor sessions belonging to the specified peer group are torn down, and their received routes and sent routes are deleted.

Modes

EXEC (#)

Usage Guidelines

This command clears all BGP routes that pass through a specific peer group. If you do not specify either of the **soft-in** or **soft-out** options, the peer group is reset. All BGP neighbors under this peer group are cleared or reset.



Note

This command performs no validations on its inputs. If inputs are invalid or no matching entry is found, the command is ignored.

Examples

The following example clears certain BGP routes that are received from or advertised to a peer group named group2 within a VRF instance named default-vrf. In this example, the BGP routes to be cleared have unicast address prefixes in the IPv4 address family:

```
device# clear bgp vrf default-vrf ipv4 unicast peer-group group2
device#
```

The following example clears certain BGP routes that are received from or advertised to a peer group named group2 within a VRF instance named default-vrf. In this example, the BGP routes to be cleared have multicast address prefixes in the IPv6 address family:

```
device# clear bgp vrf default-vrf ipv6 multicast peer-group group2
device#
```

The following example clears certain BGP routes that are received from or advertised to a peer group named group3 within a VRF instance named default-vrf. In this example, the BGP routes to be cleared are received routes that have unicast address prefixes in the IPv4 address family:

```
device# clear bgp vrf default-vrf ipv4 unicast peer-group group3 soft-in
device#
```

The following example clears certain BGP routes that are received from or advertised to a peer group named group3 within a VRF instance named default-vrf. In this example, the BGP routes to be cleared are advertised routes that have multicast address prefixes in the IPv6 address family:

```
device# clear bgp vrf default-vrf ipv6 multicast peer-group group3 soft-out
device#
```

clear bgp vrf routes l2vpn-evpn nvo

Use the **clear bgp vrf routes l2vpn-evpn nvo** command to clear all BGP routes from a Layer 2 EVPN table identified by a specified NVO for one or all route distinguishers.

Syntax

```
clear bgp vrf vrf-name routes l2vpn-evpn nvo nvo-name [ route-  
distinguisher { as-number:admin-number-1 | ipv4-address:admin-  
number-2 } ]
```

Command Default

BGP routes from the Layer 2 Ethernet VPN (EVPN) table identified by the specified Network Virtualization Overlay (NVO) remain uncleared.

Parameters

vrf-name

Filters routes by VRF instance name. This name must start with an alphabetic character or an underscore followed by an arbitrary sequence of alphabetic or numeric characters, underscores, hyphens, or dots. The name cannot exceed 64 characters. Do not use the **profile** reserved keyword.

nvo-name

Name of the NVO for which to clear BGP routes.

as-number:admin-number-1

(Optional) Autonomous system (AS) number and administrative number. You must enter the colon (:).

- *as-number* is an assigned AS number. Use an integer that is either 2 bytes or 4 bytes long. The range is 1 to 4294967295. You cannot use a reserved AS number.
- *admin-number-1* is an integer from 1 to 4294967295. If *as-number* is 2-byte, then *admin-number-1* must be 4-byte (an integer from 1 to 4294967295). If *as-number* is 4-byte, then *admin-number-1* must be 2-byte (an integer from 1 to 65535).

ipv4-address:admin-number-2

(Optional) IPv4 address and administrative number. You must enter the colon (:).

- *ipv4-address* is a valid Class A, Class B, or Class C address. Use the *a.b.c.d* format. You cannot use other addresses such as Class D, Class E, multidestination, loopback, or the default address.
- *admin-number-2* is an integer from 1 to 65535.

Modes

EXEC (#)

Usage Guidelines

This command clears all BGP routes of all types from a Layer 2 EVPN table identified by a specified NVO. You can limit the clearing operation to a specific route distinguisher if needed.

Examples

The following example clears all BGP routes from a Layer 2 EVPN table identified by an NVO named mynvo within a VRF instance named default-vrf. In this example, all BGP routes are cleared for all route distinguishers:

```
device# clear bgp vrf default-vrf routes l2vpn-evpn nvo mynvo
device#
```

The following example clears all BGP routes from a Layer 2 EVPN table identified by an NVO named mynvo2 within a VRF instance named default-vrf. In this example, only BGP routes with the 65530:10000 route distinguisher are cleared:

```
device# clear bgp vrf default-vrf routes l2vpn-evpn nvo mynvo2 route-distinguisher
65530:10000
device#
```

The following example clears all BGP routes from a Layer 2 EVPN table identified by an NVO named mynvo3 within a VRF instance named default-vrf. In this example, only BGP routes with the 152.100.1.100:10001 route distinguisher are cleared:

```
device# clear bgp vrf default-vrf routes l2vpn-evpn nvo mynvo3 route-distinguisher
152.100.1.100:10001
device#
```


clear bgp vrf routes l2vpn-evpn nvo route-distinguisher

Use the **clear bgp vrf routes l2vpn-evpn nvo route-distinguisher** command to clear all BGP routes from a Layer 2 EVPN table identified by a specified NVO for a specific route distinguisher.

Syntax

```
clear bgp vrf vrf-name routes l2vpn-evpn nvo nvo-name route-distinguisher  
{ as-number:admin-number-1 | ipv4-address:admin-number-2 }
```

Command Default

BGP routes from the Layer 2 Ethernet VPN (EVPN) table identified by the specified Network Virtualization Overlay (NVO) and route distinguisher remain uncleared.

Parameters

vrf-name

Filters routes by VRF instance name. This name must start with an alphabetic character or an underscore followed by an arbitrary sequence of alphabetic or numeric characters, underscores, hyphens, or dots. The name cannot exceed 64 characters. Do not use the **profile** reserved keyword.

nvo-name

Name of the NVO for which to clear BGP routes.

as-number:admin-number-1

(Optional) Autonomous system (AS) number and administrative number. You must enter the colon (:).

- *as-number* is an assigned AS number. Use an integer that is either 2 bytes or 4 bytes long. The range is 1 to 4294967295. You cannot use a reserved AS number.
- *admin-number-1* is an integer from 1 to 4294967295. If *as-number* is 2-byte, then *admin-number-1* must be 4-byte (an integer from 1 to 4294967295). If *as-number* is 4-byte, then *admin-number-1* must be 2-byte (an integer from 1 to 65535).

ipv4-address:admin-number-2

(Optional) IPv4 address and administrative number. You must enter the colon (:).

- *ipv4-address* is a valid Class A, Class B, or Class C address. Use the *a.b.c.d* format. You cannot use other addresses such as Class D, Class E, multidestination, loopback, or the default address.
- *admin-number-2* is an integer from 1 to 65535.

Modes

EXEC (#)

Usage Guidelines

This command clears all BGP routes of all types from a Layer 2 EVPN table identified by a specified NVO and a specific route distinguisher.

Examples

The following example clears all BGP routes from a Layer 2 EVPN table identified by an NVO named mynvo2 within a VRF instance named default-vrf. In this example, BGP routes with the 65530:10000 route distinguisher are cleared:

```
device# clear bgp vrf default-vrf routes l2vpn-evpn nvo mynvo2 route-distinguisher
65530:10000
device#
```

The following example clears all BGP routes from a Layer 2 EVPN table identified by an NVO named mynvo3 within a VRF instance named default-vrf. In this example, BGP routes with the 152.100.1.100:10001 route distinguisher are cleared:

```
device# clear bgp vrf default-vrf routes l2vpn-evpn nvo mynvo3 route-distinguisher
152.100.1.100:10001
device#
```

clear bgp vrf routes l2vpn-evpn nvo route-distinguisher route-type

Use the **clear bgp vrf routes l2vpn-evpn nvo route-distinguisher route-type** command to clear BGP routes of a specified route distinguisher and route type from a Layer 2 EVPN table that is identified by a specified NVO.

Syntax

```
clear bgp vrf vrf-name routes l2vpn-evpn nvo nvo-name route-distinguisher  
    { as-number:admin-number-1 | ipv4-address:admin-number-2 } route-type  
    { imr | mac | arp | nd | ipv4-prefix | ipv6-prefix }
```

Command Default

BGP routes from the Layer 2 Ethernet VPN (EVPN) table identified by the specified Network Virtualization Overlay (NVO), route distinguisher, and route type remain uncleared.

Parameters

vrf-name

Filters routes by VRF instance name. This name must start with an alphabetic character or an underscore followed by an arbitrary sequence of alphabetic or numeric characters, underscores, hyphens, or dots. The name cannot exceed 64 characters. Do not use the **profile** reserved keyword.

nvo-name

Name of the NVO for which to clear BGP routes.

as-number:admin-number-1

(Optional) Autonomous system (AS) number and administrative number. You must enter the colon (:).

- *as-number* is an assigned AS number. Use an integer that is either 2 bytes or 4 bytes long. The range is 1 to 4294967295. You cannot use a reserved AS number.
- *admin-number-1* is an integer from 1 to 4294967295. If *as-number* is 2-byte, then *admin-number-1* must be 4-byte (an integer from 1 to 4294967295). If *as-number* is 4-byte, then *admin-number-1* must be 2-byte (an integer from 1 to 65535).

ipv4-address:admin-number-2

(Optional) IPv4 address and administrative number. You must enter the colon (:).

- *ipv4-address* is a valid Class A, Class B, or Class C address. Use the *a.b.c.d* format. You cannot use other addresses such as Class D, Class E, multidestination, loopback, or the default address.
- *admin-number-2* is an integer from 1 to 65535.

Type of EVPN route:

- **imr**—Inclusive multicast (EVPN Type 3) routes
- **mac**—MAC/IP (EVPN Type 2) advertisement routes
- **arp**—IPv4 address resolution (ARP) protocol routes
- **nd**—IPv6 network discovery routes
- **ipv4-prefix**—IPv4 prefix (EVPN Type 5) routes
- **ipv6-prefix**—IPv6 prefix (EVPN Type 5) routes

Modes

EXEC (#)

Usage Guidelines

This command clears all BGP routes of a specific route distinguisher and route type from a Layer 2 EVPN table identified by a specified NVO.

Examples

The following example clears all BGP ARP routes from a Layer 2 EVPN table identified by an NVO named `mynvo2` within a VRF instance named `default-vrf`. In this example, only BGP ARP routes with the `65530:10000` route distinguisher are cleared:

```
device# clear bgp vrf default-vrf routes l2vpn-evpn nvo mynvo2 route-distinguisher
65530:10000 route-type arp
device#
```

The following example clears all BGP IPv4 prefix routes from a Layer 2 EVPN table identified by an NVO named `mynvo3` within a VRF instance named `default-vrf`. In this example, only BGP IPv4 prefix routes with the `152.100.1.100:10001` route distinguisher are cleared:

```
device# clear bgp vrf default-vrf routes l2vpn-evpn nvo mynvo3 route-distinguisher
152.100.1.100:10001 route-type ipv4-prefix
device#
```

clear bgp vrf routes l2vpn-evpn nvo route-type

Use the **clear bgp vrf routes l2vpn-evpn nvo route-type** command to clear BGP routes of a specified route type from a Layer 2 EVPN table that is identified by a specified NVO.

Syntax

```
clear bgp vrf vrf-name routes l2vpn-evpn nvo nvo-name route-type { imr | mac | arp | nd | ipv4-prefix | ipv6-prefix }
```

Command Default

BGP routes from the Layer 2 Ethernet VPN (EVPN) table identified by the specified Network Virtualization Overlay (NVO) and route type remain uncleared.

Parameters

vrf-name

Filters routes by VRF instance name. This name must start with an alphabetic character or an underscore followed by an arbitrary sequence of alphabetic or numeric characters, underscores, hyphens, or dots. The name cannot exceed 64 characters. Do not use the **profile** reserved keyword.

nvo-name

Name of the NVO for which to clear BGP routes.

{ **imr** | **mac** | **arp** | **nd** | **ipv4-prefix** | **ipv6-prefix** }

Type of EVPN route:

- **imr**—Inclusive multicast (EVPN Type 3) routes
- **mac**—MAC/IP (EVPN Type 2) advertisement routes
- **arp**—IPv4 address resolution (ARP) protocol routes
- **nd**—IPv6 network discovery routes
- **ipv4-prefix**—IPv4 prefix (EVPN Type 5) routes
- **ipv6-prefix**—IPv6 prefix (EVPN Type 5) routes

Modes

EXEC (#)

Usage Guidelines

This command clears all BGP routes of a specific route type from a Layer 2 EVPN table identified by a specified NVO.

Examples

The following example clears all BGP ARP routes from a Layer 2 EVPN table identified by an NVO named mynvo2 within a VRF instance named default-vrf. In this example, only BGP ARP routes are cleared:

```
device# clear bgp vrf default-vrf routes l2vpn-evpn nvo mynvo2 route-type arp
device#
```

The following example clears all BGP IPv4 prefix routes from a Layer 2 EVPN table identified by an NVO named mynvo3 within a VRF instance named default-vrf. In this example, only BGP IPv4 prefix routes are cleared:

```
device# clear bgp vrf default-vrf routes l2vpn-evpn nvo mynvo3 route-type ipv4-prefix
device#
```

clear bgp vrf routes l2vpn-evpn nvo route-type arp vlan ipv4-prefix mac

Use the **clear bgp vrf routes l2vpn-evpn nvo route-type arp vlan ipv4-prefix mac** command to clear BGP ARP routes from a Layer 2 EVPN table for a specific VLAN, MAC address, and NVO.

Syntax

```
clear bgp vrf vrf-name routes l2vpn-evpn nvo nvo-name [ route-distinguisher { as-number:admin-number-1 | ipv4-address:admin-number-2 } ] route-type arp vlan vlan-id ipv4-prefix ipv4-address mac mac-address
```

Command Default

BGP address resolution protocol (ARP) routes from a Layer 2 EVPN table for the specified Virtual LAN (VLAN), MAC address, and Network Virtualization Overlay (NVO) remain uncleared.

Parameters

vrf-name

Filters routes by VRF instance name. This name must start with an alphabetic character or an underscore followed by an arbitrary sequence of alphabetic or numeric characters, underscores, hyphens, or dots. The name cannot exceed 64 characters. Do not use the **profile** reserved keyword.

nvo-name

Name of the NVO for which to clear BGP routes.

route-distinguisher *as-number* : *admin-number-1*

(Optional) Autonomous system (AS) number and administrative number. You must enter the colon (:).

- *as-number* is an assigned AS number. Use an integer that is either 2 bytes or 4 bytes long. The range is 1 to 4294967295. You cannot use a reserved AS number.
- *admin-number-1* is an integer from 1 to 4294967295. If *as-number* is 2-byte, then *admin-number-1* must be 4-byte (an integer from 1 to 4294967295). If *as-number* is 4-byte, then *admin-number-1* must be 2-byte (an integer from 1 to 65535).

ipv4-address:admin-number-2

(Optional) IPv4 address and administrative number. You must enter the colon (:).

- *ipv4-address* is a valid Class A, Class B, or Class C address. Use the *a.b.c.d* format. You cannot use other addresses such as Class D, Class E, multidestination, loopback, or the default address.
- *admin-number-2* is an integer from 1 to 65535.

vlan *vlan-id*

ID of the VLAN for which to clear routes. Use an integer from 1 to 4096.

ipv4-prefix *ipv4-address*

IPv4 address prefix. Use the *a.b.c.d* format. For example, 1.1.1.1.

mac *mac-address*

MAC address. Use *xx:xx:xx:xx:xx:xx* notation (six groups of two hexadecimal digits separated by colons). For example, 01:23:45:67:89:AB.

Modes

EXEC (#)

Usage Guidelines

This command clears BGP ARP routes from a Layer 2 EVPN table for a VLAN, MAC address, and NVO that you specify. You can limit the clearing operation to a specific route distinguisher.

Examples

The following example clears all BGP ARP routes from a Layer 2 EVPN table identified by an NVO named *mynvo* within a VRF instance named *default-vrf*. In this example, all routes are cleared for VLAN 10 and MAC address 11:11:11:11:11:11:

```
device# clear bgp vrf default-vrf routes l2vpn-evpn nvo mynvo route-type arp vlan 10
ipv4-prefix 1.1.1.1 mac-address 11:11:11:11:11:11
device#
```

The following example clears all BGP ARP routes from a Layer 2 EVPN table identified by an NVO named *mynvo2* within a VRF instance named *default-vrf*. In this example, only routes with the 65530:10000 route distinguisher, VLAN 10, and MAC address 11:11:11:11:11:11 are cleared:

```
device# clear bgp vrf default-vrf routes l2vpn-evpn nvo mynvo2 route-distinguisher
65530:10000 route-type arp vlan 10 ipv4-prefix 1.1.1.1
```

The following example clears all BGP ARP routes from a Layer 2 EVPN table identified by an NVO named *mynvo3* within a VRF instance named *default-vrf*. In this example, only routes with the 152.100.1.100:10001 route distinguisher, VLAN 10, and MAC address 11:11:11:11:11:11 are cleared:

```
device# clear bgp vrf default-vrf l2vpn-evpn nvo mynvo3 route-distinguisher
152.100.1.100:10001 route-type arp vlan 10 ipv4-prefix 1.1.1.1 mac-address
11:11:11:11:11:11
device#
```


clear bgp vrf routes l2vpn-evpn nvo route-type imr vlan originator

Use the **clear bgp vrf routes l2vpn-evpn nvo route-type imr vlan originator** command to clear BGP IMR routes for a specific VLAN and IMR originator from a Layer 2 EVPN table identified by a specified NVO for one or all route distinguishers.

Syntax

```
clear bgp vrf vrf-name routes l2vpn-evpn nvo nvo-name [ route-  
distinguisher { as-number:admin-number-1 | ipv4-address:admin-  
number-2 } ] route-type imr vlan vlan-id originator ipv4-addr
```

Command Default

BGP inclusive multicast routes (IMR) from a Layer 2 EVPN table for the specified Virtual LAN (VLAN), originating device, and Network Virtualization Overlay (NVO) remain uncleared.

Parameters

vrf-name

Filters routes by VRF instance name. This name must start with an alphabetic character or an underscore followed by an arbitrary sequence of alphabetic or numeric characters, underscores, hyphens, or dots. The name cannot exceed 64 characters. Do not use the **profile** reserved keyword.

nvo-name

Name of the NVO for which to clear BGP routes.

as-number:admin-number-1

(Optional) Autonomous system (AS) number and administrative number. You must enter the colon (:).

- *as-number* is an assigned AS number. Use an integer that is either 2 bytes or 4 bytes long. The range is 1 to 4294967295. You cannot use a reserved AS number.
- *admin-number-1* is an integer from 1 to 4294967295. If *as-number* is 2-byte, then *admin-number-1* must be 4-byte (an integer from 1 to 4294967295). If *as-number* is 4-byte, then *admin-number-1* must be 2-byte (an integer from 1 to 65535).

ipv4-address:admin-number-2

(Optional) IPv4 address and administrative number. You must enter the colon (:).

- *ipv4-address* is a valid Class A, Class B, or Class C address. Use the *a.b.c.d* format. You cannot use other addresses such as Class D, Class E, multidestination, loopback, or the default address.
- *admin-number-2* is an integer from 1 to 65535.

vlan *vlan-id*

ID of the VLAN for which to clear routes. Use an integer from 1 to 4096.

originator *ipv4-addr*

IP address of the originating device. This is the originator of the IMR (EVPN Type 3) routes to clear. Use the *a.b.c.d* format. For example, 1.1.1.1.

Modes

EXEC (#)

Usage Guidelines

This command clears BGP IMR routes from a Layer 2 EVPN table for a VLAN, IMR originator, and NVO that you specify. You can limit the clearing operation to a specific route distinguisher if needed.

Examples

The following example clears specific BGP IMR routes from a Layer 2 EVPN table identified by an NVO named *mynvo* within a VRF instance named *default-vrf*. In this example, all routes are cleared for VLAN 10 and an IMR originator with address 1.1.1.1:

```
device# clear bgp vrf default-vrf routes l2vpn-evpn nvo mynvo route-type imr vlan 10  
originator 1.1.1.1  
device#
```

The following example clears specific BGP IMR routes from a Layer 2 EVPN table identified by an NVO named *mynvo2* within a VRF instance named *default-vrf*. In this example, only the routes for VLAN 10 with the 65530:10000 route distinguisher and an IMR originator with address 1.1.1.1 are cleared:

```
device# clear bgp vrf default-vrf routes l2vpn-evpn nvo mynvo2 route-distinguisher  
65530:10000 route-type imr vlan 10 originator 1.1.1.1  
device#
```

The following example clears specific BGP IMR routes from a Layer 2 EVPN table identified by an NVO named *mynvo3* within a VRF instance named *default-vrf*. In this example, only the routes with the 152.100.1.100:10001 route distinguisher, VLAN 10, and an IMR originator with address 1.1.1.1 are cleared:

```
device# clear bgp vrf default-vrf routes l2vpn-evpn nvo mynvo3 route-distinguisher  
152.100.1.100:10001 route-type imr vlan 10 originator 1.1.1.1  
device#
```

clear bgp vrf routes l2vpn-evpn nvo route-type mac vlan mac

Use the **clear bgp vrf routes l2vpn-evpn nvo route-type mac vlan mac** command to clear BGP MAC routes from a Layer 2 EVPN table for a specific VLAN, MAC address, and NVO.

Syntax

```
clear bgp vrf vrf-name routes l2vpn-evpn nvo nvo-name [ route-  
distinguisher { as-number:admin-number-1 | ipv4-address:admin-  
number-2 } ] route-type mac vlan vlan-id mac mac-address
```

Command Default

BGP MAC routes from a Layer 2 EVPN table for the specified Virtual LAN (VLAN), MAC address, and Network Virtualization Overlay (NVO) remain uncleared.

Parameters

vrf-name

Filters routes by VRF instance name. This name must start with an alphabetic character or an underscore followed by an arbitrary sequence of alphabetic or numeric characters, underscores, hyphens, or dots. The name cannot exceed 64 characters. Do not use the **profile** reserved keyword.

nvo-name

Name of the NVO for which to clear BGP routes.

as-number:admin-number-1

(Optional) Autonomous system (AS) number and administrative number. You must enter the colon (:).

- *as-number* is an assigned AS number. Use an integer that is either 2 bytes or 4 bytes long. The range is 1 to 4294967295. You cannot use a reserved AS number.
- *admin-number-1* is an integer from 1 to 4294967295. If *as-number* is 2-byte, then *admin-number-1* must be 4-byte (an integer from 1 to 4294967295). If *as-number* is 4-byte, then *admin-number-1* must be 2-byte (an integer from 1 to 65535).

ipv4-address:admin-number-2

(Optional) IPv4 address and administrative number. You must enter the colon (:).

- *ipv4-address* is a valid Class A, Class B, or Class C address. Use the *a.b.c.d* format. You cannot use other addresses such as Class D, Class E, multidestination, loopback, or the default address.
- *admin-number-2* is an integer from 1 to 65535.

vlan *vlan-id*

ID of the VLAN for which to clear routes. Use an integer from 1 to 4096.

mac *mac-address*

MAC address of the MAC/IP (EVPN Type 2) advertisement routes to clear. Use *xx:xx:xx:xx:xx:xx* notation (six groups of two hexadecimal digits separated by colons). For example, 01:23:45:67:89:AB.

Modes

EXEC (#)

Usage Guidelines

This command clears BGP MAC routes from a Layer 2 EVPN table for a VLAN, MAC address, and NVO that you specify. You can limit the clearing operation to a specific route distinguisher.

Examples

The following example clears all BGP MAC routes from a Layer 2 EVPN table identified by an NVO named *mynvo* within a VRF instance named *default-vrf*. In this example, all routes are cleared for VLAN 10 and MAC address 11:11:11:11:11:11:

```
device# clear bgp vrf default-vrf routes l2vpn-evpn nvo mynvo route-type mac vlan 10
ipv4-prefix 1.1.1.1 mac-address 11:11:11:11:11:11
device#
```

The following example clears all BGP MAC routes from a Layer 2 EVPN table identified by an NVO named *mynvo2* within a VRF instance named *default-vrf*. In this example, only routes with the 65530:10000 route distinguisher, VLAN 10, and MAC address 11:11:11:11:11:11 are cleared:

```
device# clear bgp vrf default-vrf routes l2vpn-evpn nvo mynvo2 route-distinguisher
65530:10000 route-type mac vlan 10 ipv4-prefix 1.1.1.1 mac-address 11:11:11:11:11:11
device#
```

The following example clears all BGP MAC routes from a Layer 2 EVPN table identified by an NVO named *mynvo3* within a VRF instance named *default-vrf*. In this example, only routes with the 152.100.1.100:10001 route distinguisher, VLAN 10, and MAC address 11:11:11:11:11:11 are cleared:

```
device# clear bgp vrf default-vrf routes l2vpn-evpn nvo mynvo3 route-distinguisher
152.100.1.100:10001 route-type mac vlan 10 ipv4-prefix 1.1.1.1 mac-address
11:11:11:11:11:11
device#
```

clear bgp vrf routes l2vpn-evpn nvo route-type nd vlan ipv6-prefix mac

Use the **clear bgp vrf routes l2vpn-evpn nvo route-type nd vlan ipv6-prefix mac** command to clear BGP ND routes from a Layer 2 EVPN table for a specific VLAN, MAC address, and NVO.

Syntax

```
clear bgp vrf vrf-name routes l2vpn-evpn nvo nvo-name [ route-  
distinguisher { as-number:admin-number-1 | ipv6-address:admin-  
number-2 } ] route-type nd vlan vlan-id ipv6-prefix ipv6-address mac  
mac-address
```

Command Default

BGP network discovery (ND) routes from a Layer 2 EVPN table for the specified Virtual LAN (VLAN), MAC address, and Network Virtualization Overlay (NVO) remain uncleared.

Parameters

vrf-name

Filters routes by VRF instance name. This name must start with an alphabetic character or an underscore followed by an arbitrary sequence of alphabetic or numeric characters, underscores, hyphens, or dots. The name cannot exceed 64 characters. Do not use the **profile** reserved keyword.

nvo-name

Name of the NVO for which to clear BGP routes.

as-number:admin-number-1

(Optional) Autonomous system (AS) number and administrative number. You must enter the colon (:).

- *as-number* is an assigned AS number. Use an integer that is either 2 bytes or 4 bytes long. The range is 1 to 4294967295. You cannot use a reserved AS number.
- *admin-number-1* is an integer from 1 to 4294967295. If *as-number* is 2-byte, then *admin-number-1* must be 4-byte (an integer from 1 to 4294967295). If *as-number* is 4-byte, then *admin-number-1* must be 2-byte (an integer from 1 to 65535).

ipv6-address:admin-number-2

(Optional) IPv6 address and administrative number. You must enter the colon (:).

- *ipv6-address* is a valid Class A, Class B, or Class C address. Use the *a.b.c.d* format. You cannot use other addresses such as Class D, Class E, multidestination, loopback, or the default address.
- *admin-number-2* is an integer from 1 to 65535.

vlan *vlan-id*

ID of the VLAN for which to clear routes. Use an integer from 1 to 4096.

ipv6-prefix *ipv6-address*

IPv6 address prefix. Use the *x:x::x:x* format. For example, 1:2::3:4.

mac *mac-address*

MAC address. Use *xx:xx:xx:xx:xx:xx* notation (six groups of two hexadecimal digits separated by colons). For example, 01:23:45:67:89:AB.

Modes

EXEC (#)

Usage Guidelines

This command clears BGP ND routes from a Layer 2 EVPN table for a VLAN, MAC address, and NVO that you specify. You can limit the clearing operation to a specific route distinguisher.

Examples

The following example clears all BGP ND routes from a Layer 2 EVPN table identified by an NVO named *mynvo* within a VRF instance named *default-vrf*. In this example, all routes are cleared for VLAN 10 and MAC address 11:11:11:11:11:11:

```
device# clear bgp vrf default-vrf routes l2vpn-evpn nvo mynvo route-type nd vlan 10 ipv6-  
prefix 1:2::3:4 mac-address 11:11:11:11:11:11  
device#
```

The following example clears all BGP ND routes from a Layer 2 EVPN table identified by an NVO named *mynvo2* within a VRF instance named *default-vrf*. In this example, only routes with the 65530:10000 route distinguisher, VLAN 10, and MAC address 11:11:11:11:11:11 are cleared:

```
device# clear bgp vrf default-vrf routes l2vpn-evpn nvo mynvo2 route-distinguisher  
65530:10000 route-type nd vlan 10 ipv6-prefix 1:2::3:4
```

The following example clears all BGP ND routes from a Layer 2 EVPN table identified by an NVO named *mynvo3* within a VRF instance named *default-vrf*. In this example, only routes with the 152.100.1.100:10001 route distinguisher, VLAN 10, and MAC address 11:11:11:11:11:11 are cleared:

```
device# clear bgp vrf default-vrf routes l2vpn-evpn nvo mynvo3 route-distinguisher  
152.100.1.100:10001 route-type nd vlan 10 ipv6-prefix 1:2::3:4 mac-address  
11:11:11:11:11:11  
device#
```

clear bgp vrf routes l2vpn-evpn nvo route-type prefix vlan ipv4-prefix

Use the **clear bgp vrf routes l2vpn-evpn nvo route-type prefix vlan ipv4-prefix** command to clear BGP IPv4 prefix routes from a Layer 2 EVPN table for a specific VLAN and NVO.

Syntax

```
clear bgp vrf vrf-name routes l2vpn-evpn nvo nvo-name [ route-  
distinguisher { as-number:admin-number-1 | ipv4-address:admin-  
number-2 } ] route-type prefix vlan vlan-id ipv4-prefix ipv4-address
```

Command Default

BGP IPv4 prefix routes from a Layer 2 EVPN table for the specified Virtual LAN (VLAN) and Network Virtualization Overlay (NVO) remain uncleared.

Parameters

vrf-name

Filters routes by VRF instance name. This name must start with an alphabetic character or an underscore followed by an arbitrary sequence of alphabetic or numeric characters, underscores, hyphens, or dots. The name cannot exceed 64 characters. Do not use the **profile** reserved keyword.

nvo-name

Name of the NVO for which to clear BGP routes.

as-number:admin-number-1

(Optional) Autonomous system (AS) number and administrative number. You must enter the colon (:).

- *as-number* is an assigned AS number. Use an integer that is either 2 bytes or 4 bytes long. The range is 1 to 4294967295. You cannot use a reserved AS number.
- *admin-number-1* is an integer from 1 to 4294967295. If *as-number* is 2-byte, then *admin-number-1* must be 4-byte (an integer from 1 to 4294967295). If *as-number* is 4-byte, then *admin-number-1* must be 2-byte (an integer from 1 to 65535).

ipv4-address:admin-number-2

(Optional) IPv4 address and administrative number. You must enter the colon (:).

- *ipv4-address* is a valid Class A, Class B, or Class C address. Use the *a.b.c.d* format. You cannot use other addresses such as Class D, Class E, multidestination, loopback, or the default address.
- *admin-number-2* is an integer from 1 to 65535.

vlan *vlan-id*

ID of the VLAN for which to clear routes. Use an integer from 1 to 4096.

ipv4-prefix *ipv4-address*

IPv4 address prefix. Use the *a.b.c.d* format. For example, 1.1.1.1.

Modes

EXEC (#)

Usage Guidelines

A BGP IPv4 prefix route is an EVPN route type for the advertisement of IPv4 prefixes. This command clears these routes from a Layer 2 EVPN table for a VLAN, IPv4 prefix, and NVO that you specify. You can limit the clearing operation to a specific route distinguisher.

Examples

The following example clears all BGP IPv4 prefix routes from a Layer 2 EVPN table identified by an NVO named *mynvo* within a VRF instance named *default-vrf*. In this example, all routes matching IPv4 prefix 1.1.1.1 are cleared for VLAN 10:

```
device# clear bgp vrf default-vrf routes l2vpn-evpn nvo mynvo route-type prefix vlan 10
ipv4-prefix 1.1.1.1
device#
```

The following example clears all BGP IPv4 prefix routes from a Layer 2 EVPN table identified by an NVO named *mynvo2* within a VRF instance named *default-vrf*. In this example, only routes matching IPv4 prefix 1.1.1.1 and route distinguisher 65530:10000 for VLAN 10 are cleared:

```
device# clear bgp vrf default-vrf routes l2vpn-evpn nvo mynvo2 route-distinguisher
65530:10000 route-type prefix vlan 10 ipv4-prefix 1.1.1.1
device#
```

The following example clears all BGP IPv4 prefix routes from a Layer 2 EVPN table identified by an NVO named *mynvo3* within a VRF instance named *default-vrf*. In this example, only routes matching IPv4 prefix 1.1.1.1 and route distinguisher 152.100.1.100:10001 for VLAN 10 are cleared:

```
device# clear bgp vrf default-vrf routes l2vpn-evpn nvo mynvo3 route-distinguisher
152.100.1.100:10001 route-type prefix vlan 10 ipv4-prefix 1.1.1.1
device#
```


clear bgp vrf routes l2vpn-evpn nvo route-type prefix vlan ipv6-prefix

Use the **clear bgp vrf routes l2vpn-evpn nvo route-type prefix vlan ipv6-prefix**

Syntax

```
clear bgp vrf vrf-name routes l2vpn-evpn nvo nvo-name [ route-  
distinguisher { as-number:admin-number-1 | ipv6-address:admin-  
number-2 } ] route-type prefix vlan vlan-id ipv6-prefix ipv6-address
```

Command Default

BGP IPv6 prefix routes from a Layer 2 EVPN table for the specified Virtual LAN (VLAN) and Network Virtualization Overlay (NVO) remain uncleared.

Parameters

vrf-name

Filters routes by VRF instance name. This name must start with an alphabetic character or an underscore followed by an arbitrary sequence of alphabetic or numeric characters, underscores, hyphens, or dots. The name cannot exceed 64 characters. Do not use the **profile** reserved keyword.

nvo-name

Name of the NVO for which to clear BGP routes.

as-number:admin-number-1

(Optional) Autonomous system (AS) number and administrative number. You must enter the colon (:).

- *as-number* is an assigned AS number. Use an integer that is either 2 bytes or 4 bytes long. The range is 1 to 4294967295. You cannot use a reserved AS number.
- *admin-number-1* is an integer from 1 to 4294967295. If *as-number* is 2-byte, then *admin-number-1* must be 4-byte (an integer from 1 to 4294967295). If *as-number* is 4-byte, then *admin-number-1* must be 2-byte (an integer from 1 to 65535).

ipv6-address:admin-number-2

(Optional) IPv6 address and administrative number. You must enter the colon (:).

- *ipv6-address* is a valid Class A, Class B, or Class C address. Use the *x::x::x* format. For example, 1:2::3:4. You must use a unicast or link-local IPv6 address. You cannot use other addresses such as Class D, Class E, multidestination, loopback, or the default address.
- *admin-number-2* is an integer from 1 to 65535.

vlan *vlan-id*

ID of the VLAN for which to clear routes. Use an integer from 1 to 4096.

ipv6-prefix *ipv6-address*

IPv6 address prefix. Use the *a.b.c.d* format. For example, 5:6::7:8.

Modes

EXEC (#)

Usage Guidelines

A BGP IPv6 prefix route is an EVPN route type for the advertisement of IPv6 prefixes. This command clears these routes from a Layer 2 EVPN table for a VLAN, IPv6 prefix, and NVO that you specify. You can limit the clearing operation to a specific route distinguisher.

Examples

The following example clears all BGP IPv6 prefix routes from a Layer 2 EVPN table identified by an NVO named *mynvo* within a VRF instance named *default-vrf*. In this example, all routes matching IPv6 prefix 1:2::3:4 are cleared for VLAN 10:

```
device# clear bgp vrf default-vrf routes l2vpn-evpn nvo mynvo route-type prefix vlan 10
ipv6-prefix 1:2::3:4
device#
```

The following example clears all BGP IPv6 prefix routes from a Layer 2 EVPN table identified by an NVO named *mynvo2* within a VRF instance named *default-vrf*. In this example, only routes matching IPv6 prefix 1:2::3:4 and route distinguisher 65530:10000 for VLAN 10 are cleared:

```
device# clear bgp vrf default-vrf routes l2vpn-evpn nvo mynvo2 route-distinguisher
65530:10000 route-type prefix vlan 10 ipv6-prefix 1:2::3:4
device#
```

The following example clears all BGP IPv6 prefix routes from a Layer 2 EVPN table identified by an NVO named *mynvo3* within a VRF instance named *default-vrf*. In this example, only routes matching IPv6 prefix 1:2::3:4 and route distinguisher 1111:12 for VLAN 10 are cleared:

```
device# clear bgp vrf default-vrf routes l2vpn-evpn nvo mynvo3 route-distinguisher
1111:12 route-type prefix vlan 10 ipv6-prefix 1:2::3:4
device#
```

clear bgp vrf routes mac-vrf bridge-domain

Use the **clear bgp vrf routes mac-vrf bridge-domain** command to clear all BGP routes from a MAC VRF table identified by a specified bridge domain.

Syntax

```
clear bgp vrf vrf-name routes mac-vrf bridge-domain bridge-domain-name
```

Command Default

BGP routes from the MAC VRF table identified by the specified bridge domain remain uncleared.

Parameters

vrf-name

Filters routes by VRF instance name. This name must start with an alphabetic character or an underscore followed by an arbitrary sequence of alphabetic or numeric characters, underscores, hyphens, or dots. The name cannot exceed 64 characters. Do not use the **profile** reserved keyword.

bridge-domain-name

Name of the bridge domain for which to clear routes.

Modes

EXEC (#)

Usage Guidelines

This command clears BGP routes from a MAC VRF table identified by a specified bridge domain.

Examples

The following example clears all BGP routes from a MAC VRF table for bridge domain 100 within a VRF instance named default-vrf:

```
device# clear bgp vrf default-vrf routes mac-vrf bridge-domain 100
```

clear bgp vrf routes mac-vrf bridge-domain route-type arp

Use the **clear bgp vrf routes mac-vrf bridge-domain route-type arp** command to clear BGP ARP routes from a MAC VRF table identified by a specified bridge domain.

Syntax

```
clear bgp vrf vrf-name routes mac-vrf bridge-domain bridge-domain-name  
route-type arp [ vlan vlan-id ipv4-prefix ipv4-address mac-address  
mac-address ]
```

Command Default

BGP routes from the MAC VRF table identified by the specified bridge domain and ARP routes remain uncleared.

Parameters

vrf-name

Filters routes by VRF instance name. This name must start with an alphabetic character or an underscore followed by an arbitrary sequence of alphabetic or numeric characters, underscores, hyphens, or dots. The name cannot exceed 64 characters. Do not use the **profile** reserved keyword.

bridge-domain-name

Name of the bridge domain for which to clear routes.

vlan *vlan-id*

(Optional) ID of the VLAN for which to clear routes. Use an integer from 1 to 4096.

ipv4-prefix *ipv4-address*

(Optional) IPv4 address prefix. Use the a.b.c.d format. For example, 1.1.1.1.

mac-address *mac-address*

(Optional) MAC address of the specific ARP routes to clear. Use xx:xx:xx:xx:xx:xx notation (six groups of two hexadecimal digits separated by colons). For example, 01:23:45:67:89:AB.

Modes

EXEC (#)

Usage Guidelines

This command clears BGP ARP routes from a MAC VRF table identified by a specified bridge domain and (optionally) a specific VLAN, specific IPv4 prefix, and specific MAC address.

Examples

The following example clears all BGP ARP routes from a MAC VRF table for bridge domain 100 within a VRF instance named default-vrf:

```
device# clear bgp vrf default-vrf routes mac-vrf bridge-domain 100 route-type arp
device#
```

The following example clears specific BGP ARP routes from a MAC VRF table for bridge domain 100 within a VRF instance named default-vrf. In this example, all routes are cleared for VLAN 10, IPv4 prefix 1.1.1.1, and MAC address 00:00:5E:00:53:00:

```
device# clear bgp vrf default-vrf routes mac-vrf bridge-domain 100 route-type arp vlan 10
ipv4-prefix 1.1.1.1 mac-address 00:00:5E:00:53:00
device#
```

clear bgp vrf routes mac-vrf bridge-domain route-type imr

Use the **clear bgp vrf routes mac-vrf bridge-domain route-type imr** command to clear all BGP IMR routes from a MAC VRF table identified by a specified bridge domain.

Syntax

```
clear bgp vrf vrf-name routes mac-vrf bridge-domain bridge-domain-name  
route-type imr [ vlan vlan-id originator ip-address ]
```

Command Default

BGP routes from the MAC VRF table identified by the specified bridge domain and IMR routes remain uncleared.

Parameters

vrf-name

Filters routes by VRF instance name. This name must start with an alphabetic character or an underscore followed by an arbitrary sequence of alphabetic or numeric characters, underscores, hyphens, or dots. The name cannot exceed 64 characters. Do not use the **profile** reserved keyword.

bridge-domain-name

Name of the bridge domain for which to clear routes.

vlan *vlan-id*

(Optional) ID of the VLAN for which to clear routes. Use an integer from 1 to 4096.

originator *ip-address*

(Optional) IP address of the originating device. This is the originator of the inclusive multicast (EVPN Type 3) routes (IMR) to clear. Use the *a.b.c.d* format. For example, 1.1.1.1..

Modes

EXEC (#)

Usage Guidelines

This command clears BGP IMR routes from a MAC VRF table identified by a specified bridge domain and (optionally) a specified VLAN with IMR originator.

Examples

The following example clears all BGP IMR routes from a MAC VRF table for bridge domain 100 within a VRF instance named default-vrf:

```
device# clear bgp vrf default-vrf routes mac-vrf bridge-domain 100 route-type imr
device#
```

The following example clears specific BGP IMR routes from a MAC VRF table for bridge domain 100 within a VRF instance named default-vrf. In this example, all routes are cleared for VLAN 10 and an IMR originator with IPv4 address 1.1.1.1:

```
device# clear bgp vrf default-vrf routes mac-vrf bridge-domain 100 route-type imr vlan 10
originator 1.1.1.1
device#
```

clear bgp vrf routes mac-vrf bridge-domain route-type mac

Use the **clear bgp vrf routes mac-vrf bridge-domain route-type mac** command to clear BGP MAC routes from a MAC VRF table identified by a specified bridge domain.

Syntax

```
clear bgp vrf vrf-name routes mac-vrf bridge-domain bridge-domain-name  
route-type mac [ vlan vlan-id mac-address mac-address ]
```

Command Default

BGP routes from the MAC VRF table identified by the specified bridge domain and MAC routes remain uncleared.

Parameters

vrf-name

Filters routes by VRF instance name. This name must start with an alphabetic character or an underscore followed by an arbitrary sequence of alphabetic or numeric characters, underscores, hyphens, or dots. The name cannot exceed 64 characters. Do not use the **profile** reserved keyword.

bridge-domain-name

Name of the bridge domain for which to clear routes.

vlan *vlan-id*

(Optional) ID of the VLAN for which to clear routes. Use an integer from 1 to 4096.

mac-address *mac-address*

(Optional) MAC address of the specific MAC routes to clear. Use *xx:xx:xx:xx:xx:xx* notation (six groups of two hexadecimal digits separated by colons). For example, 01:23:45:67:89:AB.

Modes

EXEC (#)

Usage Guidelines

This command clears BGP MAC routes from a MAC VRF table identified by a specified bridge domain and (optionally) a specific VLAN and specific MAC address.

Examples

The following example clears all BGP MAC routes from a MAC VRF table for bridge domain 100 within a VRF instance named default-vrf:

```
device# clear bgp vrf default-vrf routes mac-vrf bridge-domain 100 route-type mac
device#
```

The following example clears specific BGP MAC routes from a MAC VRF table for bridge domain 100 within a VRF instance named default-vrf. In this example, all routes are cleared for VLAN 10 and MAC address 00:00:5E:00:53:00:

```
device# clear bgp vrf default-vrf routes mac-vrf bridge-domain 100 route-type mac vlan 10
mac-address 00:00:5E:00:53:00
device#
```

clear bgp vrf routes mac-vrf bridge-domain route-type nd

Use the **clear bgp vrf routes mac-vrf bridge-domain route-type nd** command to clear BGP network discovery (ND) routes from a MAC VRF table identified by a specified bridge domain.

Syntax

```
clear bgp vrf vrf-name routes mac-vrf bridge-domain bridge-domain-name  
route-type nd [ vlan vlan-id ipv6-prefix ipv6-address mac-address  
mac-address ]
```

Command Default

BGP routes from the MAC VRF table identified by the specified bridge domain and ND routes remain uncleared.

Parameters

vrf-name

Filters routes by VRF instance name. This name must start with an alphabetic character or an underscore followed by an arbitrary sequence of alphabetic or numeric characters, underscores, hyphens, or dots. The name cannot exceed 64 characters. Do not use the **profile** reserved keyword.

bridge-domain-name

Name of the bridge domain for which to clear routes.

vlan *vlan-id*

(Optional) ID of the VLAN for which to clear routes. Use an integer from 1 to 4096.

ipv6-prefix *ipv4-address*

(Optional) IPv6 address prefix. Use the X:X::X:X format. For example, 2001:db8:85a3::8a2e:370:7334.

mac-address *mac-address*

(Optional) MAC address of the specific ND routes to clear. Use xx:xx:xx:xx:xx:xx notation (six groups of two hexadecimal digits separated by colons). For example, 01:23:45:67:89:AB.

Modes

EXEC (#)

Usage Guidelines

This command clears BGP ND routes from a MAC VRF table identified by a specified bridge domain and (optionally) a specific VLAN, IPv6 prefix, and specific MAC address.

Examples

The following example clears all BGP ND routes from a MAC VRF table for bridge domain 100 within a VRF instance named default-vrf:

```
device# clear bgp vrf default-vrf routes mac-vrf bridge-domain 100 route-type nd
device#
```

The following example clears specific BGP ND routes from a MAC VRF table for bridge domain 100 within a VRF instance named default-vrf. In this example, all routes are cleared for VLAN 10, IPv6 prefix 2001:db8:85a3::8a2e:370:7334, and MAC address 00:00:5E:00:53:00:

```
device# clear bgp vrf default-vrf routes mac-vrf bridge-domain 100 route-type nd vlan 10
ipv6-prefix 2001:db8:85a3::8a2e:370:7334 mac-address 00:00:5E:00:53:00
device#
```

clear counters access-list

Use the **clear counters access-list** command to clear statistics for IP access control lists (ACLs), IPv6 ACLs, or MAC ACLs.

Syntax

```
clear counters { ipv4 | ipv6 | mac } access-list { access-list-name | all }  
clear counters access-list all
```

Command Default

Any existing ACL statistics remain uncleared.

Parameters

ipv4

Clears statistics for all IPv4 ACLs or a specified IPv4 ACL.

ipv6

Clears statistics for all IPv6 ACLs or a specified IPv6 ACL.

mac

Clears statistics for all MAC ACLs or a specified MAC ACL.

access-list-name

Name of the ACL.

all

Clears statistics for all ACLs or ACLs for the protocol that you specify (IP, IPv6, or MAC).

Modes

EXEC (#)

Usage Guidelines

The counter counts the number of packets that match each **permit** or **deny** rule in an access list.

To view the counter statistics before clearing, see [show access-list](#) on page 937.

You must clear the counters if you want to start at zero for more recent statistics.

Examples

The following example clears statistics for an IP ACL named v4acl:

```
device# clear counters ipv4 access-list v4acl
device#
```

The following example clears statistics for an IPv6 ACL named v6acl:

```
device# clear counters ipv6 access-list v6acl
device#
```

The following example clears statistics for a MAC ACL named macacl:

```
device# clear counters mac access-list macacl
device#
```

The following example clears statistics for all existing IP ACLs:

```
device# clear counters ipv4 access-list all
device#
```

clear counters bfd

Use the **clear counters bfd** command to clear statistics for Bidirectional Forwarding Detection (BFD) sessions.

Syntax

```
clear counters bfd { dest-ip { dest-ipv4-addr | dest-ipv6-addr } [ vrf
    { vrf-name | all } ] | interface { ethernet { interface-name | all }
    | port-channel { port-channel-range | all } | ve { interface-name |
    all } | loopback { interface-name | all } } }
```

Command Default

Any existing BFD statistics remain uncleared.

Parameters

dest-ip { *dest-ipv4-addr* | *dest-ipv6-addr* }

Clear BFD sessions by destination IP address:

- *dest-ipv4-addr* filters sessions by IPv4 address. Use the *a.b.c.d* format.
- *dest-ipv6-addr* filters sessions by IPv6 address. Use the *x:x::x:x* format.

vrf { *vrf-name* | **all** }

- *vrf-name* filters sessions by VRF instance name. This name must start with an alphabetic character or an underscore followed by an arbitrary sequence of alphabetic or numeric characters, underscores, hyphens, or dots. The name cannot exceed 64 characters. Do not use the "profile" reserved keyword.
- **all** clears counters of BFD sessions that match the specified destination IP address across all VRF instances.

ethernet *interface-name*

Filter sessions by Ethernet interface name. Use the *slot/port* format or the *slot/port:breakout* format to specify an interface or one or more comma-separated ranges of interfaces respectively. For example, 0/1 or 1/1-3,5,2/7-9,10:1-4 respectively.

port-channel *port-channel-range*

Filter sessions by port-channel number. Use an integer or one or more comma-separated ranges of integers from 1 to 255. For example, 1 or 1-3,5,7-9.

ve *interface-name*

Filter sessions by Virtual Ethernet (VE) interface name. Use an integer or one or more comma-separated ranges of integers from 1 to 8192. For example, 2 or 2-3,5,7-9.

loopback *interface-name*

Filter sessions by loopback interface name. Use an integer or one or more comma-separated ranges of integers from 1 to 255. For example, 3 or 3-4,5,7-9.

Modes

EXEC (#)

Usage Guidelines

This command clears BFD session counters. The command lets you group sessions for clearing based on destination IP address, interface, VRF instance name, and client application type.

You must clear the counters if you want to start at zero for more recent statistics.

Examples

The following example clears BFD session counters on Ethernet interface 0/1 as well as BFD session counters that match destination IPv4 address 10.0.0.1 on a VRF instance named default-vrf:

```
device# clear counters bfd interface ethernet 0/1
device# clear counters bfd dest-ip 10.0.0.1 vrf default-vrf
device#
```

clear counters bridge-domain

Use the **clear counters bridge-domain** command to clear statistics for a specific bridge domain, a range of bridge domains, or all bridge domains.

Syntax

```
clear counters bridge-domain { bridge-domain-range | all }
```

Command Default

Any existing bridge domain statistics remain uncleared.

Parameters

bridge-domain-range

Bridge domain ID or range of IDs. Use an integer or range of integers from 1 to 8192. For example, 10 or 100-200.

all

Clears statistics for all bridge domains.

Modes

EXEC (#)

Usage Guidelines

To view the counter statistics before clearing, see [show counters bridge-domain](#) on page 975.

You must clear the counters if you want to start at zero for more recent statistics.

Examples

The following example clears statistics for bridge domain 100:

```
device# clear counters bridge-domain 100
device#
```

The following example clears statistics for a range of bridge domains from 100 through 200:

```
device# clear counters bridge-domain 100-200
device#
```

The following example clears statistics for all bridge domains on the device:

```
device# clear counters bridge-domain all
device#
```


clear counters interface ethernet

Use the **clear counters interface ethernet** command to clear statistics for one or more Ethernet interfaces.

Syntax

```
clear counters interface ethernet { interface-name | all }
```

Command Default

Any existing Ethernet interface statistics remain uncleared.

Parameters

interface-name

Ethernet interface name. Use the *slot/port* format or the *slot/port:breakout* format to specify an interface or one or more comma-separated ranges of interfaces respectively. For example, 0/1 or 1/1-3,5,2/7-9,10:1-4 respectively.

all

Clears statistics for all Ethernet interfaces.

Modes

EXEC (#)

Usage Guidelines

You must clear the counters if you want to start at zero for more recent statistics.

To view the counters before clearing them, see [show counters interface ethernet](#) on page 978.

Examples

The following example clears statistics for Ethernet interface 0/3:

```
device# clear counters interface ethernet 0/3
device#
```

The following example clears statistics for all Ethernet interfaces on the device:

```
device# clear counters interface ethernet all
device#
```

clear counters interface port-channel

Use the **clear counters interface port-channel** command to clear statistics for one or more port channel interfaces.

Syntax

```
clear counters interface port-channel { port-channel-range | all }
```

Command Default

Any existing port-channel interface statistics remain uncleared.

Parameters

port-channel-range

Port channel range. Use an integer or one or more comma-separated ranges of integers from 1 to 255. For example, 1 or 1-3,5,7-9.

all

Clears statistics for all port channel interfaces.

Modes

EXEC (#)

Usage Guidelines

To view the statistics for one or more port channel interfaces before clearing them, see [show counters interface port-channel](#) on page 983.

You must clear the counters if you want to start at zero for more recent statistics.

Examples

The following example clears statistics for port channel interface 2:

```
device# clear counters interface port-channel 2
device#
```

The following example clears statistics for all port channel interfaces on the device:

```
device# clear counters interface port-channel all
device#
```

clear counters interface tunnel

Use the **clear counters interface tunnel** command to clear statistics for one or more tunnel interfaces.

Syntax

```
clear counters interface tunnel { interface-name | all }
```

Command Default

Any existing tunnel interface statistics remain uncleared.

Parameters

interface-name

Tunnel interface name. Use a string or an integer from 1 to 100000.

all

Clears statistics for all tunnel interfaces.

Modes

EXEC (#)

Usage Guidelines

You must clear the counters if you want to start at zero for more recent statistics.

To view the counters before clearing them, see [show counters interface tunnel](#) on page 986.

Examples

The following example clears statistics for tunnel interface 1:

```
device# clear counters interface tunnel 1
device#
```

The following example clears statistics for all tunnel interfaces on the device:

```
device# clear counters interface tunnel all
device#
```

clear counters lacp

Use the **clear counters lacp** command to clear statistics for member ports of dynamic link aggregation groups (LAGs).

Syntax

```
clear counters lacp
```

Parameters

This command has no keywords or variables.

Modes

EXEC (#)

Usage Guidelines

To view the counter statistics before clearing, see [show counters lacp](#) on page 987.

You must clear the counters if you want to start at zero for more recent statistics.

You can use the **show running-config lacp** command to display the LAG configuration that is running currently on a device.

Examples

The following example clears statistics for member ports of the dynamic link aggregation groups that are configured on the device:

```
device# clear counters lacp
device#
```

clear counters lldp

Use the **clear counters lldp** command to clear all link layer discovery protocol (LLDP) statistics or LLDP statistics for one or more Ethernet interfaces.

Syntax

```
clear counters lldp { all | interface ethernet { interface-name | all } }
```

Command Default

Any existing LLDP statistics remain uncleared.

Parameters

all

Clears all LLDP statistics.

interface-name

Ethernet interface name. Use the *slot/port* format or the *slot/port:breakout* format to specify an interface or one or more comma-separated ranges of interfaces respectively. For example, 0/1 or 1/1-3,5,2/7-9,10:1-4 respectively.

all

Clears LLDP statistics for all Ethernet interfaces.

Modes

EXEC (#)

Usage Guidelines

You must clear the counters if you want to start at zero for more recent statistics.

To view the global LLDP statistics for the device before clearing them, see [show counters lldp summary](#) on page 993. To view the LLDP statistics for one or more Ethernet interfaces before clearing them, see [show counters lldp interface](#) on page 991.

Examples

The following example clears LLDP statistics for Ethernet interface 0/2:

```
device# clear counters lldp interface ethernet 0/2
device#
```

The following example clears LLDP statistics for all Ethernet interfaces:

```
device# clear counters lldp interface ethernet all
device#
```

The following example clears all LLDP statistics:

```
device# clear counters lldp all  
device#
```

clear counters mlag

Use the **clear counters mlag** command to clear the counters of a multichassis link aggregation (MLAG) peer.

Syntax

clear counters mlag peer *peer-name*

Command Default

Any existing MLAG peer statistics remain uncleared.

Parameters

peer *peer-name*

Name of the MLAG peer. The name must be a string 1-64 characters long.

Modes

EXEC (#)

Usage Guidelines

Use this command to clear statistics (total number of TX/RX counts for the primary keepalive and the secondary keepalive).

You must clear the counters if you want to start at zero for more recent statistics.

You can use the **show running-config mlag** command to display the MLAG configuration that is running currently on a device.

Examples

The following example clears the counters on an MLAG peer named m12345:

```
device# clear counters mlag peer m12345
device#
```

clear counters qos interface control-plane queue

Use the **clear counters qos interface control-plane queue** command to clear statistics for a specific CPU queue or all CPU queues on the control plane interface.

Syntax

```
clear counters qos interface control-plane queue { queue-name | all }
```

Command Default

Any existing QoS queue statistics on the control plane interface remain uncleared.

Parameters

queue-name

Name of the QoS queue for which to clear statistics. Use a string from q0 through q47 to specify one of the 48 queues. For example, q27.

all

Clears statistics for all QoS queues.

Modes

EXEC (#)

Usage Guidelines

You must clear the counters if you want to start at zero for more recent statistics.

To view the statistics for a specific QoS queue or all QoS queues on the control plane interface before clearing them, see [show counters qos interface control-plane queue](#) on page 997.

Examples

The following example clears statistics for QoS queue q0:

```
device# clear counters qos interface control-plane queue q0
device#
```

The following example clears statistics for all QoS queues:

```
device# clear counters qos interface control-plane queue all
device#
```


clear counters qos interface ethernet queue

Use the **clear counters qos interface ethernet queue** command to clear QoS statistics for a specific QoS queue or all QoS queues on one or more Ethernet interfaces.

Syntax

```
clear counters qos interface ethernet { interface-name | all } queue  
    { queue-name | all }
```

Command Default

Any existing QoS statistics on an Ethernet interface remain uncleared.

Parameters

interface-name

Ethernet interface name. Use the *slot/port* format or the *slot/port:breakout* format to specify an interface or one or more comma-separated ranges of interfaces respectively. For example, 0/1 or 1/1-3,5,2/7-9,10:1-4 respectively.

all

Clears QoS statistics for all Ethernet interfaces.

queue-name

Name of the QoS queue for which to clear statistics. Use a string from q0 through q7 to specify one of the 8 queues. For example, q2.

all

Clears statistics for all QoS queues.

Modes

EXEC (#)

Usage Guidelines

You must clear the counters if you want to start at zero for more recent statistics.

To view the QoS statistics for a specific Ethernet interface or all Ethernet interfaces before clearing them, see [show counters qos interface ethernet queue](#) on page 1000.

Examples

The following example clears statistics for Ethernet interface 0/2 and QoS queue q0:

```
device# clear counters qos interface ethernet 0/2 queue q0  
device#
```

The following example clears statistics for Ethernet interface 0/2 and all QoS queues:

```
device# clear counters qos interface ethernet 0/2 queue all
device#
```

clear ipv4 neighbor interface (Extreme ONE OS Switching)

Use the **clear ipv4 neighbor interface** command to clear the address resolution protocol (ARP) entries in the neighbor cache for a specific interface.

Syntax

```
clear ipv4 neighbor interface { ethernet interface-name | port-channel port-channel | tunnel interface-number | ve interface-name } [ mlag | dynamic | any | evpn ]
```

Parameters

ethernet *interface-name*

Clears entries by Ethernet interface name. Use the *slot/port* format or the *slot/port:breakout* format to specify an interface. For example, 0/1.

port-channel *port-channel*

Clears entries by port channel number. Use an integer from 1 to 255.

tunnel *interface-number*

Clears entries by tunnel interface number. Use an integer from 1 to 100000.

ve *interface-name*

Clears entries by Virtual Ethernet (VE) interface name. Use an integer from 1 to 8192.

mlag

Clears multichassis link aggregation (MLAG) learned ARP entries.

dynamic

Clears dynamic ARP entries.

any

Clears any source ARP entries.

evpn

Clears Ethernet VPN (EVPN) learned ARP entries.

Modes

EXEC (#)

Usage Guidelines

Use this command to clear the ARP entries in the neighbor cache for a specific interface (and a specific source, if specified).

ARP establishes correspondences between pairs of network-layer addresses (Layer 3) and LAN hardware addresses (Layer 2 MAC addresses). Each correspondence is recorded in a cache on the device for a predetermined period and then deleted.

Clearing the ARP cache is a quick way to remove accumulated data for hosts that are connected directly to the device.

Examples

The following example clears ARP entries for Ethernet interface 0/1 on the device:

```
device# clear ipv4 neighbor interface ethernet 0/1
device#
```

The following example clears MLAG learned ARP entries for Ethernet interface 0/1 on the device:

```
device# clear ipv4 neighbor interface ethernet 0/1 mlag
device#
```

The following example clears ARP entries for port channel port 3 on the device:

```
device# clear ipv4 neighbor interface port-channel 3
device#
```

The following example clears dynamic ARP entries for port channel port 3 on the device:

```
device# clear ipv4 neighbor interface port-channel 3 dynamic
device#
```

The following example clears ARP entries for tunnel interface 5 on the device:

```
device# clear ipv4 neighbor interface tunnel 5
device#
```

The following example clears EVPN learned ARP entries for tunnel interface 5 on the device:

```
device# clear ipv4 neighbor interface tunnel 5 evpn
device#
```

clear ipv4 neighbor vrf

Use the **clear ipv4 neighbor vrf** command to clear the address resolution protocol (ARP) entries in the neighbor caches for Virtual Routing and Forwarding (VRF) instances.

Syntax

```
clear ipv4 neighbor vrf { all | vrf-name } [ { ethernet interface-name |  
  ve interface-name } | ipv4-address ]
```

Parameters

vrf { **all** | *vrf-name* }

- all clears ARP entries for all VRF instances.
- *vrf-name* clears ARP entries for a specific VRF instance. This name must start with an alphabetic character or an underscore followed by an arbitrary sequence of alphabetic or numeric characters, underscores, hyphens, or dots. The name cannot exceed 64 characters. Do not use the profile reserved keyword.

ethernet *interface-name*

Filter ARP entries by Ethernet interface name. Use the *slot/port* format. For example, 0/1.

ve *interface-name*

Filter ARP entries by Virtual Ethernet (VE) interface name. Use an integer from 1 to 8192.

ipv4-address

Filter ARP entries by the IPv4 address of a neighboring device. Use the *a.b.c.d* format.

Modes

EXEC (#)

Usage Guidelines

ARP establishes correspondences between pairs of network-layer addresses (Layer 3) and LAN hardware addresses (Layer 2 MAC addresses). Each correspondence is recorded in a cache on the device for a predetermined period and then deleted. Clearing the ARP cache is a quick way to remove accumulated data for hosts that are connected directly to the device.

Use the **clear ipv4 neighbor vrf** command to clear the types of data that are listed in the following table. You must clear this data if you want to start at zero for more

recent statistics. This table shows the data names (as they appear in the output from the **show ipv4 neighbor vrf** command) and the data descriptions:

Output field	Description
Vrf or VRF instance	Name of a VRF instance that is configured on the device.
Total number of arp entries	Number of ARP entries in the neighbor cache for the corresponding VRF.
Ip-address or IP Address	Network address of the neighboring device that corresponds to the LAN hardware address.
Mac-address or MAC Address	LAN hardware address (data link address) of the MAC address of the neighboring device that corresponds to the network address.
Type	Current state (either dynamic or static) of the ARP entry (the Ethernet MAC to IPv4 address link) in the cache. Most ARP entries are dynamic, which means that they remain on a device for a certain time, if they are in use. ARP automatically creates and updates dynamic ARP entries (which means that they are learned on an interface). A dynamic ARP entry is deleted when its ages out, or when the output interface goes down. A dynamic ARP entry can be overwritten by a static ARP entry. A static ARP entry is configured manually. It does not age out and cannot be overwritten by a dynamic ARP entry. A static ARP entry protects communication between devices, because attack packets cannot modify its IP-to-MAC mapping. ARP entries that correspond to the local IP address on the interface never age out.
Interface	Logical interface to which this network address mapping is assigned. This is either the VE interface or the Layer 2 interface (if no VE interface is mapped to this interface).
L2 Interface	Layer 2 (physical) interface to which this network MAC address is associated.
Age	Age expressed in standard time format (hour, minutes, and seconds) of the ARP entry in the cache.
Seq No.	Sequence number. This number determines which “type” is selected as the best route for a given IP address. The best type is indicated by an asterisk (*).

Examples

The following example clears ARP entries for all VRF instances on the device:

```
device# clear ipv4 neighbor vrf all
device#
```

The following example clears ARP entries for a VRF instance named vrf1 on the device:

```
device# clear ipv4 neighbor vrf vrf1
device#
```

The following example clears ARP entries for a VRF instance named vrf1 on the device:

```
device# clear ipv4 neighbor vrf vrf1
device#
```

The following example clears ARP entries for a VRF instance named vrf1 on the device. In this example, the entries are filtered by the IPv4 address of a neighboring device:

```
device# clear ipv4 neighbor vrf vrf1 101.1.1.2
device#
```

clear ipv6 neighbor interface

Use the **clear ipv6 neighbor interface** command to clear the address resolution protocol (ARP) entries in the neighbor cache for a specific interface.

Syntax

```
clear ipv6 neighbor interface { ethernet interface-name | port-channel port-channel | tunnel interface-number | ve interface-name } [ mlag | dynamic | any | evpn ]
```

Parameters

ethernet *interface-name*

Clears entries by Ethernet interface name. Use the *slot/port* format to specify an interface. For example, 0/1.

port-channel *port-channel*

Clears entries by port channel number. Use an integer from 1 to 255.

tunnel *interface-number*

Clears entries by tunnel interface number. Use an integer from 1 to 100000.

ve *interface-name*

Clears entries by Virtual Ethernet (VE) interface name. Use an integer from 1 to 8192.

mlag

Clears multichassis link aggregation (MLAG) learned ARP entries.

dynamic

Clears dynamic ARP entries.

any

Clears any source ARP entries.

evpn

Clears Ethernet VPN (EVPN) learned ARP entries.

Modes

EXEC (#)

Usage Guidelines

Use this command to clear the ARP entries in the neighbor cache for a specific interface (and a specific source, if specified).

ARP establishes correspondences between pairs of network-layer addresses (Layer 3) and LAN hardware addresses (Layer 2 MAC addresses). Each correspondence is recorded in a cache on the device for a predetermined period and then deleted.

Clearing the ARP cache is a quick way to remove accumulated data for hosts that are connected directly to the device.

Use the **clear ipv6 neighbor interface** command to clear the types of data that are listed in the following table. You must clear this data if you want to start at zero for more recent statistics. This table shows the data names (as they appear in the output from the **show ipv6 neighbor interface** command) and the data descriptions:

Examples

The following example clears ARP entries for Ethernet interface 0/1 on the device:

```
device# clear ipv6 neighbor interface ethernet 0/1
device#
```

The following example clears MLAG learned ARP entries for Ethernet interface 0/1 on the device:

```
device# clear ipv6 neighbor interface ethernet 0/1 mlag
device#
```

The following example clears ARP entries for port channel 3 on the device:

```
device# clear ipv6 neighbor interface port-channel 3
device#
```

The following example clears dynamic ARP entries for port channel 3 on the device:

```
device# clear ipv6 neighbor interface port-channel 3 dynamic
device#
```

The following example clears ARP entries for tunnel interface 5 on the device:

```
device# clear ipv6 neighbor interface tunnel 5
device#
```

The following example clears EVPN learned ARP entries for tunnel interface 5 on the device:

```
device# clear ipv6 neighbor interface tunnel 5 evpn
device#
```

clear ipv6 neighbor vrf

Use the **clear ipv6 neighbor vrf** command to clear the neighbor discovery protocol (NDP) entries in the neighbor caches for Virtual Routing and Forwarding (VRF) instances.

Syntax

```
clear ipv6 neighbor vrf { all | vrf-name } [ { ethernet interface-name |  
  ve interface-name } | ipv6-address ]
```

Parameters

vrf { **all** | *vrf-name* }

- **all** clears NDP entries for all VRF instances.
- *vrf-name* clears NDP entries for a specific VRF instance. This name must start with an alphabetic character or an underscore followed by an arbitrary sequence of alphabetic or numeric characters, underscores, hyphens, or dots. The name cannot exceed 64 characters. Do not use the profile reserved keyword.

ethernet *interface-name*

Filter NDP entries by Ethernet interface name. Use the *slot/port* format to specify an interface. For example, 0/1.

ve *interface-name*

Filter NDP entries by Virtual Ethernet (VE) interface name. Use an integer from 1 to 8192.

ipv6-address

Filter NDP entries by the IPv6 address of a neighboring device. Use the *xx:xx:xx* format.

Modes

EXEC (#)

Usage Guidelines

NDP establishes correspondences between pairs of network-layer (Layer 3) addresses and LAN hardware addresses (Layer 2 MAC addresses). Each correspondence is recorded in a cache on the device for a predetermined period and then deleted. Clearing the NDP cache is a way to immediately remove this accumulated data for hosts that are connected directly to the device.

Output

Use the **clear ipv6 neighbor vrf** command to clear the types of data that are listed in the following table. You must clear this data if you want to start at zero for more recent statistics. This table shows the data names (as they appear in the output from the **show ipv6 neighbor vrf** command) and the data descriptions:

Output field	Description
Vrf or VRF instance	Name of a VRF instance that is configured on the device.
Total number of neighbor entries	Number of NDP entries in the neighbor cache for the corresponding VRF.
Ip-address or IP Address	Network address of the neighboring device that corresponds to the LAN hardware address.
Mac-address or MAC Address	LAN hardware address (data link address) of the MAC address of the neighboring device that corresponds to the network address.
Type	Current state (either dynamic or static) of the NDP entry (the Ethernet MAC to IPv6 address link) in the cache. Most NDP entries are dynamic, which means that they remain on a device for a certain time, if they are in use. NDP automatically creates and updates dynamic NDP entries (which means that they are learned on an interface). A dynamic NDP entry is deleted when its ages out, or when the output interface goes down. A dynamic NDP entry can be overwritten by a static NDP entry. A static NDP entry is configured manually. It does not age out and cannot be overwritten by a dynamic NDP entry. A static NDP entry protects communication between devices, because attack packets cannot modify its IP-to-MAC mapping. NDP entries that correspond to the local IPv6 address on the interface never age out.
Interface	Logical interface to which this network address mapping is assigned. This is either the VE interface or the Layer 2 interface (if no VE interface is mapped to this interface).
L2 Interface	Layer 2 (physical) interface with which this network MAC address mapping is associated.
Age	Age in standard time format (hours, minutes, and seconds) of the NDP entry in the cache.
Seq No.	Sequence number. This number determines which “type” is selected as the best route for a given IP address. The best type is indicated by an asterisk (*).

Examples

The following example clears NDP entries for all VRF instances on the device:

```
device# clear ipv6 neighbor vrf all
device#
```

The following example clears NDP entries for a VRF instance named vrf1 on the device:

```
device# clear ipv6 neighbor vrf vrf1
device#
```

The following example clears NDP entries for a VRF instance named vrf1 on the device:

```
device# clear ipv6 neighbor vrf vrf1
device#
```

The following example clears NDP entries for a VRF instance named vrf1 on the device. In this example, the entries are filtered by the IPv6 address of a neighboring device:

```
device# clear ipv6 neighbor vrf vrf1 1111::2
device#
```

clear l2 mac-move-detection shut-list

Use the **clear l2 mac-move-detection shut-list** command to clear the shut list and set the port state to up for all ports in a list or for a specified port.

Syntax

```
clear l2 mac-move-detection shut-list [ interface { ethernet interface-name | port-channel port | ve interface-name | loopback interface-name } ]
```

Parameters

interface

(Optional) Specifies a specific interface.

ethernet *interface-name*

Ethernet interface. Use the *slot/port* format. For example, 0/1.

port-channel *port*

LAG interface. Use an integer from 1 to 255.

ve *interface-name*

Virtual Ethernet (VE) interface. Use an integer from 1 to 8192.

loopback *interface-name*

Loopback interface. Use an integer from 1 to 255.

Modes

EXEC (#)

Usage Guidelines

This command clears the shut list and sets the port state to up for a single interface name (as it is shown in the shut list) or for all interfaces on the device.

To view the contents of the list before clearing it, see [show l2 mac-move-detection shut-list](#) on page 1055.

Examples

The following example clears the shut list and sets the port state to up for all interfaces on the device:

```
device# clear l2 mac-move-detection shut-list
device#
```

The following example clears the shut list and sets the port state to up for Ethernet interfaces 0/1:2.100 on the device:

```
device# clear 12 mac-move-detection shut-list interface ethernet 0/1:2.100
device#
```

clear ipv4 route vrf (IP routing)

Use the **clear ipv4 route vrf** command to clear entries in the IPv4 routing table for a Virtual Routing and Forwarding (VRF) instance.

Syntax

```
clear ipv4 route vrf vrf-name [ ipv4-address / prefix-length ]
```

Command Default

Old entries in the IPv4 routing table for a Virtual Routing and Forwarding (VRF) instance remain uncleared.

Parameters

vrf-name

Name of the VRF with the entries to be cleared.

ipv4-address / *prefix-length*

IPv4 address. Use the *a.b.c.d* format for *ipv4-address*, and use an integer from 0 to 32 for the prefix. For example, 10.1.1.0/24. You must include the slash (/).

Modes

EXEC (#)

Usage Guidelines

You must clear this data if you want to start at zero for more recent statistics.

To view the IPv4 routing table details before clearing them, see [show ipv4 route vrf \(IP routing\)](#) on page 1033.

Examples

The following example clears all entries in the IPv4 routing table for a VRF instance named default-vrf on the device:

```
device# clear ipv4 route vrf default-vrf
device#
```

The following example clears all entries with the 12.1.1.10/32 prefix in the IPv4 routing table for a VRF instance named default-vrf on the device:

```
device# clear ipv4 route vrf default-vrf 12.1.1.10/32
device#
```

clear ipv6 route vrf (IP routing)

Use the **clear ipv6 route vrf** command to clear entries in the IPv6 routing table for a Virtual Routing and Forwarding (VRF) instance.

Syntax

```
clear ipv6 route vrf vrf-name [ ipv6-address / prefix-length ]
```

Command Default

Old entries in the IPv6 routing table for a Virtual Routing and Forwarding (VRF) instance remain uncleared.

Parameters

vrf-name

Name of the VRF with the entries to be cleared.

ipv6-address / *prefix-length*

IPv6 address. Use the *x:x::x:x* format for *ipv6-address*, and use an integer from 0 to 128 for the prefix. For example, 2001::1/64. You must include the slash (/).

Modes

EXEC (#)

Usage Guidelines

You must clear this data if you want to start at zero for more recent statistics.

To view the IPv6 routing table details before clearing them, see [show ipv6 route vrf \(IP routing\)](#) on page 1038.

Examples

The following example clears all entries in the IPv6 routing table for a VRF instance named vrf1 on the device:

```
device# clear ipv6 route vrf vrf1
device#
```

The following example clears all entries with the 2001::30/128 prefix in the IPv6 routing table for a VRF instance named vrf1 on the device:

```
device# clear ipv6 route vrf vrf1 2001::30/128
device#
```


clear mac-address-table bridge-domain

Use the **clear mac-address-table bridge-domain** command to clear the MAC address table for a specific bridge domain.

Syntax

```
clear mac-address-table bridge-domain ID { static | dynamic | mlag | evpn  
      | mac-address }
```

Command Default

The MAC address table for the bridge domain remains uncleared.

Parameters

ID

ID number of the bridge domain to clear. Use an integer from 1 to 8192.

static

MAC addresses learned from the configuration.

dynamic

MAC addresses learned dynamically.

mlag

MAC addresses learned via multichassis link aggregation (MLAG).

evpn

MAC addresses learned via BGP Ethernet VPN (EVPN).

mac-address

Specific MAC address. Use six groups of two hex digits separated by colons. For example, 01:23:45:67:89:AB.

Modes

EXEC (#)

Usage Guidelines

You can clear a specific MAC address table or all MAC addresses of a specific type: static, dynamic, multichassis link aggregation (MLAG), or Ethernet VPN (EVPN).

To view the contents of the MAC address table for a specific bridge domain before clearing it, see [show mac-address-table bridge-domain](#) on page 1056.

Examples

The following example clears the table for dynamically learned MAC addresses for bridge domain 100:

```
device# clear mac-address-table bridge-domain 100 dynamic
device#
```

The following example clears the table for MAC address 01:01:01:01:01:01 for bridge domain 100:

```
device# clear mac-address-table bridge-domain 100 01:01:01:01:01:01
device#
```

cluster-id

Use the **cluster-id** command to assign a custom cluster ID to a BGP peer group within a Virtual Routing and Forwarding (VRF) instance.

Syntax

```
cluster-id { integer | dotted-quad-number }  
no cluster-id { integer | dotted-quad-number }
```

Command Default

No cluster ID is enabled for the specified BGP peer group.

Parameters

integer | *dotted-quad-number*

ID to be used by the BGP cluster. You can specify an integer from 1 to 4294967295 or a number in dotted quad format (x.x.x.x) such as 8.8.8.8.

Modes

VRF BGP peer group configuration (config-vrf-bgp-pg)

Usage Guidelines

Use the **no** form of this command to remove the cluster ID.

Examples

The following example configures two BGP peer groups (group1 and group2) under a VRF instance named violet. group1 is assigned a cluster ID (10) in integer format, and group2 is assigned a cluster ID (8.8.8.8) in dotted quad format:

```
device# configure terminal  
device(config)# vrf violet  
device(config-vrf-violet)# router bgp  
device(config-vrf-bgp)# peer-group group1  
device(config-vrf-bgp-pg)# cluster-id 10  
device(config-vrf-bgp-pg)# exit  
device(config-vrf-bgp)# peer-group group2  
device(config-vrf-bgp-pg)# cluster-id 8.8.8.8  
device(config-vrf-bgp-pg)#
```

The following example displays the configuration of a VRF instance named violet that is running currently on the device. In this example, both peer groups are assigned their own cluster ID:

```
device# show running-config vrf violet  
  
vrf violet
```

```
!  
router bgp  
  peer-group group1  
    cluster-id 10  
  peer-group group2  
    cluster-id 8.8.8.8  
!  
device#
```

community-set

Use the **community-set** command to configure a BGP community set and enter Routing policy BGP community set configuration (config-route-policy-bgp-set-community-*name*) mode.

Syntax

community-set *set-name*

no community-set *set-name*

Command Default

No BGP community sets are configured on the device.

Parameters

set-name

Name of the BGP community set to configure. This is used to reference the community set in match conditions. Use any combination of uppercase or lowercase alphabetic characters, numerals, underscores, or hyphens. You must use an alphabetic character as the first character.

Modes

Routing policy configuration (config-route-policy)

Usage Guidelines

Use this command to create a named BGP community set and enter the mode for configuring the community set.

To remove a BGP community set from the device, use the **no** form of this command.

Examples

The following example creates a BGP community set named comm1:

```
device# configure terminal
device(config)# route-policy
device(config-route-policy)# bgp-defined-sets
device(config-route-policy-bgp-set)# community-set comm1
device(config-route-policy-bgp-set-community-comm1)# match-set-options all
device(config-route-policy-bgp-set-community-comm1)# member 65535 65535:100
device(config-route-policy-bgp-set-community-comm1)#
```

The following example displays the routing policy configuration that is running currently on the device. In this example, the device is configured with one BGP defined set, which is an community set named comm1:

```
device# show running-config route-policy

route-policy
  bgp-defined-sets
    community-set comm1
      member 65535 65535:100
      match-set-options all
device#
```

conditions (routing policy map statement configuration)

Use the **conditions** command to enter Routing policy statement conditions configuration (config-route-policy-policy-*name*-stmt-*number*-conditions) mode.

Syntax

conditions

no conditions

Command Default

The specified routing policy statement contains no conditions.

Parameters

This command has no keywords or variables.

Modes

Routing policy statement configuration (config-route-policy-policy-*name*-stmt-*number*)

Usage Guidelines

This command enters the mode for adding conditions to the specified routing policy statement. Each statement in a routing policy can have only one BGP actions container. This container can contain multiple actions. A statement can have single "conditions" configuration.

To remove all match conditions from a routing policy statement, use the **no** form of this command.

Examples

The following example creates a routing policy named map1 that contains a policy statement with sequence number 10 and enters Routing policy statement conditions configuration (config-route-policy-policy-*name*-stmt-*number*-conditions) mode:

```
device# configure terminal
device(config)# route-policy
device(config-route-policy)# policy map1
device(config-route-policy-policy-map1)# statement 10
device(config-route-policy-policy-map1-stmt-10)# conditions
device(config-route-policy-policy-map1-stmt-10-conditions)#
```

The following example displays the routing policy configuration that is running currently on the device. In this example, a routing policy named map1 contains a policy

statement with sequence number 10 containing a condition for matching members of a community set named comm1:

```
device# show running-config route-policy

route-policy
  policy map1
    statement 10
      conditions
        bgp-conditions
          match-community-set comm1
device#
```


connect-retry

Use the **connect-retry** command to set the BGP connection retry interval to a nondefault value for all members of a BGP peer group within a Virtual Routing and Forwarding (VRF) instance.

Syntax

connect-retry *seconds*

no connect-retry *seconds*

Command Default

The BGP connection retry interval is set to 30 seconds.

Parameters

seconds

Duration of interval. Use an integer from 1 to 65535.

Modes

VRF BGP peer group timer configuration (config-vrf-bgp-pg-timers)

Usage Guidelines

Use this command to set the wait time between attempts to establish a BGP session to apply to all members of a specified BGP peer group.

Use the **no** form of this command to reset the wait time to the default value for all members of the BGP peer group.

Examples

The following example configures a BGP peer group named `group1` under a VRF instance named `violet`. This example sets the interval between attempts to establish a BGP connection to two seconds for the members of `group1`:

```
device# configure terminal
device(config)# vrf violet
device(config-vrf-violet)# router bgp
device(config-vrf-bgp)# peer-group group1
device(config-vrf-bgp-pg)# timers
device(config-vrf-bgp-pg-timers)# connect-retry 2
device(config-vrf-bgp-pg-timers)#
```

The following example displays the configuration of a VRF instance named violet that is running currently on the device. In this example, the group1 peer group has a reconnection wait time of two seconds for each member:

```
device# show running-config vrf violet

vrf violet
!
  router bgp
    peer-group group1
      timers
        connect retry 2
!
device#
```

cos traffic-class

Use the **cos traffic-class** command to define the CoS-to-traffic-class entries in a QoS traffic-class map.

Syntax

```
cos cos-value traffic-class traffic-class-value  
no cos cos-value
```

Command Default

The specified QoS traffic-class map contains no CoS-to-traffic-class entries.

Parameters

cos-value

Class of Service (CoS) value. Use an integer from 0 to 7.

traffic-class-value

Traffic class value. Use an integer from 0 to 7.

Modes

QoS traffic class map configuration

Usage Guidelines

To remove a CoS-to-traffic-class entry from a QoS traffic-class map, use the no form of this command.

Examples

The following example configures two CoS-to-traffic-class entries in a QoS traffic-class map named abc:

```
device# configure terminal  
device(config)# qos  
device(config-qos)# traffic-class-map abc  
device(config-qos-traffic-class-map-abc)# cos 1 traffic-class 3  
device(config-qos-traffic-class-map-abc)# cos 2 traffic-class 4  
device(config-qos-traffic-class-map-abc)#
```

The following example displays the QoS configuration that is running currently on the device. In this example, a QoS traffic-class map named abc is configured with two CoS-to-traffic-class entries:

```
device# show running-config qos  
  
qos  
    traffic-class-map abc
```

```
    cos 1 traffic-class 3
    cos 2 traffic-class 4
device#
```

cpus

Use the **cpus** command to configure the number of CPUs for a virtual machine (VM).

Syntax

cpus *number*

no cpus

Command Default

The specified VM uses two CPUs.

Parameters

number

Number of CPUs. Use an integer from 1 to 16.

Modes

IAH VM configuration (config-iah-vm-*name*)

Usage Guidelines

Use the **no** form of this command to remove the CPUs from a VM. Note that configuration is not allowed when the VM is in the deployed state.

The maximum number of CPUs allocated for IAH depends on the platform (1 to 16 for the 8730 platform and 1 to 4 for other platforms).

To display details about a specific VM, see [show iah vm](#) on page 1006.

Examples

The following example configures four CPUs for a VM named Ubuntu1 on the device:

```
device# configure terminal
device(config)# iah
device(config-iah)# vm Ubuntu1
device(config-iah-vm-Ubuntu1)# cpus 4
device(config-iah-vm-Ubuntu1)#
```

The following example displays the IAH configuration that is running currently on the device. In this example, four CPUs are configured for a VM named Ubuntu1:

```
device# show running-config iah

iah
  vm Ubuntu1
    description Ubuntu demo vm
    auto-boot true
```

```
cpus 4
ram 1024
disk sda
    size 5000
    format qcow2
    bus sata
    file disk://iah/disk1.qcow2
!
!
!
device#
```

decapsulation (QoS NVO configuration)

Use the **decapsulation** command to set the fabric QoS tunnel decapsulation mode for a Network Virtualization Overlay (NVO) endpoint.

Syntax

```
decapsulation { uniform | pipe }
```

Command Default

Fabric QoS tunnel decapsulation on the NVO endpoint is set to pipe mode.

Parameters

uniform

Sets fabric QoS tunnel decapsulation to uniform mode. With uniform mode, the core network's QoS settings are applied to inner packets (even after exiting the tunnel), which enables centralized QoS control in the network.

pipe

Sets fabric QoS tunnel decapsulation to pipe mode. With pipe mode, traffic is untouched when traversing the fabric (the inner packet QoS settings are preserved throughout transmission to ensure end-to-end consistency).

Modes

QoS NVO configuration (*config-qos-nvo-name*)

Usage Guidelines

This command configures fabric QoS tunnel mode decapsulation for an NVO on a device. The two primary tunneling modes manage how packets are decapsulated and how their QoS marking is handled:

- In uniform mode decapsulation, the QoS marking of the outer tunnel headers propagate to the inner packets when the tunnel is decapsulated. This means that the inner packet QoS settings can be overwritten by the outer tunnel's QoS markings. Uniform mode applies the core network's QoS settings to inner packets (even after they exit the tunnel) to enable centralized QoS control in the network.
- In pipe mode decapsulation, the QoS markings (DSCP bits) of inner packets are preserved and not altered by the outer tunnel's QoS marking.

To restore the decapsulation setting to the default value on the device, use the **no** form of this command.

To configure fabric QoS tunnel mode *encapsulation* for an NVO on a device, see [encapsulation \(QoS NVO configuration\)](#) on page 605.

Examples

The following example enters QoS NVO tunnel mode configuration (config-qos-nvo-*name*-tunnel-mode) mode and configures fabric QoS tunnel encapsulation and decapsulation on the device:

```
device# configure terminal
device(config)# qos
device(config-qos)# nvo mynvo
device(config-qos-nvo-mynvo)# tunnel-mode
device(config-qos-nvo-mynvo-tunnel-mode)# encapsulation pipe dscp 34
device(config-qos-nvo-mynvo-tunnel-mode)# decapsulation uniform
device(config-qos-nvo-mynvo-tunnel-mode)#
```

The following example shows the QoS configuration that is running currently on the device. In this example, a QoS node has an associated NVO instance named mynvo with QoS tunnel encapsulation and decapsulation configured:

```
device# show running-config qos

qos
  nvo mynvo
    tunnel-mode
      encapsulation pipe dscp 34
      decapsulation uniform
    !
  !
!
device#
```


default-route-originate (BGP peer group address family configuration)

Use the **default-route-originate** command to originate the default route and advertise it to the neighbors in a specific peer group within a routing process in a VRF instance.

Syntax

```
default-route-originate  
no default-route-originate
```

Command Default

The default route is not originated or advertised to the neighbors in the specified peer group.

Parameters

This command has no keywords or variables.

Modes

VRF BGP peer group address family configuration (*config-vrf-bgp-pg-address-family-type*)

Usage Guidelines

This command applies only to IPv4 unicast or IPv6 unicast address families.

The default route is generated artificially and does not need to be present in the routing table.

Use the **no** form of this command to restore the default functionality.

Examples

The following example configures a routing process under a VRF instance named violet. This example specifies an IPv4 unicast address family in a peer group named group1 and enables the peer group to originate the default route and advertise it to its neighbors:

```
device# configure terminal  
device(config)# vrf violet  
device(config-vrf-violet)# router bgp  
device(config-vrf-bgp)# peer-group group1  
device(config-vrf-bgp-pg)# address-family ipv4 unicast  
device(config-vrf-bgp-pg-ipv4u)# default-route-originate  
device#
```

The following example displays the configuration of a VRF instance named violet that is running currently on the device. In this example, a peer group named group1 is configured and enabled to originate the default route and advertise it to its neighbors. An IPv4 unicast address family is specified in this example:

```
device# show running-config vrf violet

vrf violet
  router bgp
    peer-group group1
      address-family ipv4 unicast
        default-route-originate
device#
```

default-traffic-class

Use the **default-traffic-class** command to change the default traffic class in a QoS traffic-class map to a nondefault value.

Syntax

```
default-traffic-class traffic-class-value  
no default-traffic-class
```

Command Default

The specified QoS traffic-class map uses 1 as the default traffic class.

Parameters

traffic-class-value
Traffic class value. Use an integer from 0 to 7.

Modes

QoS traffic class map configuration (config-qos-traffic-class-map-*name*)

Usage Guidelines

To restore the default traffic class to the default value in a QoS traffic-class map, use the no form of this command.

Examples

The following example configures the default traffic class in a QoS traffic-class map named ing_tc_map:

```
device# configure terminal  
device(config)# qos  
device(config-qos)# traffic-class-map ing_tc_map  
device(config-qos-traffic-class-map-ing_tc_map)# default-traffic-class 3  
device(config-qos-traffic-class-map-ing_tc_map)#
```

The following example displays the QoS configuration that is running currently on the device. In this example, a QoS traffic-class map named ing_tc_map is configured with a default traffic class:

```
device# show running-config qos  
  
qos  
  traffic-class-map ing_tc_map  
    default-traffic-class 3  
device#
```

deny (IPv4 ACL configuration)

Use the **deny** command to command to configure an access control rule in an IPv4 access control list (ACL) to deny forwarding of packets.

Syntax

```
[ seq sequence-num ] deny { ip-protocol-num | ipv4 | esp | icmp | igmp
| udp | tcp [ flag-list ] } { source-ipv4-addr subnet-mask | source-ipv4-addr/prefix-length | any } { dest-ipv4-addr subnet-mask | dest-ipv4-addr/prefix-length | any } [ { dontfragment | morefragment } ]
[ dport dest-l4-port ] [ sport source-l4-port ] [ dscp packet-dsp-value ] [ vlan packet-vlan-id ] [ count ] [ log ]
```

no seq *sequence-num*

```
no deny { ip-protocol-num | ipv4 | esp | icmp | igmp | udp | tcp
[ flag-list ] } { source-ipv4-addr subnet-mask | source-ipv4-addr/prefix-length | any } { dest-ipv4-addr subnet-mask | dest-ipv4-addr/prefix-length | any } [ { dontfragment | morefragment } ] [ dport
dest-l4-port ] [ sport source-l4-port ] [ dscp packet-dsp-value ] [ vlan packet-vlan-id ] [ count ] [ log ]
```

Command Default

The ACL is not configured with deny rules.

Parameters

seq *sequence-num*

(Optional) Sequence number to identify the relative priority of this rule in the ACL. Specify an integer from 1 to 65535. You should specify an unused sequence number in a multiple of 10. If you do not specify a value, the rule is added to the end of the list.

ip-protocol-num

Matches packets that use a specific IP protocol. Specify an integer from 1 to 254. For example, specify 6 for TCP packets or 115 for L2TP packets. For a list of IP protocol numbers, see <https://www.iana.org/assignments/protocol-numbers/protocol-numbers.xhtml>.

ipv4

Matches packets that use any IPv4 protocol (including ICMPv4, IGMPv4, UDP, and TCP).

esp

Matches packets encrypted with Encapsulation Security Payload (ESP).

icmp

Matches packets carrying Internet Control Message Protocol v4 (ICMPv4) messages. These packets contain the control messages such as Echo Request/Echo Reply or Destination Unreachable.

igmp

Matches packets carrying Internet Group Management Protocol v4 (IGMPv4) messages. Hosts and adjacent routers use these on IPv4 networks to establish multicast group memberships, which optimize performance for one-to-many networking applications such as streaming video.

udp

Matches packets that use the User Datagram Protocol (UDP) format. This allows packets using connectionless protocols such as TFTP and DNS.

tcp [*flag-list*]

Matches TCP packets. This allows packets that use TCP connection-oriented protocols such as HTTPS, FTP, and Telnet. You can omit flags to match all TCP packets, or you can specify any space-separated combination of the flags listed in [Table 7](#) on page 559 below.

source-ipv4-addr subnet-mask | *source-ipv4-addr/prefix-length* | **any**

Source IPv4 addresses. These represent the networks or hosts sending the packets. Use one of the following methods:

source-ipv4-addr subnet-mask matches packets from a range of IPv4 addresses via a subnet mask. Use the *a.b.c.d* format for the IPv4 address and the subnet mask.

source-ipv4-addr/prefix-length matches packets from a range of IPv4 addresses via a subnet prefix length. Use the *a.b.c.d* format for the IPv4 address, and use an integer from 0 to 32 for the prefix. You must include the slash (/).

any matches packets from any source IPv4 address.

dest-ipv4-addr/prefix-length | **any**

Destination IPv4 addresses. These represent the networks or hosts to receive the packets. Use one of the following methods:

dest-ipv4-addr subnet-mask matches packets going to a range of IPv4 addresses via a subnet mask. Use the *a.b.c.d* format for the IPv4 address and the subnet mask.

dest-ipv4-addr/prefix-length matches packets going to a range of IPv4 addresses via a subnet prefix length. Use the *a.b.c.d* format for the IPv4 address, and use an integer from 0 to 32 for the prefix. You must include the slash (/).

any matches packets going to any destination IPv4 address.

dontfragment | **morefragment**

(Optional) Matches only IPv4 nonfragmented or only IPv4 fragmented packets respectively.

dport *dest-l4-port*

(Optional) Destination layer 4 port. Specify an integer from 1 to 65535.

For example, 22 (for SSH traffic) or 443 (for HTTPS traffic). For details, see

<https://www.iana.org/assignments/service-names-port-numbers/service-names-port-numbers.xhtml>.

sport *source-l4-port*

(Optional) Source layer 4 port. Specify an integer from 1 to 65535. For details, see <https://www.iana.org/assignments/service-names-port-numbers/service-names-port-numbers.xhtml>.

dscp *packet-dsp-value*

(Optional) Matches packets containing a specific Differentiated Services Code Point (DSCP) value. Differentiated services are for managing traffic and providing quality of service (QoS). For example, they can provide low latency to voice or streaming media but only best effort service to web traffic or file transfers. Specify an integer from 0 to 63.

vlan *packet-vlan-id*

(Optional) Matches packets containing a specific Virtual LAN (VLAN) ID. Specify an integer from 0 to 4095. Use this argument to apply the rule to a VLAN with an ID other than 1 (the default VLAN ID). You can use this keyword if your physical LAN is partitioned into VLANs (logical broadcast domains to isolate services for improved security and management).

count

(Optional) Enables the counter for this rule. This counts the number of packets and bytes that match this rule in its access list.

To display the counters as a number of matches, see [show access-list](#) on page 937.

You must clear the counters if you want to start at zero for a more recent count of the packets and bytes that match an access list. For details, see [clear counters access-list](#) on page 508.

Modes

IPv4 ACL configuration (config-ip-acl)

Usage Guidelines

The order of the rules in an ACL is critical, because the first matching rule stops further processing. Rules that specify sequence values will determine the order of rule processing. If you do not specify a sequence value, the rule is added to the end of the list.

Traffic not specifically permitted is dropped by a default final deny rule.

Conflicting rules (ones with the same match condition but different actions) are not allowed.

To remove a rule from an IPv4 ACL, use the **no** form of this command:

- If you know the rule number, enter **no seq** *sequence-num*.

- If you *do not* know the rule number, enter **no** followed by the full syntax (without **seq sequence-num**).

The following table lists the optional TCP flags:

Table 7: Optional TCP flags

Flag	Protocol (Packet Type)
ack	Acknowledgment. These packets confirm that the data packets were received and confirm initiation requests and teardown requests. They are used along with synchronization packets for the three-way TCP handshake.
cwr	Congestion Window has been Reduced (CWR). These packets indicate that the sending host has received a TCP segment with the ECE flag set. The congestion window is an internal variable maintained by TCP to manage the size of the send window.
ece	Explicit Congestion Notification (ECN)- Echo. These packets indicate that a TCP peer is ECN-capable during the TCP three-way handshake and also indicate that a TCP segment was received on the connection with the ECN field in the IP header set to 11.
fin	Finish sending data. These packets indicate that the connection is being torn down. Both the sender and receiver send these packets to terminate the connection gracefully.
push	Push function. These packets indicate that the contents of the TCP receive buffer should be passed on to the application layer protocol directly instead of being buffered.
reset	Reset the connection. These packets indicate that the connection is down or the service is not accepting requests.
sync	Synchronization packets. These packets initiate a connection. They are used with acknowledgment packets as part of the three-way TCP handshake.
urg	These packets indicate that the data inside the packet should be processed immediately by the TCP stack.

Examples

The following example configures an IPv4 ACL named v4acl on the device and adds a deny rule to the ACL:

```
device# configure terminal
device(config)# ipv4 access-list v4acl
device(config-ip-acl)# seq 10 deny ipv4 1.1.1.1/32 2.2.2.0 255.255.255.0 count
device(config-ip-acl)#
```

The following example shows the ACLs that are configured currently on the device:

```
device# show running-config access-list

ipv4 access-list v4acl
seq 10 deny ipv4 1.1.1.1 255.255.255.255 2.2.2.0 255.255.255.0 count
device#
```


deny (IPv6 ACL configuration)

Use the **deny** command to command to configure an access control rule in an IPv6 access control list (ACL) to deny forwarding of packets.

