

# Configuration Guide

## Netvisor ONE - Version 5.2.1

June 2020





## Table of Contents

---

Pluribus Netvisor One 5.2.1 .....	10
Legal Notice .....	11
EULA .....	12
Preface .....	20
Audience .....	21
Conventions .....	22
Document Feedback .....	24
Obtaining Documentation and Submitting a Service Request .....	25
About the Netvisor ONE CLI .....	26
Understanding Important Terms .....	27
Entering Commands and Getting Help .....	35
Finding Command Options .....	37
About Alternate Command Format .....	38
Specifying IP Address Netmasks .....	39
Specifying Measurement Units .....	40
Customizing Show Output Formats .....	41
Specifying a Switch or Fabric for Command Scope .....	44
Running Commands on a Local Switch .....	46
Changing Switch Setup Parameters .....	47
Creating Switch Groups .....	49
About GREP Support with Netvisor ONE OS .....	52
Installing Netvisor ONE & Initial Configuration .....	53
Changes to the End User License Agreement EULA .....	54
Adding License Keys to Netvisor ONE .....	55
Using the Serial Console Port for Initial Configuration .....	56
Managing Netvisor ONE Certificates .....	60
Enabling Administrative Services .....	63
Configuring Administrative Session Timeout .....	65
Confirming Connectivity on the Network .....	66
Setting the Date and Time .....	67
Viewing User Sessions on a Switch .....	69
Archiving Log Files Outside the Switch .....	70
Displaying and Managing Boot Environment Information .....	73
Exporting Configurations Using Secure Copy Protocol (SCP) .....	74
Upgrading the Netvisor ONE Software .....	76
Implementing a Fabric Upgrade .....	80
Saving and Restoring Netvisor OS Configurations .....	88
Copying and Importing Configuration Files .....	91
Autoconfiguration of IPv6 Addresses on the Management Interface Support .....	92
Configuring 10/100M Override Using Bel-Fuse SFP-1GBT-05 on F9372-X .....	93
Support for Local Loopback IP Addresses .....	94
Configuring REST API Access .....	95
Running Shell Commands Using REST API .....	104
Managing RMAs for Switches .....	106
Contacting Technical Assistance .....	107

Configuring Switch Ports .....	109
Displaying Port Numbering .....	110
Displaying Physical Port Layer1 and Layer2 Information .....	111
Displaying Port Status .....	112
Displaying Port Statistics .....	113
Configuring Ports for Different Throughput .....	115
Configuring Port Storm Control .....	117
Optimizing Port Link Status Detection on Pluribus Switches .....	118
Displaying Transceiver Information .....	120
Restoring Ports for Cluster Configurations .....	121
Limiting the Number of MAC Addresses per Port .....	125
Using Port Buffering .....	126
Changing Class of Service CoS Behavior .....	128
Configuring Minimum and Maximum Bandwidth on Ports .....	130
Enabling Jumbo Frame Support .....	132
Configuring Layer 2 Features .....	133
Understanding the Supported L2 Protocols .....	134
Configuring LLDP .....	135
Understanding and Configuring VLANs .....	137
Configuring Rapid Spanning Tree Protocol (RSTP) .....	144
Configuring Multiple Spanning Tree Protocol (MSTP) .....	153
Achieving a Loop-Free Layer 2 Topology .....	155
Fast Failover for STP and Cluster .....	159
Configuring Auto Recovery of a Disabled Port .....	161
Configuring STP Root Guard .....	163
Configuring Fabric Guard .....	164
About Layer 2 Hardware Hashing .....	165
Configuring Layer 3 Features .....	167
Understanding the Supported L3 Protocols .....	168
Configuring Packet Relay for DHCP Servers .....	169
Configuring vRouter Services .....	170
Configuring vRouter Interfaces .....	171
Configuring MTU Parameters for vRouter Interfaces .....	173
Configuring 802.1p-based Prioritization and Marking on Egress Interfaces .....	174
Configuring IPv6 for vRouter Loopback Addresses .....	175
Displaying FRR Routing and Debug Information for vRouters .....	177
Viewing FRR Logs .....	179
Configuring Hardware-based Routing .....	180
Configuring Hardware Routing for a vRouter .....	181
IPv6 Hardware Routing .....	182
Layer 3 Table Validation .....	184
Displaying Hardware Routes History .....	185
Understanding ECMP Path Selection and Load Balancing .....	186
About Layer 3 Hardware Hashing .....	187
Configuring Static Routes .....	189
Adding IPv6 Link-Local Addresses for Static Routing .....	190
Configuring Static Null Routing .....	191
Configuring Static ARP for Unicast Traffic .....	192



Configuring VRF Aware Static ARP .....	195
Guidelines and Limitations While Configuring Static ARP .....	197
Configuring IPv4 IPv6 NDP and Optimization .....	198
Configuring Routing Information Protocol (RIP) .....	201
Configuring Open Shortest Path First (OSPF) .....	203
Displaying Default Timers for OSPF Configurations .....	205
Configuring Route Maps for OSPF Routes .....	208
Enabling OSPF SNMP MIBs .....	210
Configuring Default Route Information Settings for OSPF Routing .....	211
Configuring Metric and Metric Type for Route Maps .....	213
Configuring BGP on a vRouter .....	214
Enabling BGP SNMP MIBs .....	219
Configuring AS and AS Prepending with BGP .....	220
Configuring BGP Communities .....	221
Configuring BGP Route Maps for Origin .....	224
Configuring BGP Graceful shutdown .....	225
Configuring BGP Unnumbered .....	227
Configuring Prefix Lists for BGP and OSPF .....	233
Configuring Bidirectional Forwarding Detection with BGP .....	234
Configuring BFD for OSPF Fault Detection .....	235
Configuring Optimized BFD Path .....	237
Configuring Policy-based Routing .....	240
Sending Network Traffic to an ECMP Group with PBR .....	243
Configuring Multicast Listener Discovery (MLD) .....	246
Configuring an IGMP Querier IP Address .....	248
Configuring Multicast Listener Discovery (MLD) Snooping per VLAN .....	249
Creating MLD Static Sources and Static Groups .....	251
Displaying MLD Statistics for a VLAN .....	252
Configuring and Administering the Pluribus Fabric .....	253
Understanding the Netvisor ONE Adaptive Cloud Fabric .....	255
Understanding Fabric Transactions .....	258
Understanding Fabric Status Updates, vPorts and Keepalives .....	260
Understanding the Different Fabric Deployment Models .....	261
Guidelines and Limitations .....	264
Creating an Initial Fabric .....	265
About the Default Configuration .....	266
Displaying Fabric Instances .....	267
Adding Switches to an Existing Fabric .....	268
Configuring the Fabric Over the Management Interface .....	270
Displaying Fabric Nodes .....	271
Displaying Fabric Information and Statistics .....	273
Configuring Layer 4 Ports for Fabric Communication .....	275
Configuring a Fabric Over a Layer 3 Network .....	277
Connecting Multiple Fabric Nodes over a Layer 3 Fabric .....	280
Troubleshooting the Fabric .....	282
Displaying the Transaction History .....	283
Keeping Transactions in Sync with Auto-Recovery .....	286
Rolling Back and Rolling Forward Transactions .....	288

Rolling Back the Configuration of the Fabric .....	290
Cluster Transactions Divergence .....	291
About the Fabric Default Parameters .....	294
Supported Releases .....	296
Configuring High Availability .....	297
Understanding the High Availability Feature in Netvisor ONE .....	298
Understanding Link Aggregation .....	299
Understanding the Link Aggregation Control Protocol (LACP) .....	300
Understanding Switch Clusters .....	303
About the Spanning Tree Protocol (STP) in Cluster Mode .....	308
About Split Brain .....	310
About Layer 3 Hardware Forwarding in a Cluster .....	311
Understanding vLAGs .....	312
Understanding Virtual Router Redundancy Protocol (VRRP) .....	316
About Cluster Active-Active Routing for IPv6 Addresses .....	318
Guidelines and Limitations .....	319
Configuring Static Trunking for Link Aggregation .....	320
Configuring Active-Standby Link Failover on Management Interfaces .....	321
Configuring Link Aggregation Control Protocol (LACP) .....	322
Configuring a Cluster .....	323
Performing the Cluster Re-peer Process .....	325
Configuring a vLAG .....	326
Modifying LACP Mode and Parameters on an Existing vLAG Configuration .....	327
Configuring the Cluster Slave Switch Bring-up Process .....	329
Configuring Active-Active vLAGs: a Step-by-Step Example .....	332
Understanding LAG Path Selection and Load Balancing .....	337
Understanding the vLAG Forwarding Rule .....	338
Configuring Asymmetric Routing over vLAGs .....	339
About Symmetric Routing over vLAGs .....	342
Configuring Active-Active vLAG Forwarding with Loopback Recirculation .....	346
Configuring Virtual Router Redundancy Protocol .....	349
Troubleshooting High Availability .....	354
Supported Releases .....	356
Related Documentation .....	357
Configuring VXLAN .....	358
Understanding VXLAN .....	360
About VXLAN's Packet Format and Its Implications .....	362
About Pluribus' VXLAN Implementation and Benefits .....	365
About Unicast Fabric VRFs with Anycast Gateway .....	376
About IGMP Snooping Support with VXLAN .....	380
About Distributed Multicast Forwarding with Fabric VRFs .....	382
About Pluribus Open Networking Multi-site Fabric .....	383
Guidelines and Limitations .....	385
Configuring the VXLAN Underlay Network .....	386
Configuring the Overlay: VTEP Interconnections and VNIs .....	387
Configuring the VXLAN Loopback Trunk .....	389
Checking VXLAN Recirculation's L2 and L3 Entries .....	390
Showing VXLAN Trunk Replication Counters .....	392

Displaying ECMP Load Balancing Info for VXLAN .....	393
Configuring VTEP Objects with Automatic Fabric Connections .....	395
Disabling VXLAN Termination .....	396
Configuring Unicast Fabric VRFs with Anycast Gateway .....	397
Configuring IGMP Snooping with VXLAN .....	404
Configuring Multicast Fabric VRFs .....	406
Supported Releases .....	408
Related Documentation .....	409
Configuring Advanced Layer 2 Transport Services .....	410
Understanding Virtual Link Extension .....	411
Understanding VXLAN-based Bridge Domains .....	417
Configuring Virtual Link Extension .....	419
Configuring Virtual Link Extension State Tracking .....	422
Configuring Virtual Link Extension Error Transparency .....	424
Configuring Virtual Link Extension Redundancy with Clusters .....	425
Showing Virtual Link Extension Between Ports on the Same Switch .....	427
Configuring Keep-Alive Timeout for Virtual Link Extension .....	428
Configuring VXLAN-based BDs for 802.1Q and QinQ Internetworking .....	430
Troubleshooting vLE .....	435
Guidelines and Limitations .....	436
Supported Releases .....	438
Related Documentation .....	439
Configuring and Using Network Management and Monitoring .....	440
Displaying System Statistics on a Switch .....	441
Understanding and Configuring Port Mirroring .....	442
Configuring Port Mirroring .....	443
Configuring Port Mirroring to a Remote Host .....	446
Configuring Logging .....	447
Sending Log Messages to Syslog Servers .....	451
Forwarding Log Files to an External Linux Server .....	452
Saving Diagnostic Files and Exporting to an External Server .....	454
Using Facility Codes with Log Messages .....	455
Displaying Log Counters Information .....	456
Viewing Log Events .....	457
Exceptions for Audit Logging .....	459
Understanding and Configuring SNMP .....	461
Overview .....	462
Configuring SNMP .....	463
Creating SNMP Communities on V1 and V2 .....	464
Creating SNMP Users on SNMPv3 .....	466
Modifying the SNMP Engine ID .....	469
Configuring Threshold on SNMP Processes .....	471
Supported SNMP MIBs .....	472
Routing MIBs .....	473
Enabling SNMP Traps .....	475
Using Additional SNMP commands .....	478
Supported SNMP Notifications (Traps) .....	482
Additional MIBs Supported in Netvisor ONE .....	486

Sample CLI Outputs .....	487
Related Documentation .....	489
Configuring and Using vFlows .....	490
Understanding vFlows and vFlow Objects .....	491
Configuring the Administrative Scope and State .....	493
Implementing the vFlow Policies .....	497
Filtering of Traffic Flows .....	501
Configuring vFlows with User Defined Fields (UDFs) .....	505
Forwarding Action in vFlow Filtering .....	509
Configuring vFlow Filters .....	516
Refreshing vFlow Level Statistics for Long-lived Connections .....	517
Commands and Parameters in vFlow .....	519
Guidelines and Limitations .....	522
Use Cases in vFlow .....	523
Supporting TCP Parameters using vFlows .....	524
Configuring Burst Size in vFlow for Maximum Bandwidth .....	526
Configuring Bandwidth Sharing for a Single VLAN .....	527
Enhancing vFlow Capoability .....	529
Using Application Flows and Statistics .....	531
Displaying vFlow Stats for all Switches .....	532
Understanding vFlow Statistics .....	533
Use Cases for Network Monitoring and Security .....	536
Using vFlows to Disable Communication .....	537
Configuring vFlows to Filter Packets on Management Interfaces .....	539
Configuring vFlow for Analytics .....	543
Configuring Network Security .....	546
About Port Isolation .....	547
Configuring Port Isolation .....	549
Creating and Implementing Access Control Lists (ACLs) .....	551
Using and Configuring MAC ACLs .....	552
Using and Configuring IP ACLs .....	555
Support for DHCP Snooping .....	560
Support for Router Advertisement (RA) Guard .....	562
Configuring Control Plane Traffic Protection (CPTP) .....	567
Configuring Port-based Control Plane Traffic Protection .....	570
Configuring Advanced Control Plane Traffic Protection .....	572
Configuring User-Defined Traffic Classes .....	575
Configuring Other Advanced CPTP Settings .....	577
Showing and Clearing CPTP Statistics .....	578
Use cases for QoS .....	581
Configuring DSCP to CoS Mapping .....	582
Applying CoS Queue Mapping based on Re-Marked DSCP in vFlow .....	585
Configuring and Using Network Telemetry .....	586
About sFlow .....	587
Guidelines While Configuring sFlow .....	590
Configuring the sFlow Collector .....	591
Configuring sFlow Agents on the Network .....	593
Adding or Removing Additional Ports to sFlow .....	596

Using Wireshark to Analyze Packets in Real Time .....	597
Analyzing Live Traffic Using Wireshark .....	599
Configuring TACACS+ .....	600
Understanding TACACS+ .....	601
Configuring TACACS+ .....	603
Creating User Roles .....	605
Exceptions for Audit Logging .....	608
Configuring Virtual Networks .....	609
Understanding Virtual Networks .....	609
Creating a Virtual Network (VNET) .....	612
Specifying the Type of VNET Interface .....	616
Configuring VNET High Availability (HA) .....	617
Related Documentation .....	620
Configuring Open Virtual Switch .....	621
Configuring OVSDDB with Netvisor ONE .....	622
Using OpenSSL TLS Certificates for OVSDDB and other Services .....	624
Open Virtual Switch Database (OVSDDB) Error Reporting .....	629
Open Source Acknowledgments .....	630
About Pluribus Networks .....	643

## Legal Notice

---

THE SPECIFICATIONS AND INFORMATION REGARDING THE PRODUCTS IN THIS MANUAL ARE SUBJECT TO CHANGE WITHOUT NOTICE. ALL STATEMENTS, INFORMATION, AND RECOMMENDATIONS IN THIS MANUAL ARE BELIEVED TO BE ACCURATE BUT ARE PRESENTED WITHOUT WARRANTY OF ANY KIND, EXPRESS OR IMPLIED. USERS MUST TAKE FULL RESPONSIBILITY FOR THEIR APPLICATION OF ANY PRODUCTS.

THE SOFTWARE LICENSE AND LIMITED WARRANTY FOR THE ACCOMPANYING PRODUCT ARE SET FORTH IN THE INFORMATION PACKET THAT SHIPPED WITH THE PRODUCT AND ARE INCORPORATED HEREIN BY THIS REFERENCE. IF YOU ARE UNABLE TO LOCATE THE SOFTWARE LICENSE OR LIMITED WARRANTY, CONTACT YOUR PLURIBUS NETWORKS REPRESENTATIVE FOR A COPY.

NOTWITHSTANDING ANY OTHER WARRANTY HEREIN, ALL DOCUMENT FILES AND SOFTWARE ARE PROVIDED “AS IS” WITH ALL FAULTS. PLURIBUS NETWORKS DISCLAIMS ALL WARRANTIES, EXPRESS OR IMPLIED, INCLUDING WITHOUT LIMITATION, THOSE OF MERCHANTABILITY, FITNESS FOR A PARTICULAR PURPOSE AND NONINFRINGEMENT OR ARISING FROM A COURSE OF DEALING, USAGE, OR TRADE PRACTICE.

IN NO EVENT SHALL PLURIBUS NETWORKS BE LIABLE FOR ANY INDIRECT, SPECIAL, CONSEQUENTIAL, OR INCIDENTAL DAMAGES, INCLUDING, WITHOUT LIMITATION, LOST PROFITS OR LOSS OR DAMAGE TO DATA, ARISING OUT OF THE USE OR INABILITY TO USE THIS MANUAL, EVEN IF PLURIBUS NETWORKS HAS BEEN ADVISED OF THE POSSIBILITY OF SUCH DAMAGES.

Any Internet Protocol (IP) addresses used in this document are not intended to be actual addresses. Any examples, command display output, and figures included in the document are shown for illustrative purposes only. Any use of actual IP addresses in illustrative content is unintentional and coincidental.

©2020 Pluribus Networks, Inc. All rights reserved. Pluribus Networks, the Pluribus Networks logo, nvOS, Netvisor®, vManage, vRender, PluribusCare, FreedomCare, Pluribus Cloud, and iTOR are registered trademarks or trademarks of Pluribus Networks, Inc., in the United States and other countries. All other trademarks, service marks, registered marks, registered service marks are the property of their respective owners. Pluribus Networks assumes no responsibility for any inaccuracies in this document. Pluribus Networks reserves the right to change, modify, transfer, or otherwise revise this publication without notice. © 2020 PLURIBUS NETWORKS. ALL RIGHTS RESERVED.

Version 5.2.1	June 2020
Version 5.2.0	March 2020
Version 5.1.2	February 2020
Version 5.1.1	December 2019
Version 5.1.0	September 2019
Version 5.0.0	April 2019

## EULA

---

### End-User License Agreement (“EULA”) Pluribus Networks, Inc.

IMPORTANT: PLEASE READ THIS END USER LICENSE AGREEMENT CAREFULLY. IT IS VERY IMPORTANT THAT YOU CONFIRM THAT YOU ARE PURCHASING PLURIBUS SOFTWARE OR EQUIPMENT FROM AN APPROVED SOURCE AND THAT YOU, OR THE ENTITY YOU REPRESENT (COLLECTIVELY, THE “CUSTOMER”) HAVE BEEN REGISTERED AS THE END USER FOR THE PURPOSES OF THIS PLURIBUS END USER LICENSE AGREEMENT. IF YOU ARE NOT REGISTERED AS THE END USER YOU HAVE NO LICENSE TO USE THE SOFTWARE AND THE LIMITED WARRANTY IN THIS END USER LICENSE AGREEMENT DOES NOT APPLY TO YOU.

#### 1. Introduction

- 1.1. Your use of the Pluribus Networks software (“SOFTWARE”) and related documentation (“DOCUMENTATION”) is subject to this legal agreement between you and PLURIBUS.  
“PLURIBUS” means Pluribus Networks, Inc., whose principal place of business is at 6001 America Center Drive, Suite 450, San Jose, CA 95002. This document explains how the agreement is made up, and sets out the terms of the agreement.
- 1.2. SOFTWARE means software provided to you by PLURIBUS.
- 1.3. DOCUMENTATION means written information (whether contained in user or technical manuals, training materials, specifications or other such materials provided to you by PLURIBUS) related to the SOFTWARE.
- 1.4. Unless otherwise agreed to in writing with PLURIBUS, your agreement with PLURIBUS at it relates to the SOFTWARE and DOCUMENTATION will always include, at a minimum, the terms and conditions set out in this EULA.
- 1.5. This agreement forms a legally binding agreement between you and PLURIBUS in relation to your use of the SOFTWARE and DOCUMENTATION. It is important that you take the time to read this EULA carefully. Collectively, this legal agreement is referred to below as the “EULA.”

#### 2. Accepting this EULA

- 2.1. In order to use the SOFTWARE and DOCUMENTATION, you must first agree to this EULA. You may not use the SOFTWARE and DOCUMENTATION if you do not accept this EULA.
- 2.2. You can accept this EULA by: (A) clicking to accept or agree to this EULA, where this option is made available to you by PLURIBUS; or (B) by actually downloading, installing or using the SOFTWARE, in which case, you understand and agree that PLURIBUS will treat your use of the SOFTWARE as acceptance of this EULA from that point onward.
- 2.3. If you are accepting this EULA on behalf of another legal entity, you represent that you have the authority to bind such legal entity.
- 2.4. You may not use the SOFTWARE and DOCUMENTATION and may not accept this EULA if (A) you are not of legal age to form a binding contract with PLURIBUS, or (B) you are a person barred from receiving the SOFTWARE under the laws of the United States or other countries including the country in which you are resident or from which you use the SOFTWARE.
- 2.5. PLURIBUS is willing to license this software to you only upon the condition that you purchased the software from an Approved Source and that you accept all of the terms contained in this EULA plus any additional limitations on the license set forth in a supplemental license agreement if any accompanying the product or available at the time of your order (collectively the “agreement”). To the extent of any conflict between the terms of this EULA and any supplemental license agreement, the supplemental license agreement shall apply.



2.6. You may access a copy of this EULA at: [www.pluribusnetworks.com/EULA](http://www.pluribusnetworks.com/EULA)

### 3. **Provision of the SOFTWARE AND DOCUMENTATION by PLURIBUS**

- 3.1. You acknowledge and agree that the form and nature of the SOFTWARE which PLURIBUS provides may change from time to time without prior notice to you.
- 3.2. You acknowledge and agree that PLURIBUS assumes no responsibility for any inaccuracies in the DOCUMENTATION and that PLURIBUS reserves the right to change, modify, or otherwise revise the DOCUMENTATION without notice.
- 3.3. As part of this continuing innovation, you acknowledge and agree that PLURIBUS may stop (permanently or temporarily) providing the SOFTWARE (or any features within the SOFTWARE) to you or to customers generally, at PLURIBUS' sole discretion, with 30 (thirty) days prior notice to you.

### 4. **Use of the SOFTWARE by you:**

- 4.1. You agree to use the SOFTWARE in accordance with this EULA.
- 4.2. You agree not to access (or attempt to access) the SOFTWARE by any means other than through the command line interface that is provided by PLURIBUS, unless you have been specifically allowed to do so in a separate agreement with PLURIBUS.
- 4.3. By downloading, installing, or using the software, you are representing that you purchased the software from an Approved Source and binding yourself to the agreement. If you do not agree to all of the terms of the agreement, then PLURIBUS is unwilling to license the software to you and (a) you may not download, install or use the software, and (b) you may return the software (including any unopened CD package and any written materials) for a full refund, or, if the software and written materials are supplied as part of another product, you may return the entire product for a full refund. Your right to return and refund expires 30 days after receipt of such product from an approved source, and applies only if you are the original and registered end user purchaser. For the purposes of this EULA, an "Approved Source" means (a) PLURIBUS; or (b) a distributor or systems integrator authorized by PLURIBUS to distribute/ sell PLURIBUS equipment, software and services within your territory to end users; or (c) a reseller authorized by any such distributor or systems integrator in accordance with the terms of the distributor's agreement with PLURIBUS to distribute/sell the PLURIBUS equipment, software and services within your territory to end users.
- 4.4. You agree that you will not engage in any activity that interferes with or disrupts the operation of the SOFTWARE.
- 4.5. Unless you have been specifically permitted to do so in a separate agreement with PLURIBUS, you agree that you will not reproduce, duplicate, copy, sell, trade or resell the SOFTWARE (or any portion thereof) for any purpose.
- 4.6. PLURIBUS may from time to time send updates over the Internet to you in order to update the SOFTWARE. You acknowledge and agree that PLURIBUS may make available new version of SOFTWARE without notice to you and that prior version of the SOFTWARE may be temporarily unavailable while PLURIBUS is updating the SOFTWARE. Pursuant to section 11 of this EULA, PLURIBUS is not liable for any disruptions in your use of the SOFTWARE, including while PLURIBUS is updating the SOFTWARE.

### 5. **Privacy and your Usage Information**

- 5.1. Should you enable sending product updates to the PluribusCloud, PLURIBUS may collect information ("USAGE INFORMATION") related to how you are using the SOFTWARE in accordance with your Pluribus Networks Purchase Agreement ("PURCHASE AGREEMENT"). The USAGE INFORMATION is collected and maintained in accordance with the PLURIBUS PRIVACY POLICY, as updated, located at: [www.pluribusnetworks.com/privacy](http://www.pluribusnetworks.com/privacy)
- 5.2. You agree to allow PLURIBUS to collect USAGE INFORMATION and give PLURIBUS a

perpetual, irrevocable, worldwide, royalty-free, and non-exclusive license to use (including, but not limited to, the rights to reproduce, adapt, and modify) the USAGE INFORMATION internally.

- 5.3. You give PLURIBUS a perpetual, irrevocable, worldwide, royalty-free, and non-exclusive license to: (a) generate aggregated, non-personal information, where aggregated, non-personal information is USAGE INFORMATION that is collected into groups so that it no longer reflects or references an individually identifiable person or legal entity, and (b), to the extent necessary, reproduce, adapt, modify, translate, publish, publicly perform, publicly display and distribute any generated aggregated, non-personal information.
- 5.4. You confirm and warrant to PLURIBUS that you have all the rights, power and authority necessary to grant PLURIBUS permission to collect USAGE INFORMATION and to use the USAGE INFORMATION in the manner specified in this section 5.
- 5.5. PLURIBUS agrees not to disclose the USAGE INFORMATION to any third party except in accordance with Section 5.3 or the PLURIBUS PRIVACY POLICY. In the event of a conflict between this EULA and the PLURIBUS PRIVACY POLICY, the PLURIBUS PRIVACY POLICY shall govern.

## **6. Proprietary Rights**

- 6.1. You acknowledge and agree that PLURIBUS owns all legal right, title and interest in and to the SOFTWARE and DOCUMENTATION, including any intellectual property rights which subsist in the SOFTWARE and DOCUMENTATION (whether those rights happen to be registered or not, and wherever in the world those rights may exist). You further acknowledge that the SOFTWARE and DOCUMENTATION may contain information which is designated confidential by PLURIBUS and that you shall not disclose such information without PLURIBUS' prior written consent.
- 6.2. You may use PLURIBUS' trademarks provided that such use is strictly limited and in accordance with the trademark guidelines located at [www.pluribusnetworks.com/legal](http://www.pluribusnetworks.com/legal) as adjusted from time to time.
- 6.3. Unless you have agreed otherwise to in writing with PLURIBUS, nothing in this EULA gives you a right to use any of PLURIBUS' domain names and other distinctive brand features (separate and apart from PLURIBUS' trademarks).
- 6.4. If you have been given an explicit right to use any of PLURIBUS' domain names and other distinctive brand features in a separate written agreement with PLURIBUS, then you agree that your use of such features shall be in compliance with that agreement, and any applicable provisions of this EULA.
- 6.5. You agree that you shall not remove, obscure, or alter any proprietary rights or notices (including copyright and trademark notices) which may be affixed to or contained within the SOFTWARE and DOCUMENTATION.

## **7. License from PLURIBUS**

- 7.1. PLURIBUS gives you a personal, worldwide, non-assignable, non-transferable, and non-exclusive license to use the SOFTWARE and DOCUMENTATION provided to you by PLURIBUS. Conditioned upon compliance with the terms and conditions of the Agreement, PLURIBUS grants to you a nonexclusive and nontransferable license to use for your internal business purposes the Software and the DOCUMENTATION for which you have paid the required license fees to an Approved Source. "DOCUMENTATION" means written information (whether contained in user or technical manuals, training materials, specifications or otherwise) pertaining to the SOFTWARE and made available by an Approved Source with the SOFTWARE in any manner (including on CD-Rom, or on-line). In order to use the SOFTWARE, you may be required to input a registration number or product authorization key and register your copy of the SOFTWARE online at PLURIBUS' website to obtain the necessary license key or license file.
- 7.2. Your license to use the SOFTWARE shall be limited to, and you shall not use the SOFTWARE in excess of, a single hardware chassis or card or such other limitations as are set forth in the

applicable Supplemental License Agreement or in the applicable Purchase Agreement or purchase order which has been accepted by an Approved Source and for which you have paid to an Approved Source the required license fee (the “Purchase Order”).

- 7.3. Unless otherwise expressly provided in the DOCUMENTATION or any applicable Supplemental License Agreement, you shall use the SOFTWARE solely as embedded in, for execution on, or (where the applicable DOCUMENTATION permits installation on non-Pluribus equipment) for communication with PLURIBUS equipment owned or leased by you from an Approved Source and used for your internal business purposes. No other licenses are granted by implication, estoppel or otherwise.
- 7.4. You may not (and you may not permit anyone else to) copy, modify, create a derivative work of any DOCUMENTATION, unless this is expressly permitted or required by law, or unless you have been specifically told that you may do so by PLURIBUS, in writing.
- 7.5. This is a license, not a transfer of title, to the Software and DOCUMENTATION, and PLURIBUS retains ownership of all copies of the SOFTWARE and DOCUMENTATION. Customer acknowledges that the SOFTWARE and DOCUMENTATION contain trade secrets of PLURIBUS or its suppliers or licensors, including but not limited to the specific internal design and structure of individual programs and associated interface information. Except as otherwise expressly provided under this EULA, you shall have no right, and you specifically agree not to:
  - i) transfer, assign or sublicense its license rights to any other person or entity (other than in compliance with any PLURIBUS relicensing/transfer policy then in force), or use the SOFTWARE on PLURIBUS equipment not purchased by you from an Approved Source or on secondhand PLURIBUS equipment, and you acknowledge that any attempted transfer, assignment, sublicense or use shall be void;
  - ii) make error corrections to or otherwise modify or adapt the SOFTWARE or create derivative works based upon the SOFTWARE, or permit third parties to do the same;
  - iii) reverse engineer or decompile, decrypt, disassemble or otherwise reduce the SOFTWARE to human-readable form, except to the extent otherwise expressly permitted under applicable law notwithstanding this restriction or except to the extent that PLURIBUS is legally required to permit such specific activity pursuant to any applicable open source license;
  - iv) publish any results of benchmark tests run on the SOFTWARE;
  - v) use or permit the SOFTWARE to be used on a service bureau or time sharing basis as relates to direct shared use of such SOFTWARE (and not to applications or services running upon or utilizing such SOFTWARE), without the express written authorization of PLURIBUS; or
  - vi) disclose, provide, or otherwise make available trade secrets contained within the SOFTWARE and DOCUMENTATION in any form to any third party without the prior written consent of PLURIBUS. You shall implement reasonable security measures to protect such tradesecrets
- 7.6. To the extent required by applicable law, and at your written request, PLURIBUS shall provide you with the interface information needed to achieve interoperability between the SOFTWARE and another independently created program, on payment of PLURIBUS' applicable fee, if any. You shall observe strict obligations of confidentiality with respect to such information and shall use such information in compliance with any applicable terms and conditions upon which PLURIBUS makes such information available.

## **8. Ending your relationship with PLURIBUS**

- 8.1. This EULA will continue to apply and will not come to an end until terminated by either you or PLURIBUS as set out below.
- 8.2. PLURIBUS shall terminate its legal agreement with you if: (a) you have breached any provision of this EULA (or have acted in manner which clearly shows that you do not intend to, or are unable

to comply with the provisions of this EULA), automatically and without requiring any further action by PLURIBUS; or (b) PLURIBUS is required to do so by law (for example, where the provision of the SOFTWARE or DOCUMENTATION to you is, or becomes, unlawful).

- 8.3. When this EULA comes to an end, all of the legal rights, obligations and liabilities that you and PLURIBUS have benefited from, been subject to (or which have accrued over time whilst this EULA has been in force) or which are expressed to continue perpetually, shall be unaffected by this cessation. Upon termination, you shall destroy all copies of SOFTWARE and DOCUMENTATION in your possession or control.

## 9. WARRANTY

- 9.1. Except as specifically warranted in the PURCHASE AGREEMENT, PLURIBUS disclaims all warranties as set forth in Section 10 of this agreement. Subject to the limitations and conditions set forth herein, PLURIBUS warrants that commencing from the date of shipment to you (but in case of resale by an Approved Source other than PLURIBUS, commencing not more than ninety (90) days after original shipment by PLURIBUS), and continuing for a period of the longer of (a) ninety (90) days or (b) the warranty period (if any) expressly set forth as applicable specifically to SOFTWARE in the warranty card accompanying the product of which the SOFTWARE is a part (the "Product") (if any): (a) the media on which the SOFTWARE is furnished will be free of defects in materials and workmanship under normal use; and (b) the SOFTWARE substantially conforms to the DOCUMENTATION.
- 9.2. The date of shipment of a Product by PLURIBUS is set forth on the packaging material in which the Product is shipped. Except for the foregoing, the SOFTWARE is provided "AS IS". This limited warranty extends only to the SOFTWARE purchased from an Approved Source by a user who is the first registered end user. Your sole and exclusive remedy and the entire liability of PLURIBUS and its suppliers under this limited warranty will be (i) replacement of defective media and/or (ii) at PLURIBUS' option, repair, replacement, or refund of the purchase price of the SOFTWARE, in both cases subject to the condition that any error or defect constituting a breach of this limited warranty is reported to the Approved Source supplying the Software to you, within the warranty period. PLURIBUS or the Approved Source supplying the SOFTWARE to you may, at its option, require return of the SOFTWARE and/or DOCUMENTATION as a condition to the remedy.
- 9.3. In no event does PLURIBUS warrant that the SOFTWARE is error free or that you will be able to operate the SOFTWARE without problems or interruptions. In addition, due to the continual development of new techniques for intruding upon and attacking networks, PLURIBUS does not warrant that the SOFTWARE or any equipment, system or network on which the SOFTWARE is used will be free of vulnerability to intrusion or attack.
- 9.4. This warranty does NOT apply if the SOFTWARE, Product or any other equipment upon which the SOFTWARE is authorized to be used (a) has been altered, except by PLURIBUS or its authorized representative, (b) has not been installed, operated, repaired, or maintained in accordance with instructions supplied by PLURIBUS, (c) has been subjected to abnormal physical or electrical stress, abnormal environmental conditions, misuse, negligence, or accident; or (d) is licensed for beta, evaluation, testing or demonstration purposes. The SOFTWARE warranty also does not apply to (e) any temporary SOFTWARE modules; (f) any SOFTWARE not posted on the software update or support site on Pluribus' web page; (g) any SOFTWARE that PLURIBUS expressly provides on an "AS IS" basis on PLURIBUS software update or support site on Pluribus' web page; (h) any SOFTWARE for which PLURIBUS or an Approved Source does not receive a license fee; and (i) SOFTWARE supplied by any third party which is not an Approved Source.

## 10. EXCLUSION OF WARRANTIES



- 10.1. NOTHING IN EULA, INCLUDING SECTIONS 10 AND 11, SHALL EXCLUDE OR LIMIT PLURIBUS' WARRANTY OR LIABILITY FOR LOSSES WHICH MAY NOT BE LAWFULLY EXCLUDED OR LIMITED BY APPLICABLE LAW. SOME JURISDICTIONS DO NOT ALLOW THE EXCLUSION OF CERTAIN WARRANTIES OR CONDITIONS OR THE LIMITATION OR EXCLUSION OF LIABILITY FOR LOSS OR DAMAGE CAUSED BY NEGLIGENCE, BREACH OF CONTRACT OR BREACH OF IMPLIED TERMS, OR INCIDENTAL OR CONSEQUENTIAL DAMAGES. ACCORDINGLY, ONLY THE LIMITATIONS WHICH ARE LAWFUL IN YOUR JURISDICTION WILL APPLY TO YOU AND OUR LIABILITY IS LIMITED IN ALL CASES TO THE MAXIMUM EXTENT PERMITTED BY LAW.
- 10.2. YOU EXPRESSLY UNDERSTAND AND AGREE THAT YOUR USE OF THE SOFTWARE AND DOCUMENTATION IS AT YOUR SOLE RISK AND THAT THE SOFTWARE AND DOCUMENTATION IS PROVIDED "AS IS" AND "AS AVAILABLE."
- 10.3. IN PARTICULAR, PLURIBUS DOES NOT REPRESENT OR WARRANT TO YOU THAT:
- YOUR USE OF THE SOFTWARE AND DOCUMENTATION WILL MEET YOUR REQUIREMENTS,
  - YOUR USE OF THE SOFTWARE OR ACCESS TO THE DOCUMENTATION WILL BE UNINTERRUPTED, TIMELY, SECURE OR FREE FROM ERROR,
  - ANY INFORMATION OBTAINED BY YOU AS A RESULT OF YOUR USE OF THE SOFTWARE AND DOCUMENTATION WILL BE ACCURATE OR RELIABLE, AND
  - THAT DEFECTS IN THE OPERATION OR FUNCTIONALITY OF ANY SOFTWARE PROVIDED TO YOU WILL BE CORRECTED.
- 10.4. ANY MATERIAL DOWNLOADED OR OTHERWISE OBTAINED THROUGH THE USE OF THE SOFTWARE IS DONE AT YOUR OWN DISCRETION AND RISK AND YOU, AND NOT PLURIBUS, WILL BE RESPONSIBLE FOR ANY DAMAGE TO YOUR COMPUTER SYSTEM OR OTHER DEVICES OR LOSS OF DATA THAT RESULTS FROM THE DOWNLOAD OF ANY SUCH MATERIAL.
- 10.5. NO ADVICE OR INFORMATION, WHETHER ORAL OR WRITTEN, OBTAINED BY YOU FROM PLURIBUS SHALL CREATE ANY WARRANTY NOT EXPRESSLY STATED IN THIS EULA.
- 10.6. PLURIBUS FURTHER EXPRESSLY DISCLAIMS ALL WARRANTIES AND CONDITIONS OF ANY KIND, WHETHER EXPRESS OR IMPLIED, INCLUDING, BUT NOT LIMITED TO THE IMPLIED WARRANTIES AND CONDITIONS OF MERCHANTABILITY, FITNESS FOR A PARTICULAR PURPOSE AND NON-INFRINGEMENT.

## **11. LIMITATION OF LIABILITY**

- 11.1. SUBJECT TO THE OVERALL PROVISION IN PARAGRAPH 10.1 ABOVE, YOU EXPRESSLY UNDERSTAND AND AGREE THAT PLURIBUS, SHALL NOT BE LIABLE TO YOU FOR:
- ANY DIRECT, INDIRECT, INCIDENTAL, SPECIAL CONSEQUENTIAL OR EXEMPLARY DAMAGES WHICH MAY BE INCURRED BY YOU, HOWEVER CAUSED AND UNDER ANY THEORY OF LIABILITY. THIS SHALL INCLUDE, BUT NOT BE LIMITED TO, ANY LOSS OF PROFIT (WHETHER INCURRED DIRECTLY OR INDIRECTLY), ANY LOSS OF GOODWILL OR BUSINESS REPUTATION, ANY LOSS OF DATA SUFFERED, COST OF PROCUREMENT OF SUBSTITUTE GOODS OR SERVICES, OR OTHER INTANGIBLE LOSS;
  - ANY LOSS OR DAMAGE WHICH MAY BE INCURRED BY YOU, INCLUDING BUT NOT LIMITED TO LOSS OR DAMAGE AS A RESULT OF:
    - ANY CHANGES WHICH PLURIBUS MAY MAKE TO THE SOFTWARE OR DOCUMENTATION, OR FOR ANY PERMANENT OR TEMPORARY CESSATION IN THE PROVISION OF THE SOFTWARE (OR ANY FEATURES WITHIN THE SOFTWARE); OR
    - THE DELETION OF, CORRUPTION OF, OR FAILURE TO STORE, ANY CONTENT AND OTHER COMMUNICATIONS DATA MAINTAINED OR TRANSMITTED BY OR THROUGH YOUR USE OF THE SOFTWARE.
- 11.2. THE LIMITATIONS ON PLURIBUS'S LIABILITY TO YOU IN PARAGRAPH 11.1 ABOVE SHALL APPLY WHETHER OR NOT PLURIBUS HAS BEEN ADVISED OF OR SHOULD HAVE BEEN AWARE OF THE POSSIBILITY OF ANY SUCH LOSSES ARISING.

11.3. IN NO EVENT SHALL THE AGGREGATE LIABILITY OF PLURIBUS EXCEED \$5,000. OR EXCEED THE PRICE PAID BY CUSTOMER TO ANY APPROVED SOURCE FOR THE SOFTWARE THAT GAVE RISE TO THE CLAIM OR IF THE SOFTWARE IS PART OF ANOTHER PRODUCT, THE PRICE PAID FOR SUCH OTHER PRODUCT. THIS LIMITATION OF LIABILITY FOR SOFTWARE IS CUMULATIVE AND NOT PER INCIDENT (I.E. THE EXISTENCE OF TWO OR MORE CLAIMS WILL NOT ENLARGE THIS LIMIT).