Syntax

```
[ seq sequence-num ] deny { ip-protocol-num | ipv6 | esp | icmpv6 |
  igmpv6 | udp | tcp [ flag-list ] } { source-ipv6-addr subnet-mask
  | source-ipv6-addr/prefix-length | any } { dest-ipv6-addr subnet-
  mask | dest-ipv6-addr/prefix-length | any } [ { dontfragment |
  morefragment } ] [ dport dest-l4-port ] [ sport source-l4-port ]
  [ dscp packet-dsp-value ] [ vlan packet-vlan-id ] [ count ] [ log ]

no seq sequence-num

no deny { ip-protocol-num | ipv6 | icmpv6 | esp | igmpv6 | udp | tcp
  [ flag-list ] } { source-ipv6-addr subnet-mask | source-ipv6-addr/
  prefix-length | any } { dest-ipv6-addr subnet-mask | dest-ipv6-addr/
  prefix-length | any } [ { dontfragment | morefragment } ] [ dport
  dest-l4-port ] [ sport source-l4-port ] [ dscp packet-dsp-value ]
  [ vlan packet-vlan-id ] [ count ] [ log ]
```

Command Default

The ACL is not configured with deny rules.

Parameters

seq *sequence-num*

(Optional) Sequence number to identify the relative priority of this rule in the ACL. Specify an integer from 1 to 65535. You should specify an unused sequence number in a multiple of 10. If you do not specify a value, the rule is added to the end of the list.

ip-protocol-num

Matches packets that use a specific IP protocol. Specify an integer from 1 to 254. For example, specify 6 for TCP packets or 115 for L2TP packets. For a list of IP protocol numbers, see <https://www.iana.org/assignments/protocol-numbers/protocol-numbers.xhtml>.

ipv6

Matches packets that use any IPv6 protocol (including ICMPv6, IGMPv6, UDP, and TCP).

esp

Matches packets encrypted with Encapsulation Security Payload (ESP).

icmp

Matches packets carrying Internet Control Message Protocol v6 (ICMPv6) messages. These packets contain the control messages such as Echo Request/Echo Reply or Destination Unreachable.

igmpv6

Matches packets carrying Internet Group Management Protocol v6 (IGMPv6) messages. Hosts and adjacent routers use these on IPv6 networks to establish multicast group memberships, which optimize performance for one-to-many networking applications such as streaming video.

udp

Matches packets that use the User Datagram Protocol (UDP) format. This allows packets using connectionless protocols such as TFTP and DNS.

tcp [*flag-list*]

Matches TCP packets. This allows packets that use TCP connection-oriented protocols such as HTTPS, FTP, and Telnet. You can omit flags to match all TCP packets, or you can specify any space-separated combination of the flags listed in [Table 8](#) on page 564 below.

source-ipv6-addr subnet-mask | *source-ipv6-addr/prefix-length* | **any**

Source IPv6 addresses. These represent the networks or hosts sending the packets. Use one of the following methods:

source-ipv6-addr subnet-mask matches packets from a range of IPv6 addresses via a subnet mask. Use the *x:x::x:x* format for the IPv6 address and the subnet mask.

source-ipv6-addr/prefix-length matches packets from a range of IPv6 addresses via a subnet prefix length. Use the *x:x::x:x* format for the IPv6 address, and use an integer from 0 to 128 for the prefix. You must include the slash (/).

any matches packets from any source IPv6 address.

dest-ipv6-addr/prefix-length | **any**

Destination IPv6 addresses. These represent the networks or hosts to receive the packets. Use one of the following methods:

dest-ipv6-addr subnet-mask matches packets going to a range of IPv6 addresses via a subnet mask. Use the *x:x::x:x* format for the IPv6 address and the subnet mask.

dest-ipv6-addr/prefix-length matches packets going to a range of IPv6 addresses via a subnet prefix length. Use the *x:x::x:x* format for the IPv6 address, and use an integer from 0 to 128 for the prefix. You must include the slash (/).

any matches packets going to any destination IPv6 address.

dontfragment | **morefragment**

(Optional) Matches only IPv6 nonfragmented or only IPv6 fragmented packets respectively.

dport *dest-l4-port*

(Optional) Destination layer 4 port. Specify an integer from 1 to 65535.

For example, 22 (for SSH traffic) or 443 (for HTTPS traffic). For details, see

<https://www.iana.org/assignments/service-names-port-numbers/service-names-port-numbers.xhtml>.

sport *source-l4-port*

(Optional) Source layer 4 port. Specify an integer from 1 to 65535. For details, see <https://www.iana.org/assignments/service-names-port-numbers/service-names-port-numbers.xhtml>.

dscp *packet-dsp-value*

(Optional) Matches packets containing a specific Differentiated Services Code Point (DSCP) value. Differentiated services are for managing traffic and providing quality of service (QoS). For example, they can provide low latency to voice or streaming media but only best effort service to web traffic or file transfers. Specify an integer from 0 to 63.

vlan *packet-vlan-id*

(Optional) Matches packets containing a specific Virtual LAN (VLAN) ID. Specify an integer from 0 to 4095. Use this argument to apply the rule to a VLAN with an ID other than 1 (the default VLAN ID). You can use this keyword if your physical LAN is partitioned into VLANs (logical broadcast domains to isolate services for improved security and management).

count

(Optional) Enables the counter for this rule. This counts the number of packets and bytes that match this rule in its access list.

To display the counters as a number of matches, see [show access-list](#) on page 937.

You must clear the counters if you want to start at zero for a more recent count of the packets and bytes that match an access list. For details, see [clear counters access-list](#) on page 508.

Modes

IPv6 ACL configuration (config-ipv6-acl)

Usage Guidelines

The order of the rules in an ACL is critical, because the first matching rule stops further processing. Rules that specify sequence values will determine the order of rule processing. If you do not specify a sequence value, the rule is added to the end of the list.

Traffic not specifically permitted is dropped by a default final deny rule.

Conflicting rules (ones with the same match condition but different actions) are not allowed.

To remove a rule from an IPv6 ACL, use the **no** form of this command:

- If you know the rule number, enter **no seq** *sequence-num*.

- If you *do not* know the rule number, enter **no** followed by the full syntax (without **seq sequence-num**).

The following table lists the optional TCP flags:

Table 8: Optional TCP flags

Flag	Protocol (Packet Type)
ack	Acknowledgment. These packets confirm that the data packets were received and confirm initiation requests and teardown requests. They are used along with synchronization packets for the three-way TCP handshake.
cwr	Congestion Window has been Reduced (CWR). These packets indicate that the sending host has received a TCP segment with the ECE flag set. The congestion window is an internal variable maintained by TCP to manage the size of the send window.
ece	Explicit Congestion Notification (ECN)- Echo. These packets indicate that a TCP peer is ECN-capable during the TCP three-way handshake and also indicate that a TCP segment was received on the connection with the ECN field in the IP header set to 11.
fin	Finish sending data. These packets indicate that the connection is being torn down. Both the sender and receiver send these packets to terminate the connection gracefully.
push	Push function. These packets indicate that the contents of the TCP receive buffer should be passed on to the application layer protocol directly instead of being buffered.
reset	Reset the connection. These packets indicate that the connection is down or the service is not accepting requests.
sync	Synchronization packets. These packets initiate a connection. They are used with acknowledgment packets as part of the three-way TCP handshake.
urg	These packets indicate that the data inside the packet should be processed immediately by the TCP stack.

Examples

The following example configures an IPv6 ACL named v6acl on the device and adds a deny rule to the ACL:

```
device# configure terminal
device(config)# ipv6 access-list v6acl
device(config-ipv6-acl)# deny tcp 1::0/64 2::0 ffff::0 sync
device(config-ipv6-acl)#
```

The following example shows the ACLs that are configured currently on the device:

```
device# show running-config access-list

ip access-list v6acl
seq 10 deny tcp 1::0 ffff:ffff:ffff:ffff::0 2::0 ffff::0 sync
device#
```

deny (MAC ACL configuration)

Use the **deny** command to create or modify a rule that marks packets from a specified source MAC address or to a specified destination MAC address (or both) for rejection.

Syntax

```
[ seq sequence-num ] deny { source_address address_mask | any }
    { destination_address address_mask | any } [ etype ethernet_type ]
    [ vlan vlan_ID ] [ pcp dot1p_or_cos_value ] [ count ]

no seq sequence-num

no deny { source_address address_mask | any } { destination_address
    address_mask | any } [ etype ethernet_type ] [ vlan vlan_ID ] [ pcp
    dot1p_or_cos_value ] [ count ]
```

Command Default

The ACL is not configured with deny rules.

Parameters

seq *sequence-num*

(Optional) Sequence number to identify the relative priority of this rule in the ACL. Specify an integer from 1 to 65535. You should specify an unused sequence number in a multiple of 10. If you do not specify a value, the rule is added to the end of the list.

source-address address-mask | **any**

Source MAC address and mask. Use six groups of two hex digits separated by colons. For example, 00:00:5E:00:53:00 ff:ff:ff:ff:ff:ff.

any matches packets coming from any source MAC address.

destination-address address-mask | **any**

Destination MAC address and mask. Use six groups of two hex digits separated by colons. For example, 00:00:5E:00:53:FF ff:ff:ff:00:00:00.

any matches packets going to any destination MAC address.

etype ethernet_type

(Optional) Ethernet type. You must specify a hexadecimal number in the range of 0x600 to 0xFFFF or the **arp**, **ipv4**, or **ipv6** protocol.

pcp *dot1p_or_cos_value*

(Optional) 802.1p or class of service (CoS) value for payload compression protocol (PCP). The range is an integer from 1 to 7.

vlan *vlan_ID*

(Optional) Matches packets containing a specific Virtual LAN (VLAN) ID. Specify an integer from 0 to 4095. Use this argument to apply the rule to a VLAN with an ID other than 1 (the default VLAN ID). You can use this keyword if your physical LAN is partitioned into VLANs (logical broadcast domains to isolate services for improved security and management).

count

(Optional) Enables the counter for this rule. This counts the number of packets and bytes that match this rule in its access list.

To display the counters as a number of matches, see [show access-list](#) on page 937.

You must clear the counters if you want to start at zero for a more recent count of the packets and bytes that match an access list. For details, see [clear counters access-list](#) on page 508.

Modes

MAC ACL configuration (config-mac-acl)

Usage Guidelines

The order of the rules in an ACL is critical, because the first matching rule stops further processing. Rules that specify sequence values will determine the order of rule processing. If you do not specify a sequence value, the rule is added to the end of the list.

Traffic not specifically permitted is dropped by a default final deny rule.

Conflicting rules (ones with the same match condition but different actions) are not allowed.

To remove a rule from a MAC ACL, use the **no** form of this command:

- If you know the rule number, enter **no seq** *sequence-num*.
- If you *do not* know the rule number, enter **no** followed by the full syntax (without **seq** *sequence-num*).

Examples

The following example adds a deny rule to a MAC ACL named macacl. This example matches packets with a source MAC address and address mask 11:22:33:44:55:66 ff:ff:ff:ff:ff:ff respectively and packets with any destination MAC address with a VLAN ID of 56:

```
device# configure terminal
device(config)# mac access-list macacl
device(config-mac-acl)# deny 00:00:5E:00:53:00 ff:ff:ff:ff:ff:ff any vlan 56
device(config-mac-acl)#
```

The following example displays the currently-running ACLs and their corresponding rules. In this example, only one MAC ACL (which is named macacl) is running on the device:

```
device# show running-config access-list

mac access-list macacl
deny 00:00:5E:00:53:00 ff:ff:ff:ff:ff:ff any vlan 56
device#
```


deploy (IAH VM configuration)

Use the **deploy** command to deploy a virtual machine (VM).

Syntax

```
deploy  
no deploy
```

Command Default

The specified VM is not deployed.

Parameters

This command has no keywords or variables.

Modes

IAH VM configuration (config-iah-vm-*name*)

Usage Guidelines

Make sure that you have configured a disk for the VM before using this command.

Use the **no** form of this command to disable deployment of a VM. You must undeploy the VM before performing configuration tasks on the VM.

To display details about a specific VM, see [show iah vm](#) on page 1006.

Examples

The following example deploys a VM named Ubuntu1 on the device:

```
device# configure terminal  
device(config)# iah  
device(config-iah)# vm Ubuntu1  
device(config-iah-vm-Ubuntu1)# deploy  
device(config-iah-vm-Ubuntu1)#
```

The following example displays the IAH configuration that is running currently on the device. In this example, deployment is enabled for a VM named Ubuntu1:

```
device# show running-config iah  
  
iah  
  vm Ubuntu1  
    description Ubuntu demo vm  
    auto-boot true  
    cpus 1  
    ram 1024  
    disk sda
```

```
size 5000
format qcow2
bus sata
file disk://iah/disk1.qcow2
deploy
!
!
!
device#
```

description (BGP peer group configuration)

Use the **description** command to add a description to a BGP peer group within a Virtual Routing and Forwarding (VRF) instance.

Syntax

description *description-string*

no description *description-string*

Command Default

No description is configured for the specified BGP peer group.

Parameters

description-string

String to use as the description to associate with the peer group. Use 1 to 512 characters.

Modes

VRF BGP peer group configuration (config-vrf-bgp-pg)

Usage Guidelines

Use this command to add information about the specified BGP peer group, the neighbors within it, or any other subject.

Use the **no** form of this command to remove the description from the BGP peer group.

Examples

The following example configures a BGP peer group named `group1` under a VRF instance named `violet`. This example uses the **description** command to associate a description with `group1`:

```
device# configure terminal
device(config)# vrf violet
device(config-vrf-violet)# router bgp
device(config-vrf-bgp)# peer-group group1
device(config-vrf-bgp-pg)# description This is an underlay peer group
device(config-vrf-bgp-pg)#
```

The following example displays the configuration of a VRF instance named `violet` that is running currently on the device. In this example, the `group1` peer group has a descriptive text string assigned to it:

```
device# show running-config vrf violet
```

```
vrf violet
!  
  router bgp  
    peer-group group1  
      description This is a underlay peer group  
!  
device#
```

description (bridge domain configuration)

Use the **description** command to attach a description to a bridge domain.

Syntax

description *description-string*

no description

Command Default

The current bridge domain has no description.

Parameters

description-string

Description of the interface. Use 1 to 63 characters.

Modes

Bridge domain configuration (config-bd-*name*)

Usage Guidelines

To remove the description from the bridge domain, use the **no** form of this command.

If you want to view the status of the bridge domains on a device, see [show bridge-domain](#) on page 973.

Examples

The following example configures bridge domain 100. In this example, the **description** command attaches a description of “182:Tenant L2 Extended BD” to the bridge domain:

```
device# configure terminal
device(config)# bridge-domain 100
device(config-bd-100)# description 182:Tenant L2 Extended BD
device#
```

description (Ethernet interface configuration)

Use the **description** command to attach a description to an Ethernet interface.

Syntax

description *description-string*

no description

Command Default

The specified Ethernet interface has no description. But this also depends on how you access this information: The CLI and GNMI default this field to blank, but SNMP returns the interface name (Ethernet 0/1 for example) by default unless this is configured.

Parameters

description-string

Description of the interface. Use 1 to 63 characters.

Modes

Ethernet interface configuration (config-if-eth-*name*)

Usage Guidelines

If you want to view the status of the Ethernet interfaces on a device, see [show interface ethernet link-error-disable](#) on page 1012.

Examples

The following example configures Ethernet interface 0/1. In this example, the **description** command attaches a description of “Ethernet interface 0/1” to the interface:

```
device# configure terminal
device(config)# interface ethernet 0/1
device(config-if-eth 0/10-14)# no shutdown
device(config-if-eth 0/10-14)# description Ethernet interface 0/1
device(config-if-eth 0/10-14)# fec rs-fec
device(config-if-eth 0/10-14)# link-fault-signaling
device(config-if-eth 0/10-14)# mtu 4000
device(config-if-eth 0/10-14)# loopback-mode
device#
```

description (IAH VM configuration)

Use the **description** command to add a description to a virtual machine (VM).

Syntax

```
description description-string  
no description
```

Command Default

The current VM has no description configured.

Parameters

description-string

String to use as the description to associate with the VM. Use any combination of 1 to 64 alphanumeric characters, space, underscores, hyphens, or periods.

Modes

IAH VM configuration (config-iah-vm-*name*)

Usage Guidelines

Use this command to add information about the specified VM, its configuration, or any other subject.

Use the **no** form of this command to remove the description from a VM. Note that configuration is not allowed when the VM is in the deployed state.

To display details about a specific VM, see [show iah vm](#) on page 1006.

Examples

The following example adds a description to a VM named Ubuntu1 on the device:

```
device# configure terminal  
device(config)# iah  
device(config-iah)# vm Ubuntu1  
device(config-iah-vm-Ubuntu1)# description Ubuntu demo vm  
device(config-iah-vm-Ubuntu1)#
```

The following example displays the IAH configuration that is running currently on the device. This example uses the description command to associate a description with a VM named Ubuntu1:

```
device# show running-config iah  
  
iah  
vm Ubuntu1
```

```
description Ubuntu demo vm
auto-boot true
cpus 4
ram 1024
disk sda
    size 5000
    format qcow2
    bus sata
    file disk://iah/disk1.qcow2
!
!
!
device#
```

description (mirror configuration)

Use the **description** command to add a text description to a port mirror.

Syntax

description *description*

no description

Command Default

The current port mirror has no description.

Parameters

description

Text string that describes the port mirror. Use any combination of 1 to 512 alphanumeric characters, spaces, underscores, hyphens, or periods.

Modes

Mirror configuration (config-mirror)

Usage Guidelines

Use the **no** form of this command to remove the description from the port mirror.

Examples

The following example configures a port mirror named mirror-interface-1 and adds a description to the port mirror:

```
device# configure terminal
device(config)# mirror mirror-interface-1
device(config-mirror)# description Mirror interface 1 Ethernet 0/1
device(config-mirror)#
```

The following example shows the port mirror configuration that is running currently on the device. In this example, a port mirror named mirror-interface-1 contains a text description:

```
device# show running-config mirror

mirror mirror-interface-1
  description Mirror interface 1 Ethernet 0/1
  destination interface ethernet 0/1
device#
```

destination interface (mirror configuration)

Use the **destination interface** command to configure the destination interface for a port mirror.

Syntax

```
destination interface { ethernet interface-name | internal interface-  
                        name }  
  
no destination interface
```

Command Default

No destination interface is configured for the current port mirror.

Parameters

ethernet *interface-name*

Ethernet interface name. Use the *slot/port* format. For example, 0/1.

internal *interface-name*

Internal interface name. Use integer 0 or 1.

Modes

Mirror configuration (config-mirror)

Usage Guidelines

When mirroring is configured with different speeds on the SPAN source and destination interfaces, the regular limitation of forwarding speed is important. For example, if traffic is mirrored from a 40G interface to a 10G interface, the traffic exceeding 10G is dropped.

Do not configure a destination interface to carry normal traffic.

You cannot use the same interface as both the mirror source and destination.

Only one destination interface is allowed per mirror session.

Use the **no** form of this command to remove the destination interface from the port mirror.

Examples

The following example configures a port mirror named mirror-interface-1 on the device and specifies Ethernet interface 0/1 as the destination interface:

```
device# configure terminal
device(config)# mirror mirror-interface-1
device(config-mirror)# destination interface ethernet 0/1
device(config-mirror)#
```

The following example shows the port mirror configuration that is running currently on the device. In this example, Ethernet interface 0/1 is specified as the destination interface for a port mirror named mirror-interface-1 on the device:

```
device# show running-config mirror

mirror mirror-interface-1
  destination interface ethernet 0/1
device#
```

destination (MLAG peer configuration)

Use the **destination (MLAG peer configuration)** command to specify the IP address for the keepalive destination of the multichassis link aggregation (MLAG) peer.

Syntax

```
destination { ipv4-address | ipv6-address }  
no destination
```

Command Default

No MLAG peer keepalive destination is configured.

Parameters

ipv4-address

Unicast destination IPv4 address.

ipv6-address

Unicast destination IPv6 address.

Modes

MLAG peer keepalive configuration (config-mlag-peer-keepalive-*name*)

Usage Guidelines

You can use the **show running-config mlag** command to display the MLAG configuration that is running currently on a device.

You can use the **clear counters mlag** command to clear counters of an MLAG peer.

Examples

The following example configures an MLAG peer named peer01 and also configures IPv4 address 12.1.1.1 as its keepalive destination:

```
device# configure terminal  
device(config)# mlag  
device(config-mlag)# peer peer01  
device(config-mlag-peer-peer01)# keepalive keepalive01  
device(config-mlag-peer-keepalive-keepalive01)# destination 12.1.1.1  
device(config-mlag-peer-keepalive-keepalive01)#
```

The following example displays the MLAG configuration that is running currently on a device. In this example, MLAG is configured with a peer named peer01. The peer uses IPv4 address 12.2.2.2 as its keepalive destination:

```
device# show running-config mlag
```

```
mlag
!  
    peer peer01  
!  
    keepalive keepalive01  
        destination 12.2.2.2  
!  
device#
```

The following example displays the MLAG peer configuration on a device. In this example, MLAG is configured with a peer named mypeer2. The peer uses IPv4 address 10.20.20.1 as its keepalive destination:

```
device# show mlag peer  
  
Peer mypeer2  
=====
```

Peer State	: DOWN
MCP State	: DOWN
Role	: BACKUP
Elected MAC	: 40:88:2f:c2:4c:00
Extend Bridge Count	: 2
Peer Exception	: None

```
  
    Keepalive kpl  
    -----  
    Destination Address: 10.20.20.1  
    State                : DOWN ( Connection is Down )  
    Type                 : PRIMARY  
    Address Type         : MANUAL  
    Source Interface     : port-channel 64.0 ( 10.20.20.1 )  
    First Rx              : 0000-00-00 00:00:00  
    Last Rx               : 0000-00-00 00:00:00  
device #
```

destination (tunnel interface configuration)

Use the **destination** command to specify the IP address of a remote VTEP device to do the decapsulation of traffic on the terminating end for a specific VNID.

Syntax

destination *ip-address*

no destination *ip-address*

Command Default

The current tunnel interface has no destination VTEP IP address specified.

Parameters

ip-address

IPv4 address or IPv6 address.

- For an IPv4 address, use IPv4 notation (*A.B.C.D*). For example, 10.0.0.0.
- For an IPv6 address, use the *x::x::x::x* format. For example, 1001::1.

Modes

Tunnel interface configuration (config-tunnel-*name*)

Usage Guidelines

This command specifies the IP address of a VXLAN Tunnel Endpoint (VTEP) device (such as a switch) that does the decapsulation of VXLAN traffic on the terminating (recipient) end for a specific VXLAN Network Identifier (VNID). The destination VTEP device delivers the original Ethernet frames to the intended host or virtual machine after decapsulation.

If you want to view the status and details of the tunnel interfaces on a device, see [show interface tunnel \(Extreme ONE OS Switching\)](#) on page 1016.

To remove the destination VTEP from a tunnel interface, use the **no** form of this command.

Examples

The following example configures tunnel interface 1 to use IPv4 address 1.1.1.1 as the destination VTEP on the device:

```
device# configure terminal
device(config)# interface tunnel 1
device(config-tunnel-1)# destination 1.1.1.1
device(config-tunnel-1)#
```

The following example displays the tunnel interface configuration that is running currently on the device. In this example, the tunnel 1 interface uses IPv4 address 1.1.1.1 as the destination VTEP:

```
device# show running-config interface tunnel

interface tunnel 1
  type vxlan
    source nve base
    destination 1.1.1.1
  no shutdown
!
device#
```

disk (IAH VM configuration)

Use the **disk** command to add a disk to a virtual machine (VM) and enter IAH VM disk configuration (`config-iah-vm-disk-name`) mode.

Syntax

```
disk { hda | sda | sdb | sdc | sdd }  
no disk { hda | sda | sdb | sdc | sdd }
```

Command Default

The specified VM has no disk configured.

Parameters

```
{ hda | sda | sdb | sdc | sdd }
```

Disk name. Use a name that is unique among all other configured disks under the VM. You must use `sda` as the name of the main disk and `hda` (a CD-ROM disk), `sdb`, `sd`**c**, or `sdd` as the names of any additional disks.

Modes

IAH VM configuration (`config-iah-vm-name`)

Usage Guidelines

Use this command to enter the mode for configuring disk settings such as the size, bus type, and image format.

Multiple disks are supported.

Use the **no** form of this command to remove a disk from a VM. Note that configuration is not allowed when the VM is in the deployed state.

To display details about a specific VM, see [show iah vm](#) on page 1006.

Examples

The following example adds a disk named `sda` to a VM named `Ubuntu1` and enters IAH VM disk configuration (`config-iah-vm-disk-name`) mode on the device:

```
device# configure terminal  
device(config)# iah  
device(config-iah)# vm Ubuntu1  
device(config-iah-vm-Ubuntu1)# disk sda  
device(config-iah-vm-disk-sda)#
```


The following example displays the IAH configuration that is running currently on the device. In this example, a VM named Ubuntu2 uses a disk named sdb:

```
device# show running-config iah

iah
  vm Ubuntu2
    description Ubuntu demo vm
    auto-boot true
    cpus 4
    ram 1024
    disk sdb
      size 5000
      format qcow2
      bus sata
      file disk://iah/disk1.qcow2
    !
  !
!
device#
```

dns (IAH VM configuration)

Use the **dns** command to configure the primary and secondary DNS servers and domain name for an IAH third party virtual machine (TPVM).

Syntax

```
dns primary-server ip-address [ secondary-server ip-address ] [ domain  
    domain-name ]  
  
no dns
```

Command Default

No primary or secondary DNS servers or domain name are configured on the current VM.

Parameters

primary-server *ip-address*

IPv4 address or IPv6 address of the primary DNS server.

- For an IPv4 address, use IPv4 notation (*A.B.C.D*). For example, 10.0.0.0.
- For an IPv6 address, use the *x:x::x:x* format. For example, 1001::1.

secondary-server *ip-address*

(Optional) IPv4 address or IPv6 address of the secondary DNS server.

- For an IPv4 address, use IPv4 notation (*A.B.C.D*). For example, 10.0.0.1.
- For an IPv6 address, use the *x:x::x:x* format. For example, 1001::2.

domain-name

(Optional) Fully qualified domain name of the DNS server.

Modes

IAH VM configuration (config-iah-vm-*name*)

Usage Guidelines

Before using this command, you must first enable the network interfaces on the VM. For details, see [activate-network](#) on page 428.

To remove the primary DNS server from the current VM, use the **no** form of this command. (This also removes the secondary DNS server and the domain name if they exist.)

To display details about a specific VM, see [show iah vm](#) on page 1006.

Examples

The following example configures a primary DNS server, a secondary DNS server, and a domain name for the TPVM on the device:

```
device# configure terminal
device(config)# iah
device(config-iah)# vm TPVM
device(config-iah-vm-TPVM)# primary-server 8.8.8.8 secondary-server 9.9.9.9 domain
corp.mycompany.com
device(config-iah-vm-TPVM)#
```

The following example displays the IAH configuration that is running currently on the device. In this example, a primary DNS server, a secondary DNS server, and a domain name are specified:

```
device# show running-config iah

iah
  vm TPVM
    description TPVM
    auto-boot
    cpus 4
    ram 8192
    disk sda
      size 10240
      file /mnt/onl/iah/images/TPVM/sda.qcow2
      format QCOW2
    !
    disk sdb
      size 51200
      file /mnt/onl/iah/images/TPVM/sdb.qcow2
      format QCOW2
    !
    disk hda
      size 1
      file /mnt/onl/iah/images/TPVM/hda.raw
      format RAW
    !
    network mgmt0
      interface management 0
    !
    activate-network
    deploy
    hostname host2
    timezone Asia/Kolkata
    interface management ipv4 dhcp
    interface management ipv6 dhcp
    dns primary-server 8.8.8.8 secondary-server 9.9.9.9 domain corp.mycompany.com
    ntp 10.64.12.81
    ntp 10.64.13.105
    ntp 10.32.12.81
    ntp 10.38.99.47
    ntp time1.google.com
    ldap host 1.1.1.1 port 389 basedn dc=example,dc=com rootdn
    cn=admin,dc=example,dc=com rootdnpw=hashed hlijsN-IZKdOM56XNSnTYasO58y_NTWm
    !
  !
```

dscp traffic-class

Use the **dscp traffic-class** command to define the DSCP-to-traffic-class entries in a QoS traffic-class map.

Syntax

```
dscp dscp-value traffic-class traffic-class-value
```

```
no dscp dscp-value
```

Command Default

The specified QoS traffic-class map contains no DSCP-to-traffic-class entries.

Parameters

dscp-value

Differentiated Services Code Point (DSCP) value. Use an integer from 0 to 63.

traffic-class-value

Traffic class value. Use an integer from 0 to 7.

Modes

QoS traffic class map configuration (config-qos-traffic-class-map-*name*)

Usage Guidelines

To remove a DSCP-to-traffic-class entry from a QoS traffic-class map, use the no form of this command.

Examples

The following example configures two DSCP-to-traffic-class entries in a QoS traffic-class map named abc:

```
device# configure terminal
device(config)# qos
device(config-qos)# traffic-class-map abc
device(config-qos-traffic-class-map-abc)# dscp 1 traffic-class 3
device(config-qos-traffic-class-map-abc)# dscp 2 traffic-class 4
device(config-qos-traffic-class-map-abc)#
```

The following example displays the QoS configuration that is running currently on the device. In this example, a QoS traffic-class map named abc is configured with two DSCP-to-traffic-class entries:

```
device# show running-config qos

qos
  traffic-class-map abc
```

```
dscp 1 traffic-class 3
dscp 2 traffic-class 4
device#
```

duplex (Ethernet interface configuration)

Use the **duplex** command to set the duplex mode on an Ethernet interface.

Syntax

```
duplex { half | full | auto }  
no duplex
```

Command Default

Duplex mode on the specified Ethernet interface is set to **auto** (automatic).

Parameters

half

(SDWAN platforms only) Sets the mode to half duplex.

full

(SDWAN platforms only) Sets the mode to full duplex.

auto

Sets the mode to automatic depending on the platform and hardware. For example, for 1Gb and 10 Gb copper interfaces, the only valid duplexing is full-duplex, so the command to manually set it is not present (just the auto-negotiation for the speed). For 10Mb, 100Mb, and 1 Gb interfaces, you can force half-duplex to be compatible with older management devices that might be present in the network.

Modes

Ethernet interface configuration (config-if-eth-*name*)

Usage Guidelines

If you want to view the status of the Ethernet interfaces on a device, see [show interface ethernet link-error-disable](#) on page 1012.

Examples

The following example configures Ethernet interface 0/1. In this example, the duplex mode is set to half-duplex:

```
device# configure terminal  
device(config)# interface ethernet 0/1  
device(config-if-eth 0/10-14)# no shutdown  
device(config-if-eth 0/10-14)# description Ethernet interface 0/1  
device(config-if-eth 0/10-14)# duplex half  
device(config-if-eth 0/10-14)# fec rs-fec  
device(config-if-eth 0/10-14)# link-fault-signaling
```

```
device(config-if-eth 0/10-14)# mtu 4000
device(config-if-eth 0/10-14)# loopback-mode
device#
```

ebgp-multihop

Use the **ebgp-multihop** to enable external BGP multihop and optionally set the time-to-live (TTL) value of the IP header (hop count) for a specific external BGP neighbor or all external neighbors in the specified BGP peer group within a VRF instance.

Syntax

ebgp-multihop [*max-hop-count*]

no ebgp-multihop [*max-hop-count*]

Command Default

Only external (directly connected) neighbors are allowed.

Parameters

max-hop-count

(Optional) Maximum number of hops for the specified external BGP neighbor or all external neighbors in the specified BGP peer group. Use an integer from 2 to 255.

Modes

VRF BGP peer group configuration (config-vrf-bgp-pg)

VRF BGP peer group neighbor configuration (config-vrf-bgp-pg-neighbor)

Usage Guidelines

Use this command to accept and attempt BGP connections to one or more external peers residing on networks that are not directly connected.

This command is relevant only for external BGP neighbors.

If you specify a BGP peer group by using the **peer-group** command, all the members of the peer group will inherit the characteristic configured with this command. If you specify an individual BGP peer by using the **neighbor** command, the characteristic configured with this command applies only to that BGP peer.

Use the **no** form of this command to restore the maximum hop count to the default (one hop) for the specified external BGP neighbor or all external neighbors in the specified BGP peer group.

Examples

The following example configures two BGP peer groups (group1 and group2) under a Virtual Routing and Forwarding (VRF) instance named violet. In this example, group1

allows each of its external neighbors ten hops, and group2 allows each of its external neighbors one hop (the default value):

```
device# configure terminal
device(config)# vrf violet
device(config-vrf-violet)# router bgp
device(config-vrf-bgp)# peer-group group1
device(config-vrf-bgp-pg)# ebgp-multihop 10
device(config-vrf-bgp-pg)# exit
device(config-vrf-bgp)# peer-group group2
device(config-vrf-bgp-pg)# ebgp-multihop
device(config-vrf-bgp-pg)#
```

The following example configures a BGP peer group named group1 under a VRF instance named violet. This example allows connections to or from neighbor 10.1.1.1 and neighbor 20.1.1.1, both of which reside on a network that is not directly connected. Neighbor 10.1.1.1 is allowed one hop, and neighbor 20.1.1.1 is allowed ten hops:

```
device# configure terminal
device(config)# vrf violet
device(config-vrf-violet)# router bgp
device(config-vrf-bgp)# peer-group group1
device(config-vrf-bgp-pg)# neighbor 10.1.1.1
device(config-vrf-bgp-pg-neighbor)# ebgp-multihop
device(config-vrf-bgp-pg-neighbor)# exit
device(config-vrf-bgp-pg)# neighbor 20.1.1.1
device(config-vrf-bgp-pg-neighbor)# ebgp-multihop 10
device(config-vrf-bgp-pg-neighbor)#
```

The following example displays the configuration of a VRF instance named violet that is running currently on the device. In this example, the group1 and group2 BGP peer groups allow 10 hops and one hop respectively for all of their external neighbors:

```
device # show running-config

vrf violet
!
  router bgp
    peer-group group1
      ebgp-multihop 10
    peer-group group2
      ebgp-multihop
!
device#
```

The following example displays the configuration of a VRF instance named violet that is running currently on the device. In this example, the group1 BGP peer group has two external neighbors. Neighbor 10.1.1.1 allows one hop, and neighbor 20.1.1.1 allows ten hops:

```
device# show running-config

vrf violet
!
  router bgp
    peer-group group1
      neighbor 10.1.1.1
      ebgp-multihop
      neighbor 20.1.1.1
      ebgp-multihop 10
```

```
!  
device#
```

ecmp-max-path (global system configuration)

Use the **ecmp-max-path** command to enable ECMP routing with a specified maximum number of paths on the device.

Syntax

ecmp-max-path *paths*

no ecmp-max-path

Command Default

Equal Cost Multi Path (ECMP) routing is enabled with a maximum of 8 paths.

Parameters

paths

Number of paths. Use an integer from 2 to 128 (power of 2).

Modes

Global system configuration (config-system-global)

Usage Guidelines

This is a global (system-wide) setting.

Use the **no** form of this command to reset the maximum number of paths to the default value.

When ecmp-max-path is configured under global config and vrf instance configuration, the value configured under vrf instance takes precedence.

Examples

The following example enables ECMP routing with a maximum of 32 paths:

```
device# configure terminal
device# (config)# system
device(config-system)# global
device(config-system-global)# ecmp-max-path 32
device(config-system-global)#
```

The following example displays the configuration that is running currently on the device. In this example, ECMP routing is configured with a maximum of 32 paths:

```
device# show running-config system global ecmp-max-path
!
!
system global ecmp-max-path 32
device#
```

ecmp-max-path (VRF instance configuration)

Use the **ecmp-max-path** command to enable ECMP routing on a VRF instance with a specified maximum number of paths.

Syntax

ecmp-max-path *paths*

no ecmp-max-path

Command Default

Equal Cost Multi Path (ECMP) routing on the VRF instance is enabled with a maximum of 8 paths.

Parameters

paths

Number of paths. Use an integer from 2 to 128 (power of 2).

Modes

VRF configuration (config-vrf-*vrf-name*)

Usage Guidelines

Use the **no** form of this command to disable ECMP on the VRF instance.

When **ecmp-max-path** is configured under Global configuration (config) mode and VRF configuration (config-vrf-*vrf-name*) mode, the value configured under the VRF instance takes precedence.

Examples

The following example enables ECMP routing on a VRF instance named *vrf1* with a maximum of 32 paths:

```
device# configure terminal
device# (config)# vrf vrf1
device(config-vrf-vrf1)# ecmp-max-path 32
device(config-vrf-vrf1)#
```

The following example displays the configuration that is running currently on the device. In this example, ECMP routing is configured on a VRF instance named *vrf1* with a maximum of 32 paths:

```
device# show running-config vrf vrf1

vrf vrf1
  ecmp-max-path 32
```

```
!  
device#
```

enable (IPv6 router advertisement configuration)

Use the **enable** command to enable IPv6 router advertisement on an interface.

Syntax

```
enable  
no enable
```

Command Default

IPv6 router advertisement is enabled.

Parameters

This command has no keywords or variables.

Modes

IPv6 router advertisement configuration (config-if-*name*-ipv6-router-advertisement)

Usage Guidelines

Router advertisements are ICMPv6 messages sent by an IPv6 enabled router that are used by hosts to detect routers on an attached link. This functionality is similar to the IPv4 ICMP Router Discovery protocol.

To disable IPv6 router advertisement on an interface, use the **no** form of this command. In this case, the local system does not transmit router advertisement messages and does not respond to router solicitation messages.

Examples

The following example enables IPv6 router advertisement on Ethernet interface 0/1:

```
device# configure terminal  
device(config)# interface ethernet 0/1  
device(config-if-eth-0/1)# ipv6-router-advertisement  
device(config-if-eth-0/1-ipv6-router-advertisement)# enable  
device(config-if-eth-0/1-ipv6-router-advertisement)#
```

The following example displays the configuration of Ethernet interface 0/1 that is running currently on the device. In this example, IPv6 router advertisement is enabled on the interface:

```
device# show running-config interface ethernet 0/1  
  
!  
interface ethernet 0/1  
  ipv6 address 1111::10/64  
  ipv6 neighbor 1111::1 00:04:96:eb:c4:51  
  ipv6-router-advertisement
```

```
hop-limit 133
lifetime 200
managed-config-flag
mode disable-unsolicited-ra
other-config
enable
interval 100 min 50
prefix 100::1/64
prefix 100::1/63
    valid-lifetime 200
!
no shutdown
!
device#
```

enable-bfd

Use the **enable-bfd** command to enable Bidirectional Forwarding Detection (BFD) for a specified BGP neighbor or all neighbors in the specified BGP peer group within a Virtual Routing and Forwarding (VRF) instance.

Syntax

```
enable-bfd [ profile profile-name ]  
no enable-bfd
```

Command Default

BFD is enabled with the default profile for the specified neighbor or all neighbors in the specified BGP peer group.

Parameters

profile*profile-name*

(Optional) Name of the profile to use. If you do not specify a profile name, the default profile is used.

Modes

VRF BGP peer group configuration (config-vrf-bgp-pg)

VRF BGP peer group neighbor configuration (config-vrf-bgp-pg-neighbor)

Usage Guidelines

This command creates BFD sessions to track a specified BGP neighbor or all BGP neighbors in a specified BGP peer group. BFD sends BFD control packets between a pair of neighbors to activate and maintain BFD sessions between them. For BFD sessions to be created, you must enable BFD on both BGP neighbors. Alternatively, you can enable BFD at the peer group level, which enables BFD for all the neighbors in the peer group.

BGP does not wait for the respective session to come up before sending registration. As soon as you configure **enable-bfd** under a peer (or a peer group), the registration request is sent to BFD. Registration remains intact until **enable-bfd** is removed. However, this behavior differs for dynamic peers.

As dynamic peers are formed based on remote device requests, you cannot configure **enable-bfd** under **listen-range** but only under a peer group. Hence, the registration request for any dynamic neighbor is sent to BFD whenever this dynamic peer is formed under this specific peer group (which has **enable-bfd** configured). Also, along similar lines, when a dynamic peer goes down, the deregistration happens immediately for this peer, because the peer ceases to exist because it is dynamic.

When BFD declares an Oper-Path-Down to BGP, that specific session (or sessions if enabled under a peer group), are brought down immediately for faster convergence. But after this, if BGP finds the TCP session is still reachable, BGP tries to establish a session and succeed with or without a BFD status of Oper-PATH-DOWN. A session flap will occur after the initial Oper-PATH-DOWN.

Subsequent Oper-Path-UP events are ignored if the session is already established because of TCP reachability.

An Admin-Down event is not considered for any session alteration.

BGP withdraws routes as the session terminates for any reason.

Use the **no** form of this command to disable BFD tracking for the specified BGP neighbor or all neighbors in the specified BGP peer group.

Examples

The following example configures a BGP peer group named `group1` under a VRF instance named `violet`. The **enable-bfd** command is used in BGP peer group configuration mode and therefore enables BFD for all neighbors in the peer group:

```
device# configure terminal
device(config)# vrf violet
device(config-vrf-violet)# router bgp
device(config-vrf-bgp)# peer-group group1
device(config-vrf-bgp-pg)# enable-bfd profile profile1
device(config-vrf-bgp-pg)#
```

The following example configures a BGP peer group named `group1` under a VRF instance named `violet`. The **enable-bfd** command enables BFD only for the `10.1.1.1` neighbor within the peer group:

```
device# configure terminal
device(config)# vrf violet
device(config-vrf-violet)# router bgp
device(config-vrf-bgp)# peer-group group1
device(config-vrf-bgp-pg)# neighbor 10.1.1.1
device(config-vrf-bgp-pg-neighbor)# enable-bfd profile profile1
device(config-vrf-bgp-pg-neighbor)#
```

The following example displays the configuration of a VRF instance named `violet` that is running currently on the device. This example configures a BGP peer group named `group1` and enables BFD for all neighbors in the peer group:

```
device # show running-config vrf violet

vrf violet
!
  router bgp
    peer-group group1
      enable-bfd profile profile1
!
device#
```

The following example displays the configuration of a VRF instance named violet that is running currently on the device. This example configures a BGP peer group named group1 and enables BFD only for neighbor 10.1.1.1 within the peer group:

```
device# show running-config vrf violet

vrf violet
!
  router bgp
    peer-group group1
      neighbor 10.1.1.1
        enable-bfd profile profile1
!
device#
```

enable-on-reload

Use the **enable-on-reload** command to configure the device to enter maintenance mode upon system reload.

Syntax

```
enable-on-reload  
no enable-on-reload
```

Command Default

Maintenance mode upon system reload is disabled.

Parameters

This command has no keywords or variables.

Modes

Maintenance system configuration (config-system-maintenance-mode)

Usage Guidelines

You use the maintenance mode feature when a device needs maintenance for reasons such as an image upgrade. When maintenance mode is active, all front panel ports are disabled to isolate the device from all connected devices on the network.

During maintenance mode, all physical ports will be in the "link down" state. The connected devices detect that the link is down and divert the traffic to a peer multichassis link aggregation (MLAG) node.

This command configures the device to enter maintenance mode upon system reload (reboot). This command disables all front panel ports when the device comes back up.

You can also enable maintenance mode on a running device. For details, see [system maintenance-mode](#) on page 1140.

To disable this feature, use the **no** form of this command.

Examples

The following example enables maintenance mode when the device reloads:

```
device# configure terminal  
device(config)# system  
device(config-system)# maintenance-mode  
device(config-system-maintenance-mode)# enable-on-reload  
device(config-system-maintenance-mode)#
```

The following example displays the maintenance mode details of the device. In this example, `Enable on reload : enabled` indicates that maintenance mode is configured to start upon system reload:

```
device# show system maintenance-mode

Enable on reload      : enabled
Maintenance Mode      : enabled
Maintenance Mode Status : Complete
device#
```

The following example displays the system configuration that is running currently on the device. In this example, maintenance mode is configured so that the device enters maintenance mode upon system reload:

```
device# show running-config system maintenance-mode

system
  maintenance-mode
    enable-on-reload
  !
!
device#
```

encapsulation (QoS NVO configuration)

Use the **encapsulation** command to set the fabric QoS tunnel encapsulation mode for a Network Virtualization Overlay (NVO) endpoint.

Syntax

```
encapsulation { uniform | pipe [ dscp [ dscp_value ] ] }
```

Command Default

Fabric QoS tunnel encapsulation on the NVO endpoint is set to uniform mode.

Parameters

uniform

Sets fabric QoS tunnel encapsulation to uniform mode. With uniform mode, the core network's QoS policy is applied to inner packets (even after they exit the tunnel), which enables centralized QoS control in the network.

pipe

Sets fabric QoS tunnel encapsulation to pipe mode. With pipe mode, traffic is untouched when traversing the fabric (the inner packet QoS settings are preserved throughout transmission to ensure end-to-end consistency).

dscp [*dscp_value*]

(Optional) Differentiated Services Code Point (DSCP) value (which is contained in the IP header and is used to prioritize and manage network traffic). For *dscp_value*, use an integer from 0 to 63. The default is 9.

Modes

QoS NVO configuration (config-qos-nvo-*name*)

Usage Guidelines

This command configures fabric QoS tunnel mode encapsulation for an NVO on a device. The two primary tunneling modes manage how packets are encapsulated and how their QoS marking is handled:

- In uniform mode encapsulation, the QoS marking of the inner packets are copied to the outer MPLS headers when the packets are encapsulated into the MPLS tunnel. This means that the tunnel reflects the original QoS markings of the packets. Uniform mode encapsulation ensures that the inner packet QoS markings are applied to the entire path (including through the tunnel) to ensure consistent QoS treatment throughout the entire network.
- In pipe mode encapsulation, the QoS markings (DSCP bits) of the inner packets remain separate from the outer MPLS tunnel header. The outer headers have their own QoS markings independent of the inner packet QoS markings. Pipe mode

encapsulation enables different QoS policies within the tunnel and for end-to-end traffic to let the core network apply its own QoS treatment while preserving the original QoS markings.

To restore the encapsulation setting to the default value on the device, use the **no** form of this command.

To configure fabric QoS tunnel mode *decapsulation* for an NVO on a device, see [decapsulation \(QoS NVO configuration\)](#) on page 551.

Examples

The following example enters QoS NVO tunnel mode configuration (config-qos-nvo-*name*-tunnel-mode) mode and configures fabric QoS tunnel encapsulation and decapsulation on the device:

```
device# configure terminal
device(config)# qos
device(config-qos)# nvo mynvo
device(config-qos-nvo-mynvo)# tunnel-mode
device(config-qos-nvo-mynvo-tunnel-mode)# encapsulation pipe dscp 34
device(config-qos-nvo-mynvo-tunnel-mode)# decapsulation uniform
device(config-qos-nvo-mynvo-tunnel-mode)#
```

The following example shows the QoS configuration that is running currently on the device. In this example, a QoS node has an associated NVO instance named mynvo with QoS tunnel encapsulation and decapsulation configured:

```
device# show running-config qos

qos
  nvo mynvo
    tunnel-mode
      encapsulation pipe dscp 34
      decapsulation uniform
    !
  !
!
device#
```

evpn-instance (VRF configuration)

Use the **evpn-instance** command to associate an EVPN instance with the default Virtual Routing and Forwarding (VRF) instance and enter EVPN configuration mode.

Syntax

evpn-instance *instance-name*

no evpn-instance *instance-name*

Parameters

instance-name

Name of the EVPN instance to associate. You can use any combination of 1 to 64 uppercase or lowercase alphabetic characters, numerals, underscores, hyphens, or periods.

Modes

VRF configuration (config-vrf-*vrf-name*)

Usage Guidelines

Configuration of an EVPN instance is supported only for the default VRF (not for any other user-configured VRF instance).

Use the **no** form of this command to delete the association of an EVPN instance from a specific VRF.

Examples

The following example associates an EVPN instance named `tenant_1` with a VRF named `default-vrf` and enters EVPN configuration mode:

```
device# configure terminal
device(config)# vrf default-vrf
device(config-vrf-default-vrf)# evpn-instance tenant_1
device(config-evpn-tenant_1)#
```

The following example displays the VRF instances (and their associated EVPN instances) that are running currently on the device:

```
device# show running-config

...
vrf default-vrf
    evpn-instance tenant_1
...
device#
```

export-policy (BGP peer group address family configuration)

Use the **export-policy** command to configure the outbound policy for an address family in a specific peer group within a routing process in a VRF instance.

Syntax

```
export-policy policy-name  
no export-policy
```

Command Default

The address family in the specified peer group has no export policy configured.

Parameters

policy-name

Name of the export policy to use with the address family.

Modes

VRF BGP peer group address family configuration (config-vrf-bgp-pg-*address-family-type*)

Usage Guidelines

Use this command to apply a set of rules to filter, modify, or select the BGP routes before advertising them to a group of BGP peers for a specific route type (address family). This policy is used to enforce routing control and policy compliance at scale.

This command applies only to IPv4 unicast or IPv6 unicast address families.

Use the **no** form of this command to restore the default functionality.

Examples

The following example configures a routing process under a VRF instance named violet. This example specifies an IPv4 unicast address family in a peer group named group1 and also specifies an export policy named exportpolicy1 for the address family:

```
device# configure terminal  
device(config)# vrf violet  
device(config-vrf-violet)# router bgp  
device(config-vrf-bgp)# peer-group group1  
device(config-vrf-bgp-pg)# address-family ipv4 unicast  
device(config-vrf-bgp-pg-ipv4u)# export-policy exportpolicy1  
device#
```

The following example displays the configuration of a VRF instance named violet that is running currently on the device. In this example, a peer group named group1

is configured. An IPv4 unicast address family that uses an export policy named `exportpolicy1` is specified in this example:

```
device# show running-config vrf violet

vrf violet
  router bgp
    address-family ipv4 unicast
      send-default-route
    !
    peer-group group1
      remote-as 200
      address-family ipv4 unicast
        export-policy myexportpolicy1
      !
    !
  !
!
device#
```

ext-community-set

Use the **ext-community-set** command to configure a BGP extended community set and enter Routing policy BGP extended community set configuration (config-route-policy-bgp-ext-community-*name*) mode.

Syntax

ext-community-set *set-name*

no ext-community-set *set-name*

Command Default

No BGP extended community sets are configured on the device.

Parameters

set-name

Name of the BGP extended community set to configure. This is used to reference the extended community set in match conditions. Use any combination of uppercase or lowercase alphabetic characters, numerals, underscores, or hyphens. You must use an alphabetic character as the first character.

Modes

Routing policy configuration (config-route-policy)

Usage Guidelines

Use this command to create a named BGP extended community set and enter the mode for configuring the extended community set.

To remove a BGP extended community set from the device, use the **no** form of this command.

Examples

The following example creates a BGP extended community set named ecomm1:

```
device# configure terminal
device(config)# route-policy
device(config-route-policy)# bgp-defined-sets
device(config-route-policy-bgp-set)# ext-community-set ecomm1
device(config-route-policy-bgp-ext-community-ecomm1)# match-set-options all
device(config-route-policy-bgp-ext-community-ecomm1)# member 200:1234 route-
target:303:5000 10.1.1.10:345
device(config-route-policy-bgp-ext-community-ecomm1)#
```

The following example displays the routing policy configuration that is running currently on the device. In this example, the device is configured with one BGP defined set, which is an extended community set named ecomm1:

```
device# show running-config route-policy

route-policy
  bgp-defined-sets
    ext-community-set ecomm1
      member 200:1234 route-target:303:5000 10.1.1.10:345
      match-set-options all
device#
```

extend-bd (MLAG configuration)

Use the **extend-bd** command to extend the bridge domains across the multichassis link aggregation (MLAG) peer.

Syntax

```
extend-bd { all | none | subset bridge-domain-range | except bridge-domain-range }  
no extend-bd [ bridge-domain-range ]
```

Command Default

All configured MLAG bridge domains on the device are extended.

Parameters

all

All configured bridge domains within MLAG. This keyword extends all existing bridge domains to other peer.

none

No configured bridge domains are extended across the MLAG peer.

subset *bridge-domain-range*

Subset of configured bridge domains. This keyword extends the bridge domain list provided via *bridge-domain-range*. You can specify a single bridge domain or one or more comma-separated ranges of bridge domains. For example, 2 or 1,5,100-200 respectively. The valid range is 1 to 8192.

except *bridge-domain-range*

All configured bridge domains except those specified. This keyword extends all bridge domains except those in the list provided via *bridge-domain-range*. You can specify a single bridge domain or one or more comma-separated ranges of bridge domains. For example, 3 or 7,100-700 respectively. The valid range is 1 to 8192.

Modes

MLAG peer configuration (config-mlag-peer-*name*)

Usage Guidelines

Every MLAG bridge domain that you configure is extended by default. You can use this command to limit the bridge domain extension functionality to specific bridge domains.

To remove a range of extended bridge domains or all extended bridge domains for MLAG from a device, use the **no** form of this command.

Examples

The following example configures the subset extend bridge domain mode with the range of bridge domains. In this example, the bridge domains with IDs 1,5,100-200 are extended to other MLAG peer:

```
device# configure terminal
device(config)# mlag
device(config-mlag)# peer peer01
device(config-mlag-peer-peer01)# extend-bd subset 1,5,100-200
device(config-mlag-peer-peer01)#
```

The following example configures the subset extend bridge domain mode with the range of bridge domains. In this example, all bridge domains except those with IDs 7,100-700 are extended to other MLAG peer:

```
device# configure terminal
device(config)# mlag
device(config-mlag)# peer peer01
device(config-mlag-peer-peer01)# extend-bd except 7,100-700
device(config-mlag-peer-peer01)#
```

The following example configures a peer named peer01 and sets the extend BD mode to the default value (all):

```
device# configure terminal
device(config)# mlag
device(config-mlag)# peer peer01
device(config-mlag-peer-peer01)# extend-bd all
device(config-mlag-peer-peer01)#
```

The following example deletes the range of bridge domains that are numbered 50 to 75 from the existing list:

```
device# configure terminal
device(config)# mlag
device(config-mlag)# peer peer01
device(config-mlag-peer-peer01)# no extend-bd 50-75
device(config-mlag-peer-peer01)#
```

The following example displays the MLAG configuration that is running currently on the device. In this example, MLAG is configured with an MLAG peer named mlag_peer. The MLAG peer extends bridge domain numbers 1, 5, 1000, as well as bridge domains that are numbered 100 to 200:

```
device# show running-config mlag

mlag
!
  peer mlag_peer
    extend-bd subset 1,5,1000,1000
device#
```

The following example displays the status and other information about the peer that is configured on the device:

```
device# show mlag peer

Peer DUT-14
=====
Peer State           : UP
MLAG State           : UP
Role                 : BACKUP
Elected MAC         : b4:a3:bd:0d:dc:00
Extend Bridge Count  : 579
device#
```

The below example shows the extended bridge domain configuration for peer dut6. Extended bridge domain mode is not displayed here. It lists only bridge domains that are extended to other peer and the total count.

```
device# show mlag peer extended-bridges

Peer dut6
=====
No. of Extend Bridge: 37
Extend Bridges      :
1,11,100,200-202,205,207,210,215-216,300-302,400-402,500-506,1001,2001,2500-2501,3003-3006
,4000-4003,8192
device#
```

fast-external-failover

Use the **fast-external-failover** command to quickly terminate the EBGP session when a directly-connected link to the EBGP peer goes down, without waiting for the hold-down timer to expire.

Syntax

```
fast-external-failover  
no fast-external-failover
```

Command Default

Fast external failover is disabled in the current BGP peer group.

Parameters

This command has no keywords or variables.

Modes

VRF BGP peer group configuration (config-vrf-bgp-pg)

Usage Guidelines

This command resets the BGP session if the link to an External BGP (EBGP) peer in the specified BGP peer group is lost.

Use the **no** form of this command to reset the *fast-external-failover* setting to the default value.

Examples

The following example configures a BGP peer group named group1 under a Virtual Routing and Forwarding (VRF) instance named violet. In this example, the BGP session is reset if the link to an EBGP peer goes down:

```
device# configure terminal  
device(config)# vrf violet  
device(config-vrf-violet)# router bgp  
device(config-vrf-bgp)# peer-group group1  
device(config-vrf-bgp-pg)# fast-external-failover  
device(config-vrf-bgp-pg)#
```

The following example displays the configuration of a VRF instance named violet that is running currently on the device. In this example, the BGP session is reset if the link to an EBGP peer in a peer group named group1 is lost:

```
device# show running-config vrf violet  
  
vrf violet
```

```
!  
router bgp  
  peer-group group1  
    allow-own-as 10  
    fast-external-failover  
!  
device#
```


fec (Ethernet interface configuration)

Use the **fec** command to set the forward error correction (FEC) mode on an Ethernet interface.

Syntax

```
fec { fc-fec | rs-fec | auto-negotiation | disabled }
```

Command Default

FEC on the specified Ethernet interface is auto-negotiated.

Parameters

fc-fec

Sets the mode to Fire Code Forward Error Correction (FC-FEC) in manual mode.

rs-fec

Sets the mode to Reed-Solomon Forward Error Correction (RS-FEC) in manual mode.

auto-negotiation

Sets the mode to automatic. This mode auto-negotiates FEC if the other connected device also auto-negotiates FEC.

disabled

Disables FEC.

Modes

Ethernet interface configuration (config-if-eth-*name*)

Usage Guidelines

FEC adds an error-correcting code to a message ("payload") so that this can be recovered by a receiver even when multiple errors appear during transmission. FEC is used with 25G and higher data rates to improve throughput without needing re-transmission.

FC-FEC uses weaker error correction but lower latency than RS-FEC. For error free transmission, the same type of FEC must be used on each end of a link.

This command is available only on ports with 400G, 100G, or 25G speeds.

FC-FEC supports only 25G speed.

Only RS-FEC and AutoFEC work on 400G speed interfaces.

FEC Disable does not work on 4x100G interfaces.

On 4x100G breakout, FEC should be RS-FEC, and the peer end 100G interface should be set to FEC disable, and the links come up with FEC disabled as the link FEC operational status. This is because the 100G-DR optic receives 100G-PAM4 RS-FEC and converts it to 100G-NRZ with no FEC.

If you want to view the status of the Ethernet interfaces on a device, see [show interface ethernet link-error-disable](#) on page 1012.

Examples

The following example configures Ethernet interface 0/1. In this example, the FEC mode is set to RS-FEC:

```
device# configure terminal
device(config)# interface ethernet 0/1
device(config-if-eth 0/10-14)# no shutdown
device(config-if-eth 0/10-14)# description Ethernet interface 0/1
device(config-if-eth 0/10-14)# fec rs-fec
device(config-if-eth 0/10-14)# link-fault-signaling
device(config-if-eth 0/10-14)# mtu 4000
device(config-if-eth 0/10-14)# loopback-mode
device#
```

The following example displays details about Ethernet interface 0/1. In this example, the FEC mode is set to RS-FEC:

```
device# show interface ethernet 0/1

ethernet 0/1 Admin state UP, Oper state UP
Interface index is 16777248 (0x1000020)
MTU 9216 bytes
Hardware is Ethernet mac address 50:0a:9c:97:48:1b
Current Speed 400G
Description: Ethernet 0/1
FEC:
    Mode: RS-FEC
    Uncorrected Blocks: 0
    Corrected Blocks: 2199
Link Fault Signaling: ON
Link Fault Status: No Fault
Auto Negotiation: ON
Carrier Transitions: 1
    LastClear: 0s
Input:
    Total pkts: 0
    Broadcast pkts: 0
    Discard pkts: 0
    Errors pkts: 0
    FCS Errors: 0
    MCast pkts: 0
    Octets: 0
    UCast pkts: 0
    Runt pkts: 0
    CRC Errors: 0
Input Distribution:
    64 byte pkts: 0
    65-127 byte pkts: 0
    128-255 byte pkts: 0
    256-511 byte pkts: 0
    512-1023 byte pkts: 0
    1024-1518 byte pkts: 0
```

```
Out:          Jumbo pkts: 0
              Total pkts: 2
              Broadcast pkts: 0
              Discard pkts: 0
              Errors pkts: 0
              MCast pkts: 2
              Octets: 318
              UCast pkts: 0
Rate Info:
              Input: 0.000000 Mbits/sec, 0 pkts/sec 0.00% of line-rate
              Output: 0.001432 Mbits/sec, 1 pkts/sec 0.00% of line-rate
device#
```

file (IAH VM disk configuration)

Use the **file** command to specify a disk file for a virtual machine (VM).

Syntax

```
file URL  
no file
```

Command Default

The specified VM has no file specified for the disk.

Parameters

URL

URL that points to the file. Use one of the following values:

- **usb://***filename*
- **disk://iah/***filename*

Modes

IAH VM disk configuration (config-iah-vm-disk-*name*)

Usage Guidelines

Use the **no** form of this command to remove the disk file name from the specified VM. Note that configuration is not allowed when the VM is in the deployed state.

To display details about a specific VM, see [show iah vm](#) on page 1006.

Examples

The following example specifies disk://iah/disk1.qcow2 as the file for a disk named sda within a VM named Ubuntu1:

```
device# configure terminal  
device(config)# iah  
device(config-iah)# vm Ubuntu1  
device(config-iah-vm-Ubuntu1)# disk sda  
device(config-iah-vm-disk-sda)# file disk://iah/disk1.qcow2  
device(config-iah-vm-disk-sda)#
```

The following example displays the IAH configuration that is running currently on the device. In this example, a VM named Ubuntu1 uses a disk://iah/disk1.qcow2 file for a disk named sda:

```
device# show running-config iah  
  
iah
```

```
vm Ubuntu1
  description Ubuntu demo vm
  auto-boot true
  cpus 4
  ram 1024
  disk sda
    size 5000
    format qcow2
    bus scsi
    file disk://iah/disk1.qcow2
  !
!
device#
```

format (IAH VM disk configuration)

Use the **format** command to specify a disk file format for a virtual machine (VM).

Syntax

```
format { qcow2 | raw | vmdk }  
no format
```

Command Default

The qcow2 file format is used.

Parameters

```
{ qcow2 | raw | vmdk }  
Disk file format.
```

Modes

IAH VM disk configuration (config-iah-vm-disk-*name*)

Usage Guidelines

Use the **no** form of this command to restore the disk file format to the default value. Note that configuration is not allowed when the VM is in the deployed state.

To display details about a specific VM, see [show iah vm](#) on page 1006.

Examples

The following example specifies qcow2 as the file format for a disk named sda within a VM named Ubuntu1:

```
device# configure terminal  
device(config)# iah  
device(config-iah)# vm Ubuntu1  
device(config-iah-vm-Ubuntu1)# disk sda  
device(config-iah-vm-disk-sda)# format qcow2  
device(config-iah-vm-disk-sda)#
```

The following example displays the IAH configuration that is running currently on the device. In this example, a VM named Ubuntu2 uses the qcow2 disk file format for a disk named sdb:

```
device# show running-config iah  
  
iah  
  vm Ubuntu2  
    description Ubuntu demo vm  
    auto-boot true
```

```
cpus 4
ram 1024
disk sdb
    size 5000
    format qcow2
    bus scsi
    file disk://iah/disk1.qcow2
!
!
!
device#
```

graceful-restart (BGP address family configuration)

Use the **graceful-restart** command to ensure that when graceful restart is enabled at the VRF-BGP level, the capability is advertised to the peer for that particular address-family in the BGP routing process.

Syntax

```
graceful-restart  
no graceful-restart
```

Command Default

Graceful restart mode is disabled for the specified address family in the BGP routing process.

Parameters

This command has no keywords or variables.

Modes

VRF BGP address family configuration (*config-vrf-bgp-address-family-type*)

Usage Guidelines

Graceful restart allows a routing device undergoing a restart to inform its adjacent neighbors of its condition. This command helps to ensure that no route and topology changes occur in the network during a graceful restart.

Use the **no** form of this command to disable graceful restart mode for the specified address family.

Examples

The following example configures a routing process within a VRF instance named violet. This example enables graceful restart mode for the IPv4 unicast address family:

```
device# configure terminal  
device(config)# vrf violet  
device(config-vrf-violet)# router bgp  
device(config-vrf-bgp)# address-family ipv4 unicast  
device(config-vrf-bgp-ipv4u)# graceful-restart  
device(config-vrf-bgp-ipv4u)#
```


The following example displays the configuration of a VRF instance named violet that is running currently on the device. In this example, the routing process is configured to enable graceful restart for the IPv4 unicast address family:

```
device# show running-config vrf violet

vrf violet
  router bgp
    address-family ipv4 unicast
      graceful-restart
    !
  !
!
device#
```

graceful-restart (BGP configuration)

Use the **graceful-restart** command to enable BGP neighbors in the BGP routing process within a Virtual Routing and Forwarding (VRF) instance to participate in graceful restarts and enter VRF BGP graceful restart configuration (config-vrf-bgp-gr) mode.

Syntax

```
graceful-restart  
no graceful-restart
```

Command Default

Graceful restart is disabled on the BGP neighbors in the BGP routing process.

Parameters

This command has no keywords or variables.

Modes

VRF BGP configuration (config-vrf-bgp)

Usage Guidelines

Graceful restart allows a device undergoing a restart to inform its adjacent neighbors of its condition. This command helps to ensure that no route and topology changes occur in the network during a graceful restart.

To disable graceful restart, use the **no** form of this command.

Graceful restart (that is, a restart time combined with a stale route time) and helper-only mode are mutually exclusive. Helper-only mode is the ability to assist a neighbor attempting a graceful restart. If graceful restart mode is enabled, you must disable it before enabling helper-only mode. Conversely, if helper-only mode is enabled, you must disable it before enabling graceful restart mode. For details, see [helper-only \(BGP graceful restart configuration\)](#) on page 634.

Examples

The following example configures a routing process within a VRF instance named violet. This example enables graceful restart on its neighbors and configures a restart timer and a stale route time of 100 seconds and 500 seconds respectively:

```
device# configure terminal  
device(config)# vrf violet  
device(config-vrf-violet)# router bgp
```

```
device(config-vrf-bgp)# graceful-restart
device(config-vrf-bgp-gr)# restart-time 100
device(config-vrf-bgp-gr)# stale-route-time 500
device(config-vrf-bgp)#
```

The following example configures a routing process within a VRF instance named violet. This example enables the device to act as helper-only for its neighbors:

```
device# configure terminal
device(config)# vrf violet
device(config-vrf-violet)# router bgp
device(config-vrf-bgp)# graceful-restart
device(config-vrf-bgp-gr)# helper-only
device(config-vrf-bgp)#
```

The following example displays the configuration of a VRF instance named violet that is running currently on the device. In this example, the routing process is configured to enable graceful restart (with a restart timer and a stale route time of 100 seconds and 500 seconds respectively) on its neighbors:

```
device# show running-config vrf violet

vrf violet
  router bgp
    graceful-restart
      restart-time 100
      stale-route-time 500
!
device#
```

graceful-restart (BGP peer group address family configuration)

Use the **graceful-restart** command to enable graceful restart for a specific address family in a BGP peer group within a Virtual Routing and Forwarding (VRF) instance.

Syntax

```
graceful-restart  
no graceful-restart
```

Parameters

This command has no keywords or variables.

Modes

VRF BGP peer group address family configuration (*config-vrf-bgp-pg-address-family-type*)

Usage Guidelines

Use the **no** form of this command to disable graceful restart for the specified address family.

Examples

The following example configures two BGP peer groups (group1 and group2) under a VRF instance named violet. Graceful restart is enabled for both address families in both peer groups:

```
device# configure terminal  
device(config)# vrf violet  
device(config-vrf-violet)# router bgp  
device(config-vrf-bgp)# peer-group group1  
device(config-vrf-bgp-pg)# address-family ipv4 unicast  
device(config-vrf-bgp-pg-ipv4u)# graceful-restart  
device(config-vrf-bgp-pg-ipv4u)# exit  
device(config-vrf-bgp-pg)# address-family l2vpn evpn  
device(config-vrf-bgp-pg-l2vpn-evpn)# graceful-restart  
device(config-vrf-bgp-pg-l2vpn-evpn)# exit  
device(config-vrf-bgp-pg)# exit  
device(config-vrf-bgp)# peer-group group2  
device(config-vrf-bgp-pg)# address-family ipv6 unicast  
device(config-vrf-bgp-pg-ipv6u)# graceful-restart  
device(config-vrf-bgp-pg-ipv6u)# exit  
device(config-vrf-bgp-pg)# address-family l2vpn evpn  
device(config-vrf-bgp-pg-l2vpn-evpn)# graceful-restart  
device(config-vrf-bgp-pg-l2vpn-evpn)#
```

The following example displays the configuration of a VRF instance named violet that is running currently on the device. In this example, the instance contains two BGP peer

groups (group1 and group2), and graceful restart is enabled for both address families under each group:

```
device# show running-config vrf

vrf violet
!
  router bgp
    peer-group group1
      address-family ipv4 unicast
        graceful-restart
      address-family l2vpn evpn
        graceful-restart
    peer-group group2
      address-family ipv6 unicast
        graceful-restart
      address-family l2vpn evpn
        graceful-restart
  !
device#
```

graceful-restart (BGP peer group configuration)

Use the **graceful-restart** command to enable a specific BGP neighbor or all neighbors in a specific BGP peer group within a Virtual Routing and Forwarding (VRF) instance to participate in graceful restarts.

Syntax

```
graceful-restart { restart-time seconds stale-route-time seconds |  
                  helper-only }  
  
no graceful-restart { restart-time seconds stale-route-time seconds |  
                    helper-only }
```

Command Default

Graceful restart mode (including helper mode) is disabled on the specified BGP neighbor or all neighbors in the specified BGP peer group.

Parameters

restart-time *seconds*

Duration of the BGP graceful restart period. Use an integer from 1 to 3600 seconds. This is the amount of time that the neighbor or neighbors wait to delete the stale routes from the restarting peer that they are retaining before a BGP open message is received. The stale route time must exceed the restart time.

stale-route-time *seconds*

Amount of time that the neighbor or neighbors wait to receive messages from a restarting neighbor before declaring it down. Use an integer from 1 to 3600 seconds. This is the amount of time for which the restarting speaker must defer the route selection for an address family until an End-of-RIB (EOR) marker is received from all its peers (excluding the ones with the "Restart State" bit set in the received capability and also excluding the ones that do not advertise the graceful restart capability).

helper-only

Enables the BGP graceful restart helper-only mode.

Modes

VRF BGP peer group configuration (config-vrf-bgp-pg)

Usage Guidelines

Graceful restart allows a routing device undergoing a restart to inform its adjacent neighbors of its condition. This command helps to ensure that no route and topology changes occur in the network during a graceful restart. Helper mode is the ability to assist a neighbor attempting a graceful restart.

Use the **no** form of this command to disable graceful restart mode or helper-only mode.

Graceful restart mode (that is, a restart time combined with a stale route time) and helper-only mode are mutually exclusive. If graceful restart mode is enabled, you must disable it before enabling helper-only mode. Conversely, if helper-only mode is enabled, you must disable it before enabling graceful restart mode.

Examples

The following example configures a BGP peer group named `group1` under a VRF instance named `violet`. This example enables graceful restart mode on all neighbors in `group1` and configures a restart timer and a stale route time of 100 seconds and 500 seconds respectively:

```
device# configure terminal
device(config)# vrf violet
device(config-vrf-violet)# router bgp
device(config-vrf-bgp)# peer-group group1
device(config-vrf-bgp-pg)# graceful-restart
device(config-vrf-bgp-pg-gr)# restart-time 100
device(config-vrf-bgp-pg-gr)# stale-route-time 500
device(config-vrf-bgp-pg-gr)#
```

The following example configures a BGP peer group named `group1` under a VRF instance named `violet`. This example enables helper-only mode on all neighbors in `group1`:

```
device# configure terminal
device(config)# vrf violet
device(config-vrf-violet)# router bgp
device(config-vrf-bgp)# peer-group group1
device(config-vrf-bgp-pg)# graceful-restart
device(config-vrf-bgp-pg-gr)# helper-only
device(config-vrf-bgp-pg-gr)#
```

The following example displays the configuration of a VRF instance named `violet` that is running currently on the device. In this example, the `group1` peer group is configured to enable graceful restart (with a restart timer and a stale route time of 100 seconds and 500 seconds respectively) on its neighbors, and the `group2` peer group is composed of helper-only neighbors:

```
device# show running-config vrf violet

vrf violet
!
  router bgp
    address-family ipv4 unicast
    peer-group group1
      graceful-restart
        restart-time 100
        stale-route-time 500
    peer-group group2
      graceful-restart
        helper-only
    !
  !
!
```

```
!  
device#
```


hello-timer

Use the **hello-timer** command to set the global LLDP transmit interval.

Syntax

```
hello-timer transmit-interval  
no hello-timer
```

Command Default

The global LLDP transmit interval is set to 30 seconds.

Parameters

transmit-interval
Interval (in seconds). Use an integer from 1 to 65535.

Modes

LLDP configuration (config-lldp)

Usage Guidelines

Use this command to set how often the device sends LLDP messages. This value can be adjusted to fit your network requirements. A shorter interval helps detect link changes faster, and a longer interval reduces network traffic.

Use the no form of this command to restore the global LLDP transmit interval to the default value.

Examples

The following example sets the global LLDP transmit interval to 100 seconds:

```
device# configure terminal  
device(config)# lldp  
device(config-lldp)# hello-timer 100  
device(config-lldp)#
```

The following example displays the LLDP configuration that is running currently on the device. In this example, the global LLDP transmit interval is set to 100 seconds:

```
device# show running-config lldp  
  
lldp  
  hello-timer 100  
device#
```

helper-only (BGP graceful restart configuration)

Use the **helper-only** command to enable BGP neighbors in the BGP routing process within a Virtual Routing and Forwarding (VRF) instance to participate in BGP graceful restart helper-only mode.

Syntax

```
helper-only  
no helper-only
```

Command Default

Graceful restart helper-only mode is disabled on the BGP neighbors in the BGP routing process.

Parameters

This command has no keywords or variables.

Modes

VRF BGP graceful restart configuration (config-vrf-bgp-gr)

Usage Guidelines

Graceful restart allows a device undergoing a restart to inform its adjacent neighbors of its condition. Helper-only mode assists a neighbor attempting a graceful restart to ensure that no route and topology changes occur in the network.

To disable graceful start helper-only mode, use the **no** form of this command.

Graceful restart mode (that is, a restart time combined with a stale route time) and helper-only mode are mutually exclusive. If graceful restart is enabled, you must use the **no graceful-restart** command to disable it (see [graceful-restart \(BGP configuration\)](#) on page 626) before enabling helper-only mode. Conversely, if helper-only mode is enabled, you must disable it before enabling graceful restart.

Examples

The following example configures a routing process within a VRF instance named violet. This example enables graceful restart helper-only mode on its neighbors:

```
device# configure terminal  
device(config)# vrf violet  
device(config-vrf-violet)# router bgp  
device(config-vrf-bgp)# graceful-restart
```

```
device(config-vrf-bgp-gr) # helper-only
device(config-vrf-bgp-gr) #
```

The following example displays the configuration of a VRF instance named violet that is running currently on the device. In this example, the routing process is configured to enable graceful restart helper-only mode on its neighbors:

```
device# show running-config vrf violet

vrf violet
  router bgp
    graceful-restart
      helper-only
  !
device#
```

hold-time (BGP peer group timer configuration)

Use the **hold-time** command to set the BGP hold time to a nondefault value for all members of a BGP peer group within a Virtual Routing and Forwarding (VRF) instance.

Syntax

hold-time *seconds*

no hold-time *seconds*

Command Default

The BGP hold time is set to 180 seconds.

Parameters

seconds

Duration of hold time. Use an integer of 0 or 3 to 65535 seconds. This value must exceed the keepalive interval (which is 60 seconds by default or the value set by the **keepalive-interval** command). For example, if the keepalive interval is set to 3 seconds, you can set the hold time from 9 to 65535 seconds.

Modes

VRF BGP peer group timer configuration (config-vrf-bgp-pg-timers)

Usage Guidelines

Use this command to set the hold time to apply to all members of a specified BGP peer group. The hold time is the amount of time that a local neighbor waits to receive either a BGP keepalive message or a BGP update message from a remote neighbor before it declares it to be unavailable and brings down the BGP session. This command is related to the **keepalive-interval** command, which sets how often a local neighbor sends a BGP keepalive message (also called a BGP heartbeat packet) to its remote neighbor to verify its availability. For established sessions, the configured hold time takes effect after session flap.

Use the **no** form of this command to reset the hold time to the default value for all members of the BGP peer group. For established sessions, the default value takes effect after session flap.

Examples

The following example configures a BGP peer group named `group1` under a VRF instance named `violet`. This example sets the hold time to 10 seconds for the members of `group1`:

```
device# configure terminal
device(config)# vrf violet
device(config-vrf-violet)# router bgp
device(config-vrf-bgp)# peer-group group1
device(config-vrf-bgp-pg)# timers
device(config-vrf-bgp-pg-timers)# hold-time 10
device(config-vrf-bgp-pg-timers)#
```

The following example displays the configuration of a VRF instance named `violet` that is running currently on the device. In this example, the `group1` peer group sets a hold time of 10 seconds for each member:

```
device# show running-config vrf violet

vrf violet
!
  router bgp
    peer-group group1
      timers
        hold-time 10
!
device#
```

holdtime (LLDP configuration)

Use the **holdtime** command to set the global LLDP hold time.

Syntax

```
holdtime hold-time  
no holdtime
```

Command Default

The global LLDP hold time is set to 120 seconds.

Parameters

hold-time

Duration of hold time (in seconds). Use an integer from 10 to 65535.

Modes

LLDP configuration (config-lldp)

Usage Guidelines

Use this command to set the amount of time that a receiving device should hold the LLDP information sent by your device before discarding it.

Use the no form of this command to restore the global LLDP hold time to the default value.

Examples

The following example sets the global LLDP hold time to 100 seconds:

```
device# configure terminal  
device(config)# lldp  
device(config-lldp)# holdtime 100  
device(config-lldp)#
```

The following example displays the LLDP configuration that is running currently on the device. In this example, the global LLDP hold time is set to 100 seconds:

```
device# show running-config lldp  
  
lldp  
  holdtime 100  
device#
```

hop-limit

Use the **hop-limit** command to configure the allowed maximum number of hops for the IPv6 router advertisement on an interface.

Syntax

```
hop-limit number-of-hops  
no hop-limit
```

Command Default

The hop limit is 64 hops.

Parameters

number-of-hops

Maximum number of hops. Use an integer from 0 to 255.

Modes

IPv6 router advertisement configuration (config-if-*name*-ipv6-router-advertisement)

Usage Guidelines

Router advertisements are ICMPv6 messages sent by an IPv6 enabled router that are used by hosts to detect routers on an attached link. This functionality is similar to the IPv4 ICMP Router Discovery protocol.

Examples

The following example sets a hop limit of 133 hops for IPv6 router advertisement on Ethernet interface 0/1:

```
device# configure terminal  
device(config)# interface ethernet 0/1  
device(config-if-eth-0/1)# ipv6-router-advertisement  
device(config-if-eth-0/1-ipv6-router-advertisement)# hop-limit 133  
device(config-if-eth-0/1-ipv6-router-advertisement)#
```

The following example displays the configuration of Ethernet interface 0/1 that is running currently on the device. In this example, a hop limit of 133 hops for IPv6 router advertisement is configured on the interface:

```
device# show running-config interface ethernet 0/1  
  
!  
interface ethernet 0/1  
  ipv6 address 1111::10/64  
  ipv6 neighbor 1111::1 00:04:96:eb:c4:51  
  ipv6-router-advertisement
```

```
hop-limit 133
interval 100 min 50
enable
prefix 100::1/64
prefix 100::1/63
    valid-lifetime 200
!
no shutdown
!
device#
```


hostname (IAH VM configuration)

Use the **hostname** command to set a hostname for an IAH third party virtual machine (TPVM).

Syntax

hostname *tpvm-hostname*

no hostname

Command Default

The current TPVM uses "TPVM" as the hostname. This is the default Linux behavior.

Parameters

tpvm-hostname

Hostname. Use an arbitrary sequence of alphabetic or numeric characters, underscores, hyphens, or dots. The name cannot exceed 64 characters.

Modes

IAH VM configuration (config-iah-vm-*name*)

Usage Guidelines

This command sets the guest OS hostname of the TPVM. The TPVM is a general server that resides on Extreme ONE OS devices. The guest OS that the TPVM provides is separate from Extreme ONE OS.

This command is allowed conditionally only for an integrated application hosting (IAH) VM named TPVM. Specifying this VM name extends the available IAH configurations to support TPVM.

Use the **no** form of this command to restore the hostname to the default setting. Note that configuration is not allowed when the VM is in the deployed state.

To display details about a specific VM, see [show iah vm](#) on page 1006.

Examples

The following example renames the guest OS hostname from the default name (localhost) to ganga for the IAH TPVM on the device:

```
device# configure terminal
device(config)# iah
device(config-iah)# vm TPVM
device(config-iah-vm-TPVM)# hostname ganga
device(config-iah-vm-TPVM)#
```

The following example displays the IAH configuration that is running currently on the device. In this example, the guest OS hostname is renamed from the default name (localhost) to ganga for the IAH TPVM on the device:

```
device# show running-config iah

iah
vm TPVM
  hostname ganga
  timezone Etc/GMT
  ip
    ipv4-addr 2.2.2.4
    ipv4-gw 2.2.2.1
  ntp
    server 1.1.1.1
    server 3.3.3.3
    server time.example.com
    server 2001:db8::1
  interface management ipv4 2.2.2.2/24 gw 2.2.2.1
  trusted-peer ipv4 3.3.3.3 password $6$bcdef sudo-user extreme
  !
!
device#
```

iah fileinfo

Use the **iah fileinfo** command to view the status information about an image file being used to upgrade the firmware on an IAH third party virtual machine (TPVM).

Syntax

```
iah fileinfo image-filename
```

Parameters

image-filename

IAH supported TPVM image file name. This file must reside in disk://iah or in usb://. The name must be in Debian format. For example, tpvm-4.7.6-0.amd64.deb.

Modes

EXEC (#)

Usage Guidelines

This command returns status information about an IAH image file for the TPVM on the device. For example, "Download in progress." If the file is present, this command performs a pass/fail sanity check and displays the file name, size, and version.

The TPVM is a general server that resides on Extreme ONE OS devices. The guest OS that the TPVM provides is separate from Extreme ONE OS.

Examples

The following example checks the status of a firmware image named tpvm-4.7.6-0.amd64.deb for the IAH TPVM on the device:

```
device# iah fileinfo disk://iah/tpvm-4.7.6-0.amd64.deb

File: tpvm-4.7.6-0.amd64.deb
Format: TPVM debian package
DiskSize: 2790069542 bytes
Version: 4.7.6
Success.
device#
```

The following example checks the status of a firmware image named tpvm-4.7.6-0.amd64.deb for the IAH TPVM on the device. In this example, the specified file could not be found:

```
device# iah fileinfo disk://iah/tpvm-4.7.6-0.amd64.deb

%Error: File disk://iah/tpvm-4.7.6-0.amd64.deb does not exist
device#
```

The following example checks the status of a firmware image named `tpvm-4.7.6-0.amd64.deb` for the IAH TPVM on the device. In this example, the image file failed the sanity check:

```
device# iah fileinfo disk://iah/tpvm-4.7.6-0.amd64.deb

File: tpvm-4.7.6-0.amd64.deb
Format: TPVM debian package
DiskSize: 11000 bytes
Version:
Sanity check mismatch.
device#
```

iah (global configuration)

Use the **iah** command to enter IAH configuration (config-iah) mode.

Syntax

iah

Parameters

This command has no keywords or variables.

Modes

Global configuration (config)

Usage Guidelines

If Extreme ONE OS reboots upon upgrade or downgrade, virtual machine (VM) startup depends on the auto-boot configuration.

To display details about a specific VM, see [show iah vm](#) on page 1006.

Examples

The following example enters IAH configuration (config-iah) mode and configures a VM named Ubuntu1 on the device:

```
device# configure terminal
device(config)# iah
device(config-iah)# vm Ubuntu1
device(config-iah-vm-Ubuntu1)#
```

The following example displays the IAH configuration that is running currently on the device. In this example, IAH is configured with a VM named Ubuntu1:

```
device# show running-config iah

iah
  vm Ubuntu1
    cpus 1
    ram 1024
    disk sda
      size 5000
      format qcow2
      bus sata
      file disk://iah/disk1.qcow2
    !
  !
!
device#
```

iah vm

Use the **iah vm** command to power off, shut down, reboot, start, suspend, or resume the operation of a virtual machine (VM) or create, delete, or revert a disk snapshot of the TPVM.

Syntax

```
iah vm { poweroff | shutdown | | reboot | start | suspend | resume }  
      vm-name
```

```
iah vm snapshot { create | delete | revert } vm-name
```

Command Default

The specified IAH VM on the device is not started.

Parameters

poweroff

Turns the power off to the VM.

shutdown

Shuts down the VM.

reboot

Reboots the VM.

start

Starts the VM.

suspend

Suspends operation of the VM.

resume

Resumes operation of the VM.

vm-name

Name of the VM. The name must be an arbitrary sequence of 1 to 32 alphanumeric characters, underscores, hyphens, or dots.

```
snapshot { create | delete | revert } vm-name
```

Creates, deletes, or reverts a disk snapshot of the third party virtual machine (TPVM). Use TPVM as the VM name.

Modes

EXEC (#)

Usage Guidelines

To display details about a specific VM, see [show iah vm](#) on page 1006.

Examples

The following examples power off, shut down, reboot, start, suspend, and resume the operation of a VM named UbuntuVm1 respectively on the device:

```
device# iah vm poweroff UbuntuVm1
device#

device# iah vm shutdown UbuntuVm1
device#

device# iah vm reboot UbuntuVm1
device#

device# iah vm start UbuntuVm1
device#

device# iah vm suspend UbuntuVm1
device#

device# iah vm resume UbuntuVm1
device#
```

The following example deletes a disk snapshot of a VM named TPVM on the device:

```
device# iah vm snapshot delete TPVM
device#
```

iah vm console

Use the **iah vm console** command to connect to the console of a virtual machine (VM).

Syntax

```
iah vm console vm-name
```

Command Default

The console of the specified VM on the device is disconnected.

Parameters

vm-name

Name of the VM. The name must be an arbitrary sequence of 1 to 32 alphanumeric characters, underscores, hyphens, or dots.

Modes

EXEC (#)

Usage Guidelines

You cannot connect to the console when the VM is not running.

Press **Ctrl+** to exit the VM console.

To display details about a specific VM, see [show iah vm](#) on page 1006.

Examples

The following example connects to the console of a VM named UbuntuVm1 on the device:

```
device# iah vm console UbuntuVm1
Connected to domain 'ubuntu'
Escape character is ^x (Ctrl + \)
```


import-policy (BGP peer group address family configuration)

Use the **import-policy** command to configure the inbound policy for an address family in a specific peer group within a routing process in a VRF instance.

Syntax

```
import-policy policy-name  
no import-policy
```

Command Default

The address family in the specified peer group has no import policy configured.

Parameters

policy-name

Name of the import policy to use with the address family.

Modes

VRF BGP peer group address family configuration (config-vrf-bgp-pg-*address-family-type*)

Usage Guidelines

Use this command to apply a set of rules that define how routes received from group members are filtered or modified before being accepted into the BGP table. This policy applies specifically to a given address family and helps control routing behavior and enforce policy at scale across numerous peers.

This command applies only to IPv4 unicast or IPv6 unicast address families.

Use the **no** form of this command to restore the default functionality.

Examples

The following example configures a routing process under a VRF instance named violet. This example specifies an IPv4 unicast address family in a peer group named group1 and also specifies an import policy named importpolicy2 for the address family:

```
device# configure terminal  
device(config)# vrf violet  
device(config-vrf-violet)# router bgp  
device(config-vrf-bgp)# peer-group group1  
device(config-vrf-bgp-pg)# address-family ipv4 unicast  
device(config-vrf-bgp-pg-ipv4u)# import-policy importpolicy2  
device#
```

The following example displays the configuration of a VRF instance named violet that is running currently on the device. In this example, a peer group named group1 is configured. An IPv4 unicast address family that uses an import policy named importpolicy2 is specified in this example:

```
device# show running-config vrf violet

vrf violet
  router bgp
    address-family ipv4 unicast
      send-default-route
    !
    peer-group group1
      remote-as 200
      address-family ipv4 unicast
        import-policy myimportpolicy2
      !
    !
  !
device#
```

iah vm import

Use the **iah vm import** command to import a file containing an image of a virtual machine (VM).

Syntax

```
iah vm import vm-name { ova | tpvm } file-url
```

Command Default

The device has no VMs configured.

Parameters

vm-name

Name of the VM to create. The name must be an arbitrary sequence of 1 to 32 alphanumeric characters, underscores, hyphens, or dots. Do not use the **all** reserved keyword as the complete name.

{ **ova** | **tpvm** } *file-url*

URL that points to the Open Virtual Appliance (OVA) file or the third party virtual machine (TPVM) file. Use one of the following values:

- **usb:**//*filename*
- **disk:**//**iah**/*filename*

Import is mandatory for creating a TPVM.

Modes

EXEC (#)

Usage Guidelines

Use this command to create a VM by importing an image contained in a package file.

This command performs a checksum calculation on the specified package file during the import. An error in checksum validation aborts the import.

A package file contains VM configuration metadata:

- If IAH cannot retrieve the number of vCPUs from the file, it configures the VM with four vCPUs.
- If IAH cannot retrieve the number of gigabytes of RAM from the file, it configures the VM with 8 gigabytes.
- If IAH cannot retrieve the disk image size from the file, it leaves the size unset.

You can deploy the VM after the configuration is done.

You can change the configuration after VM deployment. To do so, shut down the VM (if it is running or paused), then undeploy the VM, then use the appropriate **iah vm** command to modify the configuration, and then redeploy.

To display details about a specific VM or all VMs, see [show iah vm](#) on page 1006.

Examples

The following example imports an OVA package file from `disk://iah/ubuntu.ova` and creates a VM named `UbuntuVm1` from the file:

```
device# iah vm import UbuntuVm1 ova disk://iah/ubuntu.ova
VM import is successful, Please go to configure mode and deploy VM
device#
```

The following example imports a TPVM package file from `disk://iah/tpvm-4.7.5-2.amd64.deb` and creates the TPVM VM from the file:

```
device# iah vm import TPVM tpvm disk://iah/tpvm-4.7.5-2.amd64.deb
VM import is successful, Please go to configure mode and deploy VM
device#
```

The following example displays details about the IAH VMs on the device:

```
device# show iah vm all

Number of VMs: 1
Name: TPVM
  deploy-status: DEPLOYED_SUCCESS
  vm-status: RUNNING
  last-error: Process exited with status 1
  description: TPVM
  auto-boot: enabled
  activate-network: enabled
  cpus: 4
  ram: 8192 MiB
  disk: sda
    size: 10240 MB
    file: /mnt/onl/iah/images/TPVM/sda.qcow2
    file format: qcow2
    bus type: virtio
  disk: sdb
    size: 51200 MB
    file: /mnt/onl/iah/images/TPVM/sdb.qcow2
    file format: qcow2
    bus type: virtio
  disk: hda
    size: 1 MB
    file: /mnt/onl/iah/images/TPVM/hda.raw
    file format: raw
    bus type: ide
  network: mgmt0
    interface: management 0
    connector-type: MANAGEMENT
    mac: -
  TPVM:
    hostname: MYTPVMHOST
    user: extreme
    timezone: Etc/UTC
    version: 4.7.5
    ssh-passwordless: enabled
```

```
ipaddress: 1.1.1.1/24
gateway: 2.2.2.2
ipv6address: 2001:db8::/64
gatewayv6: 2001:db8:1234:5678
ntp:
  server: 3.3.3.3
device#
```

The following example displays the IAH configuration that is running currently on the device:

```
device# show running-config iah

iah
  vm TPVM
    description TPVM
    auto-boot
    cpus 4
    ram 8192
    disk sda
      size 10240
      file /mnt/onl/iah/images/TPVM/sda.qcow2
      format QCOW2
    !
    disk sdb
      size 51200
      file /mnt/onl/iah/images/TPVM/sdb.qcow2
      format QCOW2
    !
    disk hda
      size 1
      file /mnt/onl/iah/images/TPVM/hda.raw
      format RAW
    !
    network mgmt0
      interface management 0
    !
    activate-network
    deploy
    hostname MYTPVMHOST
    interface management ipv4 1.1.1.1/24 gw 2.2.2.2
    interface management ipv6 2001:db8::/64 gw 2001:db8:1234:5678
    trusted-peer ipv4 3.3.3.3 password-hashed yEfYBIXsq-U43HziayLs1WMzk_SJjBYVKg== sudo-
user extreme
  ntp 3.3.3.3
  !
!
device#
```

iah vm upgrade

Use the **iah vm upgrade** command to upgrade the firmware of an IAH third party virtual machine (TPVM).

Syntax

```
iah vm upgrade vm-name file image-filename [ incremental | snapshot ]
```

Parameters

vm-name

Name of the VM to upgrade. TPVM is the only supported VM name in this release.

image-filename

IAH supported VM page file name. This file must reside in disk://iah or in usb://. The name must be in Debian format. For example, tpvm_inc_upg-4.7.6-0.amd64.deb.

incremental

(Optional) Performs an incremental upgrade of the VM main disk image.

snapshot

(Optional) Creates a root filesystem (rootfs) snapshot/backup of the existing VM before upgrading.

Modes

EXEC (#)

Usage Guidelines

To display details about a specific VM, see [show iah vm](#) on page 1006.

Examples

The following example performs a full upgrade of the firmware for the IAH TPVM using an image named disk://iah/tpvm-4.7.6-0.amd64.deb on the device:

```
device iah vm upgrade TPVM file disk://iah/tpvm-4.7.6-0.amd64.deb
VM upgrade is successful
device#
```

The following example performs an incremental upgrade of the firmware for the IAH TPVM using an image named tpvm_inc_upg-4.7.6-0.amd64.deb on the device:

```
device# iah vm upgrade TPVM file disk://iah/tpvm_inc_upg-4.7.6-0.amd64.deb incremental
VM upgrade is successful
device#
```

The following example displays information about the TPVM on the device:

```
device# show iah vm TPVM

Name: TPVM
  deploy-status: PERSONALITY_CONFIG_SUCCESS
  vm-status: RUNNING
  auto-boot: enabled
  activate-network: enabled
  TPVM:
    version: 4.7.9
    ssh-passwordless: enabled
device#
```

The following example displays details about the TPVM on the device:

```
device# show iah vm TPVM details

show iah vm TPVM details
Name: TPVM
  deploy-status: PERSONALITY_CONFIG_SUCCESS
  vm-status: RUNNING
  last-error: -
  description: TPVM
  auto-boot: enabled
  activate-network: enabled
  cpus: 2
  ram: 4096 MiB
  disk: sda
    size: 10240 MB
    file: /mnt/onl/iah/images/TPVM/sda.qcow2
    file format: qcow2
    bus type: virtio
  disk: sdb
    size: 40960 MB
    file: /mnt/onl/iah/images/TPVM/sdb.qcow2
    file format: qcow2
    bus type: virtio
  disk: hda
    size: 1 MB
    file: /mnt/onl/iah/images/TPVM/hda.raw
    file format: raw
    bus type: ide
  network: mgmt0
    interface: management 0
    connector-type: MANAGEMENT
    mac: 40:88:2f:fa:78:05
  TPVM:
    hostname: TPVM-TD3-2
    user: extreme
    timezone: Asia/Kolkata
    version: 4.7.9
    ssh-passwordless: enabled
    ipaddress: 10.38.65.43/24
    gateway: 10.38.65.1
    ipv6address: -
    gatewayv6: -
    ntp:
      server: 10.64.12.81
device#
```

input (interface QoS configuration)

Use the **input** command to enter the mode for configuring access QoS in the input direction on an interface.

Syntax

input
no input

Command Default

Access QoS is not configured in the input direction on the current interface.

Parameters

This command has no keywords or variables.

Modes

Ethernet interface QoS configuration (config-qos-if-eth-*name*)

Port channel interface QoS configuration (config-qos-if-po-*name*)

Virtual Ethernet interface QoS configuration (config-qos-if-ve-*name*)

Usage Guidelines

This command enters the mode for accessing the commands for configuring access QoS in the input direction on an interface. You use these commands to configure traffic class maps and attach them to interfaces in the incoming direction and also enable trust CoS or trust DSCP in the incoming direction.

To remove access QoS configuration in the input direction on the interface, use the **no** form of this command.

Examples

The following example configures access QoS on interface Ethernet 0/1 and its subinterface 100 in the input direction:

```
device# configure terminal
device(config)# qos
device(config-qos)# interface ethernet 0/1 sub-interface 100
device(config-qos-if-eth-0/1:1.100# input
device(config-qos-if-eth-0/1:1.100-input)# trust dscp
device(config-qos-if-eth-0/1:1.100-input)# traffic-class-map ing_l2_map
device(config-qos-if-eth-0/1:1.100-input)#
```


instance-type

Use the **instance-type** command to configure the instance type of a Virtual Routing and Forwarding (VRF) instance and enter VRF instance type configuration mode.

Syntax

instance-type evpn

no instance-type evpn

Command Default

No instance type is configured for the specified VRF instance.

Parameters

evpn

Configures the instance as an Ethernet VPN (EVPN).