11.4. You acknowledge and agree that PLURIBUS has set its prices and entered into this EULA and any Purchase Agreement or Purchase Order with you in reliance upon the disclaimers of warranty and the limitations of liability set forth herein, that the same reflect an allocation of risk between the parties (including the risk that a contract remedy may fail of its essential purpose and cause consequential loss), and that the same form an essential basis of the bargain between the parties.

12. The **Software** may contain or be delivered with one or more components, which may include third-party components, identified by Pluribus in the Documentation, readme.txt file, third-party click-accept or elsewhere (e.g. on [www.pluribusnetworks.com](http://www.pluribusnetworks.com)) (the “Identified Component(s)”) as being subject to different license agreement terms, disclaimers of warranties, limited warranties or other terms and conditions (collectively, “Additional Terms”) than those set forth herein. You agree to the applicable Additional Terms for any such Identified Component(s).

12.1. Notwithstanding other statements in this EULA, third party software including free, Copyleft and open source software components (collectively referred to as “Third Party Software”) are distributed in compliance with the particular licensing terms and conditions attributable to the Third Party Software. PLURIBUS provides the Third Party Software to You “AS IS” without any warranties or indemnities of any kind.

12.2. Copyright notices and licensing terms and conditions applicable to the Third Party Software will be available for review on the Pluribus’ web site, and are included on the media on which you received the SOFTWARE within a “ATTRIBUTIONS” file (e.g., [attributions.pdf](#) or [attributions.txt](#)) included within the downloaded files, and/or reproduced within the materials or DOCUMENTATION accompanying the SOFTWARE.

### 13. Audit

13.1. PLURIBUS reserves the right to take steps PLURIBUS believes are reasonably necessary or appropriate to enforce and/or verify compliance with any part of this EULA. You agree that PLURIBUS has the right, without liability to you, to disclose any USAGE INFORMATION to law enforcement authorities, government officials, and/or a third party, as PLURIBUS believes is reasonably necessary or appropriate to enforce and/or verify compliance with any part of this EULA (including but not limited to PLURIBUS’ right to cooperate with any legal process relating to your use of the SOFTWARE, and/or a third-party claim that your use of the SOFTWARE is unlawful and/or infringes such third party rights).

### 14. General legal terms

14.1. This EULA constitutes the whole legal agreement between you and PLURIBUS and governs your use of the SOFTWARE and DOCUMENTATION and completely replaces any prior agreements between you and PLURIBUS in relation to the SOFTWARE and DOCUMENTATION (but excluding any SOFTWARE and DOCUMENTATION which PLURIBUS may provide to you under a separate written agreement).

14.2. You agree that PLURIBUS may provide you with notices, including those regarding changes to this EULA, by email, regular mail, or via the user interface implemented by the SOFTWARE.

14.3. You agree that if PLURIBUS does not exercise or enforce any legal right or remedy which is contained in this EULA (or which PLURIBUS has the benefit of under any applicable law), this will not be taken to be a formal waiver of PLURIBUS’ rights and that those rights or remedies will still

be available to PLURIBUS.

- 14.4. If any court of law or arbitration panel, having the jurisdiction to decide on this matter, rules that any provision of this EULA is invalid, then that provision will be removed from this EULA without affecting the rest of this EULA. The remaining provisions of this EULA will continue to be valid and enforceable.
- 14.5. You and PLURIBUS agree that this EULA, and your relationship with PLURIBUS under this EULA, shall be governed by the laws of the State of California without regard to its conflict of laws provisions.
- 14.6. The SOFTWARE and DOCUMENTATION is deemed to include “commercial computer software” and “commercial computer software documentation,” respectively, pursuant to DFAR Section 227.7202 and FAR Section 12.212, as applicable. Any use, modification, reproduction, release, performance, display or disclosure of the SOFTWARE and DOCUMENTATION by the United States Government shall be governed solely by this EULA.
- 14.7. In the event that the Uniform Computer Information Transaction Act, any version thereof or a substantially similar law (collectively “UCITA”) is enacted and/or interpreted as to be applicable to the performance of PLURIBUS under this Agreement, the statute shall not govern any aspect of this Purchase Agreement, any license granted hereunder, nor any of the rights and obligations of the parties pursuant to this EULA.
- 14.8. You agree that the SOFTWARE and DOCUMENTATION will not be shipped, transferred, or exported into any country or used in any manner prohibited by the United States Export Administration Act or any other exports laws, restrictions, or regulations. All rights to use the SOFTWARE and DOCUMENTATION are granted on condition that such rights are forfeited if you fail to comply with this EULA.
- 14.9. ANY CLAIM, DISPUTE, OR CONTROVERSY (WHETHER IN CONTRACT, TORT, OR OTHERWISE, WHETHER PREEXISTING, PRESENT OR FUTURE, AND INCLUDING STATUTORY, CONSUMER PROTECTION, COMMON LAW, INTENTIONAL TORT AND EQUITABLE CLAIMS) BETWEEN YOU AND PLURIBUS, its agents, employees, principals, successors, assigns, affiliates (collectively for purposes of this paragraph, “PLURIBUS”) arising from or relating to this EULA, its interpretation, or the breach, termination or validity thereof, the relationships which result from this EULA (including, to the full extent permitted by applicable law, relationships with third parties who are not signatories to this Agreement) SHALL BE EXCLUSIVELY AND FINALLY SETTLED BY ARBITRATION. THE ARBITRATION SHALL BE HELD IN SANTA CLARA, CALIFORNIA AND CONDUCTED IN ACCORDANCE WITH THE COMMERCIAL ARBITRATION RULES OF THE AMERICAN ARBITRATION ASSOCIATION. THE ARBITRATION SHALL BE CONDUCTED BEFORE THREE ARBITRATORS, ONE SELECTED BY EACH OF THE PARTIES, AND THE THIRD SELECTED BY THE FIRST TWO ARBITRATORS. JUDGMENT UPON THE AWARD RENDERED MAY BE ENTERED IN ANY COURT HAVING JURISDICTION, OR APPLICATION MAY BE MADE TO SUCH COURT FOR JUDICIAL ACCEPTANCE OF THE AWARD AND IN ORDER OF ENFORCEMENT AS THE CASE MAY BE.



## Preface

---

- [Audience](#)
- [Conventions](#)
- [Documentation Feedback](#)
- [Obtaining Documentation and Submitting a Service Request](#)

## Audience

---

This publication is for experienced network administrators responsible for configuring and maintaining network switches with some expertise in the following areas:

- Network administration
- Storage administration
- Server administration
- Application delivery administration
- Network security administration

## Conventions

This document uses the following conventions:

Convention	Indication
Bold font	Keywords, user interface elements, and user-entered text appear in <b>bold</b> font.
<i>Italic font</i>	Document titles, new or emphasized terms, and variables that you supply values are in <i>italic</i> font.
[]	Elements in square brackets are optional.
{x y z}	Required elements are grouped in curly braces and are separated by vertical bars.
[x y z]	Optional parameters are grouped in brackets and separated by vertical bars.
String	A non-quoted set of characters. Do not use quotation marks around the string or the string includes the quotation marks.
courier font	Command Line Interface (CLI) commands and samples appear in courier font.
< >	Non-printing characters such as passwords are indicated by angle brackets.
[]	Default responses to system prompts are in square brackets.
:	Indicates that you enter the following text at the command prompt.

<b>Note:</b>	Indicates information of special interest.
<b>Caution!</b>	Indicates a situation that could cause equipment failure or loss of data.
<b>TIP!</b>	Indicates information that can help you solve a problem.
<b>Warning:</b>	Indicates information that could impact you or your network.

**Time Saver:**

Indicates information that can help you save time.

## Documentation Feedback

---

To provide technical feedback on this document, or to report an error or omission, please send your comments to: [doc-feedback@pluribusnetworks.com](mailto:doc-feedback@pluribusnetworks.com)

We appreciate your feedback.

## Obtaining Documentation and Submitting a Service Request

---

For information on obtaining documentation, submitting a service request, and gathering additional information, please visit [www.pluribusnetworks.com](http://www.pluribusnetworks.com).

## About the Netvisor ONE CLI

---

This chapter provides information for understanding and using the Pluribus Networks Netvisor ONE command line interface (CLI) on a Netvisor ONE switch.

---

- [Understanding Important Terms](#)
  - [Entering Commands and Getting Help](#)
  - [Finding Command Options](#)
  - [About Alternate Command Format](#)
  - [Specifying IP Address Netmasks](#)
  - [Specifying Measurement Units](#)
  - [Customizing Show Output Formats](#)
  - [Specifying a Switch or Fabric for Command Scope](#)
  - [Running Commands on a Local Switch](#)
  - [Changing Switch Setup Parameters](#)
  - [Creating Switch Groups](#)
  - [About GREP Support with Netvisor ONE OS](#)
-



## Understanding Important Terms

The following list of important terms and concepts as well as definitions is important for understanding Netvisor One features and determine the best configuration to meet your needs.

Term	Meaning
<b>ACL</b>	An Access Control List is a list of rules that are used to filter network traffic and apply certain actions to it.
<b>Adaptive Cloud Fabric</b>	A number of Netvisor ONE-powered switches that operate and are managed as a single holistic entity is referred to as <i>Adaptive Cloud Fabric</i> ( <i>Fabric</i> in short).
<b>API</b>	An Application Programming Interface is a method to interact with Netvisor ONE switches (typically in a programmatic way) that is functionally equivalent to and has a similar scope as the CLI.
<b>ARP</b>	The Address Resolution Protocol is an IETF standard protocol used to discover the link layer address, such as a MAC address, associated with a given IPv4 address.
<b>AS</b>	An Autonomous System is a collection of connected IP routing prefixes under the control of one or more network operators on behalf of a single administrative entity or domain.
<b>BFD</b>	Bidirectional Forwarding Detection is a UDP-based protocol that provides fast detection of Layer 3 peer failures. It is used in conjunction with routing protocols to accelerate convergence in IP networks.
<b>BGP</b>	The Border Gateway Protocol is a popular standard routing protocol used to exchange routing and reach-ability data among autonomous systems.
<b>BPDU</b>	A Bridge Protocol Data Unit is a frame that carries information about the Spanning Tree Protocol (STP).
<b>CLI</b>	The Command Line Interface is a method to interact with Netvisor ONE switches through user-entered commands, which can be executed on an individual switch, on a cluster, or on a fabric.
<b>Cluster</b>	A pair of adjacent Netvisor ONE-powered switches acting as one logical unit at Layer 2 for high availability.
<b>CoS</b>	Class of Service: a 3-bit Ethernet field defined by the IEEE 802.1p standard. It's used to define and apply eight possible levels of QoS to the traffic.
<b>Cgroups</b>	cgroups (abbreviated from control groups), is a Linux kernel feature that limits, accounts for, and isolates the resource usage (such as CPU, memory, disk I/O, network) of a collection of processes.

<b>DCI</b>	Data Center Interconnect is a category of technologies, including those leveraging the VXLAN packet encapsulation, meant to enable the remote interconnection of data centers for improved scalability, performance and reliability or fault tolerance.
<b>DHCP</b>	The Dynamic Host Configuration Protocol is a network management protocol used in IP networks to dynamically assign an IP address and other network configuration parameters to each DHCP client device (host or network node) from one or more DHCP servers.
<b>DSCP</b>	The Differentiated Services Code Point is a 6-bit value in the 8-bit Differentiated Services (DS) field in the IP header. It's used for packet classification purposes for QoS and other applications.
<b>ECMP</b>	Equal-Cost Multi-Path is a routing strategy in which next-hop packet forwarding to a single destination can occur over multiple <i>best paths</i> .
<b>EULA</b>	End User License Agreement (or software license agreement) is the contract between the licensors and purchaser, establishing the purchaser's right to use the software.
<b>FIB</b>	The Forwarding Information Base is the (software or hardware) IP forwarding table used by a switch or router to forward IP packets to their destinations.
<b>Firewall</b>	A firewall is a network security system that monitors and controls incoming and outgoing network traffic based on predetermined security policies.
<b>FRR</b>	An IP routing protocol suite for Linux and Unix platforms and include protocol daemons for BGP, IS-IS, LDP, OSPF, PIM, and RIP. FR Routing (FRR) is used for connecting hosts, virtual machines, and containers to the network for network switching and routing, advertising network services, and internet peering.
<b>In-band Interface</b>	An internal interface facing the Netvisor ONE kernel used as a fabric-control port when building a fabric over any IP network.
<b>In-band IP Address</b>	The IP address of the switch on a production or management network for administration and inter-switch communication.
<b>ICMP</b>	The Internet Control Message Protocol is a supporting protocol in the Internet Protocol (IP) suite. It is used by network devices, including routers, to send error messages and operational information. With IP version 6, ICMPv6 expanded its capabilities to support additional functions such as Neighbor Discovery Protocol (NDP) and Multicast Listener Discovery (MLD).
<b>IDS</b>	An Intrusion Detection System is a device or a software application that monitors the network infrastructure and/or the end devices for malicious activity or policy violations.
<b>IGMP</b>	The Internet Group Management Protocol is a communications protocol used by hosts and adjacent routers on IPv4 networks to establish multicast group memberships.

## Insight Analytics

Insight Analytics is a Network Performance Management (NPM) add-on module to UNUM.

## IPS

An Intrusion Prevention System, also known as intrusion detection and prevention system (IDPS), is a network security appliance that monitors the network and/or the end devices for malicious activity. The main functions of an IPS are: to identify malicious activity, to log information about this activity, to report it and also to attempt to block it.

## Jumbo Frames

Jumbo frames, or jumbos in short, are Ethernet frames with more than 1500 bytes of payload.

## LACP

Link Aggregation Control Protocol (LACP) is a protocol for the collective handling of multiple physical ports that can be seen as a single logical transmission channel (also called trunk, port channel, link aggregation group or link bundle) for network purposes such as traffic load balancing and link redundancy. It was defined in the IEEE 802.3ad standard, later incorporated into 802.3 and later moved to IEEE 802.1AX-2008.

## LAG

Link aggregation is a technology used to combine multiple connections in order to increase the aggregate bandwidth beyond what a single connection can sustain, and to provide redundancy in case of link failure. A Link Aggregation Group (LAG) *bundles* a number of physical ports together to create a single high-bandwidth data path, so as to implement traffic load sharing and link redundancy. Other terms used to describe this technology include port trunking, port channel, link bundling, channel bonding "and-with" servers–NIC bonding and NIC teaming. The link aggregation process is supported by a dynamic protocol called Link Aggregation Control Protocol (LACP).

## LLDP

The Link Layer Discovery Protocol is a standard link layer protocol (IEEE 802.1AB) used by network devices to advertise their identity, capabilities and neighbors on an IEEE 802 local area network.

## MAC Address

The medium access control (MAC) is a sub-layer of the data link layer in IEEE 802 LAN/MAN standards.

## MIB

A Management Information Base is a database used for managing the entities in a computer network. MIBs are typically used with Simple Network Management Protocol (SNMP).

## MLD

Multicast Listener Discovery is a process used by IPv6 routers to discover multicast listeners on a directly attached link, much like the Internet Group Management Protocol (IGMP) is used in IPv4.

## MSTP

The Multiple Spanning Tree Protocol is a protocol introduced by the IEEE 802.1s standard and later incorporated into IEEE 802.1Q-2005, to extend the Rapid Spanning Tree Protocol to support multiple STP instances for load balancing and to introduce various other protocol enhancements.

## MTU

The Maximum Transmission Unit is the size of the largest protocol data unit (PDU) that can be transmitted in a single network layer or

	data link layer transaction.
<b>NDP or ND</b>	Neighbor Discovery (Protocol) is an IPv6 node discovery process that has similar (and improved) functionalities compared to IPv4's ARP. It is based on the ICMPv6 standard protocol.
<b>Netflow</b>	NetFlow is a feature of Cisco routers and switches that provides the ability to collect IP network traffic statistics and to export them to a collector device.
<b>Netvisor ONE</b>	Netvisor Open Networking Edition (ONE) is Pluribus' enterprise-class Network Operating System built for Open Networking hardware, which supports an extensive range of networking services: from the more basic ones such as Layer 2 and Layer 3 switching for both IPv4 and IPv6 protocols, to the more advanced ones such as data center interconnect (DCI) through VXLAN support and in-depth traffic analytics.
<b>NFS</b>	Network File System is a distributed file system protocol that enables a user on a client computer to access files over a computer network much like local storage is accessed.
<b>OSPF</b>	Open Shortest Path First is a standard routing protocol that falls into the category of interior gateway protocols (IGPs), operating within a single autonomous system.
<b>OVSDB</b>	The Open vSwitch Database Management Protocol is an SDN configuration protocol. It is used, for example, to interface with a SDN controller such as OpenDayLight or VMware NSX.
<b>Out-of-band Interface</b>	A dedicated out-of-band port on Netvisor ONE switches, used either as a management-only interface or as a fabric-control port to form the fabric and exchange fabric information over the out-of-band management network.
<b>Overlay</b>	In the VXLAN context, this term refers to all the elements built on top of the generic IP transport infrastructure in order to offer higher-level transport functionalities and services.
<b>PIM</b>	Protocol-Independent Multicast is a family of standard multicast routing protocols for IP networks that enable one-to-many and many-to-many forwarding of data over a LAN, WAN or the Internet.
<b>Quagga</b>	Quagga is a network routing software suite providing implementations of Open Shortest Path First (OSPF), Routing Information Protocol (RIP), Border Gateway Protocol (BGP) and IS-IS for Unix-like platforms.
<b>QinQ</b>	QinQ is a technique (also known as stacked VLANs, or Q-in-Q) that can apply an extra VLAN tag on top of the standard 802.1Q tag (hence the term of VLAN stacking).
<b>QoS</b>	Quality of Service refers to traffic prioritization and resource reservation control mechanisms that can provide different priorities to different applications, users, or data flows, and that can guarantee a certain level of performance to each data flow.
<b>QSFP+</b>	The Quad Small Form-factor Pluggable module is a compact, hot-

	pluggable transceiver used for data communications applications. QSFP+ is an evolution of QSFP to support four channels carrying 10 Gigabit Ethernet that can be combined to form a single 40 Gigabit Ethernet link.
<b>RA</b>	Router Advertisement is a type of ICMPv6 message used for the Neighbor Discovery (ND) process.
<b>RESTful</b>	Representational State Transfer (REST) is a software architectural style that defines a set of rules to be used for creating web services. Web services that conform to the REST architectural style are called RESTful.
<b>RIB</b>	The Routing Information Base is the IP routing table created by a switch or router by collecting routing information from multiple sources including configuration (static routes), dynamic routing protocols (RIP, OSPF, BGP), etc.
<b>RIP</b>	The Routing Information Protocol is an old distance-vector routing protocol that employs the hop count as a routing metric. It has two versions, RIPv1 and RIPv2, for IPv4 while RIPng is an extension of RIPv2 with support for IPv6.
<b>RMA</b>	A Return Merchandise Authorization is usually referred to the process of returning a product to receive a replacement or repair (and implicitly following the associated network administrator procedures).
<b>RSTP</b>	The Rapid Spanning Tree Protocol was introduced as standard IEEE 802.1w to provide significantly faster spanning tree convergence after a topology change compared to regular STP, while maintaining full backward compatibility with it.
<b>SCP</b>	The Secure Copy Protocol is a network protocol based on the BSD RCP protocol that supports secure file transfers between devices on a network. Security (authenticity and confidentiality of the data in transit) is based on the Secure Shell (SSH) protocol.
<b>SDN</b>	Software-Defined Networking is defined by the Open Networking Foundation as an emerging architecture that is dynamic, manageable, cost-effective and adaptable, making it ideal for the high-bandwidth, dynamic nature of today's applications.
<b>SFP+</b>	The enhanced Small Form-factor Pluggable module is a compact, hot-pluggable transceiver that supports data rates of up to 16 Gbit/s and is therefore used for 10 Gigabit Ethernet interfaces.
<b>SFTP</b>	The SSH File Transfer Protocol is an IETF network protocol that provides file access, file transfer and file management over any reliable data stream. It was designed as an extension of the Secure Shell protocol (SSH) version 2.0 to provide secure file transfer capabilities.
<b>SLACC</b>	IPv6 State-Less Address Auto-Configuration is the process by which IPv6 hosts can configure themselves automatically when connected to an IPv6 network using the Neighbor Discovery Protocol function via ICMPv6 router discovery messages.

<b>SNMPv3</b>	Simple Network Management Protocol is an IETF standard protocol for collecting and organizing information about managed devices on IP networks and for modifying that information to change a device's behavior. SNMPv1 is the original version of the protocol, while SNMPv2c and SNMPv3 are more recent versions that feature improvements in performance, flexibility and security.
<b>SSH</b>	Secure Shell is a cryptographic network protocol that enables network services to operate securely over an insecure network. Typical applications include remote command-line login and remote command execution.
<b>STP</b>	In a bridged network the Spanning Tree Protocol (IEEE 802.1D and IEEE 802.1Q-2014 standards) is used to turn a redundant physical topology into a loop-free, tree-like logical forwarding topology by setting one or more ports to blocking state, so as to prevent bridging loops.
<b>Syslog</b>	Syslog is a standard technology for message logging which logically separates the software that generates the messages, the system that stores them, and the software that reports them.
<b>TLS</b>	Transport Layer Security is a cryptographic protocol designed to provide communication security over a computer network with the aim of guaranteeing privacy and data integrity between two or more communicating computer applications.
<b>Traffic Flow</b>	Also known as packet flow or network flow, is a sequence of packets from a source device to a destination (a unicast destination, a multicast group, or a broadcast address).
<b>Underlay</b>	In the VXLAN context, this term refers to the generic IP transport infrastructure used to ensure IP reach-ability among all Virtual Tunnel Endpoints (VTEP) in the network that create the overlay.
<b>UNUM™</b>	Pluribus Unified Management, Automation and Analytics Platform is a multi-functional web management portal that enhances the intrinsic automation of the Adaptive Cloud Fabric architecture.
<b>vCenter Server</b>	vCenter Server is the centralized management utility for VMware. It is used to centrally manage hypervisors (ESXi), storage, virtual machines, and all dependent components (such as network and security).
<b>vFlow</b>	Pluribus' mechanism used to filter fabric-wide data center switching traffic on a granular flow level, and to apply security/QoS (Quality of Service) actions or forwarding decisions on each defined flow.
<b>VIP</b>	A Virtual IP is an IP address that does not correspond to an actual physical device but to a virtual forwarding entity (for example for redundancy purposes). In this document's context it's the IP address used by VRRP instances and by VTEPs.
<b>vLAG</b>	Virtual Link Aggregation Group is a Netvisor ONE multi-chassis link aggregation technology to bundle two or more links together when the links belong to two different chassis (behaving as a single



	virtual chassis/cluster).
<b>VLAN</b>	A Virtual LAN is a logical broadcast domain that is identified by using a specific frame tag format (defined by the IEEE 802.1Q standard) and is isolated at the data link layer in a computer network.
<b>vLE</b>	Virtual Link Extension is a Netvisor ONE technology that enables the creation of Layer 1 pseudo-wires that can emulate a direct connection between devices on top of an IP transport network.
<b>vNET</b>	A Virtual NETwork is a partition of the fabric. A vNET is defined by a group of network objects that can operate independently and have dedicated resources. This is how Netvisor ONE provides multi-tenancy support and in-depth network segmentation (beyond VLANs and VRFs).
<b>VNI</b>	In VXLAN parlance, each segment is identified through a 24-bit segment ID called the “VXLAN Network Identifier” (VNI). This allows up to 16M VXLAN segments to coexist within the same administrative domain.
<b>vPorts</b>	“Virtual ports” are software Layer 2 entries associated to all ports a Pluribus switch performs MAC address learning on.
<b>VRF</b>	Virtual Routing and Forwarding is a technology that allows multiple routing spaces to coexist on the same switch. It complements the vRouter construct, offering a highly scalable solution for multi-tenant environments.
<b>vRouter</b>	An object used to provide routing between subnets, VLANs and/or vNETs. The vRouter runs in a dedicated operating system container.
<b>VRRP</b>	Virtual Router Redundancy Protocol is a networking protocol that provides redundancy of routing paths by creation of virtual routers, which are an abstract representation of multiple routers (i.e., master and backup routers) acting as a group.
<b>VTEP</b>	A VXLAN Tunnel Endpoint is the entity responsible for encapsulating / de-encapsulating VXLAN packets.
<b>VTEP HA</b>	VTEP High Availability refers to a mechanism designed to ensure redundancy of the VTEP entity.
<b>VXLAN</b>	Virtual Extensible LAN is a standard UDP-based packet encapsulation technology defined in RFC 7348. VXLAN's Ethernet-in-UDP encapsulation is used to implement the overlaying of virtualized Layer 2 networks over Layer 3 networks.
<b>Wireshark</b>	Wireshark is a free open source packet analyzer. It is used for network troubleshooting and analysis, and for software and communications protocol development.
<b>ZTP</b>	Zero Touch Provisioning is a network device capability that enables it to be provisioned and configured automatically, reducing the overhead required for a complete network deployment.





## Entering Commands and Getting Help

Commands, options, and arguments are entered at the CLI prompt. A command name must be typed, but included command-completion and help features contribute to the command entry process.

To display a list of commands you use within a command mode, enter a question mark (?), or use the tab key, or type help at the command prompt.

You also display keywords and arguments for each command with this context-sensitive help feature.

Use the complete commands and display keywords and arguments for each command using the tab key to assist with context-sensitive command help and completion.

Table 1, lists the command that you enter to get help specific to a command, keyword, or argument.

**Table 1: Getting Help**

<code>abbreviated- command-entry?</code>	Displays a list of commands that begin with a specific character string. Do not leave a space between the string and question mark.
<code>abbreviated- command-entry &lt;tab&gt;</code>	Completes a partial command name.
<code>?</code>	Lists all commands.
<code>command ?</code>	Lists all keywords for the command. Leave a space between the command and the question mark.
<code>command keyword ?</code>	Lists all arguments for the keyword. Leave a space between the command and the question mark.

Where a text string is used, such as *name-string*, the following characters are allowed as part of the text string:

- a-z, A-Z, 0-9, \_ (underscore), . (period), , (comma), : (colon), and - (dash).

**Note:** If you enter an invalid command, using the ? and tab key has no effect and does not return any changes to the CLI.

**Note:** The CLI has an editing ability similar to UNIX and Linux functionality using emacs keys. For example, **Ap** steps backward through previous commands, **An** moves to the next command in the history, **Aa** moves to the first character in the command and **Ae** moves to the end of the line, **Au** erases the

current line, and **AW** erases the previous word. Also, you can use the up and down arrows on your keyboard to retrieve the last command entered at the CLI.

## Finding Command Options

The syntax can consist of optional or required keywords.

To display keywords for a command, enter a question mark (?) at the command prompt or after entering part of a command followed by a space.

Netvisor One CLI displays a list of available keywords along with a brief description of the keywords.

For example, if you want to see all of the keywords for the command **user**, enter **user ?**.

Table 2, displays examples of using the question mark (?) to assist you with entering commands.

**Table 2: Finding Command Options**

<i>CLI network-admin@switch &gt; ?</i>	
All commands: acl-ip-create acl-ip-delete ...	Displays a list of commands that begin with a specific character string. Do not leave a space between the string and question mark.
Switch> user auth User: <user> Password: <password>	Completes a partial command name.
?	Lists all commands.
<i>command ?</i>	Lists all keywords for the command. Leave a space between the command and the question mark.
<i>command option ?</i>	Lists all arguments for the option. Leave a space between the command and the question mark.

**Note:** Other useful options, especially for displaying statistics, include: `sort`, `interval`, `start-time`, `end-time`, `duration`, `count`, `show-interval`, and `show-diff-interval`. The commands that display the statistics have `show-diff-interval` as a command option, while other show commands have `show-interval` as a command option.

## About Alternate Command Format

---

Netvisor ONE provides with an alternate command format, where the commands start with a verb instead of a noun. This format omits the hyphen in the command names.

For example, `connection-stats-show` can also be entered as `show connection-stats`.

The command formats have the same features and can be used interchangeably.

## Specifying IP Address Netmasks

---

Some commands call for the specification of an IP address netmask. The Netvisor ONE supports both Classless Inter-Domain Routing (CIDR) and subnet notations.

For example, to specify the range of IP addresses from 192.168.0.0 to 192.168.0.255, you can express as:

- IP address: 192.168.0.0/24 or
- IP address: 192.168.0.0:: 255.255.255.0 (netmask)

Here is a sample of the CIDR to subnet notation mapping:

CIDR	Subnet Mask
/24	255.255.255.0
/25	255.255.255.128
/26	255.255.255.192

## Specifying Measurement Units

Many commands include input and output of capacity and throughput. Network values are always in bits and storage values in bytes.

Scale factors are allowed on input and displayed in output as well as shown in the following table.

**Table 3: Scale Numbers**

Scale Indicator	Meaning (Networking)	Meaning (Storage)
K or k	Kilobits	Kilobytes
M or m	Megabits	Megabytes
G or g	Gigabits	Gigabytes
T or t	Terabits	Terabytes

## Customizing Show Output Formats

The output generated by the show commands can be customized by using the optional arguments described in the following table.

**Table 4: Show Output Formats**

<code>format &lt;column_name1&gt;, &lt;column_name2&gt;, &lt;column_nameX&gt;</code>	<p>Displays only the columns matching the list of column header names.</p> <p><b>NOTE:</b> The list of column names is comma-separated without spaces.</p>
<code>format all</code>	<p>Displays all available column headers. This output is also called <b>verbose mode</b>.</p> <p>By default, show commands output a terse set of the most commonly useful column headers.</p>
<code>layout horizontal vertical</code>	<p>Specify if you want to display show output in a horizontal or vertical format. Vertical may be more useful for verbose show outputs. For example:</p> <pre>CLI (network-admin@switch) &gt; vlan- show format all layout vertical</pre>
<code>parsable-delim &lt;separator&gt;</code>	<p>Displays the output of show command by separating columns by the specified &lt;separator&gt; character(s).</p> <p>For example, <code>parsable-delim ,</code> produces a comma-separated output (CSV).</p> <p><b>NOTE:</b> If the <code>parsable-delim</code> option is specified, the column header names (titles) are suppressed from the output.</p>
<code>pager on off</code>	<p>Displays the output as pages (<code>pager on</code>) or as a single scroll-able page (<code>pager off</code>).</p>

Below are some examples of how the output for the show commands are displayed in Netvisor ONE CLI.

To display the select columns, enter the command followed by the option `format <column_name1>, <column_name2>, <column_nameX>` as below:

```
CLI (network-admin@pn-1) switch pn-1 lldp-show format switch, local-port,  
chassis-id, port-id, port-desc
```



switch	local-port	chassis-id	port-id	port-desc
pn-1	1	0e0000e8	1	spine-cluster cluster link
pn-1	2	1100013f	5	trunk from cluster-1 to spine cluster
pn-1	3	11000141	5	trunk from cluster-1 to spine cluster
pn-1	4	11000147	5	trunk from cluster-2 to spine cluster
pn-1	5	11000145	5	trunk from cluster-2 to spine cluster

To display the output on a local switch that you had logged in (the \* indicates the local switch in the example below). You don't need to specify the switch-name for all the commands that you run on the local switch.

```
CLI (network-admin@switch-1) > switch-local
```

```
CLI (network-admin@pn-sw01*) > lldp-show
```

local-port	chassis-id	port-id	port-desc	sys-name
1	0e0000e8	1	spine-cluster cluster link	pn-sw02
2	1100013f	5	trunk from cluster-1 to spine cluster	pn-sw101
3	11000141	5	trunk from cluster-1 to spine cluster	pn-sw102
13	0e000063	2	PN Switch Port(2	pn-sw106

To display all the switches in a fabric, for example, use the `switch <tab>` command:

```
CLI (network-admin@pn-sw01) > switch
```

```
pn-sw01
pn-sw104
pn-sw102
pn-sw02
leaf_clust_1      pn-sw101,pn-sw102
leaf_clust_2      pn-sw103,pn-sw104
even-cluster-nodes pn-sw102,pn-sw104,pn-sw106,pn-sw02
odd-cluster-nodes  pn-sw101,pn-sw103,pn-sw105,pn-sw01
spine_clust       pn-sw01,pn-sw02
*                 all switches in the fabric
```

To display the output in vertical format, for example, use the command:

```
CLI (network-admin@pn-sw103) > cluster-show layout vertical
```

```
switch:      pn-sw101
name:        pn-sw10102
state:       online
cluster-node-1: pn-sw101
cluster-node-2: pn-sw102
tid:         4034
mode:        master
ports:       1,125,272
```

```
remote-ports:          1,125,272
cluster-sync-timeout(ms): 2000
cluster-sync-offline-count: 3
switch:                pn-sw102
name:                  pn-sw10102
state:                 online
cluster-node-1:        pn-sw101
cluster-node-2:        pn-sw102
tid:                   4034
mode:                  slave
ports:                 1,125,272
remote-ports:          1,125,272
cluster-sync-timeout(ms): 2000
cluster-sync-offline-count: 3
```

## Specifying a Switch or Fabric for Command Scope

In Netvisor ONE Adaptive Cloud Fabric, a switch is designed as the building block of a fabric and the goal of Netvisor ONE design is to manage the fabric of switches as a single switch. Due to this capability, you can run the CLI commands on a local switch, a cluster of switches, other switches in the fabric, or the entire fabric. You do not have to login to each switch to run the commands because all the switches are part of the same fabric. By default, you can run the commands on the switch that you had logged-in.

For example, to disable a port (port 5) on the switch that you had logged in (switch-1), use the command,

```
CLI(network-admin@Switch-1) > port-config-modify port 5 disable
```

To specify a different switch for a single command, use the `switch` prefix. For example, use `switch pn-switch-2 port-config-modify port 5 enable` to enable port 5 on pn-switch-2, even if the CLI is connected to a different switch in the fabric:

```
CLI(network-admin@Switch-1) > switch pn-switch-2 port-config-modify port 5 enable
```

To specify a different switch for a series of commands, use the `switch` prefix with no command. For example,

```
CLI(network-admin@Switch-1) > switch pn-switch-2 <return>
```

```
CLI(network-admin@pn-switch-2) >
```

The CLI prompt changes to indicate that `pn-switch-2` is the switch you are executing commands. You can run other commands on `pn-switch-2` even though the switch that you are physically connected to is switch-1.

For most CLI show commands, the command displays results from all switches in the fabric by default. For example, when the CLI command `port-show` is entered on the switch, it shows the ports of all switches in the fabric.

To specify that a CLI show command should apply to a specific switch, use the switch prefix to the CLI command. For example, to view the show output of the `port-show` command of the switch named `pn-switch-1`, type the command:

```
CLI(network-admin@Switch-1) > switch pn-switch-1 port-show
```

switch	port	bezel-port	status
pn-switch-1	9	3	phy-up,host-disabled
pn-switch-1	10	3.2	phy-up,host-disabled



## Running Commands on a Local Switch

---

You can specify to run commands locally on a switch by using the `switch-local` parameter.

For instance, using `switch-local port-stats-show` displays output for the local switch ports only.

## Changing Switch Setup Parameters

---

You can modify the following switch parameters by using the `switch-setup-modify` command:

- Switch name
- Management IPv4 and IPv6 addresses
- Management IPv4 and IPv6 netmasks
- Management IPv4 and IPv6 address assignments
- In-band IPv4 address
- In-band netmask
- Gateway IPv4 address
- Gateway IPv6 address
- Primary and secondary IPv4 addresses for DNS services
- Domain name
- NTP server
- End User License Agreement (EULA) acceptance and timestamp
- Password
- Date
- Analytics store (storage type)
- Message of the Day (MOTD)
- Banner

For example if you want to change the date and time on the switch, use the command:

```
CLI (network-admin@switch-1) > switch-setup-modify date 2019-08-05  
T04:30:00
```

To view the changed date (see **bold** in output), use the show command:

```
CLI (network-admin@Leaf1) > switch-setup-show  
switch-name:          plu005sw101  
mgmt-ip:              10.80.241.235/25  
mgmt-ip-assignment:   static
```

```
mgmt-ip6:                2001:aaaa::10:80:241:235/112
mgmt-ip6-assignment:     static
in-band-ip:              192.168.241.235/24
in-band-ip6:              fe80::640e:94ff:fe3f:fafc/64
in-band-ip6-assign:      autoconf
gateway-ip:              10.80.241.129
gateway-ip6:              2001:aaaa::10:80:241:ffff
dns-ip:                   10.80.241.94
dns-secondary-ip:        10.80.241.98
domain-name:              pluribusnetworks.com
ntp-server:               10.80.241.94
timezone:                 America/Los_Angeles
date:                     2019-08-05,04:30:00
enable-host-ports:       yes
banner:                   * S2L CONVERSION SETUP ERIC05 *
```

## Creating Switch Groups

This feature allows you to create a switch group, and you can create as many switch groups as needed. You provide a name to a group of switches, and a switch can be a member of more than one group. If a switch is offline when you add it to a group, the configuration fails for that switch. Online switches are added normally.

Switch groups are static and you must manually remove a switch from a group. You cannot use a switch name for the switch group name and a warning message is displayed because the configuration is invalid.

Use the following commands:

```
CLI (network-admin@Spine1) > switch-group-create
```

---

<code>name <i>name-string</i></code>	Specify a name for the switch group.
<code>description <i>description-string</i></code>	Specify a description for the switch group.

---

To create a switch-group with the name, **rack-1-row-1**, use the following syntax:

```
CLI (network-admin@Leaf1) > switch-group-create name rack-1-row-1
description "datacenter rack 1"
```

```
CLI (network-admin@Spine1) > switch-group-delete
```

---

<code>name <i>name-string</i></code>	Specify a name for the switch group.
<code>description <i>description-string</i></code>	Specify a description for the switch group.

---

To delete a switch-group with the name, **rack-1-row-1**, use the following syntax:

```
CLI (network-admin@Leaf1) > switch-group-delete name rack-1-row-1
```

```
CLI (network-admin@Spine1) > switch-group-modify
```

---

<code>name <i>name-string</i></code>	Specify a name for the switch group.
--------------------------------------	--------------------------------------

---

To modify a switch-group with the name, **rack-1-row-1**, and change the description, use the following syntax:

```
CLI (network-admin@Leaf1) > switch-group-modify name rack-1-row-1
description datacenter
```



```
CLI (network-admin@Spine1) > switch-group-show
```

---

<code>name <i>name-string</i></code>	Displays the name of the switch group.
<code>description <i>description-string</i></code>	Displays a description of the switch group.

---

To display a switch-group with the name, **rack-1-row-1**, use the following syntax:

```
CLI (network-admin@Leaf1) > switch-group-show name rack-1-row-1
```

```
name          description
-----
rack-1-row-1  datacenter
```

## Adding Switches to Switch-Groups

```
(CLI (network-admin@Spine1) > switch-group-member-add
```

---

<code>name <i>name-string</i></code>	Specify the name of the switch group to add the member.
<code>member <i>fabric-node name</i></code>	Specify the name of the switch to add as a member.

---

To add switch, Leaf-1, to switch-group, **rack-1-row-1**, use the following syntax:

```
CLI (network-admin@Leaf1) > switch-group-member-add name rack-1-row-1
member Leaf-1
```

```
CLI (network-admin@Spine1) > switch-group-member-remove
```

---

<code>name <i>name-string</i></code>	Specify the name of the switch group to remove the member.
<code>member <i>fabric-node name</i></code>	Specify the name of the switch to remove as a member.

---

To remove switch, Leaf-1, from switch-group, **rack-1-row-1**, use the following syntax:

```
CLI (network-admin@Leaf1) > switch-group-member-remove name rack-1-row-1
member Leaf-1
```

```
CLI (network-admin@Spine1) > switch-group-member-show
```

---

<code>name <i>name-string</i></code>	Displays the name of the switch group.
<code>member <i>fabric-node name</i></code>	Displays the name of the switches in a group.

---

To display switch-group, use the following syntax:

```
CLI (network-admin@Leaf1) > switch-group-member-show
```

```
switch      name                member
-----
Spine-1    rack-1-row-1 Leaf-1
```

## Support for Enabling or Disabling LLDP

This feature provides for a generic LLDP ON/OFF toggle function set at the system level.

Currently, to disable LLDP on a switch you must disable the LLDP configuration on all ports. This resets all related configurations of LLDP protocol setting and LLDP vFlows.

Use the following CLI command to enable and disable the protocol:

```
CLI (network-admin@Leaf1) > system-settings-modify [lldp|no-lldp]
```

LLDP packets are executed on the CPU with the help of LLDP vFlows.

To clear all LLDP protocol system flows use the parameter `no-lldp`.

To add all LLDP protocol system flows use the parameter `lldp`.

This approach ensures port LLDP configurations are not disturbed.

```
CLI (network-admin@Leaf1) > system-settings-show
```

```
switch:                               Spine1
optimize-arps:                        on
lldp:                                 on
```

## About GREP Support with Netvisor ONE OS

---

Netvisor ONE supports filtering output and allows switch administrators to filter output “grep |” from the CLI. This functionality is limited to the following commands:

- `running-config-show`
- `tech-support-show`
- `help`

CLI > `help | grep "openstack"` lists all of the commands for OpenStack

## Installing Netvisor ONE and Initial Configuration

---

This section contains information about initial configuration of your switch as well as commands to manage, upgrade, and restore Netvisor ONE configurations.