Modes

VRF BGP configuration (config-vrf-bgp)

Usage Guidelines



Note

This command is supported only for user-configured VRFs but not for default-vrf (the default VRF).

Use the **no** form of this command to deconfigure the specified instance type from the specified VRF instance.

Examples

The following example configures a VRF instance named `tenant_1` on the device and specifies that it is an EVPN instance:

```
device# configure terminal
device(config)# vrf tenant_1
device(config-vrf-tenant_1)# router bgp
device(config-vrf-bgp)# instance-type evpn
```

The following example shows the configuration that is running currently on the device. In this example, a single VRF instance (in this case, `tenant_1`) exists on the device, and it is configured as an EVPN instance:

```
device# show running-config
...
vrf tenant_1
```

```
router bgp
  instance-type evpn
...
device#
```

interface ethernet

Use the **interface ethernet** command to enter Ethernet interface configuration (config-if-eth-*name*) mode.

Syntax

```
interface ethernet interface-name
```

Command Default

No Ethernet interfaces are configured on the device.

Parameters

interface-name

Interface name. Use the *slot/port* format or the *slot/port:breakout* format to specify an interface or one or more comma-separated ranges of interfaces respectively. For example, 0/1 or 1/1-3,5,2/7-9,10:1-4 respectively.

Modes

Global configuration (config)

Usage Guidelines

This command enters the mode required for entering the settings on an Ethernet interface. They include the shutdown mode, duplex mode, Forward Error Correction (FEC), Link Fault Signaling (LFS), MTU, port speed, loopback mode, and description.

If you want to view the status of the Ethernet interfaces on a device, see [show interface ethernet link-error-disable](#) on page 1012.

Examples

The following example enters Ethernet interface configuration (config-if-eth-*name*) mode and configures Ethernet interface 0/1:

```
device# configure terminal
device(config)# interface ethernet 0/1
device(config-if-eth 0/10-14)# no shutdown
device(config-if-eth 0/10-14)# description Ethernet interface 0/1
device(config-if-eth 0/10-14)# fec rs-fec
device(config-if-eth 0/10-14)# link-fault-signaling
device(config-if-eth 0/10-14)# mtu 4000
device(config-if-eth 0/10-14)# speed 10
device(config-if-eth 0/10-14)# loopback-mode
device#
```

interface (IAH VM network configuration)

Use the **interface** command to configure the network interface for a virtual machine (VM).

Syntax

```
interface { management 0 [ mac mac-address ] | internal interface-name }  
no interface { management 0 [ mac mac-address ] | internal interface-  
                  name }
```

Command Default

The network on a VM is not configured with an interface.

Parameters

management 0

Specifies the management 0 interface for networking in bridged mode. (0 is the only interface name supported in this release.) The VM will be in the same network as the VM host.

internal *interface-name*

Specifies an internal interface for VM networking in dedicated Insight interface (Intel VT-d) mode. Use either 0 or 1 as the interface name.

mac-address

MAC address to assign to the interface. Use *xx:xx:xx:xx:xx:xx* notation (six groups of two hexadecimal digits separated by colons).

Modes

IAH VM network configuration (*config-iah-vm-network-name*)

Usage Guidelines

Use the **no** form of this command to remove the interface configuration from a VM. Note that configuration is not allowed when the VM is in the deployed state.

To display details about a specific VM, see [show iah vm](#) on page 1006.

Examples

The following example configures the management interface on a network named vmmgmt for a VM named Ubuntu1:

```
device# configure terminal  
device(config)# iah  
device(config-iah)# vm Ubuntu1
```

```
device(config-iah-vm-Ubuntu1)# network vmmgmt
device(config-iah-vm-network-vmmgmt)# interface management 0 mac 11:22:33:44:55:00
device(config-iah-vm-network-vmmgmt)#
```

The following example displays the IAH configuration that is running currently on the device. In this example, a VM named Ubuntu2 uses an interface named management 0 in a network named vmmgmt:

```
device# show running-config iah

iah
vm Ubuntu2
  cpus 1
  ram 1024
  disk sda
    size 1000
    format qcow2
    bus sata
    file disk://iah/disk1.qcow2
  !
  network vmmgmt
    interface management 0
    mac 11:22:33:44:55:00
  !
!
device#
```

interface management ipv4 (IAH VM configuration)

Use the **interface management** command to configure the management interface IPv4 address and IPv4 gateway address for an IAH third party virtual machine (TPVM).

Syntax

```
interface management ipv4 { dhcp | ip-address / prefix-length [ gw  
    gateway-address ] }  
  
no interface management ipv4
```

Command Default

The management interface for the TPVM has no IPv4 address.

Parameters

ip-address / prefix-length

Range of IPv4 addresses. Use the *a.b.c.d* format for *ip-address*, and use an integer from 0 to 32 for the prefix. For example, 10.1.1.0/24. You must include the slash (/).

gw *gateway-address*

(Optional) Gateway IPv4 address. Use the *a.b.c.d* format. For example, 10.1.1.1.

Modes

IAH VM configuration (config-iah-vm-*name*)

Usage Guidelines

This command configures a DHCP IPv4 address or a static IPv4 address on the management interface of the TPVM on a device. The TPVM is a general server that resides on Extreme ONE OS devices. The guest OS that the TPVM provides is separate from Extreme ONE OS.

The TPVM management interface differs from the base Extreme ONE OS management 0 interface (although both use the same physical path).

The TPVM management interface can have one or both of IPv4 and IPv6 DHCP or static configurations. For both, you must configure each one separately. To configure a DHCP IPv6 address or a static IPv6 IP address, see [interface management ipv6 \(IAH VM configuration\)](#) on page 664.

Use the **no** form of this command to remove an IPv4 address from the TPVM. Note that configuration is not allowed when the VM is in the deployed state.

When removing an IPv6 address via the **no** form of this command from within gNMI, the system also deletes the applicable gateway if any.

If you must update an IPv4 address, you must first delete the present configuration using the appropriate **no** command before proceeding with the update.

The internal configuration database (CDB) might show a configured gateway address, but the internal state database (SDB) will show no data for the gateway too. In other words, the **no** form of this command for a gateway address is optional and redundant, especially when used via gNMI.

This command is allowed conditionally only for an integrated application hosting (IAH) VM named TPVM. Specifying this VM name extends the available IAH configurations to support TPVM.

To display details about a specific VM, see [show iah vm](#) on page 1006.

Examples

The following example sets the IPv4 address and the IPv4 gateway address to 2.2.2.2/24 and 2.2.2.1 respectively on the management interface for the IAH TPVM on the device:

```
device# configure terminal
device(config)# iah
device(config-iah)# vm TPVM
device(config-iah-vm-TPVM)# interface management ipv4 2.2.2.2/24 gw 2.2.2.1
device(config-iah-vm-TPVM)#
```

The following example displays the IAH configuration that is running currently on the device. In this example, the IPv4 address and the IPv4 gateway address are set to 2.2.2.2/24 and 2.2.2.1 respectively on the management interface for the IAH TPVM on the device:

```
device# show running-config iah

iah
vm TPVM
  hostname ganga
  timezone Africa/Casablanca
  ip
    ipv4-addr 2.2.2.4
    ipv4-gw 2.2.2.1
  ntp
    server 1.1.1.1
    server 3.3.3.3
    server time.example.com
    server 2001:db8::1
  interface management ipv4 2.2.2.2/24 gw 2.2.2.1
  trusted-peer ipv4 3.3.3.3 password $6$bcdef sudo-user extreme
  !
!
device#
```

interface management ipv6 (IAH VM configuration)

Use the **interface management** command to configure the management interface IPv6 address and IPv6 gateway address for an IAH third party virtual machine (TPVM).

Syntax

```
interface management ipv6 { dhcp | ip-address / prefix-length [ gw  
    gateway-address ] }  
  
no interface management ipv6
```

Command Default

The management interface for the TPVM has no IPv6 address.

Parameters

ip-address/prefix-length

Range of IPv6 addresses. Use the *x:x::x:x* format for *ip-address*, and use an integer from 0 to 128 for the prefix. For example, 2001::20/64. You must include the slash (/).

gw *gateway-address*

(Optional) Gateway IPv6 address. Use the *x:x::x:x* format. For example, 2001:::2.

Modes

IAH VM configuration (config-iah-vm-*name*)

Usage Guidelines

This command configures a DHCP IPv6 address or a static IPv6 address on the management interface of the TPVM on a device. The TPVM is a general server that resides on Extreme ONE OS devices. The guest OS that the TPVM provides is separate from Extreme ONE OS.

The TPVM management interface differs from the base Extreme ONE OS management 0 interface (although both use the same physical path).

The TPVM management interface can have one or both of IPv4 and IPv6 DHCP or static configurations. For both, you must configure each one separately. To configure a DHCP IPv4 address or a static IPv4 IP address, see [interface management ipv4 \(IAH VM configuration\)](#) on page 662.

Use the **no** form of this command to remove an IPv6 address from the TPVM. Note that configuration is not allowed when the VM is in the deployed state.

When removing an IPv6 address via the **no** form of this command from within gNMI, the system also deletes the applicable gateway if any.

If you must update an IPv6 address, you must first delete the present configuration using the appropriate **no** command before proceeding with the update.

The internal configuration database (CDB) might show a configured gateway address, but the internal state database (SDB) will show no data for the gateway too. In other words, the **no** form of this command for a gateway address is optional and redundant, especially when used via gNMI.

This command is allowed conditionally only for an integrated application hosting (IAH) VM named TPVM. Specifying this VM name extends the available IAH configurations to support TPVM.

To display details about a specific VM, see [show iah vm](#) on page 1006.

Examples

The following example sets the IPv6 address and the IPv6 gateway address to 2001::20/64 and 2001:::2 respectively on the management interface for the IAH TPVM on the device:

```
device# configure terminal
device(config)# iah
device(config-iah)# vm TPVM
device(config-iah-vm-TPVM)# interface management ipv6 2001::20/64 gw 2001:::2
device(config-iah-vm-TPVM)#
```

The following example displays the IAH configuration that is running currently on the device. In this example, the IPv6 address and the IPv6 gateway address are set to 2001::20/64 and 2001:::2 respectively on the management interface for the IAH TPVM on the device:

```
device# show running-config iah

iah
  vm TPVM
    hostname ganga
    timezone Africa/Casablanca
    ip
      ipv4-addr 2.2.2.4
      ipv4-gw 2.2.2.1
    ntp
      server 1.1.1.1
      server 3.3.3.3
      server time.example.com
      server 2001:db8::1
    interface management ipv6 2001::20/64 gw 2001:::2
    trusted-peer ipv4 3.3.3.3 password $6$bcdef sudo-user extreme
    !
  !
device#
```

interface (MLAG configuration)

Use the **interface (MLAG configuration)** command to configure a multichassis link aggregation (MLAG) interface.

Syntax

```
interface { ethernet interface-name | port-channel port-channel-number }  
    id { identifier-num | auto }  
  
no interface { ethernet interface-name | port-channel port-channel-  
    number }
```

Command Default

No MLAG interface is configured on the device.

Parameters

ethernet *interface-name*

Ethernet interface name. Use the *slot/port* format. For example, 0/1.

port-channel *port-channel-number*

Port channel number. Use an integer from 1 to 255. For example, 1.

id { *identifier-num* | **auto** }

MLAG identifier of this Ethernet interface or this port channel. Use an integer from 0 to 4294967295, or (with the port-channel keyword only) use the **auto** keyword to specify that the MLAG identifier is generated automatically.

Modes

MLAG configuration (config-mlag)

Usage Guidelines

To remove an MLAG interface configuration, use the no form of this command.

Examples

The following example configures an Ethernet interface on port 1 in slot 0 and specifies 234 as the MLAG identifier number:

```
device# configure terminal  
device(config)# mlag  
device(config-mlag)# interface ethernet 0/1 id 234  
device#
```

The following example configures a port channel port with an ID of 3 and specifies 4 as the MLAG identifier number:

```
device# configure terminal
device(config)# mlag
device(config-mlag)# interface port-channel 3 id 4
device#
```

The following example configures a port channel port with an ID of 5 and specifies that the MLAG identifier number is autogenerated:

```
device# configure terminal
device(config)# mlag
device(config-mlag)# interface port-channel 5 id auto
device#
```

The following example displays the MLAG configuration that is running currently on the device. In this example, one Ethernet MLAG interface and two port channel MLAG ports are configured:

```
device# show running-config mlag

mlag
  role primary
  mac 0000.aabb.ccdd
  interface ethernet 0/1 id 234
  interface port-channel 3 id 4
  interface port-channel 5 id auto
device#
```

interface port-channel (LAG configuration)

Use the **interface port-channel** command to configure a link aggregation group (LAG) interface.

Syntax

```
interface port-channel port-channel-range  
no interface port-channel port-channel-range
```

Command Default

No LAG interface is configured on the device.

Parameters

port-channel-range

Port channels. Use any combination of one or more comma separated integers and one or more comma separated ranges of integers from 1 to 255. For example, 1 or 1-3,5,7-9.

Modes

Global configuration (config)

Usage Guidelines

To remove a LAG interface configuration, use the no form of this command.

Examples

The following example configures port channel interface 10:

```
device# configure terminal  
device(config)# interface port-channel 10  
device#
```

The following example configures a port channel interface with port numbers 1-3,5,7-9:

```
device# configure terminal  
device(config)# interface port-channel 1-3,5,7-9  
device#
```

The following example displays the configuration for a specific port channel that is running currently on the device:

```
device# show running-config interface port-channel 10  
  
interface port-channel 10  
    no shutdown  
!  
device#
```

interface (QoS configuration)

Use the **interface** command to specify an interface on which to configure access QoS.

Syntax

```
interface { ethernet interface-name | port-channel port-channel-range }
    subinterface { vlan vlan-id [ inner-vlan vlan-range ] | untagged }

interface ve interface-name

no interface { ethernet interface-name | port-channel port-channel-range }
    subinterface { vlan vlan-id [ inner-vlan vlan-range ] | untagged }

no interface ve interface-name
```

Command Default

No access QoS is configured on the current interface.

Parameters

ethernet *interface-name*

Ethernet interface name. Use the *slot/port* format or the *slot/port:breakout* format to specify an interface or one or more comma-separated ranges of interfaces respectively. For example, 0/1 or 1/1-3,5,2/7-9,10:1-4 respectively.

port-channel *port-channel-range*

Port channels. Use any combination of one or more comma-separated integers and one or more comma-separated ranges of integers from 1 to 255. For example, 1 or 1-3,5,7-9.

subinterface { **vlan** *vlan-id* [**inner-vlan** *vlan-range*] | **untagged** }

Ethernet or port channel subinterface. You can specify a tagged VLAN subinterface (with an optional inner VLAN) or an untagged subinterface:

- For *vlan-id*, specify an integer from 1 to 4095.
- (Optional) For *vlan-range*, specify an integer from 1 to 4095.
- Use the **untagged** keyword to specify an untagged subinterface.

ve *interface-name*

Virtual Ethernet (VE) interface name. Use an integer or one or more comma-separated ranges of integers from 1 to 8192. For example, 2 or 2-3,5,7-9.

Modes

QoS configuration (config-qos)

Usage Guidelines

You can apply a QoS configuration to Ethernet or port channel subinterfaces.

To remove an access QoS configuration from the interface, use the **no** form of this command.

Examples

The following example configures access QoS on interface Ethernet 0/1 and its subinterface 100:

```
device# configure terminal
device(config)# qos
device(config-qos)# interface ethernet 0/1 sub-interface 100
device(config-qos-if-eth-0/1:1.100# input
device(config-qos-if-eth-0/1:1.100-input)# traffic-class-map ing_l2_map
device(config-qos-if-eth-0/1:1.100-input)#
```

The following example displays the access QoS configuration that is running currently on the device:

```
device# show running-config qos

qos
  interface ethernet 0/1 sub-interface 100
    input
      traffic-class-map ing_l2_map
device#
```

interface tunnel (Extreme ONE OS Switching)

Use the **interface tunnel** command to enter Tunnel interface configuration (config-tunnel-*name*) mode.

Syntax

```
interface tunnel interface-name  
no interface tunnel interface-name
```

Command Default

No tunnel interfaces are configured on the device.

Parameters

interface-name

Tunnel interface name. Use an arbitrary sequence of alphabetic or numeric characters, underscores, hyphens, or dots. The name cannot exceed 64 characters.

Modes

Global configuration (config)

Usage Guidelines

This command enters the mode required for entering the settings for a tunnel interface. The settings include the tunnel source Network Virtualization Endpoint (NVE), the tunnel destination, the shutdown mode, and the type of tunnel protocol (only the VXLAN protocol is allowed in this release).

If you want to view the status of the tunnel interfaces on a device, see [show interface tunnel \(Extreme ONE OS Switching\)](#) on page 1016.

To delete a tunnel interface from the device, use the **no** form of this command.

Examples

The following example enters Tunnel interface configuration (config-tunnel-*name*) mode and configures tunnel interface 1 as a Virtual eXtensible Local-Area Network (VXLAN) interface:

```
device# configure terminal  
device(config)# interface tunnel 1  
device(config-tunnel-1)# type vxlan  
device(config-tunnel-1)# source nve base  
device(config-tunnel-1)# destination 1.1.1.1
```

```
device(config-tunnel-1)# no shutdown
device(config-tunnel-1)#
```

The following example displays the tunnel interface configuration that is running currently on the device. In this example, tunnel interface 1 is configured as a VXLAN interface:

```
device# show running-config interface tunnel

interface tunnel 1
  type vxlan
    source nve base
    destination 1.1.1.1
  no shutdown
!
```


interval (BFD profile configuration)

Use the **interval (BFD profile configuration)** to configure the desired rate at which to send BFD control packets to the neighboring system.

Syntax

```
interval min-tx-milliseconds min-rx min-rx-milliseconds multiplier
           multiple
no interval
```

Command Default

The minimum transmit interval and the minimum receive interval are 300 milliseconds each. The detection time multiplier is 3.

Parameters

interval *min-tx-milliseconds*

Minimum transmit interval. This is the minimum transmission interval (less jitter) that this system wants to use to send BFD control packets. Use an integer from 50–1700 (on the Extreme 8820 platform) or 50–30000 (on all other supported platforms). The default is 300.

min-rx *min-rx-milliseconds*

Minimum receive interval. This is the minimum interval that this system is capable of receiving BFD control packets.

Use an integer from 50 to 30000. The default is 300.

multiplier *multiple*

Detection time multiplier to determine packet loss. The remote transmission interval is multiplied by this value to determine the connection loss detection timer. For example, when the local system has a multiplier of 3 and the remote system has a minimum transmit interval of 50 ms, the local system will detect failures only after 150 ms without receiving packets.

Use an integer from 3 to 50. The default is 3.

Modes

BFD profile configuration (config-bfd-profile-*name*)

Usage Guidelines

Use this command to configure the desired minimum transmit interval, minimum receive interval for BFD control packets at the local end point, and a multiplier value that helps to calculate the detection timeout of BFD sessions.

To restore the desired minimum transmit interval, the minimum receive interval for BFD control packets at the local end point, and the multiplier to their default values, use the **no** form of this command.

Examples

The following example sets a minimum transmit interval of 5,000 milliseconds, a minimum receive interval of 10,000 milliseconds, and a detection time multiplier of 4:

```
device# configure terminal
device(config)# bfd
device(config-bfd)# profile profile1
device(config-bfd-profile-profile1)# interval 5000 min-rx 10000 multiplier 4
device(config-bfd-profile-profile1)#
```

The following example shows the BFD configuration that is running currently on the device:

```
device# show running-config bfd

bfd
  profile profile1
    interval 400 min-rx 400 multiplier 4
device#
```

interval (IPv6 router advertisement configuration)

Use the **interval** command to configure the maximum and minimum intervals for IPv6 router advertisement on an interface.

Syntax

```
interval seconds [ min seconds ]  
no interval
```

Command Default

The maximum interval is 600 seconds.

The minimum interval is .33 x the maximum interval, if the maximum interval is nine seconds or more (otherwise, the default equals the maximum interval).

Parameters

seconds

Use an integer from 4 to 1800 seconds.

min *seconds*]

Interval between periodic router advertisement neighbor discovery messages sent on this interface. The range is 3 - .75 x the maximum interval (in seconds).

Modes

IPv6 router advertisement configuration (config-if-*name*-ipv6-router-advertisement)

Usage Guidelines

Router advertisements are ICMPv6 messages sent by an IPv6 enabled router that are used by hosts to detect routers on an attached link. This functionality is similar to the IPv4 ICMP Router Discovery protocol.

Neighbor discovery defines five ICMP packet types: router solicitation and advertisement, neighbor solicitation and advertisement, and redirect messages.

Router advertisements carry information such as the router link layer address, prefixes supported in the network, and the MTU setting. Router advertisements are periodic messages or are also sent in response to router solicitations. Because of the nature of its functionality, router advertisement is considered as a separate feature from neighbor discovery. Two messages are supported as part of this functionality: router solicitation and router advertisement.

Each configured interface sends a router advertisement message periodically to the all-nodes link-local multicast address (FF02::1). To avoid synchronization of advertisements

from other routers, the time interval is randomized to send these messages between the configured maximum and minimum time periods.

Router advertisement includes the following information:

- Default router—Whether the sender can be a default router and, if so, the lifetime for the router
- List of prefixes for on-link addresses
- Auto-configuration information
- Network information—Hop limit and MTU setting to use

To return the intervals to their default values on an interface, use the **no** form of this command.

Examples

The following example configures the maximum and minimum intervals for IPv6 router advertisement on Ethernet interface 0/1:

```
device# configure terminal
device(config)# interface ethernet 0/1
device(config-if-eth-0/1)# ipv6-router-advertisement
device(config-if-eth-0/1-ipv6-router-advertisement)# interval 100 min 50
device(config-if-eth-0/1-ipv6-router-advertisement)#
```

The following example displays the configuration of Ethernet interface 0/1 that is running currently on the device. In this example, the maximum and minimum intervals for IPv6 router advertisement are configured to 100 seconds and 50 seconds respectively on the interface:

```
device# show running-config interface ethernet 0/1

!
interface ethernet 0/1
  ipv6 address 1111::10/64
  ipv6 neighbor 1111::1 00:04:96:eb:c4:51
  ipv6-router-advertisement
    hop-limit 133
    lifetime 200
    managed-config-flag
    mode disable-unsolicited-ra
    other-config
    enable
    interval 100 min 50
    prefix 100::1/64
    prefix 100::1/63
    valid-lifetime 200
  !
  no shutdown
!
device#
```

ip mtu (Extreme ONE OS Switching)

Use the **ip mtu** command to set the size of the maximum transmission unit (MTU) for all Layer 3 (network layer) transactions on the device.

Syntax

```
ip mtu bytes  
no ip mtu
```

Command Default

The MTU size is 9194 bytes.

Parameters

bytes
Size in bytes. Use an integer from 1280 to 9194.

Modes

Global configuration (config)

Usage Guidelines

The MTU is the size of the largest protocol data unit (PDU) that can be communicated in a single network layer transaction. The MTU relates to (but is not identical to) the maximum frame size that can be transported on the data link layer (such as an Ethernet frame). This setting applies to all IPv4 and IPv6 packets.

This is a global (device-wide) setting, which means that it applies to all Layer 3 interfaces (that are not otherwise configured with their own MTU sizes). You can override this setting on a specific Layer 3 interface by using the **ip mtu** command in interface configuration mode on that interface.

Use the **no** form of this command in configuration mode (config) to reset the MTU size to the default value on all Layer 3 interfaces (except those with manually set MTU sizes).

Examples

The following example sets the MTU size globally at 5000 bytes:

```
device# configure terminal  
device(config)# ip mtu 5000  
device(config)#
```

The following example displays the device configuration that is running currently on the device. In this example, the MTU size is set globally at 5000 bytes:

```
device# show running-config ip mtu

ip mtu 5000
!
device#
```

ip mtu (Extreme ONE OS Switching interface configuration)

Use the **ip mtu** command to set the size of the maximum transmission unit (MTU) for all Layer 3 (network layer) transactions on a specific interface.

Syntax

```
ip mtu bytes  
no ip mtu
```

Command Default

The MTU size on the interface is 9194 bytes.

Parameters

bytes
Size in bytes. Use an integer from 1280 to 9194.

Modes

Ethernet interface configuration (*config-if-eth-name*)

Ethernet subinterface configuration (*config-subif-eth-name*)

Virtual Ethernet interface configuration (*config-if-ve-name*)

Usage Guidelines

The MTU is the size of the largest protocol data unit (PDU) that can be communicated in a single network layer transaction. The MTU relates to (but is not identical to) the largest frame that can be transported on the data link layer (such as an Ethernet frame). This setting applies to all IPv4 and IPv6 packets.

This setting determines the MTU size that is allowed on the specified interface. It overrides the global (device-wide) setting, which is either 9194 by default or is set by the **ip mtu** command in configuration mode (*config*).

If the interface is part of a VE, change the IPv4 MTU only at the VE interface but not at the physical port. All member ports of a VE inherit the VE-interface IPv4 MTU value.

Use the **no** form of this command to reset the MTU size on the interface to the default value.

Examples

The following example sets the MTU size on Ethernet interface 0/1 at 5000 bytes:

```
device# configure terminal
device(config)# interface ethernet 0/1
device(config-if-eth-0/1)# ip mtu 5000
device(config-if-eth-0/1)#
```

The following example sets the MTU size on VLAN subinterface 300 under Ethernet interface 0/1 at 6000 bytes:

```
device# configure terminal
device(config)# interface ethernet 0/1
device(config-if-eth-0/1)# subinterface vlan 300
device(config-subif-eth-0/1.300)# ip mtu 6000
device(config-subif-eth-0/1.300)#
```

The following example sets the MTU size on Virtual Ethernet interface 100 at 7000 bytes:

```
device# configure terminal
device(config)# interface ve 100
device(config-if-ve-100)# ip mtu 7000
device(config-if-ve-100)#
```

The following example displays the configuration that is running currently on the device. In this example, the global MTU size is overridden for three interfaces (Ethernet 0/1, its subinterface VLAN 300, and Virtual Ethernet 100):

```
device# show running-config

interface ethernet 0/1
  ip mtu 5000
  subinterface vlan 300
    ip mtu 6000
    no shutdown
  !
  no shutdown
!
interface ve 100
  ip mtu 7000
  no shutdown
!
device#
```

The **show running-config** command displays the global MTU size when it is not the default value. If you change the global MTU size but do *not* change the MTU size of an interface, the **show running-config** command does not display the global MTU value at the interface level.

ipv4 address anycast-gateway (Virtual Ethernet interface configuration)

Use the **ipv4 address anycast-gateway** command to configure an IPv4 address as an anycast gateway address on a Virtual Ethernet (VE) interface.

Syntax

```
ipv4 address ipv4-addr anycast-gateway  
no ipv4 address ipv4-addr anycast-gateway
```

Command Default

The interface has no IPv4 anycast gateway addresses configured.

Parameters

ipv4-addr

IPv4 address. Use IPv4 CIDR notation (*A.B.C.D/M*), where *M* is the prefix length in the range of 0–32 for IPv4 addresses. For example, use an IPv4 address of 10.0.0.0 and a prefix length of 8 to specify addresses in the range of 10.0.0.0 to 10.255.255.255.

Modes

Virtual Ethernet interface configuration (config-if-ve-*name*)

Usage Guidelines

Use the **no** form of this command to remove an IPv4 address that is configured as an anycast gateway address on a VE interface.

This command applies only to VE interfaces.

Examples

The following example configures two IPv4 addresses as anycast gateway addresses on VE interface 100:

```
device# configure terminal  
device(config)# interface ve 100  
device(config-if-ve-100)# ipv4 address 22.22.22.22/24 anycast-gateway  
device(config-if-ve-100)# ipv4 address 22.22.23.22/24 anycast-gateway  
device(config-if-ve-100)#
```

The following example removes two IPv4 addresses that are configured as anycast gateway addresses on VE interface 100:

```
device# configure terminal  
device(config)# interface ve 100  
device(config-if-ve-100)# no ipv4 address 22.22.22.22/24 anycast-gateway
```

```
device(config-if-ve-100)# no ipv4 address 22.22.23.22/24 anycast-gateway
device(config-if-ve-100)#
```

The following example displays the configuration that is currently running on VE interface 100. Two IPv4 addresses are configured as anycast gateway addresses on the interface:

```
device# show running-config interface ve 100

interface ve 100
  ipv4 address 22.22.22.22/24 anycast-gateway
  ipv4 address 22.22.23.22/24 anycast-gateway
  shutdown
device#
```

ipv4 address dhcp (Extreme ONE OS Switching)

Use the **ipv4 address dhcp** command to set an IPv4 address via DHCP on an Ethernet interface or a management interface.

Syntax

```
ipv4 address dhcp  
no ipv4 address dhcp
```

Command Default

DHCP IPv4 address configuration is disabled on the interface.

Parameters

This command has no keywords or variables.

Modes

Ethernet interface configuration (config-if-eth-*name*)

Management interface configuration (config-if-management-*name*)

Usage Guidelines

This command enables IPv4 DHCP configuration. The interface will get the IPv4 address only, and the gateway address is not provided by the DHCP server.

If an IPv4 address is already configured on the interface, you must use the **no ipv4 address** command on the interface before using this command.

To remove the IPv4 address DHCP configuration from the interface, use the **no** form of this command.

To set an IPv4 address on an Ethernet interface without using DHCP configuration, see [ipv4 address \(interface configuration\)](#) on page 685.

Examples

The following example enables DHCP IPv4 address configuration on Ethernet interface 0/1:

```
device# configure terminal  
device(config)# interface ethernet 0/1  
device(config-if-eth-0/1) ipv4 address dhcp  
device(config-if-eth-0/1)#
```

The following example displays the configuration that is currently running on Ethernet interface 0/1:

```
device# show running-config interface ethernet 0/1

interface ethernet 0/1
    ipv4 address dhcp
no shutdown
device#
```

ipv4 address (interface configuration)

Use the **ipv4 address** command to set an IPv4 address on an interface.

Syntax

```
ipv4 address ipv4-address [ secondary ]
```

```
no ipv4 address ipv4-address [ secondary ]
```

Command Default

The interface has no primary or secondary IPv4 addresses configured.

Parameters

ipv4-address

IPv4 address. Use IPv4 CIDR notation (*A.B.C.D/M*), where *M* is the prefix length in the range of 0–32 for IPv4 addresses. For example, use an IPv4 address of 10.0.0.0 and a prefix length of 8 to specify addresses in the range of 10.0.0.0 to 10.255.255.255.

secondary

(Optional) Specifies that the IPv4 address is a secondary address. A primary address must already be configured, and the secondary address must be in the primary address subnet.

Modes

Ethernet interface configuration (*config-if-eth-name*)

Port Channel interface configuration (*config-if-po-name*)

Virtual Ethernet interface configuration (*config-if-ve-name*)

Loopback interface configuration (*config-if-lo-name*)

Usage Guidelines

Typically, you must configure a primary IPv4 address for an interface. If the interface connects to multiple subnets, you must configure primary and secondary IPv4 addresses on the interface so that the subnets can communicate with each other through the interface.

The primary and secondary IPv4 addresses assigned to the interface can be on the same network segment. Different interfaces on your device must be on different network segments.

To remove an IPv4 address (and, optionally, a secondary IPv4 address if it exists) from an Ethernet interface, use the no form of this command.

Examples

The following example configures two primary IPv4 addresses and two secondary IPv4 addresses on Ethernet interface 0/1:

```
device# configure terminal
device(config)# interface ethernet 0/1
device(config-if-eth-0/1)# ipv4 address 22.22.22.22/24
device(config-if-eth-0/1)# ipv4 address 22.22.22.23/24 secondary
device(config-if-eth-0/1)# ipv4 address 22.22.23.22/24
device(config-if-eth-0/1)# ipv4 address 22.22.23.23/24 secondary
device(config-if-eth-0/1)#
```

The following example displays the configuration that is currently running on Ethernet interface 0/1. Two primary IPv4 addresses and two secondary IPv4 addresses are configured on the interface:

```
device# show running-config interface ethernet 0/1

interface ethernet 0/1
  ipv4 address 22.22.22.22/24
  ipv4 address 22.22.22.23/24 secondary
  ipv4 address 22.22.23.22/24
  ipv4 address 22.22.23.23/24 secondary
device#
```

ipv4 anycast-gateway-mac

Use the **ipv4 anycast-gateway-mac** command to configure the IPv4 static anycast gateway (SAG) MAC address for all Virtual Ethernet (VE) interfaces with SAG IP configuration.

Syntax

ipv4 anycast-gateway-mac *mac-address*

Command Default

The IPv4 SAG MAC address is set to 02:e0:52:00:01:00.

Parameters

mac-address

MAC address. Use *xx:xx:xx:xx:xx:xx* notation (six groups of two hexadecimal digits separated by colons). For example, 01:23:45:67:89:ab.

Modes

Global configuration (config)

Usage Guidelines

To restore the IPv4 SAG MAC address to the default value, use the **no** form of this command.

Examples

The following example sets the IPv4 SAG MAC address to 00:01:02:03:04:05:

```
device# configure terminal
device(config)# ipv4 anycast-gateway-mac 00:01:02:03:04:05
device(config)#
```

The following example displays the configuration of the IPv4 SAG MAC address that is currently running on the device:

```
device# show running-config ipv4 anycast-gateway-mac

ipv4 anycast-gateway-mac 00:01:02:03:04:05
!
device#
```

ipv4 neighbor (interface configuration)

Use the **ipv4 neighbor** command to configure the address resolution protocol (ARP) IPv4 address and MAC address on an interface.

Syntax

```
ipv4 neighbor ipv4-address mac-address  
no ipv4 neighbor ipv4-address mac-address
```

Parameters

ipv4-address

IPv4 address. Use the *a.b.c.d* format. For example, 1.1.1.10. You must use a unicast IPv4 address.

mac-address

MAC address. Use *xx:xx:xx:xx:xx:xx* notation (six groups of two hexadecimal digits separated by colons). For example, 01:23:45:67:89:ab.

Modes

Ethernet interface configuration (*config-if-eth-name*)

Internal interface configuration (*config-if-int-name*)

Port Channel interface configuration (*config-if-po-name*)

Virtual Ethernet interface configuration (*config-if-ve-name*)

Usage Guidelines

ARP establishes correspondences between pairs of network-layer addresses (Layer 3) and LAN hardware addresses (Layer 2 MAC addresses).

Use the **no** form of this command to remove the ARP IPv4 address and MAC address from the specified interface.

To display the ARP entries in the neighbor caches for Virtual Routing and Forwarding (VRF) instances, see [show ipv4 neighbor vrf](#) on page 1024.

Examples

The following example configures the ARP IPv4 address and MAC address on Ethernet interface 0/1:

```
device# configure terminal  
device(config)# interface ethernet 0/1  
device(config-int-eth-0/1)# ipv4 neighbor 1.1.1.10 3c:fd:fe:e4:37:a8  
device(config-int-eth-0/1)#
```


The following example displays the configuration of Ethernet interface 0/1 that is running currently on the device. In this example, the ARP IPv4 address and MAC address on the interface are set to 1.1.1.10 and 3c:fd:fe:e4:37:a8 respectively:

```
device# show running-config interface ethernet 0/1

  ipv4 address 1.1.1.1/24
  ipv4 neighbor 1.1.1.10 3c:fd:fe:e4:37:a8
  no shutdown
device#
```

ipv4 neighbor reachable-time (interface configuration)

Use the **ipv4 neighbor reachable-time** command to configure the address resolution protocol (ARP) IPv4 neighbor reachable-time configuration on an interface.

Syntax

```
ipv4 neighbor reachable-time seconds  
no ipv4 neighbor reachable-time
```

Command Default

The reachable time is set to 1200 seconds.

Parameters

seconds

Reachable time (in seconds). Use an integer from 30 to 3600.

Modes

Ethernet interface configuration (config-if-eth-*name*)

Internal interface configuration (config-if-int-*name*)

Port Channel interface configuration (config-if-po-*name*)

Virtual Ethernet interface configuration (config-if-ve-*name*)

Usage Guidelines

ARP establishes correspondences between pairs of network-layer addresses (Layer 3) and LAN hardware addresses (Layer 2 MAC addresses).

Use the no form of this command to reset the reachable time to the default value.

Examples

The following example configures the ARP IPv4 neighbor reachable time to 300 seconds on Ethernet interface 0/1:

```
device# configure terminal  
device(config)# interface ethernet 0/1  
device(config-int-eth-0/1)# ipv4 neighbor reachable-time 300  
device(config-int-eth-0/1)#
```

The following example displays the configuration of Ethernet interface 0/1 that is running currently on the device. In this example, the ARP IPv4 neighbor reachable time is set to 300 seconds:

```
device# show running-config interface ethernet 0/1

interface ethernet 0/1
  mtu 1600
  ipv4 neighbor reachable-time 300
  no shutdown
!
device#
```

ipv6 address anycast-gateway (Virtual Ethernet interface configuration)

Use the **ipv6 address anycast-gateway** command to configure an IPv6 address as an anycast gateway address on a Virtual Ethernet (VE) interface.

Syntax

```
ipv6 address ipv6-addr anycast-gateway  
no ipv6 address ipv6-addr anycast-gateway
```

Command Default

The interface has no IPv6 anycast gateway addresses configured.

Parameters

ipv6-addr

IPv6 address. Use IPv6 CIDR notation (*x:x:x:x/M*), where *M* is the prefix length in the range of 0–128 for IPv6 addresses. For example, use an IPv6 address of 2001::100 and a prefix length of 64 to specify addresses in the range of 2001:0000:0000:0000:0000:0000:0000:0000 to 2001:0000:0000:0000:ffff:ffff:ffff:ffff.

Modes

Virtual Ethernet interface configuration (config-if-ve-*name*)

Usage Guidelines

This command applies only to VE interfaces.

Use the **no** form of this command to remove an IPv6 address that is configured as an anycast gateway address on a VE interface.

Examples

The following example configures two IPv6 addresses as anycast gateway addresses on VE interface 100:

```
device# configure terminal  
device(config)# interface ve 100  
device(config-if-ve-100)# ipv6 address 2001::100/64 anycast-gateway  
device(config-if-ve-100)# ipv6 address 2002::100/64 anycast-gateway  
device(config-if-ve-100)#
```

The following example removes two IPv6 addresses that are configured as anycast gateway addresses on VE interface 100:

```
device# configure terminal  
device(config)# interface ve 100
```

```
device(config-if-ve-100)# no ipv6 address 2001::100/64 anycast-gateway
device(config-if-ve-100)# no ipv6 address 2002::100/64 anycast-gateway
device(config-if-ve-100)#
```

The following example displays the configuration that is running currently on VE interface 100. Two IPv6 addresses are configured as anycast gateway addresses on the interface:

```
device# show running-config interface ve 100

...
interface ve 100
  ipv6 address 2001::100/64 anycast-gateway
  ipv6 address 2002::100/64 anycast-gateway
  shutdown
...
device#
```

ipv6 address auto-config (Extreme ONE OS Switching)

Use the **ipv6 address auto-config** command to set an IPv6 address automatically on an Ethernet interface or a management interface.

Syntax

```
ipv6 address auto-config  
no ipv6 address auto-config
```

Command Default

Automatic IPv6 address configuration is disabled on the interface.

Parameters

This command has no keywords or variables.

Modes

Ethernet interface configuration (config-if-eth-*name*)

Management interface configuration (config-if-management-*name*)

Usage Guidelines

To set an IPv6 address on an Ethernet interface without using automatic configuration, see [ipv6 address \(interface configuration\)](#) on page 697.

Examples

The following example enables automatic IPv6 address configuration on Ethernet interface 0/1:

```
device# configure terminal  
device(config)# interface ethernet 0/1  
device(config-if-eth-0/1) ipv6 address auto-config  
device(config-if-eth-0/1)#
```

The following example displays the configuration that is currently running on Ethernet interface 0/1:

```
device# show running-config interface ethernet 0/1  
  
interface ethernet 0/1  
    ipv6 address auto-config  
no shutdown  
device#
```

ipv6 address dhcp (Extreme ONE OS Switching)

Use the **ipv6 address dhcp** command to set an IPv6 address via DHCP on an Ethernet interface or a management interface.

Syntax

```
ipv6 address dhcp  
no ipv6 address dhcp
```

Command Default

DHCP IPv6 address configuration is disabled on the interface.

Parameters

This command has no keywords or variables.

Modes

Ethernet interface configuration (config-if-eth-*name*)

Management interface configuration (config-if-management-*name*)

Usage Guidelines

This command enables IPv6 DHCP configuration. The interface will get the IPv6 address only, and the gateway address is not provided by the DHCP server.

If an IPv6 address is already configured on the interface, you must use the **no ipv6 address** command on the interface before using this command.

To remove the IPv6 address DHCP configuration from the interface, use the **no** form of this command.

To set an IPv6 address on an Ethernet interface without using DHCP configuration, see [ipv6 address \(interface configuration\)](#) on page 697.

Examples

The following example enables DHCP IPv6 address configuration on Ethernet interface 0/1:

```
device# configure terminal  
device(config)# interface ethernet 0/1  
device(config-if-eth-0/1) ipv6 address dhcp  
device(config-if-eth-0/1)#
```

The following example displays the configuration that is currently running on Ethernet interface 0/1:

```
device# show running-config interface ethernet 0/1

interface ethernet 0/1
    ipv6 address dhcp
no shutdown
device#
```


ipv6 address (interface configuration)

Use the **ipv6 address** command to set an IPv6 address on an Ethernet interface.

Syntax

```
ipv6 address ipv6-address [ secondary | eui-64 ]  
no ipv6 address ipv6-address
```

Command Default

The interface has no primary, secondary, or EUI-64 IPv6 addresses configured.

Parameters

ipv6-address

IPv6 address. Use IPv6 CIDR notation (*x.x::x.x/M*), where *M* is the prefix length in the range of 0–128 for IPv6 addresses. For example, use an IPv6 address of 2001::100 and a prefix length of 64 to specify addresses in the range of 2001:0000:0000:0000:0000:0000:0000 to 2001:0000:0000:0000:ffff:ffff:ffff:ffff.

You must specify a unicast address or a link local address. A link local address must have a prefix length of 64.

secondary

(Optional) Configures the IPv6 address as a secondary address. A primary address must already be configured, and the secondary address must be in the primary address subnet.

eui-64

(Optional) Configures the IPv6 address in Extended Unique Identifier 64-bit (EUI-64) format. A EUI-64 address must be only a prefix. For example, 2003::/64 or fe80::/64.

Modes

Ethernet interface configuration (config-if-eth-*name*)

Port Channel interface configuration (config-if-po-*name*)

Virtual Ethernet interface configuration (config-if-ve-*name*)

Loopback interface configuration (config-if-lo-*name*)

Usage Guidelines

Use this command to set the IPv6 address for an interface. This can be a global address or a link local address. The EUI-64 option generates the address based on the prefix supplied and the MAC address of the interface.

Typically, you must configure a primary IPv6 address for an interface. If the interface connects to multiple subnets, you must configure primary and secondary IPv6 addresses on the interface so that the subnets can communicate with each other through the interface.

The primary and secondary IPv6 addresses assigned to the interface can be on the same network segment. Different interfaces on your device must be on different network segments.

To remove an IPv6 address (and, optionally, a secondary IPv6 address or EUI-64 address if it exists) from an Ethernet interface, use the **no** form of this command.

To set an IPv6 address automatically on an Ethernet interface, see [ipv6 address auto-config \(Extreme ONE OS Switching\)](#) on page 694.

Examples

The following example configures two primary IPv6 addresses, two secondary IPv6 addresses, and two EUI-64 IPv6 addresses (including a link local address) on Ethernet interface 0/1 as well as a link local IPv6 address on Ethernet interface 0/2:

```
device# configure terminal
device(config)# interface ethernet 0/1
device(config-if-eth-0/1)# ipv6 address 2001::100/64
device(config-if-eth-0/1)# ipv6 address 2001::101/64 secondary
device(config-if-eth-0/1)# ipv6 address 2002::100/64
device(config-if-eth-0/1)# ipv6 address 2002::101/64 secondary
device(config-if-eth-0/1)# ipv6 address 2003::/64 eui-64
device(config-if-eth-0/1)# ipv6 address fe80::/64 eui-64
device(config-if-eth-0/1)# exit
device(config)# interface ethernet 0/2
device(config-if-eth-0/2)# ipv6 address fe80::f264:26ff:fef5:c805/64
device(config-if-eth-0/2)#
```

The following example displays the configuration that is currently running on Ethernet interface 0/1. Two primary IPv6 addresses, two secondary IPv6 addresses, and two EUI-64 addresses (including a link local address) are configured on Ethernet interface 0/1:

```
device# show running-config interface ethernet 0/1

interface ethernet 0/1
  ipv6 address 2001::100/64
  ipv6 address 2001::101/64 secondary
  ipv6 address 2002::100/64
  ipv6 address 2002::100/64 secondary
  ipv6 address 2003::/64 eui-64
  ipv6 address fe80::/64 eui-64
  no shutdown
device#
```

ipv6 anycast-gateway-mac

Use the **ipv6 anycast-gateway-mac** command to configure the IPv6 static anycast gateway (SAG) MAC address for all Virtual Ethernet (VE) interfaces with SAG IPv6 configuration.

Syntax

ipv6 anycast-gateway-mac *mac-address*

Command Default

The IPv6 SAG MAC address is set to 02:e0:52:00:01:00.

Parameters

mac-address

MAC address. Use *xx:xx:xx:xx:xx:xx* notation (six groups of two hexadecimal digits separated by colons). For example, 01:23:45:67:89:ab.

Modes

Global configuration (config)

Usage Guidelines

To restore the IPv6 SAG MAC address to the default value, use the **no** form of this command.

Examples

The following example sets the IPv6 SAG MAC address to 00:01:02:03:04:05:

```
device# configure terminal
device(config)# ipv6 anycast-gateway-mac 00:01:02:03:04:05
device(config)#
```

The following example displays the configuration of the IPv6 SAG MAC address that is running currently on the device:

```
device# show running-config ipv6 anycast-gateway-mac

ipv6 anycast-gateway-mac 00:01:02:03:04:05
!
device#
```

ipv6 neighbor (interface configuration)

Use the **ipv6 neighbor** command to configure a static entry in the IPv6 neighbor discovery cache for an interface.

Syntax

```
ipv6 neighbor ipv6-address mac-address  
no ipv6 neighbor ipv6-address mac-address
```

Parameters

ipv6-address

IPv6 address. Use the *x:x::x:x* format. For example, 1111::1. You must use a unicast or link-local IPv6 address.

mac-address

MAC address. Use *xx:xx:xx:xx:xx:xx* notation (six groups of two hexadecimal digits separated by colons). For example, 01:23:45:67:89:ab.

Modes

Ethernet interface configuration (*config-if-eth-name*)

Internal interface configuration (*config-if-int-name*)

Port Channel interface configuration (*config-if-po-name*)

Virtual Ethernet interface configuration (*config-if-ve-name*)

Usage Guidelines

Neighbor discovery establishes correspondences between pairs of network-layer addresses (Layer 3) and LAN hardware addresses (Layer 2 MAC addresses).

Use the **no** form of this command to remove the static entry in the IPv6 neighbor discovery cache from the specified interface.

To display the neighbor discovery entries in the neighbor caches for Virtual Routing and Forwarding (VRF) instances, see [show ipv6 neighbor vrf](#) on page 1030.

Examples

The following example configures the neighbor discovery IPv6 address and MAC address on Ethernet interface 0/1:

```
device# configure terminal  
device(config)# interface ethernet 0/1  
device(config-int-eth-0/1)# ipv6 neighbor 1111::1 aa:aa:aa:aa:aa:aa  
device(config-int-eth-0/1)#
```

The following example displays the configuration of Ethernet interface 0/1 that is running currently on the device. In this example, the neighbor discovery IPv6 address and MAC address on the interface are set to 1111::1 and aa:aa:aa:aa:aa:aa respectively:

```
device# show running-config interface ethernet 0/1

  ipv6 address 1111::10/64
  ipv6 neighbor 1111::1 aa:aa:aa:aa:aa:aa
  no shutdown
device#
```

ipv6 neighbor reachable-time (interface configuration)

Use the **ipv6 neighbor reachable-time** command to configure the neighbor discovery protocol (NDP) IPv6 neighbor reachable-time configuration on an interface.

Syntax

```
ipv6 neighbor reachable-time seconds  
no ipv6 neighbor reachable-time
```

Command Default

The reachable time is set to 1200 seconds.

Parameters

seconds

Reachable time (in seconds). Use an integer from 30 to 3600.

Modes

Ethernet interface configuration (config-if-eth-*name*)

Internal interface configuration (config-if-int-*name*)

Port Channel interface configuration (config-if-po-*name*)

Virtual Ethernet interface configuration (config-if-ve-*name*)

Usage Guidelines

NDP establishes correspondences between pairs of network-layer (Layer 3) addresses and LAN hardware addresses (Layer 2 MAC addresses).

Use the no form of this command to reset the reachable time to the default value.

Examples

The following example configures the NDP IPv6 neighbor reachable time to 300 seconds on Ethernet interface 0/1:

```
device# configure terminal  
device(config)# interface ethernet 0/1  
device(config-int-eth-0/1)# ipv6 neighbor reachable-time 300  
device(config-int-eth-0/1)#
```

The following example displays the configuration of Ethernet interface 0/1 that is running currently on the device. In this example, the NDP IPv6 neighbor reachable time is set to 300 seconds:

```
device# show running-config interface ethernet 0/1

interface ethernet 0/1
  mtu 1600
  ipv6 neighbor reachable-time 300
  no shutdown
!
device#
```

ipv6-router-advertisement

Use the **ipv6-router-advertisement** command to enter IPv6 router advertisement configuration (config-if-*name*-ipv6-router-advertisement) mode on an interface.

Syntax

ipv6-router-advertisement

Parameters

This command has no keywords or variables.

Modes

Ethernet interface configuration (config-if-eth-*name*)

Port Channel interface configuration (config-if-po-*name*)

Virtual Ethernet interface configuration (config-if-ve-*name*)

Usage Guidelines

Router advertisements are ICMPv6 messages sent by an IPv6 enabled router and are used by hosts to detect routers on an attached link. This functionality is similar to the IPv4 ICMP Router Discovery protocol.

Examples

The following example enters IPv6 router advertisement configuration (config-if-*name*-ipv6-router-advertisement) mode on Ethernet interface 0/1:

```
device# configure terminal
device(config)# interface ethernet 0/1
device(config-if-eth-0/1)# ipv6-router-advertisement
device(config-if-eth-0/1-ipv6-router-advertisement)#
```

The following example displays the configuration of Ethernet interface 0/1 that is running currently on the device. In this example, IPv6 router advertisement is configured on the interface:

```
device# show running-config interface ethernet 0/1

!
interface ethernet 0/1
  ipv6 address 1111::10/64
  ipv6 neighbor 1111::1 00:04:96:eb:c4:51
  ipv6-router-advertisement
    interval 100 min 50
    enable
    prefix 100::1/64
    prefix 100::1/63
    valid-lifetime 200
```



```
!  
no shutdown  
!  
device#
```

keepalive-interval

Use the **keepalive-interval** command to set the BGP keepalive interval to a nondefault value for all members of a BGP peer group within a Virtual Routing and Forwarding (VRF) instance.

Syntax

keepalive-interval *seconds*

no keepalive-interval *seconds*

Command Default

The BGP keepalive interval is set to 60 seconds.

Parameters

seconds

Duration of keepalive interval. Use an integer from 1 to 65535. This value must be less than the hold time (which is 180 seconds by default or the value set by the **hold-time** command). For example, if the hold time is set to 90 seconds, you can set the keepalive interval from 1 to 89 seconds.

Modes

VRF BGP peer group timer configuration (config-vrf-bgp-pg-timers)

Usage Guidelines

Use this command to set the keepalive interval to apply to all members of a specified BGP peer group. The keepalive interval is how often a local neighbor sends a BGP keepalive message (also called a BGP heartbeat packet) to its remote neighbor to verify its availability. This command is related to the **hold-time** command, which defines the maximum time a router will wait for a keepalive or update message from its peer before declaring the session down. Keepalive messages are sent to keep this timer from expiring.

BGP starts sending keepalive messages after a successful Open message exchange and TCP connection establishment. Specifically, upon successful negotiation of the hold time, which is a part of the Open message exchange, a keepalive message is sent. If the hold time is set to zero, no keepalive messages are sent.

The keepalive interval is typically set to one third of the hold time. For example, if the hold time is 180 seconds, the keepalive interval should be set to 60 seconds.

Use the **no** form of this command to reset the keepalive interval to the default value for all members of the BGP peer group. For established sessions, the default value takes effect.

Examples

The following example configures a BGP peer group named `group1` under a VRF instance named `violet`. This example sets the keepalive interval to one second for the members of `group1`:

```
device# configure terminal
device(config)# vrf violet
device(config-vrf-violet)# router bgp
device(config-vrf-bgp)# peer-group group1
device(config-vrf-bgp-pg)# timers
device(config-vrf-bgp-pg-timers)# keepalive interval 1
device(config-vrf-bgp-pg-timers)#
```

The following example displays the configuration of a VRF instance named `violet` that is running currently on the device. In this example, the `group1` peer group sets a keepalive interval of one second for each member:

```
device# show running-config vrf violet

vrf violet
!
  router bgp
    peer-group group1
      timers
        keepalive interval 1
!
device#
```

keepalive (MLAG configuration)

Use the **keepalive** command to specify the delay, interval, and multiplier settings for multichassis link aggregation (MLAG) keepalive transmissions.

Syntax

```
keepalive { interval milliseconds | delay milliseconds | multiplier multiple }  
  
no keepalive [ interval | delay | multiplier ]
```

Command Default

The interval is 300 milliseconds. The delay is 1000 milliseconds. The multiplier is 3.

Parameters

interval *milliseconds*

Time between any two keepalive messages. Use an integer from 0 to 1000000.

delay *milliseconds*

Initial delay before the MLAG peers begin exchanging keepalive messages with each other. This lets them synchronize their state and establish a shared understanding of the network topology. This setting should be long enough for the peers to complete their initial synchronization, but short enough for the network to be brought quickly into a stable state. Use an integer from 100 to 1000000.

multiplier *multiple*

Value used to multiply the interval (as in *milliseconds* x *multiple*) to determine the total time taken to bring down a keepalive session. Use an integer from 1 to 50 for *multiple*.

Modes

MLAG configuration (config-mlag)

Usage Guidelines

Use this command to change the settings for successive keepalive messages sent by this MLAG peer to the other MLAG peers to check that the link between the two is operating or to prevent the link from being broken.

To restore the delay, interval, or multiplier settings for MLAG keepalive transmissions to their default values, use the no form of this command.

Examples

The following example sets the interval to 1000 milliseconds, the delay to 100 milliseconds, and the multiplier to 4:

```
device# configure terminal
device(config)# mlag
device(config-mlag)# keepalive interval 1000 delay 100 multiplier 4
device(config-mlag)#
```

The following example restores the delay, interval, and multiplier settings to their default values:

```
device# configure terminal
device(config)# mlag
device(config-mlag)# no keepalive delay interval multiplier
device(config-mlag)#
```

The following example displays the keepalive settings that are running currently on a device:

```
device# show running-config mlag

mlag
    keepalive interval 1000 delay 100 multiplier 4
device#
```

keepalive (MLAG peer configuration)

Use the **keepalive (MLAG peer configuration)** command to specify the MLAG peer keepalive name for multichassis link aggregation group (MLAG) keepalive transmissions and enter MLAG peer keepalive configuration mode.

Syntax

keepalive *keepalive-name*

no keepalive *keepalive-name*

Command Default

No keepalive name for MLAG peer keepalive transmissions is configured on the device.

Parameters

keepalive-name

MLAG peer keepalive name. Use an arbitrary sequence of letters, numerals, underscores, hyphens, or periods. No special characters or symbols are allowed. Do not exceed 64 characters.

Modes

MLAG peer configuration (config-mlag-peer-*name*)

Usage Guidelines

Keepalive messages are sent by this MLAG peer to the other MLAG peers to check that the link between the two is operating or to prevent the link from breaking.

Two keepalive sessions are configured in an MLAG. The primary keepalive is used for Inter-Switch Link (ISL) connectivity, and the secondary keepalive is the management interface (by default).

You can use the **show running-config mlag** command to display the MLAG configuration that is running currently on a device.

To remove the MLAG peer keepalive configuration, use the no form of this command.

Examples

The following example configures an MLAG peer named peer01 and configure a keepalive named keepalive01:

```
device# configure terminal
device(config)# mlag
device(config-mlag)# peer peer01
```

```
device(config-mlag-peer-peer01)# keepalive keepalive01
device(config-mlag-peer-keepalive-keepalive01)#
```

The following example deletes a keepalive named keepalive01 from the MLAG peer:

```
device# configure terminal
device(config)# mlag
device(config-mlag)# peer peer01
device(config-mlag-peer-peer01)# no keepalive keepalive01
device(config-mlag-peer-peer01)#
```

The following example displays the MLAG configuration that is running currently on a device. In this example, the MLAG peer on the device has keepalive mode configured:

```
device# show running-config mlag

mlag
    peer peer01
    !
        keepalive keepalive01
    !
device#
```

l3vni (VRF instance type configuration)

Use the **l3vni** command to configure the Layer 3 virtual network identifier (VNI) for a virtual routing and forwarding (VRF) instance.

Syntax

```
l3vni vni  
no l3vni vni
```

Command Default

The specified VRF instance has no Layer 3 VNI.

Parameters

vni

Number that will identify the VRF instance. Use an integer from 1 to 16777215.

Modes

VRF instance type configuration (config-vrf-*type*)

Usage Guidelines

Layer 3 VNIs enable multiple VRF instances to operate in a shared environment while ensuring logical segmentation between them. Each VRF instance requires a unique Layer 3 VNI.

Use the **no** form of this command to delete the Layer 3 VNI from the specified VRF instance.

Examples

The following example configures a VRF instance named `tenant_1` as an Ethernet VPN (EVPN) and sets 10100 as the Layer 3 VNI to distinguish it from other VRF instances:

```
device# configure terminal  
device(config)# vrf tenant_1  
device(config-vrf-tenant_1)# router bgp  
device(config-vrf-bgp)# instance-type evpn  
device(config-vrf-bgp-vpn-evpn)# nve new  
device(config-vrf-bgp-evpn-nve-new)# l3vni 10100  
device(config-vrf-bgp-evpn-nve-new)#
```

The following example displays the configuration of a VRF instance named `tenant_1` that is running currently on the device:

```
device# show running-config  
...
```



```
vrf tenant_1
  router bgp
    instance-type evpn
    nve new
      l3vni 10100
    !
  !
!
...
device#
```

l3vni (VRF instance type NVO configuration)

Use the **l3vni** command to configure the Layer 3 virtual network identifier (VNI) for a Network Virtualization Overlay (NVO) instance within a VRF instance.

Syntax

```
l3vni vni  
no l3vni vni
```

Command Default

The specified NVO instance has no Layer 3 VNI.

Parameters

vni

Number that will identify the NVO instance. Use an integer from 1 to 16777215.

Modes

VRF instance type NVO configuration (config-vrf-type-nvo)

Usage Guidelines

Layer 3 VNIs enable multiple NVO instances to operate in a shared environment while ensuring logical segmentation between them. Each NVO instance requires a unique Layer 3 VNI.

Use the **no** form of this command to delete the Layer 3 VNI from the specified NVO instance.

Examples

The following example configures a VRF instance named `tenant_1` as an Ethernet VPN (EVPN). In this example, the VRF instance is configured with an NVO instance named `nvo1` that uses 10100 as the Layer 3 VNI to distinguish it from other NVO instances:

```
device# configure terminal  
device(config)# vrf tenant_1  
device(config-vrf-tenant_1)# instance-type evpn  
device(config-vrf-evpn)# nvo nvo1  
device(config-vrf-evpn-nvo)# l3vni 10100  
device(config-vrf-evpn-nvo)#
```

The following example displays the configuration of a VRF instance named `tenant_1` that is running currently on the device:

```
device# show running-config
```

```
...  
!  
vrf tenant_1  
    instance-type evpn  
        nvo nvol  
            l3vni 10100  
!  
...  
device#
```

lacp min-links

Use the **lacp min-links** command to configure the minimum number of links that is required for a link aggregation group (LAG).

Syntax

```
lacp min-links number-of-links  
no lacp min-links
```

Command Default

A LAG requires at least one link.

Parameters

number-of-links

Minimum number of links. Use an integer from 1 to 64.

Modes

Port Channel interface configuration (config-if-po-*name*)

Usage Guidelines

You must enable LACP globally on the device before performing any LACP configuration. For details, see [protocol lacp](#) on page 885.

To restore the required minimum number of links to the default value, use the no form of this command.

Examples

The following example enables LACP globally on the device and configures a LAG with a range of 10 and a required minimum of two links:

```
device# configure terminal  
device(config)# protocol lacp  
device(config)# interface port-channel 10  
device(config-if-po-10)# lacp min-links 2  
device(config-if-po-10)#
```

The following example displays the LAG configuration that is running currently on the device. In this example, LACP is enabled globally, a LAG with a range of 10 and a required minimum of two links is configured, and Ethernet interface 0/29 is added to the LAG in active mode:

```
device# show running-config lacp  
  
protocol lacp  
interface port-channel 10
```

```
lacp min-links 2
no shutdown
!
interface ethernet 0/29
  channel-group 10 mode active
  no shutdown
!
device#
```

lacp port-priority

Use the **lacp port-priority** command to configure the priority for a port that is a member of a link aggregation group (LAG).

Syntax

```
lacp port-priority priority-number  
no lacp port-priority
```

Command Default

The priority is set to 32768.

Parameters

priority-number

Priority. A lower number takes priority over a higher number. Use an integer from 1 to 65535.

Modes

Ethernet interface configuration (config-if-eth-*name*)

Usage Guidelines

An LACP port priority is configured on each port using LACP. The port priority determines which ports to put in standby mode when a hardware limitation prevents all compatible ports from aggregating. A link with higher priority (smaller in value) gets preference over a link with lower priority (greater in value).

You must enable LACP globally on the device before performing any LACP configuration. For details, see [protocol lacp](#) on page 885.

To restore the priority to the default value, use the no form of this command.

Examples

The following example adds Ethernet interface 0/29 to a LAG that has port channel number 10, specifies active LACP mode on the interface, and sets the port priority of the interface to 12345:

```
device# configure terminal  
device(config)# interface ethernet 0/29  
(config-if-eth-0/29)# channel-group 10 mode active  
(config-if-eth-0/29)# lacp port-priority 12345  
(config-if-eth-0/29)#
```

The following example displays the LAG configuration that is running currently on the device. In this example, Ethernet interface 0/29 is added to a LAG in active mode and with a port priority of 12345:

```
device# show running-config lacp

protocol lacp
  lacp system-priority 32768
!
interface port-channel 10
  lacp min-links 1
  no shutdown
!
interface ethernet 0/29
  channel-group 10 mode active
  lacp port-priority 12345
  no shutdown
!
device#
```

lacp interval

Use the **lacp interval** command to configure the protocol data unit (PDU) interval for a port that is a member of a link aggregation group (LAG).

Syntax

```
lacp interval { fast | slow }  
no lacp interval
```

Command Default

The interval is set to **slow** (one PDU every 30 seconds).

Parameters

fast

One PDU every second.

slow

One PDU every 30 seconds.

Modes

Ethernet interface configuration (config-if-eth-*name*)

Usage Guidelines

You must enable LACP globally on the device before performing any LACP configuration. For details, see [protocol lacp](#) on page 885.

To restore the interval to the default value, use the no form of this command.

Examples

The following example adds Ethernet interface 0/1 to a LAG that has port channel number 10, specifies active LACP mode on the interface, and sets the interval of the interface to one PDU per second:

```
device# configure terminal  
device(config)# interface ethernet 0/1  
(config-if-eth-0/1)# channel-group 10 mode active  
(config-if-eth-0/1)# lacp interval fast  
(config-if-eth-0/1)#
```


The following example displays the LAG configuration that is running currently on the device. In this example, Ethernet interface 0/1 is added to a LAG in active mode and with a fast interval (one PDU per second):

```
device# show running-config lacp

protocol lacp
lacp system-priority 32768
!
interface port-channel 10
lacp min-links 1
no shutdown
!
interface ethernet 0/1
channel-group 10 mode active
lacp interval fast
no shutdown
!
device#
```

The following example adds Ethernet interface 0/2 to a LAG that has port channel number 20, specifies active LACP mode on the interface, and sets the interval of the interface to one PDU per 30 seconds:

```
device# configure terminal
device(config)# interface ethernet 0/2
(config-if-eth-0/2)# channel-group 20 mode active
(config-if-eth-0/2)# lacp interval slow
(config-if-eth-0/2)#
```

The following example displays the LAG configuration that is running currently on the device. In this example, Ethernet interface 0/2 is added to a LAG in active mode and with a show interval (one PDU per 30 second):

```
device# show running-config lacp

protocol lacp
lacp system-priority 32768
!
interface port-channel 20
lacp min-links 1
no shutdown
!
interface ethernet 0/2
channel-group 20 mode active
lacp interval fast
no shutdown
!
device#
```

lacp system-mac

Use the **lacp system-mac** command to configure the MAC address of a link aggregation group (LAG).

Syntax

lacp system-mac *mac-address*

no lacp system-mac

Command Default

The specified LAG uses a MAC address whose last octet is one more than that of the MAC address of the immediately previous interface.

Parameters

mac-address

MAC address to assign to the LAG. Use *xx:xx:xx:xx:xx:xx* notation (six groups of two hexadecimal digits separated by colons). For example, 01:23:45:67:89:ab.

Modes

Port Channel interface configuration (config-if-po-*name*)

Usage Guidelines

You must enable LACP globally on the device before performing any LACP configuration. For details, see [protocol lacp](#) on page 885.

To restore the system MAC address of the specified LAG to the default value, use the no form of this command.

Examples

The following example enables LACP globally on the device and configures a LAG with port channel number 10 and a system MAC address of 01:23:45:67:89:ab:

```
device# configure terminal
device(config)# protocol lacp
device(config)# interface port-channel 10
device(config-if-po-10)# lacp system-mac 01:23:45:67:89:ab
device(config-if-po-10)#
```

The following example displays the LAG configuration that is running currently on the device. In this example, LACP is enabled globally, a LAG with port channel number 10

and a system MAC address of 01:23:45:67:89:ab is configured, and Ethernet interface 0/29 is added to the LAG in active mode:

```
device# show running-config lacp

protocol lacp
interface port-channel 10
    lacp system-mac 01:23:45:67:89:ab
    no shutdown
!
interface ethernet 0/29
    channel-group 10 mode active
    no shutdown
!
device#
```

lacp system-priority

Use the **lacp system-priority** command to configure the priority that determines which system resolves conflicts in the choice of link aggregation groups (LAGs).

Syntax

```
lacp system-priority priority-number  
no lacp system-priority
```

Command Default

The priority is set to 32768.

Parameters

priority-number

Priority. A lower number takes priority over a higher number. Use an integer from 1 to 65535.

Modes

Global configuration (config) and Port Channel interface configuration (config-if-po-*name*)

Usage Guidelines

You must enable LACP globally on the device before performing any LACP configuration. For details, see [protocol lacp](#) on page 885.

To restore the system priority globally or for the specified LAG to the default value, use the no form of this command in the appropriate command mode.

Examples

The following example enables LACP globally on the device and configures a LAG with channel number 10 and a system priority of 34567:

```
device# configure terminal  
device(config)# protocol lacp  
(config)# lacp system-priority 123  
(config)# interface port-channel 10  
(config-if-po-10)# lacp system-priority 34567  
device(config-if-po-10)#
```

The following example displays the LAG configuration that is running currently on the device. In this example, the global system priority is set to 123, and the system priority for a LAG with channel number 10 is set to 34567:

```
device# show running-config lacp

protocol lacp
  lacp system-priority 123
interface port-channel 10
  lacp system-priority 34567
  no shutdown
!
interface ethernet 0/29
  channel-group 10 mode active
  no shutdown
!
device#
```

ldap cacert

Use the **ldap cacert** command to configure an LDAP CA certificate for an IAH third party virtual machine (TPVM).

Syntax

ldap cacert *filename*

Command Default

An LDAP certificate authority (CA) certificate is not configured on the current VM.

Parameters

filename

Name of the CA certificate. This file must reside in the disk://iah directory.

Modes

IAH VM configuration (config-iah-vm-*name*)

Usage Guidelines

LDAP configuration is set irrespective of out of band (OOB) connectivity and the IPv4 or IPv6 static or DHCP configuration.

The CA certificate must be copied to the disk://iah/ directory on the device. The secure LDAP server configuration will fail if no CA certificate is configured.

To add a new LDAP CA certificate configuration, you must remove the prior LDAP CA certificate configuration if one exists. Note that configuration is not allowed when the VM is in the deployed state.

To display details about a specific VM, see [show iah vm](#) on page 1006.

Examples

The following example configure an LDAP CA certificate named bigco-ca-cert.pem for a TPVM on the device:

```
device# configure terminal
device(config)# iah
device(config-iah)# vm TPVM
device(config-iah-vm-TPVM)# ldap cacert disk://iah/bigco-ca-cert.pem
device(config-iah-vm-TPVM)#
```

The following example displays the IAH configuration that is running currently on the device. In this example, an LDAP CA certificate named `bigco-ca-cert.pem` is specified for a TPVM named `Ubuntu1`:

```
device# show running-config iah

iah
  vm Ubuntu1
    tpvm
    hostname ganga
    timezone Etc/GMT
    host: 1.1.1.1
    port: 639
    basedn: -
    rootdn: -
    rootdnpw: -
    secure: true
    ldap-cert: disk://iah/bigco-ca-cert.pem
  !
!
device#
```

ldap host

Use the **ldap host** command to configure LDAP for an IAH third party virtual machine (TPVM).

Syntax

```
ldap host { ipv4-addr | ipv6-addr | domain-name } [ port port-num ]
[ basedn basedn-name ] [ rootdn rootdn-name ] [ { rootdnpw rootdn-
password | rootdnpw-hashed rootdn-hashedpassword } ] [ secure ]
```

Command Default

LDAP is not configured on the current VM.

Parameters

ip-address

IPv4 address or IPv6 address of the LDAP server.

- For an IPv4 address, use IPv4 notation (*A.B.C.D*). For example, 10.0.0.0.
- For an IPv6 address, use the *x:x::x:x* format. For example, 1001::1.

domain-name

Fully qualified domain name of the LDAP server.

port *port-num*

(Optional) Port number for the LDAP connection. Use an integer from 1 to 65535. The default port is 389.

basedn *basedn-name*

(Optional) LDAP base domain name.

rootdn *rootdn-name*

(Optional) LDAP root domain name.

rootdnpw *rootdn-password*

(Optional) LDAP root domain password.

rootdnpw-hashed *rootdn-hashedpassword*

(Optional) Encrypted LDAP root domain password.

secure

(Optional) Enables a secure LDAP connection.

Modes

IAH VM configuration (*config-iah-vm-name*)

Usage Guidelines

LDAP configuration is set irrespective of out of band (OOB) connectivity and the IPv4 or IPv6 static or DHCP configuration.

The CA certificate must be copied to the disk://iah/ directory. The secure LDAP server configuration will fail if no CA certificate is configured.

To add a new LDAP host configuration, you must remove the prior LDAP host configuration if one exists. Note that configuration is not allowed when the VM is in the deployed state.

To display details about a specific VM, see [show iah vm](#) on page 1006.

Examples

The following example configure an LDAP server for a TPVM on the device:

```
device# configure terminal
device(config)# iah
device(config-iah)# vm TPVM
device(config-iah-vm-TPVM)# ldap host 1.1.1.1 port 639 secure
device(config-iah-vm-TPVM)#
```

The following example displays the IAH configuration that is running currently on the device. In this example, LDAP server host 1.1.1.1 is specified for a TPVM named Ubuntu1:

```
device# show running-config iah

iah
  vm Ubuntu1
    tpvm
    hostname ganga
    timezone Etc/GMT
    host: 1.1.1.1
    port: 639
    basedn: -
    rootdn: -
    rootdnpw: -
    secure: true
    ldap-cert: disk://iah/bigco-ca-cert.pem
  !
!
device#
```

lifetime

Use the **lifetime** command to configure the lifetime for the IPv6 router advertisement on an interface.

Syntax

```
lifetime seconds  
no lifetime
```

Command Default

The lifetime is 1800 seconds.

Parameters

seconds

Number of seconds. Use an integer from 0 to 9000.

Modes

IPv6 router advertisement configuration (config-if-*name*-ipv6-router-advertisement)

Usage Guidelines

Router advertisements are ICMPv6 messages sent by an IPv6 enabled router that are used by hosts to detect routers on an attached link. This functionality is similar to the IPv4 ICMP Router Discovery protocol.

Examples

The following example sets a lifetime of 200 seconds for IPv6 router advertisement on Ethernet interface 0/1:

```
device# configure terminal  
device(config)# interface ethernet 0/1  
device(config-if-eth-0/1)# ipv6-router-advertisement  
device(config-if-eth-0/1-ipv6-router-advertisement)# lifetime 200  
device(config-if-eth-0/1-ipv6-router-advertisement)#
```

The following example displays the configuration of Ethernet interface 0/1 that is running currently on the device. In this example, a lifetime of 200 seconds for IPv6 router advertisement is configured on the interface:

```
device# show running-config interface ethernet 0/1  
  
!  
interface ethernet 0/1  
  ipv6 address 1111::10/64  
  ipv6 neighbor 1111::1 00:04:96:eb:c4:51  
  ipv6-router-advertisement
```

```
hop-limit 133
lifetime 200
interval 100 min 50
enable
prefix 100::1/64
prefix 100::1/63
    valid-lifetime 200
!
no shutdown
!
device#
```

link-error-disable

Use the **link-error-disable** command to configure link dampening on an Ethernet interface.

Syntax

```
link-error-disable toggle-threshold number sampling-time seconds wait-  
time seconds  
no link-error-disable
```

Command Default

Link dampening is disabled on the specified interface.

Parameters

toggle-threshold

Number of times that a port link state cycles from up to down and then down to up before the wait time is activated. Use an integer from 1 through 50.

sampling-time

Number of seconds during which the specified toggle threshold can occur before the wait time is activated. Use an integer from 1 through 65535. Selecting an appropriate sampling time is important, because fast flaps are either missed by a fast timer or over-counted by a slow timer.

wait-time

Number of seconds that the port remains disabled (down) before it becomes enabled. Use an integer from 0 through 65535. 0 seconds keeps the port down until you re-enable it manually. 300 seconds (5 minutes) gives you time to investigate before the port re-enables.

Modes

Ethernet interface configuration (config-if-eth-*name*)

Usage Guidelines

Use this command to dampen port flapping on the interface, including the threshold of flapping to shut down the port and the interval during which it remains shut down.

Link dampening lets you configure a waiting period before a port, whose link goes down then up, becomes enabled again. This feature is available for all front interfaces on the device.

If the link state toggles from down to up or from up to down for a specified number of times within a specified period, the interface is physically disabled for the specified wait time. Once the wait time expires, the port link state is re-enabled. However, if the

wait time is set to zero (0) seconds, or if you want to re-enable the port before the wait period expires, the port must be manually re-enabled.

This command is supported only on front Ethernet interfaces. It is not supported on the management interface or on virtual interfaces.

To see whether link dampening is enabled on a specific interface, see [show interface ethernet link-error-disable](#) on page 1012.

Use the **no** form of the command to disable link dampening on the specified interface.

Examples

The following example configures link dampening on Ethernet interface 0/1. The toggle threshold is set to 10 times, the sampling time is set to 3 seconds, and the wait time is set to 300 seconds:

```
device# configure terminal
device(config)# interface ethernet 0/1
device(config-if-eth 0/1)# link-error-disable toggle-threshold 10 sampling-time 3 wait-
time 300
device(config-if-eth 0/1)#
```

The following example displays details about Ethernet interface 0/1. In this example, Link Error Disable: ON shows that link dampening is enabled on the interface:

```
device# show interface ethernet 0/1

ethernet 0/1 Admin state UP, Oper state UP
Interface index is 16777248 (0x1000020)
MTU 9216 bytes
Hardware is Ethernet  mac address 50:0a:9c:97:48:1d
Current Speed  400G
Description: Ethernet 0/1
FEC:
    Mode: RS-FEC
    Uncorrected Blocks: 18
    Corrected Blocks: 55475033
Link Fault Signaling: ON
Link Fault Status: No Fault
Link Error Disable: ON
Auto Negotiation: ON
Carrier Transitions: 7
    LastClear: 0s
Input:
    Total pkts: 1760026664564
    Broadcast pkts: 0
    Discard pkts: 0
    Errors pkts: 0
    FCS Errors: 0
    MCast pkts: 8
    Octets: 962203743127738
    UCast pkts: 1760026664556
    Runt pkts: 0
    CRC Errors: 0
Input Distribution:
    64 byte pkts: 0
    65-127 byte pkts: 482573358216
    128-255 byte pkts: 358308712804
    256-511 byte pkts: 493289637048
```

```
    512-1023 byte pkts: 143312386065
    1024-1518 byte pkts: 181249049849
        Jumbo pkts: 101293520582
Out:
    Total pkts: 1486325156801
    Broadcast pkts: 0
    Discard pkts: 0
    Errors pkts: 0
    MCast pkts: 20485
        Octets: 803095001984686
    UCast pkts: 1486325136316
device#
```

link-fault-signaling

Use the **link-fault-signaling** command to enable Link Fault Signaling (LFS) on an Ethernet port.

Syntax

link-fault-signaling

Command Default

LFS is enabled on all front physical interfaces.

Parameters

This command has no keywords or variables.

Modes

Ethernet interface configuration (config-if-eth-*name*)

Usage Guidelines

LFS is enabled by default on the 8730 platform.

LFS supports speeds 10Gb and higher.

LFS reflects the health of the link. LFS is a physical layer protocol that enables communication on a link between two Ethernet devices. It operates between the remote Ethernet device Reconciliation Sublayer (RS) and the local Ethernet device RS. When configured on an Ethernet port, the port can detect and report fault conditions on transmit and receive ports.

To see whether LFS is enabled on a specific port as well as the general link fault status on that port, see [show interface ethernet link-error-disable](#) on page 1012. To display the link fault signaling (LFS) configuration for each front port, see [show link-fault-signaling](#) on page 1045. To view statistics for LFS detection, see [show counters link-fault-signaling](#) on page 989.

This command is not allowed on the management interface.

Examples

The following example configures Ethernet interface 0/1. In this example, LFS is enabled on the interface:

```
device# configure terminal
device(config)# interface ethernet 0/1
device(config-if-eth 0/1)# no shutdown
device(config-if-eth 0/1)# description Ethernet interface 0/1
device(config-if-eth 0/1)# fec rs-fec
```

```

device(config-if-eth 0/1)# mtu 4000
device(config-if-eth 0/1)# speed 25000
device(config-if-eth 0/1)# loopback-mode
device#

```

The following example displays the link fault signaling (LFS) configuration for each front port:

```

device# show link-fault-signaling

Rx and Tx link-fault-signaling status
Port          Link-Faults
=====
Eth 0         ON
Eth 0/1       OFF
Eth 0/2       ON
Eth 0/3       ON
device#

```

The following example displays details about Ethernet interface 0/1. In this example, LFS is enabled on the interface, and there are no LFS faults:

```

device# show int eth 0/1
ethernet 0/1 Admin state UP, Oper state UP
Interface index is 16777248 (0x1000020)
MTU 9216 bytes
Hardware is Ethernet  mac address 50:0a:9c:97:48:1d
Current Speed  400G
Description: Ethernet 0/1
FEC:
    Mode: RS-FEC
    Uncorrected Blocks: 18
    Corrected Blocks: 55475033
Link Fault Signaling: ON
Link Fault Status: No Fault
Auto Negotiation: ON
Carrier Transitions: 7
    LastClear: 0s
Input:
    Total pkts: 1760026664564
    Broadcast pkts: 0
    Discard pkts: 0
    Errors pkts: 0
    FCS Errors: 0
    MCast pkts: 8
    Octets: 962203743127738
    UCast pkts: 1760026664556
    Runt pkts: 0
    CRC Errors: 0
Input Distribution:
    64 byte pkts: 0
    65-127 byte pkts: 482573358216
    128-255 byte pkts: 358308712804
    256-511 byte pkts: 493289637048
    512-1023 byte pkts: 143312386065
    1024-1518 byte pkts: 181249049849
    Jumbo pkts: 101293520582
Out:
    Total pkts: 1486325156801
    Broadcast pkts: 0
    Discard pkts: 0
    Errors pkts: 0
    MCast pkts: 20485
    Octets: 803095001984686

```



```
device# UCast pkts: 1486325136316
```

listen-range (BGP peer group configuration)

Use the **listen-range** command to specify a range of trusted IPv4 or IPv6 addresses to be added dynamically as neighbors to a BGP peer group within a Virtual Routing and Forwarding (VRF) instance.

Syntax

```
listen-range { ipv4-address/prefix-length | ipv6-address/prefix-length }  
    [ listen-limit limit ]
```

```
no listen-range { ipv4-address/prefix-length | ipv6-address/prefix-  
    length } [ listen-limit ]
```

Command Default

No listen range is configured for the specified BGP peer group.

Parameters

ipv4-address/prefix-length

Range of IPv4 addresses. Use the *a.b.c.d* format for *ipv4-address*, and use an integer from 0 to 32 for the prefix. For example, 10.1.1.0/24. You must include the slash (/).

ipv6-address/prefix-length

Range of IPv6 addresses. Use the *x:x::x:x* format for *ipv6-address*, and use an integer from 0 to 128 for the prefix. For example, 2001::1/64. You must include the slash (/).

listen-limit *limit*

Number of BGP neighbors accepted for this listen range. Use an integer from 1 to 2400.

Modes

VRF BGP peer group configuration (config-vrf-bgp-pg)

Usage Guidelines

This command automatically configures a range of IPv4 or IPv6 addresses so that you need not configure each of those neighbors manually. Dynamic neighbors inherit any configuration that is applied to their peer group. In larger BGP networks, dynamic neighbors can reduce the complexity of CLI configuration and save system resources.

Use the **no** form of this command to delete from the BGP peer group the listen range and all dynamic neighbors that were created as part of the listen range.

**Caution**

Peering relationships with dynamic neighbors are terminated if the peer group is deleted.

Examples

The following example configures two BGP peer groups (group1 and group2) under a VRF instance named violet. In this example, group1 is assigned 200 as its remote autonomous system (AS) number and the IPv4 address range 10.1.1.0/24 as a listen range. Also, group2 is assigned 300 as its remote AS number and the IPv6 address range 2001::1/64 as a listen range:

```
device# configure terminal
device(config)# vrf violet
device(config-vrf-violet)# router bgp
device(config-vrf-bgp)# peer-group group1
device(config-vrf-bgp-pg)# remote-as 200
device(config-vrf-bgp-pg)# listen-range 10.1.1.0/24
device(config-vrf-bgp-pg)# exit
device(config-vrf-bgp)# peer-group group2
device(config-vrf-bgp-pg)# remote-as 300
device(config-vrf-bgp-pg)# listen-range 2001::1/64
device(config-vrf-bgp-pg)#
```

The following example displays the configuration of a VRF instance named violet that is running currently on the device. In this example, both group1 and group2 have a listen range configured:

```
device# show running-config vrf violet

vrf violet
!
  router bgp
    peer-group group1
      listen-range 10.1.1.0/24
    peer-group group2
      listen-range 2001::1/64
!
device#
```

lldp

Use the **lldp** command to enter LLDP configuration (config-lldp) mode.

Syntax

lldp

no lldp

Command Default

LLDP on the device is not configured.

Parameters

This command has no keywords or variables.

Modes

Global configuration (config)

Usage Guidelines

Use this command to enter the mode for configuring LLDP settings. These settings include enablement of packet transmission and reception, global LLDP transmit interval, the period that a receiving device holds information from your device before discarding it, the LLDP management address, and suppression of optional TLV advertisements.

Use the no form of this command to remove the LLDP configuration.



Note

The no form of this command restores the default values or removes the configurations of all entries within this mode.

Examples

The following example enters LLDP configuration (config-lldp) mode:

```
device# configure terminal
device(config)# lldp
device(config-lldp)#
```

load-balance (Extreme ONE OS Switching global configuration)

Use the **load-balance** command to configure load balancing parameters on the device.

Syntax

```
load-balance ethernet { [ dst-mac ] [ etype ] [ src-mac ] [ vlan ] }  
load-balance ipv4 { [ dst-ip ] [ dst-l4-port ] [ protocol ] [ src-ip ]  
    [ src-l4-port ] }  
load-balance ipv6 { [ dst-ip ] [ dst-l4-port ] [ nxt-hdr ] [ src-ip ]  
    [ src-l4-port ] }  
no load-balance { ethernet | ipv4 | ipv6 }
```

Command Default

All load balancing parameters (hash types) are enabled for Ethernet, IPv4, and IPv6.

Parameters

dst-mac

Selection based on destination MAC address.

src-mac

Selection based on source MAC address.

vlan

Selection based on VLAN ID.

etype

Selection based on Ethertype.

dst-ip

Selection based on destination IPv4 or IPv6 address.

dst-l4-port

Selection based on destination TCP/UDP port.

protocol

Selection based on IP protocol field.

src-ip

Selection based on source IPv4 or IPv6 address.

src-l4-port

Selection based on source TCP/UDP port.

nxt-hdr

Selection based on next header field.

Modes

Global configuration (config)

Usage Guidelines

This is a global (device-wide) setting.

Use the **no** form of this command to disable load balancing for Ethernet, IPv4, or IPv6.

Examples

The following example configures Ethernet load balancing (using all hash types) globally on the device:

```
device# configure terminal
device(config)# load-balance ethernet dst-mac src-mac vlan etype
device(config)#
```

The following example displays the configuration that is running currently on the device. In this example, load balancing is configured globally (using all hash types) for Ethernet, IPv4, and IPv6 on the device:

```
device# show running-config load-balance

load-balance ethernet dst-mac src-mac vlan etype
load-balance ipv4 dst-ip dst-l4-port protocol src-ip src-l4-port
load-balance ipv6 dst-ip dst-l4-port nxt-hdr src-ip src-l4-port
!
device#
```

local-as-forced

Use the **local-as-forced** command to override the router's local autonomous system (AS) number with a forced 2-byte AS number to establish sessions with peers of the BGP peer group for which this command is configured.

Syntax

local-as-forced *as-number*

no local-as-forced *as-number*

Command Default

The local AS number override feature is disabled for the specified BGP peer group.

Parameters

as-number

Replacement AS number. Specify an integer from 1 to 65535.

Modes

BGP peer group configuration (config-vrf-bgp-pg)

Usage Guidelines

Use this command to override the router's local AS number of BGP peers to establish sessions with old BGP peers supporting only 2-byte AS numbers.

Use the **no** form of this command to disable the local AS number override so that the peer group members resume using their routers' local AS numbers.



Caution

The **no** form of this command terminates all active BGP sessions that are using the replacement AS number. Improper usage of this command might disrupt service.

Examples

The following example configures a BGP peer group named group1 under a VRF instance named violet. This example uses the **local-as-forced** command to override the routers' configured local ASs for members of group1 and forces them to use 100 as the AS number instead:

```
device# configure terminal
device(config)# vrf violet
device(config-vrf-violet)# router bgp
device(config-vrf-bgp)# peer-group group1
```

```
device(config-vrf-bgp-pg) # local-as-forced 100
device(config-vrf-bgp-pg) #
```

The following example displays the configuration of a VRF instance named violet that is running currently on the device. In this example, members of the group1 peer group are forced to use 100 as the AS number instead of their routers' configured local AS numbers:

```
device# show running-config vrf violet

vrf violet
!
  router bgp
    peer-group group1
      local-as-forced 100
!
device#
```


local-as

Use the **local-as** command to specify the BGP Autonomous System (AS) number where the router resides.

Syntax

local-as *as-number*

no local-as *as-number*

Command Default

No local AS number is set where the router resides.

Parameters

as-number

AS number in either asplain or asdot notation:

- For asplain notation, specify an integer from 1 to 4294967295. For example, 6553700.
- For asdot notation, use the *high-order-16-bit-value-in-decimal.low-order-16-bit-value-in-decimal* format, where the high order and low order values are integers from 1 to 65535. For example, 100.100 (which equals 6553700 in asplain notation).

For details about asplain and asdot notation, see RFC 5396: <https://datatracker.ietf.org/doc/rfc5396/>.

Modes

VRF BGP configuration (config-vrf-bgp)

Usage Guidelines

The **local-as-forced** command under BGP peer group configuration (config-vrf-bgp-pg) mode overrides this setting.

Use the **no** form of this command to remove the local AS number from the router.



Caution

The **no** form of this command terminates all active BGP sessions that are using the currently configured local AS number. Improper usage of this command might disrupt service.

Examples

The following example configures the BGP AS number where the router resides to 65000:

```
device# configure terminal
device(config)# router bgp
device(config-router)# local-as 65000
device(config-router)#
```

The following example displays the configurations of the VRF instances that are running currently on the device. In this example, only one VRF is configured (the default VRF), and its AS number displays in asplain notation (which is the default notation):

```
device# show running-config vrf

vrf default-vrf
  router bgp
    local-as 6553700
!
device#
```

The following example displays the configurations of the VRF instances that are running currently on the device. In this example, only one VRF is configured (the default VRF), and its AS number displays in asdot notation (which is enabled with the **as-notation** command):

```
device# show running-config vrf

vrf default-vrf
  router bgp
    local-as 100.100
!
device#
```

local-pref-eq (routing policy statement bgp conditions configuration)

Use the **local-pref-eq** command to set a BGP local preference value for a routing policy statement to indicate the relative priority of a route compared to other routes.

Syntax

```
local-pref-eq local-preference-value  
no local-pref-eq
```

Command Default

The local preference value is 100.

Parameters

local-preference-value

Local preference value. Use an integer from 0 to 4294967295.

Modes

Routing policy statement BGP conditions configuration (config-route-policy-policy-name-stmt-number-conditions-bgp-conditions)

Usage Guidelines

In BGP, the local-preference attribute influences outbound traffic within an Autonomous System (AS). It is a well-known, discretionary attribute that is exchanged only between internal BGP (iBGP) peers. Higher local preference values are preferred, which guide the router to select the exit point with the highest value when multiple paths exist to a destination. A router within an AS receives multiple routes to the same destination from different neighbors (such as via different eBGP peers). The router compares the local preference values of these routes. The route with the highest local preference value is selected as the preferred path.

This command configures one of the BGP conditions inside a route policy statement. This condition can be used to match routes that have a specified local preference value.

To remove the match condition configured for the local preference attribute, use the **no** form of this command.

Examples

The following example create a routing policy named map1. This map contains statement 10, which has a BGP condition to match routes with a local-preference value of 100:

```
device# configure terminal
device(config)# route-policy
device(config-route-policy)# policy map1
device(config-route-policy-policy-map1)# statement 10
device(config-route-policy-policy-map1-stmt-10)# conditions
device(config-route-policy-policy-map1-stmt-10-conditions)# bgp-conditions
device(config-route-policy-policy-map1-stmt-10-conditions-bgp-conditions)# local-pref-eq
100
device(config-route-policy-policy-map1-stmt-10-conditions-bgp-conditions)#
```

The following example displays the routing policy configuration that is running currently on the device. In this example, a routing policy named map1 has a policy statement with sequence number 10, which contains a BGP condition to match routes with a local-preference value of 100:

```
device# show running-config route-policy

route-policy
  policy map1
    statement 10
      conditions
        bgp-conditions
          local-pref-eq 100
device#
```

loopback-mode

Use the **loopback-mode** command to enable loopback mode on an Ethernet port.

Syntax

loopback-mode

Command Default

Loopback mode is disabled on all front physical interfaces.

Parameters

This command has no keywords or variables.

Modes

Ethernet interface configuration (config-if-eth-*name*)

Usage Guidelines

Use the **no** form of this command to disable loopback mode on the interface.

Examples

The following example configures Ethernet interface 0/1. In this example, loopback mode is enabled on the interface:

```
device# configure terminal
device(config)# interface ethernet 0/1
device(config-if-eth 0/1)# no shutdown
device(config-if-eth 0/1)# description Ethernet interface 0/1
device(config-if-eth 0/1)# fec rs-fec
device(config-if-eth 0/1)# link-fault-signaling
device(config-if-eth 0/1)# mtu 4000
device(config-if-eth 0/1)# speed 25000
device(config-if-eth 0/1)# loopback-mode
device#
```

l2 mac-move-detection

Use the **l2 mac-move-detection** command to configure Layer 2 mac-move-detection on a device.

Syntax

```
l2 mac-move-detection { [ action shutdown | auto-recovery-time seconds |  
  enable | interval seconds | threshold moves }  
  
no l2 mac-move-detection { [ action shutdown | auto-recovery-time seconds  
  | enable | interval seconds | threshold moves }
```

Parameters

action shutdown

Action to be taken if the mac-move limit is reached. The only action supported in this release is a shutdown of the port. The default action (if you do not specify this keyword) is to generate a RASlog only.

auto-recovery-time *seconds*

Time (in seconds) until the port state is reset to up. Use an integer from 0 to 1800. 0 disables auto-recovery. The default is 300 (5 minutes).

enable

Enables or disables mac-move-detection. Disabled by default.

interval *seconds*

Interval (in seconds) after which the mac-move events are deleted. Use an integer from 10 to 120. The default is 10.

threshold *moves*

Event count (maximum number of moves) after which the port is shut down or a RASlog is triggered. Use an integer from 3 to 500. The default is 20.

Modes

Global system configuration (config-system-global)

Usage Guidelines

To disable a mac-move-detection setting, use the **no** form of this command.

The no form of this command restores the default value.

Examples

The following example enables mac-move-detection on the device:

```
device# configure terminal  
device# (config)# system
```

```
device(config-system)# global
device(config-system-global)# 12 mac-move-detection enable
device(config-system-global)#
```

mac-address-table aging-time

Use the **mac-address-table aging-time** command to set the period that entries can remain in the MAC address table before being removed.

Syntax

```
mac-address-table aging-time seconds  
no mac-address-table aging-time seconds
```

Command Default

The MAC address table aging time is 1800 seconds.

Parameters

seconds

Amount of time before an entry is discarded. Use 0 (to disable aging) or an integer from 60 to 38400.

Modes

Global configuration (config)

Usage Guidelines

Use this command to set the maximum amount of time that a MAC address remains in the MAC address table. When this time expires, the MAC address is deleted.

Use the **no** form of this command to restore the aging time to the default value.

Examples

The following example configures an aging time of 120 seconds for entries in the MAC address table:

```
device# configure terminal  
device(config)# system  
device(config-system)# global  
device(config-system-global)# mac-address-table aging-time 120  
device(config-system-global)#
```

The following example displays the setting for the MAC address table aging time that is running currently on the device. In this example, a MAC address ages out after 120 seconds and is then discarded:

```
device# show running-config system global  
  
system  
  global
```



```
mac-address-table aging-time 120
device#
```

mac (MLAG configuration)

Use the **mac (MLAG configuration)** command to specify the multichassis link aggregation (MLAG) MAC address.

Syntax

mac *mac-address*

no mac

Command Default

No MLAG MAC address is configured on the device.

Parameters

mac-address

MLAG MAC address in 48-bit dotted notation (*xxxxx.xxxx.xxxx*).

Modes

MLAG configuration (config-mlag)

Usage Guidelines

To remove the MLAG MAC address, use the no form of this command.

Examples

The following example configures the MLAG MAC address on the device:

```
device# configure terminal
device(config)# mlag
device(config-mlag)# mac a001.a002.a003
device(config-mlag)#
```

The following example displays the MLAG MAC address that is running currently on the device:

```
device# show running-config mlag

mlag
  mac 0022.3344.5566
device#
```

maintenance-mode (system configuration)

Use the **maintenance-mode** command to enter Maintenance system configuration (config-system-maintenance-mode) mode.

Syntax

maintenance-mode

Command Default

Maintenance mode is not configured on the device.

Parameters

This command has no keywords or variables.

Modes

System configuration (config-system)

Usage Guidelines

This command enters the CLI command mode for configuring maintenance mode on the device, which allows you to enter the **enable-on-reload** command, which puts the device in maintenance mode upon system reload.

You use the maintenance mode feature when a device needs maintenance for reasons such as an image upgrade. When maintenance mode is active, all front panel ports are disabled to isolate the device from all connected devices on the network.

During maintenance mode, all physical ports will be in the "link down" state. The connected devices detect that the link is down and divert the traffic to a peer MLAG node.

This command does not appear in the running configuration unless there are submode parameters configured.

Examples

The following example enters Maintenance system configuration (config-system-maintenance-mode) mode:

```
device# configure terminal
device(config)# system
device(config-system)# maintenance-mode
device(config-system-maintenance-mode)#
```

The following example displays the system configuration that is running currently on the device. In this example, maintenance mode is configured so that the device enters maintenance mode upon reload:

```
device# show running-config system maintenance-mode

system
  maintenance-mode
    enable-on-reload
  !
!
device#
```

managed-config-flag

Use the **managed-config-flag** command to enable the managed address configuration flag for IPv6 router advertisement on an interface.

Syntax

managed-config-flag

no managed-config-flag

Command Default

The managed address configuration flag is disabled.

Parameters

This command has no keywords or variables.

Modes

IPv6 router advertisement configuration (config-if-*name*-ipv6-router-advertisement)

Usage Guidelines

Router advertisements are ICMPv6 messages sent by an IPv6 enabled router that are used by hosts to detect routers on an attached link. This functionality is similar to the IPv4 ICMP Router Discovery protocol.

When this feature is enabled, the managed address configuration (M) flag is set in the router advertisement. This flag indicates that there are addresses available via DHCPv6.

Examples

The following example enables the managed address configuration flag for IPv6 router advertisement on Ethernet interface 0/1:

```
device# configure terminal
device(config)# interface ethernet 0/1
device(config-if-eth-0/1)# ipv6-router-advertisement
device(config-if-eth-0/1-ipv6-router-advertisement)# managed-config-flag
device(config-if-eth-0/1-ipv6-router-advertisement)#
```

The following example displays the configuration of Ethernet interface 0/1 that is running currently on the device. In this example, the managed address configuration flag is enabled for IPv6 router advertisement is configured on the interface:

```
device# show running-config interface ethernet 0/1

!
interface ethernet 0/1
  ipv6 address 1111::10/64
```

```
ipv6 neighbor 1111::1 00:04:96:eb:c4:51
ipv6-router-advertisement
  hop-limit 133
  lifetime 200
  managed-config-flag
  interval 100 min 50
  enable
  prefix 100::1/64
  prefix 100::1/63
    valid-lifetime 200
  !
  no shutdown
  !
device#
```

management-address

Use the **management-address** command to set the global LLDP management address or the LLDP management address on an interface.

Syntax

```
management-address { ipv4-address | ipv6-address }  
no management-address
```

Command Default

No LLDP management address is set globally or on the specified interface.

Parameters

ipv4-address

IPv4 management address. Use the *a.b.c.d* format. For example, 10.1.1.1.

ipv6-address

IPv6 management address. Use the *x:x::x:x* format. For example, 1001::1.

Modes

LLDP configuration (config-lldp)

Ethernet interface LLDP configuration (config-if-eth-*name*-lldp)

Usage Guidelines

Use this command to set the global address or the address on an interface that is transmitted as the type-length-value (TLV) management address in LLDP frames. This allows easier management and troubleshooting of LLDP communications. This IPv4 or IPv6 address must be valid and reachable for successful LLDP communication.

If this address is configured globally, all LLDP enabled interfaces inherit it; otherwise only configured interfaces use it as their management address. If both global and per interface management addresses are configured, the interface address takes precedence.

Use the no form of this command in the appropriate command mode to remove the LLDP management address globally or from the specified interface.

Examples

The following example sets the global LLDP management address to 10.10.10.10:

```
device# configure terminal  
device(config)# lldp
```

```
device(config-lldp)# management-address 10.10.10.10
device(config-lldp)#
```

The following example sets the LLDP management address on Ethernet interface 0/1 to 10.10.10.10:

```
device# configure terminal
device(config)# interface ethernet 0/1
device(config-eth-0/1-lldp)# lldp
device(config-eth-0/1-lldp)# management-address 10.10.10.10
device(config-eth-0/1-lldp)#
```

The following example displays the LLDP configuration that is running currently on the device. In this example, the global LLDP management address is set to 10.10.10.10:

```
device# show running-config lldp

lldp
  management-address 10.10.10.10
device#
```

The following example displays the configuration for Ethernet interface 0/1 that is running currently on the device. In this example, the LLDP management address for the interface is set to 10.10.10.10:

```
device# show running-config interface ethernet 0/1

interface ethernet 0/1
  lldp
    management-address 10.10.10.10
device#
```


match-as-path-set (routing policy statement bgp conditions configuration)

Use the **match-as-path-set** command to add a BGP condition to a routing policy statement for matching the as-path attribute of routes with members of a specific AS path set.

Syntax

```
match-as-path-set as-path-set-name { any | all | invert }  
no match-as-path-set
```

Command Default

The specified routing policy statement contains no BGP condition for matching members of an AS path set.

Parameters

as-path-set-name

Name of the AS path set.

{ **any** | **all** | **invert** }

Matches any, all, or the inverse of the specified AS path set.

- **any**—If any member in the AS path set is present in the AS path attribute in the route, it is a match.
- **all**—If all members in the AS path set are present in AS path attribute in the route, it is a match.
- **invert**—If no members in the AS path set are present in the AS path attribute in the route, it is a match.

Modes

Routing policy statement BGP conditions configuration (config-route-policy-policy-name-stmt-number-conditions-bgp-conditions)

Usage Guidelines

This command adds a BGP condition to a routing policy statement for matching a list of AS paths that you define. Each statement in a routing policy can have only one BGP actions container. This container can contain multiple actions.

The any, all, or invert keywords determine how the members in the AS path set are matched against the AS path attribute in the route.

To create an AS path set, see [as-path-set](#) on page 449.

To remove the AS path match condition under a specific statement, use the **no** form of this command. Other statements under the same policy can have match conditions to different AS path sets. All AS path match conditions cannot be removed by one **no** command.

Examples

The following example creates a routing policy named `map1` that contains a policy statement with sequence number 10 containing a condition for matching any member of an AS path set named `aspath1`:

```
device# configure terminal
device(config)# route-policy
device(config-route-policy)# policy map1
device(config-route-policy-policy-map1)# statement 10
device(config-route-policy-policy-map1-stmt-10)# conditions
device(config-route-policy-policy-map1-stmt-10-conditions)# bgp-conditions
device(config-route-policy-policy-map1-stmt-10-conditions-bgp-conditions)# match-as-path-set aspath1 any
device(config-route-policy-policy-map1-stmt-10-conditions-bgp-conditions)#
```

The following example displays the routing policy configuration that is running currently on the device. In this example, a routing policy named `map1` has a policy statement with sequence number 10 containing a condition for matching any member of an AS path set named `aspath1`:

```
device# show running-config route-policy

route-policy
  policy map1
    statement 10
      conditions
        bgp-conditions
          match-as-path-set aspath1 any
device#
```

match-community-set (routing policy statement bgp conditions configuration)

Use the **match-community-set** command to add a BGP condition to a routing policy statement for matching the community attribute in a route to the members of the configured community set.

Syntax

```
match-community-set community-set-name  
no match-community-set community-set-name
```

Command Default

The specified routing policy statement contains no BGP condition for matching members of a community set.

Parameters

community-set-name
Name of the community set.

Modes

Routing policy statement BGP conditions configuration (config-route-policy-policy-name-stmt-number-conditions-bgp-conditions)

Usage Guidelines

This command configures a BGP condition that helps to match a community attribute in the route to the community members specified in the community set. You can use this command before creating the community set (forward referencing is allowed).

To create a community set, see [community-set](#) on page 541.

How a community set will be matched depends on the match-set-options configuration inside a community-set definition. For details, see [match-set-options \(routing policy BGP community set configuration\)](#) on page 769.

To remove a community set match configuration from a routing policy statement, use the **no** form of this command.

Examples

The following example creates a routing policy named `map1` that contains a policy statement with sequence number 10. The policy statement contains a condition for matching members of a community set named `comm1`:

```
device# configure terminal
device(config)# route-policy
device(config-route-policy)# policy map1
device(config-route-policy-policy-map1)# statement 10
device(config-route-policy-policy-map1-stmt-10)# conditions
device(config-route-policy-policy-map1-stmt-10-conditions)# bgp-conditions
device(config-route-policy-policy-map1-stmt-10-conditions-bgp-conditions)# match-
community-set comm1
device(config-route-policy-policy-map1-stmt-10-conditions-bgp-conditions)#
```

The following example displays the routing policy configuration that is running currently on the device. In this example, a routing policy named `map1` has a policy statement with sequence number 10. The policy statement contains a condition for matching members of a community set named `comm1`:

```
device# show running-config route-policy

route-policy
  policy map1
    statement 10
      conditions
        bgp-conditions
          match-community-set comm1
device#
```

match-ext-community-set (routing policy statement bgp conditions configuration)

Use the **match-ext-community-set** command to add a BGP condition to a routing policy statement for matching the extended community attribute in a route to the members of the configured extended community set.

Syntax

match-ext-community-set *community-set-name*

no match-ext-community-set *community-set-name*

Command Default

The specified routing policy statement contains no BGP condition for matching members of an extended community set.

Parameters

community-set-name

Name of the community set.

Modes

Routing policy statement BGP conditions configuration (config-route-policy-policy-name-stmt-number-conditions-bgp-conditions)

Usage Guidelines

This command configures a BGP condition that helps to match an extended community attribute in the route to the extended community members specified in the extended community set. You can use this command before creating the extended community set (forward referencing is allowed).

Each statement in a routing policy can have only one BGP actions container. This container can contain multiple actions.

To create an extended community set, see [ext-community-set](#) on page 610.

To remove an extended community set match configuration from a routing policy statement, use the **no** form of this command.

Examples

The following example creates a routing policy named `map1` that contains a policy statement with sequence number 10. The policy statement contains a condition for matching members of an extended community set named `comm1`:

```
device# configure terminal
device(config)# route-policy
device(config-route-policy)# policy map1
device(config-route-policy-policy-map1)# statement 10
device(config-route-policy-policy-map1-stmt-10)# conditions
device(config-route-policy-policy-map1-stmt-10-conditions)# bgp-conditions
device(config-route-policy-policy-map1-stmt-10-conditions-bgp-conditions)# match-ext-
community-set comm1
device(config-route-policy-policy-map1-stmt-10-conditions-bgp-conditions)#
```

The following example displays the routing policy configuration that is running currently on the device. In this example, a routing policy named `map1` has a policy statement with sequence number 10. The policy statement contains a condition for matching members of an extended community set named `comm1`:

```
device# show running-config route-policy

route-policy
  policy map1
    statement 10
      conditions
        bgp-conditions
          match-ext-community-set comm1
device#
```

match-prefix-set (routing policy map statement conditions configuration)

Use the **match-prefix-set** command to match the members of a specified prefix set.

Syntax

```
match-prefix-set prefix-set-name [ any | invert ]  
no match-prefix-set prefix-set-name
```

Command Default

The specified routing policy statement does not specify a prefix set for matching.

Parameters

prefix-set-name

Name of the prefix set.

any

(Optional) If any prefix in the prefix set is present in the set of prefixes in the route, it is a match. If you do not specify either **any** or **invert**, **any** is the default behavior.

invert

(Optional) If no prefix in the prefix set matches the prefixes in the route, it is a match.

Modes

Routing policy statement conditions configuration (config-route-policy-policy-name-stmt-number-conditions)

Usage Guidelines

This command adds a condition to a routing policy statement for matching a list of prefixes defined in a prefix set with the prefixes present in the route.

To create a prefix set, see [prefix-set](#) on page 879.

Matching on all members of a prefix set is not supported.

To remove prefix-set match conditions from a routing policy statement, use the **no** form of this command.

Examples

The following example creates a routing policy named `map1` that contains a policy statement with sequence number 10 containing a condition that uses a prefix set named `prefix1` for matching:

```
device# configure terminal
device(config)# route-policy
device(config-route-policy)# policy map1
device(config-route-policy-policy-map1)# statement 10
device(config-route-policy-policy-map1-stmt-10)# conditions
device(config-route-policy-policy-map1-stmt-10-cond)# match-prefix-set prefix1 any
device(config-route-policy-policy-map1-stmt-10-cond)#
```

The following example displays the routing policy configuration that is running currently on the device. In this example, a routing policy named `map1` contains a policy statement with sequence number 10 that uses a condition specifying a prefix set named `prefix1` for matching:

```
device# show running-config route-policy

route-policy
  policy map1
    statement 10
      conditions
        match-prefix-set prefix1 any
device#
```


match-set-options (routing policy BGP community set configuration)

Use the **match-set-options** command to specify the behavior of a match condition with a BGP community set.

Syntax

```
match-set-options { any | all | invert }  
no match-set-options { any | all | invert }
```

Command Default

Match is true if at least one community in the community attribute of the route has a match in the defined community set.

Parameters

any

Match is true if at least one community in the community attribute of the route has a match in the defined community set.

all

Match is true if every community in the community attribute of the route has a match in the defined community set.

invert

Match is true if no community in the community attribute of the route has a match in the defined community set.

Modes

Routing policy BGP community set configuration (config-route-policy-bgp-set-community-*name*)

Usage Guidelines

To restore the default behavior, use the **no** form of this command.

Examples

The following example creates a BGP community set named **comm1** where match is true if every community in the community attribute of the route has a match in the community set:

```
device# configure terminal  
device(config)# route-policy  
device(config-route-policy)# bgp-defined-sets  
device(config-route-policy-bgp-set)# community-set comm1  
device(config-route-policy-bgp-set-community-comm1)# match-set-options all
```

```
device(config-route-policy-bgp-set-community-comm1)# member 65535 65535:100
device(config-route-policy-bgp-set-community-comm1)#
```

The following example displays the routing policy configuration that is running currently on the device. In this example, the device is configured with one BGP defined set, which is a community set named comm1 where match is true if every community in the community attribute of the route has a match in the community set:

```
device# show running-config route-policy

route-policy
  bgp-defined-sets
    community-set comm1
      member 65535 65535:100
      match-set-options all
device#
```

match-set-options (routing policy BGP extended community set configuration)

Use the **match-set-options** command to specify the behavior of a match condition with an extended BGP extended community set.

Syntax

```
match-set-options { any | all | invert }  
no match-set-options { any | all | invert }
```

Command Default

Match is true if at least one community in the extended community attribute of the route has a match in the defined extended community set.

Parameters

any

Match is true if at least one community in the extended community attribute of the route has a match in the defined extended community set.

all

Match is true if every community in the extended community attribute of the route has a match in the defined extended community set.

invert

Match is true if no community in the extended community attribute of the route has a match in the defined extended community set.

Modes

Routing policy BGP extended community set configuration (config-route-policy-bgp-ext-community-*name*)

Usage Guidelines

To restore the default behavior, use the **no** form of this command.

Examples

The following example creates a BGP extended community set named `ecomm1` where match is true if every community in the community attribute of the route has a match in the extended community set:

```
device# configure terminal  
device(config)# route-policy  
device(config-route-policy)# bgp-defined-sets  
device(config-route-policy-bgp-set)# ext-community-set ecomm1
```

```
device(config-route-policy-bgp-ext-community-ecomm1)# member 10:200 route-target:123:334
device(config-route-policy-bgp-ext-community-ecomm1)# match-set-options any
device(config-route-policy-bgp-ext-community-ecomm1)#
```

The following example displays the routing policy configuration that is running currently on the device. In this example, under `bgp-defined-sets`, an extended community set named `ecomm1` is defined. Members of the extended community set are configured, and a match option is configured as an "any" set named `ecomm1`.

```
device# show running-config route-policy

route-policy
  bgp-defined-sets
    ext-community-set ecomm1
      member 200:1234 route-target:303:5000 10.1.1.10:345 10:200 route-target:123:334
      match-set-options any
device#
```

med-eq (routing policy statement bgp conditions configuration)

Use the **med-eq** command to add a BGP condition to a routing policy statement for matching against a multiple exit discriminator (MED) value.

Syntax

med-eq *med-value*

no med-eq *med-value*

Command Default

The specified routing policy statement contains no BGP condition for matching against a MED value.

Parameters

med-value

MED value. Use an integer from 0 to 4294967295.

Modes

Routing policy statement BGP conditions configuration (config-route-policy-policy-name-stmt-number-conditions-bgp-conditions)

Usage Guidelines

This command configures a BGP condition that matches the MED attribute in the route to the configured value in this command.

MED is a nontransitive traffic engineering attribute that an autonomous system (AS) sets when advertising routes to another peer. MED is a hint to external neighbors about the preferred path into an AS that has multiple entry points and plays an important role in influencing that traffic.

MED is used in route selection only if there are multiple routes learned from the same AS number, and those routes are for the same destination (prefix). If there are multiple routes, the one with the lowest MED value is preferred.

To remove the configured MED match condition from a routing policy statement, use the **no** form of this command.

Examples

The following example creates a routing policy named `map1` that contains a policy statement with sequence number 10. The policy statement contains a condition for matching against a BGP MED value of 20:

```
device# configure terminal
device(config)# route-policy
device(config-route-policy)# policy map1
device(config-route-policy-policy-map1)# statement 10
device(config-route-policy-policy-map1-stmt-10)# conditions
device(config-route-policy-policy-map1-stmt-10-conditions)# bgp-conditions
device(config-route-policy-policy-map1-stmt-10-conditions-bgp-conditions)# med-eq 20
device(config-route-policy-policy-map1-stmt-10-conditions-bgp-conditions)#
```

The following example displays the routing policy configuration that is running currently on the device. In this example, a routing policy named `map1` has a policy statement with sequence number 10. The policy statement contains a condition for matching against a BGP MED value of 20:

```
device# show running-config route-policy

route-policy
  policy map1
    statement 10
      conditions
        bgp-conditions
          med-eq 20
device#
```

member

Use the **member** command to create a double tagged subinterface and assign the specified bridge domain membership.

Syntax

```
member [ ethernet | port-channel ] interface-name vlan vlan-range inner-vlan vlan-range
```

```
no member [ ethernet | port-channel ] interface-name vlan vlan-range inner-vlan vlan-range
```

Command Default

The current bridge domain has no subinterfaces as members.

Parameters

member

Specifies the interface type.

ethernet

(Optional) Specifies the Ethernet interface.

port-channel

(Optional) Specifies the port-channel.

interface-name

Specifies the name of the interface.

vlan *vlan-range*

Specifies the VLAN range. Valid range is 1-4094.

inner-vlan *vlan range*

Specifies the inner VLAN range. Valid range is 1-4094.

Modes

Bridge domain configuration (config-bd-*name*)

Usage Guidelines

To remove the bridge domain membership of an interface (but not the subinterface), use the no form of this command.

Examples

The following example creates a subinterface, adds the sub-interface as a Bridge-domain member, and assigns the port-channel VLAN bridge domain membership:

```
device(config)# bridge-domain 200
device(config-bd-200)# member ethernet 0/2 vlan 100
device(config-bd-200)# member port-channel 1 vlan 100 inner-vlan 200

device(config-bd-200)# do show running-config bridge-domain 200

bridge-domain 200
  member ethernet 0/2 vlan 100
  member port-channel 1 vlan 100 inner-vlan 200
device(config-bd-200)#
```


member nve

Use the **member nve** command to add a Network Virtualization Edge (NVE) endpoint (also called an NVE node) to a Network Virtualization Overlay (NVO) instance.

Syntax

member nve *endpoint-name*

no nve *endpoint-name*

Command Default

The NVO has no NVE endpoints as members.

Parameters

endpoint-name

Name of the NVE endpoint to add to the NVO instance. You can use any combination of 1 to 64 uppercase or lowercase alphabetic characters, numerals, underscores, hyphens, or periods. You must use an alphabetic character or an underscore as the first character.

Modes

NVO configuration (config-nvo-*name*)

Usage Guidelines

Use the **no** form of this command to remove an NVE endpoint from an NVO instance.

Examples

The following example adds an NVE endpoint named base to an NVO instance named DG1:

```
device# configure terminal
device(config)# overlay-networks
device(config-overlay)# nvo DG1
device(config-nvo-DG1)# member nve base
device(config-nvo-DG1)#
```

The following example displays the NVO instances that are running currently on the device:

```
device# show running-config overlay-networks

overlay-networks
  default-nve base
  default-vni-offset 0
  nvo DG1
    member nve base
```

```
!  
nve base  
    source-interface loopback 1  
    vni-domain base  
!  
vni-domain base  
vni-domain DG1  
!  
device#
```

member (routing policy BGP AS path set configuration)

Use the **member** command to add an Autonomous System (AS) number to a BGP AS path set.

Syntax

```
member as-number [ as-number... ]  
no member as-number [ as-number... ]
```

Command Default

The specified BGP AS path has no AS numbers as members.

Parameters

```
as-number [ as-number... ]
```

List of AS numbers to add. Use a space to separate two or more AS numbers. Use either asplain or asdot notation:

- For asplain notation, specify an integer from 1 to 4294967295. For example, 6553700.
- For asdot notation, use the *high-order-16-bit-value-in-decimal.low-order-16-bit-value-in-decimal* format, where the high order and low order values are integers from 1 to 65535. For example, 100.100 (which equals 6553700 in asplain notation).

Regular expressions are also supported. For details about asplain and asdot notation, see [Textual Representation of Autonomous System \(AS\) Numbers, RFC 5396](#).

Modes

Routing policy BGP AS path set configuration (config-route-policy-bgp-set-as-path-set-name)

Usage Guidelines

To remove an AS number from a BGP AS path set, use the **no** form of this command.

Examples

The following example creates a BGP AS path set named aspath1 and adds AS numbers 65535 and 65400 as members:

```
device# configure terminal  
device(config)# route-policy  
device(config-route-policy)# bgp-defined-sets  
device(config-route-policy-bgp-set)# as-path-set aspath1
```

```
device(config-route-policy-bgp-set-as-path-set-aspath1)# member 65535 65400
device(config-route-policy-bgp-set-as-path-set-aspath1)#
```

The following example displays the routing policy configuration that is running currently on the device. In this example, the device is configured with one BGP defined set, which is an AS path set named aspath1 containing two members:

```
device# show running-config route-policy

route-policy
  bgp-defined-sets
    as-path-set aspath1
      member 65535 65400
device#
```

member (routing policy BGP community set configuration)

Use the **member** command to add communities as members of a BGP community set.

Syntax

```
member community-number [ community-number... ]  
no member community-number [ community-number... ]
```

Command Default

The specified BGP community set has no communities as members.

Parameters

```
community-number [ community-number... ]
```

List of communities. Use a space to separate two or more communities. Use any combination of the following formats:

- Standard community attributes represented as a 32 bit value from 0 to 4294967295 or a string of the form *ASN.nn* where *ASN* (service provider's AS number) and *nn* (number chosen to identify the customer site). *ASN* and *nn* are integers between 0 and 65535. For example, 65535:100.
- Communities specified as regular expression patterns.
- Well-known IETF community attribute values as follows: **internet**, **local-as**, **no-advertise**, and **no-export**.

Modes

Routing policy BGP community set configuration (config-route-policy-bgp-set-community-*name*)

Usage Guidelines

To remove a community from a BGP community set, use the **no** form of this command.

Examples

The following example creates a BGP community set named comm1 and adds communities 65535 and 65535:100 as members:

```
device# configure terminal  
device(config)# route-policy  
device(config-route-policy)# bgp-defined-sets  
device(config-route-policy-bgp-set)# community-set comm1  
device(config-route-policy-bgp-set-community-comm1)# match-set-options all  
device(config-route-policy-bgp-set-community-comm1)# member 65535 65535:100  
device(config-route-policy-bgp-set-community-comm1)#
```

The following example displays the routing policy configuration that is running currently on the device. In this example, the device is configured with one BGP defined set, which is a community set named comm1 containing two members:

```
device# show running-config route-policy

route-policy
  bgp-defined-sets
    community-set comm1
      member 65535 65535:100
      match-set-options all
device#
```

member (routing policy BGP extended community set configuration)

Use the **member** command to add communities as members of a BGP extended community set.

Syntax

```
member [ route-target: ] community-number [ [ route-target: ] community-number... ]
```

```
no member [ route-target: ] community-number [ [ route-target: ] community-number... ]
```

Command Default

The specified BGP community set has no extended communities as members.

Parameters

route-target:

(Optional) Specifies that the extended community number is a route target.

community-number

Extended community. Use a space to separate two or more extended communities. An extended community comprises either an autonomous system (AS) number and an arbitrary number (representing the service provider) or an IPv4 address and an arbitrary number (representing the service provider).

community-number

Extended community. Use a space to separate two or more extended communities. Each extended community is encoded as an 8-octet quantity, as follows:

- Type Field: 1 or 2 octets
- Value Field: Remaining octets

The 6 byte value field supports the following formats:

- 4 byte autonomous system (AS) and a 2 byte local value, where value represents a unique value to the service provider: *0-4294967295 AS:0-65535 value*, route-target:*0-4294967295 AS:0-65535 value*
- 4 byte IPv4 address and a 2 byte local value, where value represents a unique value to the service provider: *4b IPv4:0-65535 value*, route-target:*4b IPv4:0-65535 value*

Modes

Routing policy BGP extended community set configuration (config-route-policy-bgp-ext-community-*name*)

Usage Guidelines

To remove an extended community from an extended BGP community set, use the **no** form of this command.

Examples

The following example creates an extended BGP community set named `ecomml` and adds extended communities `200:1234`, `route-target:303:5000`, and `10.1.1.10:345` as members:

```
device# configure terminal
device(config)# route-policy
device(config-route-policy)# bgp-defined-sets
device(config-route-policy-bgp-set)# ext-community-set ecomml
device(config-route-policy-bgp-ext-community-ecomml)# match-set-options all
device(config-route-policy-bgp-ext-community-ecomml)# member 200:1234 route-
target:303:5000 10.1.1.10:345
device(config-route-policy-bgp-ext-community-ecomml)#
```

The following example displays the routing policy configuration that is running currently on the device. In this example, the device is configured with one BGP defined set, which is an extended community set named `ecomml` containing three members:

```
device# show running-config route-policy

route-policy
  bgp-defined-sets
    ext-community-set ecomml
      member 200:1234 route-target:303:5000 10.1.1.10:345
      match-set-options all
device#
```


member (BFD configuration)

Use the **member** command to activate the Administrative Down state or configure the default profile on all sessions under a specific interface that is configured for bidirectional forwarding detection (BFD).

Syntax

```
member { ethernet interface-name | port-channel port-channel-range | ve
interface-name | loopback interface-name | } { shutdown | profile
profile-name }

no member { ethernet interface-name | port-channel port-channel-range |
ve interface-name | loopback interface-name | } { shutdown | profile
profile-name }
```

Command Default

All sessions under all interfaces that are configured for BFD are active.

Parameters

ethernet *interface-name*

Ethernet interface name. Use the *slot/port* format or the *slot/port:breakout* format to specify an interface or one or more comma-separated ranges of interfaces respectively. For example, 0/1 or 1/1-3,5,2/7-9,10:1-4 respectively.

port-channel *port-channel-range*

Port channels. Use an integer or one or more comma-separated ranges of integers from 1 to 255. For example, 1 or 1-3,5,7-9.

ve *interface-name*

Virtual Ethernet (VE) interface name. Use an integer or one or more comma-separated ranges of integers from 1 to 8192. For example, 2 or 2-3,5,7-9.

loopback *interface-name*

Loopback interface name. Use an integer or one or more comma-separated ranges of integers from 1 to 255. For example, 3 or 3-4,5,7-9.

shutdown

Activates the Administrative Down state on all sessions under the specified interface.

profile *profile-name*

Configures the default profile on all sessions under the specified interface.

Modes

BFD configuration (config-bfd)

Usage Guidelines

Specifying a default profile can help reduce explicit profile configuration overhead when multiple routes use the same nexthop.

Use the **no** form of this command to deactivate the Administrative Down state or the default profile on all sessions under a specific interface that is configured for BFD. This is used in the context of static route bfd.

Examples

The following example activates the Administrative Down state on all BFD sessions under a Virtual Ethernet (VE) interface named 100:

```
device# configure terminal
device(config)# bfd
(config-bfd)# member ve 100 shutdown
device(config-bfd)#
```

The following example configures a default profile named profile1 on all BFD sessions under Ethernet interface 0/1:

```
device# configure terminal
device(config)# bfd
(config-bfd)# member ethernet 0/1 profile profile1
device(config-bfd)#
```

The following example displays the BFD configuration that is running currently on the device. In this example, Ethernet interface 0/1 and VE interface 100 are configured for BFD but are disabled. Also, Ethernet interface 0/1 is configured with a default profile named profile1:

```
device# show running-config bfd

bfd
  profile mlag
    interval 300 min-rx 300 multiplier 3
  !
  profile default
    interval 300 min-rx 300 multiplier 3
  !
  profile profile1
    interval 500 min-rx 500 multiplier 5
  !
  member ethernet 0/1 profile profile1
  member ethernet 0/1 shutdown
  member ve 100 shutdown
!
device#
```

In the following example, ethernet 0/1 hosts IP 2.2.2.1, which becomes source interface for BFD session. User would configure interface profile on 'ethernet 0/1' that becomes default profile for this BFD session.

```
static-route 10.1.0.0/24 2.2.2.2 enable-bfd

static-route 11.1.0.0/24 2.2.2.2 enable-bfd
```

member (single tagged)

Use the **member (single tagged)** command to create a single tagged subinterface and assign the specified bridge domain membership.

Syntax

```
member [ ethernet | port-channel ] interface-name vlan vlan-range  
no member [ ethernet | port-channel ] interface-name vlan vlan-range
```

Command Default

The current bridge domain has no subinterfaces as members.

Parameters

member

Specifies the interface type.

ethernet

(Optional) Specifies the Ethernet interface.

port-channel

(Optional) Specifies the port-channel.

interface-name

Specifies the name of the interface.

vlan *vlan-range*

Specifies the VLAN range. Valid range is 1-4094.

Modes

Bridge domain configuration (config-bd-*name*)

Usage Guidelines

You can use this command to add the physical ports or port channels as single-tagged members of the default mode bridge domain.

To remove the bridge domain membership of an interface (but not the subinterface), use the no form of this command.

Examples

The following example creates a single tagged subinterface and assigns the port-channel VLAN bridge domain membership:

```
device(config)# bridge-domain 200  
device(config-bd-200)# member ethernet 0/2 vlan 100
```

```
device# show running-config bridge-domain 200  
  
bridge-domain 200  
    member ethernet 0/2 vlan 100  
device(config)#
```

member (tagged)

Use the **member (tagged)** command to create a tagged subinterface and assign the specified bridge domain membership.

Syntax

member [**ethernet** | **port-channel**] *interface-name*

no member [**ethernet** | **port-channel**] *interface-name*

Command Default

The current bridge domain has no subinterfaces as members.

Parameters

member

Specifies the interface type.

ethernet

(Optional) Specifies the Ethernet interface.

port-channel

(Optional) Specifies the port-channel.

interface-name

Specifies the name of the interface.

Modes

Bridge domain VLAN configuration (config-vlan-bd-*name*)

Usage Guidelines

You can use this command to add the physical ports or port channels as tagged members of the default mode bridge domain.

To remove the bridge domain membership of an interface (but not the subinterface), use the no form of this command.

Examples

The following example creates a tagged subinterface and assigns the specified bridge domain membership:

```
device# configure terminal
device(config)# bridge-domain 200 mode vlan
device(config-vlan-bd-200)# vlan-id 200
device(config-vlan-bd-200)# member ethernet 0/2
device(config-vlan-bd-200)# member port-channel 1
device(config-vlan-bd-200)#
```

```
device# show running-config bridge-domain 200

bridge-domain 200 mode vlan
    vlan-id 200
    member ethernet 0/2
    member port-channel 1
device#
```

member (untagged)

Use the **member (untagged)** command to create an untagged subinterface and assign the specified bridge domain untagged membership.

Syntax

```
member [ ethernet | port-channel ] interface-name untagged  
no member [ ethernet | port-channel ] interface-name untagged
```

Command Default

The current bridge domain has no subinterfaces as members.

Parameters

member

Specifies the interface type.

ethernet

(Optional) Specifies the Ethernet interface.

port-channel

(Optional) Specifies the port-channel.

interface name

Specifies the name of the interface.

untagged

Specifies untagged membership.

Modes

Bridge domain configuration (config-bd-*name*)

Bridge domain VLAN configuration (config-vlan-bd-*name*)

Usage Guidelines

The traffic untagged or tagged with the VLAN is processed.

- In VLAN mode, the subinterface is created as untagged subinterface with the VLAN ID of the bridge domain.
- In bridge domain mode, a subinterface is created as untagged-strict.

To remove the bridge domain membership of the specified interface (but not the subinterface), use the no form of this command.

Examples

The following example creates an untagged subinterface and assigns the specified bridge domain membership.

```
device(config)# bridge-domain 200 mode vlan
device(config-vlan-bd-200)# member ethernet 0/2 untagged

device# show running-config

bridge-domain 200 mode vlan
    vlan-id 200
    member ethernet 0/2 untagged
device(config)#
```


member vni-domain

Use the **member vni-domain** command to add a Virtual Network Identifier (VNI) domain as a member of a specific Network Virtualization Overlay (NVO).

Syntax

```
member vni-domain vni-domain-name  
no member vni-domain vni-domain-name
```

Command Default

No VNI domains are members of the specified NVO.

Parameters

vni-domain-name

Name of the VNI domain to add. You can use any combination of 1 to 64 uppercase or lowercase alphabetic characters, numerals, underscores, hyphens, or periods. You must use an alphabetic character or an underscore as the first character.

Modes

NVO configuration (config-nvo-*name*)

Usage Guidelines

Use the **no** form of this command to remove the membership of a VNI domain from a specific NVO.

Examples

The following example adds a VNI domain named DG1 as a member to an NVO named base:

```
device# configure terminal  
device(config)# overlay-networks  
device(config-overlay)# nvo base  
device(config-nvo-base)# member vni-domain DG1  
device(config-nvo-base)#
```

The following example shows the NVO instances that are running currently on the device:

```
device# show running-config overlay-networks  
  
overlay-networks  
  default-nve base  
  default-vni-offset 0  
  nvo base
```

```
    member nve base
    member vni-domain DG1
    !
    nve base
        source-interface loopback 1
    !
    vni-domain base
    vni-domain DG1
        auto-offset 500000
    !
    !
device#
```

member (VRF configuration)

Use the **member** command to add an interface as a member of a virtual routing and forwarding (VRF) instance.

Syntax

```
member { ethernet interface-name | port-channel port-channel-range | ve interface-name | loopback interface-name | management interface-name | internal interface-name } vlan vlan-range
```

```
no member { ethernet interface-name | port-channel port-channel-range | ve interface-name | loopback interface-name | management interface-name | internal interface-name } vlan vlan-range
```

Command Default

The current VRF instance has no interfaces as members.

Parameters

ethernet *interface-name*

Ethernet interface name. Use the *slot/port* format or the *slot/port:breakout* format to specify an interface or one or more comma-separated ranges of interfaces respectively. For example, 0/1 or 1/1-3,5,2/7-9,10:1-4 respectively.

port-channel *port-channel-range*

Port channels. Use an integer or one or more comma-separated ranges of integers from 1 to 255. For example, 1 or 1-3,5,7-9.

ve *interface-name*

Virtual Ethernet (VE) interface name. Use an integer or one or more comma-separated ranges of integers from 1 to 8192. For example, 2 or 2-3,5,7-9.

loopback *interface-name*

Loopback interface name. Use an integer or one or more comma-separated ranges of integers from 1 to 255. For example, 3 or 3-4,5,7-9.

management *interface-name*

Management interface name. The only supported value is 0.

internal *interface-name*

Internal interface name.

vlan *vlan-range*

VLAN range. Use an integer or one or more comma-separated ranges of integers from 1 to 4094. For example, 4 or 4-5,6,8-10.

Modes

VRF configuration (config-vrf-*vrf-name*)

Usage Guidelines

To remove the VRF membership of an interface, use the no form of this command.

Examples

The following example adds three interfaces as members of a VRF instance named default-vrf:

```
device# configure terminal
device(config)# vrf default-vrf
device(config-vrf-default-vrf)# member ethernet 0/2 vlan 100
device(config-vrf-default-vrf)# member ve 100 vlan 100
device(config-vrf-default-vrf)# member port-channel 1 vlan 100
```

The following example displays the configuration of a VRF instance named default-vrf that is running currently on the device:

```
device# show running-config vrf default-vrf

vrf default-vrf
  member ethernet 0/2 vlan 100
  member port-channel 1 vlan 100
  member ve 100 vlan 100
!
device#
```

mirror (global configuration)

Use the **mirror** command to configure a port mirror and enter Mirror configuration (config-mirror) mode.

Syntax

```
mirror mirror-name  
no mirror mirror-name
```

Command Default

No mirrors are configured on the device.

Parameters

mirror-name

Name of the mirror to configure. This name must start with an alphabetic character or an underscore followed by an arbitrary sequence of alphabetic or numeric characters, underscores, hyphens, or dots. The name cannot exceed 64 characters.

Modes

Global configuration (config)

Usage Guidelines

Port Mirroring, also called Switched Port Analyzer (SPAN), is a feature that copies selected packets that pass through a port (or VLAN) and sends them to another port for analysis. Port Mirroring uses SPAN sessions to help monitor or troubleshoot network traffic passing through interfaces or VLANs.

Use the **no** form of this command to remove the port mirror.

Examples

The following example configures a port mirror named mirror-interface-1 on the device and enters Mirror configuration (config-mirror) mode:

```
device# configure terminal  
device(config)# mirror mirror-interface-1  
device(config-mirror)#
```

The following example shows the port mirror configuration that is running currently on the device. In this example, a single port mirror (in this case, mirror-interface-1) exists on the device:

```
device# show running-config mirror
```

```
mirror mirror-interface-1
  destination interface ethernet 0/1
device#
```

mirror (interface configuration)

Use the **mirror** command to attach a port mirror session to an interface.

Syntax

```
mirror mirror-session-name { in | out }  
no mirror mirror-session-name { in | out }
```

Command Default

No port mirror session is attached to the current interface.

Parameters

mirror-session-name

Name of the port mirror session to attach to the current interface. This name must start with an alphabetic character or an underscore followed by an arbitrary sequence of alphabetic or numeric characters, underscores, hyphens, or dots. The name cannot exceed 64 characters.

{ **in** | **out** }

Attaches a port mirror session in the ingress or egress direction respectively.

Modes

Ethernet interface configuration (config-if-eth-*name*)

Port Channel interface configuration (config-if-po-*name*)

Usage Guidelines

Port Mirroring, also called Switched Port Analyzer (SPAN), is a feature that copies selected packets that pass through a port (or VLAN) and sends them to another port for analysis. Port Mirroring uses SPAN sessions to help monitor or troubleshoot network traffic passing through interfaces or VLANs.

Only one destination interface is allowed per mirror session.

Use the **no** form of this command to remove a port mirror session from the interface.

Examples

The following example attaches a port mirror session named mirror1 to Ethernet interface 0/1 in the ingress and egress directions:

```
device# configure terminal  
device(config)# interface ethernet 0/1  
device(config-if-eth-0/1)# mirror mirror1 in
```

```
device(config-if-eth-0/1)# mirror mirror1 out
device(config-if-eth-0/1)#
```

The following example shows the interface configuration that is running currently on the device. In this example, a port mirror session named mirror1 is attached in the ingress and egress directions to port channel interface 1 as well as Ethernet interface 0/1:

```
device# show running-config interface

interface port-channel 1
  no shutdown
  mirror mirror1 in
  mirror mirror1 out
!
interface ethernet 0/1
  no shutdown
  mirror mirror1 in
  mirror mirror1 out
device#
```


mlag

Use the **mlag** command to enter MLAG configuration (config-mlag) mode.

Syntax

mlag

Command Default

Multichassis link aggregation (MLAG) is not configured on the device.

Parameters

This command has no keywords or variables.

Modes

Global configuration (config)

Usage Guidelines

Use this command to enter the mode for configuring MLAG settings. These settings include the bringup delay, interfaces, keepalive, MAC address, peers, role (primary, backup, or automatic), and uplink tracking interfaces.

Examples

The following example enters MLAG configuration (config-mlag) mode:

```
device# configure terminal
device(config)# mlag
device(config-mlag)#
```

mode disable-unsolicited-ra

Use the **mode disable-unsolicited-ra** command to disable the transmission of unsolicited messages for IPv6 router advertisement on an interface.

Syntax

```
mode disable-unsolicited-ra  
no mode
```

Command Default

Transmission of unsolicited router advertisement messages is enabled.

Parameters

This command has no keywords or variables.

Modes

IPv6 router advertisement configuration (config-if-*name*-ipv6-router-advertisement)

Usage Guidelines

Router advertisements are ICMPv6 messages sent by an IPv6 enabled router that are used by hosts to detect routers on an attached link. This functionality is similar to the IPv4 ICMP Router Discovery protocol.

This mode controls which set of behaviors the local system should perform to support IPv6 router advertisements. When you use this command, the system does not transmit unsolicited router advertisement messages on this interface (but continues to transmit responses to router solicitation messages).

To re-enable transmission of unsolicited router advertisement messages, use the **no** form of this command.

Examples

The following example configures the system to disable transmission of unsolicited router advertisement messages on Ethernet interface 0/1:

```
device# configure terminal  
device(config)# interface ethernet 0/1  
device(config-if-eth-0/1)# ipv6-router-advertisement  
device(config-if-eth-0/1-ipv6-router-advertisement)# mode disable-unsolicited-ra  
device(config-if-eth-0/1-ipv6-router-advertisement)#
```

The following example displays the configuration of Ethernet interface 0/1 that is running currently on the device. In this example, the system does not transmit unsolicited router advertisement messages on the interface:

```
device# show running-config interface ethernet 0/1

!
interface ethernet 0/1
  ipv6 address 1111::10/64
  ipv6 neighbor 1111::1 00:04:96:eb:c4:51
  ipv6-router-advertisement
    hop-limit 133
    lifetime 200
    managed-config-flag
    mode disable-unsolicited-ra
    interval 100 min 50
    enable
    prefix 100::1/64
    prefix 100::1/63
      valid-lifetime 200
  !
  no shutdown
!
device#
```

mtu (Extreme ONE OS Switching interface configuration)

Use the **mtu** command to set the size of the maximum transmission unit (MTU) for all Layer 2 (data link layer) transactions on a specific port.

Syntax

mtu *bytes*
no mtu

Command Default

The MTU size on the port is 9216 bytes.

Parameters

bytes
Size in bytes. Use an integer from 1500 to 9216.

Modes

Ethernet interface configuration (config-if-eth-*name*)

Port Channel interface configuration (config-if-po-*name*)

Usage Guidelines

The MTU is the size of the largest protocol data unit (PDU) that can be communicated in a single data link layer transaction. The MTU relates to (but is not identical to) the largest packet that can be transported on the network layer (such as an IPv4 packet). This setting applies to all Ethernet and port channel frames.

This setting determines the MTU size that is allowed on the specified port. It overrides the global (device-wide) setting, which is either 9216 by default or is set by the **mtu** command in configuration mode (config).

This release does not support the *mtu* command on the out-of-band (OOB) management port.

Use the **no** form of this command to reset the MTU size on the port to the default value.

Examples

The following example configures Ethernet interface 0/1. In this example, the MTU is set to 5000 bytes on the interface:

```
device# configure terminal
device(config)# interface ethernet 0/1
```

```
device(config-if-eth-0/1)# mtu 5000
device(config-if-eth-0/1)#
```

The following example sets the MTU size on LAG interface 1 at 6000 bytes:

```
device# configure terminal
device(config)# interface port-channel 1
device(config-if-po-1)# mtu 6000
device(config-if-po-1)#
```

The following example displays details about Ethernet interface 0/32. In this example, the MTU is set to 9216 bytes on the interface:

```
device# show interface ethernet 0/32

ethernet 0/32 Admin state UP, Oper state DOWN
Interface index is 16778240 (0x1000400)
MTU 9216 bytes
Hardware is Ethernet  mac address 50:0a:9c:97:30:3c
Current Speed  400G
Description: Ethernet 0/32
FEC:
    Mode: disabled
    Uncorrected Blocks: 18
    Corrected Blocks: 285179
Link Fault Signaling: ON
Link Fault Status: No Fault
Auto Negotiation: ON
Carrier Transitions: 8
    LastClear: 0s
Input:
    Total pkts: 88
    Broadcast pkts: 0
    Discard pkts: 0
    Errors pkts: 0
    FCS Errors: 0
    MCast pkts: 88
    Octets: 15312
    UCast pkts: 0
    Runt pkts: 0
    CRC Errors: 0
Input Distribution:
    64 byte pkts: 0
    65-127 byte pkts: 0
    128-255 byte pkts: 88
    256-511 byte pkts: 0
    512-1023 byte pkts: 0
    1024-1518 byte pkts: 0
    Jumbo pkts: 0
Out:
    Total pkts: 94
    Broadcast pkts: 0
    Discard pkts: 0
    Errors pkts: 0
    MCast pkts: 94
    Octets: 15134
    UCast pkts: 0
Rate Info:
    Input: 0.000000 Mbits/sec, 0 pkts/sec 0.00% of line-rate
    Output: 0.000000 Mbits/sec, 0 pkts/sec 0.00% of line-rate
device#
```

The following example displays the configuration that is running currently on the device. In this example, the global MTU size is overridden for Ethernet interface 0/1 and LAG interface 1):

```
device# show running-config

interface ethernet 0/1
    mtu 5000
    no shutdown
interface port-channel 1
    mtu 6000
    no shutdown
!
device#
```

The **show running-config** command displays the global MTU size when it is not the default value. If you change the global MTU size but do *not* change the MTU size of a port, the **show running-config** command does not display the global MTU value at the port level.

mtu (Extreme ONE OS Switching global configuration)

Use the **mtu** command to set the size of the maximum transmission unit (MTU) for all Layer 2 (data link layer) transactions on the device.

Syntax

mtu *bytes*

no mtu

Command Default

The MTU size is 9216 bytes.

Parameters

bytes

Size in bytes. Use an integer from 1500 to 9216.

Modes

Global configuration (config)

Usage Guidelines

The MTU is the size of the largest protocol data unit (PDU) that can be communicated in a single data link layer transaction. The MTU relates to (but is not identical to) the largest packet that can be transported on the network layer (such as an IPv4 packet). This setting applies to all Ethernet and port channel ports.

This is a global (device-wide) setting, which means that it applies to all Layer 2 ports (that are not otherwise configured with their own MTU sizes). You can override this setting on a specific Layer 2 port by using the **mtu** command in interface configuration mode on that port.

Use the **no** form of this command in configuration mode (config) to reset the MTU size to the default value on all Layer 2 ports (except those with manually set MTU sizes).

Examples

The following example sets the MTU size globally at 5000 bytes:

```
device# configure terminal
device(config)# mtu 5000
device(config)#
```

The following example displays the configuration that is running currently on the device. In this example, the MTU size is set globally at 5000 bytes:

```
device# show running-config mtu

mtu 5000
!
device#
```


nd-proxy

Use the **nd-proxy** command to enter Bridge domain VLAN neighbor discovery proxy configuration (config-vlan-bd-*name*-nd-proxy) mode.

Syntax

nd-proxy

Parameters

This command has no keywords or variables.

Modes

Bridge domain VLAN configuration (config-vlan-bd-*name*)

Usage Guidelines

This command enters the mode for Neighbor Discovery (ND) proxy configuration in a bridge domain. This configuration includes agetime, snooping, and ND suppression.

Examples

The following example enters Bridge domain VLAN neighbor discovery proxy configuration (config-vlan-bd-*name*-nd-proxy) mode:

```
device# configure terminal
device(config)# bridge-domain 100 mode vlan
device(config-vlan-bd-100)# nd-proxy
device(config-vlan-bd-100-nd-proxy)#
```

The following example displays the configuration for bridge domain 100 that is running currently on the device. In this example, ND snooping is enabled on the neighbor discovery proxy in the bridge domain:

```
device# show running-config bridge-domain 100

bridge-domain 100 mode vlan
    nd-proxy
    nd-snooping
    !
device#
```

nd-snooping

Use the **nd-snooping** command to enable neighbor discovery protocol (NDP) snooping on the neighbor discovery proxy in a bridge domain.

Syntax

```
nd-snooping  
no nd-snooping
```

Parameters

This command has no keywords or variables.

Modes

Bridge domain VLAN neighbor discovery proxy configuration (config-vlan-bd-*name*-nd-proxy)

Usage Guidelines

To disable NDP snooping on the neighbor discovery proxy in a bridge domain, use the **no** form of this command.

Examples

The following example enables NDP snooping on the neighbor discovery proxy in a bridge domain with 100 as its ID:

```
device# configure terminal  
device(config)# bridge-domain 100 mode vlan  
device(config-vlan-bd-100)# nd-proxy  
device(config-vlan-bd-100-nd-proxy)# nd-snooping  
device(config-vlan-bd-100-nd-proxy)#
```

The following example displays the configuration for bridge domain 100 that is running currently on the device. In this example, NDP snooping is enabled on the neighbor discovery proxy in the bridge domain:

```
device# show running-config bridge-domain 100  
  
bridge-domain 100 mode vlan  
    nd-proxy  
    nd-snooping  
    !  
device#
```

neighbor (BGP peer group configuration)

Use the **neighbor** command to specify an IPv4 or IPv6 address of a neighbor router in a BGP peer group within a Virtual Routing and Forwarding (VRF) instance.

Syntax

```
neighbor { ipv4-address | ipv6-address }  
no neighbor { ipv4-address | ipv6-address }
```

Command Default

No neighbor is configured for the specified BGP peer group.

Parameters

ipv4-address

IPv4 address. Use the *a.b.c.d* format.

ipv6-address

IPv6 address. Use the *x:x::x:x* format.

Modes

VRF BGP peer group configuration (config-vrf-bgp-pg)

Usage Guidelines

This command specifies a neighbor router in a BGP peer group.

Use the **no** form of this command to remove a neighbor from the current BGP peer group.

Examples

The following example configures two BGP peer groups (group1 and group2) under a VRF instance named violet. In this example, group1 is assigned 200 as its remote autonomous system (AS) number, the IPv4 address range 10.1.1.0/24 as a listen range, and 1.1.1.0 as a neighbor router. Also, group2 is assigned 300 as its remote AS number, the IPv6 address range 2001::1/64 as a listen range, and 1.1.1.1 as a neighbor router:

```
device# configure terminal  
device(config)# vrf violet  
device(config-vrf-violet)# router bgp  
device(config-vrf-bgp)# peer-group group1  
device(config-vrf-bgp-pg)# remote-as 200  
device(config-vrf-bgp-pg)# listen-range 10.1.1.0/24  
device(config-vrf-bgp-pg)# neighbor 1.1.1.0  
device(config-vrf-bgp-pg)# exit  
device(config-vrf-bgp)# peer-group group2
```

```
device(config-vrf-bgp-pg)# remote-as 300
device(config-vrf-bgp-pg)# listen-range 2001::1/64
device(config-vrf-bgp-pg)# neighbor 1.1.1.1
device(config-vrf-bgp-pg)#
```

The following example displays the configuration of a VRF instance named violet that is running currently on the device. In this example, both group1 and group2 have a neighbor router configured:

```
device# show running-config vrf violet

vrf violet
!
  router bgp
    peer-group group1
      listen-range 10.1.1.0/24
      neighbor 1.1.1.0
    peer-group group2
      listen-range 2001::1/64
      neighbor 1.1.1.1
!
device#
```

network (BGP address family configuration)

Use the **network** command to advertise a network to the routing table of a specific BGP address family within a routing process in a VRF instance.

Syntax

```
network { ipv4-addr/prefix-length | ipv6-addr/prefix-length }  
no network { ipv4-addr/prefix-length | ipv6-addr/prefix-length }
```

Command Default

No networks are advertised to the routing table for the specified BGP address family.

Parameters

ipv4-addr/prefix-length

Range of IPv4 addresses to be advertised. Use the *a.b.c.d* format for the IPv4 address, and use an integer from 0 to 32 for the prefix. You must include the slash (/).

ipv6-addr/prefix-length

Range of IPv6 addresses to be advertised. Use the *x:x::x:x* format for *dest-ipv6-addr*, and use an integer from 0 to 128 for the prefix. You must include the slash (/).

Modes

VRF BGP address family configuration (*config-vrf-bgp-address-family-type*)

Usage Guidelines

This command broadcasts routing information to neighbors in the specified address family. This command identifies which networks to advertise from your local networks. If BGP finds a network in the local routing table that matches the network statement (and the mask) then BGP advertises it.

Use the **no** form of this command to remove the entries from the local routing table and restore the default behavior for the specified address family.

Examples

The following example configures a routing process under a VRF instance named violet. This example creates two address families and advertises a network in each one:

```
device# configure terminal  
device(config)# vrf violet  
device(config-vrf-violet)# router bgp  
device(config-vrf-bgp)# address-family ipv4 unicast
```

```
device(config-vrf-bgp-ipv4u)# network 11.1.1.1/32
device(config-vrf-bgp-ipv4u)# exit
device(config-vrf-bgp)# address-family ipv6 unicast
device(config-vrf-bgp-ipv6u)# network 10:1:1::2/32
device(config-vrf-bgp-ipv6u)#
```

The following example displays the configuration of a VRF instance named violet that is running currently on the device. In this example, the instance contains a routing process with two address families and an advertised network in each one:

```
device# show running-config vrf violet

vrf violet
  router bgp
    address-family ipv4 unicast
      network 11.1.1.1/32
    !
    address-family ipv6 unicast
      network 10:1:1::2/32
      activate
    !
  !
device#
```

network (IAH VM configuration)

Use the **network** command to specify the network name for a virtual machine (VM) and enter IAH VM network configuration (config-iah-vm-network-*name*) mode.

Syntax

network *network-name*

no network *network-name*

Command Default

The specified VM has no network configured.

Parameters

network-name

Network name. Use any combination of 1 to 32 alphanumeric characters, space, underscores, hyphens, or periods.

Modes

IAH VM configuration (config-iah-vm-*name*)

Usage Guidelines

Use this command to enter the mode for configuring network settings for the VM such as the network interface and its interface type and MAC address.

Internal port 0 or 1 can be used either for Redundant Management or for IAH TPVM functionality (not simultaneously).

Use the **no** form of this command to remove the network configuration from a VM. Note that configuration is not allowed when the VM is in the deployed state.

To display details about a specific VM, see [show iah vm](#) on page 1006.

Examples

The following example configures a network named ins2 for a VM named Ubuntu1 on the device and enters IAH VM network configuration (config-iah-vm-network-*name*) mode:

```
device# configure terminal
device(config)# iah
device(config-iah)# vm Ubuntu1
device(config-iah-vm-Ubuntu1)# network ins2
device(config-iah-vm-network-ins2)#
```

The following example displays the IAH configuration that is running currently on the device. In this example, a VM named Ubuntu2 uses a network named ins1:

```
device# show running-config iah

iah
vm Ubuntu2
  description Ubuntu demo vm
  auto-boot true
  cpus 4
  ram 4096
  network ins1
    interface internal 1
  disk sdb
    size 5000
    format qcow2
    bus sata
    file disk://iah/disk1.qcow2
  !
!
device#
```


next-hop-enable-default (BGP address family configuration)

Use the **next-hop-enable-default** command to enable BGP next-hop address resolution via the default route in a specific address family forwarding table within a VRF instance.

Syntax

```
next-hop-enable-default  
no next-hop-enable-default
```

Command Default

BGP next-hop address resolution in the specified address family and subaddress family via the default route is disabled.

Parameters

This command has no keywords or variables.

Modes

VRF BGP address family configuration (*config-vrf-bgp-address-family-type*)

Usage Guidelines

Use the **no** form of this command to disable this feature.

Examples

The following example configures a Virtual Routing and Forwarding (VRF) instance named default-vrf. In this example, the VRF enables BGP next-hop address resolution via the default route within the IPv4 address family and unicast subaddress family:

```
device# configure terminal  
device(config)# vrf default-vrf  
device(config-vrf-default-vrf)# router bgp  
device(config-vrf-bgp)# address-family ipv4 unicast  
device(config-vrf-bgp-ipv4u)# next-hop-enable-default  
device(config-vrf-bgp-ipv4u)#
```

The following example displays the configuration of a VRF instance named default-vrf that is running currently on the device. In this example, the VRF enables BGP next-hop address resolution via the default route within the IPv4 address family and unicast subaddress family:

```
device# show running-config vrf default-vrf  
  
vrf default-vrf  
!
```

```

router bgp
  address-family ipv4 unicast
    next-hop-enable-default
    next-hop-recursion
    use-multiple-paths ibgp maximum-paths 2
    use-multiple-paths ebgp maximum-paths 2 allow-multiple-as
!
device(config-vrf-bgp)#

```

The following example displays a summary of the BGP protocol running state, current route received/advertised count status, and neighbor details for the IPv4 address family and unicast subaddress family within a VRF named default-vrf. In this example, Default-Route: Enabled indicates that BGP next-hop address resolution via the default route is enabled:

```

device# show bgp vrf default-vrf summary ipv4-unicast

VRF Name : default-vrf
Local AS: 1, Router ID: 1.1.1.1
Confederation Identifier: not configured
Confederation Peers: not configured
Graceful Restart: Disabled

IPv4 Unicast Summary Information
-----
Graceful Restart: Disabled
Prefix Independent Convergence: Enabled
Nexthop address resolution - Recursion: Enabled, Default-Route: Enabled
Number of Routes :65536 (Paths: 393216)
Number of Peer(s) Configured : 7, Up : 7
D: Dynamically created based on a listen range command, Dynamically created neighbors: 0
C: Route Reflector Client

Peer IP      Peer AS  Local IP  State      Health    UpTime    RIB-IN  RIB-OUT  Flags
Peer Group
=====
12.2.1.      2        12.2.1.1  Established -        2h43m4s   65536   0         -    to-
dut2-ipv4
13.3.1.3     3        13.3.1.1  Established -        2h43m24s  65536   65536    -    to-
dut3-ipv4
31.1.1.30    301      31.1.1.1  Established -        59m55s    0       65536    -    to-
stc-ipv4
12.1.1.2     2        12.1.1.   Established -        2h43m10s  65536   0         -    to-
dut2-ipv4
12.3.1.2     2        12.3.1.1  Established -        2h43m9s   65536   0         -    to-
dut2-ipv4
13.1.1.3     3        13.1.1.1  Established -        2h43m23s  65536   65536    -    to-
dut3-ipv4
13.2.1.3     3        13.2.1.1  Established -        2h43m13s  65536   65536    -    to-
dut3-ipv4
device#

```

next-hop-in (routing policy statement bgp conditions configuration)

Use the **next-hop-in** command to add a BGP condition to a routing policy statement for matching against IPv4 or IPv6 next hop attribute tokens.

Syntax

```
next-hop-in ip-address [ ip-address... ]  
no next-hop-in [ ip-address [ ip-address... ] ]
```

Command Default

The specified routing policy statement contains no BGP condition for matching against IPv4 or IPv6 next hop attribute tokens.

Parameters

ip-address

List of IPv4 or IPv6 addresses to add. Use a space to separate two or more addresses.

Modes

Routing policy statement BGP conditions configuration (config-route-policy-policy-name-stmt-number-conditions-bgp-conditions)

Usage Guidelines

This command configures a BGP condition that matches an IPv4 address or IPv6 address next hop attribute token in the route to the configured IP addresses in this command.

To remove IP addresses from a routing policy statement, use the **no** form of this command.

Examples

The following example creates a routing policy named map1 that contains a policy statement with sequence number 10. The policy statement contains a condition for matching against IPv4 address 192.0.2.0:

```
device# configure terminal  
device(config)# route-policy  
device(config-route-policy)# policy map1  
device(config-route-policy-policy-map1)# statement 10  
device(config-route-policy-policy-map1-stmt-10)# conditions  
device(config-route-policy-policy-map1-stmt-10-conditions)# bgp-conditions  
device(config-route-policy-policy-map1-stmt-10-conditions-bgp-conditions)# next-hop-in  
192.0.2.0  
device(config-route-policy-policy-map1-stmt-10-conditions-bgp-conditions)#
```

The following example displays the routing policy configuration that is running currently on the device. In this example, a routing policy named map1 has a policy statement with sequence number 10. The policy statement contains a condition for matching against IPv4 address 192.0.2.0:

```
device# show running-config route-policy

route-policy
  policy map1
    statement 10
      conditions
        bgp-conditions
          next-hop-in 192.0.2.0
device#
```

next-hop-recursion (BGP address family configuration)

Use the **next-hop-recursion** command to enable BGP next-hop address resolution through BGP routes in a specific address family forwarding table within a VRF instance.

Syntax

```
next-hop-recursion  
no next-hop-recursion
```

Command Default

BGP next-hop address resolution in the specified address family and subaddress family through BGP routes is disabled.

Parameters

This command has no keywords or variables.

Modes

VRF BGP address family configuration (*config-vrf-bgp-address-family-type*)

Usage Guidelines

Use the **no** form of this command to disable this feature.

Examples

The following example configures a Virtual Routing and Forwarding (VRF) instance named default-vrf. In this example, the VRF enables BGP next-hop address resolution through BGP routes within the IPv4 address family and unicast subaddress family:

```
device# configure terminal  
device(config)# vrf default-vrf  
device(config-vrf-default-vrf)# router bgp  
device(config-vrf-bgp)# address-family ipv4 unicast  
device(config-vrf-bgp-ipv4u)# next-hop-recursion  
device(config-vrf-bgp-ipv4u)#
```

The following example displays the configuration of a VRF instance named default-vrf that is running currently on the device. In this example, the VRF enables BGP next-hop address resolution through BGP routes within the IPv4 address family and unicast subaddress family:

```
device# show running-config vrf default-vrf  
  
vrf default-vrf  
!  
    router bgp  
        address-family ipv4 unicast
```

```

next-hop-enable-default
next-hop-recursion
use-multiple-paths ibgp maximum-paths 2
use-multiple-paths ebgp maximum-paths 2 allow-multiple-as
!
device(config-vrf-bgp)#

```

The following example displays a summary of the BGP protocol running state, current route received/advertised count status, and neighbor details for the IPv4 address family and unicast subaddress family within a VRF named default-vrf. In this example, Default-Route: Enabled indicates that BGP next-hop address resolution through BGP routes is enabled:

```

device# show bgp vrf default-vrf summary ipv4-unicast

VRF Name : default-vrf
Local AS: 1, Router ID: 1.1.1.1
Confederation Identifier: not configured
Confederation Peers: not configured
Graceful Restart: Disabled

IPv4 Unicast Summary Information
-----
Graceful Restart: Disabled
Prefix Independent Convergence: Enabled
Nexthop address resolution - Recursion: Enabled, Default-Route: Enabled
Number of Routes :65536 (Paths: 393216)
Number of Peer(s) Configured : 7, Up : 7
D: Dynamically created based on a listen range command, Dynamically created neighbors: 0
C: Route Reflector Client

Peer IP      Peer AS  Local IP  State      Health    UpTime    RIB-IN  RIB-OUT  Flags
Peer Group
=====
12.2.1.      2        12.2.1.1  Established -        2h43m4s   65536   0        -        to-
dut2-ipv4
13.3.1.3     3        13.3.1.1  Established -        2h43m24s  65536   65536    -        to-
dut3-ipv4
31.1.1.30    301      31.1.1.1  Established -        59m55s   0       65536    -        to-
stc-ipv4
12.1.1.2     2        12.1.1.   Established -        2h43m10s  65536   0        -        to-
dut2-ipv4
12.3.1.2     2        12.3.1.1  Established -        2h43m9s   65536   0        -        to-
dut2-ipv4
13.1.1.3     3        13.1.1.1  Established -        2h43m23s  65536   65536    -        to-
dut3-ipv4
13.2.1.3     3        13.2.1.1  Established -        2h43m13s  65536   65536    -        to-
dut3-ipv4
device#

```

nexthop-self (BGP peer group address family configuration)

Use the **nexthop-self** command to disable the next hop calculation for neighbors for an address family in a peer group within a routing process in a VRF instance.

Syntax

```
nexthop-self  
no nexthop-self
```

Command Default

The address family in the specified peer group does not have **nexthop-self** enabled.

Parameters

This command has no keywords or variables.

Modes

VRF BGP peer group address family configuration (*config-vrf-bgp-pg-address-family-type*)

Usage Guidelines

This command causes the device to advertise itself as the next hop in updates that are sent to neighbors.

This command applies only to IPv4 unicast or IPv6 unicast address families.

Use the **no** form of this command to restore the default functionality.

Examples

The following example configures a routing process under a VRF instance named violet. This example specifies an IPv4 unicast address family in a peer group named group1 and also disables the next hop calculation for neighbors for the address family:

```
device# configure terminal  
device(config)# vrf violet  
device(config-vrf-violet)# router bgp  
device(config-vrf-bgp)# peer-group group1  
device(config-vrf-bgp-pg)# address-family ipv4 unicast  
device(config-vrf-bgp-pg-ipv4u)# nexthop-self  
device#
```

The following example displays the configuration of a VRF instance named violet that is running currently on the device. In this example, a peer group named group1 is

configured. An IPv4 unicast address family that disables the next hop calculation for neighbors is specified in this example:

```
device# show running-config vrf violet

vrf violet
  router bgp
    address-family ipv4 unicast
      send-default-route
    !
    peer-group group1
      remote-as 200
      address-family ipv4 unicast
        nexthop-self
    !
  !
!
device#
```


nexthop-unchanged (BGP peer group address family configuration)

Use the **nexthop-unchanged** command to keep the route next-hop attribute unchanged for an address family in a specific peer group within a routing process in a VRF instance.

Syntax

nexthop-unchanged
no nexthop-unchanged

Command Default

The address family in the specified peer group does not have **nexthop-unchanged** enabled.

Parameters

This command has no keywords or variables.

Modes

VRF BGP peer group address family configuration (config-vrf-bgp-pg-*address-family-type*)

Usage Guidelines

BGP speakers change the next hop while sending the updates to BGP neighbors. Use this command to override this behavior. With this command, the next-hop attribute remains unchanged while updates are sent to BGP peers, and the BGP speaker must retain the next-hop address in the BGP updates received from neighbors.

This command applies only to IPv4 unicast or IPv6 unicast address families.

Use the **no** form of this command to restore the default functionality.

Examples

The following example configures a routing process under a VRF instance named violet. This example specifies an IPv4 unicast address family in a peer group named group1 and also keeps the route next-hop attribute unchanged for the address family:

```
device# configure terminal
device(config)# vrf violet
device(config-vrf-violet)# router bgp
device(config-vrf-bgp)# peer-group group1
device(config-vrf-bgp-pg)# address-family ipv4 unicast
device(config-vrf-bgp-pg-ipv4u)# nexthop-unchanged
device#
```

The following example displays the configuration of a VRF instance named violet that is running currently on the device. In this example, a peer group named group1 is configured. An IPv4 unicast address family that keeps the route next-hop attribute unchanged is specified in this example:

```
device# show running-config vrf violet

vrf violet
  router bgp
    address-family ipv4 unicast
      send-default-route
    !
    peer-group group1
      remote-as 200
      address-family ipv4 unicast
        nexthop-unchanged
      !
    !
  !
!
device#
```

ntp (IAH VM configuration)

Use the **ntp** command to configure a Network Time Protocol (NTP) server for an IAH third party virtual machine (TPVM).

Syntax

```
ntp { ipv4 ipv4-address | ipv6 ipv6-address | domain-name }  
no ntp { ipv4 ipv4-address | ipv6 ipv6-address | domain-name }
```

Command Default

The TPVM has no NTP servers configured.

Parameters

ipv4-address

NTP server IPv4 address. Use the *a.b.c.d* format.

ipv6-address

NTP server IPv6 address. Use the *x::x::x::x* format.

domain-name *pass_string*

NTP server fully qualified domain name (FQDN).

Modes

IAH VM configuration (*config-iah-vm-name*)

Usage Guidelines

This command configures an NTP server for the TPVM on the device. The TPVM is a general server that resides on Extreme ONE OS devices. The guest OS that the TPVM provides is separate from Extreme ONE OS.

You use NTP to synchronize the clocks over your network. The NTP service is a global network of public and private servers that provides precise time synchronization to devices worldwide. It works by exchanging time data packets among client and server devices.

The NTP configuration is set irrespective of out of band (OOB) connectivity and the IPv4 or IPv6 static or DHCP configuration.

A maximum of five NTP servers is allowed.

This command is allowed conditionally only for an integrated application hosting (IAH) VM named TPVM. Specifying this VM name extends the available IAH configurations to support TPVM.

Use the **no** form of this command to remove an NTP server from the TPVM. Note that configuration is not allowed when the VM is in the deployed state.

To display details about a specific VM, see [show iah vm](#) on page 1006.

Examples

The following example sets an NTP server with IPv4 address 1.1.1.1 for the IAH TPVM on the device:

```
device# configure terminal
device(config)# iah
device(config-iah)# vm TPVM
device(config-iah-vm-TPVM)# ntp 1.1.1.1
device(config-iah-vm-TPVM)#
```

The following example displays the IAH configuration that is running currently on the device. In this example, three IP addresses and a domain name are specified for the four NTP servers for the IAH TPVM on the device:

```
device# show running-config iah

iah
vm TPVM
  hostname ganga
  timezone Africa/Casablanca
  ip
    ipv4-addr 2.2.2.4
    ipv4-gw 2.2.2.1
  ntp
    server 1.1.1.1
    server 3.3.3.3
    server time.example.com
    server 2001:db8::1
  interface management ipv4 2.2.2.2/24 gw 2.2.2.1
  trusted-peer ipv4 3.3.3.3 password $6$bcdef sudo-user extreme
  !
!
device#
```

nve (EVPN instance configuration)

Use the **nve** command to associate a Network Virtualization Edge (NVE) endpoint (also called an NVE node) with a specific EVPN instance.

Syntax

nve *nve-name*

no nve

Command Default

No NVE endpoints are associated with the specified EVPN instance.

Parameters

nve-name

Name of the NVE endpoint to associate with the specified EVPN instance. You can use any combination of 1 to 64 uppercase or lowercase alphabetic characters, numerals, underscores, hyphens, or periods. You must use an alphabetic character or an underscore as the first character.

Modes

Ethernet VPN instance configuration (*config-evpn-name*)

Usage Guidelines

You can associate the same NVE endpoint with multiple EVPN instances.

Use the **no** form of this command to remove the association of an NVE endpoint from the specified EVPN instance.

Examples

The following example associates three NVE endpoints named *nve1*, *nve2*, and *nve3*. In this example, EVPN instances named *tenant_2* and *tenant_3* share *nve2*:

```
device# configure terminal
device(config)# vrf default-vrf
device(config-default-vrf)# evpn-instance tenant_1
device(config-evpn-tenant_1)# nve nve1
device(config-evpn-tenant_1)# exit
device(config-default-vrf)# evpn-instance tenant_2
device(config-evpn-tenant_2)# nve nve2
device(config-evpn-tenant_2)# exit
device(config-default-vrf)# evpn-instance tenant_3
device(config-evpn-tenant_3)# nve nve2
device(config-evpn-tenant_3)#
```

The following example shows the EVPN instances (with associated NVE endpoints) that are running currently on the device:

```
device# show running-config

...

vrf red
  member ve 10,102-103,112,122,200,202-203,1000
  table-connections
    src-protocol static dst-protocol bgp address-family ipv4
    static-route 112.20.20.0/24 112.1.1.1
  !
router bgp
  local-as 64512
  router-id 22.22.22.22
  route-distinguisher 22.0.0.22:3
  as-notation as-plain
  graceful-restart
  address-family ipv6 unicast
    activate
  !
  address-family ipv4 unicast
    network 112.1.1.0/24
    activate
  !
instance-type evpn
  nve base
  l3vni 10010
  address-family ipv4 unicast
    ignore-as
  !
  address-family ipv6 unicast
    ignore-as
  !
  !
device#
```

nve (overlay configuration)

Use the **nve** command to create a Network Virtualization Edge (NVE) endpoint (also called an NVE node) and enter NVE configuration mode.

Syntax

```
nve nve-name
```

```
no nve nve-name
```

Command Default

No NVE endpoints exist on the device.

Parameters

nve-name

Name of the NVE endpoint to create. You can use any combination of 1 to 64 uppercase or lowercase alphabetic characters, numerals, underscores, hyphens, or periods. You must use an alphabetic character or an underscore as the first character.

Modes

Overlay network configuration (config-overlay)

Usage Guidelines

Use the **no** form of this command to delete an NVE endpoint from the device.

Examples

The following example creates an NVE named domain2 and enters NVE configuration mode:

```
device# configure terminal
device(config)# overlay-networks
device(config-overlay)# nve domain2
device(config-nve-domain2)#
```

The following example shows the Network Virtualization Overlay (NVO) instances that are running currently on the device:

```
device# show running-config overlay-networks

overlay-networks
  default-nve base
  default-vni-offset 0
  nvo base
    member nve base
  !
```

```
nve domain2
!  
nve base  
    source-interface loopback 1  
    vni-domain base  
!  
vni-domain base  
!  
device#
```


nvo (BGP peer group configuration)

Use the **nvo** command to associate a Network Virtualization Overlay (NVO) instance with a BGP peer group.

Syntax

```
nvo instance-name  
no nvo instance-name
```

Command Default

The specified BGP peer group has no associated NVO instance.

Parameters

instance-name

Name of the NVO instance to associate with the BGP peer group. You can use any combination of 1 to 64 uppercase or lowercase alphabetic characters, numerals, underscores, hyphens, or periods. You must use an alphabetic character or an underscore as the first character.

Modes

VRF BGP peer group configuration (config-vrf-bgp-pg)

Usage Guidelines

Use the **no** form of this command to disassociate the NVO instance from the specified BGP peer group.

Examples

The following example associates two NVO instances (isp1_nvo and isp2_nvo) with three peer groups (pg1, pg2, and pg3). Note that this example demonstrates you can associate a specific NVO instance with multiple BGP peer groups (in this case, isp1_nvo is associated with groups pg1 and pg2):

```
device# configure terminal  
device(config)# vrf default-vrf  
device(config-vrf-default-vrf)# router bgp  
device(config-vrf-bgp)# peer-group pg1  
device(config-vrf-bgp-pg)# nvo isp1_nvo  
device(config-vrf-bgp-pg)# exit  
device(config-vrf-bgp)# peer-group pg2  
device(config-vrf-bgp-pg)# nvo isp1_nvo  
device(config-vrf-bgp-pg)# exit  
device(config-vrf-bgp)# peer-group pg3  
device(config-vrf-bgp-pg)# nvo isp2_nvo
```

The following example shows the configured NVO instances that are running currently on the device. In this example, three BGP peer groups are configured on the device, and each one has an associated NVO instance:

```
device# show running-config vrf

vrf default-vrf
!
router bgp
  peer-group pg1
    nvo isp1_nvo

  peer-group pg2
    nvo isp1_nvo

  peer-group pg3
    nvo isp2_nvo
!
device#
```

nvo (QoS configuration)

Use the **nvo** command to associate a Network Virtualization Overlay (NVO) instance with a Quality of Service (QoS) node.

Syntax

```
nvo instance-name  
no nvo instance-name
```

Command Default

The QoS node has no associated NVO instance (endpoint).

Parameters

instance-name

Name of the NVO instance to associate with the QoS node. You can use any combination of 1 to 64 uppercase or lowercase alphabetic characters, numerals, underscores, hyphens, or periods. You must use an alphabetic character or an underscore as the first character.

Modes

QoS configuration (config-qos)

Usage Guidelines

Use the **no** form of this command to disassociate the NVO instance from the QoS node.

Examples

The following example associates an NVO instance named mynvo with a QoS node:

```
device# configure terminal  
device(config)# qos  
device(config-qos)# nvo mynvo  
device(config-qos-nvo-mynvo)#
```

The following example shows the QoS configuration that is running currently on the device. In this example, a QoS node has an associated NVO instance named mynvo:

```
device# show running-config qos  
  
qos  
  nvo mynvo  
!  
device#
```

nvo

Use the **nvo** command to create a Network Virtualization Overlay (NVO) instance and enter NVO configuration mode.

Syntax

```
nvo instance-name  
no nvo instance-name
```

Command Default

The device is configured with an NVO instance named “base.”

Parameters

instance-name

Name of the NVO instance to create. You can use any combination of 1 to 64 uppercase or lowercase alphabetic characters, numerals, underscores, hyphens, or periods. You must use an alphabetic character or an underscore as the first character.

Modes

Overlay network configuration (config-overlay)

Usage Guidelines

Use the **no** form of this command to remove an NVO instance from the device.

Examples

The following example creates an NVO instance named DG1 and enters NVO configuration mode:

```
device# configure terminal  
device(config)# overlay-networks  
device(config-overlay)# nvo DG1  
device(config-nvo-DG1)#
```

The following example shows the configured NVO instances that are running currently on the device:

```
device# show running-config overlay-networks  
  
overlay-networks  
  default-nve base  
  default-vni-offset 0  
  nvo DG1  
  !  
  nvo base
```

```
    member nve base
    member vni-domain base
    split-horizon base
    !
nve base
    source-interface loopback 1
    !
vni-domain base
    !
device#
```

nvo (VRF instance type configuration)

Use the **nvo** command to configure a Network Virtualization Overlay (NVO) instance for a virtual routing and forwarding (VRF) instance and enter VRF instance type NVO configuration mode.

Syntax

```
nvo instance-name  
no nvo instance-name
```

Command Default

The specified VRF instance has no NVO instances.

Parameters

instance-name

Name of the NVO instance to create. You can use any combination of 1 to 64 uppercase or lowercase alphabetic characters, numerals, underscores, hyphens, or periods. You must use an alphabetic character or an underscore as the first character.

Modes

VRF instance type configuration (config-vrf-*type*)

Usage Guidelines

Use the **no** form of this command to remove an NVO instance from the specified VRF instance. This removes the entire NVO configuration (such as the address family, route target, and Layer 3 VNI).

Examples

The following example configures a VRF instance named `tenant_1` as an Ethernet VPN (EVPN) and enters VRF instance type NVO configuration mode (config-vrf-*type*-nvo). In this example, the VRF instance is configured with an NVO instance named `nvo1`:

```
device# configure terminal  
device(config)# vrf tenant_1  
device(config-vrf-tenant_1)# instance-type evpn  
device(config-vrf-evpn)# nvo nvo1  
device(config-vrf-evpn-nvo)#
```

The following example displays the configuration of a VRF instance named `tenant_1` that is running currently on the device:

```
device# show running-config
```

```
...  
vrf tenant_1  
    instance-type evpn  
        nvo nvo1  
...  
device#
```

onlink (IPv6 router advertisement prefix configuration)

Use the **onlink** command to enable onlink within a prefix for IPv6 router advertisement on an interface.

Syntax

```
onlink  
no onlink
```

Command Default

Onlink within the current prefix is enabled.

Parameters

This command has no keywords or variables.

Modes

IPv6 router advertisement configuration (config-if-*name*-ipv6-router-advertisement)

Usage Guidelines

Router advertisements are ICMPv6 messages sent by an IPv6 enabled router that are used by hosts to detect routers on an attached link. This functionality is similar to the IPv4 ICMP Router Discovery protocol.

Neighbor discovery defines five ICMP packet types: router solicitation and advertisement, neighbor solicitation and advertisement, and redirect messages.

Router advertisements carry information such as the router link layer address, prefixes supported in the network, and the MTU setting. Router advertisements are periodic messages or are also sent in response to router solicitations. Because of the nature of its functionality, router advertisement is considered as a separate feature from neighbor discovery. Two messages are supported as part of this functionality: router solicitation and router advertisement.

Each configured interface sends a router advertisement message periodically to the all-nodes link-local multicast address (FF02::1). To avoid synchronization of advertisements from other routers, the time interval is randomized to send these messages between the configured maximum and minimum time periods.

Router advertisement includes the following information:

- Default router—Whether the sender can be a default router and, if so, the lifetime for the router
- List of prefixes for on-link addresses

- Auto-configuration information
- Network information—Hop limit and MTU setting to use

When onlink is enabled, the prefix is marked as being "on link" via the L flag bit for the prefix within a router advertisement. This flag tells hosts that the prefix is reachable directly on the local link. This means that packets to destinations within that prefix will be sent directly (without going through a router).

To disable onlink within a prefix for IPv6 router advertisement on an interface, use the **no** form of this command.

Examples

The following example configures a prefix of 100::1/64 for IPv6 router advertisement on Ethernet interface 0/1 and enables onlink within that prefix:

```
device# configure terminal
device(config)# interface ethernet 0/1
device(config-if-eth-0/1)# ipv6-router-advertisement
device(config-if-eth-0/1-ipv6-router-advertisement)# prefix 100::1/64
device(config-if-ipv6-router-advertisement-prefix-100::1/64)# onlink
device(config-if-ipv6-router-advertisement-prefix-100::1/64)#
```

The following example displays the configuration of Ethernet interface 0/1 that is running currently on the device. In this example, onlink is enabled within prefix 100::1/64 on the interface:

```
device# show running-config interface ethernet 0/1

!
interface ethernet 0/1
  ipv6 address 1111::10/64
  ipv6 neighbor 1111::1 00:04:96:eb:c4:51
  ipv6-router-advertisement
    hop-limit 133
    lifetime 200
    managed-config-flag
    mode disable-unsolicited-ra
    other-config
    enable
    interval 100 min 50
    prefix 100::1/63
    prefix 100::1/64
      advertisement
      autoconfiguration
      onlink
      valid-lifetime 200
  !
  no shutdown
!
device#
```

other-config

Use the **other-config** command to enable the other configuration (O) flag for IPv6 router advertisement on an interface.

Syntax

```
other-config  
no other-config
```

Command Default

The O flag is disabled.

Parameters

This command has no keywords or variables.

Modes

IPv6 router advertisement configuration (config-if-*name*-ipv6-router-advertisement)

Usage Guidelines

Router advertisements are ICMPv6 messages sent by an IPv6 enabled router that are used by hosts to detect routers on an attached link. This functionality is similar to the IPv4 ICMP Router Discovery protocol.

When this feature is enabled, the O flag is set in the advertised router advertisement. The O flag indicates that there is other configuration available via DHCPv6 (such as DNS servers).

To disable the O flag, use the **no** form of this command.

Examples

The following example configures the O flag on Ethernet interface 0/1:

```
device# configure terminal  
device(config)# interface ethernet 0/1  
device(config-if-eth-0/1)# ipv6-router-advertisement  
device(config-if-eth-0/1-ipv6-router-advertisement)# other-config  
device(config-if-eth-0/1-ipv6-router-advertisement)#
```

The following example displays the configuration of Ethernet interface 0/1 that is running currently on the device. In this example, the O flag is enabled on the interface:

```
device# show running-config interface ethernet 0/1  
  
!  
interface ethernet 0/1  
  ipv6 address 1111::10/64
```

```
ipv6 neighbor 1111::1 00:04:96:eb:c4:51
ipv6-router-advertisement
  hop-limit 133
  lifetime 200
  managed-config-flag
  mode disable-unsolicited-ra
  other-config
  interval 100 min 50
  enable
  prefix 100::1/64
  prefix 100::1/63
    valid-lifetime 200
  !
  no shutdown
  !
device#
```

output (interface QoS configuration)

Use the **output** command to enter the mode for configuring access QoS in the output direction on an interface.

Syntax

output

no output

Command Default

Access QoS is not configured in the output direction on the current interface.

Parameters

This command has no keywords or variables.

Modes

Ethernet interface QoS configuration (config-qos-if-eth-*name*)

Port channel interface QoS configuration (config-qos-if-po-*name*)

Virtual Ethernet interface QoS configuration (config-qos-if-ve-*name*)

Usage Guidelines

This command enters the mode for accessing the commands for configuring access QoS in the output direction on an interface. You use these commands to configure QoS maps for remarking and attach them to interfaces in the incoming direction and also enable CoS remarking or DSCP remarking in the incoming direction.

To remove access QoS configuration in the output direction on the interface, use the **no** form of this command.

Examples

The following example configures access QoS on interface Ethernet 0/1 and its subinterface 100 in the output direction:

```
device# configure terminal
device(config)# qos
device(config-qos)# interface ethernet 0/1 sub-interface 100
device(config-qos-if-eth-0/1:1.100# output
device(config-qos-if-eth-0/1:1.100-output)# remark dscp
device(config-qos-if-eth-0/1:1.100-output)# remark-map egr_l2_map
device(config-qos-if-eth-0/1:1.100-output)#
```

overlay-networks

Use the **overlay-networks** command to enter overlay network configuration mode.

Syntax

overlay-networks

Parameters

This command has no keywords or variables.

Modes

Global configuration (config)

Usage Guidelines

This mode lets you access the commands for setting up an overlay network. For example, these commands let you configure Network Virtualization Overlay (NVO) instances, create and add Network Virtualization Endpoints (NVEs) to an NVO instance, and create Virtual Network Identifier (VNI) domains.

Examples

The following example enters overlay network configuration mode:

```
device# configure terminal
device(config)# overlay-networks
device(config-overlay)#
```

The following example displays information about the NVO instances that are configured currently on the device:

```
device# show running-config overlay-networks

overlay-networks
  default-nve base
  default-vni-offset 0
  nvo base
    member nve base
  !
  nve base
    vni-domain base
  !
  vni-domain base
!
device#
```

password (IAH VM configuration)

Use the **password** command to configure a new password for the "extreme" username for an IAH third party virtual machine (TPVM).

Syntax

password *password*

no password

Command Default

The default password is configured on the current VM.

Parameters

password

New password. Use 8 to 128 characters. This string must contain at least one uppercase letter, one lowercase case letter, one digit, and one special character. The password must follow any additional attributes that are determined by the **password-attributes** command if they exist.

Modes

IAH VM configuration (config-iah-vm-*name*)

Usage Guidelines

To restore the password to the default value, use the **no** form of this command.

To display details about a specific VM, see [show iah vm](#) on page 1006.

Examples

The following example configures a password for the "extreme" username for the TPVM on the device:

```
device# configure terminal
device(config)# iah
device(config-iah)# vm TPVM
device(config-iah-vm-TPVM)# password qwertyQ123&%
device(config-iah-vm-TPVM)#
```

peer-group

Use the **peer-group** command to configure a BGP peer group within a Virtual Routing and Forwarding (VRF) instance and enter BGP peer group configuration mode.

Syntax

peer-group *peer-group-name*

no peer-group *peer-group-name*

Command Default

The specified VRF instance contains no BGP peer groups.

Parameters

peer-group-name

Name of the BGP peer group to configure. Use 1 to 120 uppercase or lowercase alphabetic characters, numerals, underscores, or hyphens. You must use an alphabetic character or an underscore as the first character.

Modes

VRF BGP configuration (config-vrf-bgp)

Usage Guidelines

Use this command to create a BGP peer group to contain neighbors that have the same update policies (such as outbound route maps, distribute lists, filter lists, and update source). You add neighbors to a peer group and then apply all of your configurations to the peer group. BGP will prepare the updates for the peer group (which requires fewer CPU resources than preparing them for each neighbor separately).

Although peer group members inherit the configuration options of the peer group, you can configure individual members to override those options that do not affect outbound updates.

Use the **no** form of this command to remove a BGP peer group.



Caution

The **no** form of this command deletes the entire peer group configuration, such as the remote autonomous system (AS) number, peer group neighbors, and all the routes that the peer group neighbors have learned. Improper usage of this command might disrupt service.

Examples

The following example configures a BGP peer group named underlay within a VRF instance named blue:

```
device# configure terminal
device(config)# vrf blue
device(config-vrf-blue)# router bgp
device(config-vrf-bgp)# peer-group underlay
device(config-vrf-bgp-pg)#
```

The following example displays the configuration of a VRF instance named blue that is running currently on the device. In this example, the instance contains a BGP peer group named underlay:

```
device# show running-config vrf blue

vrf blue
!
  router bgp
    peer-group underlay
!
device#
```


peer (MLAG configuration)

Use the **peer (MLAG configuration)** command to configure a multichassis link aggregation (MLAG) peer and enter MLAG peer configuration mode.

Syntax

```
peer peer-name  
no peer peer-name
```

Command Default

No MLAG peer exists on the device.

Parameters

peer-name

Name of the MLAG peer to be configured. You can use any combination of 1 to 64 numerals, underscores, hyphens, periods, and uppercase or lowercase letters. The name must start with a letter or underscore.

Modes

MLAG peer configuration (config-mlag-peer-*name*)

Usage Guidelines

Only one MLAG peer is allowed on a device.

To remove the MLAG peer from a device, use the no form of this command.

Examples

The following example configures an MLAG peer named peer01 and enters MLAG peer configuration mode:

```
device# configure terminal  
device(config)# mlag  
device(config-mlag)# peer peer01  
device(config-mlag-peer-peer01)#
```

The following example displays the MLAG peer configuration that is running currently on the device. The output includes the name of the peer:

```
device# show running-config mlag peer  
  
mlag  
  keepalive interval 1000 delay 100 multiplier 4  
  mac a001.a002.a003  
  interface port-channel 1 id auto  
!
```

```
peer mypeer1
!  
device#
```

The following example displays the status and other information about the peer that is configured on the device:

```
device# show mlag peer

Peer DUT-6
=====
Peer State           : UP
MCP State            : UP
Role                 : PRIMARY
Elected MAC         : e4:44:e5:00:94:00
Extend Bridge Count  : 361
Peer Exception       : None
  Keepalive prim
  -----
  Destination Address: 20.20.20.164
  State               : UP
  Type                : PRIMARY
  Address Type        : MANUAL
  Source Interface    : loopback 1024 ( 20.20.20.163 )
  First Rx            : 2025-06-27 12:11:27
  Last Rx             : 2025-06-27 12:50:59
  Keepalive Second
  -----
  Destination Address: 10.32.100.35
  State               : UP
  Type                : SECONDARY
  Address Type        : AUTO
  Source Interface    : management 0 ( 10.32.99.45 )
  First Rx            : 2025-06-27 12:11:41
  Last Rx             : 2025-06-27 12:50:59
device#
```

permit (IPv4 ACL configuration)

Use the **permit** command to command to configure an access control rule in an IPv4 access control list (ACL) to permit forwarding of packets.

Syntax

```
[ seq sequence-num ] permit { ip-protocol-num | ipv4 | esp | icmp | igmp
| udp | tcp [ flag-list ] } { source-ipv4-addr subnet-mask | source-ipv4-addr/prefix-length | any } { dest-ipv4-addr subnet-mask | dest-ipv4-addr/prefix-length | any } [ { dontfragment | morefragment } ]
[ dport dest-l4-port ] [ sport source-l4-port ] [ dscp packet-dsp-value ] [ vlan packet-vlan-id ] [ count ] [ log ]
```

no seq *sequence-num*

```
no permit { ip-protocol-num | ipv4 | esp | icmp | igmp | udp | tcp
[ flag-list ] } { source-ipv4-addr subnet-mask | source-ipv4-addr/prefix-length | any } { dest-ipv4-addr subnet-mask | dest-ipv4-addr/prefix-length | any } [ { dontfragment | morefragment } ] [ dport
dest-l4-port ] [ sport source-l4-port ] [ dscp packet-dsp-value ] [ vlan packet-vlan-id ] [ count ] [ log ]
```

Command Default

The ACL is not configured with permit rules.

Parameters

seq *sequence-num*

(Optional) Sequence number to identify the relative priority of this rule in the ACL. Specify an integer from 1 to 65535. You should specify an unused sequence number in a multiple of 10. If you do not specify a value, the rule is added to the end of the list.

ip-protocol-num

Matches packets that use a specific IP protocol. Specify an integer from 1 to 254. For example, specify 6 for TCP packets or 115 for L2TP packets. For a list of IP protocol numbers, see <https://www.iana.org/assignments/protocol-numbers/protocol-numbers.xhtml>.

ipv4

Matches packets that use any IPv4 protocol (including ICMPv4, IGMPv4, UDP, and TCP).

esp

Matches packets encrypted with Encapsulation Security Payload (ESP).

icmp

Matches packets carrying Internet Control Message Protocol v4 (ICMPv4) messages. These packets contain the control messages such as Echo Request/Echo Reply or Destination Unreachable.

igmp

Matches packets carrying Internet Group Management Protocol v4 (IGMPv4) messages. Hosts and adjacent routers use these on IPv4 networks to establish multicast group memberships, which optimize performance for one-to-many networking applications such as streaming video.

udp

Matches packets that use the User Datagram Protocol (UDP) format. This allows packets using connectionless protocols such as TFTP and DNS.

tcp [*flag-list*]

Matches TCP packets. This allows packets that use TCP connection-oriented protocols such as HTTPS, FTP, and Telnet. You can omit flags to match all TCP packets, or you can specify any space-separated combination of the flags listed in [Table 9](#) on page 854 below.

source-ipv4-addr subnet-mask | *source-ipv4-addr/prefix-length* | **any**

Source IPv4 addresses. These represent the networks or hosts sending the packets. Use one of the following methods:

source-ipv4-addr subnet-mask matches packets from a range of IPv4 addresses via a subnet mask. Use the *a.b.c.d* format for the IPv4 address and the subnet mask.

source-ipv4-addr/prefix-length matches packets from a range of IPv4 addresses via a subnet prefix length. Use the *a.b.c.d* format for the IPv4 address, and use an integer from 0 to 32 for the prefix. You must include the slash (/).

any matches packets from any source IPv4 address.

dest-ipv4-addr/prefix-length | **any**

Destination IPv4 addresses. These represent the networks or hosts to receive the packets. Use one of the following methods:

dest-ipv4-addr subnet-mask matches packets going to a range of IPv4 addresses via a subnet mask. Use the *a.b.c.d* format for the IPv4 address and the subnet mask.

dest-ipv4-addr/prefix-length matches packets going to a range of IPv4 addresses via a subnet prefix length. Use the *a.b.c.d* format for the IPv4 address, and use an integer from 0 to 32 for the prefix. You must include the slash (/).

any matches packets going to any destination IPv4 address.

dontfragment | **morefragment**

(Optional) Matches only IPv4 nonfragmented or only IPv4 fragmented packets respectively.

dport *dest-l4-port*

(Optional) Destination layer 4 port. Specify an integer from 1 to 65535.

For example, 22 (for SSH traffic) or 443 (for HTTPS traffic). For details, see

<https://www.iana.org/assignments/service-names-port-numbers/service-names-port-numbers.xhtml>.

sport *source-l4-port*

(Optional) Source layer 4 port. Specify an integer from 1 to 65535. For details, see <https://www.iana.org/assignments/service-names-port-numbers/service-names-port-numbers.xhtml>.

dscp *packet-dsp-value*

(Optional) Matches packets containing a specific Differentiated Services Code Point (DSCP) value. Differentiated services are for managing traffic and providing quality of service (QoS). For example, they can provide low latency to voice or streaming media but only best effort service to web traffic or file transfers. Specify an integer from 0 to 63.

vlan *packet-vlan-id*

(Optional) Matches packets containing a specific Virtual LAN (VLAN) ID. Specify an integer from 0 to 4095. Use this argument to apply the rule to a VLAN with an ID other than 1 (the default VLAN ID). You can use this keyword if your physical LAN is partitioned into VLANs (logical broadcast domains to isolate services for improved security and management).

count

(Optional) Enables the counter for this rule. This counts the number of packets and bytes that match this rule in its access list.

To display the counters as a number of matches, see [show access-list](#) on page 937.

You must clear the counters if you want to start at zero for a more recent count of the packets and bytes that match an access list. For details, see [clear counters access-list](#) on page 508.

Modes

IPv4 ACL configuration (config-ip-acl)

Usage Guidelines

The order of the rules in an ACL is critical, because the first matching rule stops further processing. Rules that specify sequence values will determine the order of rule processing. If you do not specify a sequence value, the rule is added to the end of the list.

Traffic not specifically denied is allowed by a default final permit rule.

Conflicting rules (ones with the same match condition but different actions) are not allowed.

To remove a rule from an IPv4 ACL, use the **no** form of this command:

- If you know the rule number, enter **no seq** *sequence-num*.

- If you *do not* know the rule number, enter **no** followed by the full syntax (without **seq sequence-num**).

The following table lists the optional TCP flags:

Table 9: Optional TCP flags

Flag	Protocol (Packet Type)
ack	Acknowledgment. These packets confirm that the data packets were received and confirm initiation requests and teardown requests. They are used along with synchronization packets for the three-way TCP handshake.
cwr	Congestion Window has been Reduced (CWR). These packets indicate that the sending host has received a TCP segment with the ECE flag set. The congestion window is an internal variable maintained by TCP to manage the size of the send window.
ece	Explicit Congestion Notification (ECN)- Echo. These packets indicate that a TCP peer is ECN-capable during the TCP three-way handshake and also indicate that a TCP segment was received on the connection with the ECN field in the IP header set to 11.
fin	Finish sending data. These packets indicate that the connection is being torn down. Both the sender and receiver send these packets to terminate the connection gracefully.
push	Push function. These packets indicate that the contents of the TCP receive buffer should be passed on to the application layer protocol directly instead of being buffered.
reset	Reset the connection. These packets indicate that the connection is down or the service is not accepting requests.
sync	Synchronization packets. These packets initiate a connection. They are used with acknowledgment packets as part of the three-way TCP handshake.
urg	These packets indicate that the data inside the packet should be processed immediately by the TCP stack.

Examples

The following example configures an IPv4 ACL named v4acl on the device and adds a permit rule to the ACL:

```
device# configure terminal
device(config)# ipv4 access-list v4acl
device(config-ip-acl)# seq 10 permit ipv4 1.1.1.1/32 2.2.2.0 255.255.255.0 count
device(config-ip-acl)#
```

The following example shows the ACLs that are configured currently on the device:

```
device# show running-config access-list

ipv4 access-list v4acl
seq 10 permit ipv4 1.1.1.1 255.255.255.255 2.2.2.0 255.255.255.0 count
device#
```

permit (IPv6 ACL configuration)

Use the **permit** command to command to configure an access control rule in an IPv6 access control list (ACL) to permit forwarding of packets.

Syntax

```
[ seq sequence-num ] permit { ip-protocol-num | ipv6 | esp | icmpv6 |
  igmpv6 | udp | tcp [ flag-list ] } { source-ipv6-addr subnet-mask
  | source-ipv6-addr/prefix-length | any } { dest-ipv6-addr subnet-
  mask | dest-ipv6-addr/prefix-length | any } [ { dontfragment |
  morefragment } ] [ dport dest-l4-port ] [ sport source-l4-port ]
  [ dscp packet-dsp-value ] [ vlan packet-vlan-id ] [ count ] [ log ]

no seq sequence-num

no permit { ip-protocol-num | ipv6 | icmpv6 | esp | igmpv6 | udp | tcp
  [ flag-list ] } { source-ipv6-addr subnet-mask | source-ipv6-addr/
  prefix-length | any } { dest-ipv6-addr subnet-mask | dest-ipv6-addr/
  prefix-length | any } [ { dontfragment | morefragment } ] [ dport
  dest-l4-port ] [ sport source-l4-port ] [ dscp packet-dsp-value ]
  [ vlan packet-vlan-id ] [ count ] [ log ]
```

Command Default

The ACL is not configured with permit rules.