---

- [Changes to the End User License Agreement \(EULA\)](#)
  - [Adding License Keys to Netvisor One](#)
  - [Using the Serial Console Port for Initial Configuration](#)
  - [Managing Netvisor ONE Certificates](#)
  - [Enabling Administrative Services](#)
  - [Configuring Administrative Session Timeout](#)
  - [Confirming Connectivity on the Network](#)
  - [Setting the Date and Time](#)
  - [Viewing User Sessions on a Switch](#)
  - [Archiving Log Files Outside the Switch](#)
  - [Exporting Configurations Using Secure Copy Protocol \(SCP\)](#)
  - [Displaying and Managing Boot Environment Information](#)
  - [Upgrading the Netvisor ONE Software](#)
  - [Auto-configuration of IPv6 Addresses on the Management Interface Support](#)
  - [Support for Local Loopback IP Addresses](#)
  - [Running Shell Commands or Scripts Using REST API](#)
  - [Managing RMAs for Switches](#)
  - [Contacting Technical Assistance](#)
-

## Changes to the End User License Agreement (EULA)

---

Currently, the Netvisor One EULA is displayed when the switch is setup.

Netvisor OS Command Line Interface 5.1.0

By ANSWERING "YES" TO THIS PROMPT YOU ACKNOWLEDGE THAT YOU HAVE READ THE TERMS OF THE PLURIBUS NETWORKS END USER LICENSE AGREEMENT (EULA) AND AGREE TO THEM. [YES | NO | EULA]?: yes

If you enter the EULA option, the output displays the complete EULA text. After this action, it is not possible to confirm EULA acceptance again. In some cases, an integrator may have accepted the EULA on behalf of the actual end user.

A new command is now available to display the EULA acceptance with a timestamp of the event:

```
CLI (network-admin@pn-sw-01) > eula-show
End User License Agreement
Pluribus Networks, Inc.'s ("Pluribus", "we", or "us") software products are
designed to provide fabric networking and analytics solutions that simplify
operations, reduce operating expenses, and introduce applications online
more rapidly. Before you download and/or use any
  of our software, whether alone or as loaded on a piece of equipment, you
will need to agree to the terms of this End User License Agreement (this
"Agreement").
...
PN EULA v. 2.1

eula-show: No fabric
eula-show: Fabric required. Please use fabric-create/join/show
CLI (network-admin@pn-sw-01) >
```

## Adding License Keys to Netvisor ONE

---

Netvisor ONE binds the license key to the serial number of the switch and when downloading the Netvisor ONE software, the Pluribus Networks Cloud locates the serial number.

To install the license key, use the following syntax:

```
CLI (network-admin@switch) > software-license-install key license-key
```

The license key has the format of four words separated by commas. For example,

```
License Key: rental,deer,sonic,solace
```

Once the license key is installed, you can display information about the key using the following command:

```
CLI (network-admin@Leaf1) > software-license-show
```

```
switch:          T6001-ON
license-id:      NVOS-CLD-LIC-60D
description:     Pluribus Open Netvisor OS Linux Cloud Edition License
expires-on:      never
status:          VALID
```

## Using the Serial Console Port for Initial Configuration

This procedure assumes that you have installed the switch in the desired location and it is powered on.

**Warning:** Do not connect any ports to the network until the switch is configured. You can accidentally create loops or cause IP address conflicts on the network.

If you are going to cable host computers to the switch, there is an option to enable or disable host ports by default.

1. Connect the console port on the rear or front (depending on the model) of the switch to your laptop or terminal concentrator using a serial cable.
2. From the terminal emulator application on your computer, log into the switch with the username **network-admin** and the default password **admin**.

**Note:** Netvisor ONE supports both IPv4 and IPv6 addresses for the in-band interface.

**Warning:** Be sure to type in a static IP address for the management interface during the initial configuration. Netvisor One initially uses DHCP to obtain an IP address, but DHCP is not supported after the initial configuration.

3. Begin the initial configuration using the initialization procedure displayed.
4. Enter the following details when prompted, an example is provided in the output below:
  - Accept the EULA agreement
  - Type-in the switch name. An example is provided in the output below.
  - Enter and re-enter the password
  - Enter the Management IP and netmask. An example is provided in the output below.
  - Enter the In-band IP and netmask. An example is provided in the output below.
  - Enter the IP address of the Gateway.
  - Enter the IP address for the primary and secondary DNS.
  - Enter the domain name.

```
switch console login: network-admin
Password: admin
```

```
Netvisor OS Command Line Interface 5.1.0
```

```
By ANSWERING "YES" TO THIS PROMPT YOU ACKNOWLEDGE THAT YOU HAVE READ THE
TERMS OF THE PLURIBUS NETWORKS END USER LICENSE AGREEMENT (EULA) AND AGREE
TO THEM. [YES | NO | EULA]?: yes
```

```
Switch setup required:
```

```
Switch Name (netvisor): pn-switch-01
```

```
network-admin Password: password <return>
```



```

Re-enter Password: ***** <return>
Mgmt IP/Netmask (dhcp): 10.14.2.42/23
Mgmt IPv6/Netmask:
In-band IP/Netmask: 12.1.165.21/24
In-band IPv6/Netmask:
Loopback IP:
Loopback IPv6:
Gateway IP (10.14.2.1):
Gateway IPv6:
Primary DNS IP: 10.135.2.13
Secondary DNS IP: 10.20.4.1
Domain name: pluribusnetworks.com
Automatically Upload Diagnostics (yes):
Enable host ports by default (yes):
  
```

```

nvOS system info:
  serial number: 1918PN8500165
  hostid:                                     b001720
  device id:                                561TG02
  
```

```

Switch Setup:
Switch Name           : pn-switch-01
Switch Mgmt IP        : 10.14.2.42/23
Switch Mgmt IPv6       : fe80::4e76:25ff:feef:5140/64
Switch In-band IP     : 12.1.165.21/24
Switch In-band IPv6   : fe80::640e:94ff:fe20:8787/64
Switch Loopback IP    : ::
Switch Loopback IPv6  : ::
Switch Gateway        : 10.14.2.1
Switch IPv6 Gateway   : ::
Switch DNS Server     : 10.135.2.13
Switch DNS2 Server    : 10.20.4.1
Switch Domain Name    : pluribusnetworks.com
Switch NTP Server     :
Switch Timezone       : America/Los_Angeles
Switch Date           : 2019-09-20,11:30:49
Enable host ports     : yes
Analytics Store       : default
Fabric required. Please use fabric-create/join/show
Connected to Switch pn-switch-01; nvOS Identifier:0xb001720; Ver: 5.1.0-5010014980
  
```

When you setup a switch for initial configuration, the host facing ports are enabled by default. However, you can disable the host ports until you are ready to plug-in host cables to the switch. If Netvisor ONE does not detect adjacency on a port during the quickstart procedure, the ports remain in the disabled state.

To enable the ports after plugging in cables, use the `port-config-modify port port-list host-enable` command. Netvisor ONE enables host ports by default unless you specify NO during the quickstart procedure as displayed below.

Netvisor OS Command Line Interface 5.1.0

By ANSWERING "YES" TO THIS PROMPT YOU ACKNOWLEDGE THAT YOU HAVE READ THE

TERMS OF THE PLURIBUS NETWORKS END USER LICENSE AGREEMENT (EULA) AND AGREE TO THEM. [YES | NO | EULA]?: yes

Switch setup required:

Switch Name (netvisor): pn-switch-01

network-admin Password: password <return>

Re-enter Password: \*\*\*\*\* <return>

Mgmt IP/Netmask (dhcp): 10.14.2.42/23

Mgmt IPv6/Netmask:

In-band IP/Netmask: 12.1.165.21/24

In-band IPv6/Netmask:

Loopback IP:

Loopback IPv6:

Gateway IP (10.14.2.1):

Gateway IPv6:

Primary DNS IP: 10.135.2.13

Secondary DNS IP: 10.20.4.1

Domain name: pluribusnetworks.com

Automatically Upload Diagnostics (yes):

Enable host ports by default (yes): no

To verify, use the command:

```
CLI (network-admin@pn-switch-01) > port-show port 9,10
```

switch	port	bezel-port	status	config
pn-switch-01	9	3	phy-up,host-disabled	10g
pn-switch-01	10	3.2	phy-up,host-disabled	10g

To enable the port (s), use the command:

```
CLI (network-admin@pn-switch-01) > port-config-modify port 9,10 enable
host-enable
```

```
CLI (network-admin@pn-switch-01) > port-show port 9,10
```

switch	port	bezel-port	status	config
pn-switch-01	9	3	up,vlan-up	fd,10g
pn-switch-01	10	3.2	up,vlan-up	fd,10g

You cannot change (enable or disable) the host-ports by using the switch setup process after the initial configuration is done. If you try to modify the host-ports, Netvisor ONE displays an error as displayed in the example here:

```
CLI (network-admin@pn-switch-01) > switch-setup-modify disable-host-ports
```

```
switch-setup-modify: disable/enable host ports can be set only at initial
switch-setup time
```

During the initial configuration of the switch, if the host ports are disabled, then all ports having the same port configuration will be disabled. This can be viewed using the following command:

```
<CLI (network-admin@pn-switch-01) > port-config-show port port-list host-  
enable
```

In this mode, when any port comes up physically, Netvisor ONE automatically sends and receives LLDP packets to look for peer switches. If Netvisor ONE does not detect an adjacency within 5 seconds, the port is flagged as `host-disabled`. With this flag set, Netvisor ONE only accepts LLDP packets and does not initiate packet transmission.

```
CLI (network-admin@pn-switch-01) > port-config-show port 9,10
```

switch	port	bezel-port	status	config
pn-switch-01	9	3	up,vlan-up,PN-other,LLDP	fd,10g
pn-switch-01	10	3.2	up,vlan-up	fd,10g

After completing switch discovery and fabric creation, use the `host-enable` option to enable host, server, or router traffic switching, and ports:

```
CLI (network-admin@pn-switch-01) > port-config-modify port 9 host-enable
```

## Managing Netvisor ONE Certificates

Pluribus Networks includes the Netvisor ONE certificates along with the switches during shipment and you can access the certificates from `/var/nvos/certs` directory. These certificates are necessary for communication between switches in a fabric and hinders the transactions between fabric members if the certificate expires. You can view the validity (dates valid from and dates valid until) for Netvisor ONE certificate using the `switch-info-show` command.

When you configure the alarm, the certificate is checked every 24 hours and an alarm is issued if the number of days of expiry is equal to or less than 30 days . The certificate expiry alert is enabled by default for 30 days, but can configured between 7 days through 180 days on Netvisor ONE. You can disable this feature using the `cert-expiration-alert-modify no-netvisor` command.

You can view the certificate expiration alert or alarm configuration by using the `cert-expiration-alert-show` command and can schedule an alert notification before the certificate expires. You can view the alarm or alert notification in the `event.log` file and also by running the `log-alert-show` command . You can also configure a new SNMP trap for certificate expiry on the SNMP services.

Alarm is an event in the *event log*, an alert in `log-alert-show` command and a new SNMP trap if the trap server is configured. Frequency of alarm will be every 24 hours until the certificate has expired.

To configure the certificate expiry alert, use the command:

```
CLI (network-admin@switch01) > cert-expiration-alert-modify
```

Specify one or more of the following options:	
<code>netvisor no-netvisor</code>	Specify whether to enable or disable Netvisor ONE certificate expiration alerts.
<code>days-before-expiration 7..180</code>	Modify the number of days before expiration to send alerts (Default 30 days). The value ranges from 7 through 180 days.

To view the alert configuration for the certificate expiry, use the command:

```
CLI (network-admin@switch01) > cert-expiration-alert-show
```

```
switch:                switch01
days-before-expiration(d): 30
```

To enable or disable the SNMP trap for certificate expiry alert, use the command:

```
CLI (network-admin@switch01) > snmp-trap-enable-modify cert-expiry|no-cert-expiry
where,
```

---

`cert-expiry|no-cert-expiry`

---

Specify whether to monitor certificate expiry or not.

---

To view the alert configuration details older than an hour, use the command:

```
CLI (network-admin@switch01) > log-alert-show older-than 1h
```

time	switch	code	name	count	last-message
00:17:05	switch01	31008	smf_nvOSd_stop	1	SMF Service stopping nvOSd
00:17:08	switch01	11008	nvOSd_start	1	version 5.1.5010014665
00:35:49	switch01	31016	certificate_expiry	1	<b>switch cert expiring in 19 days</b>

The `switch-info-show` command displays the validity (dates valid from and dates valid until) for Netvisor ONE certificate. For example,

```
CLI (network-admin@nru03-sw-1*) > switch-info-show
```

```
model: NRU03
chassis-serial: 1937ST9100075
cpu1-type: Intel(R) Xeon(R) CPU D-1557 @ 1.50GHz
cpu2-type: Intel(R) Xeon(R) CPU D-1557 @ 1.50GHz
cpu3-type: Intel(R) Xeon(R) CPU D-1557 @ 1.50GHz
cpu4-type: Intel(R) Xeon(R) CPU D-1557 @ 1.50GHz
system-mem: 30.6G
switch-device: OK
fan1-status: OK
fan2-status: OK
fan3-status: OK
fan4-status: OK
fan5-status: OK
fan6-status: OK
fan7-status: OK
fan8-status: OK
fan9-status: OK
fan10-status: OK
fan11-status: OK
fan12-status: OK
ps1-status: OK
ps2-status: OK
disk-model: Micron_1300_MTFDDAV256TDL
disk-firmware: M5MU000
disk-size: 238G
disk-type: Solid State Disk, TRIM Supported
bios-vendor: American Megatrends Inc.
bios-version: 1.00.00
netvisor-cert-valid-from: Sep 13 07:00:00 2019 GMT
netvisor-cert-valid-till: Sep 14 06:59:59 2039 GMT
```



## Enabling Administrative Services

---

There are many features of the Pluribus Networks fabric that require or can be enhanced using remote access. For example, when packets are written to a log file, you may want to transfer that file from a switch to a different system for analysis. Also, if you are creating a NetVM environment, an IOS image of the guest OS must be loaded on the switch.

You can enhance or modify several services such as SSH, NFS, Web, SNMP, SFTP.

To check the status of various services, use the following command:

```
CLI (network-admin@Leaf-1) > admin-service-show
```

```
switch:          Leaf-1
if:              mgmt
ssh:             on
nfs:             on
web:             on
web-ssl:         off
web-ssl-port:    443
web-port:        80
web-log:         off
snmp:            on
net-api:         on
icmp:            on
```

```
switch:          Leaf-1
if:              data
ssh:             on
nfs:             on
web:             on
web-ssl:         off
web-ssl-port:    443
web-port:        80
web-log:         off
snmp:            on
net-api:         on
icmp:            on
```

Netvisor ONE supports the file transfer method, SFTP and SFTP is enabled by default on Netvisor ONE. Because SFTP relies on Secure Shell (SSH), you must enable SSH before enabling SFTP.

To enable SSH, use the following command

```
CLI (network-admin@Leaf1) > admin-service-modify nic mgmt ssh
```

To enable SFTP, use the following command:

```
CLI (network-admin@Leaf1) > admin-sftp-modify enable
```

```
sftp password: <password>
confirm sftp password: <password>
```

The default SFTP username is sftp and the password can be changed using the admin-sftp-modify command:

```
CLI (network-admin@Leaf1) > admin-sftp-modify
```

```
sftp password: <password>
confirm sftp password: <password>
```

To display the details, use the following commands:

```
CLI (network-admin@Leaf-1) > admin-service-show
```

switch	if	ssh	nfs	web	web-ssl	web-ssl-port	web-port	snmp	net-api	icmp
Leaf-1	mgmt	on	off	off	off	443	80	on	off	on
Leaf-1	data	on	off	off	off	443	80	on	off	on

admin-service-show: Fabric required. Please use fabric-create/join/show

```
CLI (network-admin@Leaf1) > admin-sftp-show
```

```
switch:      Leaf1
sftp-user:   sftp
enable:      yes
```

Use SFTP from a host to the switch, and login with the username **sftp** and the password configured for SFTP. Then you can download the available files or upload files to the switch.

```
CLI (network-admin@Leaf1) > admin-service-show
```

switch	nic	ssh	nfs	web	web-port	snmp	net-api	icmp
Leaf1	mgmt	on	off	on	80	off	on	on



## Configuring Administrative Session Timeout

---

Netvisor ONE sets the administrator sessions to timeout after 60 minutes (by default) of idle time or no activity, but allows you to change the timeout value to a user desirable time. During the session timeout, you are logged out of the CLI and the Shell prompt and your privileges changes to *root user*. To access the switch, you must login again using the CLI or Shell prompt.

To verify the default or user configured session timeout value, use the command:

```
CLI (network-admin@Spine1) > admin-session-timeout-show
```

```
switch: Spine1  
timeout: 1h
```

To modify the timeout value, use the command:

```
CLI (network-admin@Spine1) > admin-session-timeout-modify
```

---

`timeout duration: #d#h#m#s`

Specify the maximum time to wait for user inactivity before terminating login session.

---

## Confirming Connectivity on the Network

---

After connecting your switch, take the time to ensure connectivity by pinging an external IP address (supports both IPv4 and IPv6), and pinging a domain to ensure domain name resolution.

To ping the external network from the switch, use the `ping` command:

```
CLI (network-admin@switch) > ping 2010::2
```

```
PING 2010::2(2010::2) 56 data bytes
64 bytes from 2010::2: icmp_seq=1 ttl=64 time=1.69 ms
64 bytes from 2010::2: icmp_seq=2 ttl=64 time=0.412 ms
64 bytes from 2010::2: icmp_seq=3 ttl=64 time=0.434 ms
64 bytes from 2010::2: icmp_seq=4 ttl=64 time=0.418 ms
```

To ping an IP address from the switch, use the `ping` command:

```
CLI (network-admin@switch) > ping 98.138.253.109 : 56 data bytes
```

```
PING 98.138.253.109 (98.138.253.109) 56(84) bytes of data.
64 bytes from 98.138.253.109: icmp_seq=1 ttl=47 time=51.8 ms
64 bytes from 98.138.253.109: icmp_seq=2 ttl=47 time=51.9 ms
64 bytes from 98.138.253.109: icmp_seq=3 ttl=47 time=53.6 ms
```

To ping a domain, use the `ping` command again:

```
CLI (network-admin@Leaf1) > ping yahoo.com
```

```
PING yahoo.com (98.138.253.109) 56(84) bytes of data.
64 bytes from irl.fp.vip.net.yahoo.com (98.138.253.109): icmp_seq=1 ttl=47 time=52.2 ms
64 bytes from irl.fp.vip.net.yahoo.com (98.138.253.109): icmp_seq=2 ttl=47 time=52.5 ms
64 bytes from irl.fp.vip.net.yahoo.com (98.138.253.109): icmp_seq=3 ttl=47 time=51.9 ms
64 bytes from irl.fp.vip.net.yahoo.com (98.138.253.109): icmp_seq=4 ttl=47 time=51.8 ms
```

## Setting the Date and Time

You can set the date and time on a switch by modifying the switch configuration using the `switch-setup-modify` command. For example, to change the date and time to September 24, 2019, 09:30:00, use the following command syntax:

```
CLI (network-admin@Leaf1) > switch-setup-modify date 2019-09-24 T09:30:00
```

To display the configured setting, use the `switch-setup-show` command:

```
CLI (network-admin@Leaf2) > switch-setup-show
switch-name:                Leaf2
mgmt-ip:                    10.14.30.18/23
mgmt-ip-assignment:         static
mgmt-ip6:                   2721::3617:ebff:fef7:94c4/64
mgmt-ip6-assignment:        autoconf
mgmt-link-state:            up
mgmt-link-speed:            1g
in-band-ip:                 192.168.101.7/24
in-band-ip6:                fe80::640e:94ff:fe83:cefa/64
in-band-ip6-assign:         autoconf
gateway-ip:                 10.14.30.1
dns-ip:                     10.20.4.1
dns-secondary-ip:           172.16.1.4
domain-name:                pluribusnetworks.com
ntp-server:                 0.us.pool.ntp.org
ntp-secondary-server:       0.ubuntu.pool.ntp.org
timezone:                   America/Los_Angeles
date:                       2019-09-24,09:30:00
hostid:                     184555395
location-id:                7
enable-host-ports:          yes
banner:                     * Welcome to Pluribus Networks Inc. Netvisor(R).
This is a monitored system. *
device-id:                  1WDQX42
banner:                     *
                                ACCESS RESTRICTED TO AUTHORIZED
USERS ONLY                  *
banner:                     * By using the Netvisor(R) CLI,you agree to the
terms of the Pluribus Networks *
banner:                     * End User License Agreement (EULA). The EULA
can be accessed via         *
banner:                     * http://www.pluribusnetworks.com/eula or by
using the command "eula-show" *
```

## Changing the Default Timezone

By default, Netvisor sets the default timezone to US/Pacific Standard Time (PST).

To change the timezone, use the switch-setup-modify command:

```
CLI (network-admin@Leaf1) > switch-setup-modify timezone time-zone name
```

## Viewing User Sessions on a Switch

Netvisor ONE enables you to view the user sessions on a specified switch and displays all currently logged-in users along with the IP address of the user and login time when you run the command, `mgmt-session-show`. This information is useful for troubleshooting purposes or while dealing on issues with Pluribus Customer Support teams.

```
CLI (network-admin@Leaf1) > mgmt-session-show
```

Specify any of the following:

<code>user <i>user-string</i></code>	Displays the user name.
<code>cli-user <i>cli-user-string</i></code>	Displays the name used to log into the switch.
<code>pid <i>pid-number</i></code>	Displays the process ID.
<code>terminal <i>terminal-string</i></code>	Displays the terminal ID
<code>from-ip <i>ip-address</i></code>	Displays the IP address of the user.
<code>login-time date/time: <i>yyyy-mm-ddTHH:mm:ss</i></code>	Displays the time and date that the user logged into the switch.
<code>remote-node <i>remote-node-string</i></code>	Displays the name of the remote node.
<code>vnet <i>vnet-string</i></code>	Displays the vNET assigned to the user.
<code>type <i>cli api shell</i></code>	Displays the type of login session.

For example,

```
CLI (network-admin@Leaf1) > mgmt-session-show
```

user	cli-user	pid	terminal	from-ip	login-time	type
----	-----	---	-----	-----	-----	----
admin	network-admin	13805	pts/3	10.60.1.216	11:20:52	cli
root	network-admin	8589	pts/2	10.14.20.109	11-15,17:16:17	cli
	network-admin				08:24:10	cli
root		19139	pts/1	10.14.22.54	11-15,11:01:08	shell

In this example, the `root` user represents the user who has all access by default, while the `admin` user has only customized access privileges.

## Archiving Netvisor ONE Log Files Outside the Switch

Netvisor ONE enables you to archive the nvOSd log files to an external file server periodically and these log files may be helpful for troubleshooting purposes. As a network administrator, you can configure the following parameters to enable archiving of the log files:

- Server IP address and hostname
- Username and password
- Log archival interval (minimum interval is 30 minutes and the default value is 24 hours)

On configuring this feature, the log archival configuration parameters are saved in the `log_archival_config.xml` file with an encrypted password string. A binary file deciphers the configuration parameters and also the files that are to be archived. Netvisor ONE sends an empty *time-stamped directory* to the configured remote server path and subsequently, all the log files are archived to the newly created remote directory. The new directory in the remote server is created in the `nvOS_archive.yyyymmdd_hh.mm.ss` format.

Netvisor ONE uses the Secure Copy Protocol (SCP) to archive the log files from the switch to the remote external server at specific intervals. Using the `enable` or `disable` parameter in the CLI command, you can start or stop archiving of the log files. You can archive regular log files, a set pattern of log files, or a whole directory from one of the following paths only. If you add files from other paths than the directories specified here, Netvisor ONE displays an error.

- `/var/nvOS/log/*`
- `/nvOS/log/*`
- `/var/log/*`

Use the below CLI commands to configure the log archival parameters and schedule the archival interval.

To modify the archival schedule parameters for the log files, use the command,

```
CLI (network-admin@switch-1) > log-archival-schedule-modify
```

<code>enable disable</code>	Specify to enable or disable the log archival schedule.
Specify one or many of the following options:	
<code>archive-server-username &lt;string&gt;</code>	Specify the SCP username of log archival server.
<code>archive-server &lt;string&gt;</code>	Specify the IP address or hostname of the log archival server.
<code>archive-server-path &lt;string&gt;</code>	Specify the SCP server path to archive the log files in.
<code>archive-interval &lt;30..4294967295&gt;</code>	Specify the log archival interval in minutes. The

	range varies from 30 minutes to 4294967295 minutes with a default value of 1440 minutes (one day).
<code>archive-server-password &lt;string&gt;</code>	Enter the SCP server password.

For example, if you had modified the log-archival-schedule by specifying the archive-server-username as `pn-user`, archive-server as `pn-server`, and archive-server-path as `/home/pn-server/workspace/log_archival_tests`, use the command,

```
CLI (network-admin@switch-1) > log-archival-schedule-modify archive-server-username pn-user archive-server pn-server archive-server-path /home/pn-server/workspace/log_archival_tests
```

To display the modified configuration, use the command,

```
CLI (network-admin@switch-1) > log-archival-schedule-show
```

```
switch:                switch-1
archive-server-username: pn-user
archive-server:         pn-server
archive-server-path:    /home/pn-server/workspace/log_archival_tests
enable:                 no
archive-interval(m):    1440
```

To add the log files to the archival list, use the command,

```
CLI (network-admin@switch-1) > log-archival-schedule-files-add log-file log-file-string
```

<code>log-file log-file-string</code>	Specify a comma-separated list of log file names to add to the archive list.
---------------------------------------	--

For example, to add the `nvOSd.log` or `audit.log` or all `log` files or a whole directory, use the following commands:

```
CLI (network-admin@switch-1) > log-archival-schedule-files-add log-file /var/nvOS/log/nvOSd.log
```

```
CLI (network-admin@switch-1) > log-archival-schedule-files-add log-file /var/nvOS/log/*.log
```

```
CLI (network-admin@switch-1) > log-archival-schedule-files-add log-file /var/log
```

```
CLI (network-admin@switch-1) > log-archival-schedule-files-add log-file /nvOS/log/audit.log
```

To view the list of log files that you had scheduled to be archived, use the command,

```
CLI (network-admin@switch-1) > log-archival-schedule-files-show
```

```
/var/nvOS/log/nvOSd.log  
/var/nvOS/log/*.log  
/var/log  
/nvOS/log/audit.log
```

If you try to add an unsupported file or directory, an error message is displayed. For example,

```
CLI (network-admin@switch-1) > log-archival-schedule-files-add log-  
file /var/nvOS
```

```
/var/nvOS, not from valid logs supported
```

To remove the log files or a list of log files from the archival list, use the command,

```
CLI (network-admin@switch-1) > log-archival-schedule-files-remove log-file  
log-file-string
```

## Guidelines and Tips

- When the `log-archival-schedule` is enabled and if all files are removed from the archival list, the `log-archival-schedule` gets disabled.
- If the `systemd` timer expires before the previous `log-archival` process is finished, then the `systemd` waits for the process to complete before starting the new process.



## Displaying and Managing Boot Environment (BE) Information

---

Netvisor ONE provides two boot environments (BEs): the current boot environment, and the previous boot environment. Having the two BEs allows you to rollback or rollforward the software versions or configurations.

To display boot environment information, use the following command:

```
CLI (network-admin@Leaf1) > bootenv-show
```

name	version	current	reboot	space	created	apply-current-config
netvisor-1	3.1.1-13800	no	no	0	08-29,14:13:35	false
netvisor-2	5.0.0-14540	yes	yes	0	08-29,17:24:17	false

To reset the boot environment and reboot using the previous environment, use the following syntax:

```
CLI (network-admin@Leaf1) > bootenv-activate-and-reboot name netvisor-1
```

To delete a boot environment, use the following syntax:

```
CLI (network-admin@Leaf1) > bootenv-delete name netvisor-2
```

You can display information about different boot environments on the switch.

## Exporting Configurations Using Secure Copy Protocol (SCP)

---

The SCP is a network protocol based on the BSD RCP protocol supporting file transfers between hosts on a network. SCP uses Secure Shell (SSH) for data transfer and uses the same mechanisms for authentication, and ensures the authenticity and confidentiality of the data in transit. A client can upload files to a server, optionally including basic attributes such as permissions or timestamps. Clients can also download files or directories from a server. SCP runs over TCP port 22 by default. Like RCP, there is no RFC that defines the specifics of the protocol.

In Netvisor One, you are prompted for a password when the `upload-server` option is provided in the CLI.

```
CLI (network-admin@Leaf1) > switch-config-export upload-server 10.1.1.1
server password:
```

During the software upgrade process, Netvisor One exports the switch configuration and moves it to a shared directory. The exported configuration archive is accessible from all boot environments. Netvisor One exports the configuration before the start of the software upgrade. Netvisor One stores a maximum of three configuration archives on the switch. Older configurations are deleted.

New parameters in Netvisor One support this feature:

```
CLI (network-admin@Leaf1) > switch-config-export
export-file                export a nvOS config file
upload-server              upload config file to server via scp
```

```
CLI (network-admin@Leaf1) > switch-config-export
```

---

Specify any of the following options:

<code>export-file switch-config export-file</code>	Exports the specified nvOS configuration file.
<code>upload-server upload-server-string</code>	Uploads the config file to server via SCP

---

For example,

```
CLI (network-admin@Leaf1) > switch-config-export upgrade-location-mappings
0xb001a48=1 export-file switch-config-reset-backup.2019-10-
15T02.12.40.tar.gz upload-server root@ursa-scale-leaf1:/root
server password:
Uploaded configuration to server at /root
CLI (network-admin@Leaf1) >
```

```
CLI (network-admin@aries-test-1) > switch-config-export export-file switch-
config-reset-backup.2019-10-15T02.12.40.tar.gz upload-server root@ursa-
```

```
scale-leaf1:/root
server password:
Uploaded configuration to server at /root
CLI (network-admin@aries-test-1) >
```

## Upgrading the Netvisor ONE Software

Software upgrades are a routine maintenance procedure that must be completed every so often. However, there are some variables to consider before you start the upgrade procedures.

Before you start the upgrade process, obtain the required upgrade software. You can download the software manually and copy it to a switch before beginning the upgrade procedures.

Software and fabric upgrades are comprised of two distinct phases: the installation (upgrade) of the new software and a switch reboot to activate the new software

To start a software upgrade:

- View the current version of Netvisor ONE on the switch by using command:

```
CLI network-admin@switch > software-show
version: 5.1.1-5010115300
```

- Identify the software package. Depending on the version of Netvisor ONE running on your switch, you should use appropriate software upgrade package.

Select the appropriate upgrade bundle based from the following upgrade matrix table:

Current Software	Target Release for Upgrade	Upgrade Type	Upgrade Software Package
3.x GA	5.2.1 GA	Software Upgrade using release upgrade bundle	nvOS-relupg-5.2.1-5020115690-onvl.pkg
5.x.x	5.2.1 GA	Software Upgrade using regular offline upgrade bundle	nvOS-5.2.1-5020115690-onvl.pkg

- Copy the upgrade package to the switch:
  1. Enable Secure File Transfare Protocol (SFTP) on the switch:

```
CLI (network-admin@switch1)>admin-sftp-modify enable
sftp password:
confirm sftp password:
CLI (network-admin@switch1)>
```

2. Upload the Software package to the switch:

```
root@server-os-9:~/# sftp sftp@switch1
The authenticity of host 'switch1 (10.0.0.02)' can't be established.
```

```
RSA key fingerprint is
SHA256:SI8VQZgJCpbbrF4sRcby36Fx7rz3Hh5EJl1PPyScLZU.
Are you sure you want to continue connecting (yes/no)? yes
Warning: Permanently added 'switch1, 10.0.0.02 (RSA)' to the list of
known hosts.
* Welcome to Pluribus Networks Inc. Netvisor(R). This is a monitored
system. *
* ACCESS RESTRICTED TO AUTHORIZED USERS ONLY *
* By using the Netvisor(R) CLI, you agree to the terms of the
Pluribus Networks *
* End User License Agreement (EULA). The EULA can be accessed via *
* http://www.pluribusnetworks.com/eula or by using the command
"eula-show" *
Password:
Connected to switch1
sftp> cd import
sftp> put nvOS-5.2.1-5020115690-onvl.pkg
Uploading nvOS-5.2.1-5020115690-onvl.pkg
nvOS-5.2.1-5020115690-onvl.pkg
nvOS-5.2.1-5020115690-onvl.pkg 100% 1870MB 7.5MB/s 04:00
```

- Start the upgrade process by using the `software-upgrade` command with package parameter, which allows you to specify the name of the upgrade file. Netvisor does not support fabric upgrade for Netvisor 5.1.1. Each switch in fabric must be upgraded individually.

```
CLI (network-admin@Leaf1) > software-upgrade package nvOS-5.2.1-5020115690-
onvl.pkg
Scheduled background update. Use software-upgrade-status-show to check.
Switch will
reboot itself. DO NOT reboot manually.
```

---

**Caution:** Do not reboot or power off any switch during the upgrade procedure. When software upgrade is complete, switch reboots automatically.

---

- Monitor the upgrade process using the `software-upgrade-status-show` command:

```
CLI (network-admin@Leaf1)>software-upgrade-status-show show-interval5

[Jun04.21:30:41] Starting software upgrade ...
[Jun04.21:30:42] Cleaning old package bundles
[Jun04.21:30:42] Checking available disk space...
[Jun04.21:30:42] Avbl free space: 12.69G, Required: 0.62G
[Jun04.21:30:42] Unpacking local package bundle...
[Jun04.21:30:42] Extracting initial bundle.
.
.
.
```

log

```
-----
[Jun04.21:30:41] Starting software upgrade ...
[Jun04.21:30:42] Cleaning old package bundles
[Jun04.21:30:42] Checking available disk space...
[Jun04.21:30:42] Avbl free space: 12.69G, Required: 0.62G
[Jun04.21:30:42] Unpacking local package bundle...
[Jun04.21:30:42] Extracting initial bundle.
[Jun04.21:30:50] Decrypting signed bundle.
[Jun04.21:30:52] Extracting signed bundle.
[Jun04.21:31:00] Extracting packages.
[Jun04.21:31:09] Fetching repository metadata.
[Jun04.21:31:09] Skipping dpkg update in current boot image
[Jun04.21:31:11] Computing package update requirements.
[Jun04.21:31:12] Upgrade agent version: 5.1.1-5010115300
[Jun04.21:31:12] Upgrading software upgrade framework
[Jun04.21:31:16] Fetching repository metadata.
[Jun04.21:31:17] Skipping dpkg update in current boot image
[Jun04.21:31:17] Computing package update requirements.
[Jun04.21:31:17] Upgrade agent version: 5.1.2-5010215459
[Jun04.21:31:17] Upgrading nvOS 5.1.1-5010115300 -> 5.2.1-5020115690
[Jun04.21:32:28] Cleaning up old BEs.
[Jun04.21:32:30] Upgrading nvOS 5.1.1-5010115300 -> 5.2.1-5020115690
[Jun04.21:32:30] Software upgrade completed. Rebooting.
```

After the switch reboots twice and you see following message in serial console of the switch, you can SSH to switch as the **network-admin**:

```
nvOS system info:
serial number: 1XXXXXXXX00059
hostid: 090XXX9d
device id:
[ OK ] Started Netvisor Operating System.
Starting nvOSd Monitor...
[ OK ] Started nvOSd Monitor.
[ OK ] Reached target Multi-User System.
[ OK ] Reached target Graphical Interface.
[ OK ] Started Stop ureadahead data collection 45s after completed startup.
Starting Update UTMP about System Runlevel Changes...
[ OK ] Started Update UTMP about System Runlevel Changes.
* Welcome to Pluribus Networks Inc. Netvisor(R). This is a monitored
system. *
* ACCESS RESTRICTED TO AUTHORIZED USERS ONLY *
* By using the Netvisor(R) CLI, you agree to the terms of the Pluribus
Networks *
* End User License Agreement (EULA). The EULA can be accessed via *
* http://www.pluribusnetworks.com/eula or by using the command "eula-show"
*
switch1 login: network-admin
Password:
Netvisor OS Command Line Interface 5.1
```

```
Connected to Switch switch1; nvOS Identifier:0x90XXX9; Ver: 5.2.1-  
5020115690  
CLI (network-admin@Leaf1) > software-show  
version: 5.2.1-5020115690
```

## Implementing a Fabric Upgrade

---

A switch that is part of a fabric can be upgraded locally using software-upgrade process or you can start a fabric-wide upgrade of all nodes in a fabric.

While doing a fabric wide upgrade, the switch on which `fabric-upgrade` command is issued acts as the controller node. It is mandatory to copy the package to `/sftp/import/` directory of the controller node.

Netvisor copies the upgrade package to other nodes in the fabric as part of fabric-wide upgrade. The controller node monitors the progress of the upgrade on each node and you can view the status of the upgrade using the `fabric-upgrade-status-show` command. The controller node is identified by an “\*” after the switch name in the status output.

Netvisor ONE enables you to implement a fabric-wide upgrade and reboot the switches at the same time or in a sequential order.

### Upgrading the Fabric

Follow the below tasks to upgrade all switches in the fabric:

#### Upgrade Commands

Following are the commands that control the software or fabric upgrade process:

- **`fabric-upgrade-start`** – begin the upgrade process specifying the package name
- **`fabric-upgrade-status-show`** – monitor the progress of the upgrade for each node in the fabric
- **`fabric-upgrade-finish`** – assuming auto-finish option is not used, begin the reboot process based on options specified when upgrade is started
- **`fabric-upgrade-abort`** – abort the entire upgrade process and return switches to their prior state

The `fabric-upgrade-start` command defines all the future behavior of the upgrade process, meaning any optional settings need to be defined with the `start` command. In addition, the `fabric-upgrade-start` command acquires a configuration lock from all the members of the fabric. No configuration changes are permitted during the upgrade process.

#### Before you start the fabric-wide upgrade

1. Copy image to `/sftp/import/` directory of controller node
2. Ensure there is a reliable in-band and/or out-of-band connectivity between fabric members, which helps to distribute the software for the upgrade and monitor the progress of the upgrade process. The distribution of software to the nodes of the fabric is done in parallel, that is, each node receives the software approximately at the same time. An independent communications link is established over the fabric communications path to distribute the software to each node in the fabric.
3. Console access to switches are recommended
4. Switches do not accept any configuration commands once upgrade starts, so plan accordingly

#### Copying Image to the Switch



To copy the image:

- First, enable Secure File Transfare Protocol (SFTP) service by using the CLI command and create an /sftp/import directory:

```
CLI (network-admin@switch1)>admin-sftp-modify enable
sftp password:
confirm sftp password:
CLI (network-admin@switch1)>
```

OR

Enable shell access to copy the file to the folder by using the command:

```
CLI(admin@netvisor) > role-modify name network-admin shell
```

And access the shell:

```
CLI(admin@netvisor) > shell
network-admin@netvisor:~$ cd /sftp/import
network-admin@netvisor:/sftp/import$
```

- Copy the image to /sftp/import directory

```
root@server-os-9:~/# sftp sftp@switch1
The authenticity of host 'switch1 (10.0.0.02)' can't be established.
RSA key fingerprint is
SHA256:SI8VQZgJCpbbrF4sRcby36Fx7rz3Hh5EJl1PPyScLZU.
Are you sure you want to continue connecting (yes/no)? yes
Warning: Permanently added 'switch1, 10.0.0.02 (RSA)' to the list of known
hosts.
* Welcome to Pluribus Networks Inc. Netvisor(R). This is a monitored
system. *
* ACCESS RESTRICTED TO AUTHORIZED USERS ONLY *
* By using the Netvisor(R) CLI,you agree to the terms of the Pluribus
Networks *
* End User License Agreement (EULA). The EULA can be accessed via *
* http://www.pluribusnetworks.com/eula or by using the command "eula-
show" *
Password:
Connected to switch1
sftp> cd import
sftp> put nvOS-5.2.1-5020115690-onvl.pkg
Uploading nvOS-5.2.1-5020115690-onvl.pkg
nvOS-5.2.1-5020115690-onvl.pkg
nvOS-5.2.1-5020115690-onvl.pkg 100% 1870MB 7.5MB/s 04:00
```

## **Fabric upgrade with *manual-reboot* option**

This option completes in three phases:

- Copy upgrade package to switches in fabric and start upgrade with `fabric-upgrade-start` command.
- Finish or abort fabric upgrade with `fabric-upgrade-finish` or `fabric-upgrade-abort` commands.
- Manually reboot switches with the `switch-reboot` command.

## Starting the Fabric Upgrade

Before starting the upgrade process, ensure that all the nodes of the fabric are online, you can use the command `fabric-node-show` and check that the `state` is online for all the nodes.

Use the following command to copy the upgrade package from controller switch to all other switches in the fabric and start the upgrade process. Run the `fabric-upgrade-finish` command to reboot the fabric and complete the upgrade process:

```
CLI network-admin@switch >fabric-upgrade-start packages <image>
manual-reboot
```

The `fabric-upgrade-start` command defines all behavior of the upgrade process during the upgrade, that is, any optional settings need to be defined with the “start” command (see optional settings below). In addition, the `fabric-upgrade-start` command acquires a configuration lock from all the members of the fabric. No configuration changes are permitted during the upgrade process.