Parameters

seq *sequence-num*

(Optional) Sequence number to identify the relative priority of this rule in the ACL. Specify an integer from 1 to 65535. You should specify an unused sequence number in a multiple of 10. If you do not specify a value, the rule is added to the end of the list.

ip-protocol-num

Matches packets that use a specific IP protocol. Specify an integer from 1 to 254. For example, specify 6 for TCP packets or 115 for L2TP packets. For a list of IP protocol numbers, see <https://www.iana.org/assignments/protocol-numbers/protocol-numbers.xhtml>.

ipv6

Matches packets that use any IPv6 protocol (including ICMPv6, IGMPv6, UDP, and TCP).

esp

Matches packets encrypted with Encapsulation Security Payload (ESP).

icmp

Matches packets carrying Internet Control Message Protocol v6 (ICMPv6) messages. These packets contain the control messages such as Echo Request/Echo Reply or Destination Unreachable.

igmpv6

Matches packets carrying Internet Group Management Protocol v6 (IGMPv6) messages. Hosts and adjacent routers use these on IPv6 networks to establish multicast group memberships, which optimize performance for one-to-many networking applications such as streaming video.

udp

Matches packets that use the User Datagram Protocol (UDP) format. This allows packets using connectionless protocols such as TFTP and DNS.

tcp [*flag-list*]

Matches TCP packets. This allows packets that use TCP connection-oriented protocols such as HTTPS, FTP, and Telnet. You can omit flags to match all TCP packets, or you can specify any space-separated combination of the flags listed in [Table 10](#) on page 859 below.

source-ipv6-addr subnet-mask | *source-ipv6-addr/prefix-length* | **any**

Source IPv6 addresses. These represent the networks or hosts sending the packets. Use one of the following methods:

source-ipv6-addr subnet-mask matches packets from a range of IPv6 addresses via a subnet mask. Use the *x:x::x:x* format for the IPv6 address and the subnet mask.

source-ipv6-addr/prefix-length matches packets from a range of IPv6 addresses via a subnet prefix length. Use the *x:x::x:x* format for the IPv6 address, and use an integer from 0 to 128 for the prefix. You must include the slash (/).

any matches packets from any source IPv6 address.

dest-ipv6-addr/prefix-length | **any**

Destination IPv6 addresses. These represent the networks or hosts to receive the packets. Use one of the following methods:

dest-ipv6-addr subnet-mask matches packets going to a range of IPv6 addresses via a subnet mask. Use the *x:x::x:x* format for the IPv6 address and the subnet mask.

dest-ipv6-addr/prefix-length matches packets going to a range of IPv6 addresses via a subnet prefix length. Use the *x:x::x:x* format for the IPv6 address, and use an integer from 0 to 128 for the prefix. You must include the slash (/).

any matches packets going to any destination IPv6 address.

dontfragment | **morefragment**

(Optional) Matches only IPv6 nonfragmented or only IPv6 fragmented packets respectively.

dport *dest-l4-port*

(Optional) Destination layer 4 port. Specify an integer from 1 to 65535.

For example, 22 (for SSH traffic) or 443 (for HTTPS traffic). For details, see

<https://www.iana.org/assignments/service-names-port-numbers/service-names-port-numbers.xhtml>.

sport *source-l4-port*

(Optional) Source layer 4 port. Specify an integer from 1 to 65535. For details, see <https://www.iana.org/assignments/service-names-port-numbers/service-names-port-numbers.xhtml>.

dscp *packet-dsp-value*

(Optional) Matches packets containing a specific Differentiated Services Code Point (DSCP) value. Differentiated services are for managing traffic and providing quality of service (QoS). For example, they can provide low latency to voice or streaming media but only best effort service to web traffic or file transfers. Specify an integer from 0 to 63.

vlan *packet-vlan-id*

(Optional) Matches packets containing a specific Virtual LAN (VLAN) ID. Specify an integer from 0 to 4095. Use this argument to apply the rule to a VLAN with an ID other than 1 (the default VLAN ID). You can use this keyword if your physical LAN is partitioned into VLANs (logical broadcast domains to isolate services for improved security and management).

count

(Optional) Enables the counter for this rule. This counts the number of packets and bytes that match this rule in its access list.

To display the counters as a number of matches, see [show access-list](#) on page 937.

You must clear the counters if you want to start at zero for a more recent count of the packets and bytes that match an access list. For details, see [clear counters access-list](#) on page 508.

Modes

IPv6 ACL configuration (config-ipv6-acl)

Usage Guidelines

The order of the rules in an ACL is critical, because the first matching rule stops further processing. Rules that specify sequence values will determine the order of rule processing. If you do not specify a sequence value, the rule is added to the end of the list.

Traffic not specifically denied is allowed by a default final permit rule .

Conflicting rules (ones with the same match condition but different actions) are not allowed.

To remove a rule from an IPv6 ACL, use the **no** form of this command:

- If you know the rule number, enter **no seq** *sequence-num*.

- If you *do not* know the rule number, enter **no** followed by the full syntax (without **seq sequence-num**).

The following table lists the optional TCP flags:

Table 10: Optional TCP flags

Flag	Protocol (Packet Type)
ack	Acknowledgment. These packets confirm that the data packets were received and confirm initiation requests and teardown requests. They are used along with synchronization packets for the three-way TCP handshake.
cwr	Congestion Window has been Reduced (CWR). These packets indicate that the sending host has received a TCP segment with the ECE flag set. The congestion window is an internal variable maintained by TCP to manage the size of the send window.
ece	Explicit Congestion Notification (ECN)- Echo. These packets indicate that a TCP peer is ECN-capable during the TCP three-way handshake and also indicate that a TCP segment was received on the connection with the ECN field in the IP header set to 11.
fin	Finish sending data. These packets indicate that the connection is being torn down. Both the sender and receiver send these packets to terminate the connection gracefully.
push	Push function. These packets indicate that the contents of the TCP receive buffer should be passed on to the application layer protocol directly instead of being buffered.
reset	Reset the connection. These packets indicate that the connection is down or the service is not accepting requests.
sync	Synchronization packets. These packets initiate a connection. They are used with acknowledgment packets as part of the three-way TCP handshake.
urg	These packets indicate that the data inside the packet should be processed immediately by the TCP stack.

Examples

The following example configures an IPv6 ACL named v6acl on the device and adds a permit rule to the ACL:

```
device# configure terminal
device(config)# ipv6 access-list v6acl
device(config-ipv6-acl)# permit tcp 1::0/64 2::0 ffff::0 sync
device(config-ipv6-acl)#
```

The following example shows the ACLs that are configured currently on the device:

```
device# show running-config access-list

ip access-list v6acl
seq 10 permit tcp 1::0 ffff:ffff:ffff:ffff::0 2::0 ffff::0 sync
device#
```

permit (MAC ACL configuration)

Use the **permit** command to create or modify a rule that marks packets from a specified source MAC address or to a specified destination MAC address (or both) for admission.

Syntax

```
[ seq sequence-num ] permit { source_address address_mask | any }
    { destination_address address_mask | any } [ etype ethernet_type ]
    [ vlan vlan_ID ] [ pcp dot1p_or_cos_value ] [ count ]

no seq sequence-num

no permit { source_address address_mask | any } { destination_address
    address_mask | any } [ etype ethernet_type ] [ vlan vlan_ID ] [ pcp
    dot1p_or_cos_value ] [ count ]
```

Command Default

The ACL is not configured with permit rules.

Parameters

seq *sequence-num*

(Optional) Sequence number to identify the relative priority of this rule in the ACL. Specify an integer from 1 to 65535. You should specify an unused sequence number in a multiple of 10. If you do not specify a value, the rule is added to the end of the list.

source-address address-mask | **any**

Source MAC address and mask. Use six groups of two hex digits separated by colons. For example, 00:00:5E:00:53:00 ff:ff:ff:ff:ff:ff.

any matches packets coming from any source MAC address.

destination-address address-mask | **any**

Destination MAC address and mask. Use six groups of two hex digits separated by colons. For example, 00:00:5E:00:53:FF ff:ff:ff:00:00:00.

any matches packets going to any destination MAC address.

etype ethernet_type

(Optional) Ethernet type. You must specify a hexadecimal number in the range of 0x600 to 0xFFFF or the **arp**, **ipv4**, or **ipv6** protocol.

pcp *dot1p_or_cos_value*

(Optional) 802.1p or class of service (CoS) value for payload compression protocol (PCP). The range is an integer from 1 to 7.

vlan *vlan_ID*

(Optional) Matches packets containing a specific Virtual LAN (VLAN) ID. Specify an integer from 0 to 4095. Use this argument to apply the rule to a VLAN with an ID other than 1 (the default VLAN ID). You can use this keyword if your physical LAN is partitioned into VLANs (logical broadcast domains to isolate services for improved security and management).

count

(Optional) Enables the counter for this rule. This counts the number of packets and bytes that match this rule in its access list.

To display the counters as a number of matches, see [show access-list](#) on page 937.

You must clear the counters if you want to start at zero for a more recent count of the packets and bytes that match an access list. For details, see [clear counters access-list](#) on page 508.

Modes

MAC ACL configuration (config-mac-acl)

Usage Guidelines

The order of the rules in an ACL is critical, because the first matching rule stops further processing. Rules that specify sequence values will determine the order of rule processing. If you do not specify a sequence value, the rule is added to the end of the list.

Traffic not specifically denied is allowed by a default final permit rule.

Conflicting rules (ones with the same match condition but different actions) are not allowed.

To remove a rule from a MAC ACL, use the **no** form of this command:

- If you know the rule number, enter **no seq** *sequence-num*.
- If you *do not* know the rule number, enter **no** followed by the full syntax (without **seq** *sequence-num*).

Examples

The following example adds a permit rule to a MAC ACL named macacl. This example matches packets with a source MAC address and address mask 11:22:33:44:55:66 ff:ff:ff:ff:ff:ff respectively and packets with any destination MAC address with a VLAN ID of 56:

```
device# configure terminal
device(config)# mac access-list macacl
device(config-mac-acl)# permit 00:00:5E:00:53:00 ff:ff:ff:ff:ff:ff any vlan 56
device(config-mac-acl)#
```

The following example displays the currently-running ACLs and their corresponding rules. In this example, only one MAC ACL (which is named macacl) is running on the device:

```
device# show running-config access-list

mac access-list macacl
permit 00:00:5E:00:53:00 ff:ff:ff:ff:ff:ff any vlan 56
device#
```

ping

Use the **ping** command to verify network connectivity between a source and a destination on a TCP/IP network.

Syntax

```
ping { dest-ipv4-addr [ ipv6 dest-ipv6-addr ] [ host-name ] } [ count  
  [ number ] [ datagram-size bytes ] [ interface { ethernet interface-  
    name | management | port-channel port-number | ve vlan-id } ]  
  [ quiet ] [ source source-addr ] [ timeout seconds ] [ vrf { mgmt-vrf  
    | default-vrf | vrf-name } ]
```

Parameters

dest-ipv4-addr

IPv4 address of the destination device.

dest-ipv6-addr

IPv6 address of the destination device.

host-name

Hostname of the destination device.

count *number*

(Optional) Number of transmissions (pings). The range is 1 to 1000. The default value is 5.

datagram-size *bytes*

(Optional) Datagram size (also called the maximum transmission unit, or MTU) in bytes. The range is 18 to 9200. The default is 64.

ethernet

(Optional) Specifies a physical Ethernet interface.

slot

Valid slot number.

port

Valid port number.

management

(Optional) Specifies the management interface.

port-channel *port-number*

(Optional) Port channel (LAG) interface number.

ve *vlan-id*

(Optional) Virtual Ethernet (VE) interface VLAN ID.

source *source-addr*

(Optional) Source IPv4 or IPv6 address.

quiet

(Optional) Displays only the first line and the last line of the command output.

timeout *seconds*

(Optional) Time (in seconds) to wait for a response. The range is 1 through 60. The default value is 5.



Note

This option applies only to IPv4.

vrf

(Optional) Pings the specified VRF instance.

mgmt-vrf

Specifies the management VRF.

default-vrf

Specifies the default VRF.

vrf-name

User-created VRF name.

Modes

EXEC (#)

Usage Guidelines

This command sends a specified number of pings with configured parameters to the specified destination device.

Examples

The following example pings an IPv4 destination address:

```
device# ping 10.32.94.135 vrf mgmt-vrf

PING 10.32.94.135 (10.32.94.135) 56(84) bytes of data.
64 bytes from 10.32.94.135 icmp_seq=1 ttl=54 time=304 ms
64 bytes from 10.32.94.135 icmp_seq=2 ttl=54 time=304 ms
64 bytes from 10.32.94.135 icmp_seq=3 ttl=54 time=303 ms
device#
```

The following example pings an IPv4 destination address in quiet mode:

```
device# ping 10.32.94.135 vrf mgmt-vrf quiet

PING 10.37.36.116 (10.37.36.116) from 10.37.36.116 default-vrf: 56(84) bytes of data.

--- 10.37.36.116 ping statistics ---
5 packets transmitted, 5 received, 0% packet loss, time 4089ms
device#
```

The following example pings an IPv6 destination address:

```
device# ping ipv6 fec0:60:69bc:92:218:8bff:fe40:1470 vrf mgmt-vrf
```

```
ping fec0:60:69bc:92:218:8bff:fe40:1470
PING fec0:60:69bc:92:218:8bff:fe40:1470 (fec0:60:69bc:92:218:8bff:fe40:1470) from
fec0:60:69bc:92:218:8bff:fe40:1470 default-vrf: 54(82) bytes of data.

--- fec0:60:69bc:92:218:8bff:fe40:1470 ping statistics ---
3 packets transmitted, 3 received, 0% packet loss, time 4192ms
device#
```

policy-result (routing policy statement actions configuration)

Use the **policy-result** command to configure the result of a routing policy statement.

Syntax

```
policy-result { permit | deny }  
no policy-result { permit | deny }
```

Command Default

If the specified routing policy statement matches for a route, the route is denied, and no set actions of that statement are executed.

Parameters

permit

Routes are allowed, and set actions of the statement take effect.

deny

Routes are denied, and no set actions of the statement are executed.

Modes

Routing policy statement actions configuration (config-route-policy-policy-*name*-stmt-*number*-actions)

Usage Guidelines

This command configures the result of the route policy statement. If a statement matches for a route, the configured policy result is applied. If the result is "deny," routes are denied, and no set actions of that statement are executed. If the result is "permit," the routes are allowed, and set actions of that statement come into effect. If no result is configured, by default it will be "deny."

To restore the default policy result behavior for a routing policy statement, use the **no** form of this command.

Examples

The following example creates a routing policy named `map1` that contains a policy statement with sequence number 10. In this example, if the statement matches for a route, the route is allowed, and set actions of that statement take effect:

```
device# configure terminal  
device(config)# route-policy  
device(config-route-policy)# policy map1  
device(config-route-policy-policy-map1)# statement 10
```

```
device(config-route-policy-policy-map1-stmt-10)# actions
device(config-route-policy-policy-map1-stmt-10-actions)# policy-result permit
```

The following example displays the routing policy configuration that is running currently on the device. In this example, a routing policy named map1 contains a policy statement with sequence number 10. In this example, if the statement matches for a route, the route is allowed, and set actions of that statement take effect:

```
device# show running-config route-policy

route-policy
  policy map1
    statement 10
      actions
        policy-result permit
device#
```

policy (routing policy configuration)

Use the **policy** command to create a routing policy and enter Routing policy configuration (config-route-policy) mode.

Syntax

```
policy policy-name  
no policy policy-name
```

Command Default

No routing policies are configured on the device.

Parameters

policy-name

Name of the routing policy. Use an arbitrary sequence of 1 to 64 letters, numerals, underscores, hyphens, or periods that starts with a letter.

Modes

Routing policy configuration (config-route-policy)

Usage Guidelines

This command creates a named routing policy and enters the mode for adding policy statements to that routing policy.

To remove a routing policy from the device, use the **no** form of this command.

Examples

The following example creates a routing policy named `map1` that contains a policy statement with sequence number 10:

```
device# configure terminal  
device(config)# route-policy  
device(config-route-policy)# policy map1  
device(config-route-policy-policy-map1)# statement 10  
device(config-route-policy-policy-map1-stmt-10)#
```

The following example displays the routing policy configuration that is running currently on the device. In this example, a routing policy named `map1` contains a policy statement with sequence number 10:

```
device# show running-config route-policy  
  
route-policy  
  policy map1  
    statement 10
```

```
    conditions
      bgp-conditions
        match-community-set comm1
device#
```

port ethernet

Use the **port ethernet** command to configure an Ethernet interface as a Redundant Management Ethernet (RME) port on a device.

Syntax

```
port ethernet interface-name performance-mode internal { 0 | 1 }  
no port ethernet interface-name performance-mode internal { 0 | 1 }
```

Command Default

No RME port is configured on the device.

Parameters

interface-name

Ethernet front panel interface number. Use the *slot* format, the *slot/port* format, or the *slot/port:breakout* format to specify an interface or one or more comma-separated ranges of interfaces. For example, 1, 1/1, or 1/1-3,5,2/7-9,10:1-4 respectively.

performance-mode internal

Specifies the performance mode.

Traffic from ASIC to CPU and vice versa can be forwarded either using ASIC - CPU PCIe bus (called CPU mode) or ASIC - CPU directly connected ports referred as internal ports (called performance mode). CPU mode based RME is not implemented in this release because it restricts OOB bandwidth. Whereas internal port mode provides a constant 10G bandwidth, so regardless of ASIC port speed, when this redundant path is active for OOB purposes, 10G is the maximum achievable bandwidth.

{ 0 | 1 }

Internal port number. This is also called the sideband or insight port number.

Modes

RME system configuration (config-system-rme)

Usage Guidelines

Use this command to specify an ASIC front panel port as a redundant port of the out of band (OOB) management port. This feature provides management traffic path redundancy by allowing two ports to connect to the management LAN. This feature forwards management traffic to the CPU via an ASIC front panel port in both directions if the OOB management port fails.

This feature can use only one Ethernet interface as an RME interface for OOB purposes.

Use the **no** form of this command to remove the RME configuration from the Ethernet interface.

Examples

The following example configures Ethernet port 0/15 as the RME port in performance internal port mode on the device:

```
device# configure terminal
device(config)# system
device(config-system)# redundant-management
device(config-system-rme)# port ethernet 0/15 performance-mode internal 1
device(config-system-rme)#
```

The following example displays the system configuration that is running currently on the device. In this example, Ethernet port 0/15 is configured as the RME port in performance internal port mode on the device:

```
device# show running-config system

system
  redundant-management
    port ethernet 0/15 performance-mode internal 0
  !
!
device#
```


preferred-lifetime (IPv6 router advertisement prefix configuration)

Use the **preferred-lifetime** command to configure the preferred lifetime of the address within a prefix for IPv6 router advertisement on an interface.

Syntax

```
preferred-lifetime seconds  
no preferred-lifetime
```

Command Default

The preferred lifetime for the address within the current prefix is 604,800 seconds.

Parameters

seconds

Number of seconds. Use an integer from 0 to 4294967295. This value cannot exceed the **valid-lifetime** value.

Modes

IPv6 router advertisement configuration (config-if-*name*-ipv6-router-advertisement)

Usage Guidelines

Router advertisements are ICMPv6 messages sent by an IPv6 enabled router that are used by hosts to detect routers on an attached link. This functionality is similar to the IPv4 ICMP Router Discovery protocol.

Neighbor discovery defines five ICMP packet types: router solicitation and advertisement, neighbor solicitation and advertisement, and redirect messages.

Router advertisements carry information such as the router link layer address, prefixes supported in the network, and the MTU setting. Router advertisements are periodic messages or are also sent in response to router solicitations. Because of the nature of its functionality, router advertisement is considered as a separate feature from neighbor discovery. Two messages are supported as part of this functionality: router solicitation and router advertisement.

Each configured interface sends a router advertisement message periodically to the all-nodes link-local multicast address (FF02::1). To avoid synchronization of advertisements from other routers, the time interval is randomized to send these messages between the configured maximum and minimum time periods.

Router advertisement includes the following information:

- Default router—Whether the sender can be a default router and, if so, the lifetime for the router
- List of prefixes for on-link addresses
- Auto-configuration information
- Network information—Hop limit and MTU setting to use

The preferred lifetime is the length of time that the address within the prefix remains in the preferred state, which means that unrestricted use is allowed by upper-layer protocols.

To restore the preferred lifetime for the address within a prefix to the default value, use the **no** form of this command.

Examples

The following example configures a prefix of 100::1/64 for IPv6 router advertisement on Ethernet interface 0/1 and configures a preferred lifetime of 5,000 seconds within that prefix:

```
device# configure terminal
device(config)# interface ethernet 0/1
device(config-if-eth-0/1)# ipv6-router-advertisement
device(config-if-eth-0/1-ipv6-router-advertisement)# prefix 100::1/64
device(config-if-ipv6-router-advertisement-prefix-100::1/64)# preferred-lifetime 5000
device(config-if-ipv6-router-advertisement-prefix-100::1/64)#
```

The following example displays the configuration of Ethernet interface 0/1 that is running currently on the device. In this example, a preferred lifetime of 5,000 seconds is configured within prefix 100::1/64 on the interface:

```
device# show running-config interface ethernet 0/1

!
interface ethernet 0/1
  ipv6 address 1111::10/64
  ipv6 neighbor 1111::1 00:04:96:eb:c4:51
  ipv6-router-advertisement
    hop-limit 133
    lifetime 200
    managed-config-flag
    mode disable-unsolicited-ra
    other-config
    enable
    interval 100 min 50
    prefix 100::1/63
    prefix 100::1/64
      advertisement
      autoconfiguration
      onlink
      preferred-lifetime 5000
      valid-lifetime 200
  !
  no shutdown
!
device#
```

prefix (IPv6 router advertisement configuration)

Use the **prefix** command to configure a prefix for IPv6 router advertisement and enter IPv6 router advertisement prefix configuration (*config-if-name-ipv6-router-advertisement-prefix-prefix-addr*) mode on an interface.

Syntax

```
prefix ipv6-prefix  
no prefix ipv6-prefix
```

Command Default

No prefixes for IPv6 router advertisement are configured on the current interface.

Parameters

ipv6-prefix

An individual prefix within an IPv6 router advertisement. Use IPv6 CIDR notation (*x.x.x.x/M*), where *M* is the prefix length in the range of 0–128 for IPv6 addresses.

For example, use an IPv6 address of 2001::100 and a prefix length of 64 to specify addresses in the range of 2001:0000:0000:0000:0000:0000:0000:0000 to 2001:0000:0000:0000:ffff:ffff:ffff:ffff.

You must specify a unicast address or a link local address. A link local address must have a prefix length of 64.

Modes

IPv6 router advertisement configuration (*config-if-name-ipv6-router-advertisement*)

Usage Guidelines

Router advertisements are ICMPv6 messages sent by an IPv6 enabled router that are used by hosts to detect routers on an attached link. This functionality is similar to the IPv4 ICMP Router Discovery protocol.

Neighbor discovery defines five ICMP packet types: router solicitation and advertisement, neighbor solicitation and advertisement, and redirect messages.

Router advertisements carry information such as the router link layer address, prefixes supported in the network, and the MTU setting. Router advertisements are periodic messages or are also sent in response to router solicitations. Because of the nature of its functionality, router advertisement is considered as a separate feature from neighbor discovery. Two messages are supported as part of this functionality: router solicitation and router advertisement.

Each configured interface sends a router advertisement message periodically to the all-nodes link-local multicast address (FF02::1). To avoid synchronization of advertisements from other routers, the time interval is randomized to send these messages between the configured maximum and minimum time periods.

Router advertisement includes the following information:

- Default router—Whether the sender can be a default router and, if so, the lifetime for the router
- List of prefixes for on-link addresses
- Auto-configuration information
- Network information—Hop limit and MTU setting to use

To remove a prefix for IPv6 router advertisement on an interface, use the **no** form of this command.

Examples

The following example configures a prefix of 100::1/64 for IPv6 router advertisement and enters IPv6 router advertisement prefix configuration (config-if-*name*-ipv6-router-advertisement-prefix-*prefix-addr*) mode on Ethernet interface 0/1:

```
device# configure terminal
device(config)# interface ethernet 0/1
device(config-if-eth-0/1)# ipv6-router-advertisement
device(config-if-eth-0/1-ipv6-router-advertisement)# prefix 100::1/64
device(config-if-ipv6-router-advertisement-prefix-100::1/64)#
```

The following example displays the configuration of Ethernet interface 0/1 that is running currently on the device. In this example, two prefixes are configured for IPv6 router advertisement on the interface:

```
device# show running-config interface ethernet 0/1

!
interface ethernet 0/1
  ipv6 address 1111::10/64
  ipv6 neighbor 1111::1 00:04:96:eb:c4:51
  ipv6-router-advertisement
    hop-limit 133
    lifetime 200
    managed-config-flag
    mode disable-unsolicited-ra
    other-config
    enable
    interval 100 min 50
    prefix 100::1/64
    prefix 100::1/63
    valid-lifetime 200
  !
  no shutdown
!
device#
```

prefix-independent-convergence (Extreme ONE OS Switching)

Use the **prefix-independent-convergence** command to accelerate the data-path convergence for an address family during failover conditions by installing a secondary path along with the primary path in the data plane for faster failover when the primary path's next hop is unreachable.

Syntax

```
prefix-independent-convergence  
no prefix-independent-convergence
```

Command Default

Prefix-independent convergence is disabled for routes configured under the specified BGP address family.

Parameters

This command has no keywords or variables.

Modes

VRF BGP address family configuration (*config-vrf-bgp-address-family-type*)

Usage Guidelines

Prefix-Independent Convergence (PIC) is an IETF standard that speeds up data path convergence during a failover by pre-computing and installing backup routes. Convergence is the time period that the network needs to update its routing tables and become stable after a change or failure. Because it is a path vector protocol, BGP typically has slower convergence than other protocols such as OSPF or EIGRP, especially in large networks. PIC helps to mitigate this issue.

This command enables PIC for a specific address family to achieve faster failover by installing the secondary path in the data plane when the primary path is unreachable.

Use the **no** form of this command to disable PIC for a specific address family.

Examples

The following example configures a routing process under a VRF instance named violet. This example creates and activates two address families. In this example, PIC is enabled for the address families:

```
device# configure terminal  
device(config)# vrf violet  
device(config-vrf-violet)# router bgp  
device(config-vrf-bgp)# address-family ipv4 unicast  
device(config-vrf-bgp-ipv4u)# prefix-independent-convergence
```

```
device(config-vrf-bgp-ipv4u)# activate
device(config-vrf-bgp-ipv4u)# exit
device(config-vrf-bgp)# address-family ipv6 unicast
device(config-vrf-bgp-ipv6u)# prefix-independent-convergence
device(config-vrf-bgp-ipv6u)# activate
device(config-vrf-bgp-ipv6u)#
```

The following example displays the configuration of a VRF instance named violet that is running currently on the device. In this example, the instance contains a routing process with two activated address families. Each address family uses PIC:

```
device# show running-config vrf violet

vrf violet
  router bgp
    address-family ipv4 unicast
      prefix-independent-convergence
      activate
    !
    address-family ipv6 unicast
      prefix-independent-convergence
      activate
    !
  !
!
device#
```

prefix-set

Use the **prefix-set** command to create an IPv4 or IPv6 prefix set and enter Routing policy prefix set configuration (config-route-policy-prefix-*name*) configuration mode.

Syntax

```
prefix-set prefix-set-name  
no prefix-set prefix-set-name
```

Command Default

No IPv4 or IPv6 prefix sets are configured on the device.

Parameters

prefix-set-name

Name of the prefix set. This name is used to reference the prefix set in match conditions. Use an arbitrary sequence of 1 to 64 letters, numerals, underscores, hyphens, or periods that starts with a letter.

Modes

Routing policy configuration (config-route-policy)

Usage Guidelines

This command creates a named prefix set and enters the mode for adding IPv4 or IPv6 prefixes to that prefix set.

To remove a prefix set from the routing policy configuration, use the **no** form of this command.

Examples

The following example creates a prefix set named `prefix1` that contains two IPv4 prefixes:

```
device# configure terminal  
device(config)# route-policy  
device(config-route-policy)# prefix-set prefix1  
device(config-route-policy-prefix-set-prefix1)# prefix 10.1.1.0/24 mask-range exact  
device(config-route-policy-prefix-set-prefix1)# prefix 20.1.1.0/24 mask-range 25..30  
device(config-route-policy-prefix-set-prefix1)#
```

The following example displays the routing policy configuration that is running currently on the device. In this example, a prefix set named prefix1 contains two IPv4 prefixes:

```
device# show running-config route-policy

route-policy
prefix-set prefix1
  prefix 10.1.1.0/24 mask-range exact
  prefix 20.1.1.0/24 mask-range 25..30
device#
```


prefix

Use the **prefix** command to add IPv4 or IPv6 prefixes to a prefix set.

Syntax

```
prefix { ipv4-address/prefix-length | ipv6-address/prefix-length } mask-range { exact | start..end }  
no prefix { ipv4-address | ipv6-address } mask-range-start { exact | start..end }
```

Command Default

The specified prefix set contains no IPv4 or IPv6 prefixes.

Parameters

ipv4-address/prefix-length

Range of IPv4 addresses. Use the *a.b.c.d* format for *ipv4-address*, and use an integer from 0 to 32 for the prefix. For example, 10.1.1.0/24. You must include the slash (/).

ipv6-address/prefix-length

Range of IPv6 addresses. Use the *x:x::x:x* format for *ipv6-address*, and use an integer from 0 to 128 for the prefix. For example, 2001::1/64. You must include the slash (/).

exact

Use an exact mask match.

start..end

Specifies that the prefix length of the route must fall with this range. For example, 25..30. For the start and end values, use an integer from 1 to 32.

The mask range cannot start with a value less than prefix length.

Modes

Routing policy prefix set configuration (*config-route-policy-prefix-name*)

Usage Guidelines

Use this command to add IPv4 or IPv6 prefixes to the specified prefix set on the device.

To remove a prefix from the specified prefix set, use the **no** form of this command.

Examples

The following example configures a prefix named prefix1 and adds two IPv4 prefixes:

```
device# configure terminal
device(config)# route-policy
device(config-route-policy)# prefix prefix1
device(config-route-policy-prefix-prefix1)# prefix 10.1.1.0/24 mask-range exact
device(config-route-policy-prefix-prefix1)# prefix 20.1.1.0/24 mask-range 25..30
device(config-route-policy-prefix-prefix1)#
```

The following example displays the routing policy configuration that is running currently on the device. In this example, a prefix named prefix1 contains two IPv4 prefixes:

```
device# show running-config route-policy

route-policy
prefix prefix1
    prefix 10.1.1.0/24 mask-range exact
    prefix 20.1.1.0/24 mask-range 25..30
device#
```

profile (BFD configuration)

Use the **profile (BFD configuration)** command to create a Bidirectional Forwarding Detection (BFD) profile and enter BFD profile configuration mode.

Syntax

```
profile profile-name  
no profile profile-name
```

Command Default

No BFD profiles exist on the device.

Parameters

profile-name

Name of the profile to be created. You can use any combination of 1 to 64 numerals, underscores, hyphens, periods, and uppercase or lowercase letters. The name must start with a letter or underscore. You cannot use a reserved keyword as the complete name.

Modes

BFD configuration (config-bfd)

Usage Guidelines

A BFD profile contains the BFD settings that are applied (or requested to be applied) to a routing protocol that is configured (registered) as a BGP client (meaning that it uses BFD for link failure detection instead of its own mechanism).

To remove a BFD profile, use the no form of this command.

Examples

The following example creates a BFD profile named profile1 and enters BFD profile configuration mode:

```
device# configure terminal  
device(config)# bfd  
device(config-bfd)# profile profile1  
device(config-bfd-profile-profile1)#
```

The following example shows the BFD profiles that are running currently on the device:

```
device# show running-config bfd  
  
bfd
```

```
profile profile1
device#
```

protocol lacp

Use the **protocol lacp** command to enable link aggregation control protocol (LACP) globally on a device.

Syntax

```
protocol lacp  
no protocol lacp
```

Command Default

LACP is not enabled.

Parameters

This command has no keywords or variables.

Modes

Global configuration (config)

Usage Guidelines

You must enable LACP globally on the device before performing any LAG configuration.

To disable LACP globally, use the no form of this command.

Examples

The following example enables LACP globally on the device:

```
device# configure terminal  
device(config)# protocol lacp  
device(config)#
```

The following example displays the LAG configuration that is running currently on the device. In this example, LACP is enabled globally, a LAG with a range of 10 is configured, and Ethernet interface 0/29 is added to the LAG in active mode:

```
device# show running-config lacp  
  
protocol lacp  
interface port-channel 10  
    no shutdown  
!  
interface ethernet 0/29  
    channel-group 10 mode active  
    no shutdown  
!  
device#
```

protocol lldp

Use the **protocol lldp** command to enable link layer discovery protocol (LLDP) globally on a device.

Syntax

```
protocol lldp  
no protocol lldp
```

Command Default

LLDP is not enabled.

Parameters

This command has no keywords or variables.

Modes

Global configuration (config)

Usage Guidelines

You must enable LLDP globally on the device before performing any LLDP configuration.

To disable LLDP globally, use the no form of this command.

Examples

The following example enables LLDP globally on the device:

```
device# configure terminal  
device(config)# protocol lldp  
device(config)#
```

The following example displays the LLDP configuration that is running currently on the device. In this example, LLDP is enabled globally. Also, LLDP is configured with packet transmission disabled, the global LLDP transmit interval is 100 seconds, the period that a receiving device holds information from your device before discarding it is 100 seconds, the LLDP management address is 1.1.1.1, and several optional TLV advertisements are suppressed:

```
device# show running-config lldp  
  
protocol lldp  
lldp  
no transmit  
hello-timer 100  
holdtime 100  
management-address 1.1.1.1
```

```
suppress-tlv management-address,port-description,system-name  
device#
```

qos (Extreme ONE OS Switching)

Use the **qos** command to enter QoS configuration (config-qos) mode.

Syntax

qos

Parameters

This command has no keywords or variables.

Modes

Global configuration (config)

Usage Guidelines

This command enters the mode for accessing the commands for configuring access QoS on a device. You use these commands to configure traffic class maps and remark maps and attach them to interfaces in the incoming and outgoing directions respectively. You also use these commands to enable trust CoS or trust DSCP in the incoming direction and enable remark CoS or remark DSCP in the outgoing direction.

This command does not appear in the running configuration unless there are sub-mode parameters configured.

Examples

The following example enters QoS configuration (config-qos) mode and configures access QoS on interface Ethernet 0/1 and its subinterface 100:

```
device# configure terminal
device(config)# qos
device(config-qos)# interface ethernet 0/1 sub-interface 100
device(config-qos-if-eth-0/1:1.100# input
device(config-qos-if-eth-0/1:1.100-input)# traffic-class-map ing_12_map
device(config-qos-if-eth-0/1:1.100-input)#
```


ram

Use the **ram** command to set a nondefault memory size for a virtual machine (VM).

Syntax

ram *size*

no ram

Command Default

The specified VM has 2048 MiB (mebibytes) of memory.

Parameters

size

Amount of memory in MiB. Use an integer from 1024 to 8192. One MiB is 1,048,576 bytes.

Modes

IAH VM configuration (config-iah-vm-*name*)

Usage Guidelines

The maximum amount of RAM allocated for IAH depends on the platform.

Use the **no** form of this command to restore the memory size to the default value. Note that configuration is not allowed when the VM is in the deployed state.

To display details about a specific VM, see [show iah vm](#) on page 1006.

Examples

The following example configures 2048 MiB of memory for a VM named Ubuntu1 on the device:

```
device# configure terminal
device(config)# iah
device(config-iah)# vm Ubuntu1
device(config-iah-vm-Ubuntu1)# ram 2048
device(config-iah-vm-Ubuntu1)#
```

The following example displays the IAH configuration that is running currently on the device. In this example, a VM named Ubuntu2 uses 4096 MiB of memory:

```
device# show running-config iah

iah
  vm Ubuntu2
    description Ubuntu demo vm
```

```
auto-boot true
cpus 4
ram 4096
disk sdb
    size 5000
    format qcow2
    bus sata
    file disk://iah/disk1.qcow2
!
!
!
device#
```

receive (LLDP configuration)

Use the **receive** command to enable global LLDP packet receive mode on the device.

Syntax

```
receive  
no receive
```

Command Default

Global LLDP packet receive mode on the device is enabled.

Parameters

This command has no keywords or variables.

Modes

LLDP configuration (config-lldp)

Usage Guidelines

Use this command to enable the LLDP agent on the device to receive information about LLDP capabilities and the LLDP current status of remote systems. When a device receives an LLDP packet, it indicates that another device sent an LLDP message to discover or update its neighboring devices. This can help you identify and manage network devices and troubleshoot network issues.

Use the no form of this command to disable global LLDP packet receive mode on the device.

To enable global LLDP packet transmit mode on the device, see [transmit \(LLDP configuration\)](#) on page 1161.

Examples

The following example enables global LLDP packet receive mode:

```
device# configure terminal  
device(config)# lldp  
device(config-lldp)# receive  
device(config-lldp)#
```

The following example displays the LLDP configuration that is running currently on the device. In this example, global LLDP packet receive mode is enabled:

```
device# show running-config lldp  
  
lldp
```

```
receive  
device#
```

remark-map (interface QoS configuration)

Use the **remark-map** command to attach a QoS map to an interface for remarking the QoS markings on egress (outgoing) packets.

Syntax

remark-map *qos-map-name*

no remark-map

Command Default

The specified interface has no attached QoS map for remarking in the egress direction.

Parameters

qos-map-name

Name of the QoS map to configure. Use any combination of 1 to 64 uppercase or lowercase alphabetic characters, numerals, underscores, hyphens, or periods. You must use an alphabetic character or an underscore as the first character. Do not use the "default" reserved keyword.

Modes

Ethernet interface QoS output configuration (config-qos-if-eth-*name*-output)

Port Channel interface QoS output configuration (config-qos-if-po-*name*-output)

Virtual Ethernet interface QoS output configuration (config-qos-if-ve-*name*-output)

Usage Guidelines

Differentiated Services Code Point (DSCP) and Class of Service (CoS) are the two methods that are available in the operating system for remarking egress traffic. Remarking is based on CoS for layer 2 LIFs unless you configure it to use DSCP. On layer 3 LIFs, remarking always uses DSCP.

To remove the QoS map from an interface, use the no form of this command.

Examples

The following example attaches a QoS map named egr_l2_map in the egress direction for remarking on subinterface 100 under Ethernet interface 0/1:

```
device# configure terminal
device(config)# qos
device(config-qos)# interface ethernet 0/1 sub-interface 100
device(config-qos-if-eth-0/1.100# output
```

```
device(config-qos-if-eth-0/1.100-output)# remark-map egr_l2_map
device(config-qos-if-eth-0/1.100-output)#
```

The following example displays the QoS configuration that is running currently on the device. In this example, a QoS map named `egr_l2_map` for remarking is attached in the egress direction to subinterface 100 under Ethernet interface 0/1:

```
device# show running-config qos

qos
  interface ethernet 0/1 sub-interface 100
    output
      remark-map egr_l2_map
device#
```

remark-map (QoS configuration)

Use the **remark-map** command to configure a QoS map for remarking the QoS markings on packets for remark mapping and enter QoS remark map configuration (config-qos-remark-map-*name*) mode.

Syntax

remark-map *qos-map-name*

no remark-map *qos-map-name*

Command Default

The device has no QoS map for remarking.

Parameters

qos-map-name

Name of the QoS map to configure. Use any combination of 1 to 64 uppercase or lowercase alphabetic characters, numerals, underscores, hyphens, or periods. You must use an alphabetic character or an underscore as the first character. Do not use the "default" reserved keyword.

Modes

QoS configuration (config-qos)

Usage Guidelines

Use this command to specify a QoS map name and enter the mode for configuring its traffic-class-to-DSCP entries or traffic-class-to-CoS entries for remarking.

Differentiated Services Code Point (DSCP) and Class of Service (CoS) are the two methods that are available in the operating system for remarking egress traffic. Remarking is based on CoS for layer 2 LIFs unless you configure it to use DSCP. On layer 3 LIFs, remarking always uses DSCP.

To remove a QoS map, use the no form of this command.

To define the traffic-class-to-DSCP entries or the traffic-class-to-CoS entries in the QoS map for remarking, see [traffic-class dscp \(QoS remark map configuration\)](#) on page 1155 or [traffic-class cos \(QoS remark map configuration\)](#) on page 1153.

Examples

The following example configures a QoS map named `egr_l2_map` for remarking and enters QoS remark map configuration (`config-qos-remark-map-name`) mode:

```
device# configure terminal
device(config)# qos
device(config-qos)# remark-map egr_l2_map
device(config-qos-remark-map-egr_l2_map)#
```

The following example displays the QoS configuration that is running currently on the device. In this example, a QoS map named `egr_l2_map` for remarking is configured:

```
device# show running-config qos

qos
  remark-map egr_l2_map
device#
```


remark (QoS configuration)

Use the **remark** command to enable an interface to remark the packet CoS or DSCP values on egress (outgoing) packets based on remark mapping.

Syntax

```
remark { cos | dscp }  
no remark { cos | dscp }
```

Command Default

Remarking on the specified interface is disabled.

Parameters

cos

Uses QoS Class of Service (CoS) remarking.

dscp

Uses QoS Differentiated Services Code Point (DSCP) remarking.

Modes

Ethernet interface QoS output configuration (config-qos-if-eth-*name*-output)

Port Channel interface QoS output configuration (config-qos-if-po-*name*-output)

Virtual Ethernet interface QoS output configuration (config-qos-if-ve-*name*-output)

Usage Guidelines

DSCP and CoS are the two methods that are available in the operating system for remarking egress traffic. Use this command to enable CoS remarking or DSCP remarking on layer 2 LIFs. If configured, CoS or DSCP is remarked based on internal traffic class values.

Note that on layer 3 LIFs, remarking always uses DSCP (even if you have enabled CoS remarking).

To restore the default behavior on the interface, use the no form of this command.

Examples

The following example enables DSCP remarking on subinterface 100 under Ethernet interface 0/1:

```
device# configure terminal  
device(config)# qos
```

```
device(config-qos)# interface ethernet 0/1 sub-interface 100
device(config-qos-if-eth-0/1.100# output
device(config-qos-if-eth-0/1.100-output)# remark dscp
device(config-qos-if-eth-0/1.100-output)#
```

The following example displays the QoS configuration that is running currently on the device. In this example, DSCP remarking is enabled on subinterface 100 under Ethernet interface 0/1:

```
device# show running-config qos

qos
  interface ethernet 0/1 sub-interface 100
    output
      remark dscp
device#
```

remote-as (BGP peer group configuration)

Use the **remote-as** command to set the autonomous system (AS) number for all neighbors in a specific BGP peer group within a Virtual Routing and Forwarding (VRF) instance.

Syntax

remote-as *as-number*

no remote-as *as-number*

Command Default

No remote AS number is set for the neighbors in the specified BGP peer group.

Parameters

as-number

AS number in either asplain or asdot notation:

- For asplain notation, specify an integer from 1 to 4294967295. For example, 6553700.
- For asdot notation, use the *high-order-16-bit-value-in-decimal.low-order-16-bit-value-in-decimal* format, where the high order and low order values are integers from 1 to 65535. For example, 100.100 (which equals 6553700 in asplain notation).

For details about asplain and asdot notation, see [Textual Representation of Autonomous System \(AS\) Numbers, RFC 5396](#).

Modes

VRF BGP peer group configuration (config-vrf-bgp-pg)

Usage Guidelines

Use this command to set the neighbors' AS number. This AS number is called the remote AS, and it is the same for all of the peer group members.

Use the **no** form of this command to remove the remote AS number configuration.



Caution

The **no** form of this command terminates all active BGP sessions that are using the currently configured remote AS number. Improper usage of this command might disrupt service.

Examples

The following example configures two BGP peer groups (group1 and group2) under a VRF instance named violet. In this example, the neighbors in group1 are assigned 300 as their remote AS, and the neighbors in group2 are assigned 100.200 as their remote AS. The two AS numbers are entered using the asplain format and the asdot format respectively:

```
device# configure terminal
device(config)# vrf violet
device(config-vrf-violet)# router bgp
device(config-vrf-bgp)# peer-group group1
device(config-vrf-bgp-pg)# remote-as 300
device(config-vrf-bgp-pg)# exit
device(config-vrf-bgp)# peer-group group2
device(config-vrf-bgp-pg)# remote-as 100.200
device(config-vrf-bgp-pg)#
```

The following example displays the configuration of a VRF instance named violet that is running currently on the device. In this example, the neighbors in group1 are assigned 300 (which was configured using the asplain format) as their remote AS, and the neighbors in group2 are assigned 100.200 (which was configured using the asdot format) as their remote AS:

```
device# show running-config vrf violet

vrf violet
!
  router bgp
    peer-group group1
      remote-as 300
    peer-group group2
      remote-as 100.200
!
device#
```

resilient-hash (VRF configuration)

Use the **resilient-hash** command to enable the resilient hashing feature for ECMP in a specific Virtual Routing and Forwarding (VRF) instance.

Syntax

```
resilient-hash  
no resilient-hash
```

Command Default

Resilient hashing for ECMP is disabled in the VRF.

Parameters

This command has no keywords or variables.

Modes

VRF configuration (*config-vrf-vrf-name*)

Usage Guidelines

If an interface becomes unavailable, it is restored by ensuring that a previously hashed 5-tuple combination does not appear in a different interface.

Use the **no** form of this command to disable resilient hashing in the current VRF instance.

Examples

The following example enables resilient hashing for ECMP in the VRF:

```
device# configure terminal  
device(config)# vrf vrf1  
kvm-x86-64(config-vrf-vrf1)# resilient-hash  
kvm-x86-64(config-vrf-vrf1)#
```

The following example shows the configuration for a VRF named `vrf1` that is running currently on the device. In this example, resilient hashing for ECMP is enabled in the VRF:

```
device# show running-configuration vrf vrf1  
  
vrf vrf1  
    resilient-hash  
!  
device#
```

restart-time (BGP graceful restart configuration)

Use the **restart-time** command to set the restart period for BGP graceful restart for neighbors in the BGP routing process within a Virtual Routing and Forwarding (VRF) instance.

Syntax

restart-time *seconds*

no restart-time *seconds*

Parameters

seconds

Duration of the graceful restart period. Use an integer from 1 to 3600 seconds. This is the amount of time that each neighbor waits to delete stale routes before a BGP open message is received. The restart time must equal or exceed the stale route time (for details, see [stale-route-time \(BGP graceful restart configuration\)](#) on page 1115).

Modes

VRF BGP graceful restart configuration (config-vrf-bgp-gr)

Usage Guidelines

Graceful restart allows a device undergoing a restart to inform its adjacent neighbors of its condition. This command sets the duration of the BGP graceful restart period.

To restore the graceful start period to the default value, use the **no** form of this command.

Graceful restart (that is, a restart time combined with a stale route time) and helper-only mode are mutually exclusive. If graceful restart is enabled, you must use the **no graceful-restart** command to disable it (see [graceful-restart \(BGP configuration\)](#) on page 626) before enabling helper-only mode. Conversely, if helper-only mode is enabled, you must disable it before enabling graceful restart. For details, see [helper-only \(BGP graceful restart configuration\)](#) on page 634.

Examples

The following example configures a routing process within a VRF instance named violet. This example enables graceful restart on its neighbors and configures a restart time and a stale route time of 100 seconds and 500 seconds respectively:

```
device# configure terminal
device(config)# vrf violet
```

```
device(config-vrf-violet)# router bgp
device(config-vrf-bgp)# graceful-restart
device(config-vrf-bgp-gr)# restart-time 100
device(config-vrf-bgp-gr)# stale-route-time 500
device(config-vrf-bgp-gr)#
```

The following example configures a routing process within a VRF instance named violet. This example enables graceful restart on its neighbors and configures a restart time and a stale route time of 100 seconds and 500 seconds respectively:

```
device# show running-config vrf violet

vrf violet
  router bgp
    graceful-restart
      restart-time 100
      stale-route-time 500
!
device#
```

retain-route-target-all (BGP L2VPN EVPN address family configuration)

Use the **retain-route-target-all** command to retain the routes of any route targets for the BGP L2VPN EVPN address family within a routing process in a VRF instance.

Syntax

```
retain-route-target-all  
no retain-route-target-all
```

Command Default

Route target retention is disabled for the BGP L2VPN EVPN address family.

Parameters

This command has no keywords or variables.

Modes

VRF BGP L2VPN EVPN address family configuration (config-vrf-bgp-l2vpn-evpn)

Usage Guidelines

This command configures a route reflector to accept all route targets for the L2VPN EVPN address family. This bypasses the route target check for the address family in the case of VPN route processing.

Use the **no** form of this command to disable the ability to bypass the route target check for the L2VPN EVPN address family.

Examples

The following example configures a routing process under a VRF instance named violet. This example creates an L2VPN EVPN address family in the routing process and bypasses the route target check:

```
device# configure terminal  
device(config)# vrf violet  
device(config-vrf-violet)# router bgp  
device(config-vrf-default-bgp)# address-family l2vpn evpn  
device(config-vrf-bgp-l2vpn-evpn)# retain-route-target-all  
device(config-vrf-bgp-l2vpn-evpn)#
```

The following example displays the configuration of a VRF instance named violet that is running currently on the device. In this example, the route target check is bypassed for the L2VPN EVPN address family:

```
device# show running-config vrf violet  
  
vrf violet
```



```
!  
router bgp  
    address-family l2vpn evpn  
        retain-route-target-all  
!  
device#
```

role (MLAG configuration)

Use the **role (MLAG configuration)** command to specify the type of role for multichassis link aggregation (MLAG).

Syntax

```
role { auto | primary | backup }  
no role
```

Command Default

The MLAG role type is set automatically as either primary or backup when the MLAG session comes up.

Parameters

auto

Sets the MLAG role to automatic. The MLAG role is selected as primary or backup only when the MLAG session comes up.

With the **auto** role type, the role is selected based on the primary keepalive source IP address. Otherwise, the node with the highest primary keepalive source IP address is selected as primary, and the lowest will be the backup node.

If one peer is configured using **auto**, the other node should also be configured using **auto**.

primary

Sets the MLAG role as primary.

backup

Sets the MLAG role as backup.

Modes

MLAG configuration (config-mlag)

Usage Guidelines

You can use the **show running-config mlag** command to display the MLAG configuration that is running currently on a device.

Use the no form of this command to revert the type of role for MLAG so that it is set automatically.

Examples

The following example sets the MLAG role as primary:

```
device# configure terminal
device(config)# mlag
device(config-mlag)# role primary
device(config-mlag)#
```

The following example displays the MLAG configuration that is running currently on a device. In this example, MLAG is configured with a backup role:

```
device# show running-config mlag

mlag
  interface port-channel 71 id 101
  interface port-channel 81 id auto
  interface ethernet 0/32:4 id 301
  peer Peer1
    keepalive kp1
      type primary
      address-type manual
      destination 56.1.1.7
      source-interface port-channel 56
    !
    keepalive kp2
      type secondary
      address-type auto
      destination 10.38.60.29
      source-interface management 0
    !
  !
!
device#
```

The following example displays the MLAG configuration on a device. In this example, MLAG is configured with a primary role:

```
device# show mlag

Role           : PRIMARY
MAC            : A0:01:A0:02:A0:03
Bring up delay : 90 seconds
Keepalive Delay : 100 ms
Tx Interval    : 1000 ms
Multiplier     : 4
No. of Peers   : 1
No. of Interfaces : 1

Peer Info:
=====
Peer           State
-----
mypeer1        UP

Interfaces Info:
=====
Id      Interface      Peer           Local/Remote State Exceptions
-----
0       port-channel 1   mypeer1        DOWN / UNKNOWN  Interface Down
device#
```

route-distinguisher (EVPN configuration)

Use the **route-distinguisher** command to configure the route distinguisher for a specific Ethernet VPN (EVPN) instance.

Syntax

```
route-distinguisher { as-number:admin-number-1 | ipv4-address:admin-  
number-2 }
```

```
no route-distinguisher { as-number:admin-number-1 | ipv4-address:admin-  
number-2 }
```

Command Default

A route distinguisher is derived automatically for the specified EVPN instance.

Parameters

as-number:admin-number-1

Autonomous system (AS) number and administrative number. You must enter the colon (:).

- *as-number* is an assigned AS number. Use an integer that is either 2 bytes or 4 bytes long. The range is 1 to 4294967295. You cannot use a reserved AS number.
- *admin-number-1* is an integer from 1 to 4294967295. If *as-number* is 2-byte, then *admin-number-1* must be 4-byte (an integer from 1 to 4294967295). If *as-number* is 4-byte, then *admin-number-1* must be 2-byte (an integer from 1 to 65535).

ipv4-address:admin-number-2

IPv4 address and administrative number. You must enter the colon (:).

- *ipv4-address* is a valid Class A, Class B, or Class C address. Use the *a.b.c.d* format. You cannot use other addresses such as Class D, Class E, multidestination, loopback, or the default address.
- *admin-number-2* is an integer from 1 to 65535.

Modes

Ethernet VPN instance configuration (*config-evpn-name*)

Usage Guidelines

Use this command to specify an identifier that corresponds to a route. This identifier determines the routes that belong to the EVPN.

If you do not specify a route distinguisher, the default is a Type 1 route distinguisher with a value that uses the format *VTEP-IP-address:bridge-domain-ID*. For example, 1.1.1.2:1000, where 1.1.1.2 is the VXLAN Tunnel End Point (VTEP) IP address, and 1000 is the bridge domain ID.

Each routing instance must use its own route distinguisher. The route distinguisher places bounds around an EVPN so that the same IP address prefixes can be used in multiple EVPNs without overlap. You must create a unique route distinguisher for every EVPN to ensure that unique prefixes are generated.

Improper usage of this command might disrupt service. You should change an existing route distinguisher only during a maintenance window. To do so, deactivate the routing instance, then change the route distinguisher, and then re-activate the routing instance.

Use the **no** form of this command to remove the route distinguisher from the specified EVPN instance.

Examples

The following example uses an AS number to configure a route distinguisher for an EVPN instance named `tenant_1`:

```
device# configure terminal
device(config)# vrf default-vrf
device(config-vrf-default-vrf)# evpn-instance tenant_1
device(config-evpn-tenant_1)# route-distinguisher 65530:10000
device(config-evpn-tenant_1)#
```

The following example uses an IPv4 address to configure a route distinguisher for an EVPN instance named `tenant_2`:

```
device# configure terminal
device(config)# vrf default-vrf
device(config-vrf-default-vrf)# evpn-instance tenant_2
device(config-evpn-tenant_2)# route-distinguisher 152.100.1.100:10001
device(config-evpn-tenant_2)#
```

The following example shows the EVPN instances (and their configured route distinguishers) that are running currently on the device:

```
device# show running-config

...
vrf default-vrf
  evpn-instance tenant_1
    route-distinguisher 65530:10000
  evpn-instance tenant_2
    route-distinguisher 152.100.1.100:10000
...
device#
```

route-distinguisher (VRF configuration)

Use the **route-distinguisher** command to configure the route distinguisher for a specific Virtual Routing and Forwarding (VRF) instance.

Syntax

```
route-distinguisher { as-number:admin-number-1 | ipv4-address:admin-number-2 }
```

```
no route-distinguisher { as-number:admin-number-1 | ipv4-address:admin-number-2 }
```

Command Default

The route distinguisher is derived automatically for the specified VRF instance.

Parameters

as-number:admin-number-1

Autonomous system (AS) number and administrative number. You must enter the colon (:).

- *as-number* is an assigned AS number. Use an integer that is either 2 bytes or 4 bytes long. The range is 1 to 4294967295. You cannot use a reserved AS number.
- *admin-number-1* is an integer from 1 to 4294967295. If *as-number* is 2-byte, then *admin-number-1* must be 4-byte (an integer from 1 to 4294967295). If *as-number* is 4-byte, then *admin-number-1* must be 2-byte (an integer from 1 to 65535).

ipv4-address:admin-number-2

IPv4 address and administrative number. You must enter the colon (:).

- *ipv4-address* is a valid Class A, Class B, or Class C address. Use the *a.b.c.d* format. You cannot use other addresses such as Class D, Class E, multidestination, loopback, or the default address.
- *admin-number-2* is an integer from 1 to 65535.

Modes

VRF BGP configuration (config-vrf-bgp)

Usage Guidelines

Use this command to specify an identifier that corresponds to a route. This identifier determines the routes that belong to the VRF instance.

This command applies only to user-configured VRFs, but not default-vrf (the default VRF).

Each VRF instance must use its own route distinguisher. The route distinguisher places bounds around a VRF instance so that the same IP address prefixes can be used in multiple VRF instances without overlap. You must create a unique route distinguisher for every VRF instance to ensure that unique prefixes are generated.

Improper usage of this command might disrupt service. You should change an existing route distinguisher only during a maintenance window. To do so, deactivate the VRF instance, then change the route distinguisher, and then re-activate the VRF instance.

Use the **no** form of this command to remove the route distinguisher from the specified VRF instance.

Examples

The following example uses an AS number to configure a route distinguisher for a VRF instance named `tenant_1`:

```
device# configure terminal
device(config)# vrf tenant_1
device(config-vrf-tenant_1)# router bgp
device(config-vrf-bgp)# route-distinguisher 65530:10000
device(config-vrf-bgp)#
```

The following example uses an IPv4 address to configure a route distinguisher for an a VRF instance named `tenant_2`:

```
device# configure terminal
device(config)# vrf tenant_2
device(config-vrf-tenant_2)# router bgp
device(config-vrf-bgp)# route-distinguisher 152.100.1.100:10001
device(config-vrf-bgp)#
```

The following example shows the VRF instances (and their configured route distinguishers) that are running currently on the device:

```
device# show running-configuration

...
vrf tenant_1
  router bgp
    route-distinguisher 65530:10000
vrf tenant_2
  router bgp
    route-distinguisher 152.100.1.100:10000
...
device#
```

route-policy

Use the **route-policy** command to enter Routing policy configuration (config-route-policy) mode.

Syntax

route-policy

Parameters

This command has no keywords or variables.

Modes

Global configuration (config)

Usage Guidelines

This mode lets you access the commands for configuring the routing policy, which controls (filters) the routes that a routing protocol imports into the routing table and the routes that a routing protocol exports from the routing table. A routing policy also lets you set the information associated with a route as it is being imported into or exported from the routing table.

Examples

The following example enters Routing policy configuration (config-route-policy) mode:

```
device# configure terminal
device(config)# route-policy
device(config-route-policy)#
```


route-reflector-client

Use the **route-reflector-client** command to configure the members of an internal BGP (iBGP) peer group within a Virtual Routing and Forwarding (VRF) instance as route reflector clients.

Syntax

```
route-reflector-client  
no route-reflector-client
```

Command Default

The members of the specified BGP peer group are not configured as route reflector clients.

Parameters

This command has no keywords or variables.

Modes

VRF BGP peer group configuration (config-vrf-bgp-pg)

Usage Guidelines

Use this command to enable the members of an iBGP peer group to be iBGP route reflector clients. You combine these clients with an iBGP route reflector (also called a route reflector server) to form a BGP route reflection cluster.

Use the **no** form of this command to remove the route reflector client capability from the iBGP peer group.

Examples

The following example configures a BGP peer group named `group1` under a VRF instance named `violet`. This example configures members of `group1` to act as route reflector clients:

```
device# configure terminal  
device(config)# vrf violet  
device(config-vrf-violet)# router bgp  
device(config-vrf-bgp)# peer-group group1  
device(config-vrf-bgp-pg)# route-reflector-client  
device(config-vrf-bgp-pg)#
```

The following example displays the configuration of a VRF instance named violet that is running currently on the device. In this example, the members of the group1 peer group are configured as route reflector clients:

```
device# show running-config vrf violet

vrf violet
!
  router bgp
    peer-group group1
      route-reflector-client
!
device#
```

route-target (Extreme ONE OS Switching)

Use the **route-target** command to configure the route target for a specific EVPN instance.

Syntax

```
route-target { import | export | both } { as-number:admin-number-1 |  
  ipv4-address:admin-number-2 } [ ignore-as ]  
  
no route-target { import | export | both } { as-number:admin-number-1 |  
  ipv4-address:admin-number-2 } [ ignore-as ]
```

Command Default

No route target is configured for the specified EVPN instance.

Parameters

import | **export** | **both**

Specifies whether the route target is used for import processing, export processing, or both.

as-number:admin-number-1

Autonomous system (AS) number and administrative number. You must enter the colon (:).

- *as-number* is an assigned AS number. Use an integer that is either 2 bytes or 4 bytes long. The range is 1 to 4294967295. You cannot use a reserved AS number.
- *admin-number-1* is an integer from 1 to 4294967295. If *as-number* is 2-byte, then *admin-number-1* must be 4-byte (an integer from 1 to 4294967295). If *as-number* is 4-byte, then *admin-number-1* must be 2-byte (an integer from 1 to 65535).

ipv4-address:admin-number-2

IPv4 address and administrative number. You must enter the colon (:).

- *ipv4-address* is a valid Class A, Class B, or Class C address. Use the *a.b.c.d* format. You cannot use other addresses such as Class D, Class E, multidestination, loopback, or the default address.
- *admin-number-2* is an integer from 1 to 65535.

ignore-as

(Optional) Disables the AS number check if the route target is used for import processing. Only the administrative number is used as a match.

Modes

Ethernet VPN instance configuration (config-evpn-*name*)

VRF BGP L2VPN EVPN NVE address family configuration (*config-vrf-bgp-l2vpn-evpn-nve-name-address-family*)

Usage Guidelines

Use this command to configure route targets to control route transmission and reception. A route target is an extended community attribute of BGP.

Route targets are used to import or export routes (or both). When routes are imported to a VRF routing table, the routes with the route target that match any import route target in the table are added to the table. When routes are exported from a VRF routing table, the export route target marks the VRF routes.

You do not need to deactivate the EVPN instance before changing an existing route target.

Use the **no** form of this command to remove the route target from the specified EVPN instance.

Examples

The following example uses an AS number to configure a route target for an EVPN instance named *tenant_1*:

```
device# configure terminal
device(config)# vrf default-vrf
device(config-vrf-default-vrf)# evpn-instance tenant_1
device(config-evpn-tenant_1)# route-target both 65530:10000
device(config-evpn-tenant_1)#
```

The following example uses an IPv4 address to configure a route target for an EVPN instance named *tenant_2*:

```
device# configure terminal
device(config)# vrf default-vrf
device(config-vrf-default-vrf)# evpn-instance tenant_2
device(config-evpn-tenant_2)# route-target both 152.100.1.100:10001
device(config-evpn-tenant_2)#
```

The following example shows the EVPN instances (and their configured route targets) that are running currently on the device:

```
device# show running-config

...
vrf default-vrf
    evpn-instance tenant_1
        route-target both 65530:10000
    evpn-instance tenant_2
        route-target both 152.100.1.100:10000
...
device#
```

router bgp

Use the **router bgp** command to configure a BGP routing process within a Virtual Routing and Forwarding (VRF) instance and enter BGP configuration mode.

Syntax

router bgp

no router bgp

Command Default

No BGP routing process is enabled within the specified VRF instance.

Parameters

This command has no keywords or variables.

Modes

VRF configuration (*config-vrf-vrf-name*)

Usage Guidelines

This mode enables a BGP routing process within a VRF instance and lets you access the commands for configuring BGP for that instance.

Use the no form of this command to clear all BGP configuration under the current VRF instance.



Caution

Improper usage of the **no** form of this command might disrupt service.

Examples

The following example configures a VRF instance named red on the device and enters BGP configuration mode (*config-vrf-bgp*):

```
device# configure terminal
device(config)# vrf red
device(config-vrf-red)# router bgp
device(config-vrf-bgp)
```

The following example displays the VRF instances (including their BGP configurations) that are running currently on the device. In this example, a single instance named red is running currently:

```
device# show running-config vrf

vrf red
  router bgp
```

```
local-as 100
router-id 1.1.1.1
device#
```

router-id (BGP configuration)

Use the **router-id** command to configure a custom BGP router ID within a Virtual Routing and Forwarding (VRF) instance.

Syntax

```
router-id ipv4-address  
no router-id ipv4-address
```

Command Default

No router ID for the specified VRF instance is specified.

Parameters

ipv4-address

Router ID in IPv4 format. Use the *a.b.c.d* format. For example, 100.100.100.1.

Modes

VRF BGP configuration (config-vrf-bgp)

Usage Guidelines

Use this command to override the default router ID for BGP. This AS number is called the BGP router ID, and it is one per router BGP instance.

BGP is designed to refuse a session with itself because of the router ID check. This command lets you assign a separate BGP router ID per VRF, which enables VRF-to-VRF peering in BGP on the same device.

Use the **no** form of this command to revert to the default router ID.



Caution

The **no** form of this command terminates all active BGP sessions that are using the configured AS number. Improper usage of this command might disrupt service.

Examples

The following example overrides the default BGP router ID for a VRF instance named blue:

```
device# configure terminal  
device(config)# vrf blue  
device(config-vrf-blue)# router bgp  
device(config-vrf-bgp)# router-id 100.100.100.1  
device(config-vrf-bgp)#
```

The following example displays the configuration of a VRF instance named blue that is running currently on the device. In this example, the VRF is using a custom router ID of 100.100.100.1:

```
device# show running-config vrf blue

vrf blue
  member ethernet 0/15
  address-family ipv4 unicast
  address-family ipv6 unicast
  router bgp
    local-as 100
    router-id 100.100.100.1
    address-family ipv6 unicast
      activate
    !
  peer-group cust-1
    remote-as 200
    enable-bfd
    address-family ipv6 unicast
      activate
    !
  neighbor 10:1:1::2
  !
!
device#
```


send-default-route (BGP address family configuration)

Use the **send-default-route** command to override exclusion of the default route from the BGP routing table when redistribution of routes between protocols is enabled for the BGP IPv4 or IPv6 address families within a routing process in a VRF instance.

Syntax

send-default-route

no send-default-route

Command Default

The default route is not redistributed into the BGP routing table when redistribution of routes between protocols is enabled for the address family.

Parameters

This command has no keywords or variables.

Modes

VRF BGP address family configuration (*config-vrf-bgp-address-family-type*)

Usage Guidelines

If you have enabled the redistribution of routes from one protocol to another for the specified address family, BGP checks for the default route in the redistribution slot and does not add it to the BGP routing table. But if this default route from redistribution must be used, enter the **send-default-route** command for the specified address family to ensure its redistribution to the BGP routing table (thereby advertising it to neighbors).

This command applies only to IPv4 unicast or IPv6 unicast address families.

Use the **no** form of this command to restore the default functionality.

Examples

The following example configures a routing process under a VRF instance named violet. This example creates an IPv4 unicast address family in the routing process and overrides the exclusion of the default route from the BGP routing table when redistribution of routes between protocols is enabled:

```
device# configure terminal
device(config)# vrf violet
device(config-vrf-violet)# router bgp
device(config-vrf-violet-bgp)# address-family ipv4 unicast
```

```
device(config-vrf-bgp-l2vpn-evpn)# send-default-route
device(config-vrf-bgp-l2vpn-evpn)#
```

The following example displays the configuration of a VRF instance named violet that is running currently on the device. This example overrides the exclusion of the default route from the BGP routing table when redistribution of routes between protocols is enabled for the IPv4 unicast address family and the IPv6 unicast address family:

```
device# show running-config vrf violet

vrf violet

  table-connections
    src-protocol static dst-protocol bgp address-family ipv4
    src-protocol static dst-protocol bgp address-family ipv6
  !
  static-route 20.1.1.1/32 30.1.1.101 enable-bfd profile default
  static-route 200:20:1:1::/64 200:30:1:1::65 enable-bfd profile computes
  static-route 20.1.1.8/32 30.1.8.101 enable-bfd profile default
  static-route 200:20:1:5::/64 200:30:1:5::65 enable-bfd profile computes
  static-route 26.1.1.0/24 25.1.1.10
  static-route 26:1:1:100::/64 200:25:1:1::100
  router bgp
    local-as 10001
    router-id 2.2.2.2
    graceful-restart
    use-multiple-paths ebgp maximum-paths 8
    address-family ipv4 unicast
      graceful-restart
      send-default-route
      activate
    !
    address-family ipv6 unicast
      graceful-restart
      send-default-route
      activate
    !
  !
device#
```

set-as-path-prepend (routing policy statement BGP actions configuration)

Use the **set-as-path-prepend** command to configure a routing policy statement to use a BGP action that prepends an autonomous system (AS) number to the AS path from which a route originates.

Syntax

```
set-as-path-prepend [ as-number ] count  
no set-as-path-prepend
```

Command Default

The routing policy statement contains no BGP action to prepend an AS number to the AS path.

Parameters

as-number

(Optional) AS number to prepend to the AS path. Use an integer from 0 to 4294967295.

count

Number of times to prepend the AS number to the AS path.

Modes

Routing policy statement actions configuration (config-route-policy-policy-*name*-stmt-*number*-actions)

Usage Guidelines

The AS path is classified as a well-known mandatory BGP attribute, which means that it is recognized by all BGP devices, passed to all BGP peers, and present in all update messages. You use this command to configure a routing policy statement to add an AS number to the beginning of an AS path to make it less preferable to BGP. This occurs after the actual AS number from which the route originates is added to the path.

AS path prepending ensures proper return path selection and distributes the traffic load for multihomed customers. If you use multihoming, you might prefer a specific path for incoming traffic to reach your network. For example, a lower-cost connection or a much slower connection to use as a backup if your primary connection is down. AS path prepending is an easy way to influence inbound routing to your AS.

The local AS number is usually prepended multiple times to increase the AS path length.

To remove the BGP action to prepend an AS number from a routing policy statement, use the **no** form of this command.

Examples

The following example creates a routing policy named `map1` that contains a policy statement with sequence number 10 with a BGP action to prepend AS number 110 five times to the AS path:

```
device# configure terminal
device(config)# route-policy
device(config-route-policy)# policy map1
device(config-route-policy-policy-map1)# statement 10
device(config-route-policy-policy-map1-stmt-10)# actions
device(config-route-policy-policy-map1-stmt-10-actions)# bgp-actions
device(config-route-policy-policy-map1-stmt-10-actions-bgp-actions)# set-as-path-prepend
110 5
device(config-route-policy-policy-map1-stmt-10-actions-bgp-actions)#
```

The following example displays the routing policy configuration that is running currently on the device. In this example, a routing policy named `map1` contains a policy statement with sequence number 10 containing a BGP action to prepend AS number 110 five times to the AS path:

```
device# show running-config route-policy

route-policy
  policy map1
    statement 10
      actions
        bgp-actions
          set-as-path-prepend 110 5
device#
```

set-community (routing policy statement BGP actions configuration)

Use the **set-community** command to configure a routing policy statement to use a BGP action that adds, removes, or replaces BGP community sets in a route.

Syntax

```
set-community { add | remove | replace } community-set-name  
no set-community
```

Command Default

The routing policy statement contains no BGP action to add, remove, or replace BGP community sets.

Parameters

add

Add all members of the specified community set to the community attribute in the route.

remove

Delete members of the specified community set from the community attribute in the route.

replace

Replace the community attribute in the route with the members of the specified community set.

community-set-name

Name of the community set to use. Use 1 to 255 characters.

Modes

Routing policy statement actions configuration (config-route-policy-policy-*name-stmt-number*-actions)

Usage Guidelines

The list of communities in a route is classified as an optional transitive BGP attribute, which means that it is possibly recognized by BGP devices and passed to BGP peers. Optional transitive attributes that are not recognized are marked as partial when sent. You use this command to configure a routing policy statement with actions that manipulate the communities in the route. For example, you can override the communities in the route with a community that you specify.

You can create multiple routing policy statements with each one containing a different set of actions in a routing policy.

To remove a BGP action to add, remove, or replace a BGP community set in a routing policy statement, use the **no** form of this command.

Examples

The following example creates a routing policy named `map1` that contains a policy statement with sequence number 10 with a BGP action to add a community set named `comm1` to the route:

```
device# configure terminal
device(config)# route-policy
device(config-route-policy)# policy map1
device(config-route-policy-policy-map1)# statement 10
device(config-route-policy-policy-map1-stmt-10)# actions
device(config-route-policy-policy-map1-stmt-10-actions)# bgp-actions
device(config-route-policy-policy-map1-stmt-10-actions-bgp-actions)# set-community add
comm1
device(config-route-policy-policy-map1-stmt-10-actions-bgp-actions)#
```

The following example displays the routing policy configuration that is running currently on the device. In this example, a routing policy named `map1` contains a policy statement with sequence number 10 containing a BGP action to add a community set named `comm1` to the route:

```
device# show running-config route-policy

route-policy
  policy map1
    statement 10
      actions
        bgp-actions
          set-community add comm1
device#
```

set-ext-community (routing policy statement BGP actions configuration)

Use the **set-ext-community** command to configure a routing policy statement to use a BGP action that adds, removes, or replaces BGP extended community sets in a route.

Syntax

```
set-ext-community { add | remove | replace } community-set-name  
no set-ext-community
```

Command Default

The routing policy statement contains no BGP action to add, remove, or replace BGP extended community sets.

Parameters

add

Add the specified extended community set to the communities in the route.

remove

Delete the specified extended community set from the communities in the route.

replace

Replace any extended communities that were in the route with the specified extended community set.

community-set-name

Name of the extended community set to use. Use 1 to 255 characters.

Modes

Routing policy statement actions configuration (config-route-policy-policy-*name*-stmt-*number*-actions)

Usage Guidelines

The list of extended communities in a route is classified as an optional transitive BGP attribute, which means that it is possibly recognized by BGP devices and passed to BGP peers. Optional transitive attributes that are not recognized are marked as partial when sent. You use this command to configure a routing policy statement with actions that manipulate the extended communities in the route. For example, you can override the communities in the route with an extended community that you specify.

User-configured routing policies let you control the flow of routing information. Routing policies can decide which routes should be accepted or advertised by dynamic routing protocols. For example, if do not want a protocol to import all routes into the

routing table or export all the active routes it learns, or if you want to control the routes advertised via route redistribution.

Each statement in a routing policy can have only one BGP actions container. This container can contain multiple actions.

To remove a BGP action to add, remove, or replace a BGP extended community set in a routing policy statement, use the **no** form of this command.

Examples

The following example creates a routing policy named `map1` that contains a policy statement with sequence number 10 with a BGP action to add an extended community set named `extcomm2` to the route:

```
device# configure terminal
device(config)# route-policy
device(config-route-policy)# policy map1
device(config-route-policy-policy-map1)# statement 10
device(config-route-policy-policy-map1-stmt-10)# actions
device(config-route-policy-policy-map1-stmt-10-actions)# bgp-actions
device(config-route-policy-policy-map1-stmt-10-actions-bgp-actions)# set-ext-community
add extcomm2
device(config-route-policy-policy-map1-stmt-10-actions-bgp-actions)#
```

The following example displays the routing policy configuration that is running currently on the device. In this example, a routing policy named `map1` contains a policy statement with sequence number 10 containing a BGP action to add an extended community set named `extcomm2` to the route:

```
device# show running-config route-policy

route-policy
  policy map1
    statement 10
      actions
        bgp-actions
          set-ext-community add extcomm2
device#
```


set-local-pref (routing policy statement BGP actions configuration)

Use the **set-local-pref** command to configure a routing policy statement to use a BGP action that sets a local preference value in a route update.

Syntax

```
set-local-pref local-preference-value  
no set-local-pref
```

Command Default

The routing policy statement contains no BGP action to set a local preference value.

Parameters

local-preference-value
Local preference value. Use an integer from 1 to 65535.

Modes

Routing policy statement actions configuration (config-route-policy-policy-*name*-stmt-*number*-actions)

Usage Guidelines

The local preference value is classified as a well-known discretionary BGP attribute, which means that is recognized by all BGP devices, passed to all BGP peers, and optionally included in the update message. You use this command to configure a routing policy statement with actions that manipulate the local preference value in the route.

You can create multiple routing policy statements with each one containing a different set of actions in a routing policy.

To remove a BGP action to set a local preference value from a routing policy statement, use the **no** form of this command.

Examples

The following example creates a routing policy named map1 that contains a policy statement with sequence number 10 with a BGP action to set a local preference value of 50:

```
device# configure terminal  
device(config)# route-policy  
device(config-route-policy)# policy map1  
device(config-route-policy-policy-map1)# statement 10  
device(config-route-policy-policy-map1-stmt-10)# actions
```

```
device(config-route-policy-policy-map1-stmt-10-actions)# bgp-actions
device(config-route-policy-policy-map1-stmt-10-actions-bgp-actions)# set-local-pref 50
device(config-route-policy-policy-map1-stmt-10-actions-bgp-actions)#
```

The following example displays the routing policy configuration that is running currently on the device. In this example, a routing policy named map1 contains a policy statement with sequence number 10 containing a BGP action to set a local preference value of 50:

```
device# show running-config route-policy

route-policy
  policy map1
    statement 10
      actions
        bgp-actions
          set-local-pref 50
device#
```

set-med (routing policy statement BGP actions configuration)

Use the **set-med** command to configure a routing policy statement to use a BGP action that sets a multiple exit discriminator (MED) value in a route update.

Syntax

```
set-med med-value  
no set-med
```

Command Default

The routing policy statement contains no BGP action to set a MED value.

Parameters

med-value
MED value. Use an integer from 1 to 65535.

Modes

Routing policy statement actions configuration (config-route-policy-policy-*name*-stmt-*number*-actions)

Usage Guidelines

The MED value is classified as an optional nontransitive BGP attribute, which means that it is possibly recognized by BGP devices but not passed to BGP peers. You use this command to configure a routing policy statement with actions that manipulate the MED value in the route.

You can create multiple routing policy statements with each one containing a different set of actions in a routing policy.

To remove a BGP action to set a MED value from a routing policy statement, use the **no** form of this command.

Examples

The following example creates a routing policy named `map1` that contains a policy statement with sequence number 10 with a BGP action to set a MED value of 20:

```
device# configure terminal  
device(config)# route-policy  
device(config-route-policy)# policy map1  
device(config-route-policy-policy-map1)# statement 10  
device(config-route-policy-policy-map1-stmt-10)# actions  
device(config-route-policy-policy-map1-stmt-10-actions)# bgp-actions  
device(config-route-policy-policy-map1-stmt-10-actions-bgp-actions)# set-med 20  
device(config-route-policy-policy-map1-stmt-10-actions-bgp-actions)#
```

The following example displays the routing policy configuration that is running currently on the device. In this example, a routing policy named map1 contains a policy statement with sequence number 10 containing a BGP action to set a MED value of 20:

```
device# show running-config route-policy

route-policy
  policy map1
    statement 10
      actions
        bgp-actions
          set-med 20
device#
```

set-next-hop (routing policy statement BGP actions configuration)

Use the **set-next-hop** command to configure a routing policy statement to use a BGP action that sets the next-hop IP address in a route.

Syntax

```
set-next-hop { ip-address | self }  
no set-next-hop
```

Command Default

The routing policy statement contains no BGP action to set a next-hop IP address.

Parameters

ip-address

IPv4 address or IPv6 address of the next hop.

- For an IPv4 address, use IPv4 notation (*A.B.C.D*). For example, use an IPv4 address of 10.0.0.0.
- For an IPv6 address, use the *x::x::x::x* format. For example, 1001::1.

self

Uses the local peer IP address of the device as the next hop.

Modes

Routing policy statement actions configuration (config-route-policy-policy-*name*-stmt-*number*-actions)

Usage Guidelines

User-configured routing policies let you control the flow of routing information. Routing policies can decide which routes should be accepted or advertised by dynamic routing protocols. For example, if do not want a protocol to import all routes into the routing table or export all the active routes it learns, or if you want to control the routes advertised via route redistribution.

Each statement in a routing policy can have only one BGP actions container. This container can contain multiple actions.

To remove a BGP action to set an IP address in a routing policy statement, use the **no** form of this command.

Examples

The following example creates a routing policy named `map1` that contains a policy statement with sequence number 10 with a BGP action to add a next-hop address of 1.1.1.1 to the route:

```
device# configure terminal
device(config)# route-policy
device(config-route-policy)# policy map1
device(config-route-policy-policy-map1)# statement 10
device(config-route-policy-policy-map1-stmt-10)# actions
device(config-route-policy-policy-map1-stmt-10-actions)# bgp-actions
device(config-route-policy-policy-map1-stmt-10-actions-bgp-actions)# set-next-hop 1.1.1.1
device(config-route-policy-policy-map1-stmt-10-actions-bgp-actions)#
```

The following example displays the routing policy configuration that is running currently on the device. In this example, a routing policy named `map1` contains a policy statement with sequence number 10 containing a BGP action to add a next-hop address of 1.1.1.1 to the route:

```
device# show running-config route-policy

route-policy
  policy map1
    statement 10
      actions
        bgp-actions
          set-next-hop 1.1.1.1
device#
```

set-route-origin (routing policy statement BGP actions configuration)

Use the **set-route-origin** command to configure a routing policy statement to use a BGP action that sets the route origin of the NLRI in a route to IGP or EGP.

Syntax

```
set-route-origin { igp | egp | incomplete }  
no set-route-origin
```

Command Default

The routing policy statement contains no BGP action to set the route origin of the Network Layer Reachability Information (NLRI).

Parameters

igp

The route origin of the NLRI is Interior Gateway Protocol (IGP).

egp

The route origin of the NLRI is Exterior Gateway Protocol (EGP).

incomplete

The route origin of the NLRI is not IGP or EGP.

Modes

Routing policy statement actions configuration (config-route-policy-policy-*name*-stmt-*number*-actions)

Usage Guidelines

User-configured routing policies let you control the flow of routing information. Routing policies can decide which routes should be accepted or advertised by dynamic routing protocols. For example, if do not want a protocol to import all routes into the routing table or export all the active routes it learns, or if you want to control the routes advertised via route redistribution.

Each statement in a routing policy can have only one BGP actions container. This container can contain multiple actions.

To remove a BGP action to set the route origin of the NLRI in a routing policy statement, use the **no** form of this command.

Examples

The following example creates a routing policy named `map1` that contains a policy statement with sequence number 10 with a BGP action to set the route origin of the NLRI to IGP to the route:

```
device# configure terminal
device(config)# route-policy
device(config-route-policy)# policy map1
device(config-route-policy-policy-map1)# statement 10
device(config-route-policy-policy-map1-stmt-10)# actions
device(config-route-policy-policy-map1-stmt-10-actions)# bgp-actions
device(config-route-policy-policy-map1-stmt-10-actions-bgp-actions)# set-route-origin igp
device(config-route-policy-policy-map1-stmt-10-actions-bgp-actions)#
```

The following example displays the routing policy configuration that is running currently on the device. In this example, a routing policy named `map1` contains a policy statement with sequence number 10 containing a BGP action to set the route origin of the NLRI to IGP to the route:

```
device# show running-config route-policy

route-policy
  policy map1
    statement 10
      actions
        bgp-actions
          set-route-origin igp
device#
```


show access-list

Use the **show access-list** command to display IPv4, IPv6, or MAC access control lists (ACLs).

Syntax

```
show { ipv4 | ipv6 | mac } access-list { acl-name | all }
```

Parameters

ipv4

Displays IPv4 ACLs.

ipv6

Displays IPv6 ACLs.

mac

Displays MAC ACLs.

acl-name

Name of the ACL to be displayed. You can use any combination of 1 to 64 uppercase or lowercase alphabetic characters, numerals, underscores, hyphens, or periods. You must use an alphabetic character or an underscore as the first character.

all

Displays all IPv4 ACLs, IPv4 ACLs, or MAC ACLs.

Modes

EXEC (#)

Examples

The following example displays an ACL named v4acl:

```
device# show ipv4 access-list v4acl

ipv4 access-list v4acl
seq 10 permit ipv4 1.1.1.1 255.255.255.255 2.2.2.2 255.255.255.255 dport 44 count ( 0
Packets, 0 Bytes)
device#
```

The following example displays an ACL named v6acl:

```
device# show ipv6 access-list v6acl

ipv6 access-list v6acl
seq 10 permit ipv6 1::1 ffff:ffff:ffff:ffff:ffff:ffff:ffff:ffff 2::2
ffff:ffff:ffff:ffff:ffff:ffff:ffff:ffff sport 22 log ( 0 Packets, 0 Bytes)
device#
```

The following example displays an ACL named macacl:

```
device# show mac access-list macacl

mac access-list mac1
seq 10 permit 11:22:33:44:00:00 ff:ff:ff:ff:ff:00 any log count ( 0 Packets, 0 Bytes)
device#
```

The following example displays the list of all existing ACLs that are running currently on the device:

```
device# show running-config access-list

ipv4 access-list v4acl
ipv6 access-list v6acl
mac access-list macacl
device#
```

show anycast-gateway (IPv4 and IPv6)

Use the **show anycast-gateway** command to display the configuration details of the anycast gateway on a device.

Syntax

```
show { ipv4 | ipv6 } anycast-gateway [ interface ve interface-name | vrf vrf-name ]
```

Parameters

interface ve *interface-name*

Name range of a specific Virtual Ethernet (VE) interface. Use an integer or one or more comma-separated ranges of integers from 1 to 8192. For example, 2 or 2-3,5,7-9.

vrf *vrf-name*

Name of a specific VRF instance.

Modes

EXEC (#)

Output

The **show anycast-gateway** command displays the following information:

Output field	Description
L3 Interface	Type (physical or VE) and name of the Layer 3 interface that assigned to the VRF.
IP-Address	IP address that is configured on the interface.
Vrf	Name of the VRF on which the interface is configured.
Status	Indicates whether the interface is functioning. The possible statuses are UP and DOWN.

Examples

The following example shows the configuration details of the IPv4 anycast gateway on the device:

```
device# show ipv4 anycast-gateway

IPv4 Anycast Gateway Mac: 02:e0:52:00:01:00

L3 Interface      IP-Address      Vrf              Status
=====
Ve 100            10.10.10.1      default-vrf      UP
```

```
Ve 200          200.200.200.1    default-vrf    UP
device#
```

The following example shows the configuration details of the IPv6 anycast gateway on the device:

```
device# show ipv6 anycast-gateway

IPv6 Anycast Gateway Mac: 02:e0:52:00:02:00

L3 Interface      IP-Address      Vrf              Status
=====
Ve 100            2001::100       default-vrf      UP
Ve 200            2002::100       default-vrf      UP
device#
```

The following example shows the configuration details of the IPv4 anycast gateway within a VRF named vrf1 on the device:

```
device# show ipv4 anycast-gateway vrf vrf1

IPv4 Anycast Gateway Mac: 02:e0:52:00:01:00

L3 Interface      IP-Address      Status
=====
Ve 300            210.210.210.1   UP
device#
```

show bfd neighbors application (Extreme ONE OS Switching)

Use the **show bfd neighbors application** command to display a list of BFD sessions for the selected client application or static route that is registered with BFD.

Syntax

```
show bfd neighbors application { bgp | mlag | static-ip } details
```

Parameters

bgp

Include only BGP clients.

mlag

Include only MLAG clients.

static-ip

Include only static route clients.

details

Display detailed BFD session information (such as protocol parameters and timers) for each neighbor.

Modes

EXEC (#)

Usage Guidelines

Use this command to display a line-by-line listing of existing BFD adjacencies (sessions). You can filter the sessions according to the type of BFD client application (such as BGP) or a static route.

A BFD session is established based on the needs of the client application or static route that will use it. BFD will detect a link failure, but it requires one or more registered client applications or a static route to take action (such as failover to another route) on the failure. The client application or static route determines the need for BFD and the addresses to use (BFD has no discovery mechanism).

Output

The `bgp` keyword of the `show bfd neighbors application` command displays the following information:

Table 11: show bfd neighbors application Field Descriptions

Output field	Description
OurAddr	IPv4 or IPv6 address of the local device (the system on which this command is entered) that establishes BFD sessions to its neighbors.
NeighAddr	IPv4 or IPv6 address of the neighboring device with which BFD sessions are established.
State	Present condition of BFD sessions with the neighboring device.
Local State	
Received State	
Diag	Diagnostic message for the local device's reason for the last change in session state. Following are the possible diagnostic messages and the corresponding numeric codes (which are not included in the display) for these messages as specified by RFC 5880: • NO_DIAGNOSTIC (0) • CONTROL_DETECTION_TIME_EXPIRED (1) • ECHO_FUNCTION_FAILED (2) • NEIGHBOR_SIGNED_SESSION_DOWN (3) • FORWARDING_PLANE_RESET (4) • PATH_DOWN (5) • CONCATENATED_PATH_DOWN (6) • ADMINISTRATIVELY_DOWN (7) • REVERSE_CONCATENATED_PATH_DOWN (8) This message lets remote systems determine the reason that the previous session failed, for example.
Local MinTxInt(ms)	Desired minimum amount of time, in milliseconds, between control packet transmissions on the local system.
Local MinRxInt(ms)	Desired minimum amount of time, in milliseconds, between control packet receptions on the local system.
Local Multiplier	Desired multiplier on the local system by which the time interval is multiplied to determine the detection time for the transmitting system.
Received MinTxInt(ms)	Desired minimum amount of time, in milliseconds, between control packet transmissions on the remote system.
Received MinRxInt(ms)	Desired minimum amount of time, in milliseconds, between control packet receptions on the remote system.
Received Multiplier	Desired multiplier on the remote system by which the time interval is multiplied to determine the detection time for the transmitting system.

Table 11: show bfd neighbors application Field Descriptions (continued)

Output field	Description
Rx Count	Number of BFD control packets that have been received from the BFD neighbor.
Tx Count	Number of BFD control packets that have been sent to the BFD neighbor.
LD/RD	Local discriminator (LD) and remote discriminator (RD) values that are used for each BFD session: • LD is a unique, nonzero discriminator value generated by the transmitting system used to demultiplex received BFD packets to the proper session when multiple BFD sessions exist between the same pair of systems. This value is sent in the My Discriminator field in the BFD control packet. • RD is the discriminator value received from the corresponding remote system. This field shows the received value of LD (or zero if that value is unknown). The value sent in the My Discriminator field in the BFD control packet from the sender is echoed back in the Your Discriminator field of packets sent from the remote end. After the remote end echoes back the local discriminator, subsequent received packets are demultiplexed using the Your Discriminator field only (which means that, among other things, the source address field can change or the interface over which the packets are received can change, but the packets remain associated with the correct session). Note that the value of Your Discriminator is zero during demultiplexing of the initial packets. Also note that a system might change its discriminator during a session without affecting the session state, because only that system uses its discriminator for demultiplexing (by having the other system reflect it back).
Heard from Remote	Indicates whether the remote BFD neighbor was heard.
Profile(Applied)	Profile containing BFD settings that are applied to the routing protocols that are in use. This field also displays the type of routing protocol (such as BGP or static routes) to which the profile is applied.
Profile(Requested)	Profile containing BFD settings that are requested to be applied to the routing protocols that are registered with BFD.
Current Registered Protocols	Routing protocols (client routing applications) that are registered with BFD (such as BGP).
Uptime	Number of hours, minutes, and seconds that the session has been in the UP state.

Examples

The following example displays a list of BFD sessions for the BGP client application on the device:

```
device# show bfd neighbors application bgp details
```

OurAddr	NeighAddr	State	Intf

1.10.0.1	1.10.0.2	UP	Ve 100

```

Local State: UP                      Diag: NO_DIAGNOSTIC
Received State: UP                  Diag: NO_DIAGNOSTIC
Local  MinTxInt(ms) :300            MinRxInt(ms):300      Multiplier:4
Received MinTxInt(ms) :300          MinRxInt(ms):300      Multiplier:3
Rx Count: 2104                      Tx Count: 5822
LD/RD: 2/2                          Heard from Remote: Yes
Session ID: 1
Profile(Applied): profile1
Profile(Also Requested): profile1 (BGP), profile2 (STATIC-ROUTES)
Current Registered Protocols: BGP
Uptime: 1h14m22.576449224s
device#
```

The following example displays a list of BFD sessions for the static IP client application on the device:

```
device# show bfd neighbors application static-ip details
device#
```


show bfd neighbors dest-ip (Extreme ONE OS Switching)

Use the **show bfd neighbors dest-ip** command to display information about BFD sessions that match a specific destination IP address between the local device and its neighbors.

Syntax

```
show bfd neighbors dest-ip { dest-ipv4-addr | dest-ipv6-addr } { vrf
  { vrf-name | all } | interface { ethernet interface-name | port-
channel port-channel-range | ve interface-name | loopback interface-
name } } details
```

Parameters

```
{ dest-ipv4-addr | dest-ipv6-addr }
```

Clear BFD sessions by destination IPv4 address or IPv6 address:

- *dest-ipv4-addr* filters sessions by IPv4 address. Use the *a.b.c.d* format.
- *dest-ipv6-addr* filters sessions by IPv6 address. Use the *x::x::x::x* format.

```
vrf { vrf-name | all }
```

- *vrf-name* filters sessions by VRF instance name. This name must start with an alphabetic character or an underscore followed by an arbitrary sequence of alphabetic or numeric characters, underscores, hyphens, or dots. The name cannot exceed 64 characters. Do not use the profile reserved keyword.
- **all** clears counters of BFD sessions that match the specified destination IP address across all VRF instances.

```
interface
```

Filter sessions by interface name.

```
ethernet interface-name
```

Filter sessions by Ethernet interface name. Use the *slot/port* format or the *slot/port:breakout* format to specify an interface or one or more comma-separated ranges of interfaces respectively. For example, 0/1 or 1/1-3,5,2/7-9,10:1-4 respectively.

```
port-channel port-channel-range
```

Filter sessions by port channel number. Use an integer or one or more comma-separated ranges of integers from 1 to 255. For example, 1 or 1-3,5,7-9.

```
ve interface-name
```

Filter sessions by Virtual Ethernet (VE) interface name. Use an integer or one or more comma-separated ranges of integers from 1 to 8192. For example, 2 or 2-3,5,7-9.

```
loopback interface-name
```

Filter sessions by loopback interface name. Use an integer or one or more comma-separated ranges of integers from 1 to 255. For example, 3 or 3-4,5,7-9.

Modes

EXEC (#)

Examples

The following example displays information about BFD sessions that match destination IPv4 address 10.0.0.1 on a VRF named violet:

```
device# show bfd neighbors dest-ip 10.0.0.1 vrf violet
device#
```

The following example displays information about BFD sessions that match destination IPv4 address 10.0.0.1 on all VRFs:

```
device# show bfd neighbors vrf all

OurAddr      NeighAddr    State      Intf
Profile
-----
66.66.66.2   66.66.66.1   DOWN      Eth 0/4
Applied: abc
Requested: abc(BGP),

4004::2      4004::1      UP        Eth 0/4
Applied: abc
Requested: abc(BGP),

33.33.33.2   33.33.33.1   UP        Eth 0/4
Applied: abc
Requested: abc(BGP),
duta#
device#
```

show bfd neighbors interface (Extreme ONE OS Switching)

Use the **show bfd neighbors interface** command to display information about BFD sessions that match a specific interface between the local device and its neighbors.

Syntax

```
show bfd neighbors interface { ethernet interface-name | port-channel port-channel-range | ve interface-name | loopback interface-name } details
```

Parameters

ethernet *interface-name*

Filter sessions by Ethernet interface name. Use the *slot/port* format or the *slot/port:breakout* format to specify an interface or one or more comma-separated ranges of interfaces respectively. For example, 0/1 or 1/1-3,5,2/7-9,10:1-4 respectively.

port-channel *port-channel-range*

Filter sessions by port channel number. Use an integer or one or more comma-separated ranges of integers from 1 to 255. For example, 1 or 1-3,5,7-9.

ve *interface-name*

Filter sessions by Virtual Ethernet (VE) interface name. Use an integer or one or more comma-separated ranges of integers from 1 to 8192. For example, 2 or 2-3,5,7-9.

loopback *interface-name*

Filter sessions by loopback interface name. Use an integer or one or more comma-separated ranges of integers from 1 to 255. For example, 3 or 3-4,5,7-9.

Modes

EXEC (#)

Examples

The following example displays information about BFD sessions that match destination IPv4 address 10.0.0.1 on a VRF named violet:

```
device# show bfd neighbors interface ethernet 0/1
device#
```

show bfd neighbors

Use the **show bfd neighbors** command to display a summary of BFD session information between the local device and its neighbors.

Syntax

```
show bfd neighbors [ vrf { vrf-name | all } ]
```

Parameters

vrf *vrf-name*

(Optional) Displays information for a specific VRF.

all

(Optional) Displays information for all VRFs.

Modes

EXEC (#)

Output

The **show bfd neighbors** command displays the following information:

Output field	Description
OurAddr	IPv4 or IPv6 address of the local device from which BFD established the session with the remote peers.
Profile	Profile containing BFD settings that are applied (or requested to be applied) to the routing protocol or static route. This field also displays the type of routing protocol (such as BGP or OSPF) to which the profile is applied.
NeighAddr	IPv4 or IPv6 address of the remote peer with which the BFD session is established.