The optional setting parameters for the `fabric-upgrade-start` command includes:

- `auto-finish` — you can specify to automatically reboot the entire fabric after the upgrade is complete. The default is `no-auto-finish`.
- `abort-on-failure` — specify if you want the upgrade to stop if there is a failure during the process.
- `manual-reboot` — specify if you want to manually reboot individual switches after the upgrade process. If you specify `no-manual-reboot`, all switches reboot automatically after the upgrade is complete.
- `prepare` — specify if you want to perform setup steps prior to performing the upgrade. This step copies the offline software package and then extracts and prepares for the final upgrade process. Once you begin the prepare process, you cannot add new switches to the fabric.

A sample upgrade process is explained below:

Start the upgrade process by using the command:

```
CLI (network-admin@switch1) > fabric-upgrade-start packages nvOS-5.2.1-
5020115690-onvl.pkg auto-finish
Warning: This will start software upgrade on your entire fabric.
Please confirm y/n (Default: n):y
Scheduled background update.
```

## Monitoring the Upgrade Process

The controller node monitors the progress of the upgrade on each node and reports the status of the upgrade by using the `fabric-upgrade-status-show` command. There are many interim steps to the upgrade process and to continually monitor the upgrade process use the `show-interval (in seconds)` option with the `fabric-upgrade-status-show` command:

Use the following commands to:

- To monitor the progress of the upgrade for each node in the fabric:

```
CLI (network-admin@switch1) > fabric-upgrade-status-show
```

For example,

```
CLI (network-admin@tucana-colo-6) > fabric-upgrade-status-show show-interval 5
```

log	switch	state	cluster
-----			
(0:00:36)Agent needs restart	eq-colo-7	Agent restart wait	agr07-08(sec)
(0:00:34)Agent needs restart	tucana-colo-7	Agent restart wait	spine-cl(sec)
(0:03:57)Extracting signed bundle.	aquarius-test-1	Running	aquarius-test-1-2(sec)
(0:00:45)Agent needs restart	dorado-test-3	Agent restart wait	dorado-test-2-3(sec)
(0:03:57)Extracting signed bundle.	agr08	Running	agr07-08(pri)
(0:00:28)Agent needs restart	tucana-colo-6*	Agent restart wait	spine-cl(pri)
(0:03:57)Extracting signed bundle.	aquarius-test-2	Running	aquarius-test-1-2(pri)
(0:00:38)Agent needs restart	dorado-test-2	Agent restart wait	dorado-test-2-3(pri)
(0:01:00)Agent needs restart	scorpius10	Agent restart wait	none
(0:00:47)Agent needs restart	vnn-mini-1	Agent restart wait	none
log	switch	state	cluster
-----			
(0:00:36)Agent needs restart	eq-colo-7	Agent restart wait	agr07-08(sec)
(0:00:34)Agent needs restart	tucana-colo-7	Agent restart wait	spine-cl(sec)
(0:04:02)Extracting packages.	aquarius-test-1	Running	aquarius-test-1-2(sec)
(0:00:45)Agent needs restart	dorado-test-3	Agent restart wait	dorado-test-2-3(sec)
(0:04:02)Extracting signed bundle.	agr08	Running	agr07-08(pri)
(0:00:28)Agent needs restart	tucana-colo-6*	Agent restart wait	spine-cl(pri)
(0:04:02)Extracting packages.	aquarius-test-2	Running	aquarius-test-1-2(pri)
(0:00:38)Agent needs restart	dorado-test-2	Agent restart wait	dorado-test-2-3(pri)
(0:01:00)Agent needs restart	scorpius10	Agent restart wait	none
(0:00:47)Agent needs restart	vnn-mini-1	Agent restart wait	none
.			
.			
log	switch	state	cluster
-----			
-----			
(0:01:53)Waiting for completion processing	eq-colo-7	Upgrade complete	
agr07-08(sec)			
(0:01:25)Waiting for completion processing	tucana-colo-7	Upgrade complete	

```

spine-cl(sec)
(0:06:24)Waiting for completion processing aquarius-test-1 Upgrade complete
aquarius-test-1-2(sec)
(0:02:29)Waiting for completion processing dorado-test-3 Upgrade complete
dorado-test-2-3(sec)
(0:06:43)Waiting for completion processing aqr08 Upgrade complete
aqr07-08(pri)
(0:01:23)Waiting to reboot tucana-colo-6* Upgrade complete
spine-cl(pri)
(0:06:16)Waiting for completion processing aquarius-test-2 Upgrade complete
aquarius-test-1-2(pri)
(0:02:19)Waiting for completion processing dorado-test-2 Upgrade complete
dorado-test-2-3(pri)
(0:06:09)Waiting for completion processing scorpius10 Upgrade complete
none
(0:08:09)Upgrading nvOS 5.1.2-5010215446 -> 5.2.1-5020115690 vnv-mini-1 Running
none
.
.
log switch state cluster
-----
(0:01:53)Current/Reboot BE: netvisor-16 eq-colo-7 Upgrade complete aqr07-08(sec)
(0:01:25)Waiting for completion processing tucana-colo-7 Upgrade complete spine-cl(sec)
(0:06:24)Waiting for completion processing aquarius-test-1 Upgrade complete aquarius-test-1-2(sec)
(0:02:29)Destroy BE: netvisor-45 dorado-test-3 Upgrade complete dorado-test-2-3(sec)
(0:06:43)Waiting for completion processing aqr08 Upgrade complete aqr07-08(pri)
(0:01:23)Waiting to reboot tucana-colo-6* Upgrade complete spine-cl(pri)
(0:06:16)Current/Reboot BE: netvisor-10 aquarius-test-2 Upgrade complete aquarius-test-1-2(pri)
(0:02:19)Software upgrade done. Waiting for reboot dorado-test-2 Upgrade complete dorado-test-2-3(pri)
(0:06:09)Waiting for completion processing scorpius10 Upgrade complete none
(0:13:17)Waiting for completion processing vnv-mini-1 Upgrade complete none
log switch state cluster
-----
(0:01:53)Upgrade complete eq-colo-7 Reboot wait aqr07-08(sec)
(0:01:25)Upgrade complete tucana-colo-7 Reboot wait spine-cl(sec)
(0:06:24)Upgrade complete aquarius-test-1 Reboot wait aquarius-test-1-2(sec)
(0:02:29)Upgrade complete dorado-test-3 Reboot wait dorado-test-2-3(sec)
(0:06:43)Upgrade complete aqr08 Reboot wait aqr07-08(pri)
(0:01:23)Sending Reboot wait message to handler tucana-colo-6* Reboot wait spine-cl(pri)
(0:06:16)Upgrade complete aquarius-test-2 Reboot wait aquarius-test-1-2(pri)
(0:02:19)Upgrade complete dorado-test-2 Reboot wait dorado-test-2-3(pri)
(0:06:09)Upgrade complete scorpius10 Reboot wait none
(0:13:17)Waiting for completion processing vnv-mini-1 Upgrade complete none
Connection to tucana-colo-6 closed by remote host.

```

Connection to tucana-colo-6 closed.

The first entry in the log is the elapsed time of the upgrade process. It does not include waiting time. The switch with the asterisk (\*) is the upgrade controller node where the `fabric-upgrade-start` command was issued.

During a fabric-wide upgrade, the messages displayed by the `fabric-upgrade-status-show` command, based on the current progress status is described in table below:

Message	Description
Downloading package bundle	The upgrade package is downloaded from the initial node to all the other nodes.
Extracting initial bundle	Once successfully downloaded, the offline bundle is extracted.
Extracting signed bundle	The signature of the package is verified.
Extracting packages	The packages are extracted and readied to install.
Agent needs restart	The nodes wait for the package to be extracted on all nodes of the fabric.
Upgrading nvOS *	The switch upgrades Netvisor from the older version to the newer one
Waiting for fabric-upgrade-finish/abort	The switches wait for the user to complete the upgrade once it completes using either of the commands mentioned above.

- Once the upgrade package is copied to all switches by fabric upgrade process and the upgrade process is completed, run the `fabric-upgrade-finish` or `fabric-upgrade-abort` command to either finish the upgrade or abort it.

```
CLI (network-admin@switch1) > fabric-upgrade-finish
```

You can issue this command any time during the fabric upgrade to reboot all nodes when upgrade is complete. Once the upgrade phase is complete, all switches display the *Upgrade complete* message in the log field. You can then reboot the fabric. Following is an example:

```
CLI (network-admin@switch1) > fabric-upgrade-finish
```

log	switch	state	cluster
(0:13:00)Waiting for fabric-upgrade-finish/abort	sw2	Upgrade complete	spine(sec)
(0:12:04)Waiting for fabric-upgrade-finish/abort	sw1*	Upgrade complete	spine(pri)
(0:16:49)Waiting for fabric-upgrade-finish/abort	sw1	Upgrade complete	none
(0:15:27)Waiting for fabric-upgrade-finish/abort	sw2	Upgrade complete	none

Finalizing upgrade. Manual reboot of nodes required.

- Manual reboot: each switch in the fabric need to be manually rebooted after the upgrade is completed. The `fabric-upgrade-status-show` command displays the status as *switch waiting to reboot*. For example,

```
CLI (network-admin@switch1) > fabric-upgrade-status-show
fabric-upgrade-status-show: Switch waiting to reboot
```

At this point, upgrade is completed on all switches, reboot switches one at a time by the following command:

```
CLI (network-admin@switch1) > switch-reboot
```

**Note:** You must **reboot** the *controller switch* at the end only.

**Note:** All the nodes of the fabric should be running the same software version for the Netvisor ONE features to work correctly.

- During the installation, if there is any issue, the upgrade process can be rolled back using the command `fabric-upgrade-abort`. To abort the upgrade process and return the switches to their prior state (no reboot needed):

```
CLI (network-admin@switch1) > fabric-upgrade-abort
```

Aborts the fabric upgrade process. All changes to the switches are cleaned up and the server-switches do not reboot. The configuration lock on the fabric is also released. If you issue the `fabric-upgrade-abort` command during the upgrade process, it may take some time before the process stops because the upgrade has to reach a logical completion point before the changes are rolled back on the fabric. This allows the proper cleanup of the changes.

---

**Warning:** DO NOT use the `switch-reboot` command to reboot the switch while upgrade is in progress.

---

**Note:** During the fabric-upgrade process, the fabric configuration is locked throughout the entire process and you cannot change any configurations during the process.

### Related Command:

Other related commands for fabric-upgrade includes:

- `fabric-upgrade-prepare-cancel` — cancels a fabric upgrade that was prepared earlier.
- `fabric-upgrade-prepare-resume` — resume a fabric upgrade that was prepared earlier.
- `fabric-upgrade-prepare-show` — displays the status of prepared upgrades on the fabric nodes.

## Review bootenv

A new boot environment is built during the upgrade process. Upon reboot this new boot environment becomes active and the new software is up-and-running on the switch. Generally, it is not required to interact with the boot environments during the upgrade process. It may be necessary to review the boot environments using the command `bootenv-show` if there is some failure during the upgrade process.

## Saving and Restoring Netvisor ONE Configurations

---

A switch contains local configuration information such as port settings as well as fabric configuration information. Fabric configurations are stored on every switch in the fabric and does not require that you save and restore before replacing a switch. When a switch is replaced, removed, or otherwise disrupted, you can save and restore the local configuration information.

The information that is saved and restored on the local switch includes the following:

- VNETs with vNET manager running on the switch
- Port VLAN associations
- Network services running on the switch

To display a full list of the current configuration details for a switch, use the `running-config-show` command.

SFTP and NFS can be used to transfer the configuration file, but you must enable the two features before using them.

**Caution:** There is a potential for data loss when restoring a configuration. The configuration on the switch is replaced by the configuration stored in the import file. Although ISO images and disk-library images are not likely to disappear, you should only perform `switch-config-import` on a switch that doesn't have important data stored on it. As a precaution, consider using the command `switch-config-export` to save the data on the switch that you are importing the configuration file. Also, copy the ISO images and disk images from the switch using the `iso-image-library` and `disk-library-image-export` commands and copying the files from the switch.

To save the switch configuration to a file, use the following command:

```
CLI (network-admin@Leaf1) > switch-config-export export-file pleiades24
```

```
Exported configuration to /nvOS/export/pleiades24.2013-11-04T22.33.31.tar.gz
```

To display the files available for import and export, use the following command:

```
CLI (network-admin@Leaf1) > switch-config-show
```

```
switch      export-file
pleiades24  pleiades24.2013-11-04T22.33.31.tar.gz
```

You can now copy the configuration file to a different host using SFTP or NFS. For example, you can SFTP to the `switch-ip-address`, and login using the SFTP password. Then use `cd /nvOS/import`, and use `get` to download the configuration file.

The `switch-config-export` command is used to export the configuration of the local switch. The file that is created is a tar file that includes a number of configuration files for the switch. The file is created



under `/nvOS/export`. This is the command used to export the current configuration on the local switch. Also, each time you reset the switch using the command, `switch-config-reset`, a backup of the configuration is made and places a file in the same location.

Once the switch configuration is exported, it becomes available to import on the same switch, by using the `switch-config-copy-to-import` command. Netvisor One copies the configuration tar file from the `/nvOS/export` to the `/nvOS/import` directory. Once in the `/nvOS/import` directory, it is possible to use the `switch-config-import` command to import the switch configuration.

- The `switch-config-import` command is used to import a configuration on the local switch. When using that command, the intention is to import a switch configuration that was previously exported by the same switch.
- The `switch-config-import` command has a few parameters to it. The `ignore-system-config` and the `apply-system-config` parameters are two options that allow the imported configuration of the switch to override or not override the currently configured information found under the `switch-setup-show` command. When you select the `ignore-system-config` parameter, the local configuration is saved to an archive. If you select `apply-system-config`, the settings in the tar file are applied to the local switch.
- When you import a configuration using the `switch-config-import` command, the current configuration on the switch is overwritten by the imported configuration file.

When a switch that was part of a cluster is replaced, the `fabric-join repeer-to-cluster-node` command is used for the new switch to receive all required switch configuration, including the local configuration.

To upload a configuration file to a switch and set the configuration for the switch using the configuration file, you must transfer the configuration file to the target switch using the following sequence of commands:

```
sftp@<switch-ip-address>
Connecting to switch-ip-address
Password: <password>
sftp> cd nvOS/import
sftp> put pleiades24.2013-11-04T22.33.31.tar.gz
```

**Note:** The configuration file must use the `*.tar.gz` extension to be recognized by nvOS.

**Caution:** Loading the configuration file causes nvOS to restart which results in a brief interruption to switch traffic flow.

Now load the configuration file which replaces the current configuration on the switch with the information in the file.

```
CLI (network-admin@Leaf1) > switch-config-import import-file
pleiades24.2019-11-04T22.33.31.tar.gz
```

New configuration imported. Restarting nvOS...

Connected to Switch pleiades24; nvOS Identifier:0xb000011; Ver: 5.1.2.15459

There are many options available that allow you to control how the `switch-config-import` modifies the switch, including the following:

- `ignore-system-config` — ignore the current system configuration. The settings in the `*.tar` file are not applied to the local switch.
- `apply-system-config` — apply the system configuration in the imported file. The settings in the `*.tar` file are applied to the local switch. You typically do not want to use this option as it changes the in-band IP address and other settings.

By default, the initial switch system configuration, management IP addresses and other parameters, are not applied if there is another switch in the fabric with the same settings. To apply the initial settings, use the `apply-system-config` option. Also, by default, the imported configuration attempts to join the same fabric that the original switch was a member.

## Copying and Importing Configuration Files

---

You can create a configuration file to import to another switch by using the `switch-config-copy-to-import` command. To create a configuration file with the name `config-092613` to import on another switch, use the following syntax:

```
CLI (network-admin@Leaf1) > switch-config-copy-to-import export-file  
config-092613
```

After you create the configuration file, you can export it to `/nvOS/export/` directory, and SFTP to it from the target switch.

To review the available files for import and export, use the following syntax:

```
CLI (network-admin@Leaf1) > switch-config-show  
  
switch          export-file  
pbg-nvos        config-092613.tar.gz
```

Depending on the available remote access services, you can now copy the configuration file to a different switch. For example, you can SFTP to another switch using the IP address of the switch, login as SFTP with the password that you previously set, `cd /nvOS/import` and `get` the configuration file.

To upload the configuration file to the target switch and set the configuration from the configuration file, transfer the configuration file to the target switch with the IP address, `192.168.3.35`.

To export a configuration to a server, use the `switch-config-export` command:

```
CLI (network-admin@Leaf1) > switch-config-export
```

## Auto-configuration of IPv6 Addresses on the Management Interface Support

---

### IPv6 Stateless Address Auto-Configuration (SLAAC)

Like IPv4 addresses, you can configure hosts in a number of different ways for IPv6 addresses. Dynamic Host Configuration Protocol (DHCP) assigns IPv4 addresses dynamically and static addresses assign fixed IP addresses. DHCP provides a method of dynamically assigning addresses, and provides a way to assign the host devices other service information like DNS servers, domain names, and a number of different custom information.

SLAAC allows you to address a host based on a network prefix advertised from a local network router using Router Advertisements (RA). RA messages are sent by default by IPv6 router.

These messages are sent out periodically by the router and include following details:

- One or more IPv6 prefixes (Link-local scope)
- Prefix lifetime information
- Flag information
- Default device information (Default router to use and its lifetime)

Netvisor ONE enables SLAAC by default on the switch.

When you configure IPv6 address on the management interface during setup, the parameter, **assignment**, has two options:

- **none** — Disables IPv6 addresses.
- **autoconf** — Configure the interface with SLAAC.

## Configuring 10/100M Override Using Bel-Fuse SFP-1GBT-05 on F9372-X Platforms

Auto-negotiation function between two connected devices stipulates a set of shared parameters - link speed, duplex mode, and flow control. The F9372-X/AS5812-54X platform, with the Bel-Fuse SFP-1GBT-05 copper transceiver plugged into one of its ports, can support legacy 10M/100M devices that do not have auto-negotiation capability. As the SFP-1GBT-05 transceiver performs auto-negotiation by default, this function should be disabled on the F9372-X switch through nvOS. An operating link speed of either 10M or 100M can be configured. With the exception of F9372-X, the 10/100M override capability is not available on other switch platforms and is supported only on the SFP-1GBT-05 transceiver.

Use the command below for disabling auto-negotiation and overriding the link speed on F9372-X platform. Enter the *port number* on which the transceiver is connected in the command:

```
CLI (network-admin@Leaf1) > port-config-modify port <port number> speed
<10m|100m> no-autoneg
```

<code>port-config-modify</code>	Use this command to update port configuration.
<code>port &lt;port-list&gt;</code>	Specify the ports that need to be configured as a list separated by commas.  <b>Note:</b> For 10/100M override, specify the port number on which the transceiver is connected.
Specify one or more of the following options:	
<code>speed &lt;disable 10m 100m 1g&gt;</code>	Specify one among the options as the speed at which the port should operate.  <b>Note:</b> specify the speed as either 10m or 100m for legacy device support.
<code>autoneg no-autoneg</code>	Specify either of the options to enable auto-negotiation or to disable it.  <b>Note:</b> auto-negotiation has to be disabled to achieve 10/100M override.

For example, to disable auto-negotiation on port 38 and set a speed of 10M, use the command below:

```
CLI (network-admin@Leaf1) port-config-modify port 38 speed 10m no-autoneg
```

However, the device do not get reprogrammed upon reinsertion. To recover from this, remove and re-add the 10M/100M override port configuration . [PR30110]

## Support for Local Loopback IP Addresses

Netvisor ONE uses the loopback interface as an always up and available virtual interface, and you can assign it a unique IPv4 or IPv6 address. Netvisor ONE uses a loopback interface as a termination address for some routing protocols, because of the availability of the interface. Netvisor ONE allows you to configure a loopback address for a global zone.

- Send a dedicated ping to loopback interface
- Create a BGP neighbor using the loopback Interface with OSPF so reach-ability is there for BGP and BGP next hop self
- Make sure log messages do not show any issues

Netvisor ONE deploys the loopback IP address as persistent in the configuration and not affected by a reboot or reset of Netvisor ONE.

To add a loopback IPv4 or IPv6 address or both to an existing configuration, use the following syntax:

```
CLI (network-admin@switch1) > switch-setup-modify loopback-ip ip-  
address loopback-ipv6 ipv6-address
```

For example, to add the IPv4 address, 12.1.1.1, and the IPv6 address, 1212::1, use the following syntax:

```
CLI (network-admin@switch1) > switch-setup-modify loopback-ip 12.1.1.1  
loopback-ip6 1212::1
```

```
CLI (network-admin@switch1) > switch-setup-show format in-band-ip,in-band-  
ip6,loopback-ip,loopback-ip6, layout horizontal
```

in-band-ip	in-band-ip6	loopback-ip	loopback-ip6
150.1.1.1/24	2001::1/96	12.1.1.1	1212::1
150.1.1.2/24	2001::2/96	12.1.1.2	1212::2

After configuring the loopback address, you can SSH to the switch over the management, in-band, or loopback interface using the following syntax:

```
CLI (network-admin@switch1) > ssh network-admin@<mgmt/inband/loopback ip-  
address>
```

Then from CLI, execute the `shell` command to access the switch shell:

```
CLI (network-admin@switch1) > network-admin@switch:~$
```

## Configuring REST API Access

Netvisor ONE enables you to use REST API over HTTP and HTTPS to manage the switches in a fabric, in addition to using the CLI. Though REST API access over HTTP is simpler to configure, Pluribus recommends using HTTPS for security reasons. The vREST web application that runs on the switch enables the REST API client to access the resources on the switch.

Follow the steps below to configure REST API access over HTTP:

- Enable the web service using the command `admin-service-modify`.

CLI (network-admin@switch1) `admin-service-modify if mgmt web`

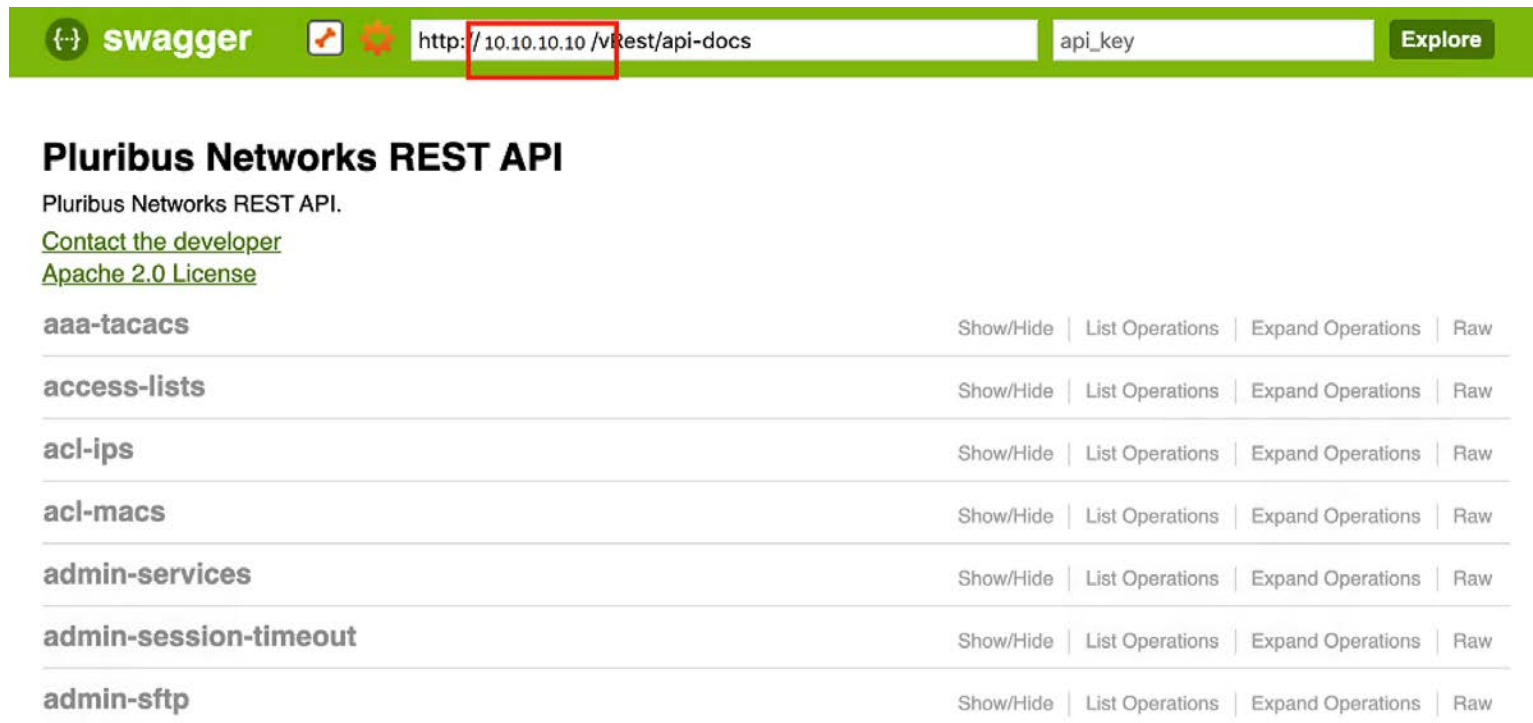
<code>admin-service-modify</code>	Modifies services on the switch.
<code>if if-string</code>	Specify the administrative service interface. The options are <code>mgmt</code> or <code>data</code> .
<code>web no-web</code>	Specify if you want to enable web management. <b>Use this option to enable REST API access over HTTP.</b>
<code>web-ssl no-web-ssl</code>	Specify if you want to use SSL and certificates for web services. <b>Use this option to enable REST API access over HTTPS.</b>
<code>web-ssl-port web-ssl-port-number</code>	Specify the web SSL port.
<code>web-port web-port-number</code>	Specify the port for web management.
<code>web-log no-web-log</code>	Specify if you want to turn on or off web logging.

- Verify configuration using the command `admin-service-show`:

CLI (network-admin@switch1) `admin-service-show`

switch	if	ssh	nfs	web	web-ssl	web-ssl-port	web-port	snmp	net-api	icmp
switch1	mgmt	on	off	on	on	443	80	on	off	on
switch1	data	on	off	on	off	443	80	on	off	on

- Download Swagger tool onto the desktop and open index.html file.
- Paste the URL `http://<ip-address>/vRest/api-docs` into the URL field.  
 <ip-address> should be the hostname or management IP address of your switch.
- Click 'Explore'. Commands are populated as below:



**Pluribus Networks REST API**

Pluribus Networks REST API.  
[Contact the developer](#)  
[Apache 2.0 License](#)

<b>aaa-tacacs</b>	Show/Hide	List Operations	Expand Operations	Raw
<b>access-lists</b>	Show/Hide	List Operations	Expand Operations	Raw
<b>acl-ips</b>	Show/Hide	List Operations	Expand Operations	Raw
<b>acl-macs</b>	Show/Hide	List Operations	Expand Operations	Raw
<b>admin-services</b>	Show/Hide	List Operations	Expand Operations	Raw
<b>admin-session-timeout</b>	Show/Hide	List Operations	Expand Operations	Raw
<b>admin-sftp</b>	Show/Hide	List Operations	Expand Operations	Raw

**Figure 2- 1: Swagger UI for Pluribus REST API**

- Right click to enable Chrome Inspect, and view the network tab to ensure that all the commands are loaded.
- Expand one of the CLI commands to see the corresponding GET/POST/DELETE/PUT options.
- Click on the model schema to view the list of configurable options supported under that model (vRest API) which are populated as key-value pairs. Make the required modifications to the values and click 'Try it out!'.



## Using cURL with REST API

To create a VLAN, use the vREST API:

```
$ curl -u network-admin:pluribus!23 -H "Content-Type:application/json" -X POST
http://switch1/vRest/vlans -d
{
  "scope": "local",
  "id": 1111,
  "description": "hello world"
}
```

By default, all the vRest APIs provide fabric level information. To specifically access the resources of the switch (scope : local ), the switch ID needs to be specified in the URL.

For switch ID specific information, use the command:

```
$ curl -u network-admin:pluribus!23 http://switch1/vRest/vlans?api.switch={hostid} |
python -m json.tool
```

as in the following example:

```
$ curl -u network-admin:pluribus!23 http://10.10.10.10/vRest/vlans?
api.switch=201327131 | python -m json.tool
```

For switch information listing a local scope:

```
$ curl -u network-admin:pluribus!23 http://switch1/vRest/vlans?api.switch=fabric |
python -m json.tool
```

or

```
$ curl -u network-admin:pluribus!23 http://switch1/vRest/vlans | python -m json.tool
```

as in the following example:

```
$ curl -u network-admin:pluribus!23 http://10.110.0.48/vRest/vlans | python -m json.tool
```

## Configuring REST API Access over HTTPS

To enable HTTPS communication between a REST API client and Netvisor vREST web service, you have two options:

1. You can generate a self-signed certificate using Netvisor CLI and use this certificate for REST web service.
2. After creating a self-signed certificate using Netvisor CLI, create a certificate request, get the certificate request signed by a trusted Certificate Authority (CA), import the signed certificate and CA certificate into

Netvisor ONE and use the certificates for REST web service.

Follow the steps below to create the certificates and deploy them:

- Generate self-signed certificate (the private key and the certificate file, in PEM format) using the `web-cert-self-signed-create` command.

```
CLI (network-admin@switch1) > web-cert-self-signed-create
```

<code>web-cert-self-signed-create</code>	This command creates a self-signed certificate and deletes any existing certificates.
<code>country country-string</code>	Specify the contact address of the organization, starting with the country code.
<code>state state-string</code>	Specify the state or province.
<code>city city-string</code>	Specify the city.
<code>organization organization-string</code>	Specify the name of the organization.
<code>organizational-unit organizational-unit-string</code>	Specify the organizational unit.
<code>common-name common-name-string</code>	Specify the common name. The common name must precisely match the hostname where the certificate is installed.

For example:

```
CLI (network-admin@switch1) > web-cert-self-signed-create country US state
California city "Santa Clara" organization "Pluribus Networks Inc"
organizational-unit Engineering common-name switch1.pluribusnetworks.com
Successfully generated self-signed certificate.
```

- If you want to get the certificate signed by a trusted Certificate Authority(CA), generate a CSR from the self-signed certificate by using the command `web-cert-request-create`.

```
CLI (network-admin@switch1) > web-cert-request-create
Certificate signing request successfully generated
at /sftp/export/switch1.pluribusnetworks.com.csr.
```

- To view the CSR, use the command `web-cert-request-show`.

```
CLI (network-admin@switch1) > web-cert-request-show
```

<code>web-cert-request-show</code>	Displays the certificate signing request.
<code>cert-request cert-request-string</code>	Specify the name of the CSR.

For example:

```
CLI (network-switch1) > web-cert-request-show
```

```
cert-request
```

-----BEGIN CERTIFICATE REQUEST-----

```
MIICnDCCAYQCAQEwVzELMAkGA1UEBhMCVVMxCzAJBgNVBAGMAkNBMQswCQYDVQQH
DAJTSjELMAkGA1UECgwCUE4xDALBgNVBASMBEVuZ2cxEjAQBgNVBAMMCWVxLWNv
bG8tMTCCASIwDQYJKoZIhvcNAQEBBQADggEPADCCAQoCggEBALMmrZ8hvZ5J+FRs
LolsfVtwmmLEaxyhaxD/HNVdXSRhzbQDT20+qySfOudxtWGKyCsuCFFbgMUz7rgu
HlXle8uwPSoxgTjLGq20sgBQIfNBT5UwDLDuZUUPzMEEjFb3/9Cg1VWju2t1KPim
Ggg3rcA3PCsMeCr/q+9Gz6gfLe6Rfx91yxTA44ZWsOWnvGdDxAPfHOLZ5zBWG8a3
ohgOwMLjy21ytDTA6aR1M9I12MkJwev3t0y6n/CLp6Zigp5wXiArPPnR9sZ+E7so
MqpEzz0rjFDfrNwNAGMzT3WPcmlYRjYrUJ0QsOEQ+OluHJaNbwlpJEmK2jm97kbbk
/HvEFmMCAwEAAaAAMA0GCSqGSIB3DQEBBQUAA4IBAQCnlgEwzoesbuiCYG7HZJN/
Rxm/NcznpvJXXdlTAdzSbTWWLswrZMyX6bQqUTWEb3qvVccD4tIZShyIGiR0CpCD
22m8LD4+e6/FA6NiJjanHkKsRW9Z7ka97TFpsUaH27sUTtfFDDkDImwRIGfns+nu
kTRNMuNiyC/+uHovsvCxS8is3OasQtS1lkG28sZgxisvP17qmfjlb9fQC3pcvR4t
K8GciPMUfgcIA5qLDmCZAg1A6JBMB/UHTUuEnztLrLz4qjWqJJK3pWvdLWZcKDEz
C0t5Dre9ByJ2RT75GdUq2cl6xYBGAWZNCzjdhparyBnvn00Mwb6PpPmLGcBQiRnN
```

-----END CERTIFICATE REQUEST-----

- Send the CSR to your trusted CA. You can copy the `web-cert-request-show` output and send it to the CA for signing the certificate.

You can also connect to the switch by using SFTP and copy the certificate file from `/sftp/export` location and send it to the CA.

If disabled, use the command `admin-sftp-modify enable` to enable SFTP.

- Upload the signed certificate, the CA root certificate, and the intermediate CA certificate (if the certificate is signed by an intermediate CA) to `/sftp/import` directory on the switch using SFTP.

For example, to upload the file `server-cert.pem` to the `/sftp/import` directory, follow the steps below:

```
$ sftp sftp@switch1
```

Password:

```
sftp> cd /sftp/import
```

```
sftp> put server-cert.pem
```

- Import the signed server certificate, CA root certificate, and the intermediate certificate (if available) onto the switch using the `web-cert-import` command:

```
CLI (network-admin@switch1) > web-cert-import
```

<code>web-cert-import</code>	This command imports certificates from <code>/sftp/import</code> directory.
<code>file-ca file-ca-string</code>	Specify the name of the CA certificate file.
<code>file-server file-server-string</code>	Specify the name of server certificate file (signed by CA).
<code>file-inter file-inter-string</code>	Specify the name of intermediate CA certificate

file.

```
CLI (network-admin@switch1) > web-cert-import file-ca ca.pem file-server
server-cert.pem file-inter intermediate.pem
Successfully imported certificates.
```

- After the import is successful, enable web-ssl using the admin-service-modify command.

```
CLI (network-admin@switch) > admin-service-modify if mgmt web-ssl
```

- Download the Swagger tool and follow the general steps above to complete the configuration of REST API access.

## Related Commands

- web-cert-clear

Use this command to delete previously generated certificates.

For example:

```
CLI (network-admin@switch1) > web-cert-clear
Successfully deleted all certificate files.
```

- web-cert-info-show

Use this command to display web certificate information.

```
CLI (network-admin@switch1) web-cert-info-show
```

web-cert-info-show	Displays the web certificate information.
Specify any of the following options:	
cert-type ca intermediate server	Specify the one among the options as the certificate type.
subject <i>subject-string</i>	Specify the the subject of the certificate.
issuer <i>issuer-string</i>	Specify the issuer of the certificate.
serial-number <i>serial-number</i>	Specify the serial number of the certificate.
valid-from <i>valid-from-string</i>	Specify the time from which the certificate is valid.
valid-to <i>valid-to-string</i>	Specify the time at which the certificate expires and is no longer valid.

For example:

CLI (network-admin@switch1) web-cert-info-show

```
switch:      switch1
cert-type:   ca
subject:     /C=US/ST=CA/L=SJ/O=PN/OU=Engg/CN=switch1
issuer:      /C=US/ST=CA/L=SJ/O=PN/OU=Engg/CN=switch1
serial-number: 1
valid-from:  May   7 18:16:10 2019 GMT
valid-to:    May   6 18:16:10 2020 GMT
-----
```

```
switch:      switch1
cert-type:   server
subject:     /C=US/ST=CA/L=SJ/O=PN/OU=Engg/CN=switch1
issuer:      /C=US/ST=CA/L=SJ/O=PN/OU=Engg/CN=switch1
```

## Using cURL to Implement SSL Certificates

Use cURL to automate the upload of the CA root, CA intermediate, and signed switch certificates.

Run the following command for each of the PEM formatted certificates:

```
awk 'NF {sub(/\r/, ""); printf "%s\\n",$0;}' <file-name>.pem
```

For example:

```
$ awk 'NF {sub(/\r/, ""); printf "%s\\n",$0;}' /tmp/server-cert.pem.bkp
```

```
-----BEGIN CERTIFICATE-----
```

```
\nMIIDHDCCAgQCAQEwDQYJKoZIhvcNAQELBQAwVDELMAkGA1UEBhMCSU4xCzAJBgNV\n\nBAAgMAktBMQwwCgYD\nVQqHDANCTFIxCzAJBgNVBAAoMAlBOMQwwCgYDVQQLDANFTkcx\n\nDzANBgNVBAMMB1NQSU5FMTAeFw0yMDA1MDQxODM4NTZa\n\nMFQxQzAJBgNVBAYTAklOMQswCQYDVQQIDAJLQTEMMAoGA1UEBwwDQkxSMQswCQYD\n\nVQqKDAJQTjEMMAoGA1UECwwDRU5HMQ8wDQYDVQQDDAZTUElORTEWggEiMA0GCSqG\n\nSIb3DQEBAQUAA4IBDwAwggEKAoIBAQCot16ddH0LNHOrWZt63FHYiArVXYIYbCDh\n\nWCY6MX3suoXYvKstvRgJkUe/6G5As+vYtwRi2bDqdDsgTC5+Qo4SnjrdTcTM98F+\n\n0Qzqv02c+dbzk5GclUkljqQ0PHGXRPGOMhou8B/6LI9Hg5XkG+FSfaDGQTM39uj2\n\nzzvrMOFn96gzpTBoh40sMoIpnKQLrWeGjlnXaBxhM342cljnlCVmXss/uHMQeang\n\nsVhPTynikyXIrDwl9gh/2XlEwzVzpAnUBTUZvJ9rgrceC9GcuGmiPZgxxSruNb0w\n\nK8xSyH8/hLwhK4Axgu3a+lfmmKFmSWjywmcxlmQl+jwiMPA/Ty55AgMBAAEwDQYJ\n\nKoZIhvcNAQELBQADggEBAEG0D/2FcNU6Z6w/6eKbyH855kHSrJyqeU8eoCW9rnOb\n\nnqdnAsFX3aYwiUCjzSFXpWA3bRr3L7X0Y01x7VSvWITuDVwO431lK29rQfrSvoPiw\n\nnf7fhU7bszlUc2GAumU9OEdYBnSi1DzfBawUcPmbDmm+ci27k0po53KDWTbxbkBIZR\n\nn2Oh25LXkmq8ZBZE4vgS+mAw436nToazB1/vDTMWOBuLVzOUlU8cdcjJUnJBvTbX\n\nnThP691sHVMED8B8Fhl08BzIjMqQ9qp1tjplFq1Ea9oEFnT5U5gKvJYy48qEPlW+r\n\nhRIHysvZXF/dghtrLXDMSBWLlLofUsDQsh+qLxpol+k=\n\n-----END CERTIFICATE-----\n
```

Copy the output into the JSON payload.