Output field	Description
State	Current state of the three-way handshake for BFD session establishment or session teardown with the remote peer. Following are the possible states and the corresponding numeric codes (which are not included in the display) for these messages as specified by RFC 5880: • A.DOWN (0)—The session is forced administratively down indefinitely. This causes the remote system to enter DOWN state and remain there until the local system exits this state. • DOWN (1)—The session is just created, never came up, or is down because the control detection time expired in an established BFD session. For example, if the minimum transmit interval is set to 100 ms, and the multiplier is 3, then no response after 300 ms makes the system go down. A session remains in DOWN state until the remote system indicates that it agrees that the session is down by sending a BFD control packet with the State field set to anything other than UP. If that packet signals DOWN state, the session advances to INIT state; if that packet signals INIT state, the session advances to UP state. Semantically, DOWN state indicates that the forwarding path is unavailable and that appropriate actions (such as failover to a different route) should be taken by the applications monitoring the state of the BFD session. A system might hold a session in DOWN state indefinitely (by refusing to advance the session state). This can be done for operational or administrative reasons, among others. • INIT (2)—The remote system is communicating, and the local system wants to bring the session up at the start of a BFD handshake, but the remote system does not yet realize it. A session remains in this state until either a BFD Control Packet is received that signals the INIT or UP state (in which case the session advances to UP state) or the Detection Time expires, meaning that communication with the remote system was lost (in which case the session advances to DOWN state). • UP (3)—The BFD session was successfully established, which implies that connectivity between the systems is working. The session stays in UP state until connectivity fails or the session is taken down administratively. If either the remote system signals DOWN state or the Detection Time expires, the session advances to DOWN state. • UNKNOWN (no numeric code)—The session state cannot be determined.
Intf	Name of the interface on which the BFD neighbor is reachable.

Examples

The following example displays a summary of BFD session information on the device:

```
device# show bfd neighbors
```

OurAddr Profile		NeighAddr	State	Intf
1.10.0.1		1.10.0.2	UP	Ve 100
profile1	(Applied)			
profile1 (BGP), profile2 (OSPF)	(Requested)			
1.10.1.1		1.10.1.2	UP	Ve 101
profile1	(Applied)			
profile1 (BGP)	(Requested)			
1.10.2.1		1.10.2.2	UP	Ve 102
profile1	(Applied)			
profile1 (BGP)	(Requested)			

```
device#
```

The following example displays a summary of BFD session information for a VRF named test on the device:

```
device# show bfd neighbors vrf test
```

OurAddr Profile	NeighAddr	State	Intf
10.1.1.20	10.1.1.21	UP	Eth 0/1:1
Applied: default			
Requested: default [if-prof] (STATIC),			

```
device#
```

The following example displays a summary of BFD session information for all VRFs on the device:

```
device# show bfd neighbors vrf all
```

OurAddr Profile	NeighAddr	State	Intf
66.66.66.2	66.66.66.1	DOWN	Eth 0/4
Applied: abc			
Requested: abc (BGP),			
4004::2	4004::1	UP	Eth 0/4
Applied: abc			
Requested: abc (BGP),			
33.33.33.2	33.33.33.1	UP	Eth 0/4
Applied: abc			
Requested: abc (BGP),			

```
device#
```

show bfd profile

Use the **show bfd profile** command to display the parameters of a specific BFD profile or all BFD profiles that are configured on a device.

Syntax

```
show bfd profile { profile-name | all }
```

Parameters

profile-name

Name of a specific BFD profile for which to display parameters. You can use any combination of 1 to 64 numerals, underscores, hyphens, periods, and uppercase or lowercase letters. The name must start with a letter or underscore. You cannot use the profile reserved keyword as the complete name.

all

Display the parameters of all BFD profiles that are configured on the device.

Modes

EXEC (#)

Output

The **show bfd profile** command displays the following information:

Output field	Description
BFD Profile	Profile containing the BFD settings that are applied to the routing protocols or static routes that are in use.
MinTx	Minimum transmit interval that is associated with the profile. This is the minimum transmission interval (less jitter) that this system wants to use to send BFD control packets. This value is specified by the <i>min-tx-milliseconds</i> parameter of the BFD interval (BFD profile configuration) command.

Output field	Description
MinRx	Minimum receive interval that is associated with the profile. This is the minimum interval that this system is capable of receiving BFD control packets. This value is specified by the min-rx <i>min-rx-milliseconds</i> parameter of the interval (BFD profile configuration) command.
Mult	Detection time multiplier that is associated with the profile. The remote transmission interval will be multiplied by this value to determine the connection loss detection timer. For example, when the local system has a multiplier of 3 and the remote system has a minimum transmit interval of 50 ms, the local system will detect failures only after 150 ms without receiving BFD control packets. This value is specified by the multiplier <i>multiple</i> parameter of the interval (BFD profile configuration) command.

Examples

The following example displays the parameters that are defined in a BFD profile named `profile1`:

```
device# show bfd profile profile1
```

BFD Profile	MinTx	MinRx	Mult
profile1	400	400	4

```
device#
```


show bfd

Use the **show bfd** command to display a summary of BFD interface related information.

Syntax

```
show bfd
```

Parameters

This command has no keywords or variables.

Modes

EXEC (#)

Output

The **show bfd** command displays the following information:

Output field	Description
BFD State	
Version	
Supported Protocols	Client routing protocols (such as static-ip, OSPF2, BGP, and OSPF) that are configured on the device under which BFD is enabled.
All Sessions	Total number of BFD sessions of all states (A.DOWN, DOWN, INIT, UP, or UNKNOWN) as seen by the device.
Current	Number of active BFD sessions on the device. These are sessions that are in the UP state.
Max Allowed	Maximum number of BFD sessions that are supported on the device.
Max Exceeded Count	Number of BFD attempts to set up a session beyond the maximum.
Port Sessions	Name of the interface on which BFD is enabled.
Admin State	Administrative state of the interface on which BFD is enabled.
Sessions	Number of active BFD sessions on the interface.

Examples

The following example display a summary of BFD interface related information on the device:

```
device# show bfd

BFD State: ENABLED, Version: 1
Supported Protocols: bgp, mlag, static-ip
All Sessions: Current: 0
Max Allowed: 1000 Max Exceeded Count :0
Port Sessions          Admin State          Sessions
-----
Ve 100                  Enabled              0
device#
```

show bgp vrf l2vpn-evpn

Use the **show bgp vrf l2vpn-evpn** command to display information about an L2VPN EVPN instance or tunnel within a specific VRF.

Syntax

```
show bgp vrf { default-vrf | vrf-name } l2vpn-evpn instance instance-name [ bridge-domain ID ]
```

```
show bgp vrf { default-vrf | vrf-name } l2vpn-evpn tunnels nvo instance-name [ nve endpoint-name ] [ tunnel-destination { ipv4-addr | ipv6-addr } ]
```

Parameters

vrf-name

VRF instance name. This name must start with an alphabetic character or an underscore followed by an arbitrary sequence of alphabetic or numeric characters, underscores, hyphens, or dots. The name cannot exceed 64 characters. Do not use the **profile** reserved keyword.

instance *instance-name*

Name of the EVPN instance for which to display EVPN instance-specific information. You can use any combination of 1 to 64 uppercase or lowercase alphabetic characters, numerals, underscores, hyphens, or periods. This name must be unique across all VRF instances on the device. You must use an alphabetic character or an underscore as the first character.

bridge-domain *ID*

(Optional) ID number of a bridge domain. Use an integer from 1 to 8192.

tunnels nvo *instance-name*

Name of the Network Virtualization Overlay (NVO) instance for which to display NVO-specific information. This is the overlay network used to deliver the Layer 2 and Layer 3 VPN services. You can use any combination of 1 to 64 uppercase or lowercase alphabetic characters, numerals, underscores, hyphens, or periods. You must use an alphabetic character or an underscore as the first character.

nve *endpoint-name*

(Optional) Name of the Network Virtualization Edge (NVE) endpoint (also called an NVE node) of the specified NVO instance. This is the provider edge node within the NVO environment that encapsulates tenant traffic into the overlay network. You can use any combination of 1 to 64 uppercase or lowercase alphabetic characters, numerals, underscores, hyphens, or periods. You must use an alphabetic character or an underscore as the first character.

tunnel-destination { *ipv4-addr* | *ipv6-addr* }

(Optional) IPv4 address or IPv6 address of the tunnel destination.

- For an IPv4 address, use the *a.b.c.d* format.

- For an IPv6 address, use the `x::x::x::x` format.

Modes

EXEC (#)

Examples

show bgp vrf neighbor

Use the **show bgp vrf neighbor** command to display information about a BGP neighbor within a specific VRF.

Syntax

```
show bgp vrf { default-vrf | vrf-name } neighbor { ipv4-addr | ipv6-addr } [ policy | capability | timers | graceful-restart | bfd ]

show bgp vrf { default-vrf | vrf-name } neighbor { ipv4-addr | ipv6-addr } { received-routes | advertised-routes } { ipv4-unicast | ipv6-unicast } [ detail | ipv4-addr | ipv6-addr ]

show bgp vrf { default-vrf | vrf-name } neighbor { ipv4-addr | ipv6-addr } { received-routes | advertised-routes } l2vpn-evpn [ route-type { imr | mac | arp | nd | prefix } [ detail ] ]
```

Parameters

vrf-name

VRF instance name. This name must start with an alphabetic character or an underscore followed by an arbitrary sequence of alphabetic or numeric characters, underscores, hyphens, or dots. The name cannot exceed 64 characters. Do not use the **profile** reserved keyword.

neighbor { *ipv4-addr* | *ipv6-addr* }

Address of the neighbor for which to display information:

- For an IPv4 address, use the *a.b.c.d* format.
- For an IPv6 address, use the *x:x::x:x* format.

policy

(Optional) Displays neighbor policies.

capability

(Optional) Displays neighbor capabilities such as graceful restart, multiprotocol BGP (MPGBP), and route refresh.

timers

(Optional) Displays neighbor timers such as the BGP connection retry interval, BGP hold time, and BGP keepalive interval.

graceful-restart

(Optional) Displays neighbor graceful restart details such as helper-only mode, restart time, and stale route time.

bfd

(Optional) Displays neighbor bidirectional forwarding detection (BFD) details.

received-routes

Displays routes received from this neighbor.

advertised-routes

Displays routes advertised to this neighbor.

ipv4-unicast

Displays IPv4 address family and unicast address family routes information.

ipv6-unicast

Displays IPv6 address family and unicast address family routes information.

detail | *ipv4-addr* | *ipv6-addr*

(Optional) Displays detailed route information for all routes or route information for a specific address and prefix.

- For an IPv4 address, use IPv4 CIDR notation (*A.B.C.D/M*), where *M* is the prefix length in the range of 0–32 for IPv4 addresses. For example, use an IPv4 address of 10.0.0.0 and a prefix length of 8 to specify addresses in the range of 10.0.0.0 to 10.255.255.255.
- For an IPv6 address, use IPv6 CIDR notation (*X::X:X/M*), where *M* is the prefix length in the range of 0–128 for IPv6 addresses. For example, use an IPv6 address of 2001::100 and a prefix length of 64 to specify addresses in the range of 2001:0000:0000:0000:0000:0000:0000 to 2001:0000:0000:0000:ffff:ffff:ffff:ffff.

route-type { **imr** | **mac** | **arp** | **nd** | **prefix** }

(Optional) Displays specific route type information:

- **imr**—Displays IMR route-specific information.
- **mac**—Displays MAC route-specific information.
- **arp**—Displays ARP route-specific information.
- **nd**—Displays ND route-specific information.
- **prefix**—Displays IPv4 prefix route-specific information.

detail

(Optional) Displays detailed route information.

Modes

EXEC (#)

Examples

The following example displays information about neighbor 21.1.1.1 within a VRF named default-vrf:

```
device# show bgp vrf default-vrf neighbor 21.1.1.1

Peer: 21.1.1.1, Remote Port: 40870, Peer-AS: 65535
Localhost: 21.1.1.0, Local Port: 179, Local-AS: 64512, Local-AS-Forced: -
Peer Router ID: 1.1.1.5, Local Router ID: 2.2.2.2, VRF: default-vrf
Route Reflector Client: No, Cluster-ID: -
Fast External Failover: false
State: Established, Uptime: 1s, Dynamic Peer: false, Peer Group: Spine-v4
```

```

Last Notification Time      : 2024-12-20 08:03:19 +0000 UTC
Last Connection Reset Reason: Open Message Error Bad Peer AS
Send Community: No, Send Extended Community: No

```

```

Capabilities  Negotiated  Advertised  Received
=====
MPBGP        true        true        true
AS-4Byte     true        true        true
Route-Refresh true        true        true
Cisco-RR     true        true        true

```

```

MPBGP:
=====
  Advertised: IPv4Ucast,L2VpnEvpn
  Received  : IPv4Ucast,L2VpnEvpn
  Negotiated: IPv4Ucast,L2VpnEvpn

```

```

Timers          Interval(Sec)    Method          Remaining(Sec)
=====
Connect Retry   31                      Default         -
Hold Timer      30                      Negotiated      30
Keepalive Timer 10                      Derived         10
Start Timer     5                       System          -
Update Interval 0                      Default         -
device#

```

The following example displays information about IPv4 unicast routes advertised to neighbor 10.2.5.1 within a VRF named default-vrf:

```

device# show bgp vrf default-vrf neighbor 10.2.5.1 advertised-routes ipv4-unicast

Seq#      Prefix                Nexthop          Status
=====
1         5.5.5.5/32            10.2.5.2         VALID
2         10.1.5.0/24           10.2.5.2         VALID
3         10.2.5.0/24           10.2.5.2         VALID
device#

```

The following example displays information about BGP capabilities of neighbor 10.2.5.1 within a VRF named default-vrf:

```

device# show bgp vrf default-vrf neighbor 10.2.5.1 capability

Capabilities  Advertised  Received
=====
MPBGP        Yes         Yes
Route Refresh Yes         Yes
device#

```

The following example displays information about timers for neighbor 10.2.5.1 within a VRF named default-vrf:

```

device# show bgp vrf default-vrf neighbor 10.2.5.1 timers

Timers          Interval(Sec)    Method          Remaining(Sec)
=====
Connect Retry   31                      Default         -
Hold Timer      180                 Negotiated      155
Keepalive Timer 60                  Default         35
Start Timer     5                   System          -
Update Interval 0                  Default         -
device#

```

The following example displays information about Ethernet VPN (EVPN) routes received from neighbor 10.2.5.1 within a VRF named default-vrf:

```
device# show bgp vrf default-vrf neighbor 10.2.5.1 received-routes l2vpn-evpn

VRF Name: default-vrf
Total number of routes: 0
device#
```

The following example displays detailed information about Ethernet (EVPN) routes received from neighbor 10.2.5.1 within a VRF named default-vrf:

```
Device# show bgp vrf default-vrf neighbor 21.1.1.1 received-routes l2vpn-evpn detail

MAC VRF Routing Table for Bridge Domain: 0
BGP routing table information for VRF default-vrf
Status: *-valid, >-best s-suppressed, S-stale, d-dampened, h-history
Path type: i-internal, e-external, c-confed, l-local, a-aggregate, r-redist, I-injected
Origin codes: i - IGP, e - EGP, ? - incomplete, | - multipath, & - backup

Prefix: [RD:] [ARP]:[0]:[10.1.1.22]:[32]:[00:16:3e:48:d5:03]:[48] PeerIP: 21.1.1.1, VTEP-IP: 1.1.1.1, Status: Valid, Best
  Origin: IGP, LocalPref: -, MED: -, AS Path: {65535, 65511}
  L2VNI: 10010 L3VNI: 10010
  Communities: -
  Extended Communities: [Opq:Encap:VxLAN EVPN:MAC-Mobility:Sticky:0 AS4:RT:65511:10010
EVPN:Router-MAC:00:16:3e:48:d5:03]
  Uptime: 16h27m2s

Prefix: [RD:] [ARP]:[0]:[10.1.1.22]:[32]:[00:16:3e:48:d5:03]:[48] PeerIP: 22.1.1.1, VTEP-IP: 1.1.1.1, Status: Valid
  Origin: IGP, LocalPref: -, MED: -, AS Path: {65535, 65511}
  L2VNI: 10010 L3VNI: 10010
  Communities: -
  Extended Communities: [Opq:Encap:VxLAN EVPN:MAC-Mobility:Sticky:0 AS4:RT:65511:10010
EVPN:Router-MAC:00:16:3e:48:d5:03]
  Uptime: 16h27m2s

Prefix: [RD:] [ARP]:[0]:[10.1.1.2]:[32]:[00:16:3e:4a:08:03]:[48] PeerIP: 21.1.1.1, VTEP-IP: 1.1.1.1, Status: Valid, Best
  Origin: IGP, LocalPref: -, MED: -, AS Path: {65535, 65511}
  L2VNI: 10010 L3VNI: 10010
  Communities: -
  Extended Communities: [Opq:Encap:VxLAN EVPN:MAC-Mobility:Sticky:0 AS4:RT:65511:10010
EVPN:Router-MAC:00:16:3e:48:d5:03]
  Uptime: 16h27m2s

Prefix: [RD:] [ARP]:[0]:[10.1.1.2]:[32]:[00:16:3e:4a:08:03]:[48] PeerIP: 22.1.1.1, VTEP-IP: 1.1.1.1, Status: Valid
  Origin: IGP, LocalPref: -, MED: -, AS Path: {65535, 65511}
  L2VNI: 10010 L3VNI: 10010
  Communities: -
  Extended Communities: [Opq:Encap:VxLAN EVPN:MAC-Mobility:Sticky:0 AS4:RT:65511:10010
EVPN:Router-MAC:00:16:3e:48:d5:03]
  Uptime: 16h27m2s

Prefix: [RD:] [MAC]:[0]:[48]:[00:10:94:21:21:21] PeerIP: 21.1.1.1, VTEP-IP: 1.1.1.1, Status: Valid, Best
  Origin: IGP, LocalPref: -, MED: -, AS Path: {65535, 65511}
  L2VNI: 10100
  Communities: -
  Extended Communities: [Opq:Encap:VxLAN AS4:RT:65511:10100]
  Uptime: 16h27m2s
```



```
Prefix: [RD:] [MAC]:[0]:[48]:[00:10:94:21:21:21] PeerIP: 22.1.1.1, VTEP-IP: 1.1.1.1,
Status: Valid
  Origin: IGP, LocalPref: -, MED: -, AS Path: {65535, 65511}
  L2VNI: 10100
  Communities: -
  Extended Communities: [Opq:Encap:VxLAN AS4:RT:65511:10100]
  Uptime: 16h27m2s

Prefix: [RD:] [MAC]:[0]:[48]:[00:16:3e:48:d5:03] PeerIP: 21.1.1.1, VTEP-IP: 1.1.1.1,
Status: Valid, Best
  Origin: IGP, LocalPref: -, MED: -, AS Path: {65535, 65511}
  L2VNI: 10100
  Communities: -
  Extended Communities: [Opq:Encap:VxLAN EVPN:MAC-Mobility:Sticky:0 AS4:RT:65511:10100]
  Uptime: 16h27m2s

Prefix: [RD:] [MAC]:[0]:[48]:[00:16:3e:48:d5:03] PeerIP: 22.1.1.1, VTEP-IP: 1.1.1.1,
Status: Valid
  Origin: IGP, LocalPref: -, MED: -, AS Path: {65535, 65511}
  L2VNI: 10100
  Communities: -
  Extended Communities: [Opq:Encap:VxLAN EVPN:MAC-Mobility:Sticky:0 AS4:RT:65511:10100]
  Uptime: 16h27m2s

...

device#
```

show bgp vrf neighbors

Use the **show bgp vrf neighbors** command to display detailed information about all BGP neighbors within a specific VRF.

Syntax

```
show bgp vrf { default-vrf | vrf-name } neighbors
```

Parameters

vrf-name

VRF instance name. This name must start with an alphabetic character or an underscore followed by an arbitrary sequence of alphabetic or numeric characters, underscores, hyphens, or dots. The name cannot exceed 64 characters. Do not use the **profile** reserved keyword.

Modes

EXEC (#)

Examples

The following example displays detailed information about all neighbors within a VRF named default-vrf:

```
device# show bgp vrf default-vrf neighbors

Peer: 10.1.5.1, Remote Port: 179, Peer-AS: 65001
Localhost: 10.1.5.2, Local Port: 51928, Local-AS: 65005, Local-AS-Forced: -
Peer Router ID: 1.1.1.1, Local Router ID: 5.5.5.5, VRF: default-vrf
Route Reflector Client: No, Cluster-ID: -
State: Established, Uptime: 169h46m23s, Dynamic Peer: false, Peer Group: spine-peers

Last Connection Reset Reason:
Send Community: No, Send Extended Community: No
Capabilities   Advertised Received
=====
AS-4Byte      true      true      true
Route-Refresh true      true      true
Graceful-Restart true     true      true
Cisco-RR      true      true      true
MPBGP         true      true      true

Timers          Interval(Sec)   Method          Remaining(Sec)
=====
Connect Retry   31              Default         -
Hold Timer      180             Negotiated      165
Keepalive Timer 60              Default         45
Start Timer     5               System          -
Update Interval 0               Default         -

Graceful Restart(Enabled):
-----
Helper-Only: No, Advertised Restart-Time: 13s
Peer-Restarted: No, Local-Restarted: No
```

```

Received Restart-Time: 30s (Timer not running)
Stale-Routes-Time: 50s (Timer not running)

Address-family Enabled Advertised Received EOR-Sent EOR-Received
=====
IPv4 UNICAST   Yes      No       No       Yes      Yes

Peer: 10.2.5.1, Remote Port: 179, Peer-AS: 65001
Localhost: 10.2.5.2, Local Port: 46178, Local-AS: 65005, Local-AS-Forced: -
Peer Router ID: 2.2.2.2, Local Router ID: 5.5.5.5, VRF: default-vrf
Route Reflector Client: No, Cluster-ID: -
State: Established, Uptime: 169h46m23s, Dynamic Peer: false, Peer Group: spine-peers, Add-
Path: receive

Last Connection Reset Reason:
Send Community: No, Send Extended Community: No

Capabilities   Advertised Received
=====
AS-4Byte      true      true      true
Route-Refresh true      true      true
Graceful-Restart true     true      true
Cisco-RR      true      true      true
MPBGP         true      true      true

Timers          Interval(Sec)   Method          Remaining(Sec)
=====
Connect Retry   31              Default         -
Hold Timer      180             Negotiated      165
Keepalive Timer 60              Default         45
Start Timer     5               System          -Update Interval 0
Default         -

Graceful Restart(Enabled):
-----
Helper-Only: No, Advertised Restart-Time: 13s
Peer-Restarted: No, Local-Restarted: No
Received Restart-Time: 30s (Timer not running)
Stale-Routes-Time: 50s (Timer not running)

Address-family Enabled Advertised Received EOR-Sent EOR-Received
=====
IPv4 UNICAST   Yes      No       No       Yes      Yes
device#

```

show bgp vrf routes

Use the **show bgp vrf routes** command to display information about the BGP protocol running state and applied route configurations within a specific VRF.

Syntax

```
show bgp vrf { default-vrf | vrf-name } routes { ipv4-unicast | ipv6-unicast } [ detail | ipv4-addr | ipv6-addr ]
```

```
show bgp vrf { default-vrf | vrf-name } routes l2vpn-evpn nvo instance-name [ route-type { imr | mac | arp | nd | prefix } ] [ detail ]
```

```
show bgp vrf { default-vrf | vrf-name } routes mac-vrf bridge-domain ID [ route-type { imr | mac | arp | nd } ] [ detail ]
```

Parameters

vrf-name

VRF instance name. This name must start with an alphabetic character or an underscore followed by an arbitrary sequence of alphabetic or numeric characters, underscores, hyphens, or dots. The name cannot exceed 64 characters. Do not use the **profile** reserved keyword.

ipv4-unicast

Displays IPv4 address family and unicast subaddress family information.

ipv6-unicast

Displays IPv6 address family and unicast subaddress family information.

detail | *ipv4-addr* | *ipv6-addr*

(Optional) Displays detailed route information or route information for a specific address and prefix.

- For an IPv4 address, use IPv4 CIDR notation (*A.B.C.D/M*), where *M* is the prefix length in the range of 0–32 for IPv4 addresses. For example, use an IPv4 address of 10.0.0.0 and a prefix length of 8 to specify addresses in the range of 10.0.0.0 to 10.255.255.255.
- For an IPv6 address, use IPv6 CIDR notation (*X:X::X/M*), where *M* is the prefix length in the range of 0–128 for IPv6 addresses. For example, use an IPv6 address of 2001::100 and a prefix length of 64 to specify addresses in the range of 2001:0000:0000:0000:0000:0000:0000:0000 to 2001:0000:0000:0000:ffff:ffff:ffff:ffff.

l2vpn-evpn nvo *instance-name*

Name of the Ethernet VPN (EVPN) Network Virtualization Overlay (NVO) instance for which to display NVO-specific information. This is the overlay network used to deliver the Layer 2 and Layer 3 VPN services. You can use any combination of 1 to 64 uppercase or lowercase alphabetic characters, numerals, underscores, hyphens, or periods. You must use an alphabetic character or an underscore as the first character.

imr

(Optional) Displays inclusive multicast route-specific information.

mac

(Optional) Displays MAC route-specific information.

arp

(Optional) Displays address resolution protocol route-specific information.

nd

(Optional) Displays neighbor discovery route-specific information.

prefix

(Optional) Displays IPv4 prefix route-specific information.

detail

(Optional) Displays detailed route information.

mac-vrf bridge-domain ID

ID number of a bridge domain for which to display the MAC VRF routing table.

Use an integer from 1 to 8192.

Modes

EXEC (#)

Examples

The following example displays IPv4 address family and unicast subaddress family information within a VRF named default-vrf:

```
device# show bgp vrf default-vrf routes ipv4-unicast

IPv4 Routes
-----
Vrf Name: default-vrf
Total number of routes: 8
Seq#      Prefix           Nexthop      Peer          Status
-----
1         1.1.1.1/32       10.1.5.1     10.1.5.1      VALID
2         2.2.2.2/32       10.2.5.1     10.2.5.1      VALID
3         5.5.5.5/32       Localhost    Localhost      VALID
4         5.5.5.5/32       Localhost    Localhost      VALID
5         6.6.6.6/32       10.2.5.1     10.2.5.1      VALID
6         6.6.6.6/32       10.1.5.1     10.1.5.1      VALID
7         10.1.5.0/24      Localhost    Localhost      VALID
8         10.1.6.0/24      10.2.5.1     10.2.5.1      VALID
9         10.1.6.0/24      10.1.5.1     10.1.5.1      VALID
10        10.2.5.0/24      Localhost    Localhost      VALID
11        10.2.6.0/24      10.2.5.1     10.2.5.1      VALID
12        10.2.6.0/24      10.1.5.1     10.1.5.1      VALID
device#
```

The following example displays IPv6 address family and unicast subaddress family information within a VRF named default-vrf:

```
device# show bgp vrf default-vrf routes ipv6-unicast

BGP routing table information for VRF default-vrf
Status: *-valid, >-best, S-stale
Path type: i-internal, e-external, l-local, r-redist, m-multipath, a-additional,
           p-primary, s-secondary
Origin codes: I - IGP, E - EGP, ? - incomplete
IPv6 Routes
-----
Total number of routes: 1

Flags    Prefix          Nexthop          Peer
=====
*>?i    10:0:5::/64    1000::2          10.1.5.2
*?i     10:0:5::/64    1000::2          1000::2
        10:0:5::/64    ::ffff:10.2.5.2  10.2.5.2
device#
```

The following example displays detailed IPv4 address family and unicast subaddress family information within a VRF named default-vrf:

```
device# show bgp vrf default-vrf routes ipv4-unicast detail

Prefix: string_val:"1.1.1.1/32", Nexthop: , Peer: string_val:"10.1.5.1", Path ID:
uint_val:4294967295, AS Path:
  Age: 152h10m39s, Status: VALID
  Origin: , Local Preference: -, MED: -, Weight: -, Admin Distance: -
  Communities: -
  Extended-Communities: -

Prefix: string_val:"2.2.2.2/32", Nexthop: , Peer: string_val:"10.2.5.1", Path ID:
uint_val:4294967295, AS Path:
  Age: 152h10m39s, Status: VALID
  Origin: , Local Preference: -, MED: -, Weight: -, Admin Distance: -
  Communities: -
  Extended-Communities: -

Prefix: string_val:"5.5.5.5/32", Nexthop: , Peer: string_val:"127.0.0.1", Path ID:
uint_val:2, AS Path:
  Age: 152h10m39s, Status: VALID
  Origin: , Local Preference: -, MED: -, Weight: -, Admin Distance: -
  Communities: -
  Extended-Communities: -
device#
```

The following example displays IPv4 address family and unicast subaddress family information for the 1.1.1.1/32 prefix within a VRF named default-vrf:

```
device# show bgp vrf default-vrf routes ipv4-unicast 1.1.1.1/32

Prefix: string_val:"1.1.1.1/32", Nexthop: , Peer: string_val:"10.1.5.1", Path ID:
uint_val:4294967295, AS Path:
  Age: 152h10m16s, Status: VALID
  Origin: , Local Preference: -, MED: -, Weight: -, Admin Distance: -
  Communities: -
  Extended-Communities: -
device#
```

The following example displays information about all routes for an Ethernet VPN (EVPN) NVO instance named `nvo1` within a VRF named `default-vrf`:

```
device# show bgp vrf default-vrf routes l2vpn-evpn nvo nvo1
```

```
EVPN Routing Table for NVO: nvo1
```

```
RD: 0.0.0.0:3
```

```
EVPN Type-Five (Prefix) Routes:
```

```
=====
```

Prefix		PeerIP
VTEP-IP	L3VNI	Uptime

*> 100.100.100.0/24		10.2.5.1
6.6.6.6	10200	152h13m2s
*		10.1.5.1
6.6.6.6	10200	152h13m2s
*> 100:100::/64		10.2.5.1
6.6.6.6	0	152h13m2s
*		10.1.5.1
6.6.6.6	0	152h13m2s

```
RD: 5.5.5.5:3
```

```
EVPN Type-Five (Prefix) Routes:
```

```
=====
```

Prefix		PeerIP
VTEP-IP	L3VNI	Uptime

*> 100.100.100.0/24		localhost
5.5.5.5	10200	152h13m2s
*> 100:100::/64		localhost
5.5.5.5	10200	152h13m2s

```
RD: 5.5.5.5:100
```

```
EVPN Type-Two (Mac) Routes:
```

```
=====
```

MacAddress	TAG	PeerIP			VTEP-
IP			L2VNI	ESI	Uptime

*> 00:01:00:01:00:01	0	localhost			
5.5.5.5			10100		152h13m2s
*> 00:05:00:05:00:05	0	localhost			
5.5.5.5			10100		152h13m2s
*> 00:16:3e:2e:4c:03	0	localhost			
5.5.5.5			10100		152h13m2s

```
RD: 5.5.5.5:100
```

```
EVPN Type-Three (IMET) Routes:
```

```
=====
```

TAG	Originator-IP		PeerIP
VTEP-IP		L2VNI	Uptime

*> 0	5.5.5.5		localhost
5.5.5.5		0	152h13m2s

```
RD: 6.6.6.6:100
```

```
EVPN Type-Two (Mac) Routes:
```

```
=====
```

MacAddress	TAG	PeerIP			VTEP-
IP			L2VNI	ESI	Uptime

```

-----
*> 00:16:3e:60:51:03 0 10.2.5.1
6.6.6.6 10100 152h13
m2s
* 10.1.5.1
6.6.6.6 10100 152h13
m2s
*> 06:01:00:01:00:01 0 10.2.5.1
6.6.6.6 10100 152h13
m2s
* 10.1.5.1
6.6.6.6 10100 152h13
m2s
*> 06:02:00:02:00:02 0 10.2.5.1
6.6.6.6 10100 152h13
m2s
* 10.1.5.1
6.6.6.6 10100 152h13
m2s
*> 06:03:00:03:00:03 0 10.2.5.1
6.6.6.6 10100 152h13
m2s
* 10.1.5.1
6.6.6.6 10100 152h13
m2s
*> 06:04:00:04:00:04 0 10.2.5.1
6.6.6.6 10100 152h13
m2s
* 10.1.5.1
6.6.6.6 10100 152h13
m2s
*> 06:05:00:05:00:05 0 10.2.5.1
6.6.6.6 10100 152h13
m2s
* 10.1.5.1
6.6.6.6 10100 152h13
m2s
RD: 6.6.6.6:100

EVPN Type-Two (ARP) Routes:
=====
      IP Address      MacAddress      TAG      PeerIP      VTEP-
IP                               L2VNI      L3VNI      ESI
                               Uptime
-----
-----
*> 100.100.100.6 00:16:3e:60:51:03 0 10.2.5.1
6.6.6.6 10100 10200
152h13m2s
* 0 10.1.5.1
6.6.6.6 10100 10200
152h13m2s
*> 100.100.100.61 06:01:00:01:00:01 0 10.2.5.1
6.6.6.6 10100 10200
152h13m2s
* 0 10.1.5.1
6.6.6.6 10100 10200
152h13m2s
*> 100.100.100.62 06:02:00:02:00:02 0 10.2.5.1
6.6.6.6 10100 10200
152h13m2s
* 0 10.1.5.1
6.6.6.6 10100 10200

```



```

152h13m2s
*> 100.100.100.63 06:03:00:03:00:03 0 10.2.5.1
6.6.6.6 10100 10200

152h13m2s
* 0 10.1.5.1
6.6.6.6 10100 10200

152h13m2s
*> 100.100.100.64 06:04:00:04:00:04 0 10.2.5.1
6.6.6.6 10100 10200

152h13m2s
* 0 10.1.5.1
6.6.6.6 10100 10200

152h13m2s
*> 100.100.100.65 06:05:00:05:00:05 0 10.2.5.1
6.6.6.6 10100 10200

152h13m2s
* 0 10.1.5.1
6.6.6.6 10100 10200

152h13m2s
RD: 6.6.6.6:100

EVPN Type-Two (ND) Routes:
=====
      IP Address                      MacAddress      TAG
PeerIP                               VTEP-IP
L3VNI      ESI                      Uptime      L2VNI
-----
*> 100:100::100:6                      00:16:3e:60:51:03 0
10.2.5.1                      6.6.6.6                      10100
10200                      152h13m2s

*
10.1.5.1                      6.6.6.6                      10100
10200                      152h13m2s
RD: 6.6.6.6:100
EVPN Type-Three (IMET) Routes:
=====
      TAG      Originator-IP
VTEP-IP                               L2VNI      PeerIP
Uptime
-----
*> 0      6.6.6.6                      10.2.5.1
6.6.6.6                      0      152h13m2s
* 0      6.6.6.6                      10.1.5.1
6.6.6.6                      0      152h13m2s
device#

```

The following example displays information about inclusive multicast routes for an EVPN NVO instance named `nvo1` within a VRF named `default-vrf`:

```

device# show bgp vrf default-vrf routes l2vpn-evpn nvo nvo1 route-type imr

EVPN Routing Table for NVO: nvo1
RD: 5.5.5.5:100
EVPN Type-Three (IMET) Routes:
=====
      TAG      Originator-IP
VTEP-IP                               L2VNI      PeerIP
Uptime
-----
*> 0      5.5.5.5                      localhost
5.5.5.5                      0      152h12m2s
RD: 6.6.6.6:100

```

```
EVPN Type-Three (IMET) Routes:
```

```
=====
```

TAG	Originator-IP		PeerIP
VTEP-IP		L2VNI	Uptime

*> 0	6.6.6.6		10.2.5.1
6.6.6.6		0	152h12m2s
* 0	6.6.6.6		10.1.5.1
6.6.6.6		0	152h12m2s
device#			

The following example displays detailed information about inclusive multicast routes for an EVPN NVO instance named nvo1 within a VRF named default-vrf:

```
device# show bgp vrf default-vrf routes l2vpn-evpn nvo nvo1 route-type imr detail
```

```
EVPN Routing Table for NVO: nvo1
```

```
[RD: 5.5.5.5:100][TAG: 0][Originator: 5.5.5.5]
PeerIP: localhost, VTEP-IP: 5.5.5.5, PathID: 4294967295, VALID, Best
Origin: INCOMPLETE, LocalPref: 100, MED: -, AS Path:
L2VNI: 0
Communities: -
Extended Communities: [Opq:Encap:VxLAN AS4:RT:65005:10100]
Uptime: 152h12m36s
```

```
[RD: 6.6.6.6:100][TAG: 0][Originator: 6.6.6.6]
PeerIP: 10.2.5.1, VTEP-IP: 6.6.6.6, PathID: 4294967295, VALID, Best
Origin: INCOMPLETE, LocalPref: -, MED: -, AS Path: {65001, 65006}
L2VNI: 0
Communities: -
Extended Communities: [Opq:Encap:VxLAN AS4:RT:65006:10100]
Uptime: 152h12m36s
PeerIP: 10.1.5.1, VTEP-IP: 6.6.6.6, PathID: 4294967295, VALID
Origin: INCOMPLETE, LocalPref: -, MED: -, AS Path: {65001, 65006}
L2VNI: 0
Communities: -
Extended Communities: [Opq:Encap:VxLAN AS4:RT:65006:10100]
Uptime: 152h12m36s
```

```
device#
```

show bgp vrf summary

Use the **show bgp vrf summary** command to display a summary of the BGP protocol running state, current route received/advertised count status, and neighbor details within a specific VRF.

Syntax

```
show bgp vrf { default-vrf | vrf-name } { ipv4-unicast | ipv6-unicast | l2vpn-evpn }
```

Parameters

vrf-name

VRF instance name. This name must start with an alphabetic character or an underscore followed by an arbitrary sequence of alphabetic or numeric characters, underscores, hyphens, or dots. The name cannot exceed 64 characters. Do not use the "profile" reserved keyword.

ipv4-unicast

Displays IPv4 address family and unicast subaddress family information.

ipv6-unicast

Displays IPv6 address family and unicast subaddress family information.

l2vpn-evpn

Displays Layer 2 VPN address family and Ethernet VPN (EVPN) subaddress family information.

Modes

EXEC (#)

Examples

The following example displays summarized IPv4 address family and unicast subaddress family information for a VRF named default-vrf:

```
device# show bgp vrf default-vrf summary ipv4-unicast

VRF Name : default-vrf
Local AS: 65005, Router ID: 5.5.5.5
Confederation Identifier: not configured
Confederation Peers: not configured
Graceful Restart: Enabled
    Restart-Time(seconds): 20, Stale-Routes-Time(seconds): 30

IPv4 Unicast Summary Information
-----
Graceful Restart: Enabled
Number of Routes :0 (Paths: 0)
Number of Peer(s) Configured : 2, Up : 2
D: Dynamically created based on a listen range command, Dynamically created neighbors: 0
C: Route Reflector Client
```

```

Peer IP      Peer AS  Local IP      State      Health      UpTime      RIB-
IN          RIB-OUT  Flags Peer Group
=====
10.1.5.1      65001    10.1.5.2      Established -          150h1m3s
0            0        -      spine-peers
10.2.5.1      65001    10.2.5.2      Established -          150h1m3s
0            0        -      spine-peers
device#

```

The following example displays summarized IPv6 address family and unicast subaddress family information for a VRF named default-vrf:

```

device# show bgp vrf default-vrf summary ipv6-unicast

VRF Name : default-vrf
Local AS: 65000, Router ID: 4.4.4.4
Confederation Identifier: not configured
Confederation Peers: not configured
Graceful Restart: Disabled

IPv6 Unicast Summary Information
-----
Graceful Restart: Disabled
Prefix Independent Convergence: Disabled
Nexthop address resolution - Recursion: Disabled, Default-Route: Disabled
Number of Routes :1 (Paths: 1)
Number of Peer(s) Configured : 3, Up : 3
D: Dynamically created based on a listen range command, Dynamically created neighbors: 0
C: Route Reflector Client

Peer IP      Peer AS  Local IP      State      Health  UpTime  RIB-IN  RIB-OUT  Flags  Peer
Group
=====
10.1.5.1  65000    10.1.5.2  Established -      16m25s  0       1       -      spine-
peers
10.2.5.1  65000    10.2.5.2  Established -      16m23s  0       1       -      spine-
peers
1000::1   65000    1000::2   Established -      58s     0       1       -      ipv6-
peers
device#

```

The following example displays summarized Layer 2 VPN address family and EVPN subaddress family information for a VRF named default-vrf:

```

device# show bgp vrf default-vrf summary l2vpn-evpn

VRF Name : default-vrf
Local AS: 65005, Router ID: 5.5.5.5
Confederation Identifier: not configured
Confederation Peers: not configured
L2VPN EVPN Summary Information
-----
Number of Routes :0 (Paths: 0)
Number of Peer(s) Configured : 2, Up : 2
D: Dynamically created based on a listen range command, Dynamically created neighbors: 0
C: Route Reflector Client

Peer IP      Peer AS  Local IP      State      Health  UpTime      RIB-IN  RIB-OUT  Flags  Peer Group
=====
10.1.5.1  65001    10.1.5.2  Established -      150h11m13s  0       0       -      spine-peers
10.2.5.1  65001    10.2.5.2  Established -      150h11m13s  0       0       -      spine-peers
device#

```

show bridge-domain

Use the **show bridge-domain** command to display the bridge domain information.

Syntax

```
show bridge-domain ID
```

Parameters

bridge-domain *ID*

Bridge domain ID.

Modes

EXEC (#)

Output

The **show bridge-domain** command displays the following information:

Output field	Description
Bridge domain	Bridge domain ID.
Mode	Domain configuration mode. This is either L2VSI_P2MP (the default mode) or L2VSI_VLAN (VLAN mode).
Vni Domain	Virtual Network Identifier (VNI) domain name.
Vni	VNI value for the bridge domain in the VNI domain.
Total number of member ports	Number of subinterfaces that are assigned to the bridge domain.
If Name	Type (Ethernet or Port Channel) and name of the member subinterface.
Vlan	VLAN range of the member subinterface.
Inner Vlan	Inner VLAN range of the member subinterface.
Admin Status	Administrative status of the member subinterface. The possible values are DOWN (forced administratively down indefinitely) or UP.
Oper Status	Operational status of the member subinterface. The possible values are DOWN or UP.

Examples

The following example displays the bridge domain information:

```
device# show bridge-domain all  
  
Bridge Domain 100    Mode L2VSI_P2MP
```

```
Vni Domain base Vni 64000
Total number of member ports 1
Member ports:
If Name  Vlan      Inner Vlan Admin Status Oper Status
=====  =====  =====  =====  =====
Eth 0/7  100                UP          DOWN

Bridge Domain 200      Mode L2VSI_P2MP
Total number of member ports 2
Member ports:
If Name  Vlan      Inner Vlan Admin Status Oper Status
=====  =====  =====  =====  =====
Eth 0/7  200                UP          DOWN
Eth 0/7  201                UP          DOWN
device#
```

show counters bridge-domain

Use the **show counters bridge-domain** command to display statistics for a specific bridge domain, a range of bridge domains, or all bridge domains.

Syntax

```
show counters bridge-domain { bridge-domain-range | all }
```

Parameters

bridge-domain-range

Bridge domain ID or range of IDs. Use an integer or range of integers from 1 to 8192. For example, 10 or 100-200.

all

Displays statistics for all bridge domains.

Modes

EXEC (#)

Usage Guidelines

You must clear the counters if you want to start at zero for more recent statistics. For details, see [clear counters bridge-domain](#) on page 512.

Output

The **show counters bridge-domain** command displays the following information:

Output Field	Description
Bridge Domain	Bridge domain ID.
Mode	Domain configuration mode.
Total number of member ports	Number of subinterfaces that are assigned to the bridge domain.
Rx Pkts	The number of packets received on all member subinterfaces.
Rx Bytes	The number of bytes received on all member subinterfaces.
Tx Pkts	The number of packets transmitted from all member subinterfaces.
Tx Bytes	The number of bytes transmitted from all member subinterfaces.
IF Name	Type (Ethernet or Port Channel) and name of the member subinterface.
Vlan	VLAN range of the member subinterface.
Inner Vlan	Inner VLAN range of the member subinterface.

Output Field	Description
Rx Pkts	The number of packets received at the member subinterface.
Rx Bytes	The number of bytes received at the member subinterface.
Tx Pkts	The number of packets transmitted from the member subinterface.
Tx Bytes	The number of bytes transmitted from the member subinterface.

**Note**

Bridge domain LIF statistics in VLAN mode or default mode are not supported because of a BCM limitation. Only global bridge domain statistics are supported for the 8730-32D platform.

Examples

The following example displays statistics for bridge domain 100:

```
device# show counters bridge-domain 100

*untag-s --accepts untagged only  *untag --accepts untagged+tagged

Bridge Domain 100      Mode L2VSI_P2MP
Total number of member ports  2
Rx Pkts  Rx Bytes Tx Pkts  Tx Bytes
=====  =====
1340      171520   670      83080

If Name    Vlan      Inner Vlan Rx Pkts  Rx Bytes Tx Pkts  tx bytes
=====  =====
Eth 0/1:1 untag-s  --        670      85760   335      41540
Eth 0/1:2 untag-s  --        670      85760   335      41540
device#
```

The following example displays statistics for a range of bridge domains from 100 through 200:

```
device# show counters bridge-domain 100-200

*untag-s --accepts untagged only  *untag --accepts untagged+tagged

Bridge Domain 100      Mode L2VSI_P2MP
Total number of member ports  2
Rx Pkts  Rx Bytes Tx Pkts  Tx Bytes
=====  =====
1340      171520   670      83080

If Name    Vlan      Inner Vlan Rx Pkts  Rx Bytes Tx Pkts  tx bytes
=====  =====
Eth 0/1:1 untag-s  --        670      85760   335      41540
Eth 0/1:2 untag-s  --        670      85760   335      41540

Bridge Domain 200      Mode L2VSI_P2MP
Total number of member ports  2
Rx Pkts  Rx Bytes Tx Pkts  Tx Bytes
=====  =====
1340      171520   670      83080
```



```

If Name      Vlan      Inner Vlan Rx Pkts  Rx Bytes Tx Pkts  tx bytes
=====
Eth 0/1:3   untag-s  --         0         0         0         0
Eth 0/1:4   untag-s  --         0         0         0         0
device#

```

The following example displays statistics for all bridge domains on the device:

```

device# show counters bridge-domain all

*untag-s --accepts untagged only  *untag --accepts untagged+tagged
Bridge Domain 2078      Mode L2VSI_P2MP
Rx Pkts  Rx Bytes Tx Pkts  Tx Bytes
=====
34453    5374640  34453    5133518
Bridge Domain 3078      Mode L2VSI_VLAN Vlan 3078
Rx Pkts  Rx Bytes Tx Pkts  Tx Bytes
=====
248673   40409294 249188   34049660
device#

```

show counters interface ethernet

Use the **show counters interface ethernet** command to display statistics for one or more Ethernet interfaces.

Syntax

```
show counters interface ethernet { interface-name | all | brief }
```

Parameters

interface-name

Ethernet interface name. Use the *slot/port* format or the *slot/port:breakout* format to specify an interface or one or more comma-separated ranges of interfaces respectively. For example, 0/1 or 1/1-3,5,2/7-9,10:1-4 respectively.

all

Displays detailed statistics for all Ethernet interfaces.

brief

Displays brief statistics for all Ethernet interfaces.

Modes

EXEC (#)

Usage Guidelines

You must clear the counters if you want to start at zero for more recent statistics. For details, see [clear counters interface ethernet](#) on page 513.

Output

The **show counters interface ethernet** command displays the following information:

Output field	Description
Interface Statistics	Ethernet interface for which statistics are displayed.
Carrier Transitions	Carrier Transitions—Number of times the interface has “flapped” (completed a cycle of failing and then recovering). This number increases when the cable is unplugged, the remote system is powered down and then up, or another problem occurred. If this number increments quickly (such as every 10 seconds), this means that the cable, the remote system, or the PIC or PIM are malfunctioning.

Output field	Description
LastClear	Time elapsed since the counters that measure cumulative statistics (such as number of bytes transmitted and received) shown in this report were last reset to zero (such as when the device was rebooted). Note that variables that might affect routing (for example, load and reliability) are not cleared when the counters are cleared.
Input	Statistics for data received on the interface: • Total pkts—Total number of packets of all types received by the interface. • Broadcast pkts—Total number of broadcast packets received by the interface. • Discard pkts—Number of input packets that were silently discarded (possibly because of buffer overflow). • Errors pkts—Total number of no buffer, runts, giants, CRC, frame, overrun, ignored, and abort counts. Other input-related errors can also increment the count, so that this sum might not balance with the other counts. • FCS Errors—Total number of errors where a legal sized frame with a bad frame check sequence (CRC error). An FCS error can be caused by a duplex mismatch, faulty NIC or driver, cabling, hub, or induced noise. • MCast pkts—Total number of multicast packets received by the interface. • Octets—The number of octets (bytes) received by the interface. • UCast pkts—Total number of unicast packets received by the interface. • Runt pkts—Number of packets that are discarded because they are smaller than the minimum packet size of the medium. • CRC Errors—Number of times that the cyclic redundancy checksum (CRC) generated by the originating LAN station or remote device does not match the checksum calculated from the data received. On a LAN, this usually means noise or transmission problems on the LAN interface or the LAN bus. A high number is usually from collisions or a station transmitting bad data. On a serial link, CRCs usually indicate noise, gain hits, or other transmission problems on the data link.

Output field	Description
Input Distribution	Numbers of packets of all types received by the interface (grouped by size). Following are the size groups: • 64 or fewer bytes • 65 to 127 bytes • 128 to 255 bytes • 256 to 511 bytes • 512 to 1,023 bytes • 1,024 to 1,518 bytes • Jumbo packets—1,519 or more bytes (larger than the standard Ethernet IEEE 802.3 frame size of 1,518 bytes)
Out	Counts of packets sent by the interface: • Total pkts—Total number of packets of all types sent by the interface. • Broadcast pkts—Total number of broadcast packets sent by the interface. • Discard pkts—Number of output packets that were silently discarded (possibly because of buffer overflow). • Errors pkts—Total number of no buffer, runts, giants, CRC, frame, overrun, ignored, and abort counts. Other input-related errors can also increment the count, so that this sum might not balance with the other counts. • MCast pkts—Total number of multicast packets sent by the interface. • Octets—The number of octets ((bytes) sent by the interface. • UCast pkts—Total number of unicast packets received by the interface.

Examples

The following example displays statistics for Ethernet interface 0/80:

```
device# show counters interface ethernet 0/80
```

```
Interface Statistics: ethernet 0/80
```

```
Carrier Transitions: 2
```

```
LastClear: 0s
```

```
Input:
```

```
Total pkts: 24832
```

```
Broadcast pkts: 24101
```

```
Discard pkts: 505
```

```
Errors pkts: 503
```

```
FCS Errors: 0
```

```
MCast pkts: 413
```

```
Octets: 3104
```

```
UCast pkts: 1922
```

```
Runt pkts: 502
```

```
CRC Errors: 0
```

```
Input Distribution:
```

```
64 byte pkts: 24002
```

```
65-127 byte pkts: 131
```

```
128-255 byte pkts: 232
```

```
256-511 byte pkts: 396
```

```

    512-1023 byte pkts: 29
    1024-1518 byte pkts: 42
        Jumbo pkts: 0
Out:
    Total pkts: 2746
    Broadcast pkts: 2002
    Discard pkts: 101
    Errors pkts: 21
    MCast pkts: 42
        Octets: 343
    UCast pkts: 237
Rate Info:
    Input: 301.000000 Mbits/sec, 25402 pkts/sec 91.00% of line-rate
    Output: 322.000000 Mbits/sec, 26312 pkts/sec 93.00% of line-rate
device#

```

The following example displays statistics for all Ethernet interfaces on the device:

```

device# show counters interface ethernet all

Interface Statistics: ethernet 0/80
  Carrier Transitions: 2
    LastClear: 0s
Input:
    Total pkts: 24832
    Broadcast pkts: 24101
    Discard pkts: 505
    Errors pkts: 503
    FCS Errors: 0
    MCast pkts: 413
    Octets: 3104
    UCast pkts: 1922
    Runt pkts: 502
    CRC Errors: 0
Input Distribution:
    64 byte pkts: 24002
    65-127 byte pkts: 131
    128-255 byte pkts: 232
    256-511 byte pkts: 396
    512-1023 byte pkts: 29
    1024-1518 byte pkts: 42
    Jumbo pkts: 0
Out:
    Total pkts: 2746
    Broadcast pkts: 2002
    Discard pkts: 101
    Errors pkts: 21
    MCast pkts: 42
    Octets: 343
    UCast pkts: 237
Rate Info:
    Input: 301.000000 Mbits/sec, 25402 pkts/sec 91.00% of line-rate
    Output: 322.000000 Mbits/sec, 26312 pkts/sec 93.00% of line-rate

Interface Statistics: ethernet 0/90
  Carrier Transitions: 2
    LastClear: 0s
Input:
    Total pkts: 24822
    Broadcast pkts: 24099
    Discard pkts: 503
    Errors pkts: 501
    FCS Errors: 0
    MCast pkts: 411

```

```

        Octets: 3104
        UCast pkts: 1920
        Runt pkts: 502
        CRC Errors: 0
Input Distribution:
    64 byte pkts: 23992
    65-127 byte pkts: 129
    128-255 byte pkts: 230
    256-511 byte pkts: 394
    512-1023 byte pkts: 27
    1024-1518 byte pkts: 40
    Jumbo pkts: 0
Out:
    Total pkts: 2736
    Broadcast pkts: 2000
    Discard pkts: 91
    Errors pkts: 19
    MCast pkts: 40
    Octets: 343
    UCast pkts: 235
Rate Info:
    Input: 319.000000 Mbits/sec, 25702 pkts/sec 92.00% of line-rate
    Output: 344.000000 Mbits/sec, 26912 pkts/sec 94.00% of line-rate
device#
```

show counters interface port-channel

Use the **show counters interface port-channel** command to display statistics for one or more port channel interfaces.

Syntax

```
show counters interface port-channel { port-channel-range | all }
```

Parameters

port-channel-range

Port channels. Use an integer or one or more comma-separated ranges of integers from 1 to 255. For example, 1 or 1-3,5,7-9.

all

Displays statistics for all port channels.

Modes

EXEC (#)

Usage Guidelines

You must clear the counters if you want to start at zero for more recent statistics. For details, see [clear counters interface port-channel](#) on page 514.

Examples

The following example displays statistics for port channel port 1:

```
device# show counters interface port-channel 1

Interface Statistics: port-channel 1
  Carrier Transitions: 2
    LastClear: 0s
Input:
  Total pkts: 24832
  Broadcast pkts: 24101
  Discard pkts: 505
  Errors pkts: 503
  FCS Errors: 0
  MCast pkts: 413
  Octets: 3104
  UCast pkts: 1922
  Runt pkts: 502
  CRC Errors: 0
Input Distribution:
  64 byte pkts: 24002
  65-127 byte pkts: 131
  128-255 byte pkts: 232
  256-511 byte pkts: 396
  512-1023 byte pkts: 29
  1024-1518 byte pkts: 42
```

```

                Jumbo pkts: 0
Out:
    Total pkts: 2746
    Broadcast pkts: 2002
    Discard pkts: 101
    Errors pkts: 21
    MCast pkts: 42
    Octets: 343
    UCast pkts: 237
Rate Info:
    Input: 301.000000 Mbits/sec, 25402 pkts/sec 91.00% of line-rate
    Output: 322.000000 Mbits/sec, 26312 pkts/sec 93.00% of line-rate
device#

```

The following example displays statistics for all port channel interfaces on the device:

```

device# show counters interface port-channel all

Interface Statistics: port-channel 1
  Carrier Transitions: 2
  LastClear: 0s
Input:
  Total pkts: 24832
  Broadcast pkts: 24101
  Discard pkts: 505
  Errors pkts: 503
  FCS Errors: 0
  MCast pkts: 413
  Octets: 3104
  UCast pkts: 1922
  Runt pkts: 502
  CRC Errors: 0
Input Distribution:
  64 byte pkts: 24002
  65-127 byte pkts: 131
  128-255 byte pkts: 232
  256-511 byte pkts: 396
  512-1023 byte pkts: 29
  1024-1518 byte pkts: 42
  Jumbo pkts: 0
Out:
  Total pkts: 2746
  Broadcast pkts: 2002
  Discard pkts: 101
  Errors pkts: 21
  MCast pkts: 42
  Octets: 343
  UCast pkts: 237
Rate Info:
  Input: 301.000000 Mbits/sec, 25402 pkts/sec 91.00% of line-rate
  Output: 322.000000 Mbits/sec, 26312 pkts/sec 93.00% of line-rate

Interface Statistics: port-channel 2
  Carrier Transitions: 2
  LastClear: 0s
Input:
  Total pkts: 24822
  Broadcast pkts: 24099
  Discard pkts: 503
  Errors pkts: 501
  FCS Errors: 0
  MCast pkts: 411
  Octets: 3104
  UCast pkts: 1920

```



```
          Runt pkts: 502
          CRC Errors: 0
Input Distribution:
          64 byte pkts: 23992
          65-127 byte pkts: 129
          128-255 byte pkts: 230
          256-511 byte pkts: 394
          512-1023 byte pkts: 27
          1024-1518 byte pkts: 40
          Jumbo pkts: 0
Out:
          Total pkts: 2736
          Broadcast pkts: 2000
          Discard pkts: 91
          Errors pkts: 19
          MCast pkts: 40
          Octets: 343
          UCast pkts: 235
Rate Info:
          Input: 319.000000 Mbits/sec, 25702 pkts/sec 92.00% of line-rate
          Output: 344.000000 Mbits/sec, 26912 pkts/sec 94.00% of line-rate
device#
```

show counters interface tunnel

Use the **show counters interface tunnel** command to display statistics for one or more tunnel interfaces.

Syntax

```
show counters interface tunnel { interface-name | all }
```

Parameters

interface-name

Tunnel interface name. Use a string or an integer from 1 to 100000

all

Displays statistics for all tunnel interfaces.

Modes

EXEC (#)

Usage Guidelines

You must clear the counters if you want to start at zero for more recent statistics. For details, see [clear counters interface tunnel](#) on page 515.

Examples

The following example displays statistics for tunnel interface 1:

```
device# show counters interface tunnel 1
```

Tunnel Name	Rx Pkts	Rx Bytes	Tx Pkts	Tx Bytes
=====	=====	=====	=====	=====
tunnel 1	363	41382	624	54288

```
device#
```

The following example displays statistics for all tunnel interfaces on the device:

```
device# show counters interface tunnel all
```

Tunnel Name	Rx Pkts	Rx Bytes	Tx Pkts	Tx Bytes
=====	=====	=====	=====	=====
tunnel 1	363	41382	624	54288
tunnel 2	390	40347	514	49279
tunnel 3	249	31903	914	33286

```
device#
```

show counters lacp

Use the **show counters lacp** command to display statistics for member ports of dynamic link aggregation groups (LAGs).

Syntax

```
show counters lacp
```

Parameters

This command has no keywords or variables.

Modes

EXEC (#)

Usage Guidelines

You can use the **show running-config lacp** command to display the LAG configuration that is running currently on a device.

You must clear the counters if you want to start at zero for a more recent count of the packets that match an access list. To do so, use the **clear counters lacp** command.

Output

The **show counters lacp** command displays the following information:

Output field	Description
Port	Interface name of the Ethernet or LAG (port channel) interface.
Channel group	LAG of which the interface is a member.
ethernet 0/21	Name of the Ethernet member interface.
in-pkts	Number of LACP packets received on the member interface. This number is updated every second.
out-pkts	Number of LACP packets transmitted on the member interface. This number is updated every 10 second.
TxErr	Number of packet transmit errors on the member interface.
RxErr	Number of packet receive errors on the member interface.
unknownErr	Number of unknown errors on the member interface.
LacpErr	Number of LACP errors on the member interface.

Examples

The following example displays statistics for member ports of the dynamic link aggregation groups that are configured on the device:

```
device# show counters lacp
```

Port	in-pkts	out-pkts	TxErr	RxErr	unknownErr	LacpErr

Channel group: 10						
ethernet 0/21	81	76	0	0	0	0

```
device#
```

show counters link-fault-signaling

Use the **show counters link-fault-signaling** command to display statistics for Link Fault Signaling (LFS) detection.

Syntax

```
show counters link-fault-signaling
```

Parameters

This command has no keywords or variables.

Modes

EXEC (#)

Usage Guidelines

This command displays the status of each LFS enabled link to reflect the health of those links. It shows the number of times that local faults and remote faults occurred on each port. It also displays the date and time of the last fault seen on each port.

LFS is a physical layer protocol that enables communication on a link between two Ethernet devices. It operates between the remote Ethernet device Reconciliation Sublayer (RS) and the local Ethernet device RS. When configured on an Ethernet port, the port can detect and report fault conditions on transmit and receive ports.

To see whether LFS is enabled on a specific port as well as the general link fault status on that port, see [show interface ethernet link-error-disable](#) on page 1012. To see the LFS configuration for each front port, see [show link-fault-signaling](#) on page 1045.

You must clear the counters if you want to start at zero for more recent statistics. For details, see xxxxxx.

Examples

The following example displays statistics for LFS detection on the device:

```
device# show counters link-fault-signaling
```

Port	Local-Fault-Count	Last-Local-Fault	Remote-Fault-Count	Last-Remote-Fault
=====	=====	=====	=====	=====
Eth 1/1	2	2024-09-22T01:37:04Z	1	2024-09-22T01:36:51Z
Eth 1/2	0	NA	0	NA
Eth 1/3	0	NA	0	NA
Eth 1/4	1	2024-09-22T01:36:50Z	1	2024-09-22T01:36:52Z
Eth 1/5	0	NA	0	NA
Eth 1/6	0	NA	0	NA
Eth 1/7	1	2024-09-22T01:36:50Z	1	2024-09-22T01:36:54Z
Eth 1/8	1	2024-09-22T01:36:50Z	1	2024-09-22T01:36:53Z

```
device#
```

Z indicates Zulu Time, which is the zone designator for the zero UTC offset. 01:37:04 with the zero UTC offset therefore displays as T01:37:04Z in the example above.

show counters lldp interface

Use the **show counters lldp interface** command to display link layer discovery protocol (LLDP) statistics for one or more Ethernet interfaces.

Syntax

```
show counters lldp interface ethernet { interface-name | all }
```

Parameters

interface-name

Ethernet interface name. Use the *slot/port* format or the *slot/port:breakout* format to specify an interface or one or more comma-separated ranges of interfaces respectively. For example, 0/1 or 1/1-3,5,2/7-9,10:1-4 respectively.

all

Displays LLDP statistics for all Ethernet interfaces.

Modes

EXEC (#)

Usage Guidelines

This command provides information about the number of LLDP messages received and sent on a particular network interface. This information helps troubleshoot issues related to LLDP communication and ensure proper network configuration. The statistics can be used to identify errors in the LLDP protocol and optimize network performance.

You must clear the counters if you want to start at zero for more recent statistics. For details, see [clear counters lldp](#) on page 517.

To view the global LLDP statistics for the device, see [show counters lldp summary](#) on page 993.

Output

The **show counters lldp interface** command displays the following information:

Output field	Description
FrameIn	Number of LLDP frames received on this interface.
FrameOut	Number of LLDP frames sent on this interface.
FrameDiscard	Number of LLDP frames received on this interface that were discarded because of problems.

Output field	Description
FrameErrorIn	Number of LLDP frames with errors received on this interface.
FrameErrorOut	Number of LLDP frames with errors sent on this interface.
TlvDiscard	Number of LLDP TLVs (information elements that contains the type, length, and value fields) received and then discarded on an interface. TLVs carry information about the link-layer capabilities of a device, such as the type of link-layer protocol supported, the maximum transmission unit (MTU), and the device's MAC address.
TlvUnknown	Number of unrecognized LLDP TLVs received on an interface.
LastClear	Time elapsed since the counters that measure cumulative statistics shown in this report were last reset to zero (such as when the device was rebooted).

Examples

The following example displays LLDP statistics for Ethernet interface 0/13:

```
device# show counters lldp interface ethernet 0/13

LLDP Interface Statistics: ethernet 0/13
    FrameIn: 0
    FrameOut: 0
    FrameDiscard: 0
    FrameErrorIn: 0
    FrameErrorOut: 0
    TlvDiscard: 0
    TlvUnknown: 0
    LastClear: 13h1m34s
device#
```


show counters lldp summary

Use the **show counters lldp summary** command to display global (aggregated) link layer discovery protocol (LLDP) statistics for a device.

Syntax

```
show counters lldp summary
```

Parameters

This command has no keywords or variables.

Modes

EXEC (#)

Usage Guidelines

You must clear the counters if you want to start at zero for more recent statistics. For details, see [clear counters lldp](#) on page 517.

To view the LLDP statistics for one or more Ethernet interfaces, see [show counters lldp interface](#) on page 991.

Output

The **show counters lldp summary** command displays the following information:

Output field	Description
FrameIn	Number of LLDP frames received.
FrameOut	Number of LLDP frames sent.
EntriesAgedOut	Number LLDP messages sent by the device to discover its neighbors that were not received or processed by the intended recipients after a certain period of time. This can occur because of reasons such as network congestion, device restart, or incorrect configuration. As a result, the LLDP entries might become invalid or outdated, which causes inaccurate network mapping and potential connectivity issues.
FrameDiscard	Number of LLDP frames received that were discarded because of problems.
FrameErrorIn	Number of LLDP frames with errors received.

Output field	Description
TlvAccepted	Number of LLDP TLVs (information elements that contains the type, length, and value fields) received and then accepted. TLVs carry information about the link-layer capabilities of a device, such as the type of link-layer protocol supported, the maximum transmission unit (MTU), and the device's MAC address.
TlvDiscard	Number of LLDP TLVs received and then discarded.
TlvUnknown	Number of unrecognized LLDP TLVs received.
LastClear	Date and time (in ISO 8601 format) when the counters that measure cumulative statistics in this report were last reset to zero (such as when the device was rebooted).

Examples

The following example displays global LLDP statistics for the device:

```
device# show counters lldp summary

LLDP Global Statistics:
    FrameIn: 0
    FrameOut: 0
    EntriesAgedOut: 0
    FrameDiscard: 0
    FrameErrorIn: 0
    TlvAccepted: 0
    TlvDiscard: 0
    TlvUnknown: 0
    LastClear: 2021-01-12T09:38:02Z
device#
```

show counters mlag peer

Use the **show counters mlag peer** command to display statistics for multichassis link aggregation (MLAG).

Syntax

```
show counters mlag peer [ peer-name ]
```

Parameters

peer-name

(Optional) Displays statistics for the specified MLAG peer.

Modes

EXEC (#)

Usage Guidelines

You can use this command to display the MLAG keepalive counters for primary and secondary keepalives.

Output

The **show counters mlag peer** command displays the following information:

Output field	Description
Peer	MLAG peer to be display.
Transitions	Number of transitions per peer.
No. of Restarts	Number of times MLAG MS restarted.
Last clear	Last time when counters were cleared.
Keepalive	Keepalive name.
Transitions	Number of transitions per keepalive.
Receive	Number of MLAG keepalive packets received.
Transmit	Number of MLAG keepalive packets transmitted.

Examples

The following example displays statistics for the MLAG peer:

```
device# show counters mlag peer

Peer dut3
=====
Transitions      : 2
```

```
No. of Restarts : 2
Last clear      : 0000-00-00 00:00:00

  Keepalive kp-primary
  -----
  Transitions    : 1
  Receive        : 8  (BFD)
  Transmit       : 8  (BFD)
  Last clear     : 0000-00-00 00:00:00

  Keepalive kp-secondary
  -----
  Transitions    : 1
  Receive        : 199603
  Transmit       : 199603
  Last clear     : 0000-00-00 00:00:00
device#
```

show counters qos interface control-plane queue

Use the **show counters qos interface control-plane queue** command to display statistics for a specific CPU queue or all CPU queues on the control plane interface.

Syntax

```
show counters qos interface control-plane queue { queue-name | all }
```

Parameters

queue-name

Name of the QoS queue for which to display statistics. Use a string from q0 through q47 to specify one of the 48 queues. For example, q27.

all

Displays statistics for all QoS queues.

Modes

EXEC (#)

Usage Guidelines

To clear the statistics for a specific QoS queue or all QoS queues on the control plane interface, see [clear counters qos interface control-plane queue](#) on page 520.

Output

The **show counters qos interface control-plane queue** command displays the following information:

Output field	Description
interface-id	Name of the interface (in this case, the control plane interface) for which statistics are displayed.
Queue	Name of the QoS queue for which statistics are displayed. This is a string from q0 through q47 to indicate one of the 48 queues. For example, q27.
Rx Packets	The number of packets received by the interface.
Rx Octets	The number of octets (bytes) received by the interface.
Drop Packets	Number of packets that were dropped according to the QoS policy.
Drop Octets	Number of octets that were dropped according to the QoS policy.

Examples

The following example displays statistics for QoS queue q0:

```
device# show counters qos interface control-plane queue q0

-----
interface-id := control-plane
-----
Queue      Rx Packets Rx Octets  Drop Packets Drop Octets
=====
q0         0         0         0         0
device#
```

The following example displays statistics for all QoS queues:

```
device# show counters qos interface control-plane queue all

-----
interface-id := control-plane
-----
Queue      Rx Packets Rx Octets  Drop Packets Drop Packets
=====
q0         0         0         0         0
q1         0         0         0         0
q2         0         0         0         0
q3         0         0         0         0
q4         0         0         0         0
q5         0         0         0         0
q6         0         0         0         0
q7         0         0         0         0
q8         0         0         0         0
q9         0         0         0         0
q10        0         0         0         0
q11        0         0         0         0
q12        0         0         0         0
q13        0         0         0         0
q14        0         0         0         0
q15        0         0         0         0
q16        0         0         0         0
q17        0         0         0         0
q18        0         0         0         0
q19        0         0         0         0
q20        0         0         0         0
q21        0         0         0         0
q22        0         0         0         0
q23        0         0         0         0
q24        0         0         0         0
q25        0         0         0         0
q26        0         0         0         0
q27        0         0         0         0
q28        0         0         0         0
q29        0         0         0         0
q30        0         0         0         0
q31        0         0         0         0
q32        0         0         0         0
q33        0         0         0         0
q34        0         0         0         0
q35        0         0         0         0
q36        0         0         0         0
q37        0         0         0         0
q38        20        1360        0         0
q39        0         0         0         0
q40        0         0         0         0
q41        0         0         0         0
```

```
q42      0      0      0      0
q43      0      0      0      0
q44      0      0      0      0
q45      0      0      0      0
q46      0      0      0      0
q47      0      0      0      0
device#
```

show counters qos interface ethernet queue

Use the **show counters qos interface ethernet queue** command to display statistics for a specific QoS queue or all QoS queues on one or more Ethernet interfaces.

Syntax

```
show counters qos interface ethernet { interface-name | all } queue  
    { queue-name | all }
```

Parameters

interface-name

Ethernet interface name. Use the *slot/port* format or the *slot/port:breakout* format to specify an interface or one or more comma-separated ranges of interfaces respectively. For example, 0/1 or 1/1-3,5,2/7-9,10:1-4 respectively.

all

Shows QoS statistics for all Ethernet interfaces.

queue-name

Name of the QoS queue for which to display statistics. Use a string from q0 through q7 to specify one of the 8 queues. For example, q2.

all

Displays statistics for all QoS queues.

Modes

EXEC (#)

Usage Guidelines

You must clear the queues if you want to start at zero for more recent statistics. To clear the statistics for a specific QoS queue or all QoS queues on one or more Ethernet interfaces, see [clear counters qos interface ethernet queue](#) on page 521.

Output

The **show counters qos interface ethernet queue** command displays the following information:

Output field	Description
interface-id	Name of the interface for which statistics are displayed.
Queue	Name of the QoS queue for which statistics are displayed. This is a string from q0 through q47 to indicate one of the 48 queues. For example, q27.
Tx Packets	The number of packets transmitted by the interface.

Output field	Description
Tx Octets	The number of octets (bytes) transmitted by the interface.
Drop Packets	Number of packets that were dropped according to the QoS policy.
Drop Octets	Number of octets that were dropped according to the QoS policy.

Examples

The following example displays statistics for Ethernet interface 0/2 QoS queue q0:

```
device# show counters qos interface ethernet 0/2 queue q0

-----
interface-id := ethernet 0/2
-----
Queue      Tx Packets Tx Octets  Drop Packets Drop Octets
=====
q0         1945      318980    0             0
device#
```

The following example displays statistics for Ethernet interface 0/2 and all QoS queues:

```
device# show counters qos interface ethernet 0/2 queue all

-----
interface-id := ethernet 0/2
-----
Queue      Tx Packets Tx Octets  Drop Packets Drop Octets
=====
q0         1945      318980    0             0
q1          0           0         0             0
q2          0           0         0             0
q3          0           0         0             0
q4          0           0         0             0
q5          0           0         0             0
q6          0           0         0             0
q7          0           0         0             0
device#
```

show iah host

Use the **show iah host** command to display integrated application hosting (IAH) host information for virtual machines (VMs).

Syntax

```
show iah host
```

Parameters

This command has no keywords or variables.

Modes

EXEC (#)

Usage Guidelines

This command displays host information, maximum resources limits, resources used by VMs, and resources available for VMs.

If Extreme ONE OS reboots upon upgrade or downgrade, VM startup depends on the auto-boot configuration.

To display details about a specific VM, see [show iah vm](#) on page 1006.

Output

The **show iah host** command displays the following information:

Output field	Description
Host information	CPU topology of the Kernel-based Virtual Machine (KVM): • Name—hostname. • CPU model—The processor architecture of the CPU host “physical” machine (determined by the hypervisor) that is available for your guest VMs. • CPU frequency—Speed of the CPU host “physical” machine in megahertz (MHz). • CPU(s)—Total number of vCPUs. This number is calculated from the following formula: (threads x cores) x physical CPUs = Number vCPUs. • CPU socket(s)—Number of virtual processor sockets. A socket behaves as a physical CPU from a guest VM perspective. • Core(s) per socket:—Number of individual processing cores in each socket. • Thread(s) per core—Number of threads (logical CPUs) within each core. • NUMA cell(s)—Number of non-uniform memory access cells (also called NUMA nodes). CPU resources of a guest VM are isolated to a single NUMA cell. • Memory—Amount of RAM in kibibytes (KiBs). A KiB is 1,024 bytes. • Kernel version—Version of the Linux kernel that contains the KVM kernel module. • Hypervisor—Type of software emulator and virtualizer. • Hypervisor driver version—Version of the software layer that allows communication between the hypervisor and the VM. • Libvirt version—Version of the toolkit that is provided for managing the virtualization platform. KVM is a virtualization module that allows the Linux kernel to act as a bare-metal hypervisor. The CPU topology is the sockets, cores, and threads that are applied to the virtual CPUs for your guest VMs.
Platform resources limits for VMs	Overall resources allocated by the KVM to guest VMs. These resources exist in the separate computing space in which the guest VMs run. • VMs—Number of VMs allowed on the host. • vCPU(s)—Number of allocated vCPUs. • Memory—Amount of allocated RAM in KiBs. • Disk space—Amount of allocated storage.

Output field	Description
Platform resources used by VMs	Overall resources allocated by the KVM that are in use by configured guest VMs. • vCPU(s)—Number of vCPUs configured for all guest VMs. • Memory—Amount of RAM in KiBs configured for all guest VMs. • Disk space—Amount of storage configured for all guest VMs.
Platform resources available for VMs	Overall resources allocated by the KVM to guest VMs that are not in use. • vCPU(s)—Number of vCPUs not yet configured for use by guest VMs. • Memory—Amount of RAM in KiBs not yet configured for use by guest VMs. • Disk space—Amount of storage not yet configured for use by guest VMs.

Examples

The following example displays the IAH host information for VMs:

```
device# show iah host

Host information
  Name: ExtremeOS
  CPU model: x86_64
  CPU frequency: 1800 MHz
  CPU(s): 8
  CPU socket(s): 1
  Core(s) per socket: 4
  Thread(s) per core: 2
  NUMA cell(s): 1
  Memory: 16384 MiB
  Kernel version: Linux 5.4.0-117-generic
  Hypervisor: QEMU
  Hypervisor driver version: qemu-4.2.0
  Libvirt version: libvirt-8.3.0
  Platform resources limits for VMs:
    VMs: 4
    vCPU(s): 4
    Memory: 8192 MiB
    Disk space: 48000 MB
  Platform resources used by VMs:
    vCPU(s): 2
    Memory: 4096 MiB
    Disk space: 10000 MB
  Platform resources available for VMs:
    vCPU(s): 2
    Memory: 4096 MiB
    Disk space: 38000 MB
device#
```

show iah snapshot

Use the **show iah snapshot** command to display snapshot information about the integrated application hosting (IAH) third party virtual machine (TPVM).

Syntax

show iah snapshot *vm-name*

Parameters

vm-name

Name of a specific VM for which to display snapshot information. TPVM is the only VM supported in this release.

Modes

EXEC (#)

Examples

The following example displays snapshot information about the TPVM on the device:

```
device# show iah snapshot TPVM

Snapshot information
  Domain Version: 4.7.5
  Snapshot size: 9 GB
  Description: CLI
device#
```

show iah vm

Use the **show iah vm** command to display information about integrated application hosting (IAH) virtual machines (VMs).

Syntax

```
show iah vm { all | vm-name }
```

Parameters

all

Displays information about all VMs on the device.

vm-name

Name of a specific VM for which to display information. Use any combination of 1 to 32 alphanumeric characters, underscores, hyphens, or periods.

Modes

EXEC (#)

Output

The **show iah vm** command displays the following information:

Output field	Description
Name	Name that was assigned to the VM when it was configured.
deploy-status	Deployment status of the VM (such as DEPLOYED_SUCCESS).
vm-status	Operational status of the VM (such as RUNNING).
last-error	The latest error encountered.
description	Description that was assigned to the VM when it was configured.
auto-boot	Indicates whether auto-boot is enabled. If auto-boot is enabled, the VM starts automatically after the host reboots.
activate-network	Indicated whether the network interfaces are enabled on the VM.
cpus	Number of CPUs configured for the VM.
ram	Amount of memory in MiB (mebibytes) configured for the VM. One MiB is 1,048,576 bytes.

Output field	Description
disk	Disk name as well as the following settings configured for the VM: - size—Size of the disk in megabytes (MB). - file—Disk file name of the VM. The format is <code>disk://iah/<i>filename</i></code>. - file format—Disk file format of the VM. The possible values are <code>iso</code>, <code>qcow2</code> (QEMU copy-on-write), or <code>raw</code>. - bus type—Type of disk bus. The possible values are <code>ide</code>, <code>sata</code>, <code>scsi</code>, or <code>virtio</code> (Virtual I/O). The default is <code>virtio</code>.
network	Details about the network interfaces on the VM.
TPVM	Details about the IAH third party virtual machine (TPVM) such as the hostname, timezone, and IP address.

Examples

The following example displays information about all VMs on the device:

```
device# show iah vm all

Number of VMs: 2

Name: vm1
  deploy-status: DEPLOYED_SUCCESS
  vm-status: RUNNING
  last-error: Update of attribute ipv6-addr failed
  description: testvm1
  auto-boot: enabled
  activate-network: enabled
  cpus: 2
  ram: 2048 MiB
  disk: sda
    size: 10000 MB
    file: disk://iah/ubuntu22.04.qcow2
    file format: qcow2
    bus type: virtio
  networks: -

Name: vm2
  deploy-status: DEPLOYED_SUCCESS
  vm-status: RUNNING
  last-error:
  description: testvm2
  auto-boot: enabled
  activate-network: enabled
  cpus: 2
  ram: 2048 MiB
  disk: sda
    size: 10000 MB
    file: disk://iah/ubuntu22.04.qcow2
    file format: qcow2
    bus type: virtio
  network: 1
    interface: management 0
    connector-type: MANAGEMENT
    mac: e4:44:e5:08:ac:00
device#
```

The following example displays information about a VM named vm2 on the device:

```
device# show iah vm vm2

Name: vm2
  deploy-status: DEPLOYED_SUCCESS
  vm-status: RUNNING
  last-error:
  description: testvm2
  auto-boot: enabled
  activate-network: enabled
  cpus: 2
  ram: 2048 MiB
  disk: sda
    size: 10000 MB
    file: disk://iah/ubuntu22.04.qcow2
    file format: qcow2
    bus type: virtio
  network: 1
    interface: management 0
    connector-type: MANAGEMENT
    mac: e4:44:e5:08:ac:00
device#
```

The following example displays information about the TPVM on the device:

```
device# show iah vm all details

Number of VMs: 1
Name: TPVM
  deploy-status: DEPLOYED_SUCCESS
  vm-status: RUNNING
  last-error: -
  description: TPVM
  auto-boot: enabled
  activate-network: enabled
  cpus: 4
  ram: 8192 MiB
  disk: sda
    size: 10240 MB
    file: /mnt/onl/iah/images/TPVM/sda.qcow2
    file format: qcow2
    bus type: virtio
  disk: sdb
    size: 51200 MB
    file: /mnt/onl/iah/images/TPVM/sdb.qcow2
    file format: qcow2
    bus type: virtio
  disk: hda
    size: 1 MB
    file: /mnt/onl/iah/images/TPVM/hda.raw
    file format: raw
    bus type: ide
  network: mgmt0
    interface: management 0
    connector-type: MANAGEMENT
    mac: -
  TPVM:
    hostname: host123
    user: extreme
    timezone: Asia/Kolkata
    version: 4.7.6
    ssh-passwordless: enabled
    ipaddress: 10.32.83.114/24
```



```
gateway: 10.32.83.1
ipv6address: 2620:100:c:e083:10:32:83:114/64
gatewayv6: 2620:100:c:e083:10:32:83:1
dns-primary-server: 134.141.79.202
dns-secondary-server: 8.8.8.8
dns-domain: corp.extremenetworks.com
peer-ip: 10.32.83.113
peer-ipv6: -
peer-password: UUU09KogIYED-TIc_biqM3F7zLqyfcSJ
peer-sudoer: extreme
host: 10.32.89.183
port: 389
basedn: dc=example,dc=com
rootdn: cn=admin,dc=example,dc=com
rootdnpw: glFI1UIKnQgCcs04k7T5alNVinl_bd3o
secure: false
ntp:
  server: 10.64.12.81
  server: 10.64.13.105
  server: 10.32.12.81
  server: 10.38.99.47
  server: time1.google.com
device#
```

show interface ethernet

Use the **show interface ethernet** command to display the status of the Ethernet interfaces on a device.

Syntax

```
show interface ethernet [ interface-name ] [ brief ]
```

Parameters

interface-name

(Optional) Interface name. Use the *slot/port* format or the *slot/port:breakout* format to specify an interface or one or more comma-separated ranges of interfaces respectively. For example, 0/1 or 1/1-3,5,2/7-9,10:1-4 respectively.

brief

(Optional) Displays the information in brief format.

Modes

EXEC (#)

Usage Guidelines

For SDWAN platforms, the FCS and CRC error information is not displayed.

Examples

The following example shows the status of Ethernet interface 0/51:

```
device# show interface ethernet 0/51
ethernet 0/51 Admin state UP, Oper state UP
  Interface index is 16778848 (0x1000660)
  MTU 9216 bytes
  Hardware is Ethernet  mac address 20:6c:8a:de:2c:3b
  Current Speed 100G
  Description: Ethernet 0/51
  FEC:
    Mode: RS-FEC
    Uncorrected Blocks: 0
    Corrected Blocks: 0
  Link Fault Signaling: ON
  Link Fault Status: No Fault
  Auto Negotiation: OFF
  Carrier Transitions: 3
    LastClear: 0s
  Input:
    Total pkts: 4
    Broadcast pkts: 0
    Discard pkts: 0
    Errors pkts: 0
    FCS Errors: 0
    MCast pkts: 4
```

```

        Octets: 748
        UCast pkts: 0
        Runt pkts: 0
        CRC Errors: 0
Input Distribution:
    64 byte pkts: 0
    65-127 byte pkts: 0
    128-255 byte pkts: 4
    256-511 byte pkts: 0
    512-1023 byte pkts: 0
    1024-1518 byte pkts: 0
    Jumbo pkts: 0
Out:
    Total pkts: 4
    Broadcast pkts: 0
    Discard pkts: 0
    Errors pkts: 0
    MCast pkts: 4
        Octets: 748
        UCast pkts: 0
Rate Info:
    Input: 0.000087 Mbits/sec, 0 pkts/sec 0.00% of line-rate
    Output: 0.000087 Mbits/sec, 0 pkts/sec 0.00% of line-rate
```

Link Fault Status displays one of the following:

- No Fault—Port is administratively down
- Local fault—Port does not detect the laser
- Remote fault—Port received a remote fault
- No fault—Port is operationally up, and no faults are found

show interface ethernet link-error-disable

Use the **show interface ethernet link-error-disable** command to display the link dampening status and dampening configuration for each port on a device.

Syntax

```
show interface ethernet link-error-disable
```

Parameters

This command has no keywords or variables.

Modes

EXEC (#)

Usage Guidelines

Use this command to show the link dampening enabled/disabled status and the dampening configuration for all front ports on a device.

Examples

The following example shows the link dampening status and the dampening configuration for each port. In this example, link dampening is enabled and configured only on Ethernet port 0/1:

```
device# show interface ethernet link-error-disable
```

Port Updated	Link Error Disable	Toggle-Threshold	Sampling Time	Wait Time	Status	Last
=====	=====	=====	=====	=====	=====	=====
Eth 0	Disable	0	0	0	0	
2025-10-09T17:52:13+05:30						
Eth 0/1	Enable	10	3	300	0	
2025-10-09T17:34:24+05:30						
Eth 0/2	Disable	0	0	0	0	
2025-10-09T17:34:25+05:30						
Eth 0/3	Disable	0	0	0	0	
2025-10-09T18:01:28+05:30						
Eth 0/4	Disable	0	0	0	0	
2025-10-09T18:01:28+05:30						
Eth 0/5	Disable	0	0	0	0	
2025-10-09T18:01:28+05:30						
Eth 0/6	Disable	0	0	0	0	
2025-10-09T18:01:28+05:30						

```
device#
```

show interface loopback-mode

Use the **show interface loopback-mode** command to display whether loopback mode is enabled for the interfaces on a device.

Syntax

```
show interface loopback-mode
```

Parameters

This command has no keywords or variables.

Modes

EXEC (#)

Output

The **show interface loopback-mode** command displays the following information:

Output field	Description
Port	Name of the interface.
Loop-back mode	Indicates whether loopback mode is enabled or disabled on the interface.

Examples

The following example shows whether loopback mode is enabled for the interfaces on the device:

```
device# show interface loopback-mode

Port          Loop-back mode
=====
Eth 0         OFF
Eth 0/1       OFF
Eth 0/2       OFF
Eth 0/3       OFF
Eth 0/4       OFF
Eth 0/5       OFF
Eth 0/6       OFF
Eth 0/7       OFF
Eth 0/8       OFF
Eth 0/9       OFF
Eth 0/10      OFF
Eth 0/11      OFF
Eth 0/12      OFF
Eth 0/13      OFF
Eth 0/14      OFF
Eth 0/15      OFF
Eth 0/16      OFF
device#
```

show interface subinterface

Use the **show interface subinterface** command to display information about the subinterfaces under the specified interface.

Syntax

```
show interface { ethernet interface-name | port-channel port-channel-number } subinterface
```

Parameters

ethernet *interface-name*

Ethernet interface containing one or more subinterfaces. Use the *slot/port* format or the *slot/port:breakout* format to specify an interface or one or more comma-separated ranges of interfaces respectively. For example, 0/1 or 1/1-3,5,2/7-9,10:1-4 respectively.

port-channel *port-channel-number*

Port channel containing one or more subinterfaces. For the port number, use an integer from 1 to 256.

Modes

EXEC (#)

Output

The **show interface subinterface** command displays the following information:

Output field	Description
Interface <i>type name</i>	Type and name of the parent interface.
Vlan	VLAN range of the subinterface.
Inner Vlan	Inner VLAN range of the subinterface. A dash (-) indicates that no inner VLAN is configured.
Admin status	Administrative state of the subinterface. This is either UP or DOWN (forced administratively down indefinitely).
Oper status	Indicates whether the subinterface is UP or DOWN.
If Mode	Layer in which the subinterface operates (Layer 2 or Layer 3).

Examples

The following example displays information about the subinterfaces under Ethernet interface 0/1:

```
device# show interface ethernet 0/1:1-3 subinterface

Interface ethernet 0/1:1
Vlan      Inner Vlan Admin Status Oper Status If Mode
=====
45        0          UP          UP          IF_MODE_L2Interface ethernet 0/1:2
Vlan      Inner Vlan Admin Status Oper Status If Mode
=====
untag(45) -          UP          UP          IF_MODE_L2Interface ethernet 0/1:3
Vlan      Inner Vlan Admin Status Oper Status If Mode
=====
untag(45) -          UP          UP          IF_MODE_L2
device#
```

show interface tunnel (Extreme ONE OS Switching)

Use the **show interface tunnel** command to display information about the tunnel interfaces on a device.

Syntax

```
show interface tunnel { brief | interface-name }
```

Parameters

brief

Displays brief information about all tunnel interfaces on the device.

interface-name

Name of the interface for which to display detailed information. Use an integer or one or more comma-separated ranges of integers. For example, 4 or 4-5,5,7-9.

Modes

EXEC (#)

Examples

The following example displays brief information about all tunnel interfaces on the device:

```
device# show interface tunnel brief

Tunnel: tunnel 1  Type: vxlan-tunnel  Admin state: UP      Operational state: UP
  Ifindex: 0x3100002  Config source: STATIC
  Source NVE: base
  Destination: 12.1.1.1
Tunnel: tunnel 2  Type: vxlan-tunnel  Admin state: UP      Operational state: UP
  Ifindex: 0x3100001  Config source: STATIC
  Source NVE: base
  Destination: 12.1.1.2
Tunnel: tunnel 3  Type: vxlan-tunnel  Admin state: UP      Operational state: UP
  Ifindex: 0x3100004  Config source: STATIC
  Source NVE: Nve2
  Destination: 12.1.1.3
Tunnel: tunnel 4  Type: vxlan-tunnel  Admin state: UP      Operational state: UP
  Ifindex: 0x3100003  Config source: STATIC
  Source NVE: Nve3
  Destination: 12.1.1.4
device#
```

The following example displays detailed information about tunnel interface 5 on the device:

```
device# show interface tunnel 5

Tunnel: tunnel 5  Type: vxlan-tunnel  Admin state: UP      Operational state: UP
  Ifindex: 0x3100007  Config source: STATIC
  Source NVE: base
```



```
Destination: 55::55
Nexthops:
  IP 101::2      VRF default-vrf      L3 interface ethernet 0/9:1.0
  L2 interface ethernet 0/9:1.0      Bridge domain 0
  Mac Address 00:16:3e:17:f0:0a
device#
```

show ipv4 interface

Use the **show ipv4 interface** command to display information about the IPv4 interfaces on a device.

Syntax

```
show ipv4 interface [ ethernet interface-name | loopback port-number
| port-channel port-channel-number | tunnel interface-name | ve
interface-name | internal interface-name ] [ brief ]
```

Parameters

ethernet *interface-name*

(Optional) Ethernet interface name. Use the *slot/port* format or the *slot/port:breakout* format to specify an interface or one or more comma-separated ranges of interfaces respectively. For example, 0/1 or 1/1-3,5,2/7-9,10:1-4 respectively.

loopback *port-number*

(Optional) Loopback port number. Use an integer from 1 to 1024.

port-channel *port-channel-number*

(Optional) Link aggregation group (LAG) interface port channel number. Use an integer or one or more comma-separated ranges of integers from 1 to 255. For example, 1 or 1-3,5,7-9.

tunnel *interface-name*

(Optional) Tunnel interface name. Use an integer or one or more comma-separated ranges of integers from 1 to 100000. For example, 4 or 4-5,5,7-9.

ve *interface-name*

(Optional) Virtual Ethernet (VE) interface name. Use an integer or one or more comma-separated ranges of integers from 1 to 8192. For example, 2 or 2-3,5,7-9.

internal *interface-name*

(Optional) Internal interface name. Use integers 0 or 1.

brief

(Optional) Displays brief information for all IPv4 interfaces.

Modes

EXEC (#)

Examples

The following example shows brief information about all IPv4 interfaces on the device:

```
device# show ipv4 interface brief

L3 Interface IP-Address Vrf Admin status Oper status Address status
=====
```

```

Eth 0/13:1 1.1.1.1 vrf1 UP UP PREFERRED
Eth 0/13:2.2 2.2.2.1 vrf1 UP UP PREFERRED
Po 13 13.13.13.1 vrf1 UP UP PREFERRED
Ve 3 3.3.3.1 vrf1 UP UP PREFERRED
Ve 4 4.4.4.1 vrf1 UP UP PREFERRED
Ve 5 5.5.5.1 vrf1 UP UP PREFERRED
Ve 11 11.11.11.1 vrf1 UP UP PREFERRED
Ve 12 12.12.12.1 vrf1 UP UP PREFERRED
Ve 14 14.14.14.1 vrf1 UP UP PREFERRED
Lo 1 34.34.34.34 vrf1 UP UP PREFERRED
device#

```

The following example shows detailed information about IPv4 loopback interface 1 on the device:

```

device# show ipv4 interface loopback 1

Interface: Lo 1 Admin-status:UP Oper-status:UP
IP MTU: 1500
Vrf: vrf1
Address      Status
=====
34.34.34.34/32 PREFERRED
device#

```

The following example shows detailed information about IPv4 VE interface 3 on the device:

```

device# show ipv4 interface ve 3

Interface: ve 3 Admin-status:UP Oper-status:UP
Hardware is Virtual Ethernet, address is b4:a3:bd:0e:54:18
IP MTU: 1500
Vrf: vrf1
Address      Status
=====
3.3.3.1/24 PREFERRED
device#

```

The following example shows detailed information about IPv4 LAG port number 13 on the device:

```

device# show ipv4 interface port-channel 13

Interface: Po 13 Admin-status:UP Oper-status:UP
IP MTU: 1500
Vrf: vrf1
Address      Status
=====
13.13.13.1/24 PREFERRED
device#

```

show ipv6 interface

Use the **show ipv6 interface** command to display information about the IPv6 interfaces on a device.

Syntax

```
show ipv6 interface [ ethernet interface-name | loopback port-number
| port-channel port-channel-number | tunnel interface-name | ve
interface-name | internal interface-name ] [ brief ]
```

Parameters

ethernet *interface-name*

(Optional) Ethernet interface name. Use the *slot/port* format or the *slot/port:breakout* format to specify an interface or one or more comma-separated ranges of interfaces respectively. For example, 0/1 or 1/1-3,5,2/7-9,10:1-4 respectively.

loopback *port-number*

(Optional) Loopback port number. Use an integer from 1 to 1024.

port-channel *port-channel-number*

(Optional) Link aggregation group (LAG) interface port channel number. Use an integer or one or more comma-separated ranges of integers from 1 to 255. For example, 1 or 1-3,5,7-9.

tunnel *interface-name*

(Optional) Tunnel interface name. Use an integer or one or more comma-separated ranges of integers from 1 to 100000. For example, 4 or 4-5,5,7-9.

ve *interface-name*

(Optional) Virtual Ethernet (VE) interface name. Use an integer or one or more comma-separated ranges of integers from 1 to 8192. For example, 2 or 2-3,5,7-9.

internal *interface-name*

(Optional) Internal interface name. Use integers 0 or 1.

brief

(Optional) Displays brief information for all IPv6 interfaces.

Modes

EXEC (#)

Examples

The following example shows brief information about all IPv6 interfaces on the device:

```
device# show ipv6 interface brief

L3 Interface IPv6-Address Vrf  Admin status Oper status Address status
=====
```

```

Eth 0/13:1  1:1:1::1      vrf1 UP          UP          PREFERRED
Eth 0/13:2.2 2:2:2::1      vrf1 UP          UP          PREFERRED
Po 13       13:13:13::1   vrf1 UP          UP          PREFERRED
Ve 3        3:3:3::1     vrf1 UP          UP          PREFERRED
Ve 4        4:4:4::1     vrf1 UP          UP          PREFERRED
Ve 5        5:5:5::1     vrf1 UP          UP          PREFERRED
Ve 11       11:11:11::1  vrf1 UP          UP          PREFERRED
Ve 12       12:12:12::1  vrf1 UP          UP          PREFERRED
Ve 14       14:14:14::1  vrf1 UP          UP          PREFERRED
device#

```

The following example shows detailed information about IPv6 Ethernet interface 0/13:1 on the device:

```

device# show ipv6 interface eth 0/13:1

Interface: Eth 0/13:1 Admin-status:UP Oper-status:UP
IP MTU: 1500
Vrf: vrf1
Address                               Status
=====
1:1:1:1::1/64                        PREFERRED
fe80::b6a3:bdff:fe0e:5427/64        PREFERRED
device#

```

The following example shows detailed information about IPv6 LAG interface 13 on the device:

```

device# show ipv6 interface port-channel 13

Interface: Po 13 Admin-status:UP Oper-status:UP
IP MTU: 1500
Vrf: vrf1
Address                               Status
=====
13:13:13::1/64                      PREFERRED
fe80::b6a3:bdff:fe0e:5441/64        PREFERRED
device#

```

show ipv4 neighbor interface

Use the **show ipv4 neighbor interface** command to display the address resolution protocol (ARP) entries in the neighbor cache for a specific interface.

Syntax

```
show ipv4 neighbor interface { ethernet interface-name | port-channel
    port-channel-range | tunnel interface-number | ve interface-name }
```

Parameters

ethernet *interface-name*

Filters entries by Ethernet interface name. Use the *slot/port* format or the *slot/port:breakout* format to specify an interface. For example, 0/1.

port-channel *port-channel-range*

Filters entries by port channel number. Use an integer from 1 to 255.

tunnel *interface-number*

Filters entries by tunnel interface number. Use an integer from 1 to 100000.

ve *interface-name*

Filters entries by Virtual Ethernet (VE) interface name. Use an integer from 1 to 8192.

Modes

EXEC (#)

Usage Guidelines

ARP establishes correspondences between pairs of network-layer addresses (Layer 3) and LAN hardware addresses (Layer 2 MAC addresses). Each correspondence is recorded in a cache on the device for a predetermined period and then deleted.

Examples

The following example displays ARP entries for VE interface 1 on the device:

```
device# show ipv4 neighbor interface ve 1

-----
Interface Name:  _bridge_domain 1
IP Address      MAC Address      Type
Interface       L2 Interface      Age
=====
-----
111.0.0.1       88:7e:25:d3:da:07    MLAG-Static      ve
1               tunnel ISL_30.0.0.1  16h9m54s
111.0.0.2       88:7e:25:d3:ca:07    Local            ve
1               -                    16h10m52s
```

```

111.0.0.5      00:00:11:22:00:61      Dynamic      ve
1      ethernet 0/1:1.30      5m32s
111.0.0.6      00:00:11:22:00:62      MLAG      ve
1      tunnel ISL_30.0.0.1      13h42m50s
111.0.0.11     00:00:11:22:00:24      MLAG      ve
1      port-channel 101.4088      12h59m34s
111.0.0.22     00:00:11:22:00:13      Dynamic      ve
1      port-channel 102.409610      4m27s
111.0.0.55     00:00:11:22:00:27      Dynamic      ve
1      ethernet 0/15:1.30      11m29s
111.0.0.56     00:00:11:22:00:06      Dynamic      ve
1      port-channel 101.4088      4m18s
111.0.0.66     00:00:11:22:00:28      Dynamic      ve
1      ethernet 0/1:1.30      4m30s
111.0.0.72     00:00:11:22:00:50      MLAG      ve
1      tunnel ISL_30.0.0.1      13h42m50s
111.0.0.99     00:00:11:22:00:14      Dynamic      ve
1      port-channel 200.400      4m24s
111.0.0.112    00:00:11:22:00:5d      Dynamic      ve
1      port-channel 102.409610      11m29s
111.0.0.118    00:00:11:22:00:5b      MLAG      ve
1      port-channel 101.4088      13h7m6s
111.0.0.253    02:e0:52:00:01:00      Local      ve
1      -      16h10m52s
device#

```

show ipv4 neighbor vrf

Use the **show ipv4 neighbor vrf** command to display the address resolution protocol (ARP) entries in the neighbor caches for Virtual Routing and Forwarding (VRF) instances.

Syntax

```
show ipv4 neighbor vrf { all | vrf-name } [ ipv4-address ]
```

Parameters

vrf { **all** | *vrf-name* }

- **all** displays ARP entries for all VRF instances.
- *vrf-name* displays ARP entries for a specific VRF instance. This name must start with an alphabetic character or an underscore followed by an arbitrary sequence of alphabetic or numeric characters, underscores, hyphens, or dots. The name cannot exceed 64 characters. Do not use the profile reserved keyword.

ipv4-address

Filter ARP entries by the IPv4 address of a neighboring device. Use the *a.b.c.d* format.

Modes

EXEC (#)

Usage Guidelines

ARP establishes correspondences between pairs of network-layer addresses (Layer 3) and LAN hardware addresses (Layer 2 MAC addresses). Each correspondence is recorded in a cache on the device for a predetermined period and then deleted. Examining the ARP cache can be a quick way to view or verify hosts that are connected directly to the device.

Output

The **show ipv4 neighbor vrf** command displays the following information:

Output field	Description
Vrf or VRF instance	Name of a VRF instance that is configured on the device.
Total number of arp entries	Number of ARP entries in the neighbor cache for the corresponding VRF.
Ip-address or IP Address	Network address of the neighboring device that corresponds to the LAN hardware address.

Output field	Description
Mac-address or MAC Address	LAN hardware address (data link address) of the MAC address of the neighboring device that corresponds to the network address.
Type	Current state (either dynamic or static) of the ARP entry (the Ethernet MAC to IPv4 address link) in the cache. Most ARP entries are dynamic, which means that they remain on a device for a certain time, if they are in use. ARP automatically creates and updates dynamic ARP entries (which means that they are learned on an interface). A dynamic ARP entry is deleted when it ages out, or when the output interface goes down. A dynamic ARP entry can be overwritten by a static ARP entry. A static ARP entry is configured manually. It does not age out and cannot be overwritten by a dynamic ARP entry. A static ARP entry protects communication between devices, because attack packets cannot modify its IP-to-MAC mapping. ARP entries that correspond to the local IP address on the interface never age out.
Interface	Logical interface to which this network address mapping is assigned. This is either the VE interface or the Layer 2 interface (if no VE interface is mapped to this interface).
L2 Interface	Layer 2 (physical) interface to which this network MAC address is associated.
Age	Age expressed in standard time format (hour, minutes, and seconds) of the ARP entry in the cache.
Seq No.	Sequence number. This number determines which “type” is selected as the best route for a given IP address. The best type is indicated by an asterisk (*).

Examples

The following example displays ARP entries for all VRF instances on the device:

```
device# show ipv4 neighbor vrf
```

```
Vrf:default-vrf
```

```
-----
```

```
Total number of arp entries: 4
```

Ip-address	Mac-address	Type	Interface	L2 Interface	Age
=====	=====	=====	=====	=====	
101.1.1.2	aa:aa:aa:aa:aa:aa	Dynamic	ethernet 0/1:1	ethernet 0/1:1	21
(s)					
101.1.1.3	bb:bb:bb:bb:bb:bb	Static	ethernet 0/1:1	ethernet 0/1:1	31
(s)					
102.1.1.4	cc:cc:cc:cc:cc:cc	Dynamic	ethernet 0/1:3	ethernet 0/1:3	41
(s)					
103.1.1.5	dd:dd:dd:dd:dd:dd	static	ve 100	ethernet 0/1:4	51
(s)					

```
Vrf:vrf1
```

```

-----
Total number of arp entries: 4

Ip-address   Mac-address           Type      Interface      L2 Interface      Age
=====
11.1.1.2     aa:aa:aa:aa:aa:a1     Dynamic   ethernet 0/2:1  ethernet 0/1:1     21 (s)
11.1.1.3     bb:bb:bb:bb:bb:b1     Static    ethernet 0/2:1  ethernet 0/1:1     31 (s)
12.1.1.4     cc:cc:cc:cc:cc:c1     Dynamic   ethernet 0/2:3  ethernet 0/1:3     41 (s)
13.1.1.5     dd:dd:dd:dd:dd:d1     static    ve 101          ethernet 0/1:4     51 (s)
device#

```

The following example displays ARP entries for a VRF instance named vrf1 on the device:

```

device# show ipv4 neighbor vrf vrf1

Vrf:vrf1
-----
Total number of arp entries: 4

Ip-address   Mac-address           Type      Interface      L2 Interface      Age
=====
11.1.1.2     aa:aa:aa:aa:aa:a1     Dynamic   ethernet 0/2:1  ethernet 0/1:1     21 (s)
11.1.1.3     bb:bb:bb:bb:bb:b1     Static    ethernet 0/2:1  ethernet 0/1:1     31 (s)
12.1.1.4     cc:cc:cc:cc:cc:c1     Dynamic   ethernet 0/2:3  ethernet 0/1:3     41 (s)
13.1.1.5     dd:dd:dd:dd:dd:d1     static    ve 101          ethernet 0/1:4     51 (s)
device#

```

The following example displays ARP entries for a VRF instance named vrf1 on the device:

```

device# show ipv4 neighbor vrf vrf1 interface ve 100

Vrf:vrf1
-----
Total number of arp entries: 1

Ip-address   Mac-address           Type      Interface      L2 Interface      Age
=====
111.1.1.2    aa:aa:aa:aa:aa:a1     Dynamic   ve 100          ethernet 0/2:1     21 (s)
device#

```

The following example displays ARP entries for a VRF instance named vrf1 on the device. In this example, the entries are filtered by the IPv4 address of a neighboring device:

```

device# show ipv4 neighbor vrf vrf1 101.1.1.2

IP Address   : 101.1.1.2
MAC Address   : aa:aa:aa:aa:aa:aa

Source       : Local
=====
Seq No.      : 10                      Age      : 21 (s)
Type         : Dynamic                 VRF instance : default-vrf
Interface    : ethernet 0/1:1          Index      : 1048580
L2 Interface : ethernet 0/1:1

Source       : BGP
=====
Seq No.      : 101                      Age      : 21 (s)
Type         : Dynamic                 VRF instance : default-vrf
Interface    : Ve 100                  Index      : 1048581

```

```
L2 Interface : ethernet 0/1:4  
device#
```

show ipv6 neighbor interface

Use the **show ipv6 neighbor interface** command to display the address resolution protocol (ARP) entries in the neighbor cache for a specific interface.

Syntax

```
show ipv6 neighbor interface { ethernet interface-name | port-channel
    port-channel-range | tunnel interface-number | ve interface-name }
```

Parameters

ethernet *interface-name*

Filters entries by Ethernet interface name. Use the *slot/port* format or the *slot/port:breakout* format to specify an interface. For example, 0/1.

port-channel *port-channel-range*

Filters entries by port channel number. Use an integer from 1 to 255.

tunnel *interface-number*

Filters entries by tunnel interface number. Use an integer from 1 to 100000.

ve *interface-name*

Filters entries by Virtual Ethernet (VE) interface name. Use an integer from 1 to 8192.

Modes

EXEC (#)

Usage Guidelines

ARP establishes correspondences between pairs of network-layer addresses (Layer 3) and LAN hardware addresses (Layer 2 MAC addresses). Each correspondence is recorded in a cache on the device for a predetermined period and then deleted.

Examples

The following example displays ARP entries for Ethernet interface 0/1 on the device:

```
device# show ipv6 neighbor interface ve 1

-----
Interface Name:  _bridge_domain 1
IP Address      MAC Address      Type
Interface      L2 Interface      Age
=====
-----
1111:1111:1111:1111:1111:1111:1  02:e0:52:00:01:00      Local      ve
1                                -      16h16m0s
1111:1111:1111:1111:1111:1111:9  00:00:11:22:00:1d      Dynamic    ve
1                                ethernet 0/1:1.30      9m35s
```

```

1111:1111:1111:1111:1111:1111:1111:11 00:00:11:22:00:16      MLAG      ve
1      tunnel ISL_30.0.0.1      13h47m57s
1111:1111:1111:1111:1111:1111:1111:22 00:00:11:22:00:17      MLAG      ve
1      port-channel 101.4088      13h47m57s
1111:1111:1111:1111:1111:1111:1111:9c 00:00:11:22:00:a6      MLAG      ve
1      tunnel ISL_30.0.0.1      13h47m57s
1111:1111:1111:1111:1111:1111:1111:120 00:00:11:22:00:61      Dynamic    ve
1      ethernet 0/1:1.30      2m0s
1111:1111:1111:1111:1111:1111:1111:121 00:00:11:22:00:62      MLAG      ve
1      tunnel ISL_30.0.0.1      13h47m57s
1111:1111:1111:1111:1111:1111:1111:123 00:00:11:22:00:06      MLAG      ve
1      port-channel 101.4088      13h47m57s
1111:1111:1111:1111:1111:1111:1111:889 00:00:11:22:00:18      Dynamic    ve
1      port-channel 200.400      9m34s
1111:1111:1111:1111:1111:1111:1111:1111 88:7e:25:d3:da:07      MLAG-Static ve
1      tunnel ISL_30.0.0.1      16h15m1s
1111:1111:1111:1111:1111:1111:1111:1122 00:00:11:22:00:15      Dynamic    ve
1      ethernet 0/15:1.30      9m35s
1111:1111:1111:1111:1111:1111:1111:2211 00:00:11:22:00:5b      MLAG      ve
1      port-channel 101.4088      13h12m13s
1111:1111:1111:1111:1111:1111:1111:2222 88:7e:25:d3:ca:07      Local      ve
1      -      16h16m0s
1111:1111:1111:1111:1111:1111:1111:3345 00:00:11:22:00:3b      MLAG      ve
1      port-channel 102.409610      13h47m57s
fe80::8a7e:25ff:fed3:ca07      88:7e:25:d3:ca:07      Local      ve
1      -      16h16m0s
fe80::8a7e:25ff:fed3:da07      88:7e:25:d3:da:07      MLAG-Static ve
1      tunnel ISL_30.0.0.1      16h15m1s
device#

```

show ipv6 neighbor vrf

Use the **show ipv6 neighbor vrf** command to display the neighbor discovery protocol (NDP) entries in the neighbor caches for Virtual Routing and Forwarding (VRF) instances.

Syntax

```
show ipv6 neighbor vrf { all | vrf-name } [ ipv6-address ]
```

Parameters

vrf { **all** | *vrf-name* }

- **all** displays NDP entries for all VRF instances.
- *vrf-name* displays NDP entries for a specific VRF instance. This name must start with an alphabetic character or an underscore followed by an arbitrary sequence of alphabetic or numeric characters, underscores, hyphens, or dots. The name cannot exceed 64 characters. Do not use the profile reserved keyword.

ipv6-address

Filter NDP entries by the IPv6 address of a neighboring device. Use the *x::x::x* format.

Modes

EXEC (#)

Usage Guidelines

NDP establishes correspondences between pairs of network-layer addresses (Layer 3) and LAN hardware addresses (Layer 2 MAC addresses). Each correspondence is recorded in a cache on the device for a predetermined period and then deleted. Examining the NDP cache can be a quick way to view or verify hosts that are connected directly to the device.

Output

The **show ipv6 neighbor vrf** command displays the following information:

Output field	Description
Vrf or VRF instance	Name of a VRF instance that is configured on the device.
Total number of neighbor entries	Number of NDP entries in the neighbor cache for the corresponding VRF.
Ip-address or IP Address	Network address of the neighboring device that corresponds to the LAN hardware address.

Output field	Description
Mac-address or MAC Address	LAN hardware address (data link address) of the MAC address of the neighboring device that corresponds to the network address.
Type	Current state (either dynamic or static) of the NDP entry (the Ethernet MAC to IPv6 address link) in the cache. Most NDP entries are dynamic, which means that they remain on a device for a certain time, if they are in use. NDP automatically creates and updates dynamic NDP entries (which means that they are learned on an interface). A dynamic NDP entry is deleted when its ages out, or when the output interface goes down. A dynamic NDP entry can be overwritten by a static NDP entry. A static NDP entry is configured manually. It does not age out and cannot be overwritten by a dynamic NDP entry. A static NDP entry protects communication between devices, because attack packets cannot modify its IP-to-MAC mapping. NDP entries that correspond to the local interface never age out.
Interface	Logical interface to which this network address mapping is assigned. This is either the VE interface or the Layer 2 interface (if no VE interface is mapped to this interface).
L2 Interface	Layer 2 (physical) interface to which this network address mapping is assigned.
Age	Age (in seconds) of the NDP entry in the cache.
Seq No.	

Examples

The following example displays NDP entries for all VRF instances on the device:

```
device# show ipv6 neighbor vrf all

Vrf: default-vrf
-----
Total number of neighbor entries: 4

Ipv6 address      Mac-address      Type      Interface      L2 Interface      Age
=====
1111::1           aa:aa:aa:aa:aa:aa Dynamic    ethernet 0/1:1  ethernet 0/1:1    21 (s)
1112::2           bb:bb:bb:bb:bb:bb Static     ethernet 0/1:1  ethernet 0/1:1    31 (s)
1113::3           cc:cc:cc:cc:cc:cc Dynamic    ethernet 0/1:3  ethernet 0/1:3    41 (s)
1114::4           dd:dd:dd:dd:dd:dd Static     ve 100         ethernet 0/1:4    51 (s)

Vrf :vrf1
-----
Total number of neighbor entries: 4

Ipv6 address      Mac-address      Type      Interface      L2 Interface      Age
=====
1111::1           aa:aa:aa:aa:aa:aa Dynamic    ethernet 0/1:1  ethernet 0/1:1    21 (s)
1112::2           bb:bb:bb:bb:bb:bb Static     ethernet 0/1:1  ethernet 0/1:1    31 (s)
1113::3           cc:cc:cc:cc:cc:cc Dynamic    ethernet 0/1:3  ethernet 0/1:3    41 (s)
```

```

1114::4          dd:dd:dd:dd:dd:dd  Static    ve 100          ethernet 0/1:4   51 (s)
device#

```

The following example displays NDP entries for a VRF instance named `vrf1` on the device:

```

device# show ipv6 neighbor vrf vrf1

Vrf :vrf1
-----
Total number of neighbor entries: 4

Ipv6 address      Mac-address        Type      Interface        L2 Interface      Age
=====
1111::1           aa:aa:aa:aa:aa:aa  Dynamic   ethernet 0/1:1    ethernet 0/1:1     21 (s)
1112::2           bb:bb:bb:bb:bb:bb  Static    ethernet 0/1:1    ethernet 0/1:1     31 (s)
1113::3           cc:cc:cc:cc:cc:cc  Dynamic   ethernet 0/1:3    ethernet 0/1:3     41 (s)
1114::4           dd:dd:dd:dd:dd:dd  Static    ve 100            ethernet 0/1:4     51 (s)
device#

```

The following example displays NDP entries for a VRF instance named `vrf1` on the device:

```

device# show ipv6 neighbor vrf vrf1 interface ve 100

Vrf : vrf1
-----
Total number of neighbor entries: 1

Ipv6 address      Mac-address        Type      Interface        L2 Interface      Age
=====
1114::4           dd:dd:dd:dd:dd:dd  static    ve 100            ethernet 0/1:4     51 (s)
device#

```

The following example displays NDP entries for a VRF instance named `vrf1` on the device. In this example, the entries are filtered by the IPv6 address of a neighboring device:

```

device# show ipv6 neighbor vrf vrf1 1111::2

IP Address : 1111::2
MAC Address : aa:aa:aa:aa:aa:ac

Source : Local
=====
Seq No.          : 10                               Age           : 21 (s)
Type             : Dynamic                          VRF instance   : default-vrf
Interface        : ethernet 0/1:1                    Index          : 1048580
L2 Interface     : ethernet 0/1:1

Source : BGP
=====
Seq No.          : 101                               Age           : 21 (s)
Type             : Dynamic                          VRF instance   : default-vrf
Interface        : Ve 100                             Index          : 1048581
L2 Interface     : ethernet 0/1:4
device#

```


show ipv4 route vrf (IP routing)

Use the **show ipv4 route vrf** command to display entries in the IPv4 routing table for a Virtual Routing and Forwarding (VRF) instance or all VRF instances.

Syntax

```
show ipv4 route vrf vrf-name [ ipv4-address / prefix-length] [ connected  
| static | arp | bgp | brief ]  
  
show ipv4 route vrf all [ brief ]
```

Parameters

vrf-name

Name of the VRF with the entries to be displayed.

ipv4-address / prefix-length

(Optional) Range of IPv4 addresses. Use the *a.b.c.d* format for *ipv4-address*, and use an integer from 0 to 32 for the prefix. For example, 10.1.1.0/24. You must include the slash (/).

connected

(Optional) Displays entries for routes that are destined for all hosts in subnets directly connected to the device's active interfaces.

static

(Optional) Displays static entries.

arp

(Optional) Displays address resolution protocol (ARP) entries.

bgp

(Optional) Displays BGP entries.

brief

(Optional) Displays abbreviated information for all entries for the VRF instance.

all [**brief**]

- all displays all entries for the VRF instance.
- (Optional) brief displays abbreviated information in a table for all entries for the VRF instance.

Modes

EXEC (#)

Usage Guidelines

To clear the IPv4 routing table details, see [clear ipv4 route vrf \(IP routing\)](#) on page 535.

Output

The **show ipv4 route vrf** command displays the following information:

Output field	Description
VRF	Name of the VRF instance that is configured on the device.
Total number of IPv4 routes	Number of IPv4 entries in the routing table for the VRF.
Max Routes	Maximum number of IPv4 routes that the VRF can learn.
12.1.1.10/32, ARP-local, [3/0], tag 0, 2h54m9s	Details of an IPv4 route: 12.1.1.10/32—Network identifier of the final destination. The network identifier consists of the IPv4 address of the final destination and the subnet mask (a 32-bit network address that identifies whether a host belongs to the local or remote network). - ARP-local—Routing protocol code (ARP, BGP, connected, or static), which indicates where the routes came from before they were loaded into the routing table. [3/0]—Slash-separated routing preference and routing metric (respectively) of the path through which the packet will be sent. These values are assigned to each available route to a specific network. The routing preference is the order in which routes are tried, with the best path chosen based on the highest preference value. The routing metric is the distance or cost associated with each route, which is used to determine the best path based on the lowest metric value. If multiple routes exist to the same destination network, the route proceeds toward the gateway with the lowest metric (the best path). - tag 0—Integer that is used to tag the route. This is a unique identifier assigned to each route in the table. It is used to differentiate between multiple routes with the same destination network and to track the specific route that is used to reach that network. The route tag is assigned by the router or network administrator. 2h54m9s—Last time that the route was updated (in hours, minutes, and seconds)
via ethernet 0/13, [00:10:94:00:02:61, ethernet 0/13.0]	MAC address and name of the locally available interface that is used to reach the next hop (the gateway).

Examples

The following example displays routing table entries for a VRF instance named default-vrf on the device:

```
device# show ipv4 route vrf default-vrf

Total number of IPv4 routes: 5, Max Routes: Not Set
'[x/y]' denotes [preference/metric]
1.1.1.1/32, attached, [0/0], tag 0, 2h52m21s
  via direct, loopback 1
12.1.1.1/32, local, [0/0], tag 0, 2h52m21s
  via direct, ethernet 0/13
12.1.1.10/32, ARP-local, [3/0], tag 0, 2h52m20s
  via , ethernet 0/13, [00:10:94:00:02:61, ethernet 0/13.0]
12.1.1.0/24, attached, [0/0], tag 0, 2h52m21s
  via direct, ethernet 0/13
24.1.1.0/24, static, [1/1], tag 0, 2h52m21s
  via 12.1.1.10, ethernet 0/13, [00:10:94:00:02:61, ethernet 0/13]
device#
```

The following example displays routing table entries for ARP routes for a VRF instance named default-vrf on the device:

```
device# show ipv4 route vrf default-vrf arp

Total number of IPv4 routes: 5, Max Routes: Not Set
'[x/y]' denotes [preference/metric]
12.1.1.10/32, ARP-local, [3/0], tag 0, 2h52m35s
  via , ethernet 0/13, [00:10:94:00:02:61, ethernet 0/13.0]
device#
```

The following example displays routing table entries for static routes for a VRF instance named default-vrf on the device:

```
device# show ipv4 route vrf default-vrf static

Total number of IPv4 routes: 5, Max Routes: Not Set
'[x/y]' denotes [preference/metric]
24.1.1.0/24, static, [1/1], tag 0, 2h52m48s
  via 12.1.1.10, ethernet 0/13, [00:10:94:00:02:61, ethernet 0/13]
# device
```

The following example displays routing table entries for a VRF instance named default-vrf on the device. In this example, the entries are filtered by IPv4 prefix 24.1.1.0/24:

```
device# show ipv4 route vrf default-vrf 24.1.1.0/24

'*' denotes best route-source
'[x/y]' denotes [preference/metric]
*24.1.1.0/24, static, [1/1], tag 0, 2h52m58s, nhGrpId 4
  Protocol-Nexthops:
    via 12.1.1.10, ethernet 0/13, [00:10:94:00:02:61, ethernet 0/13]
  Resolved-Nexthops (Aggregate):
    via 12.1.1.10, ethernet 0/13, [00:10:94:00:02:61, ethernet 0/13]
24.1.1.0/24, ibgp, [200/0], tag 0, 2h52m57s, nhGrpId 4
  Protocol-Nexthops:
    via 12.1.1.10, ethernet 0/13, [00:10:94:00:02:61, ethernet 0/13]
  Resolved-Nexthops (Aggregate):
    via 12.1.1.10, ethernet 0/13, [00:10:94:00:02:61, ethernet 0/13]
device#
```

The following example displays abbreviated information in a table for all entries for a VRF instance named default-vrf on the device:

```
device# show ipv4 route vrf default-vrf brief

Total number of IPv4 routes: 5, Max Routes: Not Set
Type Codes - B:BGP L:Local O:OSPF D:Direct/Connected S:Static A: Arp
BGP Codes - i:iBGP e:eBGP E:evpn
OSPF Codes - i:Inter Area 1:External Type 1 2:External Type 2
Hardware Status Codes - #:Failed
IP Prefix      Next Hop      Interface      Pref/Metric    Type
-----
1.1.1.1/32     DIRECT      loopback 1     0/0            D
12.1.1.1/32    DIRECT      ethernet 0/13  0/0            L
12.1.1.10/32   00:10:94:00:02:61  ethernet 0/13  3/0            A
12.1.1.0/24    DIRECT      ethernet 0/13  0/0            D
24.1.1.0/24    12.1.1.10  ethernet 0/13  1/1            S
device#
```

The following example displays routing table entries for connected routes for a VRF instance named default-vrf on the device:

```
device# show ipv4 route vrf default-vrf connected

Total number of IPv4 routes: 5, Max Routes: Not Set
'[x/y]' denotes [preference/metric]
1.1.1.1/32, attached, [0/0], tag 0, 2h53m47s
  via direct, loopback 1
12.1.1.0/24, attached, [0/0], tag 0, 2h53m47s
  via direct, ethernet 0/13
device#
```

The following example displays routing table entries for BGP routes for a VRF instance named default-vrf on the device:

```
device# show ipv4 route vrf default-vrf bgp

Total number of IPv4 routes: 5, Max Routes: Not Set
'[x/y]' denotes [preference/metric]
24.1.1.0/24, ibgp, [200/0], tag 0, 3h13m30s
  via 12.1.1.10, ethernet 0/13, [00:10:94:00:02:61, ethernet 0/13]
device#
```

The following example displays routing table entries for all VRF instances on the device:

```
device# show ipv4 route vrf all

VRF:mgmt-vrf
-----
Total number of IPv4 routes: 2, Max Routes: Not Set
'[x/y]' denotes [preference/metric]
192.168.122.70/32, local, [0/0], tag 0, 2h54m11s
  via direct, management 0
192.168.122.0/24, attached, [0/0], tag 0, 2h54m11s
  via direct, management 0

VRF:default-vrf
-----
Total number of IPv4 routes: 5, Max Routes: Not Set
'[x/y]' denotes [preference/metric]
1.1.1.1/32, attached, [0/0], tag 0, 2h54m11s
  via direct, loopback 1
12.1.1.1/32, local, [0/0], tag 0, 2h54m11s
  via direct, ethernet 0/13
```

```
12.1.1.10/32, ARP-local, [3/0], tag 0, 2h54m9s
  via , ethernet 0/13, [00:10:94:00:02:61, ethernet 0/13.0]
12.1.1.0/24, attached, [0/0], tag 0, 2h54m11s
  via direct, ethernet 0/13
24.1.1.0/24, static, [1/1], tag 0, 2h54m11s
  via 12.1.1.10, ethernet 0/13, [00:10:94:00:02:61, ethernet 0/13]
device#
```

show ipv6 route vrf (IP routing)

Use the **show ipv6 route vrf** command to display entries in the IPv6 routing table for a Virtual Routing and Forwarding (VRF) instance or all VRF instances.

Syntax

```
show ipv6 route vrf vrf-name [ ipv6-address / prefix-length] [ connected  
| static | neighbor | bgp | brief ]  
  
show ipv6 route vrf all [ brief ]
```

Parameters

vrf-name

Name of the VRF with the entries to be displayed.

ipv6-address / prefix-length

(Optional) Range of IPv6 addresses. Use the *x:x::x:x* format for *ipv6-address*, and use an integer from 0 to 128 for the prefix. For example, 2001::1/64. You must include the slash (/).

connected

(Optional) Displays entries for routes that are destined for all hosts in subnets directly connected to the device's active interfaces.

static

(Optional) Displays static entries.

neighbor

(Optional) Displays address resolution protocol (ARP) entries.

bgp

(Optional) Displays BGP entries.

brief

(Optional) Displays abbreviated information for all entries for the VRF instance.

all [**brief**]

- all displays all entries for the VRF instance.
- (Optional) brief displays abbreviated information in a table for all entries for the VRF instance.

Modes

EXEC (#)

Usage Guidelines

To clear the IPv6 routing table details, see [clear ipv6 route vrf \(IP routing\)](#) on page 536.

Output

The **show ipv6 route vrf** command displays the following information:

Output field	Description
VRF	Name of the VRF instance that is configured on the device.
Total number of IPv6 routes	Number of IPv6 entries in the routing table for the VRF.
Max Routes	Maximum number of IPv6 routes that the VRF can learn.
200:25:2f:1::/64, static, [1/1], tag 0, 12m34s	Details of an IPv6 route: 200:25:2f:1::/64—Network identifier of the final destination. The network identifier consists of the IPv6 address of the final destination and the subnet mask (a 32-bit network address that identifies whether a host belongs to the local or remote network). - static—Routing protocol code, which indicates where the routes came from before they were loaded into the routing table. [1/1]—Slash-separated routing preference and routing metric (respectively) of the path through which the packet will be sent. These values are assigned to each available route to a specific network. The routing preference is the order in which routes are tried, with the best path chosen based on the highest preference value. The routing metric is the distance or cost associated with each route, which is used to determine the best path based on the lowest metric value. If multiple routes exist to the same destination network, the route proceeds toward the gateway with the lowest metric (the best path). - tag 0—Integer that is used to tag the route. This is a unique identifier assigned to each route in the table. It is used to differentiate between multiple routes with the same destination network and to track the specific route that is used to reach that network. The route tag is assigned by the router or network administrator. 12m34s—Last time that the route was updated (in hours, minutes, and seconds)
via 200:21:1:2f::2, ve 1047, [e4:44:e5:08:04:0b, port-channel 51.1047]	MAC address and name of the locally available interface that is used to reach the next hop (the gateway).

Examples

The following example displays routing table entries for a VRF instance named VR47 with prefix length 200:25:2f:1::/64 on the device:

```
device# show ipv6 route vrf VR47 200:25:2f:1::/64
```

```
'*' denotes best route-source
'[x/y]' denotes [preference/metric]
*200:25:2f:1::/64, static, [1/1], tag 0, 12m34s, nhGrpId 8409
Protocol-Nexthops (Active):
  via 200:21:1:2f::2, ve 1047, [e4:44:e5:08:04:0b, port-channel 51.1047]
  via 200:21:2:2f::2, ve 1347, [e4:44:e5:08:06:0b, port-channel 52.1347]
Resolved-Nexthops (Aggregate):
  via 200:21:1:2f::2, ve 1047, [e4:44:e5:08:04:0b, port-channel 51.1047]
  via 200:21:2:2f::2, ve 1347, [e4:44:e5:08:06:0b, port-channel 52.1347]
device#
```

The following example displays routing table BGP entries for ARP routes for a VRF instance named default-vrf on the device:

```
device# show ipv6 route vrf default-vrf bgp

Resilient Hash: Enabled
Ecmp Max Path: 128
Total number of IPv6 routes: 117, Max Routes: Not Set
'[x/y]' denotes [preference/metric]
1001::1/128, ebgp, [20/0], tag 0, 5m32s
  via 20:20:20::2, ethernet 0/23:2, [e4:db:ae:c3:84:1a, ethernet 0/23:2]
  via 30:30:30::2, ethernet 0/23:3, [e4:db:ae:c3:84:1b, ethernet 0/23:3]
1001::2/128, ebgp, [20/0], tag 0, 5m32s
  via 20:20:20::2, ethernet 0/23:2, [e4:db:ae:c3:84:1a, ethernet 0/23:2]
  via 30:30:30::2, ethernet 0/23:3, [e4:db:ae:c3:84:1b, ethernet 0/23:3]
1001::3/128, ebgp, [20/0], tag 0, 5m32s
  via 20:20:20::2, ethernet 0/23:2, [e4:db:ae:c3:84:1a, ethernet 0/23:2]
  via 30:30:30::2, ethernet 0/23:3, [e4:db:ae:c3:84:1b, ethernet 0/23:3]
1001::4/128, ebgp, [20/0], tag 0, 5m32s
  via 20:20:20::2, ethernet 0/23:2, [e4:db:ae:c3:84:1a, ethernet 0/23:2]
  via 30:30:30::2, ethernet 0/23:3, [e4:db:ae:c3:84:1b, ethernet 0/23:3]
1001::5/128, ebgp, [20/0], tag 0, 5m32s
  via 20:20:20::2, ethernet 0/23:2, [e4:db:ae:c3:84:1a, ethernet 0/23:2]
  via 30:30:30::2, ethernet 0/23:3, [e4:db:ae:c3:84:1b, ethernet 0/23:3]
1001::6/128, ebgp, [20/0], tag 0, 5m32s
  via 20:20:20::2, ethernet 0/23:2, [e4:db:ae:c3:84:1a, ethernet 0/23:2]
  via 30:30:30::2, ethernet 0/23:3, [e4:db:ae:c3:84:1b, ethernet 0/23:3]
1001::7/128, ebgp, [20/0], tag 0, 5m32s
  via 20:20:20::2, ethernet 0/23:2, [e4:db:ae:c3:84:1a, ethernet 0/23:2]
...
device#
```


show lacp interface

Use the **show lacp interface** command to display statistics for specific member ports of dynamic link aggregation groups (LAGs).

Syntax

```
show lacp interface ethernet interface-name
```

Parameters

ethernet *interface-name*

Filter interfaces by Ethernet interface name. Use the *slot/port* format or the *slot/port:breakout* format to specify an interface or one or more comma-separated ranges of interfaces respectively. For example, 0/1 or 1/1-3,5,2/7-9,10:1-4 respectively.

Modes

EXEC (#)

Usage Guidelines

You can use the **show running-config lacp** command to display the LAG configuration that is running currently on a device.

Output

The **show lacp interface** command displays the following information:

Output field	Description
interface Eth 0/21 is up	Name and operational state of the interface that is a member of the LAG.
Channel group is 10 port channel is Po10	Port channel number of the LAG to which the interface is assigned.
PDUs sent	Number of LACP protocol data units (PDUs) sent over the LACP link. LACP continuously exchanges PDUs to monitor the health of each member link. The PDUs contain information about the links in the network, including their status and the amount of traffic that they carry.
PDUs rcvd	Number of PDUs received over the LACP link. The PDUs contain information about the links in the network, including their status and the amount of traffic that they carry.
Local Port: Eth 0/21	Name of the port on the local device that is participating in the LACP link aggregation.

Output field	Description
MAC Address	MAC address of the port on the local device that is participating in the LACP link aggregation.
System Identifier	Unique identifier assigned to the LACP port, which determines the LACP port's role in a LAG. The system identifier is the combination of the LACP system priority value and the MAC address of the device.
Port Identifier	Port priority (expressed in hexadecimal) and port number (expressed in hexadecimal). For example, a port identifier value of 0x8000, 0x15 is equivalent to a port priority of 32768 (the default value) and port number 21.
Operational key	Administrative key, which defines the ability of a port to aggregate with other ports. LACP automatically configures an administrative key value equal to the channel group identification number on each port configured to use LACP.
LACP_Activity	Locally configured LACP mode (active or passive).
LACP_Timeout	Duration of the LACP timeout. This is either Long Timeout (30s) or Short Timeout (1s).
Synchronization	Indicates whether this member port is in sync with the peer device. This is either IN_SYNC or OUT_OF_SYNC. IN_SYNC is set only when a PDU is received from the peer.
Collecting	Indicates whether this member port is ready to receive the data traffic from the peer.
Distributing	Indicates whether this member port is ready to send the data traffic to the peer.
Partner information	Information about the other interface in the LAG.
Partner-id	ID of the other interface in the LAG.
Partner-key	Key that identifies the partner link in the LAG. This key is typically set to a specific value by the administrator. Partner keys enables more efficient management of LAGs, because they eliminate the need for manual configuration of each partner link. Devices can configure an aggregation link only when the partner key of the port on the device matches the actor key of the port on the other side of the link.

Examples

The following example displays statistics for member ports of the dynamic link aggregation groups that are configured on the device:

```
device# show lacp interface ethernet 0/21

interface Eth 0/21 is up
  Channel group is 10 port channel is Po10
  PDUs sent: 51
  PDUs rcvd: 56
  LACP Rx errors: 0
  LACP Tx errors: 0
```

```
    LACP unknown errors: 0
    LACP errors: 0
Local Port: Eth 0/21    MAC Address = 88:7e:25:d4:09:00
System Identifier = 80:00:88:7e:25:d4:09:00
Port Identifier = 0x8000, 0x15
Operational key = 10
LACP_Activity = active
LACP_Timeout = Long Timeout (30s)
Synchronization = IN_SYNC
Collecting = true
Distributing = true

Partner information
    Partner-id = 80:00:88:7e:25:d4:07:00
    Partner-key = {10, 25}
device#
```

show lacp system-identifier

Use the **show lacp system-identifier** command to display the unique identifier assigned to a link aggregation group (LAG).

Syntax

```
show lacp system-identifier
```

Parameters

This command has no keywords or variables.

Modes

EXEC (#)

Usage Guidelines

The system identifier is used to identify the LACP group at the aggregated Ethernet interface level and determine the state of the links in the group. It is used with other LACP parameters to manage the links in a network. The system identifier is the combination of the LACP system priority value and the MAC address of the device.

You can use the **show running-config lacp** command to display the LAG configuration that is running currently on a device.

Output

The **show lacp system-identifier** command displays the following information:

Output field	Description
0x8000	System priority (expressed in hexadecimal). For example, 0x8000 is equivalent to a system priority of 32768 (the default value)
88:7e:25:d4:09:00	System MAC address.

Examples

The following example displays the unique system identifier for the LAG that is configured on the device:

```
device# show lacp system-identifier

      System ID: 0x8000, 88:7e:25:d4:09:00
device#
```

show link-fault-signaling

Use the **show link-fault-signaling** command to display the link fault signaling (LFS) configuration for each front port.

Syntax

```
show link-fault-signaling
```

Parameters

This command has no keywords or variables.

Modes

EXEC (#)

Usage Guidelines

LFS is a physical layer protocol that enables communication on a link between two Ethernet devices. It operates between the remote Ethernet device Reconciliation Sublayer (RS) and the local Ethernet device RS. When configured on an Ethernet port, the port can detect and report fault conditions on transmit and receive ports.

To see whether LFS is enabled on a specific port as well as the general link fault status on that port, see [show interface ethernet link-error-disable](#) on page 1012. To see statistics for LFS detection, see [show counters link-fault-signaling](#) on page 989.

You must clear the counters if you want to start at zero for more recent statistics. For details, see xxxxxx.

Examples

The following example displays the LFS configuration for each front port on the device:

```
device# show link-fault-signaling

Rx and Tx link-fault-signaling status

Port          Link-Faults
=====
Eth 0/1       ON
Eth 0/2       ON
Eth 0/3       ON
Eth 0/4       ON
Eth 0/5       ON
Eth 0/6       ON
Eth 0/7       ON
Eth 0/8       OFF
device#
```

show lldp neighbors interface

Use the **show lldp neighbors interface** command to display detailed information about link layer discovery protocol (LLDP) neighbor interfaces.

Syntax

```
show lldp neighbors interface ethernet { interface-name | all }
```

Parameters

ethernet *interface-name*

Filter interfaces by Ethernet interface name. Use the *slot/port* format or the *slot/port:breakout* format to specify an interface or one or more comma-separated ranges of interfaces respectively. For example, 0/1 or 1/1-3,5,2/7-9,10:1-4 respectively.

all

Displays information for all neighbor interfaces.

Modes

EXEC (#)

Usage Guidelines

This command displays detailed information about interfaces on neighboring devices discovered using LLDP.

For more general information about neighboring devices discovered using LLDP, see [show lldp neighbors](#) on page 1049.

Output

The **show lldp neighbors interface** command displays the following information:

Output field	Description
LLDP Interface	Name of the neighbor interface for which detailed information is displayed.
Chassis id	Identifier of the 802 LAN neighboring device's chassis.
Port id	Interface and port number of the neighboring device.
Local Port id	Number of the local port to which the neighbor is connected.
Port Description	Configured port description, the port name, or the SNMP ifIndex. The SNMP ifIndex is a unique value to identify a number associated with a logical or physical interface.

Output field	Description
System Name	Name of the system that is gathered from the neighbor.
System Description	Description supplied by the system on the interface.
Time remaining	Number of seconds for which this information is valid.
System Capabilities	Capabilities (such as bridge, bridge router, and bridge telephone) that are supported by the system on the interface.
Enabled Capabilities	Capabilities (such as bridge, bridge router, and bridge telephone) that are enabled on the interface.
Management Address	Global LLDP IPv4 address that is transmitted as the type-length-value (TLV) management address in LLDP frames. This allows easier management and troubleshooting of LLDP communications. This IP address must be valid and reachable for successful LLDP communication. If this address is configured globally, all LLDP enabled interfaces inherit this address; otherwise only configured interfaces use it as their management address. If both global and per interface management addresses are configured, the interface configuration takes precedence.
Management Address IPV6	Global LLDP IPv6 address if a global LLDP IPv4 address is not configured.

Examples

The following example displays detailed information about Ethernet interface 0/1-2 on neighbor devices discovered using LLDP:

```
device# show lldp neighbors interface ethernet 0/1-2

Capability codes:
  (R) Router, (B) Bridge, (T) Telephone, (C) DOCSIS Cable Device
  (W) WLAN Access Point, (P) Repeater, (S) Station, (O) Other

LLDP Interface: ethernet 0/1
Chassis id: 88f0.31f9.a340
Port id: ethernet 1/10
Local Port id: ethernet 0/1
Port Description: ethernet 1/10
System Name: Lab switch1
System Description: "ExtremEXOS (Summit) version
                    31.1.1.4 v311b4 by release-manager on Fri Feb 13 16:30:06 EST 2021"
Time remaining: 91 seconds
System Capabilities: B, R
Enabled Capabilities: B, R
Management Address: 10.77.143.43
Management Address IPV6: not advertised

LLDP Interface: ethernet 0/2
Chassis id: 88f0.31f9.a341
Port id: ethernet 1/11
Local Port id: ethernet 0/2
Port Description: ethernet 1/11
System Name: Lab switch 2
System Description: Cisco Nexus Operating System (NX-OS) Software 9.2(1) TAC support:
http://www.cisco.com/tac Copyright (c) 2002-2018, Cisco Systems, Inc. All rights reserved.
```

```
Time remaining: 85 seconds
System Capabilities: B, R
Enabled Capabilities: B, R
Management Address: 10.77.143.43
Management Address IPV6: not advertised
device#
```


show lldp neighbors

Use the **show lldp neighbors** command to display information about link layer discovery protocol (LLDP) neighbors.

Syntax

```
show lldp neighbors
```

Parameters

This command has no keywords or variables.

Modes

EXEC (#)

Usage Guidelines

This command displays information about neighboring devices discovered using LLDP. To display detailed LLDP neighbor information for a specific interface, see [show lldp neighbors interface](#) on page 1046.

Output

The **show lldp neighbors** command displays the following information:

Output field	Description
Chassis ID	Identifier of the 802 LAN neighboring device's chassis.
Local-Port	Number of the local port to which the neighbor is connected.
Dead Intvl	Time period during which the device does not send LLDP messages after receiving an LLDP message from a neighbor. This interval is used to prevent LLDP messages from being sent excessively and to help maintain network stability.
Rem Life	Number of seconds for which neighbor information is valid for this interface.
Remote Port-ID	Interface and port number of the neighboring device.
Remote Port Descr	Remote port description of the connected LLDP device.
System Name	Name of the system that is gathered from the neighbor.
Total entries displayed	Number of neighbor devices for which information is displayed.

Examples

The following example displays global LLDP configuration information for the device:

```
device# show lldp neighbors
```

```
show LLDP Neighbors Information
```

Chassis ID	Local-Port	Dead Intvl	Rem Life	Remote Port-ID	Remote Port Descr	System Name
Switch 2426	1/18	180	100	Ethernet0/26	LLDP-1	DUT-1
Switch 2426	1/22	120	50	Ethernet0/27	LLDP-2	DUT-2

```
Total entries displayed: 2
```

```
device#
```

show lldp

Use the **show lldp** command to display global link layer discovery protocol (LLDP) configuration information for a device.

Syntax

```
show lldp
```

Parameters

This command has no keywords or variables.

Modes

EXEC (#)

Usage Guidelines

This command displays LLDP configuration information that applies to all LLDP enabled interfaces and to specific LLDP enabled interfaces.

Output

The **show lldp** command displays the following information:

Output field	Description
Global LLDP Information	LLDP information that applies to all interfaces that are configured for LLDP.
Status	Indicates whether LLDP is active on the device.
LLDP advertisements	Frequency (in seconds) with which LLDP advertisements are sent.
LLDP hold time	Number of seconds that a receiving device should hold LLDP information sent by this device before discarding it.
LLDP transmit	Global LLDP packet transmit mode (Off or On). The default is On.
LLDP receive	Global LLDP packet receive mode (Off or On). The default is On.

Output field	Description
LLDP Static Mgmtip	Global LLDP IPv4 or IPv6 address that is transmitted as the type-length-value (TLV) management address in LLDP frames. This allows easier management and troubleshooting of LLDP communications. This IP address must be valid and reachable for successful LLDP communication. If this address is configured globally, all LLDP enabled interfaces inherit this address; otherwise only configured interfaces use it as their management address. If both global and per interface management addresses are configured, the interface configuration takes precedence.
LLDP Suppress Tlvs	Indicates which optional TLV advertisements are configured to be suppressed globally. These optional TLV advertisements are for the management address, port description, system description, and system name. If this is enabled globally, all LLDP enabled interfaces stop advertising the configured optional TLVs from their LLDP frames. Per interface configuration of this functionality allows only configured interfaces to stop advertising TLVs from their frames. If both global and per interface functionality are configured for TLV advertisement suppression, the interface configuration takes precedence.
Interface LLDP Information	LLDP information that applies to each interface that is configured for LLDP.
Intf	Name of the interface for which LLDP configuration information is reported.
Rx	Ability of the interface to receive advertisements.
Tx	Ability of the interface to transmit advertisements.

Examples

The following example displays global LLDP configuration information for the device:

```
device# show lldp

LLDP Config Info

Global LLDP Information:
Status: INACTIVE
LLDP advertisements are sent every 200 seconds
LLDP hold time is 120 seconds
LLDP transmit: Off
LLDP receive: On
LLDP Static Mgmtip: 1.1.1.1
LLDP Suppress Tlvs:

Interface LLDP Information:
Intf              State      Rx      Tx
-----
management 0     Enabled   On      On
ethernet 0/1     Disabled  On      On
ethernet 0/2     Disabled  On      On
```

```
ethernet 0/3      Disabled  On    On
ethernet 0/4      Disabled  On    On
ethernet 0/5      Disabled  On    On
ethernet 0/6      Disabled  On    On
ethernet 0/7      Disabled  On    On
ethernet 0/8      Disabled  On    On
ethernet 0/9      Disabled  On    On
ethernet 0/10     Disabled  On    On
ethernet 0/11     Disabled  On    On
ethernet 0/12     Disabled  On    On
ethernet 0/13     Enabled   On    On
device#
```

show l2 mac-move-detection

Use the **show l2 mac-move-detection** command to display the mac-move-detection configuration state for a device.

Syntax

```
show l2 mac-move-detection
```

Parameters

This command has no keywords or variables.

Modes

EXEC (#)

Usage Guidelines

This command displays the mac-move-detection tracking interval, the threshold for triggering an action (in number of moves), type of action (such as a shutdown) taken when a threshold is reached, and auto-recovery completion time (in seconds).

Examples

The following example displays the mac-move-detection configuration state for the device:

```
device# show l2 mac-move-detection

MAC-move detection is enabled and configured:
  Tracking interval: 10 seconds
  Threshold for triggering action: 10 moves
  Action taken when threshold is reached: shutdown
  Auto-recovery is done after 80 seconds.
device#
```

show l2 mac-move-detection shut-list

Use the **show l2 mac-move-detection shut-list** command to display the mac-move-detection shut-port list for a device.

Syntax

```
show l2 mac-move-detection shut-list
```

Parameters

This command has no keywords or variables.

Modes

EXEC (#)

Usage Guidelines

This command displays the interfaces that are shut down because of mac-move-detection on the device.

To remove a port from the shut list, see [clear l2 mac-move-detection shut-list](#) on page 533.

Examples

The following example displays the list of interfaces in the mac-move-detection shut-list for the device:

```
device# show l2 mac-move-detection shut-list

MAC-move detection shut port list

Interface                BD ID      Time remaining (s)
-----
ethernet 0/1:3.100       500        2
ethernet 0/1:2.100       500       34
device#
```

show mac-address-table bridge-domain

Use the **show mac-address-table bridge-domain** command to display the MAC address table for a specific bridge domain or all bridge domains on a device.

Syntax

```
show mac-address-table bridge-domain { ID { static | dynamic | mlag |  
      evpn | mac-address } | all }
```

Parameters

ID

ID number of the bridge domain to display. Use an integer from 1 to 8192.

static

MAC addresses learned from the configuration.

dynamic

MAC addresses learned dynamically.

mlag

MAC addresses learned via multichassis link aggregation (MLAG).

evpn

MAC addresses learned via BGP Ethernet VPN (EVPN).

mac-address

Specific MAC address. Use six groups of two hex digits separated by colons. For example, 01:23:45:67:89:AB.

all

MAC address tables for all bridge domains.

Modes

EXEC (#)

Usage Guidelines

You can view the table for a specific MAC address or the details for all MAC addresses of a specific type: static, dynamic, multichassis link aggregation (MLAG), or Ethernet VPN (EVPN).

To clear the contents of the MAC address table for a specific bridge domain, use the [clear mac-address-table bridge-domain](#) on page 537 command.

Examples

The following example displays the MAC address table for bridge domain 100 on the device:

```
device# show mac-address-table bridge-domain 100

Total number of Mac Entries: 3
Hardware Status Codes - #:Failed
Mac-Address          Type          Interface
-----
00:10:94:00:00:02    Dynamic       ethernet 0/13.100
00:10:94:00:00:10    Static        ethernet 0/13.100
00:16:3e:1e:b6:03    Local         Local
device#
```

The following example displays dynamic addresses in the MAC address table for bridge domain 100 on the device:

```
device# show mac-address-table bridge-domain 100 dynamic

Total number of Mac Entries: 3
Hardware Status Codes - #:Failed
Mac-Address          Type          Interface
-----
00:10:94:00:00:02    Dynamic       ethernet 0/13.100
device#
```

The following example displays static addresses in the MAC address table for bridge domain 100 on the device:

```
device# show mac-address-table bridge-domain 100 static

Total number of Mac Entries: 3
Hardware Status Codes - #:Failed
Mac-Address          Type          Interface
-----
00:10:94:00:00:10    Static        ethernet 0/13.100
device#
```

The following example displays MLAG addresses in the MAC address table for bridge domain 100 on the device:

```
device# show mac-address-table bridge-domain 100 mlag

Total number of Mac Entries: 2
Hardware Status Codes - #:Failed
Mac-Address          Type          Interface
-----
00:10:94:10:01:01    MLAG          ethernet 0/1:1.10
device#
```

The following example displays EVPN addresses in the MAC address table for bridge domain 100 on the device:

```
device# show mac-address-table bridge-domain 8192 evpn

Total number of Mac Entries: 5
Hardware Status Codes - #:Failed
Mac-Address          Type          Interface
-----
00:00:00:00:00:71    EVPN          tunnel EVPN_default-vrf-Source-71.72.7.8
```

```
00:04:96:b7:ad:f9      EVPN-Static      tunnel EVPN_default-vrf-Source-71.72.7.8
device#
```

The following example displays details about MAC address 00:10:94:00:00:02 in the MAC address table for bridge domain 100 on the device:

```
device# show mac-address-table bridge-domain 100 00:10:94:00:00:02
```

```
'*' denotes best route-source
Mac-Address      Type      Interface      Last
Change          Seq No   Hardware Status
-----
*00:10:94:00:00:02    Dynamic    ethernet 0/13.100
1m44s           0
device#
```

The following example displays the MAC address tables for all bridge domains on the device:

```
device# show mac-address-table bridge-domain all
```

```
Bridge-Domain:100
-----
Total number of Mac Entries: 3
Hardware Status Codes - #:Failed
Mac-Address      Type      Interface
-----
00:10:94:00:00:02    Dynamic    ethernet 0/13.100
00:10:94:00:00:10    Static     ethernet 0/13.100
00:16:3e:1e:b6:03    Local      Local
Bridge-Domain:101
-----
Total number of Mac Entries: 2
Hardware Status Codes - #:Failed
Mac-Address      Type      Interface
-----
00:10:94:00:00:02    Dynamic    ethernet 0/13.101
00:16:3e:1e:b6:03    Local      Local
device#
```

show mirror

Use the **show mirror** command to display details about a specific port mirror or all port mirrors on a device.

Syntax

```
show mirror { mirror-name | all }
```

Parameters

mirror-name

Name of a specific port mirror.

all

Displays details for all port mirrors.

Modes

EXEC (#)

Usage Guidelines

This command displays details about a specific port mirror or all port mirrors that are configured on a device. These details include the name, description, source interface, and destination interface (where the mirror is attached).

Examples

The following example displays details for all port mirrors on the device:

```
device# show mirror all

Number of mirrors: 4
      Name : test_mirror
      Description : test mirror
      Destination Interface : ethernet 0/10

      Name : test_mirror_po
      Description : -
      Destination Interface : none

      Name : test_mirror_phy
      Description : -
      Source Interface : ethernet 0/3
      Direction : MIRROR_INGRESS
      Source Interface : ethernet 0/3
      Direction : MIRROR_EGRESS
      Source Interface : port-channel 20
      Direction : MIRROR_INGRESS
      Destination Interface : ethernet 0/2

      Name : mirror_test_internal
      Description : -
```

```
Destination Interface : internal 0  
device#
```

show mlag

Use the **show mlag** command to display information about the multichassis link aggregation (MLAG) configuration.

Syntax

```
show mlag
```

Parameters

This command has no keywords or variables.

Modes

EXEC (#)

Output

The **show mlag** command displays the following information:

Output field	Description
Role	Configured device role. The possible values are PRIMARY, BACKUP (also called secondary), or AUTO. The MLAG role is selected as primary or backup only when the MLAG session comes up. With the AUTO role type, the role is selected based on the primary keepalive source IP address. Otherwise, the node with the highest primary keepalive source IP address is selected as primary, and the lowest will be the backup node.
MAC	MLAG MAC address.
Bring up delay	Interval during which MLAG interfaces are disabled after an MLAG peer reboots. The default value is 90 seconds.
Keepalive Delay	Initial delay before the MLAG peers begin exchanging keepalive messages with each other. This lets them synchronize their state and establish a shared understanding of the network topology before the network is brought into a stable state. The default value is 1000 milliseconds.
TX Interval	Interval between two keepalive messages. The default value is 300 milliseconds.
Multiplier	Value used to multiply the interval (as in milliseconds x multiple) to determine the total time taken to bring down a keepalive session. This is an integer from 1 to 50. The default value is 3.
No. of Peers	Number of MLAG peers configured on the device.
No. of Interfaces	Number of MLAG interfaces configured on the device.

Output field	Description
Uplink-track Interfaces	List of uplink track ports that are configured between MLAG peers and external router/spine ports that can be blocked during reload and “split brain” scenarios to avoid traffic duplication.
Peer Info	Details about the peers (the remote devices) that are configured under the MLAG: - Peer—Name of the peer. - State—Operational state of the peer.
Interfaces Info	Number of MLAG interfaces configured on device as well as the local and remote MLAG interface states: - Id—Unique ID of the interface. - Interface—Name of the interface. - Peer—Name of the MLAG peer containing the interface. - Local/Remote State—Operational status of the MLAG member ports (the local port and the port on the peer respectively). This is either UP, DOWN, OR UNKNOWN. - Exceptions—Errors on the interface. (In this release, the only possible exception is Interface Down.)

Examples

The following example displays information about the MLAG configuration on the device:

```
device# show mlag

Role           : AUTO
MAC            : A0:01:A0:02:A0:03
Bring up delay : 90 seconds
Keepalive Delay : 1000 ms
Tx Interval    : 300 ms
Multiplier     : 3
No. of Peers   : 1
No. of Interfaces : 1
Uplink-track Interfaces : ethernet 1/1-3,5,2/7-9,3/1:1-4,  port-channel 1-3,5,7-9

Peer Info:
=====
Peer                               State
-----
mlag_peer                           UP

Interfaces Info:
=====
Id      Interface      Peer      Local/Remote State  Exceptions
-----
230     ethernet 0/9    mlag_peer DOWN / UNKNOWN      Interface Down
device#
```

show mlag interfaces

Use the **show mlag interfaces** command to display details about the multichassis link aggregation (MLAG) interfaces.

Syntax

```
show mlag interfaces [ ethernet interface-name | port-channel port-channel-number ]
```

Parameters

ethernet *interface-name*

Ethernet interface for which to display information. Use the *slot/port* format (for example, 1/1). The slash (/) is required.

port-channel *port-channel-number*

LAG interface for which to display information. Use an integer from 1 to 255.

Modes

EXEC (#)

Output

The **show mlag interfaces** command displays the following information:

Output field	Description
Interface Identifier	Unique ID of the interface.
Interface	Name of the interface.
Peer	Name of the MLAG peer containing the interface.
Local/Remote State	Operational status of the MLAG member port (the local port and the port on the peer respectively). This is either UP, DOWN, OR UNKNOWN.
Cfg/Act Member	Number of interfaces that are configured/active members in each MLAG interface (Ethernet or port channel).
Up Count	Number of MLAG interface UP/DOWN transitions
Exceptions	Errors on the interface. (In this release, the only possible exception is Interface Down.)

Examples

The following example displays information about all MLAG interfaces that are configured on the device. In this example, the Local/Remote State of the MLAG interface Ethernet 0/16 is UNKNOWN:

```
device# show mlag interfaces

Interface Identifier: 1001
  Interface: ethernet 0/4
Peer      Local/Remote State Cfg/Act Member Up Count Exceptions
-----
mlag-peer    UP / UNKNOWN          1 / 1          0

Interface Identifier: 10001
  Interface: ethernet 0/16
Peer      Local/Remote State Cfg/Act Member Up Count Exceptions
-----
mlag-peer    UNKNOWN / UNKNOWN      0 / 0          0
device#
```

The following example displays information about all MLAG interfaces that are configured on the device. In this example, the states of the MLAG interfaces are either up or down:

```
show mlag interfaces

Interface Identifier: 101
  Interface: ethernet 0/1:2
Peer      Lcl/Rmt State Rmt Cfg/Act Member Up Count Exceptions
-----
Peer1     UP / UP        1 / 1          11

Interface Identifier: 102
  Interface: ethernet 0/1:3
Peer      Lcl/Rmt State Rmt Cfg/Act Member Up Count Exceptions
-----
Peer1     UP / UP        1 / 1          11

Interface Identifier: 103
  Interface: ethernet 0/1:4
Peer      Lcl/Rmt State Rmt Cfg/Act Member Up Count Exceptions
-----
Peer1     UP / UP        1 / 1          11

Interface Identifier: 51
  Interface: port-channel 11
Peer      Lcl/Rmt State Rmt Cfg/Act Member Up Count Exceptions
-----
Peer1     UP / UP        2 / 2          16

Interface Identifier: 1012
  Interface: port-channel 12
Peer      Lcl/Rmt State Rmt Cfg/Act Member Up Count Exceptions
-----
Peer1     DOWN / DOWN      2 / 0          13
```

The following example displays information about MLAG Ethernet interface 0/1 on the device:

```
device# show mlag interfaces ethernet 0/1

Interface Identifier: 1001
```



```

Interface: ethernet 0/1
Peer          Local/Remote State Cfg/Act Member  Up Count  Exceptions
-----
mlag-peer     UP / UNKNOWN      1 / 1          0
device#

```

The following example displays information about the MLAG port-channel interface with a port number of 4 on the device:

```

device# show mlag interfaces port-channel 4

Interface Identifier: 0
Interface: port-channel 4
Peer          Local/Remote State Cfg/Act Member  Up Count  Exceptions
-----
mypeer1       UP / UNKNOWN      0 / 0          0
device#

```

show mlag peer

Use the **show mlag peer** command to display details about the multichassis link aggregation (MLAG) peers that are configured on the device.

Syntax

```
show mlag peer [ extended-bridges ] [ peer-name ]
```

Parameters

extended-bridges

(Optional) Displays information about the extended bridge domains.

peer-name

(Optional) Name of a specific peer.

Modes

EXEC (#)

Output

The **show mlag peer** command displays the following information:

Output field	Description
Peer	Name of the peer.
Peer State	Operational state of the peer (UP, DOWN, OR UNKNOWN).
MCP State	MLAG Control Protocol (MCP) state (UP, DOWN, OR UNKNOWN).
Role	Priority of the peer's role. The role is chosen by protocol. The possible values are PRIMARY or BACKUP (also called secondary).
Elected MAC	MAC address used by both devices after the MLAG session is up.

Output field	Description
Extend Bridge Count	Actual bridge domain count that is extended to the other peer.
Keepalive <i>name</i>	Details about the named keepalive on the peer: • Destination Address—IPv4 or IPv6 address of the keepalive destination. • State—Operational state of the keepalive (UP, DOWN, or UNKNOWN). • Type—Keepalive type. The possible values are PRIMARY or SECONDARY. Two keepalive sessions are configured in an MLAG. The primary keepalive is used for Inter-Switch Link (ISL) connectivity, and the secondary keepalive is the management interface (by default). • Address Type—Address type for MLAG peer keepalive mode. The possible values are AUTO or MANUAL. Using AUTO for a secondary keepalive automatically chooses the management interface as the source interface and the peer's management IP address as the destination IP address (which is learned via an MLAG sync message). Using auto for a primary keepalive has no effect. • Source Interface—Name of the source interface for the MLAG peer keepalive. • First RX—Time when the first keepalive message was received. • Last RX—Time when the last keepalive message was received.

The **show mlag peer extended-bridges** command displays the following information:

Output field	Description
Peer	Name of the peer.
No. of Extend Bridge	Number of bridge domains that are extended across the MLAG peer.
Extend Bridges	List of bridge domains that are extended across the MLAG peer.

Examples

The following example displays information about the MLAG peer that is configured on the device:

```
device# show mlag peer
```

```
Peer BETA-8730-2
=====
```

```
Peer State      : UP
MCP State       : UP
```

```

Role                : BACKUP
Elected MAC        : 50:0a:9c:97:90:00
Extend Bridge Count : 1614
Peer Exception      : None

  Keepalive peer-2
  -----
  Destination Address: 10.1.1.1
  State              : UP
  Type               : PRIMARY
  Address Type       : MANUAL
  Source Interface   : port-channel 60 ( 10.1.1.0 )
  First Rx           : 0000-00-00 00:00:00
  Last Rx            : 0000-00-00 00:00:00

  Keepalive peer-22
  -----
  Destination Address: 10.38.132.25
  State              : UP
  Type               : SECONDARY
  Address Type       : AUTO
  Source Interface   : management 0 ( 10.38.132.24 )
  First Rx           : 2025-06-11 18:58:33
  Last Rx            : 2025-06-11 19:22:03
device#

```

The following example displays information about the MLAG peer extended bridges that are configured on the device:

```

device# show mlag peer extended-bridges

Peer dut3
=====
No. of Extend Bridge: 34
Extend Bridges      :
1,100,200-202,205,209,215-216,300-302,400,402,500-506,1001,2001,2500-2501,3001-3002,3005-3
006,4000-4003,8192
device#

```

show mlag peer bfd

Use the **show mlag peer bfd** command to display Bidirectional Forwarding Detection (BFD) details about the multichassis link aggregation (MLAG) primary keepalive state on the device.

Syntax

```
show mlag peer bfd
```

Parameters

This command has no keywords or variables.

Modes

EXEC (#)

Output

The **show mlag peer bfd** command displays the following information:

Output field	Description
Peer	Name of the MLAG peer.
Keepalive <i>name</i>	Details about the named keepalive on the peer (this is the primary keepalive name configured under the MLAG peer): • Profile—BFD profile on the peer, which contains the BFD settings that are applied (or requested to be applied) to a routing protocol that is configured (registered) as a BGP client (meaning that it uses BFD for link failure detection instead of its own mechanism). • Session ID—ID of the keepalive session on the peer, which sends keepalive messages to other peers to check that the link between them is operating or to prevent the link from being broken. • Session Status—Operational state of the keepalive on the peer. This is either UP or DOWN.

Examples

The following example displays BFD details on the MLAG peer that is configured on the device:

```
device# show mlag peer bfd

Peer mlag-peer
=====

Keepalive one
```

```
-----  
Profile      : default  
Session ID   : 1  
Session Status : Up  
device#
```

show overlay-networks

Use the **show overlay-networks** command to display information about the Network Virtualization Overlay (NVO) instances that are configured currently on the device.

Syntax

```
show overlay-networks
```

Parameters

This command has no keywords or variables.

Modes

EXEC (#)

Output

The **show overlay-networks** command displays the following information:

Output field	Description
default-nve <i>name</i>	Name of the default Network Virtualization Edge (NVE) endpoint (also called an NVE node).
default-vni-offset	Default Virtual Network Identifier (VNI) domain automatic offset.
nvo <i>name</i>	Name of an NVO instance.
member nves	List of NVEs that are members of the NVO instance.
vni-domain <i>name</i>	Name of a VNI domain that is a member of the NVE.

Examples

The following example displays information about the NVO instances that are configured currently on the device:

```
device# show overlay-networks

overlay-networks
  default-nve base
  default-vni-offset 0
  nvo base
    member nves:  base
  nve base
    vni-domain base
  vni-domain base
device#
```

show qos interface

Use the **show qos interface** command to display the CoS-to-traffic-class and DSCP-to-traffic-class mappings in the traffic class maps and remarking traffic class maps that are attached to the interfaces.

Syntax

```
show qos interface [ ethernet ethernet-interface-name subinterface  
    { untagged | vlan vlan-id } | ve ve-interface-name ]
```

Parameters

ethernet-interface-name

(Optional) Ethernet interface name. Use the *slot/port* format or the *slot/port:breakout* format to specify an interface or one or more comma-separated ranges of interfaces respectively. For example, 0/1 or 1/1-3,5,2/7-9,10:1-4 respectively.

untagged

(Optional) Displays information for untagged subinterfaces. Untagged subinterfaces are ones that are not associated with a specific VLAN.

vlan-id

(Optional) VLAN ID. Use an integer from 1 to 4095.

ve-interface-name

(Optional) Virtual Ethernet (VE) interface name. Use an integer or one or more comma-separated ranges of integers from 1 to 8192. For example, 2 or 2-3,5,7-9.

Modes

EXEC (#)

Usage Guidelines

You can use the **show running-config qos** command to display the QoS configuration that is running currently on a device.

Output

The **show qos interface** command displays the following information:

Output field	Description
Interface	Name of the interface to which the maps are attached.
Traffic-Class Map	Name of the traffic class map that is attached to the interface.

Output field	Description
Default Traffic Class for map	Value of the default traffic class that is configured on the device. For details, see default-traffic-class on page 555.
COS-to-Traffic-Class Map	Table of CoS-to-traffic-class mappings that are configured in the named COS-to-Traffic-Class map that is attached to the interface.
DSCP-to-TC Map	Table of DSCP-to-traffic-class mappings that are configured in the named DSCP-to-traffic-class map that is attached to the interface. A DSCP value is a two-digit integer from 0 to 63. To determine the traffic class value that corresponds to a given DSCP value, look up the row that is marked by the first digit ("d1"), and then look up the column that is marked by the second digit ("d2"). The corresponding traffic class value is at the intersection of the row and the column. For example, DSCP 40 corresponds to traffic class 5.
Remark Map	Name of the remark map that is attached to the interface. This map is for remarking the QoS markings on packets.

Examples

The following example displays the mappings within the default traffic class map on the device:

```
device# show qos interface ethernet 0/1:2 subinterface vlan 200

Interface: ethernet 0/1:2.200
Traffic-Class Map: default
Default Traffic Class for map default: 0
COS-to-Traffic-Class Map: default
  COS                : 0  1  2  3  4  5  6  7
  -----
  Traffic-Class      : 1  0  2  3  4  5  6  7

DSCP-to-TC Map: default (DSCP = d1d2)
d1 : d2 0   1   2   3   4   5   6   7   8   9
-----
0 :   0   0   0   0   0   0   0   0   1   1
1 :   1   1   1   1   1   1   2   2   2   2
2 :   2   2   2   2   3   3   3   3   3   3
3 :   3   3   4   4   4   4   4   4   4   4
4 :   5   5   5   5   5   5   5   5   6   6
5 :   6   6   6   6   6   6   7   7   7   7
6 :   7   7   7   7   7   7   7   7   7   7

Remark Map: default
Traffic-Class-to-COS Map: default
  Traffic-Class      : 0  1  2  3  4  5  6  7
  -----
  COS                : 1  0  2  3  4  5  6  7

Traffic-Class-to-DSCP Map: default
  Traffic-Class      : 0   1   2   3   4   5   6   7
  -----
```

```
device# DSCP : 15 7 23 31 39 47 55 63
```

show qos nvo

Use the **show qos nvo** command to display the fabric QoS configurations of the Network Virtualization Overlay (NVO) endpoints.

Syntax

```
show qos nvo [ nvo-name ]
```

Parameters

nvo-name

(Optional) Displays the fabric QoS configuration for the specified NVO endpoint.

Modes

EXEC (#)

Usage Guidelines

Use the **show qos nvo** command to display the fabric QoS configuration of all NVO endpoints or a specified NVO endpoint.

Examples

The following example displays the fabric QoS configurations of all NVO endpoints on the device:

```
device# show qos nvo

nvo test
  tunnel-mode:
    encapsulation pipe dscp 34
    decapsulation pipe
device#
```

show qos remark-map

Use the **show qos remark-map** command to display the mappings in the specified traffic class map for remarking the QoS markings on packets.

Syntax

```
show qos remark-map [ map-name | default ]
```

Parameters

map-name

Name of the traffic class map for remarking the QoS markings on packets.

default

Displays the tables of the default traffic class map for remarking the QoS markings on packets.

Modes

EXEC (#)

Usage Guidelines

You can use the **show running-config qos** command to display the QoS configuration that is running currently on a device.

Output

The **show qos remark-map** command displays the following information:

Output field	Description
Default Traffic Class	Value of the default traffic class that is configured on the device. For details, see default-traffic-class on page 555.
Traffic-Class-to-COS Map	Table of CoS-to-traffic-class mappings that are configured in the specified map.
Traffic-Class-to-DSCP Map	Table of DSCP-to-traffic-class mappings that are configured in the specified map. A DSCP value is a two-digit integer from 0 to 63. To determine the traffic class value that corresponds to a given DSCP value, look up the row that is marked by the first digit ("d1"), and then look up the column that is marked by the second digit ("d2"). The corresponding traffic class value is at the intersection of the row and the column. For example, DSCP 40 corresponds to traffic class 5.

Examples

The following example displays the mappings within a QoS remark map named map2 on the device:

```
device# show qos remark-map map2

Traffic-Class-to-COS Map: map2
Traffic-Class      : 0  1  2  3  4  5  6  7
-----
COS                : 1  0  2  3  3  5  6  7

Traffic-Class-to-DSCP Map: map2
Traffic-Class      : 0      1      2      3      4      5      6      7
-----
DSCP               : 8      0      16     24     59     40     48     56
device#
```

show qos traffic-class-map

Use the **show qos traffic-class-map** command to display the CoS-to-traffic-class and DSCP-to-traffic-class mappings in a traffic class map.

Syntax

```
show qos traffic-class-map [ map-name | default ]
```

Parameters

map-name

Name of the traffic class map.

default

Displays the tables of CoS-to-traffic-class and DSCP-to-traffic-class mappings in the default traffic class map.

Modes

EXEC (#)

Usage Guidelines

You can use the **show running-config qos** command to display the QoS configuration that is running currently on a device.

Output

The **show qos** command displays the following information:

Output field	Description
Default Traffic Class	Value of the default traffic class that is configured on the device. For details, see default-traffic-class on page 555.
Traffic-Class-to-COS Map	Table of CoS-to-traffic-class mappings that are configured in the specified map.
Traffic-Class-to-DSCP Map	Table of DSCP-to-traffic-class mappings that are configured in the specified map. A DSCP value is a two-digit integer from 0 to 63. To determine the traffic class value that corresponds to a given DSCP value, look up the row that is marked by the first digit ("d1"), and then look up the column that is marked by the second digit ("d2"). The corresponding traffic class value is at the intersection of the row and the column. For example, DSCP 40 corresponds to traffic class 5.

Examples

The following example displays the mappings within the default traffic class map on the device:

```
device# show qos traffic-class-map default

Default Traffic Class for map default: 0
COS-to-Traffic-Class Map: default
  COS           : 0  1  2  3  4  5  6  7
  -----
  Traffic-Class : 1  0  2  3  4  5  6  7

DSCP-to-Traffic-Class Map: default (DSCP = d1d2)
  d1 : d2 0   1   2   3   4   5   6   7   8   9
  -----
  0 :    0  0    0    0    0    0    0    0    1    1
  1 :    1  1    1    1    1    1    2    2    2    2
  2 :    2  2    2    2    3    3    3    3    3    3
  3 :    3  3    4    4    4    4    4    4    4    4
  4 :    5  5    5    5    5    5    5    5    6    6
  5 :    6  6    6    6    6    6    7    7    7    7
  6 :    7  7    7    7    6    6    7    7    7    7
device#
```

show running-config access-list

Use the **show running-config access-list** command to display IPv4, IPv6, or MAC access control list (ACL) configuration that is running currently on the device.

Syntax

```
show running-config { ipv4 | ipv6 | mac } [ access-list acl-name | all ]
```

```
show running-config access-list
```

Parameters

ipv4

Displays IPv4 ACLs.

ipv6

Displays IPv6 ACLs.

mac

Displays MAC ACLs.

acl-name

Name of the ACL to be displayed. You can use any combination of 1 to 64 uppercase or lowercase alphabetic characters, numerals, underscores, hyphens, or periods. You must use an alphabetic character or an underscore as the first character.

all

Displays all IPv4 ACLs, IPv6 ACLs, or MAC ACLs.

Modes

EXEC (#)

Output

For IPv4 and IPv6 ACLs, the **show running-config access-list** command displays the following information:

Output field	Description
seq <i>sequence-id</i>	(If present) Sequence number to identify the relative priority of this rule in the ACL. For example, seq 10.
<i>rule-type</i>	Indicates whether the rule denies or permits forwarding of packets. The possible values are deny or permit.

Output field	Description
<i>protocol</i>	Protocol for which to match packets. Following are the possible protocols: • <i>ip-protocol-num</i> matches packets that use a specific IP protocol. For example, 6 matches TCP packets, and 115 matches L2TP packets. For a list of IP protocol numbers, see https://www.iana.org/assignments/protocol-numbers/protocol-numbers.xhtml • <i>ip</i> matches packets that use any IPv4 protocol (including ICMPv4, IGMPv4, UDP, and TCP). • <i>icmp</i> matches packets carrying Internet Control Message Protocol v4 (ICMPv4) messages. These packets contain the control messages such as Echo Request/Echo Reply or Destination Unreachable. • <i>icmpv6</i> matches packets carrying Internet Control Message Protocol v6 (ICMPv6) messages. These packets contain the control messages such as Echo Request/ Echo Reply or Destination Unreachable. • <i>igmp</i> matches packets carrying Internet Group Management Protocol v4 (IGMPv4) messages. Hosts and adjacent routers use these on IPv4 networks to establish multicast group memberships, which optimize performance for one-to-many networking applications such as streaming video. • <i>igmpv6</i> matches packets carrying Multicast Listener Discovery (MLD) messages. Hosts and adjacent routers use these on IPv6 networks to establish multicast group memberships, which optimize performance for one-to-many networking applications such as streaming video. • <i>udp</i> matches packets that use the User Datagram Protocol (UDP) format. This allows packets using connectionless protocols such as TFTP and DNS. • <i>tcp [flag-list]</i> matches TCP packets. This allows packets that use TCP connection-oriented protocols such as HTTPS, FTP, and Telnet. This field might include a space-separated combination of optional TCP flags of the permit (IPv4 ACL configuration) command. The absence of flags matches all TCP packets.
<i>udp</i>	Matches packets that use the User Datagram Protocol (UDP) format. This allows packets using connectionless protocols such as TFTP and DNS.
<i>tcp [flag-list]</i>	Matches TCP packets. This allows packets that use TCP connection-oriented protocols such as HTTPS, FTP, and Telnet. This field might include a space-separated combination of the flags of the permit (IPv4 ACL configuration) command. The absence of flags matches all TCP packets.

Output field	Description
<i>source-addresses</i> <i>destination-addresses</i>	Source and destination IPv4 or IPv6 addresses. For source and destination addresses respectively, the possible values are: <i>source-address subnet-mask</i> <i>source-address / prefix-length</i> any Source IPv4 or IPv6 addresses. These represent the networks or hosts sending the packets and use one of the following methods: • <i>source-address subnet-mask</i> matches packets from a range of addresses via a mask. • <i>source-address / prefix-length</i> matches packets from a range of addresses via a subnet prefix length. • any matches packets from any source IPv4 or IPv6 address. <i>destination-address subnet-mask</i> <i>destination-address/prefix-length</i> any Destination IPv4 or IPv6 addresses. These represent the networks or hosts to receive the packets and use one of the following methods: • <i>destination-address subnet-mask</i> matches packets going to a range of addresses via a mask. • <i>destination-address / prefix-length</i> matches packets going to a range of addresses via a subnet prefix length. • any matches packets to any destination IPv4 or IPv6 address. For IPv4 addresses, the possible values are: <i>source-ipv4-address subnet-mask</i> <i>source-ipv4-address / prefix-length</i> any Source IPv4 addresses. These represent the networks or hosts sending the packets. These use one of the following methods: • <i>source-ipv4-addr subnet-mask</i> matches packets from a range of IPv4 addresses via a subnet mask. • <i>source-ipv4-addr/prefix-length</i> matches packets from a range of IPv4 addresses via a subnet prefix length. • any matches packets from any source IPv4 address. <i>dest-ipv4-addr subnet-mask</i> <i>dest-ipv4-addr/prefix-length</i> any Destination IPv4 addresses. These represent the networks or hosts to receive the packets. These use one of the following methods: • <i>dest-ipv4-addr subnet-mask</i> matches packets going to a range of IPv4 addresses via a subnet mask. • <i>dest-ipv4-addr/prefix-length</i> matches packets going to a range of IPv4 addresses via a subnet prefix length.

Output field	Description
	any matches packets to any destination IPv4 address. For IPv6 addresses, the possible values are: <code>source-ipv6-addr subnet-mask source-ipv6-addr/prefix-length any</code> Source IPv6 addresses. These represent the networks or hosts sending the packets. These use one of the following methods: <code>source-ipv6-addr subnet-mask</code> matches packets from a range of IPv6 addresses via a subnet mask. <code>source-ipv6-addr/prefix-length</code> matches packets from a range of IPv6 addresses via a subnet prefix length. any matches packets from any source IPv6 address. <code>dest-ipv6-addr subnet-mask dest-ipv6-addr/prefix-length any</code> Destination IPv6 addresses. These represent the networks or hosts sending the packets. These use one of the following methods: <code>dest-ipv6-addr subnet-mask</code> matches packets going to a range of IPv6 addresses via a subnet mask. <code>dest-ipv6-addr/prefix-length</code> matches packets going to a range of IPv6 addresses via a subnet prefix length. any matches packets to any destination IPv6 address.
<code>fragmentation-type</code>	(If present) (applies to IPv4 ACLs only) Matches only nonfragmented or only fragmented packets. The possible values are dontfragment or morefragment.
<code>dport</code>	(If present) Destination layer 4 port. For example, 22 (for SSH traffic) or 443 (for HTTPS traffic). For details, see https://www.iana.org/assignments/service-names-port-numbers/service-names-port-numbers.xhtml .
<code>sport</code>	(If present) Source layer 4 port. For details, see https://www.iana.org/assignments/service-names-port-numbers/service-names-port-numbers.xhtml .
<code>dscp</code>	(If present) Matches packets containing a specific Differentiated Services Code Point (DSCP) value. Differentiated services are for managing traffic and providing quality of service (QoS). For example, they can provide low latency to voice or streaming media but only best effort service to web traffic or file transfers.
<code>vlan vlan-id</code>	(If present) Matches packets containing a specific VLAN ID. This applies the rule to a VLAN with an ID other than 1 (the default VLAN ID). This applies to a physical LAN that is partitioned into VLANs (logical broadcast domains to isolate services for improved security and management).

Output field	Description
count	(If present) Indicates that the counter is enabled. for this rule.
log	(If present) Indicates that a copy of matching packets is being sent to the CPU for logging.
(<i>number</i> Packets, <i>number</i> Bytes)	Indicates the number of times that the rule matched a packet as well as the byte count for those packets.

For MAC ACLs, the **show running-config access-list** command displays the following information:

Output field	Description
seq <i>sequence-id</i>	(If present) Sequence number to identify the relative priority of this rule in the ACL. For example, seq 10.
<i>rule-type</i>	Indicates whether the rule denies or permits forwarding of packets. The possible values are deny or permit.
<i>source-addresses</i> <i>destination-addresses</i>	Source and destination MAC addresses. For source and destination addresses respectively, the possible values are: <i>source-address address-mask</i> any Source MAC addresses. These represent the networks or hosts sending the packets and use one of the following methods: • <i>source-address address-mask</i> matches packets from a range of addresses via a mask. • any matches packets from any source MAC address. <i>destination-address address-mask</i> any Destination MAC addresses. These represent the networks or hosts to receive the packets. These use one of the following methods: • <i>destination-address address-mask</i> matches packets going to a range of addresses via a mask. • any matches packets from any destination MAC address.
etype <i>ethernet-type</i>	(If present) Ethernet type. This is a hexadecimal number in the range of 0x600 to 0xFFFF or the arp, ipv4, or ipv6 protocol.
pcp <i>dot1p_or_cos_value</i>	(If present) 802.1p or class of service (CoS) value for payload compression protocol (PCP). This is an integer from 1 to 7.
vlan <i>vlan-id</i>	(If present) Matches packets containing a specific VLAN ID. This applies the rule to a VLAN with an ID other than 1 (the default VLAN ID). This applies to a physical LAN that is partitioned into VLANs (logical broadcast domains to isolate services for improved security and management).

Output field	Description
count	(If present) Indicates that the counter is enabled. for this rule.
log	(If present) Indicates that a copy of matching packets is being sent to the CPU for logging.
(<i>number</i> Packets, <i>number</i> Bytes)	Indicates the number of times that the rule matched a packet as well as the byte count for those packets.

Examples

The following example displays the currently-running rules in an IP ACL named v4acl:

```
device# show running-config ip access-list v4acl

ip access-list v4acl
seq 10 permit ip 1.1.1.1 255.255.255.255 2.2.2.2 255.255.255.255 dport 44 count ( 0
Packets, 0 Bytes)
device#
```

The following example displays the currently-running rules in an IPv6 ACL named v6acl:

```
device# show running-config ipv6 access-list v6acl

ipv6 access-list v6acl
seq 10 permit ipv6 1::1 ffff:ffff:ffff:ffff:ffff:ffff:ffff:ffff 2::2
ffff:ffff:ffff:ffff:ffff:ffff:ffff:ffff sport 22 log ( 0 Packets, 0 Bytes)
device#
```

The following example displays the currently-running rules in a MAC ACL named macacl:

```
device# show running-config mac access-list macacl

mac access-list macacl
seq 10 permit 11:22:33:44:00:00 ff:ff:ff:ff:ff:00 any log count ( 0 Packets, 0 Bytes)
device#
```

The following example displays the currently-running rules in all configured IP ACLs:

```
device# show running-config ip access-list all

ip access-list v4acl
seq 10 permit ip 1.1.1.1 255.255.255.255 2.2.2.2 255.255.255.255 dport 44 count ( 0
Packets, 0 Bytes)
device#
```

The following example displays all currently-running ACLs and their corresponding rules:

```
device# show running-config access-list

ip access-list v4acl
seq 10 permit ip 1.1.1.1 255.255.255.255 2.2.2.2 255.255.255.255 dport 44 count
ipv6 access-list v6acl
seq 10 permit ipv6 1::1 ffff:ffff:ffff:ffff:ffff:ffff:ffff:ffff 2::2
ffff:ffff:ffff:ffff:ffff:ffff:ffff:ffff sport 22 log
mac access-list macacl
```

```
seq 10 permit 11:22:33:44:00:00 ff:ff:ff:ff:ff:00 any log count  
device#
```

show running-config lldp

Use the **show running-config lldp** command to display the LLDP configuration that is running currently on the device.

Syntax

```
show running-config lldp
```

Parameters

This command has no keywords or variables.

Modes

EXEC (#)

Examples

The following example displays the LLDP configuration that is running currently on the device. In this example, LLDP is enabled globally. Also, LLDP is configured with packet transmission disabled, the global LLDP transmit interval is 100 seconds, the period that a receiving device holds information from your device before discarding it is 100 seconds, the LLDP management address is 1.1.1.1, and several optional TLV advertisements are suppressed:

```
device# show running-config lldp

protocol lldp
lldp
  no transmit
  hello-timer 100
  holdtime 100
  management-address 1.1.1.1
  suppress-tlv management-address,port-description,system-name
device#
```

show system mac-address-table aging-time (Extreme ONE OS Switching)

Use the **show system mac-address-table aging-time** command to display the aging time that is configured for the MAC address forwarding table on a device.

Syntax

```
show system mac-address-table aging-time
```

Parameters

This command has no keywords or variables.

Modes

EXEC (#)

Usage Guidelines

Use this command to display the setting for the maximum amount of time (in seconds) that a MAC address remains in the MAC address table. When this time expires, the MAC address is deleted.

Examples

The following example displays the setting for the MAC address table aging time on the device. In this example, a MAC address ages out after 61 seconds and is then discarded:

```
device# show system mac-address-table aging-time

Mac Aging Time: 61
device#
```


show system maintenance-mode

Use the **show system maintenance-mode** command to display details about maintenance mode on a device.

Syntax

```
show system maintenance-mode
```

Parameters

This command has no keywords or variables.

Modes

EXEC (#)

Usage Guidelines

You use the maintenance mode feature when a device needs maintenance for reasons such as an image upgrade. When maintenance mode is active, all front panel ports are disabled to isolate the device from all connected devices on the network.

During maintenance mode, all physical ports are in the "link down" state. The connected devices detect that the link is down and divert the traffic to a peer multichassis link aggregation (MLAG) node.

Examples

The following example displays the maintenance mode details of the device. In this example, `Enable on reload : disabled` indicates that maintenance mode is not configured to start upon system reload. `Maintenance Mode : enabled` indicates that maintenance mode is presently enabled on the device. `Maintenance Mode Status : Complete` indicates that the system is not in the process of activating or de-activating maintenance mode:

```
device# show system maintenance-mode

Enable on reload      : disabled
Maintenance Mode      : enabled
Maintenance Mode Status : Complete
device#
```

show system redundant-management

Use the **show system redundant-management** command to display the configuration of a port that is redundant to the out of band (OOB) management port on a device.

Syntax

```
show system redundant-management
```

Parameters

This command has no keywords or variables.

Modes

EXEC (#)

Usage Guidelines

Use this command to display details of an ASIC front panel port that is configured as a redundant port of the out of band (OOB) management port. This feature provides management traffic path redundancy by allowing two ports to connect to the management LAN. This feature forwards management traffic to the CPU via an ASIC front panel port in both directions if the OOB management port fails..

Examples

The following example displays the redundant management configuration on the device. In this example, Ethernet port 0/15 is configured as the Redundant Management Ethernet (RME) port in internal port mode on the device:

```
device# show system redundant-management  
  
port ethernet 0/15 performance-mode internal 0  
device#
```

shutdown (BGP peer group configuration)

Use the **shutdown** command to administratively shut down a specific BGP neighbor or all neighbors in the specified BGP peer group within a VRF instance.

Syntax

shutdown

no shutdown

Command Default

The members of the specified BGP peer group on the device are enabled (not shut down). The members of the specified BGP peer group are not shut down.

Parameters

This command has no keywords or variables.

Modes

VRF BGP peer group configuration (config-vrf-bgp-pg)

VRF BGP peer group neighbor configuration (config-vrf-bgp-pg-neighbor)

Usage Guidelines

Use this command to temporarily disable BGP sessions for a specific BGP neighbor or all neighbors of the specified BGP peer group without removing them as neighbors altogether.

Use the **no** form of this command to disable shutdown of BGP sessions for a specific BGP neighbor or all neighbors of the specified BGP peer group.

Examples

The following example configures a BGP peer group named `group1` under a Virtual Routing and Forwarding (VRF) instance named `violet`. This example performs an administrative shutdown of all members in the `group1` peer group:

```
device# configure terminal
device(config)# vrf violet
device(config-vrf-violet)# router bgp
device(config-vrf-bgp)# peer-group group1
device(config-vrf-bgp-pg)# shutdown
device(config-vrf-bgp-pg)#
```

The following example displays the configuration of a VRF instance named violet that is running currently on the device. In this example, the group1 peer group members are currently shut down:

```
device# show running-config vrf violet

vrf violet
!
  router bgp
    peer-group group1
      shutdown
!
device#
```

The following example configures a BGP peer group named group1 under a VRF instance named violet. Neighbor 10.1.1.1 and neighbor 20.1.1.1 within the peer group are administratively shut down:

```
device# configure terminal
device(config)# vrf violet
device(config-vrf-violet)# router bgp
device(config-vrf-bgp)# peer-group group1
device(config-vrf-bgp-pg)# neighbor 10.1.1.1
device(config-vrf-bgp-pg-neighbor)# shutdown
device(config-vrf-bgp-pg-neighbor)# exit
device(config-vrf-bgp-pg)# neighbor 20.1.1.1
device(config-vrf-bgp-pg-neighbor)# shutdown
device(config-vrf-bgp-pg-neighbor)#
```

The following example displays the configuration of a VRF instance named violet that is running currently on the device. In this example, neighbor 10.1.1.1 and neighbor 20.1.1.1 are administratively shut down:

```
device# show running-config

vrf violet
!
  router bgp
    peer-group group1
      neighbor 10.1.1.1
        shutdown
      neighbor 20.1.1.1
        shutdown
!
device#
```

shutdown (Ethernet interface configuration)

Use the **shutdown** command to shut down an Ethernet interface on a device.

Syntax

```
shutdown  
no shutdown
```

Command Default

The current Ethernet interface on the device is enabled (not shut down).

Parameters

This command has no keywords or variables.

Modes

Ethernet interface configuration (config-if-eth-*name*)

Usage Guidelines

This command places an Ethernet interface in administrative shutdown status.

If you want to view the status of the Ethernet interfaces on a device, see [show interface brief](#) on page 247.

To disable the administrative shutdown of an Ethernet interface on the device, use the **no** form of this command.

This command is useful for new interfaces and for troubleshooting interfaces. Using **shutdown** followed by **no shutdown** sometimes fixes an interface that is not working properly.

Examples

The following example disables the administrative shutdown of Ethernet interface 0/1 on the device:

```
device# configure terminal  
device(config)# interface ethernet 0/1  
device(config-if-eth-0/1)# no shutdown  
device(config-if-eth-0/1)#
```

The following example displays the Ethernet interface configuration that is running currently on the device. In this example, Ethernet interface 0/1 is enabled by the **no shutdown** command:

```
device# show running-config interface ethernet

interface ethernet 0/1
  no shutdown
  description Ethernet interface 0/1
  fec rs-fec
  link-fault-signaling
  mtu 4000
  speed 10
  loopback-mode
!
device#
```

shutdown (subinterface configuration)

Use the **shutdown (subinterface configuration)** command to shut down a subinterface on a device.

Syntax

shutdown

no shutdown

Command Default

The current subinterface on the device is enabled (not shut down).

Parameters

This command has no keywords or variables.

Modes

Subinterface configuration (config-subif-*name*)

Usage Guidelines

This command places a subinterface in administrative shutdown status.

To disable the administrative shutdown of a subinterface on the device, use the **no** form of this command.

This command is useful for new interfaces and for troubleshooting interfaces. Using **shutdown** followed by **no shutdown** sometimes fixes an interface that is not working properly.

Examples

The following example disables the administrative shutdown of VLAN subinterface 100 on the device:

```
device# configure terminal
device(config)# interface ethernet 0/9
device(config-if-eth-0/9)# subinterface vlan 100
device(config-subif-eth-0/9.100)# shutdown
device(config-subif-eth-0/9.100)#
```

The following example displays information about the interfaces and subinterfaces that are running currently on the device: In this example, VLAN subinterface 200 that is configured under Ethernet interface 0/9 is shut down:

```
device# show running-config interface ethernet 0/9

interface ethernet 0/9
```

```
subinterface vlan 100
  shutdown
subinterface vlan 200 inner-vlan 100-110
  shutdown
!
device#
```


shutdown (tunnel interface configuration)

Use the **shutdown** command to shut down a tunnel interface on a device.

Syntax

```
shutdown  
no shutdown
```

Command Default

The current tunnel interface on the device is enabled (not shut down).

Parameters

This command has no keywords or variables.

Modes

Tunnel interface configuration (config-tunnel-*name*)

Usage Guidelines

This command places a tunnel interface in administrative shutdown status. This command is useful for new interfaces and for troubleshooting interfaces. Using **shutdown** followed by **no shutdown** sometimes fixes an interface that is not working properly.

If you want to view the status of the tunnel interfaces on a device, see [show interface tunnel \(Extreme ONE OS Switching\)](#) on page 1016.

To disable the administrative shutdown of a tunnel interface on the device, use the **no** form of this command.

Examples

The following example disables the administrative shutdown of tunnel interface 1 on the device:

```
device# configure terminal  
device(config)# interface tunnel 1  
device(config-tunnel-1)# no shutdown  
device(config-tunnel-1)#
```

The following example displays the tunnel interface configuration that is running currently on the device. In this example, tunnel interface 1 is enabled by the **no shutdown** command:

```
device# show running-config interface tunnel  
  
interface tunnel 1
```

```
type vxlan
  source nve base
  destination 1.1.1.1
  no shutdown
!
device#
```

size (IAH VM disk configuration)

Use the **size** command to specify a disk size for a virtual machine (VM).

Syntax

size *size*

no size

Command Default

10000 megabytes (MB)

Parameters

size

Disk size in MB. Use an integer from 1000 to 40000.

Modes

IAH VM disk configuration (config-iah-vm-disk-*name*)

Usage Guidelines

The maximum amount of disk storage allocated for IAH depends on the platform.

If an image file is specified, its size is calculated automatically.

Use the **no** form of this command to restore the disk file size to the default value. Note that configuration is not allowed when the VM is in the deployed state.

To display details about a specific VM, see [show iah vm](#) on page 1006.

Examples

The following example specifies 4000 MB as the size for a disk named sda within a VM named Ubuntu1:

```
device# configure terminal
device(config)# iah
device(config-iah)# vm Ubuntu1
device(config-iah-vm-Ubuntu1)# disk sda
device(config-iah-vm-disk-sda)# size 4000
device(config-iah-vm-disk-sda)#
```

The following example displays the IAH configuration that is running currently on the device. In this example, a VM named Ubuntu2 uses 5000 MB as the size for a disk named sdb:

```
device# show running-config iah
```

```
iah
vm Ubuntu2
  description Ubuntu demo vm
  auto-boot true
  cpus 4
  ram 1024
  disk sdb
    size 5000
    format qcow2
    bus scsi
    file disk://iah/disk1.qcow2
  !
!
device#
```

source-interface (MLAG peer keepalive configuration)

Use the **source-interface** command to specify the source interface for the multichassis link aggregation (MLAG) peer keepalive.

Syntax

```
source-interface { ethernet interface-name | loopback port-number |  
  management 0 | port-channel port-channel-number }  
no source-interface
```

Command Default

No source interface for the MLAG peer keepalive is configured on the device.

Parameters

ethernet *interface-name*

Ethernet interface name. Use *slot/port* format. For example, 1/1.

loopback *port-number*

Loopback port number. Use an integer from 1 to 1024.

management 0

Management port 0.

port-channel *port-channel-number*

Port-channel number. Use an integer from 1 to 255.

Modes

MLAG peer keepalive configuration (config-mlag-peer-keepalive-*name*)

Usage Guidelines

You can use the **show mlag** command to display the MLAG peer keepalive source interface configured for a MLAG keepalive.

To remove the source interface for the MLAG peer keepalive, use the no form of this command.