Note: The escape character syntax of `\n` must be used as highlighted in red in the example below. Otherwise, the script will fail, and the certificates will not install.

```
$ curl -u network-admin:test123 http://10.100.64.5/vRest/web-certs/upload -H
"content-type:application/json" -v -X POST -d '{"cert-ca": "-----BEGIN
CERTIFICATE-----
\nMIIDHDCCAgQCAQEWdQYJKoZIhvcNAQELBQAwVDELMAkGA1UEBhMCSU4xCzAJBgNV\n\nBAGMAktBMQwwCgYD
VQQHDANCTFlixCzAJBgNVBAoMAlBOMQwwCgYDVQQLDANFTkcx\n\nDzANBgNVBAMMB1NQSU5FMTAeFw0yMDA1MD
QxODM4NTZaFw0yMTA1MDQxODM4NTZa\n\nMFQxQzAJBgNVBAYTAklOMQswCQYDVQQIDAJLQTEEMMAoGA1UEBwwD
QkxSMQswCQYD\n\nVQQKDAJQTjEMMAoGA1UECwwDRU5HMQ8wDQYDVQQDDAZTUe1ORTewggEiMA0GCSqG\n\nsIb3
DQEBAQUAA4IBDwAwggEKAoIBAQCot16ddH0LNHOrWZt63FHYiArVXYIYbCDh\n\nWCY6MX3suoXYvKstvRgJkU
e/6G5As+vYtwRi2bDqdDsgTC5+Qo4SnjrdTcTM98F+
\n\n0Qzqv02c+dbzk5GclUkljqQ0PHGXRPgOMhou8B/6LI9Hg5XkG+FSfaDGQTM39uj2\n\nzzvrMOFn96gzpTBo
h40sMoIpnKQLrWeGjlnXaBxhM342c1jnlCVmXss/uHMqeang\n\nsVhPTynikyXIrDwl9gh/2XlEwzVzpAnUBT
UZvJ9rgrceC9GcuGmiPZgxxSruNb0w\n\nK8xsyH8/hLwhK4Axgu3a+lfmmKFmSWjywmcxlmQl+jwiMPA/Ty55
AgMBAAEWdQYJ\n\nKoZIhvcNAQELBQADggEBAEG0D/2FcNU6Z6w/6eKbyH855kHSrJyqeU8eoCW9rnOb\n\nqdnA
sFX3aYwiUCjzSFXpWA3bRr3L7X0Y01x7VSvwiTuDvwO43llK29rQfrSvoPiw\n\nf7fhU7bszlUc2GAumU9OEd
YBnSi1DzfBawUcPmbDmm+ci27k0po53KDWTbxbkBIZR\n\n20h25LXkmq8ZBZE4vgS+mAw436nToazB1/vDTMwo
BuLVzOUlU8cdcJJUnJBvTbX\n\nThP69lsHVMED8B8Fhl08BzIJmQQ9qplTjplFq1Ea9oEFnT5U5gKvJYy48q
EPlW+r\n\nhRIHysvZXF/dghtrLXDMSBWLlLofUsDQsh+qLxpol+k=\n\n-----END CERTIFICATE-----\n",
"cert-server": "-----BEGIN CERTIFICATE-----
\nMIIDHDCCAgQCAQEWdQYJKoZIhvcNAQELBQAwVDELMAkGA1UEBhMCSU4xCzAJBgNV\n\nBAGMAktBMQwwCgYD
VQQHDANCTFlixCzAJBgNVBAoMAlBOMQwwCgYDVQQLDANFTkcx\n\nDzANBgNVBAMMB1NQSU5FMTAeFw0yMDA1MD
QxODM4NTZaFw0yMTA1MDQxODM4NTZa\n\nMFQxQzAJBgNVBAYTAklOMQswCQYDVQQIDAJLQTEEMMAoGA1UEBwwD
QkxSMQswCQYD\n\nVQQKDAJQTjEMMAoGA1UECwwDRU5HMQ8wDQYDVQQDDAZTUe1ORTewggEiMA0GCSqG\n\nsIb3
DQEBAQUAA4IBDwAwggEKAoIBAQCot16ddH0LNHOrWZt63FHYiArVXYIYbCDh\n\nWCY6MX3suoXYvKstvRgJkU
e/6G5As+vYtwRi2bDqdDsgTC5+Qo4SnjrdTcTM98F+
\n\n0Qzqv02c+dbzk5GclUkljqQ0PHGXRPgOMhou8B/6LI9Hg5XkG+FSfaDGQTM39uj2\n\nzzvrMOFn96gzpTBo
h40sMoIpnKQLrWeGjlnXaBxhM342c1jnlCVmXss/uHMqeang\n\nsVhPTynikyXIrDwl9gh/2XlEwzVzpAnUBT
UZvJ9rgrceC9GcuGmiPZgxxSruNb0w\n\nK8xsyH8/hLwhK4Axgu3a+lfmmKFmSWjywmcxlmQl+jwiMPA/Ty55
AgMBAAEWdQYJ\n\nKoZIhvcNAQELBQADggEBAEG0D/2FcNU6Z6w/6eKbyH855kHSrJyqeU8eoCW9rnOb\n\nqdnA
sFX3aYwiUCjzSFXpWA3bRr3L7X0Y01x7VSvwiTuDvwO43llK29rQfrSvoPiw\n\nf7fhU7bszlUc2GAumU9OEd
YBnSi1DzfBawUcPmbDmm+ci27k0po53KDWTbxbkBIZR\n\n20h25LXkmq8ZBZE4vgS+mAw436nToazB1/vDTMwo
BuLVzOUlU8cdcJJUnJBvTbX\n\nThP69lsHVMED8B8Fhl08BzIJmQQ9qplTjplFq1Ea9oEFnT5U5gKvJYy48q
EPlW+r\n\nhRIHysvZXF/dghtrLXDMSBWLlLofUsDQsh+qLxpol+k=\n\n-----END CERTIFICATE-----\n"}'
```

Note: Unnecessary use of `-X` or `--request`, `POST` is already inferred.

```
* Trying 10.100.64.5...
* TCP_NODELAY set
* Connected to 10.100.64.5 (10.100.64.5) port 80 (#0)
* Server auth using Basic with user 'network-admin'
> POST /vRest/web-certs/upload HTTP/1.1
> Host: 10.100.64.5
> Authorization: Basic bmV0d29yaylhZG1pbj0ZXN0MTIz
> User-Agent: curl/7.54.0
> Accept: */*
> content-type:application/json
> Content-Length: 2348
```

```
> Expect: 100-continue
>
< HTTP/1.1 100 Continue
* We are completely uploaded and fine
< HTTP/1.1 200 OK
< Server: Apache-Coyote/1.1
< Access-Control-Allow-Origin: *
< Access-Control-Allow-Methods: GET, POST, DELETE, PUT
< Set-Cookie: JSESSIONID=C52C3170DEEAC8E4996FF428D152BF25; Path=/vRest/; HttpOnly
< Date: Tue, 05 May 2020 19:34:05 GMT
< Content-Type: application/json
< Content-Length: 162
<
* Connection #0 to host 10.100.64.5 left intact

{"result":{"status":"Success","result":[{"api.switch-
name":"local","scope":"local","status":"Success","code":0,"message":"Successfully
uploaded certificates."}]}}
```

## Running Shell Commands or Scripts Using REST API

Netvisor ONE version 5.1.0 provides the ability to run shell commands or scripts using REST API or through CLI commands. As a network administrator or as an admin user, you can run the scripts from the directories `/opt/nvOS/bin/pn-scripts` (directory and all files are delivered as part of pn-upgrade-agent package) and `/usr/bin/pn-scripts` (backup directory for running custom scripts).

The commands introduced to enable this feature are: `pn-script-show` (to view all the available scripts) and `pn-script-run name <script-name>` (to run a specified script).

**Usage Guidelines:** To run a custom script,

- You should have permission to run the script.
- You should not have any duplicate scripts in the directories, `/opt/nvOS/bin/pn-scripts` and `/usr/bin/pn-scripts`. In case of duplicate scripts, the script from the directory, `/opt/nvOS/bin/pn-scripts` takes precedence.
- It is not recommended to execute any scripts that are manually copied to the directory.

You can use the CLI commands or the vREST API to run the scripts. To run the scripts using the CLI commands, for example:

To display the available scripts, use the command:

```
CLI (network-admin@pn-lab1) > pn-script-show
```

```
switch: pn-lab1
pn-lab1: /opt/nvOS/bin/pn-scripts/:
testscript.sh
pn-testscript.sh
```

To run the script, use the command:

```
CLI (network-admin@pn-lab1) > pn-script-run name testscript.sh
```

```
Executing /opt/nvOS/bin/pn-scripts/testscript.sh:
Executing Test PN script!
```

To run the scripts using vREST API, use the following API call:

```
$ curl -u network-admin:test123 http://pn-lab1/vRest/api-docs/run-pn
```

```
{ "apiVersion": "1.0.0", "swaggerVersion": "1.2", "basePath": "/vRest", "resourcePath": "/run-pn", "produces": [ "application/json", "application/x-ndjson" ], "consumes": [ "application/json" ], "apis": [ { "path": "/run-
```



```
pn/script", "operations":
[{"method": "POST", "summary": "", "notes": "", "type": "result-
list", "nickname": "scriptRun", "consumes": ["application/json"], "parameters":
[{"name": "body", "required": false, "type": "run-pn-
script", "paramType": "body", "allowMultiple": false}]]}], "models": {"result":
{"id": "result", "required": ["api.switch-
name", "scope", "status", "code"], "properties": {"api.switch-name":
{"type": "string"}, "scope": {"type": "string", "enum":
["local", "fabric"]}, "status": {"type": "string", "enum":
["Success", "Failure"]}, "code":
{"type": "integer", "format": "int32"}, "message": {"type": "string"}}}, "result-
list": {"id": "result-list", "required": ["status", "result"], "properties":
{"status": {"type": "string", "enum": ["Success", "Failure"]}, "result":
{"type": "array", "items": {"$ref": "result"}}}], "run-pn-script": {"id": "run-pn-
script", "description": "Run PN script", "required": ["name"], "properties":
{"name": {"type": "string", "description": "desc=Script to execute:pattern=[a-
zA-Z0-9_.-]+$:pattern-help=letters, numbers, _, ., :, and -"}}}]}
```

```
$ curl -u network-admin:test123 http://pn-lab1/vRest/pn-script
```

```
{ "data": [{}], "result": { "status": "Success", "result": [ { "api.switch-
name": "local", "scope": "local", "status": "Success", "code": 0, "message": "/opt/n
vOS/bin/pn-scripts/: \ntestscript.sh\n\n/usr/bin/pn-scripts/:
\ntestscript.sh\ntest-usr.sh\n" } ] } }
```

```
$ curl -u network-admin:test123 -X POST http://pn-lab1/vRest/pn-script/run
-d '{"name": "testscript.sh"}' -H "Content-Type: application/json"
```

```
{ "result": { "status": "Success", "result": [ { "api.switch-
name": "local", "scope": "local", "status": "Success", "code": 0, "message": "Execut
ing /opt/nvOS/bin/pn-scripts/testscript.sh:\nExecuting Test PN script!
\n" } ] } }
```

## Managing RMAs for Switches

---

A primary case for an RMA is a failed switch in the network. The configuration can be restored to a replacement switch using the following commands:

- fabric-join
- fabric-join repeer-to-cluster-node
- switch-config-import

Before initiating an RMA process, please gather and provide the following details to the Technical Support team:

- Product Serial Number: This can be found at the bottom of the switch or by using the following CLI command

```
CLI (network-admin@swich > switch-info-show
model:                F64-FL1T
chassis-serial:       1510LS9000xxx
cpu1-type:            Intel(R) Xeon(R) CPU E5-2620 v2 @ 2.10GHz
cpu2-type:            Intel(R) Xeon(R) CPU E5-2620 v2 @ 2.10GHz
system-mem:           128G
switch-device:        ok
polaris-device:       ok
gandalf-version:      0xcafff0044
```

- Company/Account Name: Please provide the details
- Contact Details:
  - Full Name
  - Email Address
  - Phone Number
  - Timezone
- Shipping Address:
- Existing Case Details (if Any):

For more details on the RMA process, contact Pluribus Technical Support team.

## Contacting Technical Assistance for Troubleshooting Purposes

While configuring and using the Netvisor ONE fabric, you can contact the Technical Assistance Team for support. Before you contact the TAC team, gather all relevant details regarding the issue.

Use the `tech-support-show` command to view all details of the running configuration that can help with TAC troubleshooting assistance. You can save and export the log file using SFTP with TAC team.

```
CLI (network-admin@switch) > tech-support-show
```

```
Netvisor OS Command Line Interface 3.1
Connected to Switch leafsw01.xyz; nvOS Identifier:0xb000d95; Ver:
3.1.3010113816
===== admin-service-show =====
if    ssh nfs web web-ssl web-ssl-port web-port snmp net-api icmp
----  -
mgmt on  on  off off      443          80          on   on      on
data on  on  off off      443          80          on   on      on
===== admin-session-timeout-show =====
timeout: 10m
===== admin-sftp-show =====
sftp-user:      sftp
enable:         yes
===== cluster-bringdown-show =====
vlag-port-staggered-interval: 0s
===== cluster-bringup-show =====
state:                                ports-enabled
l3-port-bringup-mode:                 staggered
l3-port-staggered-interval:           3s
vlag-port-bringup-mode:               staggered
vlag-port-staggered-interval:         3s
maximum-sync-delay:                   1m
l3-to-vlag-delay:                     15s
l3-to-vlan-interface-delay:           0s
port-defer-bringup-delay:             30s
port-defer-bringup-mode:              staggered
port-defer-bringup-staggered-interval: 0s

.
<snip>
.
===== vxlan-stats-settings-show =====
enable:      yes
interval:    30m
disk-space:  50M
diags@jerry:/build/diags/name$
```



## Configuring Switch Ports

---

This section contains information about the configuration of your switch as well as commands to manage and maintain configurations.

---

- [Displaying Port Numbering](#)
  - [Displaying Physical Port Layer 1 and Layer 2 Information](#)
  - [Displaying Port Status](#)
  - [Displaying Port Statistics](#)
  - [Configuring Ports for Different Throughput](#)
  - [Configuring Port Storm Control](#)
  - [Optimizing Port Link Status Detection on Pluribus Switches](#)
  - [Displaying Transceiver Information](#)
  - [Restoring Ports for Cluster Configurations](#)
  - [Limiting the Number of MAC Addresses per Port](#)
  - [Using Port Buffering](#)
  - [Changing Class of Service \(CoS\) Behavior](#)
  - [Configuring Minimum and Maximum Bandwidth on Ports](#)
  - [Enabling Jumbo Frame Support](#)
-

## Displaying Port Numbering

---

To display port numbering on white box Netvisor One platforms, use the `bezel-portmap-show` command:

```
CLI (network-admin@Leaf1) > bezel-portmap-show
```

switch	port	bezel-intf
-----	----	-----
Leaf1	1	1
Leaf1	2	2
Leaf1	3	3
Leaf1	4	4
Leaf1	5	5
...		
Leaf1	49	49
Leaf1	50	49.2
Leaf1	51	49.3
Leaf1	52	49.4
Leaf1	53	50
Leaf1	54	50.2
Leaf1	55	50.3
Leaf1	56	50.4

## Displaying Physical Port Layer 1 and Layer 2 Information

---

Display physical port information at Layer 2 using the `port-phy-show` command.

The command displays information about the default VLAN, link quality, maximum frame size, Ethernet mode, speed, and status.

Display the default VLAN for a port using:

```
CLI (network-admin@Leaf1) > port-phy-show
```

port	state	speed	eth-mode	max-frame	learning	def-vlan
17	up	1000	1000base-x	1540	on	1
19	up	10000	10Gbase-cr	10232	on	1

## Displaying Port Status

---

Use the `port-show` command to display status information on all ports with active links.

Details for each port include the IP addresses and MAC addresses of hosts connected to that port.

There can be more than one host if a network device such as a switch is connected.

The command also displays the VLAN of the port, port status, and configuration details.

To display all port information for ports 1-6 on the switch, use the command, `port-show port 1-6`:

```
CLI (network-admin@Leaf1) > port-show port 1-6
```

switch	port	bezel-port	vnet	l2-net	hostname	status	config
-----	----	-----	----	-----	-----	-----	-----
Leaf1	1	1					10g
Leaf1	2	2					10g
Leaf1	3	3					10g
Leaf1	4	4					10g
Leaf1	5	5					10g
Leaf1	6	6					10g



## Displaying Port Statistics

---

Display statistics for all active ports on the switch. This information is useful for understanding the traffic on the ports.

Use the `port-stats-show` command to display the information:

```
CLI (network-admin@Leaf1) > port-stats-show port 0 format all layout vertical
```

```
switch:      Leaf1
time:        11:32:41
port:        0
description:
counter:      0
ibytes:      2.82G
ibits:        24.2G
iUpkts:       176M
iBpkts:       0
iMpks:        0
iPauseFs:     0
iCongDrops:   0
idiscards:    7
ierrs:        0
obytes:       884M
obits:        7.42G
oUpkts:       13.0M
oBpkts:       0
oMpks:        0
oPauseFs:     0
oCongDrops:   1.89G
odiscards:    1.89G
oerrs:        0
mtu-errs:     0
HER-pkts:     0
HER-bytes:    0
port-speed:   disable
```

The output headers have the following meaning:

- **switch** — switch name
- **time** — the time that the command is issued
- **port** — port number
- **counter** — number of counters for the port
- **ibytes** — number of incoming bytes in K (Kilobytes), M (Megabytes), or G (Gigabytes)

- **iUpkts** — number of incoming unicast packets
- **iBpkts** — number of incoming broadcast packets
- **iMpkts** — number of incoming multicast packets
- **iPauseFs** — number of incoming paused fragmented packets
- **iCongDrops** — number of incoming packets dropped due to congestion
- **idiscards** — number of discarded incoming packets
- **ierrs** — number of incoming packets with errors
- **obytes** — number of outgoing bytes K (Kilobytes), M (Megabytes), or G (Gigabytes)
- **oUpkts** — number of outgoing unicast packets
- **oBpkts** — number of outgoing broadcast packets
- **oMpkts** — number of outgoing multicast packets
- **oPauseFs** — number of outgoing paused fragmented packets
- **oCongDrops** — number of outgoing packets dropped due to congestion
- **odiscards** — number of discarded outgoing packets
- **oerrs** — number of outgoing packets with errors
- **mtu-errs** — number of MTU errors
- **HER-pkts** — number of VLE port packets
- **HER-bytes** — number of VLE bytes
- **port-speed** — port speed in GB.

## Configuring Ports for Different Throughput

---

Netvisor ONE enables you to configure switch ports for different throughput values. You can use 10, 40, 100 Gigabits per second.

For configuring 40Gigabit Ethernet (GbE) ports, Netvisor allows you to use them as 4 x 10GbE with the right transceiver. That is four- 10 G ports. You must first disable the ports before flexing into multiple ports.

To change the 40GbE port to 4x10GbE functionality, use the following command sequence:

```
CLI (network-admin@Leaf1) > port-config-modify port 49-52 speed 10g
```

To change the port back to 40Gb operation, use the following command sequence:

```
CLI (network-admin@Leaf1) > port-config-modify port 49 speed 40g
```

Netvisor ONE sets the default port speed to 10G and you can modify the parameters of a port:

- **Interface number** — you can change the physical interface number.
- **Speed** — you can disable the port or set the speed to 10m, 100m, 1g, 2.5g, 10g, or 40g.
- **Egress rate** — limit the egress rate or set to unlimited.
- **Ethernet mode type** — set the mode type to 1000base-x, sgmi, autonegotiate
- **Auto-negotiation of physical port** — set to auto-negotiation or no auto-negotiation.
- **Jumbo frames** — set to jumbo frames or no jumbo frames on physical port.
- **Enable or disable** a port
- **LACP priority** — between 1 and 65535
- **Reflect** — received frames are reflected for loopback testing.
- **Edge-switch** — Specify if the port connects to another Netvisor ONE device or uplinks to a third-party switch or host.
- **Pause** — pause traffic on the port.
- **Description** — description of the port
- **Loopback** — specify loopback
- **VXLAN termination** — physical port vxlan termination setting.
- **Mirror-receive** — receive mirrored traffic only.
- **MAC address** — specify a MAC address for the port.
- **Sending port number** — specify if the port number sends traffic.

- **VLAN looping** — looping vlans
- **Routing** — routing or no on physical ports
- **Defer port** — defer bring-up
- **Host-enable or disable** — Host facing port control setting
- **CRC-check-enable or disable** — CRC check on ingress and rewrite on egress
- **DSCP-map** — DSCP map name to enable on port
- **Local-switching or no-local-switching** — no-local-switching port cannot bridge traffic to another no-local-switching port.
- **TPID VLAN** — Allowed TPID in addition to 0x8100 on Vlan header
- **Fabric-guard** — fabric guard configuration
- **FEC mode** — port forward error correction (FEC) mode

## Configuring Port Storm Control

---

Port Storm Control prevents traffic on a LAN from disruption by a broadcast, multicast, or unicast storm on a port. A LAN storm occurs when packets flood the LAN, creating excessive traffic and degrading network performance.

Use `port-storm-control-modify` to modify the percentage of total available bandwidth that can be used by broadcast, multicast, or unicast traffic.

```
CLI (network-admin@Leaf1) > port-storm-control-modify port 11 unknown-ucast-level 1.1
```

Use the `port-storm-control-show` command to display the configuration:

```
CLI (network-admin@Leaf1) > port-storm-control-show
```

port	speed	unknown-ucast-level	unknown-mcast-level	broadcast-level
----	-----	-----	-----	-----
1	40g	30%	30%	30%
2	10g	30%	30%	30%
3	10g	30%	30%	30%
4	10g	30%	30%	30%
5	40g	30%	30%	30%
6	40g	30%	30%	30%
7	10g	30%	30%	30%

## Optimizing Port Link Status Detection on Pluribus Switches

With Netvisor Version 5.1.0 and later, Netvisor ONE enables you to bypass unnecessary port link status checks for RXLOS-compliant devices installed on Pluribus switches.

On select switches such as AS5712-54XS and Pluribus F9272-X, the SFP ports in 1G mode (copper) take a longer time of approximately 20 seconds to come up when the link cable is plugged-in or unplugged. The reason for the delay in bringing up the SFP ports is caused due to Pluribus support for SFP adapter modules that convert an optical SFP port into a 1G copper link.

The Optical SFP transceivers use an RXLOS signal to indicate the link status, but some SFP 1G copper modules do not support the use of RXLOS signal. For such modules, a more complex process is required to determine the port link status. Netvisor supports SFP transceivers that are both RXLOS-compliant and RXLOS-non-compliant. When the number of SFP transceivers increases on a switch, the time taken to check every SFP module for determining the link status increases proportionately, causing a delay in reporting the port link status. This processing takes place regardless of whether the SFP transceiver is RXLOS-compliant or not.

Netvisor version 5.1.0 addresses this delay by adding a new parameter flag to the `system-settings-modify` command. You can use the `optimize-rxlos` parameter flag to enable or disable the support for RXLOS-non-compliant 1G SFP adapters on the switch. When the `optimize-rxlos` parameter flag is enabled, then Netvisor determines and reports the port link status with minimal delay. Use the command:

```
CLI (network-admin@sw-test) > system-settings-modify optimize-<tab>
```

<code>optimize-arp</code>	Specify to enable ARP optimization
<code>optimize-nd</code>	Specify to enable ND optimization
<code>optimize-datapath</code>	Specify the datapath optimization for cluster, fabric and data communication
<b><code>optimize-rxlos</code></b>	<b>Specify to disable processing of port modules that do not support RXLOS</b>

For example, to enable the feature of disabling the processing of port modules that do not support RXLOS, use the command:

```
CLI (network-admin@sw-test) > system-settings-modify optimize-rxlos
Setting optimize-rxlos to ON: Link state of SFP modules that do not support
RXLOS may not be properly detected
```

To see if the feature is enabled or disabled, use the command:

```
CLI (network-admin@sw-test) > system-settings-show format optimize-rxlos
optimize-rxlos: on
```

To disable the feature, use the command:

```
CLI (network-admin@sw-test) > system-settings-modify no-optimize-<tab>
```

no-optimize-arps	Specify to disable ARP optimization
no-optimize-nd	Specify to disable ND optimization
<b>no-optimize-rxlos</b>	<b>Specify to enable processing of port modules that do not support RXLOS</b>

For example,

```
CLI (network-admin@sw-test) > system-settings-modify no-optimize-rxlos
```

To view, use the command,

```
CLI (network-admin@sw-test) > system-settings-show format optimize-rxlos,  
optimize-rxlos: off
```

## Displaying Transceiver Information

---

Display information about the transceivers connected to the switch using the `port-xcvr-show` command:

```
CLI (network-admin@Spine1) > port-xcvr-show
```

switch	port	vendor-name	part-number	serial-number
-----	----	-----	-----	-----
Spine1	3	3M	1410-P17-00-0.50	
Spine1	4	3M	1410-P17-00-0.50	
Spine1	57	3M	9QA0-111-12-1.00	V10B9252
Spine1	65	3M	9QA0-111-12-1.00	V10B9614



## Restoring Ports for Cluster Configurations

---

**Note:** This feature is applied only during the initial start up of the network.

Sub-second traffic loss for fail over events is required for a cluster configuration. There are two types of ports providing redundant data paths: 1) Layer 3 ports over ECMP redundant routed paths, and 2) virtual LAGS (VLAGs) providing redundant Layer 2 paths. During failover and recovery port events, it can take measurable time to change the hardware routing and MAC tables on larger networks. This delay incurs traffic loss on the network. To reduce delay, this feature allows you to incrementally restore these ports at start up. By incrementally restoring the ports, the changes to the hardware are prevented from contending with each other and reduces the delay between a port up and the hardware updates with the appropriate Layer 3 and Layer 2 information for the port. This process ensures sub-second fail over.

All non-Layer 3 and non-VLAG ports are restored first. This allows the cluster links to activate and the cluster configuration to synchronize information. Layer 3 and VLAG port restoration starts after the cluster synchronizes. This is predicated on the cluster becoming active, all Layer 2 and Layer 3 entries, such as status updates, exchanged, cluster STP status synchronized, and all router interfaces initialized.

Netvisor ONE enforces the following sequence for port bring up (not guaranteed for first upgrade to nvOS, allowed only for subsequent reboots):

1. Cluster ports, VXLAN-loopback-trunk ports, and Loopback ports
2. Layer 3 ports
3. VLAG ports
4. All other ports

The sequence for port bring-down is :

1. Orphan ports (non-VLAG, non-Layer3, and non-Cluster)
2. VLAG ports
3. Layer 3 ports
4. All other ports

If a port is configured with `defer-bringup` parameter, then that port is brought up along with other ports. All ports except cluster ports can be configured for `defer-bringup` using the `port-config-modify` command. You can specify the global timer value for delaying the port bringup by using the `cluster-bringup-modify` command with `port-defer-bringup-delay duration` parameter.

The parameter, `maximum-sync-delay`, controls the maximum time to wait for synchronization in the case where the cluster cannot synchronize information. After synchronization is complete, Layer 3 ports are restored first, since Layer 3 traffic can traverse the cluster link to the peer VLAG port if needed. Currently the reverse is typically not true.

If VLAG ports are restored first, a Layer 3 adjacency between the two cluster nodes may be needed but may not exist in some network configurations. After Layer 3 ports are restored, Netvisor One waits a configurable Layer 3 port to VLAG delay to allow time for the routing protocols to converge and insert the routes. The delay time defaults to 15 seconds.

After the delay, the VLAG ports are restored incrementally. Incrementally restoring ports allows enough time to move Layer 2 entries from the cluster link to the port. Incrementally restoring ports also allows the traffic loss to occur in small, 200-300ms per port, rather than one large time span. This is particularly important for server clusters where temporary small losses are no issue, but fail or timeout for a large continuous traffic loss. If the node coming up is the cluster master, then no staggering and no Layer 3 to VLAG wait is applied. And if the node is the cluster master node, that means the peer is down or coming up, and not handling traffic. Therefore Netvisor One safely restores the ports as soon as possible to start traffic flowing between the nodes.

In addition, with version 5.1.1, Netvisor ONE supports the staggered bring up of vRouter VNICs. When configured, the vRouter VNICs, which are not configured on Layer3 ports, are brought up in a staggered manner during the nvOS boot-up on a cluster peer. You can specify the wait time (0-60000 ms) between NIC bring up.

By default, the vRouter interfaces for which the VLAN is up is brought up simultaneously. The bring up can be staggered by specifying a non-zero vrouter-if-staggered-interval. The staggered bring up process is helpful to reduce the traffic loss caused by the simultaneous bring up of all VNICs.

**Note:** The vRouter interfaces on Layer 3 ports are brought up first and is not staggered. Also, after bootup, the subsequent VLAN interfaces being down or up is not affected by the staggered configuration.

To configure or modify the port bring up process, use the command:

```
CLI (network-admin@Leaf1) > cluster-bringup-modify
```

<code>cluster-bringup-modify</code>	Modifies the cluster bring up configuration.
Specify one or more of the following options	
<code>l3-port-bringup-mode staggered simultaneous</code>	Specify the Layer 3 port bring up mode during start up.
<code>l3-port-staggered-interval duration: #d#h#m#s</code>	Specify the interval between Layer 3 ports in Layer 3 staggered mode. This can be in days, hours, minutes, or seconds.
<code>vlag-port-bringup-mode staggered simultaneous</code>	Specify the VLAG port bring up mode during start up.
<code>vlag-port-staggered-interval duration: #d#h#m#s</code>	Specify the interval between VLAG ports in VLAG staggered mode. This can be in days, hours, minutes, or seconds.
<code>maximum-sync-delay duration: #d#h#m#s</code>	Specify the maximum delay to wait for cluster to synchronize before starting Layer 3 or VLAG port bring up. This can be in days, hours, minutes, or seconds.
<code>l3-to-vlag-delay duration: #d#h#m#s</code>	Specify the delay between the last Layer 3 port and the first VLAG port bring up.

<code>l3-to-vlan-interface-delay</code>	This can be in days, hours, minutes, or seconds. The default value is 15 seconds.
<code>duration: #d#h#m#s</code>	
<code>port-defer-bringup-delay</code>	Specify the delay between the last Layer 3 port and the vRouter VLAN interface bring up.
<code>duration: #d#h#m#s</code>	Specify the global timer value to be used for port delay-bring up.
<code>port-defer-bringup-mode</code>	Specify the port defer bring up mode during start up.
<code>staggered simultaneous</code>	
<code>port-defer-bringup-staggered-interval</code>	Specify the interval between ports in defer bring up mode.
<code>duration: #d#h#m#s</code>	
<code>vrouter-if-bringup-mode</code>	Specify the vRouter VLAN interface bring up mode.
<code>staggered simultaneous</code>	
<code>vrouter-if-staggered-interval</code>	Specify the interval in ms between vRouter VLAN interface bring up in staggered mode. The value ranges between 0-60000 milli-seconds.
<code>0..60000</code>	

To display the status of the cluster bring up process, use the `cluster-bringup-show` command:

```
CLI (network-admin@Leaf1) > cluster-bringup-show
```

```
switch:           Leaf1
state:            ports-enabled
l3-port-bringup-mode: staggered
l3-port-staggered-interval: 3s
vlag-port-bringup-mode: staggered
vlag-port-staggered-interval: 3s
maximum-sync-delay: 1m
l3-to-vlag-delay: 15s
l3-to-vlan-interface-delay: 0s
port-defer-bringup-delay: 30s
port-defer-bringup-mode: staggered
port-defer-bringup-staggered-interval: 0s
vrouter-if-bringup-mode: staggered
vrouter-if-staggered-interval(ms): 0
```

To display the status of ports with defer-bringup details on select ports, use the command:

```
CLI (network-admin@Leaf1) > port-config-show format port,cluster-port,defer-bringup,vlag, port 11,15,42,49
```

```
port cluster-port defer-bringup vlag
----
11    no          no          host-vlag
15    no          yes
42    yes         no
49    no          no
```

To view the details of a specified port, use the command:

```
CLI (network-admin@Leaf1) > port-show port 49
```

port	ip	mac	vlan	hostname	status	config
49	50.50.50.2	66:0e:94:b8:a3:72	4092	leaf1	up,PN-fabric,LLDP,l3-port,remote-l3-port,vlan-up	fd,10g

## Limiting the Number of MAC Addresses per Port

You can now limit the number of MAC addresses per port. You can configure port security on ports or trunks.

### New Commands

```
CLI (network-admin@Leaf1) > mac-limit-modify
```

<code>port <i>port-list</i></code>	Specify the port list.
<code>mac-limit <i>mac-limit-number</i></code>	Specify the number of MAC addresses to limit on the port.
<code>mac-limit-action log disable</code>	Specify the action to take when the MAC address limit is exceeded. If you select log, an event is logged to the event log. If you specify disable, the event is logged and the port is disabled.

```
CLI (network-admin@Leaf1) > mac-limit-show
```

<code>port <i>port-list</i></code>	Displays the port list.
<code>mac-limit <i>mac-limit-number</i></code>	Displays the number of MAC addresses to limit on the port.
<code>mac-limit-action log disable</code>	Displays the action taken when the MAC address limit is exceeded.
<code>num-macs <i>num-macs-number</i></code>	Displays the number of MAC addresses learned on the port.

```
CLI (network-admin@Leaf1) > mac-limit-show
```

switch	port	mac-limit	mac-limit-action	num-macs
Leaf02	5	0	log	0
Leaf03	5	0	log	0

## Using Port Buffering

---

Netvisor ONE enables you to display and modify the port buffering settings for the switch ports. To display the port buffering settings, use the `port-buffer-settings-show` command:

```
CLI (network-admin@Leaf1) > port-buffer-settings-show
```

```
switch: Spine1
enable: yes
interval: 1m
disk-space: 50M
```

To modify port buffering settings, use the `port-buffer-settings-modify` command:

```
CLI (network-admin@Leaf1) > port-buffer-settings-modify interval 2m
```

You can modify the buffer interval, duration, disk space, and enable or disable port buffering on the switch.

To display the port buffer, use the `port-buffer-show` command:

```
CLI (network-admin@Leaf1) > port-buffer-show
```

```
switch: Leaf1
port: 0
ingress-used-buf: 0%
ingress-used-buf-val: 0
egress-used-buf: 0%
egress-used-buf-val: 0
switch: Leaf1
port: 3
ingress-used-buf: 0%
ingress-used-buf-val: 0
egress-used-buf: 0%
egress-used-buf-val: 0
switch: Leaf2
port: 57
ingress-used-buf: 0%
ingress-used-buf-val: 0
egress-used-buf: 0%
egress-used-buf-val: 0
switch: Leaf1
port: 65
ingress-used-buf: 0%
ingress-used-buf-val: 0
egress-used-buf: 0%
egress-used-buf-val: 0
switch: Leaf2
```

```
port: 0
ingress-used-buf: 0%
ingress-used-buf-val: 0
egress-used-buf: 0%
egress-used-buf-val: 0
switch: Leaf2
port: 1
ingress-used-buf: 0%
ingress-used-buf-val: 0
egress-used-buf: 0%
egress-used-buf-val: 0
```

## Changing Class of Service (CoS) Behavior

Netvisor One automatically weights Class of Server (CoS) queues with minimum bandwidth guarantees. When you configure minimum bandwidth on a port queue using CoS, the remaining bandwidth is assigned to the rest of the queues in the same ratio as the minimum bandwidth.

You can use the command, `port-cos-bw-modify`, to change the queue weight of ports on the front panel and use the command, `port-cos-weight-modify`, to change the queue weight of ports on the control panel (PCIe and CPU ports).

When you configure a minimum bandwidth without specifying a weight value, the weight for the port and CoS is automatically set on a scale of 1 to 100.

For example, if you configure the minimum bandwidth as 10%, Netvisor One automatically assigns the queue a weight value of 1. You can also assign a specific weight value to the queue.

Additionally, you can configure strict priority scheduling for any of the queues. By default, CoS 7 assigns strict priority scheduling for the queue.

To allow automatic weight assignment for CoS queues, use the following syntax:

```
CLI (network-admin@Spine-1) > system-settings-modify cosq-weight-auto|no-cosq-weight-auto
```

<code>cos integer</code>	Specify the CoS priority between 0 and 7.
<code>port port-list</code>	Specify the physical port(s).
<code>min-bw-guarantee min-bw-guarantee-string</code>	Specify the minimum bandwidth as a percentage.
<code>max-bw-limit max-bw-limit-string</code>	Specify the maximum bandwidth as a percentage.
<code>weight no-weight</code>	Specify if the scheduling weight after the bandwidth guarantee is met.

```
CLI (network-admin@Spine-1) > port-bw-show
```

switch	cos	port	min-bw-guarantee	max-bw-limit	weight
Spine-1	0	1-72	0%	100%	0
Spine-1	1	1-72	0%	100%	0
Spine-1	2	1-72	0%	100%	0
Spine-1	3	1-72	0%	100%	0
Spine-1	4	1-72	0%	100%	0
Spine-1	5	1-72	0%	100%	0
Spine-1	6	1-72	0%	100%	0
Spine-1	7	1-72	0%	100%	0



To auto-configure bandwidth, use the following syntax:

```
CLI (network-admin@Spine-1) > port-cos-bw-modify cos 1 port 1 min-bw-guarantee 20
```

```
CLI (network-admin@Spine-1) > port-cos-bw-show
```

switch	cos	port	min-bw-guarantee	max-bw-limit	weight
Spine-1	0	1-72	0%	100%	0
Spine-1	1	2-72	0%	100%	0
Spine-1	1	1	20%	100%	2
Spine-1	2	1-72	0%	100%	0
Spine-1	3	1-72	0%	100%	0
Spine-1	4	1-72	0%	100%	0
Spine-1	5	1-72	0%	100%	0
Spine-1	6	1-72	0%	100%	0
Spine-1	7	1-72	0%	100%	0

To configure a specific weight, use the following syntax:

```
CLI (network-admin@Spine-1) > port-cos-bw-modify cos 1 port 1 weight 6
```

```
CLI (network-admin@Spine-1) > port-cos-bw-show
```

switch	cos	port	min-bw-guarantee	max-bw-limit	weight
Spine-1	0	1-72	0%	100%	0
Spine-1	1	2-72	0%	100%	0
Spine-1	1	1	20%	100%	6
Spine-1	2	1-72	0%	100%	0
Spine-1	3	1-72	0%	100%	0
Spine-1	4	1-72	0%	100%	0
Spine-1	5	1-72	0%	100%	0
Spine-1	6	1-72	0%	100%	0
Spine-1	7	1-72	0%	100%	0

## Configuring Minimum and Maximum Bandwidth on Ports

This feature introduces bandwidth guarantees on switch ports. Currently, Pluribus Networks switches allow rate limiting only for CPU facing (PCIe, data and span) ports. Using this feature, you configure bandwidth guarantees at egress CoS (Class of Service) queue level and manage prioritized traffic. Use this feature for setting SLAs (Service Level Agreements). Currently, Netvisor ONE provides maximum bandwidth policing at the vFlow level, but you cannot set guaranteed minimum bandwidth. This feature addresses the limitation.

Switch hardware supports minimum and maximum bandwidth guarantees configured at a port and CoS queue level. Netvisor ONE supports a per port configuration. Netvisor ONE allows the settings as a percentage of port speed, and determines the data rate internally on command execution. Additionally, you update port speed, the port configuration internally re-adjusts the minimum or maximum bandwidth rates for the applicable ports. The `port-config-show` command displays 100% allocations by default. Netvisor ONE displays new configurations as additional elements, sorted by CoS queue.

Netvisor ONE displays ONLY modified port configurations.

Ports not displayed in the show command output default to the settings 100% link capacity, and no minimum guarantee for each CoS queue.

```
CLI (network-admin@Spine1) > port-cos-bw-modify
```

<code>cos integer</code>	Specify the CoS priority between 0 and 7.
<code>port port-list</code>	Specify the physical port(s).
<code>min-bw-guarantee min-bw-guarantee-string</code>	Specify the minimum bandwidth as a percentage.
<code>max-bw-limit max-bw-limit-string</code>	Specify the maximum bandwidth as a percentage.

```
CLI (network-admin@Spine1) > port-cos-bw-show
```

<code>cos integer</code>	Specify the CoS priority.
<code>port port-list</code>	Specify the physical port(s).

```
CLI (network-admin@Spine1) > port-cos-bw-modify port 2-5 cos 5 min-bw-guarantee 10
```

```
CLI (network-admin@Spine1) > port-cos-bw-show
```

switch	cos	port	min-bw-guarantee	max-bw-limit	weight
Spine1	0	1-72	0%	100%	16
Spine1	1	1-72	0%	100%	32
Spine1	2	1-72	0%	100%	32

Spine1	3	1-72	0%	100%	32
Spine1	4	1-72	0%	100%	32
Spine1	5	1,6-72	0%	100%	32
Spine1	5	2-5	10%	100%	32
Spine1	6	1-72	0%	100%	64
Spine1	7	1-72	0%	100%	127

```
CLI (network-admin@Spine1) > port-cos-bw-modify port 2-511,13 cos 4 min-bw
20 max-bw 80
```

```
CLI (network-admin@Spine1) > port-cos-bw-show
```

switch	cos	port	min-bw	max-bw
-----	----	-----	-----	-----
Spine1	0	0-72	100%	100%
Spine1	1	0-72	100%	100%
Spine1	2	0-72	100%	100%
Spine1	3	0-72	100%	100%
Spine1	4	0-1,11-72	100%	100%
Spine1	4	11-13	20%	80%
Spine1	5	2-10	10%	100%
Spine1	6	0-72	100%	100%
Spine1	7	0-72	100%	100%

Changing the port settings to new values overrides the previous settings.

## Enabling Jumbo Frame Support

---

Jumbo frames are (oversized) Ethernet frames with a size greater than 1518 bytes (including the Ethernet header and FCS) or 1522 bytes (when a VLAN tag is present).

Different vendors may support different maximum frame sizes for jumbos, as the IEEE standard does not cover jumbo frames (except for baby giants). Typically jumbo frames are described with the maximum payload length they support, namely, the Maximum Transmission Unit (MTU). It is common for platforms (including servers) to support jumbo frames with (at least) an MTU of 9K bytes. 1K usually corresponds to 1024 (not 1000), so that means that the MTU is 9216 bytes.

When you enable the jumbo frame feature on a port on Pluribus switches, the port can accept and forward jumbo frames. This feature is meant to optimize server-to-server performance (by minimizing the CPU processing overhead for large block transfers).

Netvisor ONE has jumbo frame support disabled by default, so the default Ethernet MTU for all switch ports is 1500 bytes. When you enable the jumbo frame feature on a port, the MTU size is increased to accept 9K byte frames on that port.

The larger MTU supported on Pluribus switches helps on transport links to be able to deal with the transport overhead (for example, with the VXLAN and vLE functionalities described later in this document).

To enable jumbo frame support, add the `jumbo` parameter to the `port-config-modify` command:

```
CLI (network-admin@Leaf1) > port-config-modify port 1 jumbo
```

To enable jumbo frame support on trunk ports, add the `jumbo` parameter to the `trunk-modify` (or `trunk-create`) commands:

```
CLI (network-admin@Leaf1) > trunk-modify name trunk1 jumbo
```

## Configuring Layer 2 Features

---

This chapter provides information for understanding and using the Pluribus Networks Netvisor One command line interface (CLI) on a Netvisor ONE switch to configure Layer 2 features.