Examples

The following example configures an MLAG peer named peer01 and also configures an Ethernet port named 0/1 as its keepalive source interface:

```
device# configure terminal  
device(config)# mlag  
device(config-mlag)# peer peer01  
device(config-mlag-peer-peer01)# keepalive keepalive01
```

```
device(config-mlag-peer-keepalive-keepalive01)# source-interface ethernet 0/1
device(config-mlag-peer-keepalive-keepalive01)#
```

The following example displays the MLAG configuration that is running currently on a device. In this example, MLAG is configured with a peer named peer01. The peer uses port channel 4 as its keepalive source interface:

```
device# show running-config mlag

mlag
!
  peer peer01
!
    keepalive keepalive01
      destination 12.2.2.2
      source-interface port-channel 4
!
device#
```

The following example displays the MLAG peer configuration on a device. In this example, MLAG is configured with a peer named mypeer1. The peer uses port channel number 4 as its keepalive source interface:

```
device# show mlag peer

Peer mypeer1
=====
Peer State           : UP
MCP State            : UP
Role                 : PRIMARY
Elected MAC         : e4:44:e5:00:94:00
Extend Bridge Count  : 361
Peer Exception       : None
  Keepalive prim
  -----
  Destination Address: 20.20.20.164
  State               : UP
  Type                : PRIMARY
  Address Type        : MANUAL
  Source Interface    : loopback 1024 ( 20.20.20.163 )
  First Rx            : 2025-06-27 12:11:27
  Last Rx             : 2025-06-27 12:50:59
  Keepalive Second
  -----
  Destination Address: 10.32.100.35
  State               : UP
  Type                : SECONDARY
  Address Type        : AUTO
  Source Interface    : management 0 ( 10.32.99.45 )
  First Rx            : 2025-06-27 12:11:41
  Last Rx             : 2025-06-27 12:50:59
device#
```

source-interface (NVE configuration)

Use the **source-interface** (NVE configuration) command to configure the source interface that is associated with a specific Network Virtualization Edge (NVE) endpoint (also called an NVE node).

Syntax

```
source-interface loopback loopback-id  
no source-interface loopback loopback-id
```

Command Default

No source interface is associated with the specific NVE endpoint.

Parameters

```
loopback loopback-id
```

ID number of the loopback interface. Use an integer from 1 to 1024.

Modes

NVE configuration (config-nve-*name*)

Usage Guidelines

Use the **no** form of this command to delete the source interface that is associated with a specific NVE endpoint.

Examples

The following example configures a loopback interface with ID number 1 that is associated with an NVE endpoint named base:

```
device# configure terminal  
device(config)# overlay-networks  
device(config-overlay)# nve base  
device(config-nve-base)# source-interface loopback 1  
device(config-nve-base)#
```

The following example shows the Network Virtualization Overlay (NVO) instances that are running currently on the device:

```
device# show running-config overlay-networks  
  
overlay-networks  
  default-nve base  
  default-vni-offset 0  
  nvo base  
    member nve base  
  !
```

```
nve base
  source-interface loopback 1
  vni-domain base
!
vni-domain base
!
device#
```

source nve

Use the **source nve** command to specify the source Network Virtualization Endpoint (NVE) to use for a tunnel interface that runs on the Virtual eXtensible Local-Area Network (VXLAN) protocol.

Syntax

source nve *nve-name*

no source nve

Command Default

The current tunnel interface has no source NVE specified.

Parameters

nve-name

Name of the source NVE to use. This name must start with an alphabetic character or an underscore followed by an arbitrary sequence of alphabetic or numeric characters, underscores, hyphens, or dots. The name cannot exceed 64 characters.

Modes

Tunnel interface configuration (config-tunnel-*name*)

Usage Guidelines

This command specifies an NVE, which is a logical interface that performs encapsulation and de-encapsulation of VXLAN traffic for a specific VXLAN Network Identifier (VNID). This interface acts as the entry and exit point for VXLAN traffic. A VNID (also called a VXLAN Segment ID) is a 24 bit ID that defines the broadcast domain and is similar to the VLAN ID for a regular VLAN.

You configure an NVE on a VXLAN Tunnel Endpoint (VTEP) physical device (such as a switch) that does the encapsulation and decapsulation. For example, if you have a VLAN 100 in your physical network, you might map that VLAN to a VNID (such as VNID 100) and configure the NVE on your VTEP device to handle the encapsulation and decapsulation for VNID 100. A VTEP device connects the overlay and underlay networks.

Before you can use an NVE, it must be configured with a valid IPv4 or IPv6 address and associated with the VRF. It should also have an underlay route to resolve the tunnel destination IP address.(Also, the NVO must be configured with an NVE, a VNI domain, and a split horizon group so that the tunnel will come up.)

For details about creating an NVE, see [nve \(overlay configuration\)](#) on page 831.

If you want to view the status and details of the tunnel interfaces on a device, see [show interface tunnel \(Extreme ONE OS Switching\)](#) on page 1016.

To remove the NVE from a tunnel interface, use the **no** form of this command.

Examples

The following example configures tunnel interface 1 to use a source NVE named base on the device:

```
device# configure terminal
device(config)# interface tunnel 1
device(config-tunnel-1)# source nve base
device(config-tunnel-1)#
```

The following example displays the tunnel interface configuration that is running currently on the device. In this example, the tunnel 1 VXLAN interface uses a source NVE named base:

```
device# show running-config interface tunnel

interface tunnel 1
  type vxlan
    source nve base
    destination 1.1.1.1
  no shutdown
!
device#
```

speed (Ethernet interface configuration)

Use the **speed** command to set the speed on an Ethernet port.

Syntax

```
speed { 10 | 100 | 1000 | 10000 | 25000 | 40000 | 100000 | 400000 |  
        auto }
```

Command Default

The speed on the specified Ethernet interface is auto-detected. The available modes depend on the hardware capability/mode.

Parameters

10

(SDWAN platforms only) Sets the speed to 10 megabits per second (Mbps).

100

(SDWAN platforms only) Sets the speed to 100 Mbps.

1000

Sets the speed to 1 gigabit per second (Gbps).

10000

Sets the speed to 10 Gbps.

25000

Sets the speed to 25 Gbps.

40000

Sets the speed to 40 Gbps.

100000

Sets the speed to 100 Gbps.

400000

Sets the speed to 400 Gbps.

auto

Auto-negotiates the speed.

Modes

Ethernet interface configuration (config-if-eth-*name*)

Usage Guidelines

If you want to view the status of the Ethernet interfaces on a device, see [show interface ethernet link-error-disable](#) on page 1012.

Examples

The following example configures Ethernet interface 0/1. In this example, the speed is set to 25000 (25 Gbps):

```
device# configure terminal
device(config)# interface ethernet 0/1
device(config-if-eth-0/1)# no shutdown
device(config-if-eth-0/1)# speed 25000
device(config-if-eth-0/1)#
```

The following example displays statistics about Ethernet interface 0/32. In this example, the speed is set to 25000 (25 Gbps):

```
device# show interface ethernet 0/32

ethernet 0/32 Admin state UP, Oper state DOWN
Interface index is 16778240 (0x1000400)
MTU 9216 bytes
Hardware is Ethernet  mac address 50:0a:9c:97:30:3c
Current Speed 25G
Description: Ethernet 0/32
FEC:
    Mode: RS-FEC
    Uncorrected Blocks: 18
    Corrected Blocks: 55475033
Link Fault Signaling: ON
Link Fault Status: No Fault
Auto Negotiation: ON
Carrier Transitions: 8
    LastClear: 0s
Input:
    Total pkts: 88
    Broadcast pkts: 0
    Discard pkts: 0
    Errors pkts: 0
    FCS Errors: 0
    MCast pkts: 88
    Octets: 15312
    UCast pkts: 0
    Runt pkts: 0
    CRC Errors: 0
Input Distribution:
    64 byte pkts: 0
    65-127 byte pkts: 0
    128-255 byte pkts: 88
    256-511 byte pkts: 0
    512-1023 byte pkts: 0
    1024-1518 byte pkts: 0
    Jumbo pkts: 0
Out:
    Total pkts: 94
    Broadcast pkts: 0
    Discard pkts: 0
    Errors pkts: 0
    MCast pkts: 94
    Octets: 15134
    UCast pkts: 0
Rate Info:
    Input: 0.000000 Mbits/sec, 0 pkts/sec 0.00% of line-rate
    Output: 0.000000 Mbits/sec, 0 pkts/sec 0.00% of line-rate
device#
```

speed (port channel interface configuration)

Use the **speed** command to set the speed on a port channel port.

Syntax

```
speed { 10000 | 100000 | 25000 | 40000 | 400000 | auto }
```

Command Default

The speed on the specified port channel interface is auto-detected. The available modes depend on the hardware capability and mode.

Parameters

10000

Sets the speed to 10 gigabit per second (Gbps).

100000

Sets the speed to 100 Gbps.

25000

Sets the speed to 25 Gbps.

40000

Sets the speed to 40 Gbps.

400000

Sets the speed to 400 Gbps.

auto

Auto-negotiates the speed.

Modes

Port Channel interface configuration (config-if-po-*name*)

Usage Guidelines

If you want to view the status of the port channel interfaces on a device, see [show interface](#) on page 241.

Examples

The following example configures port channel interface 1. In this example, the speed is set to 25000 (25 Gbps):

```
device# configure terminal
device(config)# interface port-channel 1
device(config-if-po-1)# no shutdown
```

```
device(config-if-po-1)# speed 25000
device(config-if-po-1)#
```

The following example displays statistics about port channel interface 1. In this example, the speed is set to 25000 (25 Gbps):

```
device# show interface port-channel 1

port-channel 1 is up
  MTU 4192 Bytes
  IfIndex 0x4000001
  Mac address is 50:0a:9c:97:8c:3d
  configured member port-speed 25G
  Port mode is Full Duplex, 0G
  MinLinks is 32
  Active Members in this channel: Nil
  Members in this channel: Nil

Statistics
  Carrier Transitions: 0
    LastClear: 0s
Input:
  Total pkts: 88
  Broadcast pkts: 0
  Discard pkts: 0
  Errors pkts: 0
  FCS Errors: 0
  MCast pkts: 88
    Octets: 15312
  UCast pkts: 0
  Runt pkts: 0
  CRC Errors: 0
Input Distribution:
  64 byte pkts: 0
  65-127 byte pkts: 0
  128-255 byte pkts: 88
  256-511 byte pkts: 0
  512-1023 byte pkts: 0
  1024-1518 byte pkts: 0
  Jumbo pkts: 0
Out:
  Total pkts: 94
  Broadcast pkts: 0
  Discard pkts: 0
  Errors pkts: 0
  MCast pkts: 94
    Octets: 15134
  UCast pkts: 0
Rate Info:
  Input: 0.000000 Mbits/sec, 0 pkts/sec 0.00% of line-rate
  Output: 0.000000 Mbits/sec, 0 pkts/sec 0.00% of line-rate
device#
```

split-horizon

Use the **split-horizon** command to associate a split horizon with a specific Network Virtualization Overlay (NVO).

Syntax

```
split-horizon split-horizon-name  
no split-horizon split-horizon-name
```

Command Default

No split horizons are associated with the specific NVO.

Parameters

split-horizon-name

Name of the split horizon to add. You can use any combination of 1 to 64 uppercase or lowercase alphabetic characters, numerals, underscores, hyphens, or periods. You must use an alphabetic character or an underscore as the first character.

Modes

NVO configuration (config-nvo-*name*)

Usage Guidelines

This command configures the split horizon associated with an NVO. All tunnels associated with the NVO will use this split horizon property.

Use the **no** form of this command to remove a split horizon from a specific NVO.

Examples

The following example associates a split horizon named DG1 to an NVO named base:

```
device# configure terminal  
device(config)# overlay-networks  
device(config-overlay)# nvo base  
device(config-nvo-base)# split-horizon SH1device(config-nvo-base)#
```

The following example shows the Network Virtualization Overlay (NVO) instances that are running currently on the device:

```
device# show running-config  
  
...  
  
overlay-networks  
  default-nve base
```

```
default-vni-offset 0
nvo base
  member nve base
  member vni-domain DG1
  split-horizon SH1
!
nve base
  source-interface loopback 1
!
vni-domain base
vni-domain DG1
  auto-offset 500000
!
!
device#
```


src-protocol

Use the **src-protocol** command to add table connections configurations for route redistribution to a specific VRF instance.

Syntax

```
src-protocol { connected | static } dst-protocol bgp address-family  
    { ipv4 | ipv6 }  
  
no src-protocol { connected | static } dst-protocol bgp address-family  
    { ipv4 | ipv6 }
```

Command Default

Table connection configurations for route redistribution are disabled for the specified VRF instance.

Parameters

connected | **static** }

Specifies whether to redistribute connected routes or static routes.

dst-protocol **bgp**

Specifies BGP as the destination protocol for redistribution.

address-family { **ipv4** | **ipv6** }

Specifies whether to redistribute IPv4 routes or IPv6 routes.

Modes

VRF table connections configuration (conf-vrf-*name*-table-connections)

Usage Guidelines

This command configures the routing of redistribution between pairs of protocols for specified address families within a VRF instance. You use this command to add pairs of protocols and the address family (IPv4 or IPv6) that applies to each pair.

BGP is the only supported destination protocol in this release.

Use the **no** form of this command to clear all table connections configurations for the specified VRF instance. If the VM is already deployed, this command attaches the network interfaces to the VM. If VM is not deployed, the network interfaces are attached at its next deployment.

Examples

The following example enables table connections configurations for a VRF instance named red and configures a connections table with four pairs of protocols and the address family (either IPv4 or IPv6) that applies to each pair:

```
device# configure terminal
device(config)# vrf red
device(config-vrf-red)# table-connections
device(config-vrf-red-table-connections)# src-protocol connected dst-protocol bgp
addressfamily ipv6
device(config-vrf-red-table-connections)# src-protocol connected dst-protocol bgp
addressfamily ipv4
device(config-vrf-red-table-connections)# src-protocol static dst-protocol bgp
addressfamily ipv4
device(config-vrf-red-table-connections)# src-protocol static dst-protocol bgp
addressfamily ipv6
device(config-vrf-red-table-connections)#
```

The following example displays the configuration of a VRF instance named red that is running currently on the device. In this example, the instance is configured with a connections table containing several pairs of protocols and the corresponding address family for each pair:

```
device# show running-config vrf

vrf red
table-connections
src-protocol connected dst-protocol bgp address-family ipv6
src-protocol connected dst-protocol bgp address-family ipv4
src-protocol static dst-protocol bgp address-family ipv4
src-protocol static dst-protocol bgp address-family ipv6
device#
```

stale-route-time (BGP graceful restart configuration)

Use the **stale-route-time** command to set the stale route time for BGP graceful restart for neighbors in the BGP routing process within a Virtual Routing and Forwarding (VRF) instance.

Syntax

stale-route-time *seconds*

no stale-route-time *seconds*

Parameters

seconds

Amount of time that each neighbor waits to receive messages from a restarting neighbor before declaring it down. Use an integer from 1 to 3600 seconds. This is the maximum amount of time that the neighbor or neighbors retain the stale paths of a gracefully restarted peer. The stale route time must be less than the restart time (for details, see [restart-time \(BGP graceful restart configuration\)](#) on page 902).

Modes

VRF BGP graceful restart configuration (config-vrf-bgp-gr)

Usage Guidelines

Graceful restart allows a device undergoing a restart to inform its adjacent neighbors of its condition. This command specifies the maximum time that stale routes are kept during a restart. This command lets you set the length of time that the device waits to receive messages from restarting neighbors before declaring them down.

To reset the graceful restart stale route time to the default value, use the **no** form of this command.

Graceful restart mode (that is, a restart time combined with a stale route time) and helper-only mode are mutually exclusive. If graceful restart is enabled, you must use the **no graceful-restart** command to disable it (see [graceful-restart \(BGP configuration\)](#) on page 626) before enabling helper-only mode. Conversely, if helper-only mode is enabled, you must disable it before enabling graceful restart. For details, see [helper-only \(BGP graceful restart configuration\)](#) on page 634.

Examples

The following example configures a routing process within a VRF instance named violet. This example enables graceful restart on its neighbors and configures a restart time and a stale route time of 100 seconds and 500 seconds respectively:

```
device# configure terminal
device(config)# vrf violet
device(config-vrf-violet)# router bgp
device(config-vrf-bgp)# graceful-restart
device(config-vrf-bgp-gr)# restart-time 100
device(config-vrf-bgp-gr)# stale-route-time 500
device(config-vrf-bgp-gr)#
```

The following example configures a routing process within a VRF instance named violet. This example enables graceful restart on its neighbors and configures a restart time and a stale route time of 100 seconds and 500 seconds respectively:

```
device# show running-config vrf violet

vrf violet
  router bgp
    graceful-restart
      restart-time 100
      stale-route-time 500
!
device#
```

statement (routing policy configuration)

Use the **statement** command to create a routing policy statement and enter Routing policy statement configuration (config-route-policy-policy-name-stmt-number) mode.

Syntax

```
statement sequence-number  
no statement sequence-number
```

Command Default

The specified routing policy contains no statements.

Parameters

sequence-number

Unique number that determines the routing policy statement's position in the order of evaluation (lowest first). Use an integer from 1 to 65535.

Modes

Routing policy map configuration (config-route-policy-policy-name)

Usage Guidelines

This command creates a named routing policy statement and enters the mode for adding conditions and actions to that routing policy statement. Each statement in a routing policy can have only one BGP actions container. This container can contain multiple actions.

You can create multiple routing policy statements within a routing policy.

To remove a routing policy statement from a routing policy, use the **no** form of this command.

Examples

The following example creates a routing policy named map1 that contains a policy statement with sequence number 10:

```
device# configure terminal  
device(config)# route-policy  
device(config-route-policy)# policy map1  
device(config-route-policy-policy-map1)# statement 10  
device(config-route-policy-policy-map1-stmt-10)#
```

The following example displays the routing policy configuration that is running currently on the device. In this example, a routing policy named map1 contains a policy

statement with sequence number 10 and a policy statement with sequence number 20:

```
device# show running-config route-policy

route-policy
  policy map1
    statement 10
      conditions
        bgp-conditions
          match-community-set comm1
    statement 20
      conditions
        bgp-conditions
          match-community-set comm2
device#
```

static-mac-address

Use the **static-mac-address** command to configure a static MAC address entry.

Syntax

```
static-mac-address mac-prefix { ethernet interface-name | port-channel
    interface-name } { untagged | vlan vlan-name [ inner-vlan inner-vlan-
    name ] }

static-mac-address tunnel interface-name

no static-mac-address mac-prefix
```

Command Default

No default static MAC address exists under the bridge domain.

Parameters

mac-prefix

Static MAC address that points to the interface. Use *xx:xx:xx:xx:xx:xx* notation (six groups of two hexadecimal digits separated by colons). For example, 01:23:45:67:89:ab.

ethernet *interface-name*

Specifies the outgoing interface type as Ethernet. Use the *slot/port* format to specify an interface name. For example, 0/1.

port-channel *interface-name*

Specifies the outgoing interface type as port channel. Use an integer from 1 to 255 for the interface name.

untagged

Specifies that the interface is untagged.

vlan *vlan-name*

ID of the VLAN associated with the static MAC address. Use an integer from 1 to 4094.

inner-vlan *vlan-name*

Uses double tagging. Use an integer from 1 to 4094.

tunnel *interface-name*

Specifies the outgoing interface as a tunnel interface. Use an integer from 1 to 100000.

Modes

Bridge domain configuration (*config-bd-name*)

Bridge domain VLAN configuration (config-vlan-bd-*name*)

Usage Guidelines

Multicast static MAC addresses are not allowed.

To disable a static MAC address, use the **no** form of this command.

Examples

The following example configures six static MAC addresses on bridge domain 1:

```
device# configure terminal
device(config)# bridge-domain 1
device(config-bd-1)# static-mac-address 40:88:2f:f9:c0:03 ethernet 0/1 vlan 1
device(config-bd-1)# static-mac-address f0:64:26:f5:c8:03 ethernet 0/2 vlan 401
device(config-bd-1)# static-mac-address f0:64:26:f5:c8:03 port-channel 2 untagged
device(config-bd-1)# static-mac-address f0:64:26:f5:c8:05 port-channel 2 vlan 10
device(config-bd-1)# static-mac-address 02:e0:52:11:11:11 ethernet 0/2 untagged
device(config-bd-1)# static-mac-address 02:e0:52:11:11:11 ethernet 0/4 vlan 10
device(config-bd-1)#
```

The following example shows the bridge domain configuration on the device:

```
device# show running-config bridge-domain

bridge-domain 10 mode vlan
  vlan-id 1
  static-mac-address 40:88:2f:f9:c0:03 ethernet 0/1 untagged

bridge-domain 1
  static-mac-address 40:88:2f:f9:c0:03 ethernet 0/1 vlan 1
  static-mac-address f0:64:26:f5:c8:03 ethernet 0/2 vlan 401
  static-mac-address f0:64:26:f5:c8:03 port-channel 2 untagged
  static-mac-address f0:64:26:f5:c8:05 port-channel 2 vlan 10
  static-mac-address 02:e0:52:11:11:11 ethernet 0/2 untagged
  static-mac-address 02:e0:52:11:11:11 ethernet 0/4 vlan 10
device#
```

The following example shows the MAC address table for all bridge domains on a device:

```
device# show mac-address-table bridge-domain all

Bridge-Domain:1
-----
Total number of Mac Entries: 1
Hardware Status Codes - #:Failed
Mac-Address          Type          Interface
-----
00:10:00:00:00:01    Static        ethernet 0/3

Bridge-Domain:10
-----
Total number of Mac Entries: 1
Hardware Status Codes - #:Failed
Mac-Address          Type          Interface
-----
00:10:00:00:00:02    Static        ethernet 0/4.10

vm1# show mac-address-table bridge-domain 10 static
Total number of Mac Entries: 1
```



```
Hardware Status Codes - #:Failed
Mac-Address          Type          Interface
-----
00:10:00:00:00:02    Static      ethernet 0/4.10

vm1# show mac-address-table bridge-domain 10 00:10:00:00:00:02
'*' denotes best route-source
Mac-Address          Type          Interface          Last
Change              Seq No      Hardware Status
-----
*00:10:00:00:00:02    Static      ethernet 0/4.10
48s                   0
device#
```

static-route

Use the **static-route** command to establish a static route for a VPN routing and forwarding (VRF) instance.

Syntax

```
static-route prefix [ set-tag tag-value ] next-hop-address [ { ethernet
    interface-name | port-channel interface-name | ve interface-name } ]
    [ admin-distance distance-value ] [ metric metric-value ] [ no-
    recurse ] [ description description-string ]

static-route prefix [ set-tag tag-value ] [ { ethernet interface-name |
    port-channel interface-name | ve interface-name | null 0 } ] [ admin-
    distance distance-value ] [ metric metric-value ] [ description
    description-string ]

no static-route prefix

no static-route prefix next-hop-address [ { ethernet interface-name |
    port-channel interface-name | ve interface-name } ]
```

Command Default

No static routes exist for the named VRF instance.

Parameters

prefix

Prefix for the destination. Use IPv4 or IPv6 CIDR notation (*A.B.C.D/M* or *X::X/M* respectively) where *M* is the prefix length.

The prefix length can be in the range 0–32 for IPv4 addresses and 0–128 for IPv6 addresses. For example, use a destination IPv4 address of 10.0.0.0 and a prefix length of 8 to route addresses in the range 10.0.0.0 to 10.255.255.255. Also, for example, use an IPv6 address of 2001::100 and a prefix length of 64 to route addresses in the range of 2001:0000:0000:0000:0000:0000:0000:0000 to 2001:0000:0000:0000:ffff:ffff:ffff:ffff.

set-tag

tag-value

(Optional) Label (tag) value for the prefix. This is the route to use for route filtering with a route map. You must specify an integer from 0 to 4294967295. The default is 0.

next-hop-address

Address of the next hop (the forwarding router that can be used to reach that network). You can specify an IPv4 or IPv6 address. You cannot specify a broadcast, multicast, class E, or loopback IP address.

ethernet

interface-name

(Optional) Specifies the outgoing interface type as Ethernet.

port-channel

interface-name

(Optional) Specifies the outgoing interface type as port channel.

ve

interface-name

(Optional) Specifies the outgoing interface type as Virtual Ethernet (VE).

null 0

(Optional) Sends all traffic that matches the prefix to the null 0 outgoing interface where it is then dropped.

admin-distance

distance-value

(Optional) Administrative distance for the next hop. This is the administrative distance of the route. You must specify an integer from 1 to 254. The default is 1. When comparing otherwise equal routes to a destination, lower administrative distances will override higher ones.

metric

metric-value

(Optional) Metric for the next hop. You must specify an integer from 1 to 16. The default is 1. When more than two static routes exist with the same destination network, the route with the lower metric is used.

no-recurse

(Optional) Disables recursive resolution of the next hop.

description

description-string

(Optional) Description of the prefix. You can use 1 to 63 characters.

Modes

VRF configuration (config-vrf-vrf-name)

Usage Guidelines

A link-local IPv6 next hop must be configured with an outgoing interface.

Multiple next hops for the same prefix are not allowed for null 0 or interface-based next hops.

The tag and description are properties of the prefix and hence overwrite previous values even if configured with different next hops.

A prefix with an interface (Ethernet, port channel, or VE) next hop is considered as directly connected. Traffic is forwarded based on address resolution protocol (ARP) or neighbor discovery (ND) success for each packet's destination address.

To configure a static route with Bidirectional Forwarding Detection (BFD), see [static-route enable-bfd](#) on page 1127.

To disable a static route for a VPN VRF instance, use the **no** form of this command. **no static-route** deletes the entire prefix (all next hops). **no static-route prefix nexthop-address** next removes a single next hop from the prefix.

Examples

The following example configures a simple IPv4 static route:

```
device# configure terminal
device(config)# vrf default-vrf
device(config-vrf-default-vrf)# static-route 1.1.1.0/24 2.2.2.2
device(config-vrf-default-vrf)#
```

The following example configures a simple IPv6 static route:

```
device# configure terminal
device(config)# vrf default-vrf
device(config-vrf-default-vrf)# static-route 10:187::/64 55::55
device(config-vrf-default-vrf)#
```

The following example configures an IPv4 static route with an interface next hop (and therefore directly connected):

```
device# configure terminal
device(config)# vrf default-vrf
device(config-vrf-default-vrf)# static-route 1.1.4.0/24 set-tag 9999 interface ethernet
0/1 description This is a connected static-route
device(config-vrf-default-vrf)#
```

The following example configures an IPv6 static route with an interface next hop (and therefore directly connected):

```
device# configure terminal
device(config)# vrf default-vrf
device(config-vrf-default-vrf)# static-route 10:94::/64 set-tag 9999 interface ethernet
0/1 description This is a connected static-route
device(config-vrf-default-vrf)#
```

The following example configures an IPv4 static route with a tag value of 100 to the null 1 output interface (which drops all packets that are sent to the destination):

```
device# configure terminal
device(config)# vrf default-vrf
device(config-vrf-default-vrf)# static-route 1.1.5.0/24 set-tag 100 null 0 description
This is a drop route
device(config-vrf-default-vrf)#
```

The following example configures an IPv6 static route with a tag value of 100 to the null 1 output interface (which drops all packets that are sent to the destination):

```
device# configure terminal
device(config)# vrf default-vrf
```

```
device(config-vrf-default-vrf)# static-route 10:1::/64 set-tag 100 null 0 description
This is a drop route
device(config-vrf-default-vrf)#
```

The following example configures an IPv4 static route with a tag value of 9999 and a next-hop address (with no recursive resolution) of 5.5.5.5. This example sets an administrative distance as well as a metric value for the next hop to help decide route preference:

```
device(config-vrf-default-vrf)# static-route 10.1.77.0/24 set-tag 9999 5.5.5.5 admin-
distance 5 metric 10 no-recurse description All parameters
device(config-vrf-default-vrf)#
```

The following example configures an IPv6 static route with a tag value of 9999 and a next-hop address (with no recursive resolution) of 5.5.5.5. This example sets an administrative distance as well as a metric value for the next hop to help decide route preference:

```
device# configure terminal
device(config)# vrf default-vrf
device(config-vrf-default-vrf)# static-route 10.1.77.0/24 set-tag 9999 5.5.5.5 admin-
distance 5 metric 10 no-recurse description All parameters
device(config-vrf-default-vrf)#
```

The following example configures an IPv6 static route with a tag value of 9999 and a next-hop address (with no recursive resolution) of 5.5.5.5. This example sets an administrative distance as well as a metric value for the next hop to help determine route preference.

In this example, the next hop uses a link-local address (instead of a global unicast address). It also specifies an output interface, which is required for IPv6 link-local addresses:

```
device# configure terminal
device(config)# vrf default-vrf
device(config-vrf-default-vrf)# static-route 10:d7::/64 set-tag 9999 fe80::29 interface
port-channel 25 admin-distance 5 metric 10 Description Link-local nexthop
device(config-vrf-default-vrf)#
```

The following example shows the output from a **show running-config** command that displays the currently-running static routes for a VRF instance named default-vrf:

```
device# show running-config vrf

vrf mgmt-vrf
  address-family ipv4 unicast
  address-family ipv6 unicast
vrf default-vrf
  address-family ipv4 unicast
  address-family ipv6 unicast
  static-route 1.1.1.0/24 2.2.2.2
  static-route 10:94::/64 set-tag 9999 interface ethernet 0/1 description this is a
connected static route
  static-route 10:101::/64 set-tag 9999 55::55 admin-distance 5 metric 10 no-recurse
description all parameters
  static-route 10:187::/64 55::55
  static-route 10.1.77.0/24 set-tag 9999 5.5.5.5 admin-distance 5 metric 10 no-recurse
description all parameters
  static-route 10:1::/64 set-tag 100 null 0 description this is a drop route
  static-route 1.1.4.0/24 set-tag 9999 interface ethernet 0/1 description this is a
```

```
connected static route
  static-route 1.1.5.0/24 set-tag 100 null 0 description this is drop route
  static-route 10:d7::/64 set-tag 9999 fe80::29 interface port-channel 25 admin-distance
5 metric 10 description link-local nexthop
device#
```

static-route enable-bfd

Use the **static-route enable-bfd** command to establish a BFD-enabled static route for a VPN routing and forwarding (VRF) instance.

Syntax

```
static-route prefix [ set-tag tag-value ] next-hop-address [ { ethernet
    interface-name | port-channel interface-name | ve interface-name } ]
    enable-bfd [ bfd-source-interface loopback interface-name ] [ profile
    profile-name ]

no static-route prefix

no static-route prefix next-hop-address [ { ethernet interface-name |
    port-channel interface-name | ve interface-name } ]
```

Command Default

No BFD-enabled static routes exist for the named VRF instance.

Parameters

prefix

Prefix for the destination. Use IPv4 or IPv6 CIDR notation (*A.B.C.D/M* or *X::X/M* respectively) where *M* is the prefix length.

The prefix length can be in the range 0–32 for IPv4 addresses and 0–128 for IPv6 addresses. For example, use a destination IPv4 address of 10.0.0.0 and a prefix length of 8 to route addresses in the range 10.0.0.0 to 10.255.255.255. Also, for example, use an IPv6 address of 2001::100 and a prefix length of 64 to route addresses in the range of 2001:0000:0000:0000:0000:0000:0000:0000 to 2001:0000:0000:0000:ffff:ffff:ffff:ffff.

set-tag

tag-value

(Optional) Label (tag) value for the prefix. This is the route to use for route filtering with a route map. You must specify an integer from 0 to 4294967295. The default is 0.

next-hop-address

Address of the next hop (the forwarding router that can be used to reach that network). You can specify an IPv4 or IPv6 address. You cannot specify a broadcast, multicast, class E, or loopback IP address.

ethernet

interface-name

(Optional) Specifies the outgoing interface type as Ethernet.

port-channel

interface-name

(Optional) Specifies the outgoing interface type as port channel.

ve

interface-name

(Optional) Specifies the outgoing interface type as Virtual Ethernet (VE).

bfd-source-interface loopback *interface-name*

(Optional) Source loopback interface for multihop BFD sessions. Use an integer from 1 to 1024 as the loopback interface name.

profile

profile-name

(Optional) Name of the BFD profile to use. This name is a combination of 1 to 64 numerals, underscores, hyphens, periods, and uppercase or lowercase letters that starts with a letter or underscore.

Modes

VRF configuration (config-vrf-*vrf-name*)

Usage Guidelines

Use this command to enable association of a static route with a static BFD configuration in order to monitor static route reachability using the configured BFD session.

The BFD session with destination and source IP addresses is established if its next hop IP address is resolved.

If multiple BFD-enabled static routes share a next hop IP address, then the BFD session will remain up until all BFD-enabled static routes are deleted.

If a BFD-enabled static routes session is up, the route is added to the HW table.

If a BFD-enabled static routes session goes down, the route is deleted from the HW table.

If multiple BFD-enabled static routes share a next hop IP address, you can have multiple profile names and will be accepted but the best profile is applied based on the Tx and Rx time intervals. The profile with the shortest intervals will be applied.

The destination and source IP addresses must be unicast addresses of the same address family.

A link-local IPv6 next hop must be configured with an outgoing interface.

Multiple next hops for the same prefix are not allowed for interface-based next hops.

The tag is a property of the prefix and hence overwrites previous values even if configured with different next hops.

A prefix with an interface (Ethernet, port channel, or VE) next hop is considered as directly connected. Traffic is forwarded based on address resolution protocol (ARP) or neighbor discovery (ND) success for each packet's destination address.

To disable a BFD-enabled static route for a VPN VRF instance, use the **no** form of this command. **no static-route** deletes the entire prefix (all next hops). **no static-route prefix nexthop-address** removes a single next hop from the prefix.

Examples

The following example configures a simple BFD-enabled IPv4 static route:

```
device# configure terminal
device(config)# vrf vrf1
device(config-vrf-vrf1)# static-route 10.1.1.0/24 2.2.2.3 enable-bfd
device(config-vrf-vrf1)#
```

The following example configures a simple BFD-enabled IPv4 static route that uses a BFD profile named xyz:

```
device# configure terminal
device(config)# vrf vrf1
device(config-vrf-vrf1)# static-route 10.1.1.0/24 2.2.2.3 enable-bfd profile xyz
device(config-vrf-vrf1)#
```

The following example configures a BFD-enabled IPv4 static route that uses loopback 1 as the BFD source interface and uses a BFD profile named xyz:

```
device# configure terminal
device(config)# vrf vrf1
device(config-vrf-vrf1)# static-route 10.1.1.0/24 2.2.2.2 enable-bfd bfd-source-interface
loopback 1 profile xyz
device(config-vrf-vrf1)#
```

The following example configures a BFD-enabled IPv6 static route that uses port channel interface 1. The prefix is configured with an interface next hop (and therefore directly connected).

```
device# configure terminal
device(config)# vrf vrf1
device(config-vrf-vrf1)# static-route 1001::2/128 fe80::1 interface port-channel 1 enable-
bfd
device(config-vrf-vrf1)#
```

Although not provided in this example, you can specify a BFD profile under the interface port-channel 1 that would be applied to the BFD session.

The following example configures a BFD-enabled IPv6 static route that uses port-channel interface 1 and a BFD profile named abc. The prefix is configured with an interface next hop (and therefore directly connected):

```
device# configure terminal
device(config)# vrf vrf1
device(config-vrf-vrf1)# static-route 1001::2/128 fe80::1 interface port-channel 1 enable-
bfd profile abc
device(config-vrf-vrf1)#
```

The following example creates a BFD profile named customer1. The profile is configured under source interface ve 100:

```
device# configure terminal
device(config-bfd)# profile customer1
device(config-bfd-profile-customer1)# interval 400 min-rx 400 multiplier 3
device(config-bfd-profile-customer1)# exit
device(config-bfd)# member ve 100 profile customer1
```

The following configuration is a BFD-enabled IPv4 static route in which you did not provide the profile while enabling BFD. In this scenario, the system will pick up the user configured profile from the source interface if configured. In this example, no profile is specified, so the system will pick up the profile from ve 100 (that is, customer1):

```
device# configure terminal
device(config)# vrf vrf1
device(config-vrf-vrf1)# static-route 110.1.1.0/24 10.1.1.2 enable-bfd
device(config-vrf-vrf1)#
```

Similarly, in the following configuration, the system will pick up profile customer2 or profile customer1 based on smaller intervals:

```
device# configure terminal
device(config)# vrf vrf1
device(config-vrf-vrf1)# static-route 110.1.1.0/24 10.1.1.2 enable-bfd profile customer2
device(config-vrf-vrf1)#
```

The following example shows the output from a **show running-config** command that displays the currently-running static routes for a VRF instance named vrf1. Three of the static routes are BFD-enabled:

```
device# show running-config vrf

vrf vrf1
  static-route 10.1.2.0/24 interface ethernet 0/1
  static-route 1001::2/128 fe80::1 interface port-channel 1 enable-bfd profile abc
  static-route 11.1.0.0/24 interface ethernet 0/1.3
  static-route 9100::/64 interface ve 1000
  static-route 10.1.0.0/24 2.2.2.2 enable-bfd
  static-route 10.1.1.0/24 2.2.2.2
  static-route 10.1.1.0/24 2.2.2.3 enable-bfd

device#
```

statistics (bridge domain configuration)

Use the **statistics** command to enable ingress and egress statistics on a bridge domain.

Syntax

```
statistics  
no statistics
```

Command Default

Statistics are disabled on the specified bridge domain.

Parameters

This command has no keywords or variables.

Modes

Bridge domain configuration (config-bd-*name*)

Usage Guidelines

Use the no form of this command to disable statistics on a bridge domain.

Examples

The following example enables statistics on a bridge domain named 200:

```
device# configure terminal  
device(config)# bridge-domain 200  
device(config-bd-200)# statistics  
device(config-bd-200)#
```

The following example displays the configuration of a specific bridge domain. In this example, the **statistics** command enables statistics on the bridge domain:

```
device# show running-config bridge-domain 200  
  
bridge-domain 200  
  member ethernet 0/1:1 untagged  
  member ethernet 0/1:2 untagged  
  statistics  
device#
```

subinterface (tagged)

Use the **subinterface (tagged)** command to create a single tagged subinterface in the specified interface to accept the tagged traffic bound to the default bridge domain and VLAN bridge domain modes.

Syntax

subinterface vlan *vlan-id*

no subinterface vlan *vlan-id*

Command Default

No single tagged subinterfaces are configured on the device.

Parameters

vlan *vlan-id*

Specifies the tagged VLAN subinterface. Valid range is 1-4094.

Modes

Subinterface configuration (config-subif-*name*)

Usage Guidelines

To remove a single tagged VLAN subinterface from the specified interface, use the no form of this command.

Examples

The following example creates the tagged VLAN subinterface:

```
device# configure terminal
device(config)# interface ethernet 0/2
device(config-eth-0/2)# subinterface vlan 100
device(config-subif-eth-0/2.100)#

device(config-subif-eth-0/2.100)# do show running-config

interface ethernet 0/2
  subinterface vlan 100
device(config-subif-eth-0/2.100)#
```

subinterface (untagged strict)

Use the **subinterface untagged** command to create an untagged strict subinterface in the specified interface to accept the untagged traffic bound to the default bridge domain and VLAN bridge domain modes.

Syntax

```
subinterface untagged  
no subinterface untagged
```

Command Default

No untagged strict subinterfaces are configured on the device.

Parameters

untagged
Specifies the untagged strict subinterface.

Modes

Subinterface configuration (config-subif-*name*)

Usage Guidelines

An untagged strict subinterface matches only untagged packets received on the port and sends packets without a VLAN tag.



Note

Untagged strict subinterfaces are not supported on the Extreme 8730 platform. All other platforms are supported.

To remove an untagged strict subinterface from the specified interface, use the no form of this command.

Examples

The following example creates the untagged strict VLAN subinterface:

```
device(config)# interface ethernet 0/2  
device(config-eth-0/2)# subinterface untagged  
  
device(config-eth-0/2)# do show running-config  
  
interface ethernet 0/2  
    subinterface untagged  
device(config-eth-0/2)#
```

subinterface vlan (untagged)

Use the **subinterface untagged vlan** command to create an untagged subinterface in the specified interface to accept the untagged and tagged traffic that is bound to the VLAN bridge domain.

Syntax

```
subinterface untagged vlan vlan-id  
no subinterface untagged vlan vlan-id
```

Command Default

No untagged subinterfaces to accept untagged and tagged traffic that is bound to the VLAN bridge domain are configured on the device.

Parameters

untagged

Specifies the untagged subinterface.

vlan *vlan-id*

Specifies the untagged VLAN ID.

Modes

Subinterface configuration (config-subif-*name*)

Usage Guidelines

An untagged subinterface matches untagged packets, associates them with a specified VLAN, and sends packets without a VLAN tag.

This command supports bridge domains only in VLAN mode (not default mode). To configure a bridge domain in VLAN mode, see [bridge-domain](#) on page 469.

Default mode bridge domains support only untagged *strict* subinterfaces. For details, see [subinterface \(untagged strict\)](#) on page 1133.

To remove an untagged subinterface from the specified interface, use the no form of this command.

Examples

The following example creates the untagged VLAN subinterface:

```
device(config)# interface ethernet 0/2  
device(config-eth-0/2)# subinterface untagged vlan 100  
  
device(config-eth-0/2)# do show running-config
```

```
interface ethernet 0/2
  subinterface untagged vlan 100
device(config-eth-0/2)#
```

suppress-arp

Use the **suppress-arp** command to disable address resolution protocol (ARP) on a bridge domain.

Syntax

```
suppress-arp  
no suppress-arp
```

Parameters

This command has no keywords or variables.

Modes

Bridge domain VLAN ARP proxy configuration (config-vlan-bd-*name*-arp-proxy)

Usage Guidelines

ARP establishes correspondences between pairs of network-layer addresses (Layer 3) and LAN hardware addresses (Layer 2 MAC addresses). But in some cases, you might need to suppress ARP to improve network security or performance. Note that suppressing ARP can cause connectivity issues for some devices on the network.

Examples

The following example suppresses ARP on a bridge domain with 100 as its ID:

```
device# configure terminal  
device(config)# bridge-domain 100 mode vlan  
device(config-vlan-bd-100)# arp-proxy  
device(config-vlan-bd-100-arp-proxy)# suppress-arp  
device(config-vlan-bd-100-arp-proxy)#
```

The following example displays the configuration for bridge domain 100 that is running currently on the device. In this example, ARP is suppressed on the bridge domain:

```
device# show running-config bridge-domain 100  
  
bridge-domain 100 mode vlan  
    arp-proxy  
    suppress-arp  
    !  
device#
```


suppress-nd

Use the **suppress-nd** command to disable neighbor discovery protocol (NDP) on a bridge domain.

Syntax

```
suppress-nd  
no suppress-nd
```

Parameters

This command has no keywords or variables.

Modes

Bridge domain VLAN neighbor discovery proxy configuration (config-vlan-bd-*name*-nd-proxy)

Usage Guidelines

NDP establishes correspondences between pairs of network-layer addresses (Layer 3) and LAN hardware addresses (Layer 2 MAC addresses). But in some cases, you might need to suppress NDP to improve network security or performance. Note that suppressing NDP can cause connectivity issues for some devices on the network.

Examples

The following example disables NDP on a bridge domain with 100 as its ID:

```
device# configure terminal  
device(config)# bridge-domain 100 mode vlan  
device(config-vlan-bd-100)# nd-proxy  
device(config-vlan-bd-100-nd-proxy)# suppress-nd  
device(config-vlan-bd-100-nd-proxy)#
```

The following example displays the configuration for bridge domain 100 that is running currently on the device. In this example, NDP is disabled on the bridge domain:

```
device# show running-config bridge-domain 100  
  
bridge-domain 100 mode vlan  
    nd-proxy  
    suppress-nd  
    !  
device#
```

suppress-tlv

Use the **suppress-tlv** command to suppress transmission of LLDP type-length-value (TLV) advertisements globally or from a specific interface.

Syntax

```
suppress-tlv { [ management-address ] [ port-description ] [ system-  
                  description ] [ system-name ] }  
  
no suppress-tlv [ management-address ] [ port-description ] [ system-  
                  description ] [ system-name ]
```

Command Default

LLDP TLV suppression is disabled globally or on the specified interface.

Parameters

management-address

Suppresses TLV advertisement of the TLV management address. This address allows easier management and troubleshooting of LLDP communications.

port-description

Suppresses TLV advertisement of the LLDP port description.

system-description

Suppresses TLV advertisement of the LLDP system description.

system-name

Suppresses TLV advertisement of the LLDP system name.

Modes

LLDP configuration (config-lldp) or Ethernet interface LLDP configuration (config-if-eth-*name*-lldp)

Usage Guidelines

Use this command to determine which optional TLV advertisements to suppress globally (meaning on all LLDP enabled interfaces) or from a specific LLDP enabled interface. These optional TLV advertisements are for the management address, port description, system description, and system name. If TLV suppression is enabled globally, all LLDP enabled interfaces stop advertising the configured optional TLVs from their LLDP frames.

Per interface configuration of this functionality allows only specific LLDP enabled interfaces to stop advertising TLVs from their frames. If both global and per interface functionality are configured for TLV advertisement suppression, the interface configuration takes precedence.

Valid TLV tokens need to be configured for this functionality to be disabled.

Use the no form of this command in the appropriate command mode to disable TLV suppression globally or on the specified interface.

Examples

The following example enables suppression globally of all optional TLV advertisements:

```
device# configure terminal
device(config)# lldp
device(config-lldp)# suppress-tlv management-address port-description system-name system-
description
device(config-lldp)#
```

The following example enables suppression of all optional TLV advertisements on Ethernet interface 0/1:

```
device# configure terminal
device(config)# interface ethernet 0/1
device(config-eth-0/1-lldp)# lldp
device(config-eth-0/1-lldp)# suppress-tlv management-address port-description system-name
system-description
device(config-eth-0/1-lldp)#
```

The following example displays the LLDP configuration that is running currently on the device. In this example, suppression is enabled globally of all optional TLV advertisements:

```
device# show running-config lldp

lldp
  suppress-tlv management-address port-description system-name system-description
device#
```

The following example displays the configuration for Ethernet interface 0/1 that is running currently on the device. In this example, suppression is enabled on the interface of all optional TLV advertisements:

```
device# show running-config interface ethernet 0/1

interface ethernet 0/1
  lldp
    suppress-tlv management-address port-description system-name system-description
device#
```

system maintenance-mode

Use the **system maintenance-mode** command to enter or exit maintenance mode on a running device.

Syntax

```
system maintenance-mode { enable | disable }
```

Command Default

The device is not presently in maintenance mode.

Parameters

enable

Enters maintenance mode.

disable

Exits maintenance mode. If the `%Error: Maintenance mode operation in progress` error message appears, wait a few moments and then retry the command. You can enter the **show system maintenance-mode** command to see whether the maintenance mode operation is complete.

Modes

EXEC (#)

Usage Guidelines

You use the maintenance mode feature when a device needs maintenance for reasons such as an image upgrade. When maintenance mode is active, all front panel ports are disabled to isolate the device from all connected devices on the network.

During maintenance mode, all physical ports will be in the "link down" state. The connected devices detect that the link is down and divert the traffic to a peer multichassis link aggregation (MLAG) node.

This command enables maintenance mode on a running device. This command disables all front panel ports immediately.

This command does not appear in the running configuration.

You can also configure the device to enter maintenance mode upon system reload (reboot). For details, see [enable-on-reload](#) on page 603.

Examples

The following example enters maintenance mode immediately on the device:

```
device# system maintenance-mode enable
device#
```

The following example displays the maintenance mode details of the device. In this example, Maintenance Mode : enabled indicates that maintenance mode is active on the device:

```
device# show system maintenance-mode

Enable on reload      : disabled
Maintenance Mode      : enabled
Maintenance Mode Status : Complete
device#
```

table-connections

Use the **table-connections** command to enter VRF table connections configuration (conf-vrf-*name*-table-connections) mode.

Syntax

table-connections

Parameters

This command has no keywords or variables.

Modes

VRF configuration (config-vrf-*vrf-name*)

Usage Guidelines

Use this command to enter the mode for configuring table connection settings for the source and destination protocols for route redistribution for a specified address family.

Examples

The following example enters VRF table connections configuration (conf-vrf-*name*-table-connections) mode:

```
device# configure terminal
device(config)# vrf red
device(config-vrf-red)# table-connections
device(config-vrf-red-table-connections)#
```

threshold-monitor

Use the **threshold-monitor** command to configure the high limit, low limit, and actions to perform when the thresholds for usage are exceeded.

Syntax

```
threshold-monitor { acl-ipv4-in | acl-ipv4-out | acl-ipv6-in | acl-ipv6-out | acl-mac-in | bfd-session | ecmp | host | lif | mac-table | nexthop | racl-ipv4-in | racl-ipv6-in | resilient-hashing | route | vxlan-tunnel } high-limit percent low-limit percent actions { all | none | raslog | snmp }
```

Command Default

Threshold monitoring of each of these resources is enabled with the default limits and actions.

Parameters

Global configuration (config)

Modes

```
{ acl-ipv4-in | acl-ipv4-out | acl-ipv6-in | acl-ipv6-out | acl-mac-in | bfd-session | ecmp | host | lif | mac-table | nexthop | racl-ipv4-in | racl-ipv6-in | resilient-hashing | route | vxlan-tunnel }
```

Resources for which threshold monitoring is needed.

high-limit *percent*

High threshold watermark percentage. Use an integer from 10 to 100. The default is 80 if this keyword is not specified.

low-limit *percent*

Low threshold watermark percentage. Use an integer from 10 to 100. The default is 80 if this keyword is not specified.

actions { **all** | **none** | **raslog** | **snmp** }

Actions to be taken on breaching low or high watermarks. You can specify RASlog generation, SNMP notifications, or both (all) actions. The default is the RASlog (raslog) action if this keyword is not specified.

Usage Guidelines

To disable threshold monitoring for a resource, use the **no** form of this command.

Examples

The following example configures threshold monitoring for ACL IPv4 ingress with RASlog generation:

```
device# configure terminal
device(config)# threshold-monitor acl-ipv4-in high-limit 95 low-limit 85 actions raslog
device(config)#
```

The following example configures threshold monitoring for VXLAN tunnels with SNMP notifications:

```
device# configure terminal
device(config)# threshold-monitor vxlan-tunnel high-limit 85 low-limit 75 actions snmp
device(config)#
```


threshold-monitor (CPU and memory)

Use the **threshold-monitor** command to configure the high limit, low limit, and actions to perform when the thresholds for CPU and memory usage are exceeded.

Syntax

```
threshold-monitor { cpu | memory } high-limit percent low-limit percent  
  actions { none | raslog | snmp | all } [ poll-interval seconds ]  
  [ poll-retry retries ]
```

Command Default

Threshold monitoring of CPU and memory resources is enabled with the default limits and actions.

Parameters

Global configuration (config)

Modes

cpu | **memory**

Resources for which threshold monitoring is needed.

high-limit *percent*

High threshold watermark percentage. Use an integer from 10 to 100. The default is 80 if this keyword is not specified.

low-limit *percent*

Low threshold watermark percentage. Use an integer from 10 to 100. The default is 80 if this keyword is not specified.

actions { **none** | **raslog** | **snmp** | **all** }

Actions to be taken on breaching low or high watermarks. You can specify RASlog generation, SNMP notifications, or both (all) actions. The default is the RASlog (raslog) action if this keyword is not specified.

poll-interval *seconds*

(Optional) Polling interval in seconds. Use an integer from 10 to 3600. The default is 10 if this keyword is not specified.

poll-retry *number*

Number of polling retries. Use an integer from 1 to 100. The default is 1 if this keyword is not specified.

Usage Guidelines

CPU and memory usage percentage can vary with the hardware platform and the running application services. This feature lets you configure the usage percentage of

resources to configure the time between polling intervals to monitor the resources and also configure the permitted number of polling retries. In a scaled configuration, you might experience a high CPU usage percentage of all cores as well as a higher memory usage percentage.

At every polling interval, the resources CPU and memory usage are read. When this threshold upper value is crossed, the actions are executed (such as generation of a RASlog containing diagnostic information for the resource for high usage). For example, in the RASlog:

```
{"Level":"warn","LogID":27006,"Topic":16,"Current usage  
Percent":89.11329021489607,"Msg":"Warning: cpu High Resource usage detected"}  
  
{"Level":"warn","LogID":27006,"Topic":16,"Current usage  
Percent":83.22778345250256,"Msg":"Warning: memory High Resource usage detected"}
```

To restore threshold monitoring for CPU and memory to the default values, use the **no** form of this command.

Examples

The following example configures threshold monitoring with a polling interval of 20 seconds and with RASlog generation:

```
device# configure terminal  
device(config)# threshold-monitor cpu high-limit 95 low-limit 85 actions raslog poll-  
interval 20  
device(config)#
```

The following example configures threshold monitoring with RASlog generation, SNMP notifications, and one polling retry. This example uses a polling interval of 120 seconds (the default value if the interval is not specified):

```
device# configure terminal  
device(config)# threshold-monitor memory high-limit 85 low-limit 75 actions all poll-  
retry 1  
device(config)#
```

timers (BGP peer group configuration)

Use the **timers** command to enter BGP peer group timer configuration mode for a BGP peer group within a Virtual Routing and Forwarding (VRF) instance.

Syntax

```
timers  
no timers
```

Command Default

The configurable timers for members of the specified BGP peer group use the following default values:

- Connect retry: 30 seconds
- Hold time: 180 seconds
- Keepalive interval: 60 seconds

Parameters

This command has no keywords or variables.

Modes

VRF BGP peer group configuration (config-vrf-bgp-pg)

Usage Guidelines

This command enters the mode for accessing the commands used to set custom values for the configurable timers for all members of the specified BGP peer group. These commands are **connect-retry**, **hold-time**, and **keepalive-interval**.

Use the **no** form of this command to reset all configurable timers to their default values for all members of the BGP peer group.

Examples

The following example configures a BGP peer group named `group1` under a VRF instance named `violet`. This example enters BGP peer group timer configuration mode for `group1`:

```
device# configure terminal  
device(config)# vrf violet  
device(config-vrf-violet)# router bgp  
device(config-vrf-bgp)# peer-group group1  
device(config-vrf-bgp-pg)# timers  
device(config-vrf-bgp-pg-timers)#
```

The following example displays the configuration of a VRF instance named violet that is running currently on the device after the **timers** command was entered “standalone” (that is, not followed by any commands available in the BGP peer group timer configuration mode). This output displays no timer settings, which indicates that the default timer settings are being used:

```
device# show running-config vrf violet

vrf violet
!
  router bgp
    peer-group group1
!
device#
```

timezone (IAH VM configuration)

Use the **timezone** command to configure the system time zone for an IAH third party virtual machine (TPVM).

Syntax

```
timezone region/city  
no timezone
```

Command Default

The system time zone is set to Etc/UTC for the TPVM.

Parameters

```
timezone region/city
```

Supported time zone. You must include the slash (/). For example, enter America/Anchorage or Africa/Algiers. Enter the **timezone** command without keywords or variables to see the available values.

Modes

IAH VM configuration (config-iah-vm-*name*)

Usage Guidelines

This command sets the guest OS timezone of the TPVM. The TPVM is a general server that resides on Extreme ONE OS devices. The guest OS that the TPVM provides is separate from Extreme ONE OS.

This command is allowed conditionally only for an integrated application hosting (IAH) VM named TPVM. Specifying this VM name extends the available IAH configurations to support TPVM.

Use the **no** form of this command to restore the timezone to the default setting. Note that configuration is not allowed when the VM is in the deployed state.

To display details about a specific VM, see [show iah vm](#) on page 1006.

Examples

The following example changes the guest OS timezone from the default value to Africa/Casablanca for the IAH TPVM on the device:

```
device# configure terminal  
device(config)# iah  
device(config-iah)# vm TPVM
```

```
device(config-iah-vm-TPVM)# timezone Africa/Casablanca
device(config-iah-vm-TPVM)#
```

The following example displays the IAH configuration that is running currently on the device. In this example, the guest OS timezone is changed from the default value to Africa/Casablanca for the IAH TPVM on the device:

```
device# show running-config iah

iah
vm TPVM
  hostname ganga
  timezone Africa/Casablanca
  ip
    ipv4-addr 2.2.2.4
    ipv4-gw 2.2.2.1
  ntp
    server 1.1.1.1
    server 3.3.3.3
    server time.example.com
    server 2001:db8::1
  interface management ipv4 2.2.2.2/24 gw 2.2.2.1
  trusted-peer ipv4 3.3.3.3 password $6$bcdef sudo-user extreme
  !
!
device#
```

traceroute

Use the **traceroute** command to trace the network path of packets as they are forwarded to a destination address.

Syntax

```
traceroute { ipv4-address | host-name | ipv6 [ ipv6-address | host-name ] } [ interface { ethernet interface-name | management | port-channel port-number | ve vlan-id } ] [ max-ttl value ] [ min-ttl value ] [ source source-addr ] [ timeout seconds ] [ vrf { mgmt-vrf | default-vrf | vrf-name } ]
```

Parameters

ipv4-address

IPv4 address of the destination device.

host-name

Hostname of the destination device.

ipv6-address

IPv6 address of the destination device.

ethernet

(Optional) Specifies a physical Ethernet interface.

slot

Valid slot number.

port

Valid port number.

management

(Optional) Specifies the management interface.

port-channel *port-number*

(Optional) Port channel (LAG) interface number.

ve *vlan-id*

(Optional) Virtual Ethernet (VE) interface VLAN ID.

max-ttl *value*

(Optional) Maximum Time to Live (TTL) value in a number of hops. The range 1 to 255. The default is 30.

min-ttl *value*

(Optional) Minimum TTL value in a number of hops. The range 1 to 255. The default is 1.

source *source-addr*

(Optional) IPv4 or IPv6 address of the source device.

timeout *seconds*

(Optional) Traceroute timeout value. The range 1 to 60. The default is 5.

vrf

(Optional) VRF on which to run a traceroute.

mgmt-vrf

Specifies the management VRF.

default-vrf

Specifies the default VRF.

vrf-name

User-created VRF name.

Modes

EXEC (#)

Examples

The following example executes an IPv4 traceroute:

```
device# traceroute 169.254.1.0

traceroute to 169.254.1.0 (169.254.1.0), 30 hops max, 60 byte packets
 1 169.254.1.0 (169.254.1.0)  0.075 ms  0.019 ms  0.017 ms
device#
```

The following example executes an IPv6 traceroute with minimum and maximum TTL values, a source IPv6 address, and a timeout value:

```
device# traceroute ipv6 fec0:60:69bc:92:218:8bff:fe40:1470
maxttl 128 minttl 30 source fec0:60:69bc:92:205:33ff:fe9e:3f20 timeout 3

traceroute to fec0:60:69bc:92:218:8bff:fe40:1470 (fec0:60:69bc:92:218:8bff:fe40:1470),
128 hops max, 80 byte packets 30  fec0:60:69bc:92:218:8bff:fe40:1470
(fec0:60:69bc:92:218:8bff:fe40:1470)  2.145 ms  2.118 ms  2.085 ms
device#
```


traffic-class cos (QoS remark map configuration)

Use the **traffic-class cos** command to define the traffic-class-to-CoS entries in a QoS map for remarking.

Syntax

```
traffic-class traffic-class-value cos cos-value  
no traffic-class traffic-class-value cos
```

Command Default

The specified QoS map contains no traffic-class-to-CoS entries for remarking.

Parameters

traffic-class-value

Traffic class value. Use an integer from 0 to 7.

cos-value

Class of Service (CoS) value. Use an integer from 0 to 7.

Modes

QoS remark map configuration (config-qos-remark-map-*name*)

Usage Guidelines

To remove a traffic-class-to-CoS entry from a QoS map for remarking, use the no form of this command.

Examples

The following example configures two CoS entries in a QoS map named egr_l2_map for remarking:

```
device# configure terminal  
device(config)# qos  
device(config-qos)# remark-map egr_l2_map  
device(config-qos-remark-map-egr_l2_map)# traffic-class 3 cos 1  
device(config-qos-remark-map-egr_l2_map)# traffic-class 4 cos 2  
device(config-qos-remark-map-egr_l2_map)#
```

The following example displays the QoS configuration that is running currently on the device. In this example, a QoS map named egr_l2_map for remarking is configured with two entries:

```
device# show running-config qos  
  
qos  
  remark-map egr_l2_map
```

```
traffic-class 3 cos 1
traffic-class 4 cos 2
device#
```

traffic-class dscp (QoS remark map configuration)

Use the **traffic-class dscp** command to define the traffic-class-to-DSCP entries in a QoS map for remarking.

Syntax

```
traffic-class traffic-class-value dscp dscp-value
```

```
no traffic-class traffic-class-value dscp
```

Command Default

The specified QoS map contains no traffic-class-to-DSCP entries for remarking.

Parameters

traffic-class-value

Traffic class value. Use an integer from 0 to 7.

dscp-value

DSCP value. Use an integer from 0 to 7.

Modes

QoS remark map configuration (config-qos-remark-map-*name*)

Usage Guidelines

To remove a traffic-class-to-DSCP entry from a QoS map for remarking, use the no form of this command.

Examples

The following example configures two DSCP entries in a QoS map named egr_l2_map for remarking:

```
device# configure terminal
device(config)# qos
device(config-qos)# remark-map egr_l2_map
device(config-qos-remark-map-egr_l2_map)# traffic-class 1 dscp 10
device(config-qos-remark-map-egr_l2_map)# traffic-class 2 dscp 12
device(config-qos-remark-map-egr_l2_map)#
```

The following example displays the QoS configuration that is running currently on the device. In this example, a QoS map named egr_l2_map for remarking is configured with two DSCP entries:

```
device# show running-config qos

qos
  remark-map egr_l2_map
```

```
traffic-class 1 dscp 10
traffic-class 2 dscp 12
device#
```

traffic-class-map (interface QoS input configuration)

Use the **traffic-class-map** command to attach a QoS map to an interface under the input or output directions.

Syntax

traffic-class-map *map-name*

no traffic-class-map

Command Default

No traffic class map is attached to the specified interface.

Parameters

map-name

Name of the traffic class map. Use an arbitrary sequence of up to 64 letters, numerals, underscores, hyphens, or periods that starts with a letter or an underscore. Do not use the "default" reserved keyword as the complete name.

Modes

Ethernet interface QoS input configuration (config-qos-if-eth-*name*-input)

Port channel interface QoS input configuration (config-qos-if-po-*name*-input)

Virtual Ethernet interface QoS input configuration (config-qos-if-ve-*name*-input)

Usage Guidelines

By default, traffic class mapping for L2 LIF is based on Class of Service (COS), and for L3 LIF, it is based on Differentiated Services Code Point (DSCP).

To remove the traffic class map from the interface, use the no form of this command.

Examples

The following example attaches a traffic class map named ing_l2_map in the input direction to subinterface 100 under Ethernet interface 0/1:

```
device# configure terminal
device(config)# qos
device(config-qos)# interface ethernet 0/1 sub-interface 100
device(config-qos-if-eth-0/1:1.100)# input
device(config-qos-if-eth-0/1:1.100-input)# traffic-class-map ing_l2_map
device(config-qos-if-eth-0/1:1.100-input)#
```

The following example displays the QoS configuration that is running currently on the device. In this example, a traffic class map named `ing_l2_map` is attached in the input direction to subinterface 100 under Ethernet interface 0/1:

```
device# show running-config qos

qos
  interface ethernet 0/1 sub-interface 100
    input
      traffic-class-map ing_l2_map
device#
```

traffic-class-map (QoS configuration)

Use the **traffic-class-map** command to configure a QoS map for packet classification on a device and enter QoS traffic class map configuration (config-qos-traffic-class-map-*name*) mode.

Syntax

traffic-class-map *map-name*

no traffic-class-map *map-name*

Command Default

The device has no traffic class map.

Parameters

map-name

Name of the traffic class map. Use an arbitrary sequence of up to 64 letters, numerals, underscores, hyphens, or periods that starts with a letter or an underscore. Do not use the "default" reserved keyword as the complete name.

Modes

QoS configuration (config-qos)

Usage Guidelines

Use this command to enter the mode for configuring packet classification based on packet DSCP or CoS values by mapping the values to internal traffic classes.

By default, traffic class mapping for L2 LIF is based on Class of Service (COS), and for L3 LIF, it is based on Differentiated Services Code Point (DSCP).

To remove the traffic class map from the device, use the no form of this command.

Examples

The following example configures a traffic class map named ing_l2_map and enters QoS traffic class map configuration (config-qos-traffic-class-map-*name*) mode:

```
device# configure terminal
device(config)# qos
device(config-qos)# traffic-class-map ing_l2_map
device(config-qos-traffic-class-map-ing_l2_map)#
```

The following example displays the QoS configuration that is running currently on the device. In this example, a traffic class map named `ing_l2_map` is configured:

```
device# show running-config qos  
  
qos  
  traffic-class-map ing_l2_map  
device#
```


transmit (LLDP configuration)

Use the **transmit** command to enable global LLDP packet transmit mode on the device.

Syntax

```
transmit  
no transmit
```

Command Default

Global LLDP packet transmit mode on the device is enabled.

Parameters

This command has no keywords or variables.

Modes

LLDP configuration (config-lldp)

Usage Guidelines

Use this command to enable the LLDP agent on the device to transmit information about the LLDP capabilities and the LLDP current status of the device to remote systems.

Use the no form of this command to disable global LLDP packet transmit mode on the device.

To enable global LLDP packet receive mode on the device, see [receive \(LLDP configuration\)](#) on page 891.

Examples

The following example enables global LLDP packet transmit mode:

```
device# configure terminal  
device(config)# lldp  
device(config-lldp)# transmit  
device(config-lldp)#
```

The following example displays the LLDP configuration that is running currently on the device. In this example, global LLDP packet transmit mode is enabled:

```
device# show running-config lldp  
  
lldp  
  transmit  
device#
```

trust (interface QoS input configuration)

Use the **trust** command to enable an interface to trust the QoS CoS or QoS DSCP markings on ingress (incoming) packets for traffic class mapping.

Syntax

```
trust { cos | dscp }  
no trust { cos | dscp }
```

Command Default

Trust is disabled on the specified interface.

Parameters

cos

Trusts the QoS Class of Service (CoS) markings.

dscp

Trusts the QoS Differentiated Services Code Point (DSCP) markings.

Modes

Ethernet interface QoS input configuration (config-qos-if-eth-*name*-input)

Ethernet subinterface QoS input configuration (config-qos-if-eth-*name:name*-input)

Port channel interface QoS input configuration (config-qos-if-po-*name*-input)

Virtual Ethernet interface QoS input configuration (config-qos-if-ve-*name*-input)

Usage Guidelines

Use this command to enable Trust DSCP or Trust CoS on layer 2 LIFs for traffic class mapping. If configured, CoS or DSCP is used to map the markings to internal traffic-class values.

Note that on layer 3 LIFs, mapping always uses DSCP (even if you have enabled Trust CoS).

DSCP and CoS are the two methods that are available in the operating system for marking ingress traffic for easier processing by the device. When configuring QoS, you can define whether to trust the existing DSCP marking on a packet or to use CoS. Trust DSCP tells the device that if a packet arrives on the interface that is marked with a DSCP value, to process it according to the marking. Without DSCP, the device ignores the marking and then identifies the type of traffic itself.

To restore the default behavior on the interface, use the no form of this command.

Examples

The following example enables Trust DSCP on subinterface 100 under Ethernet interface 0/1:

```
device# configure terminal
device(config)# qos
device(config-qos)# interface ethernet 0/1 subinterface vlan 100
device(config-qos-if-eth-0/1.100)# input
device(config-qos-if-eth-0/1.100-input)# trust dscp
device(config-qos-if-eth-0/1.100-input)#
```

The following example displays the QoS configuration that is running currently on the device. In this example, Trust DSCP is enabled on subinterface 100 under Ethernet interface 0/1:

```
device# show running-config qos

qos
  interface ethernet 0/1 subinterface 100
    input
      trust dscp
device#
```

trusted-peer (IAH VM configuration)

Use the **trusted-peer** command to configure the trusted peer TPVM for an IAH third party virtual machine (TPVM).

Syntax

```
trusted-peer { ipv4 ipv4-address | ipv6 ipv6-address } { password pass-string | password-hashed hashed-pass-string } [ sudo-user username ]  
no trusted-peer
```

Command Default

The TPVM has no trusted peer configured.

Parameters

ipv4-address

Peer TPVM IPv4 address. Use the *a.b.c.d* format.

ipv6-address

Peer TPVM IPv6 address. Use the *x::x::x::x* format.

password *pass-string*

Password for the sudo user in the peer TPVM. Use 1 to 64 characters. Omitting *pass-string* from the command prompts you to enter the password. This lets you obscure (asterisk-protect) the string instead of entering it in clear text.

password-hashed *hashed-pass-string*

Hashed password for the sudo user in the peer TPVM.

sudo-user *username*

(Optional) Sudo username in the peer TPVM. Use 1 to 64 characters. The default sudo username "extreme" is used if you do not specify this keyword and variable.

Modes

IAH VM configuration (config-iah-vm-*name*)

Usage Guidelines

This command configures a bidirectional passwordless SSH connection between the root user accounts of the TPVM on the device and a peer TPVM instance. The TPVM is a general server that resides on Extreme ONE OS devices. The guest OS that the TPVM provides is separate from Extreme ONE OS Switching.

The peer password is stored in encrypted format for any reconfiguration requirement.

Make sure that the trusted peer TPVM is reachable before you use this command.

This command is allowed conditionally only for an integrated application hosting (IAH) VM named TPVM. Specifying this VM name extends the available IAH configurations to support TPVM.

Use the **no** form of this command to remove the trusted peer from the TPVM. Note that configuration is not allowed when the VM is in the deployed state.

To display details about a specific VM, see [show iah vm](#) on page 1006.

Examples

The following example sets the trusted peer TPVM's IP address on the device. This example shows how to omit the password string from the command to let you enter it from a prompt in obscured (asterisk-protected) format:

```
device# configure terminal
device(config)# iah
device(config-iah)# vm TPVM
device(config-iah-vm-TPVM)# trusted-peer ipv4 3.3.3.3 password
peer password: *****
device(config-iah-vm-TPVM)#
```

The following example sets the trusted peer TPVM's IP address and sudo user login credentials on the device:

```
device# configure terminal
device(config)# iah
device(config-iah)# vm TPVM
device(config-iah-vm-TPVM)# trusted-peer ipv4 3.3.3.3 password $6$bcdef sudo-user extreme
device(config-iah-vm-TPVM)#
```

The following example displays the IAH configuration that is running currently on the device. In this example, the trusted peer TPVM's IP address and sudo user login credentials are configured on the device:

```
device# show running-config iah

iah
vm TPVM
  hostname ganga
  timezone Africa/Casablanca
  ip
    ipv4-addr 2.2.2.4
    ipv4-gw 2.2.2.1
  ntp
    server 1.1.1.1
    server 3.3.3.3
    server time.example.com
    server 2001:db8::1
  interface management ipv4 2.2.2.2/24 gw 2.2.2.1
  trusted-peer ipv4 3.3.3.3 password $6$bcdef sudo-user extreme
  !
!
device#
```

tunnel-mode

Use the **tunnel-mode** command to enter QoS NVO tunnel mode configuration (config-qos-nvo-*name*-tunnel-mode) mode.

Syntax

tunnel-mode

Command Default

Fabric QoS tunnel mode encapsulation and decapsulation are not configured on the device.

Parameters

This command has no keywords or variables.

Modes

QoS NVO configuration (config-qos-nvo-*name*)

Usage Guidelines

This command enters the mode for accessing the commands for configuring access QoS tunnel mode encapsulation and decapsulation on a device. Configuration is done at NVO level, which applies to all tunnels that are defined under it. A tunnel can be in pipe mode or uniform mode on both ingress (initiating) and egress (terminating) end points.

This command does not appear in the running configuration unless there are sub-mode parameters configured.

To restore all access QoS tunnel mode encapsulation and decapsulation configuration to the default values on the device, use the **no** form of this command.

Examples

The following example enters QoS NVO tunnel mode configuration (config-qos-nvo-*name*-tunnel-mode) mode and configures access QoS tunnel mode encapsulation and decapsulation on the device:

```
device# configure terminal
device(config)# qos
device(config-qos)# nvo mynvo
device(config-qos-nvo-mynvo)# tunnel-mode
device(config-qos-nvo-mynvo-tunnel-mode)# encapsulation pipe dscp 34
device(config-qos-nvo-mynvo-tunnel-mode)# decapsulation uniform
device(config-qos-nvo-mynvo-tunnel-mode)#
```

The following example shows the QoS configuration that is running currently on the device. In this example, a QoS node has an associated Network Virtualization Overlay (NVO) instance named mynvo with QoS tunnel mode encapsulation and decapsulation configured:

```
device# show running-config qos

qos
  nvo mynvo
    tunnel-mode
      encapsulation pipe dscp 34
      decapsulation uniform
    !
  !
!
device#
```

type (MLAG peer keepalive configuration)

Use the **type (MLAG peer keepalive configuration)** command to specify the multichassis link aggregation (MLAG) peer keepalive type.

Syntax

```
type { primary | secondary }  
no type
```

Command Default

No MLAG peer keepalive type is configured on the device.

Parameters

primary

Primary keepalive type.

secondary

Secondary keepalive type.

Modes

MLAG peer keepalive configuration (config-mlag-peer-keepalive-*name*)

Usage Guidelines

Two keepalive sessions are configured in an MLAG. The primary keepalive is used for Inter-Switch Link (ISL) connectivity, and the secondary keepalive is the management interface (by default).

You can use the **show mlag** command to display information about the MLAG configuration.

To remove the MLAG peer keepalive type, use the no form of this command.

Examples

The following example configures an MLAG peer named peer01 and also configures it to use a primary keepalive type:

```
device# configure terminal  
device(config)# mlag  
device(config-mlag)# peer peer01  
device(config-mlag-peer-peer01)# keepalive keepalive01  
device(config-mlag-peer-keepalive-keepalive01)# type primary  
device(config-mlag-peer-keepalive-keepalive01)#
```


The following example displays the MLAG configuration that is running currently on a device. In this example, MLAG is configured with a peer named peer01. The peer uses a secondary keepalive type:

```
device# show running-config mlag

mlag
!
  peer peer01
!
    keepalive keepalive01
      destination 12.2.2.2
      source-interface port-channel 4
      type secondary
!
device#
```

The following example displays the MLAG peer configuration on a device. In this example, MLAG is configured with a peer named mypeer1. The peer is configured with a keepalive named prim as type PRIMARY:

```
device# show mlag peer

Peer mypeer1
=====
Peer State           : UP
MCP State            : UP
Role                 : PRIMARY
Elected MAC         : e4:44:e5:00:94:00
Extend Bridge Count  : 361
Peer Exception       : None
  Keepalive prim
  -----
    Destination Address: 20.20.20.164
    State               : UP
    Type                : PRIMARY
    Address Type        : MANUAL
    Source Interface    : loopback 1024 ( 20.20.20.163 )
    First Rx            : 2025-06-27 12:11:27
    Last Rx             : 2025-06-27 12:50:59
    Keepalive Second
    -----
      Destination Address: 10.32.100.35
      State              : UP
      Type               : SECONDARY
      Address Type       : AUTO
      Source Interface    : management 0 ( 10.32.99.45 )
      First Rx           : 2025-06-27 12:11:41
      Last Rx            : 2025-06-27 12:50:59
device#
```

type (tunnel interface configuration)

Use the **type** command to specify the type of tunnel protocol to run on a tunnel interface.

Syntax

type vxlan

no type vxlan

Command Default

The current tunnel interface uses the Virtual eXtensible Local-Area Network (VXLAN) tunneling protocol.

Parameters

vxlan

Type of tunnel protocol to use. This release supports only the VXLAN protocol.

Modes

Tunnel interface configuration (config-tunnel-*name*)

Usage Guidelines

VXLAN is the most commonly used protocol to create overlay networks to enable the use of a virtual network. VXLAN tunnels Ethernet (Layer 2) traffic over an IP (Layer 3) network. This technology provides the same Ethernet Layer 2 network services as VLAN, but with more extensibility and flexibility.

If you want to view the status of the tunnel interfaces on a device, see [show interface tunnel \(Extreme ONE OS Switching\)](#) on page 1016.

To remove the tunnel protocol type configuration on a tunnel interface, use the **no** form of this command.

Examples

The following example configures tunnel interface 1 as a VXLAN interface on the device:

```
device# configure terminal
device(config)# interface tunnel 1
device(config-tunnel-1)# type vxlan
device(config-tunnel-1)#
```

The following example displays the tunnel interface configuration that is running currently on the device. In this example, the tunnel 1 interface runs VXLAN as the tunnel protocol:

```
device# show running-config interface tunnel

interface tunnel 1
  type vxlan
    source nve base
    destination 1.1.1.1
  no shutdown
!
device#
```

update-source (BGP peer group configuration)

Use the **update-source** to specify the source of routing updates for a BGP neighbor or all members of a specified BGP peer group within a VRF instance.

Syntax

```
update-source { source-ipv4-addr | source-ipv6-addr | loopback port-  
  number | { ethernet interface-name | port-channel port-channel-number  
  | ve interface-id } }  
  
no update-source { source-ipv4-addr | source-ipv6-addr | loopback port-  
  number | { ethernet interface-name | port-channel port-channel-number  
  | ve interface-id } }
```

Command Default

No source of routing updates for a BGP neighbor or all members of a specified BGP peer group is specified.

Parameters

source-ipv4-addr

IPv4 address of the update source. Use the *a.b.c.d* format. For example, 10.1.1.1.

source-ipv6-addr

IPv6 address of the update source. Use the *x:x::x:x* format. For example, 1001::1.

loopback *port-number*

Loopback port number. Use an integer from 1 to 1024.

ethernet *interface-name*

Ethernet interface name. Use the *slot/port:breakout* format to specify an interface or one or more comma-separated ranges of interfaces. For example, 0/1 or 1/1-3,5,2/7-9,10:1-4.

port-channel *port-channel-number*

Port channel number. Use an integer from 1 to 255.

ve *interface-id*

Virtual Ethernet (VE) interface ID. .

Modes

BGP peer group configuration (config-vrf-bgp-pg)

BGP peer group neighbor configuration (config-vrf-bgp-pg-neighbor)

Usage Guidelines

Use this command to anchor all the peer group members session to interfaces or set the source IP address for members of a specified BGP peer group.

Use the **no** form of this command to reset the update source functionality for the specified neighbor or the members of the specified BGP peer group.

Examples

The following example configures six BGP peer groups (group1 through group6) under a Virtual Routing and Forwarding (VRF) instance named violet. This example configures six peer groups (group1 through group6). In each peer group, the source of routing updates is modified from the default source for its members:

```
device# configure terminal
device(config)# vrf violet
device(config-vrf-violet)# router bgp
device(config-vrf-bgp)# peer-group group1
device(config-vrf-bgp-pg)# update-source 10.1.1.1
device(config-vrf-bgp-pg)# exit
device(config-vrf-bgp)# peer-group group2
device(config-vrf-bgp-pg)# update-source 1001::1
device(config-vrf-bgp-pg)# exit
device(config-vrf-bgp)# peer-group group3
device(config-vrf-bgp-pg)# update-source ethernet 0/1
device(config-vrf-bgp-pg)# exit
device(config-vrf-bgp)# peer-group group4
device(config-vrf-bgp-pg)# update-source ethernet 0/1
device(config-vrf-bgp-pg)# exit
device(config-vrf-bgp)# peer-group group5
device(config-vrf-bgp-pg)# update-source port-channel 1
device(config-vrf-bgp-pg)# exit
device(config-vrf-bgp)# peer-group group6
device(config-vrf-bgp-pg)# update-source port-channel 1
device(config-vrf-bgp-pg)#
```

The following example configures a BGP peer group named group1 under a VRF instance named violet. This example configures a peer group named group1. In the peer group, the source of routing updates is modified from the default source for specific members:

```
device# configure terminal
device(config)# vrf violet
device(config-vrf-violet)# router bgp
device(config-vrf-bgp)# peer-group group1
device(config-vrf-bgp-pg)# neighbor 10.1.1.1
device(config-vrf-bgp-pg-neighbor)# update-source 10.1.1.2
device(config-vrf-bgp-pg-neighbor)# exit
device(config-vrf-bgp-pg)# neighbor 20.1.1.1
device(config-vrf-bgp-pg-neighbor)# update-source ethernet 0/1
device(config-vrf-bgp-pg-neighbor)# exit
device(config-vrf-bgp-pg)# neighbor 30.1.1.1
device(config-vrf-bgp-pg-neighbor)# update-source ethernet 0/1
device(config-vrf-bgp-pg-neighbor)# exit
device(config-vrf-bgp-pg)# neighbor 40.1.1.1
device(config-vrf-bgp-pg-neighbor)# update-source port-channel 2
device(config-vrf-bgp-pg-neighbor)# neighbor 50.1.1.1
device(config-vrf-bgp-pg-neighbor)# update-source loopback 1
device(config-vrf-bgp-pg-neighbor)# exit
```

```

device(config-vrf-bgp-pg-neighbor)# update-source 3001::2
device(config-vrf-bgp-pg-neighbor)# exit
device(config-vrf-bgp-pg)# neighbor 4001::1
device(config-vrf-bgp-pg-neighbor)# update-source loopback 1
device(config-vrf-bgp-pg-neighbor)# exit
device(config-vrf-bgp-pg)# neighbor 5001::1
device(config-vrf-bgp-pg-neighbor)# update-source ethernet 0/1
device(config-vrf-bgp-pg-neighbor)# exit
device(config-vrf-bgp-pg)# neighbor 6001::1
device(config-vrf-bgp-pg-neighbor)# update-source ethernet 0/1
device(config-vrf-bgp-pg-neighbor)# exit
device(config-vrf-bgp-pg)# neighbor 7001::1
device(config-vrf-bgp-pg-neighbor)# update-source port-channel 10
device(config-vrf-bgp-pg-neighbor)# exit
device(config-vrf-bgp-pg)# neighbor 8001::1
device(config-vrf-bgp-pg-neighbor)# update-source port-channel 10
device(config-vrf-bgp-pg-neighbor)#

```

The following example displays the configuration of a VRF instance named violet that is running currently on the device. In this example, six peer groups are configured (group1 through group6). In each peer group, the source of routing updates is modified from the default source for its members:

```

device # show running-config vrf violet

vrf violet
!
  router bgp
    peer-group group1
      update-source 10.1.1.1
    peer-group group2
      update-source 1001::1
    peer-group group3
      update-source ethernet 0/1.0
    peer-group group4
      update-source ethernet 0/1.1
    peer-group group5
      update-source port-channel 1.0
    peer-group group6
      update-source port-channel 1.10
!
device#

```

The following example displays the configuration of a VRF instance named violet that is running currently on the device. In this example, two peer groups are configured (group1 and group2). In each peer group, the source of routing updates is modified from the default source for specific members:

```

device# show running-config vrf violet

vrf violet
!
  router bgp
    peer-group group1
      neighbor 10.1.1.1
        update-source 10.1.1.2
      neighbor 20.1.1.1
        update-source ethernet 0/1.0
      neighbor 30.1.1.1
        update-source ethernet 0/1.2
      neighbor 40.1.1.1
        update-source port-channel 2.20

```

```
    neighbor 50.1.1.1
      update-source loopback 1
peer-group group2
  neighbor 2001::1
    update-source 3001::2
  neighbor 4001::1
    update-source loopback 1
  neighbor 5001::1
    update-source ethernet 0/1.0
  neighbor 6001::1
    update-source ethernet 0/1.3
  neighbor 7001::1
    update-source port-channel 10.0
  neighbor 8001::1
    update-source port-channel 10.3
!
device#
```

uplink-track interface (MLAG configuration)

Use the **uplink-track interface** command to configure MLAG uplink track interfaces for multichassis link aggregation (MLAG).

Syntax

```
uplink-track interface { ethernet interface-name | port-channel port-channel-range }  
  
no uplink-track interface { ethernet interface-name | port-channel port-channel-range }
```

Command Default

No MLAG uplink tracking interfaces are configured on the device.

Parameters

ethernet *interface-name*

Ethernet interface name. Use the *slot/port* format or *slot/port:channel-range* format (also called breakout format). For example, 0/1 or 1/1-3,5,2/7-9,10:1-4 respectively.

port-channel *port-channel-range*

Port-channel number. Use an integer or one or more ranges of integers from 1 to 255. For example, 1 or 1-3,5,7-9 respectively.

Modes

MLAG configuration (config-mlag)

Usage Guidelines

To remove an MLAG uplink tracking interface configuration, use the no form of this command.

Examples

The following example configures two Ethernet interfaces and one port channel port for MLAG uplink tracking on interfaces:

```
device# configure terminal  
device(config)# mlag  
device(config-mlag)# role primary  
device(config-mlag)# mac 0000.aabb.ccdd  
device(config-mlag)# uplink-track interface ethernet 0/1  
device(config-mlag)# uplink-track interface ethernet 0/2  
device(config-mlag)# uplink-track interface port-channel 1  
device(config-mlag)#
```


The following example displays the MLAG configuration that is running currently on the device:

```
device# show running-config mlag

mlag
  role primary
  mac 0000.aabb.ccdd
  uplink-track interface ethernet 0/1
  uplink-track interface ethernet 0/2
  uplink-track interface port-channel 1
device#
```

use-multiple-paths (BGP address family configuration)

Use the **use-multiple-paths** command to enable BGP to install multiple internal BGP paths or multiple external BGP paths for the same route/prefix in a specific address family forwarding table within a Virtual Routing and Forwarding (VRF) instance.

Syntax

```
use-multiple-paths { ibgp maximum-paths number-of-paths } | ebgp maximum-paths number-of-paths [ allow-multiple-as ] }  
  
no use-multiple-paths { ibgp maximum-paths number-of-paths } | ebgp maximum-paths number-of-paths [ allow-multiple-as ] }
```

Command Default

BGP installs only one best route in the forwarding table for the address family.

Parameters

ibgp

Enables for internal BGP peers.

ebgp

Enables for external BGP peers.

maximum-paths *number-of-paths*

Maximum number of parallel BGP routes that can be installed in the forwarding table. Use an integer from 2 to 64.

allow-multiple-as

Disables the Autonomous System (AS) path check to allow more than one AS number.

Modes

VRF BGP address family configuration (*config-vrf-bgp-address-family-type*)

Usage Guidelines

Use the **no** form of this command to disable the ability of BGP to install multiple paths.

Examples

The following example configures a VRF instance named default-vrf. In this example, the VRF enables a maximum of two routes in the forwarding table for internal BGP peers and external BGP peers within the specified address family:

```
device# configure terminal
device(config)# vrf default-vrf
device(config-vrf-default-vrf)# router bgp
device(config-vrf-bgp)# address-family ipv4 unicast
device(config-vrf-bgp-ipv4u)# use-multiple-paths ibgp maximum-paths 2
device(config-vrf-bgp-ipv4u)#
device(config-vrf-bgp-ipv4u)# use-multiple-paths ebgp maximum-paths 2 allow-multiple-as
device(config-vrf-bgp-ipv4u)#
```

The following example displays the configuration of a VRF instance named default-vrf that is running currently on the device. In this example, the VRF enables a maximum of two routes in the forwarding table for internal BGP peers and external BGP peers within the address family. The AS path check is disabled for external BGP peers:

```
device# show running-config vrf default-vrf

vrf default-vrf
!
  router bgp
    address-family ipv4 unicast
      use-multiple-paths ibgp maximum-paths 2
      use-multiple-paths ebgp maximum-paths 2 allow-multiple-as
    !
device(config-vrf-bgp)#
```

use-multiple-paths

Use the **use-multiple-paths** command to enable BGP to install multiple internal BGP paths or multiple external BGP paths for the same route in the forwarding table for all address-families within a Virtual Routing and Forwarding(VRF) instance.

Syntax

```
use-multiple-paths { ibgp maximum-paths number-of-paths | ebgp | maximum-paths number-of-paths [ allow-multiple-as ] }
```

```
no use-multiple-paths { ibgp maximum-paths number-of-paths | ebgp | maximum-paths number-of-paths [ allow-multiple-as ] }
```

Command Default

BGP installs only one best route in the forwarding table.

Parameters

ibgp

Enables for internal BGP peers.

ebgp

Enables for external BGP peers.

maximum-paths *number-of-paths*

Maximum number of parallel BGP routes that can be installed in the forwarding table. Use an integer from 2 to 64.

allow-multiple-as

Disables the Autonomous System (AS) path check to allow more than one AS number.

Modes

VRF BGP configuration (config-vrf-bgp)

Usage Guidelines

Use the **no** form of this command to disable the ability of BGP to select multiple routes.

Examples

The following example configures a VRF instance named default-vrf. In this example, the VRF enables a maximum of two routes in the forwarding table for internal BGP peers and external BGP peers:

```
device# configure terminal
device(config)# vrf default-vrf
device(config-vrf-default-vrf)# router bgp
device(config-vrf-bgp)# use-multiple-paths ibgp maximum-paths 2
device(config-vrf-bgp)# use-multiple-paths ebgp maximum-paths 2 allow-multiple-as
device(config-vrf-bgp)#
```

The following example displays the configuration of a VRF instance named default-vrf that is running currently on the device. In this example, the VRF enables a maximum of two routes in the forwarding table for internal BGP peers and external BGP peers. The AS path check is disabled for external BGP peers:

```
device# show running-config vrf default-vrf

vrf default-vrf
!
  router bgp
    use-multiple-paths ibgp maximum-paths 2
    use-multiple-paths ebgp maximum-paths 2 allow-multiple-as
  !
device#
```

valid-lifetime (IPv6 router advertisement prefix configuration)

Use the valid-lifetime command to configure the valid lifetime for the address within a prefix for IPv6 router advertisement on an interface.

Syntax

```
valid-lifetime seconds  
no valid-lifetime
```

Command Default

The valid lifetime for the address within the current prefix is 2,592,000 seconds.

Parameters

seconds

Number of seconds. Use an integer from 0 to 4294967295 . This value must exceed the **preferred-lifetime** value.

Modes

IPv6 router advertisement configuration (config-if-*name*-ipv6-router-advertisement)

Usage Guidelines

Router advertisements are ICMPv6 messages sent by an IPv6 enabled router that are used by hosts to detect routers on an attached link. This functionality is similar to the IPv4 ICMP Router Discovery protocol.

Neighbor discovery defines five ICMP packet types: router solicitation and advertisement, neighbor solicitation and advertisement, and redirect messages.

Router advertisements carry information such as the router link layer address, prefixes supported in the network, and the MTU setting. Router advertisements are periodic messages or are also sent in response to router solicitations. Because of the nature of its functionality, router advertisement is considered as a separate feature from neighbor discovery. Two messages are supported as part of this functionality: router solicitation and router advertisement.

Each configured interface sends a router advertisement message periodically to the all-nodes link-local multicast address (FF02::1). To avoid synchronization of advertisements from other routers, the time interval is randomized to send these messages between the configured maximum and minimum time periods.

Router advertisement includes the following information:

- Default router—Whether the sender can be a default router and, if so, the lifetime for the router
- List of prefixes for on-link addresses
- Auto-configuration information
- Network information—Hop limit and MTU setting to use

The valid lifetime is the length of time that the prefix is valid relative to the time the packet was sent.

To restore the valid lifetime to the default value, use the **no** form of this command.

Examples

The following example configures a prefix of 100::1/64 for IPv6 router advertisement on Ethernet interface 0/1 and configures a valid lifetime of 10,000 seconds within that prefix:

```
device# configure terminal
device(config)# interface ethernet 0/1
device(config-if-eth-0/1)# ipv6-router-advertisement
device(config-if-eth-0/1-ipv6-router-advertisement)# prefix 100::1/64
device(config-if-ipv6-router-advertisement-prefix-100::1/64)# valid-lifetime 10000
device(config-if-ipv6-router-advertisement-prefix-100::1/64)#
```

The following example displays the configuration of Ethernet interface 0/1 that is running currently on the device. In this example, a valid lifetime of 10,000 seconds is configured within prefix 100::1/64 on the interface:

```
device# show running-config interface ethernet 0/1

!
interface ethernet 0/1
  ipv6 address 1111::10/64
  ipv6 neighbor 1111::1 00:04:96:eb:c4:51
  ipv6-router-advertisement
    hop-limit 133
    lifetime 200
    managed-config-flag
    mode disable-unsolicited-ra
    other-config
    enable
    interval 100 min 50
    prefix 100::1/63
    prefix 100::1/64
      advertisement
      autoconfiguration
      onlink
      preferred-lifetime 5000
      valid-lifetime 10000
  !
  no shutdown
!
device#
```

The following example displays details about IPv6 router advertisement on Ethernet interface 0/1:

```
device# show ipv6 router-advertisement interface ethernet 0/1

If Name: ethernet 0/1.0
ICMPv6 active timers:
  Last Router-Advertisement sent :0s
  Next Router-Advertisement sent in : 2s
Router-Advertisement parameters:
  Ra Enable flag : true
  Router lifetime field : 1800
  Periodic interval : 20 to 10 seconds
  Managed Address Configuration flag : true
  Other config flag : true
  RA Mode : disable_unsolicited_ra
  Current Hop Limit field : 0
  MTU option value : 1500
  Reachable Time field : 1200
device#
```


vlan-id

Use the **vlan-id** command to assign the VLAN ID for the VLAN mode bridge domain.

Syntax

```
vlan-id vlan-id
```

Parameters

```
vlan-id vlan-id
```

VLAN ID. The valid range is 1-4094.

Modes

Bridge domain VLAN configuration (config-vlan-bd-*name*)

Usage Guidelines

This command does not have a **no** form. To change the VLAN ID, you must delete and then re-create the bridge domain with the intended VLAN ID.

Examples

The following example assigns the VLAN ID for the VLAN mode bridge domain:

```
device(config)# bridge-domain 200 mode vlan
device(config-vlan-bd-200)# vlan-id 200

device# show running-config

bridge-domain 100
bridge-domain 200 mode vlan
vlan-id 200
device#
```

vm (IAH configuration)

Use the **vm** command to specify a virtual machine (VM) name and enter IAH VM configuration (`config-iah-vm-name`) mode.

Syntax

vm *vm-name*

no vm *vm-name*

Parameters

vm-name

Name of the VM to configure. Use any combination of 1 to 32 alphanumeric characters, underscores, hyphens, or periods. Do not use the “all” reserved keyword.

Modes

IAH configuration (`config-iah`)

Usage Guidelines

Use the **no** form of this command to remove the configuration for a VM.

To display details about a specific VM, see [show iah vm](#) on page 1006.

Examples

The following example specifies a VM named `Ubuntu1` on the device and enters IAH VM configuration (`config-iah-vm-name`) mode:

```
device# configure terminal
device(config)# iah
device(config-iah)# vm Ubuntu1
device(config-iah-vm-Ubuntu1)#
```

The following example displays the IAH configuration that is running currently on the device. In this example, IAH is configured with a VM named `Ubuntu1`:

```
device# show running-config iah

iah
  vm Ubuntu1
    cpus 1
    ram 1024
    disk sda
      size 5000
      format qcow2
      bus sata
      file flash://iah/disk1.qcow2
    !
```

```
!  
!  
device#
```

vni-domain (bridge domain configuration)

Use the **vni-domain** command to assign a Virtual Network Identifier (VNI) value for the bridge domain in the specified VNI domain.

Syntax

```
vni-domain vni-domain-name vni vni
```

Parameters

```
vni-domain vni-domain-name
```

Specifies the VNI domain name.

```
vni vni
```

Specifies the VNI value. The valid range is 1 to 16777215.

Modes

Bridge domain configuration (config-bd-*name*)

Usage Guidelines

To remove the configured VNI mapping for a bridge domain, use the no form of this command.

Examples

The following example configures the VNI mapping bridge domain:

```
device# configure terminal
device(config)# bridge-domain 200
device(config-bd-200)# vni-domain base vni 64000
device(config-bd-200)#

device# show running-config bridge-domain 200

bridge-domain 200 mode vlan
  vlan-id 200
  member ethernet 0/21
  vni-domain base vni 64000
device#
```

vni-domain (overlay configuration)

Use the **vni-domain** command to configure a Virtual Network Identifier (VNI) domain on a device and enter VNI configuration mode.

Syntax

vni-domain *vni-domain-name*

no vni-domain *vni-domain-name*

Command Default

The device is configured with a VNI domain named “base.”

Parameters

vni-domain-name

Name of the VNI domain to configure. You can use any combination of 1 to 64 uppercase or lowercase alphabetic characters, numerals, underscores, hyphens, or periods.

Modes

Overlay network configuration (config-overlay)

Usage Guidelines

A VNI domain defines the name space used by tunnels. A Network Virtualization Edge (NVE) endpoint (also called an NVE node), must have at least one VNI domain as a member.

Use the **no** form of this command to delete a VNI domain from the device.

Examples

The following example configures a VNI domain named DG1 on the device and enters VNI configuration mode:

```
device# configure terminal
device(config)# overlay-networks
device(config-overlay)# vni-domain DG1
device(config-vni-DG1)#
```

The following example shows the Network Virtualization Overlay (NVO) instances that are running currently on the device:

```
device# show running-config overlay-networks

overlay-networks
  default-nve base
```

```
default-vni-offset 0
nvo base
  member nve base
!
nve domain2
!
nve base
  source-interface loopback 1
  vni-domain base
!
  vni-domain base
!
device#
```

vrf (Extreme ONE OS Switching)

Use the **vrf** command to configure a Virtual Routing and Forwarding (VRF) instance and enter VRF configuration mode.

Syntax

vrf *vrf-name*

Parameters

vrf-name

Name of the VRF instance to configure. This name must start with an alphabetic character or an underscore followed by an arbitrary sequence of alphabetic or numeric characters, underscores, hyphens, or dots. The name cannot exceed 64 characters. Use a name that is unique across all the VRF instances on the device. Do not use the profile reserved keyword.

Modes

Global configuration (config)

Usage Guidelines

Use the **no** form of this command to delete a VRF instance from a device.

Examples

The following example configures a VRF instance named default-vrf on the device and enters VRF configuration mode (*config-vrf-vrf-name*):

```
device# configure terminal
device(config)# vrf default-vrf
device(config-vrf-default-vrf)#
```

The following example shows the configuration that is running currently on the device. In this example, a single VRF (in this case, default-vrf) exists on the device:

```
device# show running-config

...
vrf default-vrf
...
device#
```