---

- [Understanding the Supported L2 Protocols](#)
  - [Configuring LLDP](#)
  - [Understanding and Configuring VLANs](#)
  - [Configuring Rapid Spanning Tree Protocol \(RSTP\)](#)
  - [Configuring Multiple Spanning Tree Protocol \(MSTP\)](#)
  - [Achieving a Loop-Free Layer 2 Topology](#)
  - [Fast Failover for STP and Cluster](#)
  - [Configuring Auto-Recovery of a Disabled Port](#)
  - [Configuring STP Root Guard](#)
  - [Configuring Fabric Guard](#)
  - [About Hardware Hashing](#)
-

## Understanding the Supported Layer 2 Protocols in Netvisor ONE

---

Layer 2 (the Data Link layer) enables the transfer of data between adjacent nodes in a network segment, such as local or wide area networks. Layer 2 frames do not cross the boundaries of the local network. Services provided by Layer 2 include, but are not limited to: frame encapsulation, device addressing, error detection, packet forwarding, loop prevention, flow control, frame queuing, Quality of Service (QoS), and traffic segmentation through Virtual LANs (VLANs).

Netvisor ONE supports the following Layer 2 protocols and functionalities:

- Spanning Tree Protocol (STP)
- Rapid Spanning Tree Protocol (RSTP)
- Multiple Spanning Tree Protocol (MSTP)
- Link Aggregation (LAG)
- virtual Link Aggregation (vLAG)
- Link Aggregation Control Protocol (LACP)
- Link Layer Discovery Protocol (LLDP)
- Root Guard
- Fabric Guard

## Configuring LLDP

The Link Layer Discovery Protocol (LLDP) is an open, vendor-independent protocol that advertises a device's identity, abilities, and neighboring devices connected within the Local Area Network. This protocol is based on IEEE 802.1ab standard. An LLDP device sends information as Ethernet frames at regular intervals and each frame contains one LLDP Data Unit (LLDPDU) which is a sequence of different Type-Length-Value (TLV) structures. These frames start with the mandatory TLVs that include the Chassis ID, Port ID, and Time-To-Live (TTL) followed by a number of optional TLVs.

Netvisor One provides a generic LLDP ON/OFF toggle function set at the system level. LLDP is enabled on a switch by default and without the generic function, you must disable LLDP configuration on all ports to disable LLDP on a switch. This resets all related configurations of LLDP protocol settings at the port level and LLDP vFlows. Use the following CLI command to enable or disable the protocol at the system level:

```
CLI (network-admin@leaf1) > system-settings-modify [lldp|no-lldp]
```

LLDP packets are processed on the CPU after being directed there by specific vFlow policies. To clear all LLDP protocol system redirects, use the parameter `no-lldp`. To add all LLDP protocol system redirects, use the parameter `lldp`. This approach ensures that port LLDP configurations are not disturbed.

To verify the LLDP status, use the command:

```
CLI (network-admin@leaf1) > system-settings-show
switch:                leaf1
optimize-arps:         off
lldp:                off
policy-based-routing:  off
optimize-nd:           off
reactivate-mac:        on
```

To enable/disable LLDP on one or more ports, use the command `port-lldp-modify`.

```
CLI (network-admin@Leaf1) > port-lldp-modify
```

<code>port-lldp-modify</code>	Modify the LLDP setting on ports.
Specify the following options:	
<code>port <i>port-list</i></code>	Specify the ports on which LLDP has to be enabled or disabled as a list separated by commas.
<code>lldp no-lldp</code>	Specify the option to enable or disable LLDP.

```
CLI (network-admin@switch) > port-lldp-modify port <port-list> [lldp|no-lldp]
```

`lldp-show` command displays the LLDP information of the neighbors in the network.

```
CLI (network-admin@Leaf1) > lldp-show
```

<code>lldp-show</code>	This command displays LLDP information.
------------------------	---

Specify any of the following options to view the LLDP information specific to that parameter.

<code>local-port</code> <i>local-port-number</i>	Specify a local port number.
<code>chassis-id</code> <i>chassis-id-string</i>	Specify the chassis ID of the neighboring device.
<code>port-id</code> <i>port-id-string</i>	Specify the port ID of the neighboring device.
<code>sys-name</code> <i>sys-name-string</i>	Specify the system name of the neighboring device.

For example:

```
CLI (network-admin@SW1) > lldp-show
```

switch	local-port	chassis-id	port-id	port-desc	sys-name
SW1	17	090009ef	17	PN Switch Port(17)	proto-1
SW1	121	090009ee	121	PN Switch Port(121)	proto-2
SW2	13	090009ef	13	PN Switch Port(13)	proto-1
SW2	117	090009ee	117	PN Switch Port(117)	proto-2

`port-lldp-show` command displays the LLDP status on a port basis.

```
CLI (network-admin@Leaf1) > port-lldp-show
```

<code>port-lldp-show</code>	Display LLDP configuration on the ports. This command when executed without any parameters displays the LLDP status (on/off) of all the ports on the switch.
-----------------------------	--

Specify any of the following parameters to view the LLDP information specific to that parameter.

<code>port</code> <i>&lt;port-list&gt;</i>	Specify the list of ports separated by commas.
<code>lldp no-lldp</code>	Use the option <code>lldp</code> to view all the ports on which LLDP is enabled. Use the option <code>no-lldp</code> to view all the ports on which LLDP is disabled.

For example:

```
CLI (network-admin@Leaf1) > port-lldp-show port 12,13,14,15
```

switch	port	lldp
Leaf1	12	on
Leaf1	13	on
Leaf1	14	on
Leaf1	15	on



## Understanding and Configuring VLANs

A Virtual Local Area Network (VLAN) enables devices to be segmented into logically separate broadcast domains within the same LAN. VLANs improve network performance by directing network traffic only to the parts of the network that need to receive it. Network segments so created keep traffic isolated based on the respective VLAN IDs associated to the transmitted frames. Applying targeted security features to specific network areas is also made simpler through the use of VLANs.

As per the standards, Netvisor ONE uses the 12-bit field in the header of each packet as a VLAN identifier or VLAN tag. The maximum number of VLANs that can be defined is 4092. VLANs 4093, 4094, and 4095 are reserved for internal use while VLAN 1 is the default fabric VLAN for untagged traffic. Untagged packets can be mapped to any VLAN, but Netvisor ONE maps this traffic to VLAN 1 by default.

Configuring an untagged VLAN is necessary while connecting a switch to devices that do not support IEEE80.1Q VLAN tags. The ports on a switch can be configured to automatically map untagged packets to a specific VLAN. Netvisor ONE also allows you to block untagged traffic on a port basis, that is, the untagged VLAN on a port can be removed or deleted.

### About VLAN 1

- VLAN 1 is enabled on all ports by default. However, VLAN 1 can be removed from any port on which it is the untagged VLAN. Now, the port has no untagged VLANs and all untagged traffic is dropped on that port.
- To generalize the point above, if VLAN x is the untagged VLAN for a port and if VLAN x is removed from that port, then the port has no untagged VLAN and all untagged traffic is dropped on that port.
- VLAN 1 can also function as a tagged VLAN for a port. This happens automatically in cases where VLAN 1 is the default untagged VLAN, and then another VLAN is configured as an untagged VLAN on the port.
- VLAN 1 cannot be created or deleted. VLAN 1 configuration is stored in persistent storage.

The default fabric VLAN can be changed from VLAN 1 to another VLAN ID using the command `fabric-local-modify`. For example, to set VLAN 20 as the default fabric VLAN, use the command:

```
CLI (network-admin@Leaf1) > fabric-local-modify vlan 20
```

**Warning:** If you create a VLAN with scope fabric and configure it as the untagged VLAN on all ports, it can disrupt the fabric communication.

**Note:** The untagged VLAN feature is not the same as the default VLAN using the IEEE 802.1Q tag 1.

The `vlan-create` command creates VLANs on the current switch.

```
CLI (network-admin@Leaf1) > vlan-create
```

`vlan-create`

Creates a VLAN. You can create a VLAN either by specifying a VLAN ID or by specifying a range of VLAN IDs.

<code>id 2...4092</code>	Specify the VLAN ID between 2 and 4092. Note: VLAN 0 and 1 represents all untagged or non-VLAN traffic, VLANs 4093, 4094, and 4095 are reserved for internal use.
<code>range vlan-list</code>	Specify the range of VLAN IDs. Use this parameter if you want to specify a VLAN range instead of a VLAN ID.
<code>scope [local cluster fabric]</code>	Specify the VLAN scope as local, cluster, or fabric.
Specify any of the following options:	
<code>vnet vnet-name</code>	Specify the vNET name for this VLAN. Note: A vNET segregates a physical fabric into many logical networks, each with separate resources, network services, and Quality of Service (QoS) guarantees.
<code>vxlan 1..16777215</code>	Specify the VXLAN identifier for the tunnel.
<code>vxlan-mode [standard transparent qinq-access]</code>	Specify the VXLAN encapsulation mode as standard, transparent, or Q-in-Q.
<code>replicators [vtep-group name none]</code>	Specify the replicator group. Provide a VTEP group name to add a replicator. Specify <code>none</code> to not add a replicator or remove a configured replicator.
<code>public-vlan 2..4092</code>	Specify the Public VLAN for vNET VLAN.
<code>description description-string</code>	Provide a VLAN description.
<code>stats no-stats</code>	Use the options to enable or disable statistics collection for the VLANs being created.
<code>ports port-list</code>	Specify the ports assigned to the VLAN as list separated by commas.
<code>untagged-ports port-list</code>	Specify the untagged ports assigned to the VLAN as a list separated by commas.

Note: Netvisor ONE allows you to create a large number of VLANs by using the `vlan-create` command and the `range` keyword. However, in large network topologies with several nodes with heavy CPU traffic, the CLI may timeout if you create large number of VLANs. In such scenarios, try creating smaller number of VLANs.

By default, all ports are tagged on a newly created VLAN. However, if you want to specify select ports that should be trunked, then use the optional parameter `ports` with a comma separated list of ports, or specify a range of ports.

In some cases, you may not want a VLAN to be created on all ports. You can specify the port parameter as `none` to apply the VLAN only to the internal ports. For example:

```
CLI (network-admin@Leaf1) > vlan-create id 35 scope fabric ports none
```

To delete an existing VLAN, use the command:

```
CLI (network-admin@Leaf1) > vlan-delete
```

<code>vlan-delete</code>	Deletes a VLAN either by ID or by a range of IDs.
<code>id 2...4092</code>	Specify the VLAN ID that you want to delete.
<code>range vlan-list</code>	Specify the range of VLAN IDs that you want to delete. Use this parameter instead of <code>id</code> , if you want to specify a VLAN range.
Specify the following option:	
<code>vnet vnet-name</code>	Specify the name of the vNET from which the VLANs are to be deleted.

Configuration of an existing VLAN can be modified using the `vlan-modify` command.

```
CLI (network-admin@Leaf1) > vlan-modify
```

<code>vlan-modify</code>	Modify a VLAN by specifying the VLAN ID.
<code>id 2...4092</code>	Specify the VLAN ID that you intend to modify.
between 1 and 4 of the following options:	
<code>description description-string</code>	Provide a VLAN description.
<code>vxlan 1..16777215</code>	Specify the VXLAN identifier for the tunnel.
<code>replicators [vtep-group name   none]</code>	Specify the replicator group. Provide a VTEP group name to add a replicator. Specify <code>none</code> to not add a replicator or remove a configured replicator.
<code>vnet vnet name</code>	Specify the vNET name for this VLAN.
<code>public-vlan 2..4092</code>	Specify the public VLAN ID for vNET VLAN. Note: Public VLAN ID can only be specified for private VLANs.

For example, to modify VLAN25 description from blue to red:

```
CLI (network-admin@Leaf1) > vlan-modify id 25 description red
```

This description can be removed from VLAN25 using the command:

```
CLI (network-admin@Leaf1) > vlan-modify id 25 description ""
```

Netvisor ONE allows the addition of ports to a VLAN through the `vlan-port-add` command.

CLI (network-admin@Leaf1) > `vlan-port-add`

<code>vlan-port-add</code>	Add ports to VLANs.
Specify one of the following VLAN parameters:	
<code>vlan-id 2..4092</code>	Specify the VLAN ID to which ports are to be added.
<code>vlan-range vlan-list</code>	Specify the range of VLAN IDs to which ports are to be added.
<code>vlan-vnet vnet-name</code>	Specify the vNET for the VLANs to which the ports are to be added.
Provide one of the following port arguments:	
<code>switch switch-name</code>	Specify the name of the switch on which the ports are located.
<code>ports port-list</code>	Specify the ports that need to be added to the VLANs as a list separated by commas.
<code>[untagged   tagged]</code>	Specify either of the options to configure the ports as untagged or tagged ports.

For example, to configure ports 17 and 18 to accept untagged packets and map them to VLAN 595, use the following command:

CLI (network-admin@Leaf1) > `vlan-port-add vlan-id 595 ports 17,18 untagged`

To map ports on different switches into the `scope fabric` VLAN, use the following command:

CLI (network-admin@Leaf1) > `vlan-port-add vlan-id 1-4095 switch switch-name ports port-list`

Ports can be removed from a VLAN through the `vlan-port-remove` command.

CLI (network-admin@Leaf1) > `vlan-port-remove`

<code>vlan-port-remove</code>	Remove ports from VLANs.
Specify one of the following VLAN sectors:	
<code>vlan-id 2..4092</code>	Specify the VLAN ID from which ports are to be removed.
<code>vlan-range vlan-list</code>	Specify the range of VLAN IDs from which ports are to be removed
<code>vlan-vnet vnet name</code>	Specify the vNET for the VLANs from which ports

	are to be removed.
Provide one of the following port arguments:	
<code>switch <i>switch name</i></code>	Specify the name of the switch on which the ports are located.
<code>port <i>port list</i></code>	Specify the ports that need to be removed from the VLANs as a list separated by commas.

The `vlan-show` command displays the VLAN information.

CLI (network-admin@Leaf1) > `vlan-show`

<code>vlan-show</code>	Display VLAN information.
Specify one of the following VLAN sectors:	
<code>id 2..4092</code>	Specify the VLAN ID for which the information has to be displayed.
<code>range <i>vlan-list</i></code>	Specify the range of VLAN IDs for which the information has to be displayed.
<code>vnet <i>vnet name</i></code>	Specify the vNET for which VLAN information has to be displayed.
<code>type [public   private]</code>	Specify either of the type options to display information for public VLANs or private VLANs.
<code>vxlan 1..16777215</code>	Specify the VXLAN identifier for the tunnel.
<code>vxlan-mode [standard   transparent   qinq-access]</code>	Specify any of the VXLAN modes to display information for standard, transparent, or q-in-q modes.
<code>hw-vpn <i>hw-vpn-number</i></code>	Specify the hardware VPN number to display the related information.
<code>hw-mcast-group <i>hw-mcast-group-number</i></code>	Specify the hardware multi-cast group number to display the related information.
<code>replicators [vtep-group name   none]</code>	Provide a VTEP group name to view the VLAN information for that replicator group. Specify <code>none</code> to view the information on VLANs that do not involve replicator groups.
<code>repl-vtep <i>ip-address</i></code>	Specify the IP address of the replicator VTEP to view the related information.
<code>public-vlan 2..4092</code>	Specify the public VLAN ID for vNET VLAN to view the related information.
<code>scope [local   cluster   fabric]</code>	Provide any of the scope options to view the

	information on VLANs with that specified scope.
<code>description <i>description-string</i></code>	Specify a description to view the information on VLANs with that specific description.
<code>active [yes   no]</code>	Specify yes to view information on active VLANs. Specify no to view information on inactive VLANs.

For example: CLI (network-admin@Leaf1) > `vlan-show layout vertical`

```
switch:          leaf1
id:              1
type:            public
auto-vxlan:      no
replicators:     none
scope:           local
description:     default-1
active:          yes
stats:           yes
ports:           2-72
untagged-ports:  2-69
active-edge-ports: 69-70
```

CLI (network-admin@Leaf1) > `vlan-show format all layout vertical`

```
switch:          leaf1
id:              1
type:            public
auto-vxlan:      no
hw-vpn:          0
hw-mcast-group:  0
replicators:     none
repl-vtep:       ::
scope:           local
description:     default-1
active:          yes
stats:           yes
vrg:             0:0
ports:           2-72
untagged-ports:  2-69
active-edge-ports: 69
```

Network traffic statistics per VLAN can be displayed using the `vlan-stats-show` command. This command may be useful when troubleshooting network issues.

CLI (network-admin@Leaf1) > `vlan-stats-show format all layout vertical`

```
switch:          Leaf2
time:            10:51:02
vlan:            1
vnet:
```

```
ibytes:      36.2T
ipkts:       89.0G
idrops-bytes: 119M
idrops-pkts: 313K
obytes:      0
opkts:       0
odrops-bytes: 0
odrops-pkts: 0
switch:      Leaf2
time:        10:51:02
vlan:        35
vnet:
ibytes:      10.8K
ipkts:       154
idrops-bytes: 0
idrops-pkts: 0
obytes:      0
opkts:       0
odrops-bytes: 0
odrops-pkts: 0
switch:      Leaf1
time:        10:51:02
vlan:        1
vnet:
ibytes:      34.9T
ipkts:       84.6G
idrops-bytes: 3.03M
idrops-pkts: 5.69K
obytes:      0
opkts:       0
odrops-bytes: 0
odrops-pkts: 0
```

The output displays the following information:

- `switch` — switch name
- `time` — when the output was generated
- `VLAN ID` — ID assigned to the VLAN
- `vnet` — the vNET assigned to the VLAN
- `incoming and outgoing bytes` — in K (Kilobytes), M (Megabytes), or G (Gigabytes)
- `incoming and outgoing packets` — number of packets incoming and outgoing
- `incoming and outgoing dropped bytes` — in K (Kilobytes), M (Megabytes), or G (Gigabytes)
- `incoming and outgoing dropped packets` — number of dropped packets incoming and outgoing

## Configuring Rapid Spanning Tree Protocol (RSTP)

---

Rapid Spanning Tree Protocol (RSTP), a standard inter-switch protocol, ensures a loop-free forwarding network topology at Layer 2. This protocol was defined by the IEEE 802.1w standard and is an extension of the 802.1D Spanning Tree Protocol (STP). RSTP is an improvement over STP as it provides faster convergence after a network topology change or failure. RSTP introduces new port roles, and the original five port states of STP are reduced to three.

To build a loop-free topology, switches (bridges) determine the root bridge and compute the port roles. To do this, the bridges use special data frames called Bridge Protocol Data Units (BPDUs) that exchange bridge IDs and root path cost information. BPDUs are exchanged regularly, typically at two second intervals, and enable switches to keep track of network topology changes and to start and stop forwarding on ports as required. Hosts should not send BPDUs to the switch ports and to avoid malfunctioning/malicious hosts from doing so, the switch can filter or block BPDUs. If you enable BPDU filtering on a port, BPDUs received on that port are dropped but other network traffic is forwarded as usual. If you enable BPDU blocking on a port, BPDUs received on that port are dropped and the port is shut down.

### Port Roles in RSTP

**Root Port** (one per bridge): The forwarding port on each bridge which is on the best path to reach the root bridge.

**Designated Port**: The forwarding port for each LAN segment that leads away from the root bridge.

**Alternate Port**: An alternative path to the root bridge on a particular LAN segment, which is part of a bridge other than the one that has a designated port for the LAN segment. Alternate port is the second best root port.

**Backup port**: A backup/redundant port for the segment that already has one designated port. This port leads away from the root port.

**Disabled**: A port which is manually disabled and is not a part of STP.

### Port States in RSTP

**Discarding**: No data is exchanged over the port.

**Learning**: Frames are not forwarded, but the MAC address table is populated.

**Forwarding**: Fully functional.

Switches in RSTP expect a BPDU every 2 seconds (hello time) and if they do not receive a BPDU for 6 seconds (3 hello time intervals), it is considered to be a link failure. This is significantly faster than the STP link failure detection time of 20 seconds, dictated by the max age timer. RSTP can actively confirm if a port can safely be transitioned to the forwarding state without having to rely on the timer mechanism. Ports can be configured as edge ports if they are attached to a LAN that has no other bridges connected to it. Such a port can transition directly to the forwarding state, but it loses the edge port status as soon as it receives a BPDU. RSTP achieves rapid transition to the forwarding state on edge ports and point-to-point links (operating in full-duplex mode) but not on shared links (i.e., ports connected to a shared medium, hence operating in half-duplex mode).

If network connections form loops and STP is disabled, packets are forwarded indefinitely across the switches, causing degradation of network performance. STP supports limited Layer 2 multipathing and can result in sub-optimal utilization of available network links. Therefore, a fabric of switches does not rely only on RSTP within the boundaries of the network. Pluribus Networks recommends the use of RSTP for ad hoc networks that inter-operate in a heterogeneous, multi-vendor switch environment.



**Note:** RSTP is enabled on the switch by default.

Before you begin configuring RSTP, view the status of the protocol on the switch by using the command `stp-show`

```
CLI (network-admin@Leaf1) > stp-show
switch:                Leaf1
enable:                yes
stp-mode:              rstp
bpdu-broadcast-ports: yes
bridge-id:             3a:7f:b1:43:8a:0f
bridge-priority:       32768
hello-time:            2
forwarding-delay:      15
max-age:               20
cluster-mode:          master
```

The `cluster-mode` of a switch in an STP cluster could be `master` or `slave`. The master in an STP cluster is elected on the basis of which node has been up longer. The other node is the slave.

To display the STP state, use the following command:

```
CLI (network-admin@Leaf1) > stp-state-show
```

<code>stp-state-show</code>	Displays the STP state information.
Specify one or more of the following options to view the information specific to those options. Specifying no parameter will display all the information.	
<code>vlan <i>vlan-list</i></code>	Specify the VLANs as a list separated by commas.
<code>port <i>port-list</i></code>	Specify the ports as a list separated by commas.
<code>instance-id <i>instance-id-number</i></code>	Specify the STP instance ID.
<code>name <i>name-string</i></code>	Specify the name of the STP instance.
<code>bridge-id <i>mac-address</i></code>	Specify the bridge ID for which the information has to be displayed.
<code>bridge-priority <i>bridge-priority-number</i></code>	Specify the bridge priority number.
<code>root-id <i>mac-address</i></code>	Specify the root ID.
<code>root-priority <i>root-priority-number</i></code>	Specify the STP root priority.
<code>root-port <i>root-port-number</i></code>	Specify the STP root port.

<code>root-port(peer) root-port(peer)-number</code>	Specify the root port of the peer.
<code>hello-time hello-time-number</code>	Specify the STP hello time between 1s and 10s. The hello time is the time between each bridge protocol data unit (BPDU) that is sent on a port. The default hello time is 2s.
<code>forwarding-delay forwarding-delay-number</code>	Specify the STP forwarding delay between 4s and 30s. This is the time interval that is spent in the listening and learning states. Default forwarding delay timer is 15s.
<code>max-age max-age-number</code>	Specify the maximum age between 6s and 40s. This is the maximum length of time interval that an STP switch port saves its configuration BPDU information. The default max-age timer is 20s.
<code>internal no-internal</code>	Specify if the STP state is internal or not.
<code>peer no-peer</code>	Specify if the STP state is peer state or not.

For example: CLI (network-admin@Leaf1) > stp-state-show layout vertical

```

switch:          Leaf1
vlan:            1
ports:           none
instance-id:     1
name:            stg-default
bridge-id:       66:0e:94:65:e1:ef
bridge-priority: 8193
root-id:         64:0e:94:c0:06:4b
root-priority:   4097
root-port:       128
hello-time:      2
forwarding-delay: 15
max-age:         20
disabled:        none
learning:        none
forwarding:      25-28,128-129
discarding:      none
edge:            25-28
designated:       25-28,129
alternate:       none
backup:          none
  
```

The STP information pertaining to the ports can be displayed by using the command `stp-port-show`.

CLI (network-admin@Leaf1) > stp-port-show

<code>stp-port-show</code>	Displays the STP port information.
----------------------------	------------------------------------

Specify one or more of the following options to view the information specific to those options. Specifying no parameter will display all the information.

<code>port port-list</code>	Specify the ports as a list separated by commas.
<code>block no-block</code>	Specify if BPDU blocking is enabled on the ports or not.
<code>filter no-filter</code>	Specify if BPDU filtering is enabled on the port or not.
<code>edge no-edge</code>	Specify if the ports are edge ports or non-edge ports.
<code>bpdu-guard no-bpdu-guard</code>	Specify if BPDU guard is configured on the ports or not.
<code>root-guard no-root-guard</code>	Specify if root guard is configured on the ports or not.
<code>priority 0..240</code>	Specify the priority as a value between 0 and 240.
<code>cost 1..200000000</code>	Specify the port cost as a value between 1 and 200000000.

The STP state at the port level can be viewed using the command:

CLI (network-admin@Leaf1) > `stp-port-state-show`

<code>stp-port-state-show</code>	Display STP information at the port level.
Specify one or more of the following options to view the information specific to those options. Specifying none will display the information for all the parameters below.	
<code>vlan vlan-list</code>	Specify the VLANs as a list separated by commas.
<code>port port-list</code>	Specify the ports as a list separated by commas.
<code>stp-state Disabled/Discarding/Learning/Forwarding</code>	Specify one among the options as the STP state.
<code>role Disabled/Root/Designated/Alternate/Backup</code>	Specify one among the options as the port role.
<code>selected-role Disabled/Root/Designated/Alternate/Backup</code>	Specify one among the options as the selected role.
<code>state new-info/proposing/proposed/agreed/agreed/learn/learning/forward/forwarding/reselect</code>	Specify one among the options as the port state machine state.

<code>selected/reroot/rcvd-bpdu/rcvd-msg/rcvd-tc/rcvd-tc-ack/send-rstp/tc-prop/tc-ack/update-info/sync/synced/disputed/fdb-flush/online/looping/manual-online/edge/vlag-local-up/vlag-remote-up/requested-online/first-sync/p-is-d/p-is-m/root-guard-active</code>	
<code>designated-priority designated-priority-string</code>	Specify the designated priority vector.
<code>port-priority port-priority-string</code>	Specify the port priority vector.
<code>message-priority message-priority-string</code>	Specify the message priority vector.
<code>info-is disabled/received/mine/aged</code>	Specify the origin of port information.
<code>designated-times designated-times-string</code>	Specify the designated times: age, max age, hello, and forward delay
<code>port-times port-times-string</code>	Specify the port times: age, max age, hello, and forward delay
<code>message-times message-times-string</code>	Specify the message times: age, max age, hello, and forward delay
<code>hello-timer hello-timer-number</code>	Specify the STP hello time between 1s and 10s. The hello time is the time between each Bridge Protocol Data Unit (BPDU) that is sent on a port. The default hello time is 2s.
<code>topology-timer topology-timer-number</code>	Specify the topology change timer value.
<code>forward-timer forward-timer-number</code>	Specify the STP forwarding delay between 4s and 30s. This is the time interval that is spent in the listening and learning states. The default forwarding delay time is 15s.
<code>rcvd-info-timer rcvd-info-timer-number</code>	Specify the received info timer value.
<code>recent-root-timer recent-root-timer-number</code>	Specify the recent root timer value.
<code>recent-backup-timer recent-backup-timer-number</code>	Specify the recent backup timer value.
<code>edge-delay-timer edge-delay-timer-number</code>	Specify the edge delay timer value.
<code>migration-timer migration-timer-</code>	Specify the migration delay timer value.

<i>number</i>	
root-guard-timer <i>root-guard-timer-number</i>	Specify the root guard BPDU delay timer value.
sm-table-bits <i>sm-table-bits-number</i>	Specify the state machine table state.
sm-table <i>sm-table-string</i>	Specify the state machine table description.
vlag-peer-port <i>vlag-peer-port-number</i>	Specify the VLAG peer port if active-active.
peer   no-peer	Specify the STP peer state.

RSTP can be configured using the command `stp-modify`.

CLI (network-admin@switch1) > `stp-modify`

<code>stp-modify</code>	Modify the Spanning Tree Protocol parameters.
Specify one or more of the following options:	
<code>enable   disable</code>	Specify to enable or disable STP
<b><code>stp-mode rstp   mstp</code></b>	<b>Specify the STP mode as RSTP or MSTP.</b>
<code>bpdus-bridge-ports   bpdus-all-ports</code>	Specify to send BPDUs only on switch ports or on all ports.
<code>bridge-id mac-address</code>	Specify the STP bridge ID. The first part of the bridge ID is a 2-byte bridge priority field (which can be configured) while the second part is the 6-byte MAC address of the switch.
<code>bridge-priority 0..61440</code>	Specify the STP bridge priority in multiples of 4096. The default value is 32768.
<code>hello-time 1..10</code>	Specify the STP hello time between 1s and 10s. The hello time is the time between each BPDU that is sent on a port. The default value is 2s.
<code>forwarding-delay 4..30</code>	Specify the STP forwarding delay between 4s and 30s. The forwarding delay is the time that is spent in the listening and learning states. The default forwarding delay is 15s.
<code>max-age 6..40</code>	Specify the max age time between 6s and 40s. The max age timer defines the maximum time for which a switchport stores config BPDU information. The default value is 20s. If a config BPDU does not arrive at a port for 20s (default), the switch detects a link failure and takes action to restore connectivity through the backup links.
<code>mst-max-hops 1..32</code>	Specify the maximum hop count for MSTP BPDU. The default value is 20.
<code>mst-config-name mst-config-name-string</code>	Specify the name for MST configuration instance.

<code>root-guard-wait-time 0..300</code>	Specify the root guard wait time between 0s and 300s. The default value is 20. Specify the value as 0 to disable wait.
--	--

**Note:** Hello time, forwarding delay, and max age timers are not used by RSTP but are relevant to STP.

Netvisor ONE optimizes RSTP by not sending BPDUs on any ports except on inter-switch link-ports by default. However, if you do not configure Link Layer Discovery Protocol (LLDP), Netvisor does not detect host ports (i.e., ports directly connected to end devices) or send BPDU packets. As a result, both ports are in Forwarding state.

When you add the parameter `bpdus-all-ports` to the `stp-modify` command, it allows sending BPDUs on all ports even if hosts are not detected, unless the port is configured as an edge port. On a switch with a port connected to itself with this configuration, one of the ports goes into discarding state.

For example, to send BPDUs only on switch ports, use the command:

```
CLI (network-admin@switch1) > stp-modify bpdus-bridge-ports
```

To send BPDUs on all ports, use the command:

```
CLI (network-admin@switch1) > stp-modify bpdus-all-ports
```

STP ports can be configured using the command:

```
CLI (network-admin@Leaf1) > stp-port-modify
```

<code>stp-port-modify</code>	Displays the STP port information.
<code>port port-list</code>	Specify the ports as a list separated by commas.
Specify one or more of the following options:	
<code>block no-block</code>	Specify if BPDU blocking is to be enabled on the ports or not.
<code>filter no-filter</code>	Specify if BPDU filtering is to be enabled on the port or not.
<code>edge no-edge</code>	Specify if the ports are to be configured as edge ports or non-edge ports.
<code>bpdu-guard no-bpdu-guard</code>	Specify if BPDU guard is to be configured on the ports or not.
<code>root-guard no-root-guard</code>	Specify if root guard is to be configured on the ports or not.
<code>priority 0..240</code>	Specify the priority as a value between 0 and 240.
<code>cost 1..200000000</code>	Specify the port cost as a value between 1 and 200000000.

For example: To filter BPDUs on port 17, use the following command:

```
CLI (network-admin@Leaf1) > stp-port-modify port 17 filter
```

To block BPDUs on port 17 and shut down the port if BPDUs are received on the port, use the following command:

```
CLI (network-admin@Leaf1) > stp-port-modify port 17 block
```

To stop blocking BPDUs on port 17, use the following command:

```
CLI (network-admin@Leaf1) > stp-port-modify port 17 no-block
```

Edge ports are the ports on a switch that connect to workstations or computers. An edge port does not take part in spanning tree calculations and therefore, port flapping on edge ports does cause topology changes. BPDUs are not sent on edge ports and they can quickly transition from disabled mode to forwarding mode.

To configure a port as an edge port, use the command:

```
CLI (network-admin@Leaf1) > stp-port-modify port 17 edge
```

**Note:** You can disable STP on a port or a group of ports. If the devices connected to the switch ports are hosts and not downstream switches, or you know that a loop is not possible, disable STP to enable the port much faster when the switch restarts.

To view STP events on a switch, the command `stp-port-event-show` command is used. This command displays the port states as specified by the timing parameters.

```
CLI (network-admin@Leaf1) > stp-port-event-show
```

<code>stp-port-event-show</code>	Displays information about STP port events.
<code>port port-list</code>	Specify the ports as a list separated by commas.
Specify one or more of the following options:	
<code>time date/time: yyyy-mm-ddTHH:mm:ss</code>	Specify the time to start statistics collections.
<code>start-time date/time: yyyy-mm-ddTHH:mm:ss</code>	Specify the start time of statistics collection.
<code>end-time date/time: yyyy-mm-ddTHH:mm:ss</code>	Specify the end time of statistics collection.
<code>duration duration: #d#h#m#s</code>	Specify the duration of statistics collection.
<code>interval duration: #d#h#m#s</code>	Specify the interval between statistics collection.
<code>older-than duration: #d#h#m#s</code>	Specify the time older than which the statistics has to be displayed.
<code>within-last duration: #d#h#m#s</code>	Display statistics within last specified duration.
<code>port port-number</code>	Specify the port number.

<code>vlan <i>vlan-list</i></code>	Specify the list of VLANs.
<code>instance <i>instance-number</i></code>	Specify the STP instance number.
<code>count <i>count-number</i></code>	Specify the number of STP port events.
<code>initial-state Disabled Discarding Learning Forwarding</code>	Specify the initial state as one among the options.
<code>other-state Disabled Discarding Learning Forwarding</code>	Specify the other state as one among the options.
<code>final-state Disabled Discarding Learning Forwarding</code>	Specify the final state as one among the options.

For example:

CLI (network-admin@Leaf1) > stp-port-event-show

switch	time	port	vlan	instance	count	initial-state	other-state	final-state
Leaf1	20:36:39	121	1	0	1	Forwarding	Disabled	Disabled
Leaf1	20:38:05	17	1	0	4	Disabled	Disabled	Forwarding
Leaf1	20:40:04	17	1	0	1	Forwarding	Disabled	Disabled



## Configuring Multiple Spanning Tree Protocol (MSTP)

Multiple Spanning Tree Protocol as defined in IEEE802.1s or IEEE802.1Q-2005 provides the ability to manage multiple VLANs using Multiple Spanning Tree Instances (MSTIs). MSTP allows the formation of MST regions that can run multiple MSTIs. MSTP regions and other STP bridges are interconnected using the Common and Internal Spanning Tree (CIST).

MSTP regions are defined to be a collection of switches that have the same VLANs configured on all of them. All switches in a region must have the same configuration name, revision level, VLANs, and VLAN to MSTI associations. Each MST region may have multiple MSTIs operating within it but an MSTI cannot span multiple regions. Each MSTI must have a regional root which it may or may not share with another MSTI.

The CIST is the default spanning tree instance in MSTP. CIST forms a larger spanning tree for the entire bridged network by combining MSTP regions and single-instance spanning trees. The CIST instance has an MSTI ID of 0 which cannot be deleted or changed. When a new port-based or tagged VLAN is created, it is associated with the CIST by default and is automatically given an MSTI ID of 0. The default VLAN on a switch is also associated with the CIST. When a VLAN is assigned to an MSTI, it partially remains a member of the CIST. This is because CIST is used by MSTP to enable communication with MSTP regions and RSTP/STP single-instances in the network. CIST also has regional roots.

The switch with the lowest CIST priority value functions as the root bridge for all the MSTP regions and STP/ RSTP single-instance spanning trees in the network.

The following commands support the configuration of MST instances on a local switch:

```
CLI (network-admin@Leaf1) > mst-config-create
```

<code>instance-id 0..64</code>	Specify the ID as a number between 1 and 64 for MST configuration. MST ID 0 corresponds to the CIST instance.
<code>vlan vlan-list</code>	Specify the list of VLANs associated with the MST configuration
<code>bridge-priority 0..61440</code>	Specify the bridge priority as a number between 0 and 61440 which is a multiple of 4096. For example, the values can be 0, 4096, 8192, up to 61440. The default value is 32768.

Once created, the MST instance can be modified by the command:

```
CLI (network-admin@Leaf1) > mst-config-modify
```

<code>instance-id 0..64</code>	Specify the ID as a number between 1 and 64 for MST configuration. MST ID 0 corresponds to the CIST instance.
Specify one or more of the following options:	
<code>vlan vlan-list</code>	Specify the list of VLANs associated with the MST

---

```
bridge-priority 0..61440
```

configuration.

Specify the bridge priority as a number between 0 and 61440 which is a multiple of 4096. For example, the values can be 0, 4096, 8192, up to 61440. The default value is 32768.

---

The MST configuration can be deleted using the command:

```
CLI (network-admin@Leaf1) > mst-config-delete
```

---

```
instance-id 0..64
```

Specify the MST instance ID that needs to be deleted.

---

The current MST configuration on the switch can be viewed using the command:

```
CLI (network-admin@Leaf1) > mst-config-show
```

---

```
instance-id 0..64
```

Specify the ID as a number between 1 and 64 for MST configuration. MST ID 0 corresponds to the CIST instance.

Specify one or more of the following options to view the information related to those options.

```
vlan vlan-list
```

Specify the list of VLANs associated with the MST configuration.

```
bridge-priority 0..61440
```

Specify the bridge priority as a number between 0 and 61440.

---

## Achieving a Loop-Free Layer 2 Topology

---

Note: This feature can be configured only in a full mesh topology.

Rapid Spanning Tree Protocol (RSTP) and Multiple Spanning Tree Protocol (MSTP) ensure a loop-free topology in the Layer 2 as far as the networking equipment is concerned. Though RSTP prevents loops in the network caused by mis-cabled networking equipment, the protocol does not address mis-configured hosts. Netvisor ONE Loop Detection operates in conjunction with RSTP and MSTP to detect, log, and mitigate misbehaving and misconfigured hosts to prevent looping layer 2 traffic.

**Netvisor ONE Control Plane** — The Netvisor ONE control plane includes information about every MAC address in the Layer 2 network in a vPort database. This database is distributed throughout the fabric so that each Netvisor ONE switch has a copy of it for the entire fabric.

A MAC address is stored in a vPort, which includes the following information:

- MAC address, VLAN ID, and VXLAN ID
- Owner-port and local-port
- Migration history including owner, time, and port
- vPort state as active, static, moving, or loop-probe

Based on control plane data structures including the vPort database, Netvisor ONE decides if endpoints are to be allowed to access the network.

### Detecting Loops

Netvisor ONE Loop Detection is implemented as part of Netvisor ONE source MAC address miss handling. Netvisor ONE disables hardware learning of MAC addresses, so when a packet arrives with an unknown source MAC address, the switch sends the packet to Netvisor One rather than switching the packet normally. Netvisor ONE examines the vPort table to determine if a packet with an unknown source MAC indicates a loop.

Netvisor ONE uses two criteria to detect a loop in the network:

- A MAC address associated with an in-band NIC of a node in the fabric appears as the source MAC on a packet that ingresses on a host port. Netvisor ONE detects this situation by noting the `PN-internal` status of a vPort that would otherwise migrate to a host port. Netvisor does not allow the migration to take place and starts loop mitigation.

For the purposes of Netvisor ONE Loop Detection, a host port is defined as a port not connected to another Pluribus switch, not an internal port, and does not participate in STP with Netvisor ONE which means that Netvisor One is not configured for STP or the device connected on the port is not configured for STP.

- Packets with the same source MAC address arrive on multiple host ports in the fabric at approximately the same time. In order to support VM and host migration, some rapid movement of MAC addresses

through the fabric is tolerated. When the same MAC address moves rapidly back and forth between two ports, a loop is assumed and loop mitigation starts.

VRRP MAC addresses are not subject to loop detection and mitigation, and can migrate freely.

Loops are detected on a port by port basis. A single loop typically involves two ports, either on the same switch or on two different switches. When multiple loops occur with more than two ports then Netvisor ONE responds to each port separately.

## Loop Mitigation

When Netvisor ONE detects a loop, a message appears in the system log indicating the host port and VLAN involved in the loop. In addition the host port involved in the loop has the "*loop*" status added and Netvisor ONE adds the VLAN to the host port loop-vlans VLAN map. Looping ports and VLANs are displayed in the `port-show` output.

At the start of loop mitigation, Netvisor ONE creates vPorts to send loop probe packets. The vPorts use the port MAC address for the in-band NIC port, status of PN-internal, and a state of loop-probe. Netvisor ONE propagates Loop-probe vPorts throughout the fabric. Netvisor ONE creates a loop-probe vPort for each looping VLAN.

Netvisor ONE deletes all vPorts from the looping host port and VLAN at the start of loop mitigation. This prevents the hardware from sending unicast packets to the looping port, and causes every packet arriving on the looping port to appear in the software as a source MAC miss. During loop mitigation, Netvisor ONE drops all packets arriving on the looping port.

During loop mitigation, Netvisor ONE sends loop probe packets on the looping VLANs every 3 seconds. As long as the loop persists, Netvisor ONE receives the probe packets as source MAC miss notification on the looping ports, so Netvisor ONE can determine if the loop is still present. If 9 seconds elapse with no received probe packets, Netvisor ONE detects the loop is resolved and ends loop mitigation.

At the end of loop mitigation, log messages are added to the system log, loop-probe vPorts are removed, and loop stats and loop VLANs are removed from the looping port.

To view affected ports, use the `port-show` command and add the parameter, `status loop`:

```
CLI (network-admin@switch-31) > port-show status loop
```

switch	port	hostname	status	config
switch-31	9		up,stp-edge-port,loop	fd,10g
switch-32	9		up,stp-edge-port,loop	fd,10g

**Note:** the new status, `loop`, in the `status` column. When the loops are removed from the port, the `loop` flag is removed from the `port-show status` command output and log message is added regarding the removal of `loop`.

During loop mitigation, the MAC addresses for loop probes are displayed in the vPort table:

```
CLI (network-admin@switch-31) > vport-show state loop-probe
```

owner	mac	vlan	ports	state	hostname	status
switch-32	06:c0:00:16:f0:45	42	69	loop-probe	leo-ext-32	PN-internal
switch-31	06:c0:00:19:c0:45	42	69	loop-probe	leo-ext-31	PN-internal

Note the `loop-probe` state as well as the `PN-internal` state. The loop probes use the port MAC address format, and use the internal port for the in-band NIC.

**Note:** The state and the status columns are different in the above `vport-show stats loop-probe` command output. The status column refers to the vPort peer owner state in the fabric (the *PN-internal* parameter indicates that the MAC belongs to the PN fabric). The state column displays the vPort state.

If you notice a disruption in the network, use the `port-show` command to find the looping ports, and fix the loop. Fixing the loop typically involves correcting cabling issues, configuring virtual switches, or as a stop-gap measure, using the `port-config-modify` command to change port properties for the looping host ports. Once the loop is resolved, Netvisor ONE no longer detects probes and leaves the loop mitigation state, while logging a message:

```
2016-01-12,12:18:41.911799-07:00 leo-ext-31 nvOSd(25695) system
host_port_loop_resolved(11381) : level=note : port=9 :
Traffic has stopped looping on host-port=9
```

At this point the loop status is removed from the `port-show` output for port 9 and the loop-probe vPorts are removed.

Netvisor ONE Loop Detection exposes loops using system log messages, `port-show` output, and `vport-show` output.

When Netvisor ONE detects an internal port MAC address on a host port, Netvisor ONE prints a log message as below:

```
system 2016-01-19,15:36:40.570184-07:00 mac_move_denied
11379 note MOVE DENIED mac=64:0e:94:c0:03:b3 vlan=1 vxlan=0
from switch=leo-ext-31 port=69 to deny-switch=leo-ext-31 deny-port=9
reason=internal MAC of local switch not allowed to change ports
```

Netvisor ONE starts Loop Mitigation by logging a message:

```
system 2016-01-19,15:36:40.570334-07:00 host_port_loop_detected
11380 warn Looping traffic detected on host-port=9
vlan=1. Traffic on this port/VLAN will be ignored until loop
```

resolved

During Loop Mitigation, Netvisor ONE sends loop probes. When these probes, as well as any other packets, are received on a looping host port, Netvisor ONE logs a message:

```
system 2016-01-19,15:59:54.734277-07:00 mac_move_denied
11379 note MOVE DENIED mac=06:c0:00:19:c0:45 vlan=1 vxlan=0
from switch=leo-ext-31 port=69 to deny-switch=leo-ext-31
deny-port=9 reason=port is looping
```

Netvisor ONE limits `mac_move_denied` messages are limited to one every 5 seconds for each vPort. This prevents the system log from filling up with `mac_move_denied` messages during loop mitigation.

During loop mitigation, you can use the `port-show` command to see which ports are involved in the loop:

```
CLI (network-admin@Leaf1) > port-show status loop
```

switch	port	hostname	status	loop-vlans	config
leaf1	9		up,stp-edge-port,loop	1	fd,10g
leaf1	9		up,stp-edge-port,loop	1	fd,10g

Note the `loop` status in the status column and the `loop-vlans` column.

During loop mitigation the MAC addresses for loop probes are displayed in the vPort table:

```
CLI (network-admin@Leaf1) > vport-show state loop-probe
```

owner	mac	vlan	ports	state	hostname	status
leaf1	06:c0:00:16:f0:45	42	69	loop-probe	leo-ext-32	PN-internal
leaf1	06:c0:00:19:c0:45	42	69	loop-probe	leo-ext-31	PN-internal

## Fast Failover for STP and Cluster

---

Previously, cluster STP operation did not support fast failover because Netvisor ONE did not share STP state between the two nodes. As a result, when the master failed and when it came back online, the slave had to recompute the STP state from scratch. This resulted in topology changes twice, causing traffic loss until STP converged.

Currently, Netvisor ONE supports fast failover by default. In the cluster-STP mode, the node that has been up longer is elected as the master. Cluster syncs (keep-alives) are used to detect the peer node being online, and negotiate the initial cluster state. Cluster syncs determine which node has been up longer based on exchanged uptime values.

The master runs the state machine for both nodes and sends the STP and port states to the slave. The slave in turn maintains the state as informed by the master. The slave generates its own BPDUs based on the synchronized state and forwards the BPDUs that it receives to the master. When the cluster goes offline, the slave/master uses the same bridge ID and priority, uses consistent port IDs in BPDUs, and continues from the existing synchronized state. The STP state machine state is thus never lost.

Internal state synchronization using consistent bridge ID/priority and port IDs regardless of whether the cluster is online or offline, and active-active vLAG handling ensure that an end node detects no topology change when the cluster nodes go offline/online.

When a cluster is created, the STP configuration between the two cluster nodes is checked and is synchronized. The following guidelines are true regardless of whether the cluster is online or offline, and whether the peer node is online or offline:

- Both nodes use the same bridge ID or priority.
- When node1 sends a BPDU, the port ID inside the packet is 1-256, except for active-active vLAGs.
- When node2 sends a BPDU, the port ID inside the packet is 257-512, except for active-active vLAGs.
- When either node sends a BPDU on an active-active vLAG, the port ID inside the packet is node1's port number.
- Configuration changes (STP mode, MST instances, bridge ID, etc.) are mirrored on both nodes through cluster transactions.

Due to the above guidelines, a BPDU sent on an active-active vLAG appears exactly the same to a third party receiver regardless of whether that packet came from cluster node1 or node2.

Netvisor ONE provides two show commands to view the details of this functionality: `stp-state-show` and `stp-port-state-show`.

For example:

```
CLI (network-admin@Leaf1) > stp-state-show
switch:           Leaf-1
vlan:             1
ports:            none
instance-id:      1
name:             stg-default
bridge-id:        66:0e:94:d5:b0:cc
```

```

bridge-priority: 32769
root-id: 66:0e:94:35:c2:ce
root-priority: 32769
root-port: 128
hello-time: 2
forwarding-delay: 15
max-age: 20
disabled: none
learning: none
forwarding: none
discarding: none
edge: none
designated: none
alternate: none
backup: none

```

```

CLI (network-admin@Switch2) > stp-port-state-show port 17
switch: Switch2
vlan: 1
port: 17
stp-state: Forwarding
role: Designated
selected-role: Designated
state: agreed,learn,learning,forward,forwarding,selected,send-rstp,syncd,online,requested-
online
designated-priority: 32769-66:0e:94:38:39:80,100,32769-66:0e:94:b7:65:91,32785
port-priority: 32769-66:0e:94:38:39:80,100,32769-66:0e:94:b7:65:91,32785
message-priority: 0-00:00:00:00:00:00,0,0-00:00:00:00:00:00,0
info-is: mine
hello-timer: 2
root-guard-timer: 0
sm-table-bits: 0xfaedee
sm-table: prx=discard*,bdm=not-edge*,ptx=idle*,pim=current*,prt-
disabled=disable*,prt-root=root*,prt-desg=designated*,prt-alt-
bk=block*,pst=forwarding*,tcm=active*

```



## Configuring Auto-Recovery of a Disabled Port

Netvisor ONE automatically disables physical ports due to certain violations. For example, if a port receives BPDU messages on an edge port, Netvisor ONE disables the port because receiving BPDUs on a edge port becomes a security violation. This happens because the edge port is configured for BPDU guard for the violation to take effect.

The port may be disabled due to the following errors:

- **BPDU Guard Messages:** The port is set to `err-disabled` when a BPDU is received on an edge port with BPDU guard enabled.
- **Link Flaps:** There are too many link flaps for a configured interval of time.
- **MAC address Security Violation:** The number of MAC addresses on an interface is greater than the configured limit.

To clear the counters for `err-disable` caused by BPDU guard and MAC security, use the following command:

```
CLI (network-admin@Leaf1) > err-disable-clear-counters
```

<code>err-disable-clear-counters</code>	Resets <code>err-disable</code> counters for all the ports on the switch.
Specify any of the following options:	
<code>bpduguard no-bpduguard</code>	Specify if BPDU guard counters are to be cleared.
<code>macsecurity no-macsecurity</code>	Specify if MAC security counters are to be cleared.
<code>recovery-timer duration: #d#h#m#s</code>	Specify the recovery timer value. The default timer value is 5 minutes. Example: 20s or 1d or 10d20m3h15s

To configure BPDU guard or MAC security on a switch, use the command:

```
CLI (network-admin@Leaf1) > err-disable-modify
```

<code>err-disable-modify</code>	Modifies the <code>err-disable</code> settings for the switch.
Specify any of the following options:	
<code>bpduguard no-bpduguard</code>	Specify either of the options to enable or disable BPDU guard.
<code>macsecurity no-macsecurity</code>	Specify either of the options to enable or disable MAC security.
<code>recovery-timer duration: #d#h#m#s</code>	Specify the recovery time value. The default timer value is 5 minutes.

---

Example: 20s or 1d or 10d20m3h15s

---

To view the error recovery settings on a switch, use the `err-disable-show` command. For example:

```
CLI (network-admin@Leaf1) > err-disable-show
switch:                Leaf1
bpduguard:             off
macsecurity:           off
recovery-timer:        5m
```

## Configuring STP Root Guard

The Root Guard feature is used to enforce the positioning/placement of root bridge in a network. In STP, there is no provision to have full control on the selection of the root bridge or switch. Any switch can be selected as the root bridge. If the bridge priority is set to 0, that switch is likely to become the root bridge. However, even with this configuration, there is no guarantee since there can be another switch with priority 0 and a lower MAC address that gets selected as root bridge.

The Root Guard feature forces a port to be a designated port (and does not allow it to become root port). This prevents any one of the neighboring switches from becoming the root switch. Thus, the Root Guard feature provides a way to enforce the placement or positioning of the root bridge in the network.

If a port on which the Root Guard feature is enabled receives a superior BPDU, it moves the port into a root-inconsistent state (similar to a listening state). In this state, no traffic is forwarded across this port. Root Guard must be enabled on all ports where the root bridge should not appear.

To configure root guard, use the command:

```
CLI (network-admin@Leaf1) > stp-port-modify port port-list root-guard
```

<code>stp-port-modify</code>	Modify the Spanning Tree Protocol Parameters.
<code>port <i>port-list</i></code>	Specify the port or port list.
Specify one or more of the following options:	
<code>block no-block</code>	Specify if a STP port blocks BPDUs.
<code>bpdu-guard no-bpdu-guard</code>	Enable or disable STP port BPDU guard.
<code>root-guard no-root-guard</code>	Enable or disable STP port Root guard.

To view the root guard configuration details, use the command:

```
CLI (network-admin@Leaf1) > stp-port-show
```

## Configuring Fabric Guard

---

Currently, Netvisor ONE detects a Layer 2 loop using STP, LLDP, or loop detect code. However if a third party device connected to a Pluribus Networks switch consumes LLDP such as a hypervisor vSwitch, and you configure the port as an edge port, Netvisor ONE cannot detect loops in the network.

If you configure a port as fabric-guard port, Netvisor ONE triggers sending global discovery multicast packets on this port after the port is physically up and in an adjacency wait state. If a port with fabric-guard configuration receives a global discovery packet, Netvisor ONE disables the port in the same way LLDP disables the port when receiving messages from the same switch.

To enable fabric guard, use the following syntax:

```
CLI (network-admin@Leaf1) > port-config-modify port port-number fabric-guard
```

To disable fabric guard, use the following syntax:

```
CLI (network-admin@Leaf1) > port-config-modify port port-number no-fabric-guard
```

In order to re-enable the port once you fix the loop, you must manually enable the port using the command, `port-config-modify port port-number enable`.

## About Layer 2 Hardware Hashing

A hardware hashing operation comprises two parts that can be performed at extremely high speed: first, packet field selection and extraction, then a mathematical bitwise computation.

The first step therefore is for the hardware parsing logic to select a number of packet fields 'to be hashed' depending on the traffic type.

These are the traffic types and related packet fields that Netvisor ONE currently supports *by default* to perform hashing on:

**Table Caption 4-1: Hardware Hashing Field Selection**

Application	Traffic Type <sup>2</sup>	Hashed Fields <sup>3</sup>
Non-IP packets <sup>1</sup>	Ethernet (a.k.a. 'Layer 2')	Source MAC address, Destination MAC address, Ethertype, VLAN number, source physical port
Plain unicast and multicast IPv4 packets	IP	Source IPv4 address, Destination IPv4 address, VLAN number, Destination Layer 4 port, Source Layer 4 port, IP protocol number, source physical port
Plain unicast and multicast IPv6 packets	IP	Source IPv6 address, Destination IPv6 address, VLAN number, Destination Layer 4 port, Source Layer 4 port, IP protocol number, source physical port
VXLAN packets (UDP encapsulated)	IP	Source IP address, Destination IP address, VLAN number, Destination Layer 4 port, <i>Source Layer 4 port</i> , IP protocol number, source physical port

### Note:

- <sup>1</sup>Certain traffic types such as *Mac-in-Mac*, *FCoE* and *SCTP* are currently not supported for field selection; hence they are not load balanced as opposed to the supported traffic types (see also the next section).
- <sup>2</sup>Traffic types are based on IANA's Ethertype definitions (e.g., 0x0800 for IPv4 and 0x86DD for IPv6). Packet fields are based on standard protocol definitions (IEEE 802.3 for Ethernet, RFC 791 and RFC 8200 for IPv4 and IPv6 respectively).

- <sup>3</sup>(v)LAG hashing applies to bridged as well as routed traffic without distinction.

Once the proper field values are extracted based on the traffic type (as per the table above), the second step performed by the hardware is the *hash function calculation*. Netvisor ONE uses the standard *CRC16 CCITT* cyclic redundancy check algorithm for the hash function calculation.

However, before calculating the CRC, 128-bit IPv6 source and destination addresses get 'folded' into 32-bit shortened versions with the *FoldAndXOR* method:

$$\text{address\_bits}(127:96) \wedge \text{address\_bits}(95:64) \wedge \text{address\_bits}(63:32) \wedge \text{address\_bits}(31:0)$$

where  $\wedge$  means 32-bit XOR.

This preserves the variability over the entire 128-bit address length when fed in 32-bit chunks to the CRC16 CCITT calculation.

## Configuring Layer 3 Features

---

This chapter provides information for understanding and using the Pluribus Networks Netvisor One command line interface (CLI) on a Netvisor ONE switch to configure Layer 3 protocols (aka Network layer protocols).

- 
- [Understanding Supported Layer 3 Protocols](#)
  - [Configuring Packet Relay for DHCP Servers](#)
  - [Configuring vRouter Services](#)
  - [Displaying FRR Routing and Debug Information for vRouters](#)
  - [Configuring Hardware-based Routing](#)
  - [Configuring Static Routes](#)
  - [Adding IPv6 Link-Local Addresses for Static Routing](#)
  - [Configuring Static Null Routing](#)
  - [Configuring Static ARP for Unicast Traffic](#)
  - [Configuring IPv4 IPv6 Neighbor Discovery Process Support and Optimization](#)
  - [Configuring Routing Information Protocol \(RIP\)](#)
  - [Configuring Open Shortest Path First \(OSPF\)](#)
  - [Configuring BGP on a vRouter](#)
  - [Configuring Prefix Lists for BGP and OSPF](#)
  - [Configuring BFD with BGP](#)
  - [Configuring BFD for OSPF Fault Detection](#)
  - [Configuring Optimized BFD Path](#)
  - [Configuring Policy-based Routing](#)
  - [Configuring Multicast Listener Discovery \(MLD\)](#)
-

## Understanding the Supported Layer 3 Protocols in Netvisor ONE

---

Layer 3 is the Network layer protocol that behaves as a transmission medium from the source to the destination by performing the routing and addressing aspects of a packet and ensuring data integrity without having any packet loss.

Netvisor ONE supports the following Layer 3 or Network protocols and functionalities:

- OSPF
- OSPFv3
- BGP
- MP-BGP
- RIP
- VRRP for IPv4 and IPv6
- Dual-stack support (IPv4/IPv6)
- Equal-cost multi-path routing (ECMP)
- Policy Based Routing (PBR)
- Bidirectional Forwarding Detection (BFD) (IPv4 and IPv6), OSPF, BGP and static routes
- Static routes
- Loopback interface
- DHCP relay
- MLD



## Configuring Packet Relay for DHCP Servers

---

In general, routers do not forward broadcast packets from one subnet to another. However, there are cases in which the broadcast packets need to be passed onto other subnets. One typical example is DHCP (Dynamic Host Configuration Protocol) in which a new computer needs to acquire an available IP address by broadcasting the requests to DHCP server. DHCP protocols work seamlessly if both DHCP client and DHCP server are located in the same broadcast domain. However, that does not work across broadcast domains without supporting features such as DHCP relay. That is, when the DHCP server is not in the same subnet as the clients, use DHCP relay for broadcasting traffic.

Packet Relay is the router functionality that helps forward broadcast packets between broadcast domains. In Netvisor ONE architect, the packet relay functionality is implemented as a service running within a particular vRouter.

You can configure a vRouter to relay DHCP requests from local clients to a centralized DHCP server. Since the initial DHCP request arrives from a client that typically does not have an IP address, the client must find the DHCP server using a Layer 2 broadcast.

The DHCP server needs information before the server can allocate an IP address to the client. It must know the subnet and the MAC address of the client. The DHCP server needs the subnet information to ensure that the IP address that the client receives can work on the client's subnet. The MAC address is necessary so that the DHCP server can find any information that is unique to the client.

When you configure DHCP relay on a vRouter, the vRouter converts the local broadcast packet from the client to a unicast packet and forwards it to the server.

Since the DHCP client does not have an IP address when it sends the DHCP request packet, the client uses the IP address, 0.0.0.0, as the source IP address and the general broadcast address 255.255.255.255 for the destination.

The vRouter replaces the source address with the IP address assigned to the interface where the request is received, and replaces the destination IP address with the address you specify in the vRouter packet-relay command.

To configure packet-relay for a DHCP server with the IP address 172.16.21.34 and vRouter interface (on the network where the client is connected to) - eth11.100, use the following syntax:

```
CLI (network-admin@Leaf1) > vrouter-packet-relay add vrouter-name vrouter-  
dhcp forward-proto dhcp forward-ip 172.16.21.34 nic eth11.100
```

Once you add the configuration, you cannot modify it. If you make a mistake or want to add a new configuration, you must use the `vrouter-packet-relay-remove` command.

## Configuring vRouter Services

---

A Virtual Router (vRouter) is an important part of fabric functionality. For example, for a VLAN to communicate with other VLANs, or networks external to the fabric, it may need a vRouter that spans the internal and the external network.

Fabric-wide vRouter commands can only be executed at the fabric level by the fabric administrator.

**Note:** For switches with ONVL, the only available vNET is a global vNET created when a fabric is created for the first time. Use tab complete in the CLI to display the vNET and continue the configuration. However, some switches support multiple VNETs based on the platforms. Please refer to the Release Notes for the supported platforms.

Routing protocols essentially work the same way on virtual routers as physical routers.

## Configuring vRouter Interfaces

You can now add IPv4 and IPv6 addresses to a vRouter interface. If you enable or disable the interface, both IPv4 and IPv6 are affected. Routing protocols can be enabled or configured independently using the IP address.

When you add a vRouter interface, you can configure it with two IP addresses, IPv4 and IPv6 and can have multiple IP addresses (primary and secondary).

```
CLI (network-admin@Leaf1) > vrouter-interface-add vrouter-name name-string vlan vlan-id ip ip-address netmask netmask assignment none|dhcp ip2 ip-address netmask2 netmask
```

You can display the IP addresses:

```
CLI (network-admin@Leaf1) > vrouter-interface-show vrouter-name name-string vlan vlan-id ip ip-address netmask netmask assignment none|dhcp|dhcpv6 ip2 ip-address netmask2 netmask
```

To migrate the interface from a IPv4 address to a IPv6 address, the following commands are used:

```
CLI (network-admin@Leaf1) > vrouter-interface-ip-add
```

---

<i>vrouter-name name-string</i>	Specify the name of the vRouter.
<i>nic nic-string</i>	Specify the virtual NIC assigned to the interface.
<i>ip ip-address</i>	Specify the IP address assigned to the interface.
<i>netmask netmask</i>	Specify the netmask of the IP address.

Specify any of the following options:

---

<i>vnet vnet-name</i>	Specify the vNET assigned to the vRouter.
-----------------------	---

---

```
CLI (network-admin@Leaf1) > vrouter-interface-ip-remove
```

---

<i>vrouter-name name-string</i>	Specify the name of the vRouter.
<i>nic nic-string</i>	Specify the virtual NIC assigned to the interface.
<i>ip ip-address</i>	Specify the IP address assigned to the interface.

---

```
CLI (network-admin@Leaf1) > vrouter-interface-ip-show
```

---

<code>vrouter-name name-string</code>	Specify the name of the vRouter.
<code>nic nic-string</code>	Specify the virtual NIC assigned to the interface.
<code>ip ip-address</code>	Specify the IP address assigned to the interface.
<code>netmask netmask</code>	Specify the netmask of the IP address.
Specify any of the following options	
<code>vnet vnet-name</code>	Specify the vNET assigned to the vRouter.

---

## Configuring MTU Parameters for vRouter Interfaces

Support for IPv4 and IPv6 addresses requires configuring the MTU parameter differently for each type of IP address. By default MTU is set at 1500 for the VNICs. For IPv4 addresses, the Maximum Transmission Unit (MTU) the range is to 68 to 9216, but the range for IPv6 addresses is 1280-9216. However, Netvisor One displays the MTU range as 68-9216 which is supported by IPv4 addresses. But if you configure an IPv6 interface with an MTU value outside of the 1280-9216 for IPv6 addresses, Netvisor One returns an error.

In the case of interfaces configured with IPv4 and IPv6 addresses (dual-stack), Netvisor One limits the MTU range to 1280-9216. The following special rules apply to dual-stack interfaces:

- If you configure the MTU option for a vRouter interface with an IPv6 address using the command, `vrouter-interface-create`, Netvisor One performs a range check to ensure the MTU value is within 1280-9348.
- If you configure the MTU option for a vRouter interface using the `vrouter-interface-modify` command, Netvisor One performs a check for IPv6 addresses, and then performs a range check for 1280-9348.
- If you add an IPv6 address with the command, `vrouter-interface-ip-add`, Netvisor One performs a range check to ensure the MTU of the VNIC interface is within 1280-9348.
- If you configure a hardware vRouter, Netvisor One propagates the MTU value to the switch interface, so the MTU values are in the switch ASIC.

**Informational Note:** Starting from Netvisor ONE release 5.1.1 the maximum MTU size has been increased to 9348.

## Configuring 802.1p-based Prioritization and Marking on Egress Interfaces

In Layer 3 networks, traffic priority is typically indicated in the Type of Service (ToS) or Diff Serv Code Point (DSCP) fields in the IPv4 or IPv6 headers of the packets.

Switches can be programmed to map these Quality of Service (QoS) or Class of Service (CoS) values to one of 8 traffic classes and then to use such class assignment to place packets into one of 8 egress queues of a port. This important capability (also known as class-based traffic queuing) helps to deal with traffic congestion in the network's choke-points.

However, in certain network designs such as, multi-tenant DC designs, it may not be always possible to use the ToS or DSCP fields for class-based prioritization.

An alternative solution is offered by the IEEE 802.1p standard classification, which allows any VLAN-tagged Ethernet traffic to be explicitly classified based on a Class of Service (CoS) 3-bit field, (also known as Priority Code Point (PCP)).

Therefore, since packets forwarded on a Layer 3 interface do not usually carry an IEEE VLAN tag (and associated CoS field) by default, this enhancement introduces the capability of explicitly adding a CoS field in the traffic egressing a vRouter interface.

Netvisor ONE now supports a new parameter for the configuration of an IEEE 802.1p priority tag (i.e., CoS) on an interface:

```
CLI (network-admin@switch) > vrouter-interface-add priority-tag|no-  
priority-tag
```

```
CLI (network-admin@switch) > vrouter-interface-modify priority-tag|no-  
priority-tag
```

When priority-tag is selected, Netvisor ONE adds a CoS value based on the following ingress scenarios and configuration:

- Case A: On a vRouter interface when the ingress traffic contains an IEEE 802.1p priority tag (CoS), the default behavior is to trust such classification. Hence Netvisor ONE uses the same CoS value received in ingress for the egress traffic.
- Case B: When the ingress traffic is untagged, Netvisor ONE uses the default CoS value (0) for the egress traffic on the vRouter interface.
- Case C: When the ingress traffic contains a DSCP value and if the network admin adds a DSCP map to the port, then Netvisor ONE will set the egress CoS value according to the configured DSCP map. In other words, when this specific configuration is chosen by the network admin, the cases A and B is overridden and the egress traffic's CoS value will instead be decided by the ingress DSCP map (while the ingress CoS value is ignored).

## Configuring IPv6 for vRouter Loopback Addresses

Netvisor One now supports IPv4 and IPv6 addresses on the same loopback interface.

When you configure a loopback interface, you can optionally associate a vRouter interface to the configuration. If you do not associate a vRouter interface, Netvisor One selects the first non-VRRP vRouter interface. Netvisor One uses the loopback interface for a vRouter as a host route in the Forwarding Information Database (FIB) for packets received from any port and routes them to the correct vRouter.

If there is no vRouter interface configured, the loopback interface is unreachable on the network. When you add the first vRouter interface, the loopback interface is reachable.

If you remove the associated vRouter interface, Netvisor One selects the next available vRouter interface.

Netvisor One does not support multiple vRouter loopback interfaces for IPv6 addresses.

To add an IPv4 address to a loopback interface for vRouter, **vr1**, use the following syntax:

```
CLI (network-admin@Leaf1) > vrouter-loopback-interface-add vrouter-name vr1
ip 99.1.1.1
```

To add an IPv6 vRouter loopback interface to vRouter, **vr1**, use the following syntax:

```
CLI (network-admin@Leaf1) > vrouter-loopback-interface-add vrouter-name vr1
ip 2999:1000::1
```

To display the vRouter loopback interface, use the `vrouter-loopback-interface-show` command:

```
CLI (network-admin@Leaf1) > vrouter-loopback-interface-show
```

vrouter-name	index	ip	router-if
174 -----	-----	-----	-----
175 vr1	1	99.1.1.1	eth0.100
176 vr1	2	2999:1000::1	eth0.100

```
CLI (network-admin@Leaf1) > vrouter-routes-show
```

vrouter-name	network	type	interface	next-hop	distance	metric
-----	-----	-----	-----	-----	-----	-----
vr1	10.1.1.0/24	connected	eth0.100			
vr1	10.1.1.0/24	connected	eth0.100			
vr1	20.1.1.0/24	ospf	eth0.121	12.1.1.2	110	20
vr1	88.1.1.1/32	ospf	eth0.121	12.1.1.2	110	20
vr1	99.1.1.1/32	connected	lo			
vr1	2100:100::/64	connected	eth0.100			
vr1	2121:100::/64	connected	eth0.121			
vr1	2200:100::/64	ospf6	eth0.121	fe80::640e:94ff:fe4d:d0c8	110	10

vr1	2888:1000::1/128	ospf6	eth0.121	fe80::640e:94ff:fe4d:d0c8	110	10
vr1	2999:1000::1/128	connected	lo			
vr1	fe80::/64	connected	eth0.121			



## Displaying FRR Routing and Debug Information for vRouters

This feature provides a new vRouter command to display the output of an IP Stack (FRR) routing and debug information from the Netvisor One command line interface (CLI).

```
CLI (network-admin@Spine1) > help vrouter-vtysh-cmd
```

---

<code>vrouter-vtysh-cmd</code>	display output of a FRR command.
<code>name <i>name-string</i></code>	name of service config.
<code>cmd <i>cmd-string</i></code>	any FRR show/debug/undebug/no debug/clear ip/clear ipv6 command.

---

### Show Output Examples

```
CLI (network-admin@Spine1) > vrouter-vtysh-cmd name vr1 cmd "show ip route"
```

```
Codes: K - kernel route, C - connected, S - static, R - RIP,
       O - OSPF, I - IS-IS, B - BGP, P - PIM, A - Babel,
       > - selected route, * - FIB route
```

```
C>* 100.1.1.0/24 is directly connected, eth1.100
```

```
C>* 127.0.0.0/8 is directly connected, lo0
```

```
CLI (network-admin@Spine1) > vrouter-vtysh-cmd name vr1 cmd "show running-
config"
```

```
Building configuration...
```

```
Current configuration:
```

```
!
hostname vr1
log file zebra.log
!
password zebra
enable password zebra
!
interface eth1.100
ipv6 nd suppress-ra
multicast
no link-detect
!
interface lo0
no link-detect
!
ip forwarding
ipv6 forwarding
```

```
!  
line vty  
!  
end
```

For this feature, only `show` commands are allowed as legal values in the CLI.

```
CLI (network-admin@Leaf1) > vrouter-vtysh-cmd name vr1 cmd "config t"
```

```
vrouter-vtysh-cmd: Illegal value for "cmd"
```

```
vrouter-vtysh-cmd: Legal values: commands starting with show, in quotes
```

## Viewing FRR Logs

---

FRR logs files, such as Zebra, OSPF, OSPF6, BGP, BFD and RIP can be viewed directly from the console using the Netvisor One CLI.

Since FRR log files accumulate on the switch, you may want to clear a particular log file so they do not create space issues.

Use this command to view FRR logs files directly from your console:

```
CLI (network-admin@Leaf1) > vrouter-log-show vrouter-name vrouter-name  
protocol
```

---

<code>vrouter-log-show</code>	Displays vrouter protocol logs
-------------------------------	--------------------------------

One or more of the following options:

<code>vrouter-name <i>vrouter-name</i></code>	Specify name of the vRouter.
---	------------------------------

<code>protocol zebra   ospf   ospf6   bgp   bfd   rip</code>	Specify the name of the protocol files to view.
--	---

---

Use this command to clear FRR files from a specific protocol log file:

```
CLI (network-admin@Leaf1) > vrouter-log-clear
```

---

<code>vrouter-log-clear</code>	Clears vrouter protocol logs from a protocol log file
--------------------------------	---

One or more of the following options:

<code>vrouter-name <i>vrouter name</i></code>	Specify the name of the vRouter service.
---	--

<code>protocol zebra   ospf   ospf6   bgp   bfd   rip</code>	Specify the name of the log file to clear.
--	--

---

## Configuring Hardware-based Routing

---

- [Configuring Hardware Routing for a vRouter](#)
  - [Layer 3 Table Validation](#)
  - [Displaying Hardware Routes History](#)
  - [Sending Network Traffic to an ECMP Group \(moved from vLE section\)](#)
  - [Understanding ECMP Path Selection and Load Balancing](#)
  - [About Hardware Hashing](#)
-

## Configuring Hardware Routing for a vRouter

---

Create interfaces on hardware routers and map them to virtual NICs in the vRouter zone. The number of hardware vRouters you can create depends on the platform. You can create only one vRouter on low-end platforms and can range up to 32 vRouters on high-end platforms.

The supported protocols are as follows:

- **OSPF** — OSPF does not use a TCP/IP transport protocol such as UDP or TCP, but is encapsulated in the IP datagram with protocol number 89. OSPF uses multicast addressing for route flooding on a broadcast domain. For non-broadcast network, special provisions in the configuration facilitate neighbor discovery. OSPF reserves the multicast addresses 224.0.0.5/6 for IPv4 or FF02::5/6 for IPv6.
- **BGP** — BGP uses TCP and port number 179.
- **RIP** — uses the following parameters:
  - **RIPv1** — IPv4 uses UDP and port 520, and advertise address - broadcasting
  - **RIPv2** — IPv4 uses UDP and port 520, and advertise address - 224.0.0.9
  - **RIPng** — IPv6 uses UDP and port 521, and advertise address - FF02::9
- **PIM** — IPv4 uses protocol 103 with multicast address 224.0.0.13

To create a hardware-based vRouter called `hwtest`, use the following command:

```
CLI (network-admin@Leaf1) > vrouter-create hwtest router-type hardware
```

Use the same commands as software routing to add protocols and interfaces.

## IPv6 Hardware Routing

---

Netvisor ONE supports both IPv4 and IPv6 natively, and both can be configured on the same interface.

In addition to static routing, Netvisor ONE supports the OSPFv3 and Multi-protocol BGP protocols for IPv6 unicast routing. vRouter interfaces can be configured with globally scoped IPv6 addresses.

```
CLI (network-admin@Spine-1) vrouter-interface-add vrouter-name vrouter-1 ip
2200:1111:2222:3333::1/64 vlan 100
```

```
CLI (network-admin@Spine-1) vrouter-interface-add vrouter-name vr0 ip
2200::1/64 vlan 200
```

```
CLI (network-admin@Spine-1) vrouter-interface-add vrouter-name vr0 ip
2211::1/64 vlan 211
```

```
CLI (network-admin@Spine-1) vrouter-ospf6-add vrouter-name vr0 nic eth0.200
ospf6-area 0.0.0.0
```

```
CLI (network-admin@Spine-1) vrouter-ospf6-add vrouter-name vr0 nic eth0.211
ospf6-area 0.0.0.0
```

### BGP with IPv6 Unicast Peers Distributing IPv6 Prefixes

```
CLI (network-admin@Spine1) > vlan-create id 200 scope fabric ports 9
```

```
CLI (network-admin@Spine1) > vlan-create id 211 scope fabric
```

```
CLI (network-admin@Spine1) > vnet-create name vnet0 scope fabric vlans 200
```

```
CLI (network-admin@Spine1) > vrouter-create name vr0 vnet vnet0 router-
type hardware router-id 1.1.1.1 hw-vrrp-id 1
```

```
CLI (network-admin@Spine1) > vrouter-interface-add vrouter-name vr0 ip
2200::1/16 vlan 200
```

```
CLI (network-admin@Spine1) > vrouter-interface-add vrouter-name vr0 ip
2211::1/24 vlan 211
```

```
CLI (network-admin@Spine1) > vrouter-modify name vr0 bgp-as 200 no-bgp-
ipv4-unicast
```

```
CLI (network-admin@Spine1) > vrouter-modify name vr0 bgp-redistribute
static,connected
```

```
CLI (network-admin@Spine1) > vrouter-bgp-add vrouter-name vr0 neighbor
2211::2 remote-as 222 multi-protocol ipv6-unicast
```

```
CLI (network-admin@Spine1) > vrouter-bgp-add vrouter-name vr0 neighbor  
2211::3 remote-as 233 multi-protocol ipv6-unicast
```

```
CLI (network-admin@Spine1) > vrouter-bgp-network-add vrouter-name vr0  
network 2200::/64
```

```
CLI (network-admin@Spine1) > vrouter-bgp-network-add vrouter-name vr0  
network 2211::/64
```

## **IPv6 and Prefix Lists**

```
CLI (network-admin@Spine1) > vrouter-prefix-list-add vrouter-name vr0 name  
block200 action deny prefix 2200::/16 seq 95
```

```
CLI (network-admin@Spine1) > vrouter-bgp-add vrouter-name vr0 neighbor  
2211::2 remote-as 222 multi-protocol ipv6-unicast prefix-list-out block200
```

## **Static Route**

```
CLI (network-admin@Spine1) > vrouter-static-route-add vrouter-name vr0  
network 7777::/64 gateway-ip 2200::9 distance 20
```

## Layer 3 Table Validation

---

Layer 3 entries can become unsynchronized between the software table and the hardware table when you modify routes while the routes update in the network.

Use the `l3-setting-modify` command to automatically check Layer 3 entries:

```
CLI (network-admin@spine1) > l3-setting-modify
```

---

One or more of the following options:

<code>aging-time <i>seconds</i></code>	Specify the aging-time for Layer 3 entries, use 0 to disable aging.
<code>convergence-time <i>seconds</i></code>	Specify the unicast convergence time on bootup (seconds)
<code>l3-checker no-l3-checker</code>	Specify if you want to check Layer 3 consistency
<code>l3-checker-interval duration: #d#h#m#</code>	Specify the interval for Layer 3 consistency checker
<code>l3-checker-fix no-l3-checker-fix</code>	Enable fixing of inconsistent entries

---

Netvisor ONE uses two commands to manually check and fix Layer 3 inconsistencies:

```
CLI (network-admin@spine1) > l3-check-show
```

```
CLI (network-admin@spine1) > l3-check-fix
```



## Displaying Hardware Routes History

---

You can display the history of hardware routes in the RIB table. This is useful when troubleshooting hardware routing in the network.

```
CLI (network-admin@spine1) > vrouter-rib-history-show time 15:30
```

time	caller	reason	ip	prelen	nexthop	flags	vlan	intf_ip	intf_id
15:29:41	router-if	add	10.16.111.0	24	10.16.111.1	in-hw,local-subnet	111	10.16.111.1	
15:29:43	router-if	add	10.16.100.0	24	10.16.100.1	in-hw,local-subnet	100	10.16.100.1	1

You can also modify the settings for collecting the history:

```
CLI (network-admin@spine1) > vrouter-rib-history-settings-modify
```

## Understanding ECMP Path Selection and Load Balancing

---

Pluribus Networks supports a number of technologies for traffic load balancing and path redundancy, such as LAG and vLAG.

For Layer 3 designs, Netvisor ONE supports standard routing protocols such as OSPF and BGP. When multiple next hops exist, Netvisor ONE employs ECMP (Equal-Cost Multi-Path) routing to choose amongst the available paths for traffic load balancing and redundancy.

In order to perform path selection in hardware, ECMP uses a high-performance technology called *packet field hashing*.

What that means is that the hardware extracts a number of packet fields and with them performs a special calculation to generate a *hardware index*. Such index is then used to select a next hop for an ECMP routing decision. Up to 16 next hops are supported for load balancing by the hardware-based hashing function.

## About Layer 3 Hardware Hashing

A hardware hashing operation comprises two parts that can be performed at extremely high speed: first, packet field selection and extraction, then a mathematical bitwise computation.

The first step therefore is for the hardware parsing logic to select a number of packet fields 'to be hashed' depending on the traffic type. These are the traffic types and related packet fields that Netvisor ONE currently supports *by default* to perform hashing on:

**Table Caption 5-1: Hardware Hashing Field Selection**

Application	Traffic Type <sup>1</sup>	Hashed Fields
Plain unicast and multicast IPv4 packets	IP	Source IPv4 address, Destination IPv4 address, VLAN number, Destination Layer 4 port, Source Layer 4 port, IP protocol number, source physical port
Plain unicast and multicast IPv6 packets	IP	Source IPv6 address, Destination IPv6 address, VLAN number, Destination Layer 4 port, Source Layer 4 port, IP protocol number, source physical port
VXLAN packets (UDP encapsulated)	IP	Source IP address, Destination IP address, VLAN number, Destination Layer 4 port, <i>Source Layer 4 port</i> , IP protocol number, source physical port

**Note:** <sup>1</sup> Traffic types are based on IANA's Ethertype definitions (e.g., 0x0800 for IPv4 and 0x86DD for IPv6). Packet fields are based on standard protocol definitions (RFC 791 and RFC 8200 for IPv4 and IPv6 respectively).

Once the proper field values are extracted based on the traffic type (as per the table above), the second step performed by the hardware is the *hash function calculation*. Netvisor ONE uses the standard *CRC16 CCITT* cyclic redundancy check algorithm for the hash function calculation.

However, before calculating the CRC, 128-bit IPv6 source and destination addresses get 'folded' into 32-bit shortened versions with the *FoldAndXOR* method:

$$\text{address\_bits}(127:96) \wedge \text{address\_bits}(95:64) \wedge \text{address\_bits}(63:32) \wedge \text{address\_bits}(31:0)$$

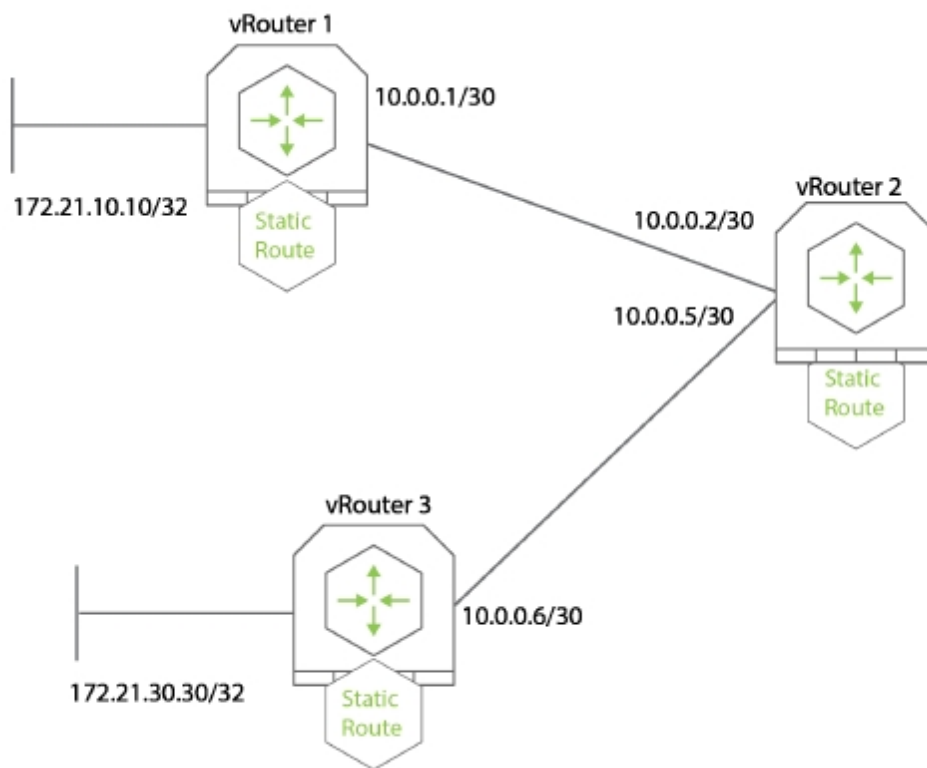
where  $\wedge$  means 32-bit XOR.

This preserves the variability over the entire 128-bit address length when fed in 32-bit chunks to the CRC16 CCITT calculation.

## Configuring Static Routes

vRouters forward packets using either routing information from route tables manually configured or routing information calculated using dynamic routing algorithms.

Static routes define explicit paths between two vRouters and are not automatically updated. When network changes occur, you have to reconfigure static routes. Static routes use less bandwidth than dynamic routes.



**Figure 5-4: Configuring a Static Route**

In this example, you configure a static route on vRouter1 for the network, 172.16.10.10/24 with a gateway IP address, 172.16.20.1:

```
CLI (network-admin@switch) > vrouter-static-route-add vrouter-name vrouter1
network 172.16.10.10/24 gateway-ip 172.16.20.1
```

## Adding IPv6 Link-Local Addresses for Static Routing

IPv6 link-local addresses have a specific prefix `fe80::/10` and are relevant only on the local link. This makes it possible to have the same link-local address on multiple IPv6 interfaces. If you add a static route reachable through a link-local address, you must specify the outgoing interface.

Use the command to specify the interface for the static route:

```
CLI (network-admin@Leaf1) > vrouter-static-route-add
```

<code>vrouter-name <i>name-string</i></code>	Specify the name of the vRouter service.
<code>network <i>ip-address</i></code>	Specify the IP subnet of the network that you want to add a static router.
<code>netmask <i>netmask</i></code>	Specify the netmask of the IP subnet.
<code>gateway-ip <i>ip-address</i></code>	Specify the IP address of the gateway that you want to route packets destined for the network IP address.
<code>bfd-dst-ip <i>ip-address</i></code>	Specify the destination IP address for BFD monitoring.
<code>distance <i>number</i></code>	<p>Specifies the administrative distance in a number from 0 to 255:</p> <ul style="list-style-type: none"> <li>• 0 – Connected interface</li> <li>• 1 – Static route</li> <li>• 110 – OSPF</li> <li>• 120 – RIP</li> <li>• 200 – Internal BGP</li> </ul>
<code>interface vrouter interface <i>nic</i></code>	Specify the vRouter interface for the static route.

You can remove or display the state using the commands `vrouter-static-route-remove`, and `vrouter-static-route-show`.

## Configuring Static Null Routing

You add a static route to a Null 0 interface when an access server with many clients is in your network. This causes Netvisor ONE to install host routes in the access server routing table. To ensure reach-ability to these clients, while not flooding the entire network with host routes, other routers in the network typically have a summary route pointing to the access server. In this type of configuration, the access server has the same summary route pointing to the access server Null 0 interface. If not, routing loops may occur when outside hosts attempt to reach IP addresses not currently assigned to a client but are part of the summary route. The access server bounces the packets back over the access server default route into the core network, and because the access server lacks a specific host route for the destination.

This feature introduces a new parameter for the command, `vrouter-static-route-add`:

```
CLI (network-admin@Leaf1) > vrouter-static-route-add vrouter-name vr1
network 192.168.20.0/24 gateway-ip ip-address
drop          Drop packets matching this route
```

Netvisor ONE designates the drop keyword as mutually exclusive with keywords such as gateway or interface. Either drop or gateway must be used for valid syntax. Static routes with gateways cannot co-exist with drop routes for the same prefix.

```
CLI (network-admin@Leaf1) > vrouter-static-route-show
```

vrouter-name	network	gateway-ip	bfd-dst-ip	distance	interface	drop
vr1	192.168.20.0/24	::	::	1		yes
vr1	192.168.30.0/24	192.168.100.2	::	1		no
vr1	192.168.22.0/23	::	::	1		yes

```
CLI (network-admin@Leaf1) > vrouter-routes-show format network, type,
interface, next-hop, distance
```

vrouter-name	network	type	interface	next-hop	distance
vr1	192.168.20.0/24	static	Null0		1
vr1	192.168.22.0/23	static	Null0		1
vr1	192.168.30.0/24	static	eth2.4092	192.168.100.2	1
vr1	192.168.100.0/24	connected	eth2.4092		
vr1	2018:192:168:100::/96	connected	eth2.4092		
vr1	fe80::/64	connected	eth2.4092		

## Configuring Static ARP for Unicast Traffic

A static Address Resolution Protocol (ARP) entry is a permanent entry in your ARP cache. You can create mappings between IP addresses and MAC addresses (called static ARP entry), which can remain active without aging out for the specified time. Netvisor ONE enables you to create static ARP entries for security and troubleshooting purposes.

One of the use cases in configuring static ARP table entries is to keep the Layer 2 and Layer 3 entries active for select hosts (IP addresses) even if those hosts do not send traffic very often. Another reason to add a static ARP entry is if you want two hosts to communicate regularly and can have a binding of the MAC address and the IP address of the hosts in general.

**Note:** You must create a vRouter interface on the switch to send ARP requests and enable ARP binding.

The static ARP binding is established only after the first ARP packet is communicated (received and replied) between the vRouter and the host. Thereafter, the binding remains permanent by sending and receiving the ARP requests and replies as in the case of *host refresh* requests.

**Note:** You can enable a static ARP on a Layer 3 entry only if the Layer 3 entry is active (see `l3-table-show` output below).

To create a static ARP entry using MAC address, use the command:

```
CLI (network-admin@switch1) > static-arp-create scope [local|cluster|fabric] mac mac-address ip ip-address
```

<code>static-arp-create</code>	Creates a unique static ARP entry
<code>scope [local cluster fabric]</code>	Specify the scope. A static entry with scope, <code>cluster</code> or <code>fabric</code> creates a static ARP entry configuration on one switch and enables the static flag for Layer 3 entry in the L3 table for all the other switches in the fabric or cluster.
<code>mac mac-address</code>	Unicast MAC address to bind with the IP address of the host
<code>ip ip-address</code>	IP address of the host to bind with the MAC address.
<code>vlan/vxlan (optional keywords)</code>	Specify the VLAN ID or VXLAN name for the static ARP entry configuration. You cannot specify both VLAN and VXLAN on the same configuration at the same time. You can specify either VLAN or VXLAN (both are optional parameters in establishing a static ARP entry.



To have the static ARP entries functional, you must have an active Layer 2 entry and a corresponding Layer 3 entry in the system.

After configuring the static ARP entry using the above CLI command, follow the tasks to ensure a working configuration:

- Check the matching Layer 3 entry (i.e, the host interface IP address and MAC address) on the switch
  - If there is no information available, no action is required. You can view the details using the `static-arp-show` command.
  - If a matching Layer 3 entry is found, that entry is marked with a *static* flag and you can verify this using the `l3-table-show ip <ip-address> mac <mac-address>` command.
- **When the Layer 3 entry age-out timer expires**, The static Layer 3 entries are kept alive by forced *arp-refresh*. That is, the vRouter sends an ARP request to the host and when a ARP reply is received. the L3 entry gets refreshed and remains active. This ARP reply ensures that the Layer 2 entry is refreshed, thereby, keeping the Layer 3 entry active and hence keeping the static ARP binding intact.
- If no ARP reply is received (when the port is down), then the Layer 2 entry ages out and the corresponding Layer 3 entry also ages out, keeping the *static* flag on the Layer 3 entry.
- If the ARP replies are received on a different port for the same IP address or MAC address, that triggers a modification of the previous Layer 2 entry with the new port details, which reactivates the previous Layer 3 entry and send out the ARP refresh messages. (MAC move happens)
- When the Layer 2 entry age-out timer expires, it deactivates the Layer 2 entry and the corresponding Layer 3 entry.

**Note:** While configuring static ARP, ensure that:

- the IP address is not 0.0.0.0 or :: for IPv6 addresses.
- the IP address is not multicast or broadcast
- the MAC is unicast only.

To delete a static ARP entry, use the command:

```
CLI (network-admin@switch1) > static-arp-delete ip ip-address
```

When you delete the static ARP configuration, the corresponding Layer 3 entry is cleared off the *static* flag. In cases where there are multiple static ARP entries within the same VLAN, you must use specific parameters to delete the static ARP entry. For example, in such cases, include the required parameters in the command:

```
CLI (network-admin@switch1) > static-arp-delete ip ip-address mac mac-address vlan/vxlan
```

To view the details, use the command:

```
CLI (network-admin@switch1) > static-arp-show scope [local|cluster|fabric] mac mac-address ip ip-address
```

Below is a sample configuration for creating static ARP using the commands described earlier in this section. To create a static ARP binding between the host IP address 172.179.1.120 with the host MAC address 00:12:c0:88:0c:1d, use the command:

```
CLI (network-admin@switch1) > static-arp-create ip 172.148.0.0 mac  
00:12:00:88:0c:00
```

```
CLI (network-admin@switch1) > static-arp-show scope local ip 172.148.0.0
```

switch	scope	ip	mac
switch1	local	172.148.0.0	00:12:00:88:0c:00

When the host is actively sending ARP replies, the show command displays:

```
CLI (network-admin@switch1) > l3-table-show ip 172.148.0.0 format all show-  
interval 1 layout vertical
```

```
mac:                00:12:00:88:0c:00  
ip:                  172.148.0.0  
vlan:                4092  
intf:                29  
hw-intf:             29  
rt-if:               eth1.4092  
state:             active,static  
egress-id:           100008  
hit:                 1
```

When the static ARP entry is removed, the show output displays:

```
CLI (network-admin@switch1) > l3-table-show ip 172.148.0.0 format all show-  
interval 1 layout vertical
```

```
mac:                00:12:00:88:0c:00  
ip:                  172.148.0.0  
vlan:                4092  
intf:                29  
hw-intf:             29  
rt-if:               eth1.4092  
state:             active  
egress-id:           100008  
hit:                 23
```

## Configuring VRF Aware Static ARP

---

Netvisor ONE offers static ARP support for VRF subnets configured with VXLAN. To have the static ARP entries functional, you must have an active VRF with subnets and a corresponding Layer 3 entry in the system. You must also create a subnet anycast gateway IP on the switch to send ARP requests. As in the case of static ARP on vRouters where ARP-reply is sent to the vRouter interface alone, the ARP-reply in this case is sent only to the subnet anycast gateway IP. For VRF configuration details, refer to the VRF section in the VXLAN chapter.

To create a static ARP entry on VXLAN enabled VRF, use the command `static-arp-create`. For example:

```
CLI (network-admin@switch1) > static-arp-create scope cluster mac
00:2d:01:00:00:02 ip 30.1.2.200 vxlan 3012
```

To view the configuration details for static ARP and VRF, use the following commands:

```
CLI (network-admin@switch1) > static-arp-show
```

scope	ip	vxlan	vlan	mac
cluster	30.1.2.200	3012	0	00:2d:01:00:00:02

```
CLI (network-admin@switch1) > vrf-show layout vertical
```

name:	vrf1
vnet:	0:0
scope:	local
anycast-mac:	64:0e:94:40:00:02
vrf-gw:	::
vrf-gw2:	::
vrf-gw-ipv6:	::
vrf-gw2-ipv6:	::
active:	yes
hw-router-mac	hw-vrid
enable	yes

```
CLI (network-admin@switch1) > subnet-show layout vertical
```

name:	sub2
scope:	local
vlan:	3012
vxlan:	3012
vrf:	vrf1
network:	30.1.2.0/24
anycast-gw-ip	30.1.2.1
packet-realy:	disable
forward-proto:	dhcp
state:	ok

enable: yes

To view the L3 table when the host is actively sending ARP replies, use the command `l3-table-show`.

```
CLI (network-admin@switch1) > l3-table-show vxlan 3012 state active layout
vertical
```

```
switch:          switch1
mac:             00:2d:01:00:00:02
ip:             30.1.2.200
vlan:           3012
vxlan:          3012
rt-if:
state:           active, static
egress-id:
tunnel:
switch:          switch2
mac:             00:2d:01:00:00:02
ip:             30.1.2.200
vlan:           3012
vxlan:          3012
rt-if:          sub2
state:           active, static, vxlan-loopback
egress-id:      100064
tunnel:          auto-tunnel-10.30.0.1_10.30.1.1
```

After the static entry is removed, the L3 table output does not show the 'static' state. This can be seen from the example below:

```
CLI (network-admin@switch1) > l3-table-show vxlan 3012 state active layout
vertical
```

```
switch:          switch1
mac:             00:2d:01:00:00:02
ip:             30.1.2.200
vlan:           3012
vxlan:          3012
rt-if:
state:           active
egress-id:
tunnel:
switch:          switch2
mac:             00:2d:01:00:00:02
ip:             30.1.2.200
vlan:           3012
vxlan:          3012
rt-if:          sub2
state:           active, vxlan-loopback
egress-id:      100064
tunnel:          auto-tunnel-10.30.0.1_10.30.1.1
```

## Guidelines and Limitations While Configuring Static ARP

---

- The host refresh time is twice the Layer 3 aging time (2\*L3 age).
- You must create a vRouter interface on the switch to send ARP request. For VRF- aware static ARP, a VRF subnet should be created.
- Host sends the ARP-reply to the vRouter interface only. In the case of VRF-aware static ARP, the ARP-reply is sent to subnet anycast gateway IP.
- In cases where there are multiple static ARP entries within the same VLAN, use the `static-arp-delete ip ip-address mac mac-address vlan/vxlan` command to delete the specific entry.
- In the beginning, the Layer 3 entry should be activated with ping for the static ARP entry to be functional.
- Static ARP entry can be created with VLAN or VXLAN options. You cannot have both VLAN and VXLAN at the same time while creating the static ARP entry. For VRF-aware static ARP, the static ARP entry should be created only with the VXLAN option.
- When you create a static ARP entry with scope, `cluster` or `fabric`, Netvisor creates a static ARP entry configuration on one switch and enables the static flag for Layer 3 entry in the L3 table for all the other switches in the fabric or cluster.
- The modify static ARP command is not supported.

## Configuring IPv4 and IPv6 Neighbor Discovery Process and Optimization

---

The IPv6 Neighbor Discovery Process (NDP) uses ICMPv6 messages and solicited-node multicast addresses to determine the link-layer address of a neighbor on the same network (local link), verify the reach-ability of a neighbor, and keep track of neighboring routers. NDP provides the same functionality as ARP in an IPv4 network. NDP has additional features such as auto-configuration of IPv6 addresses and duplicate address detection (DAD).

In an IPv6 Layer 3 network, a Netvisor ONE vRouter can be configured as a First Hop Router and send Router Advertisements to announce the presence, host configuration parameters, routes, and on-link prefixes. In a Layer 2 network, Netvisor ONE can enable NDP optimization to prevent flooding of neighbor solicitation messages.

### Supported NDP Messages

- Router Solicitation (ICMPv6 type 133)
- Router Advertisement (ICMPv6 type 134)
- Neighbor Solicitation (ICMPv6 type 135)
- Neighbor Advertisement (ICMPv6 type 136)
- Redirect (ICMPv6 type 137)

Netvisor ONE sends Neighbor Solicitation messages (ICMPv6 Type 135) on the local link by nodes attempting to discover the link-layer addresses of other nodes on the local link. Netvisor ONE sends the Neighbor Solicitation message to the solicited-node multicast address. The source address in the neighbor solicitation message is the IPv6 address of the node sending the Neighbor Solicitation message. The Neighbor Solicitation message also includes the link-layer address of the source node.

After receiving a Neighbor Solicitation message, the destination node replies by sending a Neighbor Advertisement message (ICPMv6 Type 136) on the local link. The source address in the Neighbor Advertisement message reflects the IPv6 address of the node sending the Neighbor Advertisement message. The destination address reflects the IPv6 address of the node sending the Neighbor Solicitation message. The data portion of the Neighbor Advertisement message includes the link-layer address of the node sending the Neighbor Advertisement message.

After the source node receives the Neighbor Advertisement, the source node and destination node communicate.

Netvisor ONE uses Neighbor Solicitation messages to verify the reach-ability of a neighbor after identifying the link-layer address of a neighbor. When a node requires verification of the reachability of a neighbor, the destination address in a Neighbor Solicitation message includes the unicast address of the neighbor.

Netvisor ONE sends Neighbor Advertisement messages when a change occurs in the link-layer address of a node on a local link. When there is such a change, the destination address for the Neighbor Advertisement includes the all-nodes multicast address.

Netvisor ONE periodically sends Router Advertisement messages (ICMPv6 Type 134) to each IPv6

configured interface of security appliance. Netvisor ONE also sends the Router Advertisement messages to the all-nodes multicast address.

Router Advertisement messages typically include the following information:

- One or more IPv6 prefix the nodes use on the local link to automatically configure the IPv6 addresses.
- Lifetime information for each prefix included in the advertisement.
- Sets of flags that indicate the type of auto-configuration (stateless or stateful) that can be completed.
- Default router information (whether the router sending the advertisement should be used as a default router and, if so, the amount of time (in seconds) the router should be used as a default router).
- Additional information for hosts, such as the hop limit and MTU a host should use in origination packets.
- The amount of time between neighbor solicitation message re-transmissions on a given link.
- The amount of time a node considers a neighbor reachable.

Netvisor ONE sends Router Advertisements in response to Router Solicitation messages (ICMPv6 Type 133). Hosts send Router Solicitation messages at system startup so that the host can immediately auto-configure without waiting for the next scheduled router advertisement message. Router Solicitation messages are usually sent by hosts at system startup, and the host does not have a configured unicast address, the source address in Router Solicitation messages includes the unspecified IPv6 address (0:0:0:0:0:0:0:0). If the host has a configured unicast address, the source address in the message uses the unicast address of the interface sending the Router Solicitation message. The destination address in Router Solicitation messages uses the all-routers multicast address with a scope of the link. When sending a Router Advertisement in response to a Router Solicitation message, the destination address in the Router Advertisement message uses the unicast address of the source of the Router Solicitation message.

Configure the following settings for router advertisement messages:

- The time interval between periodic Router Advertisement messages. Netvisor ONE uses the default time interval of 200 seconds with a range of 3 to 1800 seconds or 500 to 1800000 milliseconds if you specify milliseconds.
- The router lifetime value, which indicates the amount of time IPv6 nodes should consider the switch to be the default router. Valid values range from 0 to 9000 seconds. Netvisor ONE has a default value of 1800 seconds. Entering 0 indicates that the switch is not considered a default router on the selected interface.
- The IPv6 network prefixes in use on the link. In order for stateless auto-configuration to work properly, the advertised prefix length in Router Advertisement messages must always be 64 bits.
- Whether or not an interface transmits Router Advertisement messages. By default, Netvisor ONE automatically sends Router Advertisement messages in response to Router Solicitation messages. If you suppress the Router Advertisement messages, the switch appears as a regular IPv6 neighbor on the link and not as an IPv6 router.

Unless otherwise noted, the interface has specific the Router Advertisement message settings.

To configure NDP, use the `vrouter-interface-config-add` command:

CLI (network-admin@switch) > `vrouter-interface-config-add`

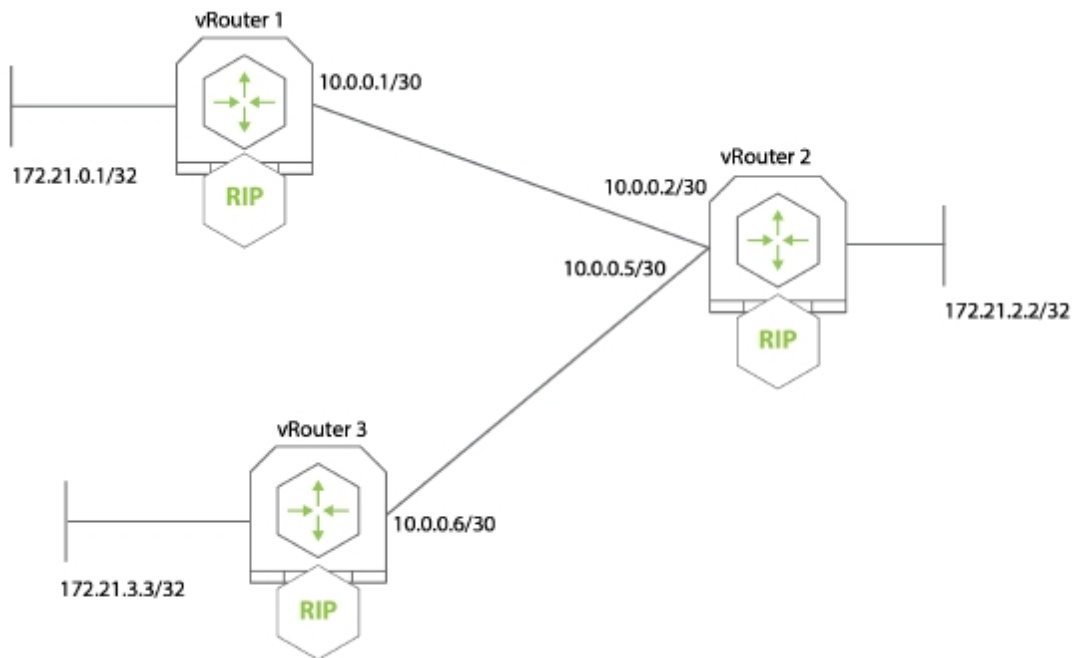
<code>vrouter-name name-string</code>	Specify the name of the service configuration.
Specify one of the options below:	Options:
<code>nic vrouter if-list nic</code>	Specify the vnic name.
<code>ospf-hello-interval &lt;1..65535&gt;</code>	Specify the OSPF hello interval from 1 to 65535. The default value is 10 (IPv4 or IPv6).
<code>ospf-dead-interval &lt;2..65535&gt;</code>	Specify the OSPF dead interval from 2 to 65535. The default value is 40 (IPv4 or IPv6).
<code>ospf-retransmit-interval &lt;3..65535&gt;</code>	Specify the OSPF retransmit interval from 3 to 65535. The default value is 5 (IPv4 or IPv6).
<code>ospf-priority &lt;0..255&gt;</code>	Specify the OSPF priority from 0 to 255. The default value is 1 (IPv4 or IPv6).
<code>ospf-auth-key ospf-auth-key-string</code>	Specify the OSPF authentication key (IPv4 only).
<code>ospf-cost &lt;0..65535&gt;</code>	Specify the OSPF Cost (IPv4 or IPv6).
<code>ospf-msg-digest-id &lt;0..255&gt;</code>	Specify the OSPF digest ID from 0 to 255 (IPv4 only).
<code>ospf-msg-digest-key ospf-msg-digest-key-string</code>	Specify the OSPF message digest key (IPv4 only).
<code>ospf-passive-if no-ospf-passive-if</code>	Specify the OSPF passive interface (IPv4 or IPv6).
<code>ospf-network-type default point-to-point</code>	Specify the OSPF network type (IPv4 or IPv6).
<code>ospf-bfd default enable disable</code>	Specify the BFD protocol support for OSPF fault detection.
<code>bfd-interval &lt;200..3000&gt;</code>	Specify the BFD desired transmit interval from 200 ms to 3000 ms. The default value is 750 ms.
<code>bfd-min-rx &lt;200..3000&gt;</code>	Specify the BFD required minimum receive interval from 200 ms to 3000 ms. The default value is 500 ms.
<code>bfd-multiplier &lt;1..20&gt;</code>	Specify the BFD detection multiplier from 1 to 20. The default value is 3.
<code>nd-suppress-ra no-nd-suppress-ra</code>	Control the transmission of IPv6 Router Advertisements.
<code>ra-interval &lt;1..1800&gt;</code>	Specify the time interval between ipv6 router advertisements.
<code>ra-lifetime &lt;0..9000&gt;</code>	Specify the time for which router is considered as default router.



## Configuring Routing Information Protocol (RIP)

Routing Information Protocol (RIP) remains the oldest routing protocol and provides networking information to routers. Routers need to know the available networks and the distance required to reach it.

RIP consists of a distance vector protocol, and uses hop counts to determine distance and destination. Every 30 seconds, RIP sends routing information to UDP port 520. If you designate the router as default gateway, the router advertises by sending 0.0.0.0 with a metric of 1.



**Figure 5-3: RIP Topology**

1) Create vRouter1 on VLAN33:

```
CLI (network-admin@switch) > vrouter-create name vrouter1 fabricname-global router-type hardware
```

You can also specify how Netvisor ONE distributes RIP routes using the parameter, `rip-redistribute static | connected | ospf | bgp`.

2) Add network 10.16.33.0/24 to vrouter1:

```
CLI (network-admin@switch) > vrouter-rip-add vrouter-name vrouter1 network 10.16.33.0/24 metric 2
```

3) Add network 10.16.35.0/24 to vrouter1:

```
CLI (network-admin@switch) > vrouter-rip-add vrouter-name vrouter1 network 10.16.55.0/24 metric 2
```

4) To view the configuration, use the `vrouter-rip-show` command. This displays all RIP routes configured using the `vrouter-rip-add` command.

To view RIP routes not configured using the `vrouter-rip-add` command, use the `vrouter-rip-routes-show` command.

## Configuring Open Shortest Path First (OSPF)

Open Shortest Path First (OSPF) is a robust link-state interior gateway protocol (IGP). It uses the concept of Areas which allows further segmentation on the network.

OSPF uses link-state information to make routing decisions, and make route calculations using the shortest path first (OSPF) algorithm. Each vRouter configured for OSPF floods link-state advertisements throughout the area that contains information about the interfaces attached to the router and routing metrics.

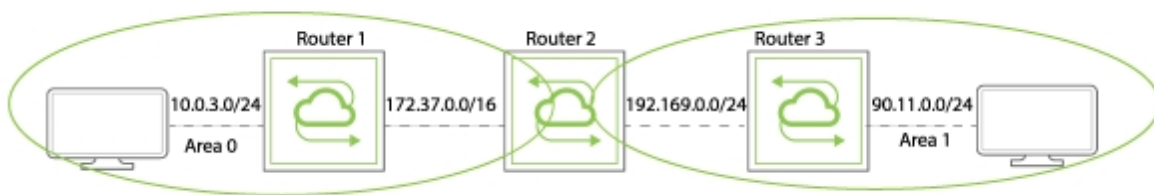
You can add more configuration options, such as hello intervals, for OSPF using the `vrouter-interface-config` commands. In addition, you can add stub or not-so-stubby areas to the OSPF configuration.

You can manually change the OSPF cost for the configuration. Cost is the metric used by OSPF to judge the feasibility of a path. If you specify 0 as the cost, the vRouter automatically calculates the cost based on the bandwidth of the interface. In Netvisor ONE, the OSPF value does not change.

**Note:** For switches with ONVL, the only available vNET is a global vNET created when a fabric is created for the first time. Use `tab complete` in the CLI to display the vNET and continue the configuration.

In this example, you configure OSPF for two vRouters with an area, zero. The network has the following configuration:

- VLAN 35 with IP addresses 10.0.3.0/24
- VLAN 55 with IP addresses 172.37.0.0/24



**Figure 5-1 - OSPF**

1) First, create the vRouter for Router1.

```
CLI (network-admin@switch) > vrouter-create name vrouter1 vnet vnet-  
name fabricname-global router-type hardware
```

2) Add vRouter interfaces to the vRouter:

```
CLI (network-admin@switch) > vrouter-interface-add vnet vnet-name vrouter-  
name vrouter1 ip 10.0.3.0 netmask 24 vlan 35 if data nic-enable
```

```
CLI (network-admin@switch) > vrouter-interface-add vrouter-name vrouter1 ip
172.37.0.0 netmask 16 vlan 55 if data nic-enable
```

3) Add the subnets, 10.0.3.0/24 and 172.37.0.0/16, to VLAN35 with the area 0:

```
CLI (network-admin@switch) > vrouter-ospf-add vrouter-name vrouter1 network
10.0.3.0/24 ospf-area 0
```

4) Add the second IP address with the area 0.

```
CLI (network-admin@switch) > vrouter-ospf-add vrouter-name vrouter1 network
172.37.0.0/16 ospf-area 0
```

5) Add interfaces for OSPF hello intervals of 30 seconds:

```
CLI (network-admin@switch) > vrouter-interface-config-add name router1 nic
eth0.35 ospf-hello-interval 30 ospf-cost 0
```

```
CLI (network-admin@switch) > vrouter-interface-config-add name router1 nic
eth0.55 ospf-hello-interval 30 ospf-cost 0
```

If you specify 0 as the cost value, the vRouter calculates the OSPF cost automatically based on the bandwidth of the interface. When you modify the OSPF hello interval, the `ospf-dead-interval` is automatically reset to 4 times the hello interval.

6) Display the configuration by using the `vrouter-ospf-show` command:

```
CLI (network-admin@switch) > vrouter-ospf-show layout vertical
```

```
vrouter-name:          vrouter1
network:               10.0.3.0
netmask:               24
ospf-area:             0
vrouter-name:          vrouter1
network:               172.37.0.0
netmask:               16
ospf-area:             0
stub-area:             11
stub-type:             stub
ospf-hello-interval:   30
metric:                34
```

The metric value can reflect the cost of routes advertised as OSPF routes. It may also reflect the cost of routes advertised with other protocols.

## Displaying Default Timers for OSPF Configurations

Netvisor One now allows you to display default timers for OSPF configurations. To add a timer or modify an existing timer, use the following commands:

```
CLI (network-admin@Leaf1) > vrouter-interface-config-add
```

<code>ospf-retransmit-interval seconds</code>	Specify the OSPF retransmit interval from 3 to 65535 seconds. The default is 5 seconds. (IPv4 or IPv6)
---	--

```
CLI (network-admin@Leaf1) > vrouter-interface-config-modify
```

<code>ospf-retransmit-interval seconds</code>	Specify the OSPF retransmit interval from 3 to 65535 seconds. The default is 5 seconds. (IPv4 or IPv6)
---	--

A new command displays the OSPF configuration for an interface:

```
CLI (network-admin@Leaf1) > vrouter-ospf-interface-show
```

<code>vrouter-name name-string</code>	Displays the name of the vRouter.
---------------------------------------	-----------------------------------

Specify any of the following optional OSPF parameters:

<code>nic vrouter interface nic</code>	Displays the name of the vNIC.
--	--------------------------------

<code>nic-state down up</code>	Displays the vNIC state.
--------------------------------	--------------------------

<code>l3-port l3-port-number</code>	Displays the Layer 3 port numbers.
-------------------------------------	------------------------------------

<code>ip ip-address</code>	Displays the IPv4 address of the interface.
----------------------------	---

<code>netmask netmask</code>	Displays the netmask of the IPv6 address.
------------------------------	---

<code>broadcast ip-address</code>	Displays the broadcast IP address.
-----------------------------------	------------------------------------

<code>area ip-address</code>	Displays the area ID for the interface in IPv4 format.
------------------------------	--

<code>mtu mtu-number</code>	Displays MTU for the interface.
-----------------------------	---------------------------------

<code>mtu-mismatch-detection </code>	Displays if MTU mismatch detection is configured.
--------------------------------------	---

---

`no-mtu-mismatch-detection`

`router-id ip-address`

Displays the router ID as an IP address.

`network-type point-to-point |  
broadcast | loopback`

Displays the OSPF network type.

`state down | loopback | waiting | point-  
to-point | dr-other | backup | dr`

Displays OSPF interface state.

`dr-id ip-address`

Displays the designated router ID.

`dr-ip ip-address`

Displays the designated router IP address.

`bdr-id ip-address`

Displays the backup designated router ID.

`bdr-ip ip-address`

Displays the designated router IP address.

`priority priority-number`

Displays the priority.

`cost cost-number`

Displays the cost.

`hello hello-number(s)`

Displays the hello-interval in seconds.

`dead dead-number(s)`

Displays the dead time in seconds.

`retransmit retransmit-number(s)`

Displays the retransmit interval time in seconds.

`hello-due hello-due-string`

Displays the hello due in.

`neighbor neighbor-number`

Displays the neighbor count.

`adjacent adjacent-number`

Displays the adjacent number count.

---

CLI (network-admin@Leaf1) > `vrouter-ospf6-interface-show`

---

`vrouter-name name-string`

Displays the name of the vRouter.

Specify any of the following optional OSPF parameters:

`nic vrouter interface nic`

Displays the name of the vNIC.

---

---

<code>nic-state down up</code>	Displays the vNIC state.
<code>l3-port l3-port-number</code>	Displays the Layer 3 port numbers.
<code>link-local ip-address</code>	Displays the IPv6 link-local IP address.
<code>ip6 ip-address</code>	Displays the IPv6 address of the interface.
<code>netmask-ip6 netmask</code>	Displays the netmask of the IP address.
<code>area ip-address</code>	Displays the area ID for the interface in IPv4 format.
<code>mtu mtu-number</code>	Displays MTU for the interface.
<code>mtu-mismatch-detection  no-mtu-mismatch-detection</code>	Displays if MTU mismatch detection is configured.
<code>state down loopback waiting point- to- point dr-other backup dr</code>	Displays OSPF interface state.
<code>dr-id ip-address</code>	Displays the designated router ID.
<code>bdr-id ip-address</code>	Displays the backup designated router ID.
<code>priority priority-number</code>	Displays the priority.
<code>cost cost-number</code>	Displays the cost.
<code>hello hello-number(s)</code>	Displays the hello-interval in seconds.
<code>dead dead-number(s)</code>	Displays the dead time in seconds.
<code>retransmit retransmit-number(s)</code>	Displays the retransmit interval time in seconds.
<code>if-scoped-lsa if-scoped-lsa-number</code>	Displays the number of interface LSAs scoped for the area.
<code>ls-update ls-update-number</code>	Displays the number of pending LSAs for LSUUpdate.
<code>ls-ack ls-ack-number</code>	Displays the number of pending LSAs for LSAck.

---

## Configuring Route Maps for OSPF Routes

Configure route maps to associate a redistributed metric or metric-type for OSPFv3. You can define a route map to prevent OSPF routes from getting added to the routing table. This filtering happens at the time when OSPF installs the route in the routing table.

Before you configure route maps, configure a list of prefixes using the following command:

```
CLI (network-admin@Leaf1) > vrouter-prefix-list-add
```

Use the following set of commands to configure route maps for OSPF:

```
CLI (network-admin@Leaf1) > vrouter-route-map-add
```

<code>vrouter-name <i>name-string</i></code>	Specify the name of the vRouter service.
<code>name <i>name-string</i></code>	Specify the name for the route map.
<code>seq <i>integer</i></code>	Specify the sequence number as an integer between 1 and 4294967295.
<code>action permit deny</code>	Specify the route map action as permit or deny.
<code>match-prefix vrouter prefix-list <i>name-string</i></code>	Specify the name of the prefix list used for the route map.
<code>community-attribute unset none no-export no-advertise internet local-AS</code>	Specify the community attribute for the route map.
<code>local-pref <i>integer</i></code>	Specify the local preference as an integer between -1 and 4294967295.
<code>metric none</code>	Specify the metric as none.
<code>metric-type 1 2</code>	Specify the metric type as 1 or 2.

```
CLI (network-admin@Leaf1) > vrouter-route-map-remove
```

<code>vrouter-name <i>name-string</i></code>	Specify the name of the vRouter service.
<code>name <i>name-string</i></code>	Specify the name for the route map.
<code>seq <i>integer</i></code>	Specify the sequence number as an integer between 1 and 4294967295.

```
CLI (network-admin@Leaf1) > vrouter-route-map-show
```

<code>vrouter-name <i>name-string</i></code>	Displays the name of the vRouter
--	----------------------------------



---

	service.
<code>name <i>name-string</i></code>	Displays the name for the route map.
<code>seq <i>integer</i></code>	Displays the sequence number as an integer between 1 and 4294967295.
<code>action permit deny</code>	Displays the route map action as permit or deny.
<code>match-prefix vrouter prefix-list <i>name-string</i></code>	Displays the name of the prefix list used for the route map.
<code>community-attribute unset none no-export no-advertise internet local-AS</code>	Displays the community attribute for the route map.
<code>local-pref <i>integer</i></code>	Displays the local preference as an integer between -1 and 4294967295.
<code>metric none</code>	Displays the metric as none.
<code>metric-type 1 2</code>	Displays the metric type as 1 or 2.

---

## Enabling or Disabling OSPF SNMP MIBs

---

You now enable or disable SNMP MIBs for OSPF configurations by using the command, `vrouter-create`, or `vrouter-modify`:

```
CLI (network-admin@Leaf1) > vrouter-create name-string ospf-snmp|no-ospf-  
snmp ospf-snmp-notification|no-ospf-snmp-notification
```

## Configuring Default Route Information Settings for OSPF Routing

An OSPF vRouter, by default, does not generate a default route into the OSPF domain. In order for OSPF to generate a default route, you must explicitly configure the vRouter with this information.

There are two ways to inject a default route into a normal area.

- 1) If the ASBR already has the default route in the routing table, you can advertise the existing 0.0.0.0/0 into the OSPF domain with the `ospf-default-information` parameter.
- 2) If the ASBR doesn't have a default route, you can add the keyword `always` to the `ospf-default-information` command. However, we recommend not to add the `always` keyword.

The OSPF vRouter advertises a default route into the OSPF domain, even if a route to 0.0.0.0 is configured.

When you configure the metric type, you can use the parameters described for the `vrouter-*` commands for configuring a route map. These parameters control default route generation for IPv4 and IPv6 default routes.

CLI (network-admin@Leaf1) > `vrouter-create`

```
ospf-default-information none |
originate | always
```

Specify if you want to use the default route information for OSPF:

- `none` — no default route is generated.
- `originate` — the default route is generated only if a default route is present in the routing table.
- `always` — the default route is generated even if no default route is present in the routing table.

```
ospf-default-info-originate-metric
none
```

Specify the metric for the default route.

```
ospf-default-info-originate-
metric-type 1 | 2
```

Specify the metric type as 1 or 2.

CLI (network-admin@Leaf1) > `vrouter-modify`

```
ospf-default-information none |
originate | always
```

Specify if you want to use the default route information for OSPF:

- `none` — no default route is generated.
- `originate` — the default route is generated only if a default route is present in the routing table.
- `always` — the default route is generated even if no default route is present in the routing table.

```
ospf-default-info-originate-metric
none
```

Specify the metric for the default route.

```
ospf-default-info-originate-
metric-type 1|2
```

Specify the metric type as 1 or 2.

```
ospf-default-info-originate-route-
map vrouter route-map name
```

Specify the OSPF default information route map.

---

```
CLI (network-admin@Leaf1) > vrouter-show
```

---

Displays the default route information for OSPF:

```
ospf-default-information none|
originate|always
```

- `none` — no default route is generated.
- `originate` — the default route is generated only if a default route is present in the routing table.
- `always` — the default route is generated even if no default route is present in the routing table.

```
ospf-default-info-originate-metric
none
```

Displays the metric for the default route.

```
ospf-default-info-originate-
metric-type 1|2
```

Displays the metric type as 1 or 2.

```
ospf-default-info-originate-route-
map vrouter route-map name
```

Displays the OSPF default information route map.

---

## Configuring Metric and Metric Type for Route Maps

Configure route maps to associate a redistribute metric and metric-type for OSPF and OSPFv3 and you cannot configure metrics using the current OSPF commands. Use route maps method to configure metric and metric-type on routers. Configuration of metrics and metric-type for OSPFv3 requires route maps.

```
CLI (network-admin@Leaf1) > vrouter-route-map-add
```

<code>metric none</code>	Specify a metric for the route map.
--------------------------	-------------------------------------

<code>metric-type none 1 2</code>	Specify the metric type.
-----------------------------------	--------------------------

The new parameters also apply to the `vrouter-route-map-modify` and `vrouter-route-map-show` commands.

You can also specify metrics and route maps for vRouters configured as OSPF and BGP vRouters.

```
CLI (network-admin@Leaf1) > vrouter-modify
```

<code>bgp-redist-static-route-map vrouter route-map name</code>	Specify the route map for BGP redistribution of static routes.
---	--

<code>bgp-redist-connected-route-map vrouter route-map name</code>	Specify the route map for BGP redistribution of connected routes.
--	---

<code>bgp-redist-ospf-route-map vrouter route-map name</code>	Specify the route map for BGP redistribution of OSPF routes.
---	--

<code>ospf-redist-static-route-map vrouter route-map name</code>	Specify the route map for OSPF redistribution of static routes.
--	---

<code>ospf-redist-connected-route-map vrouter route-map name</code>	Specify the route map for OSPF redistribution of connected routes.
---	--

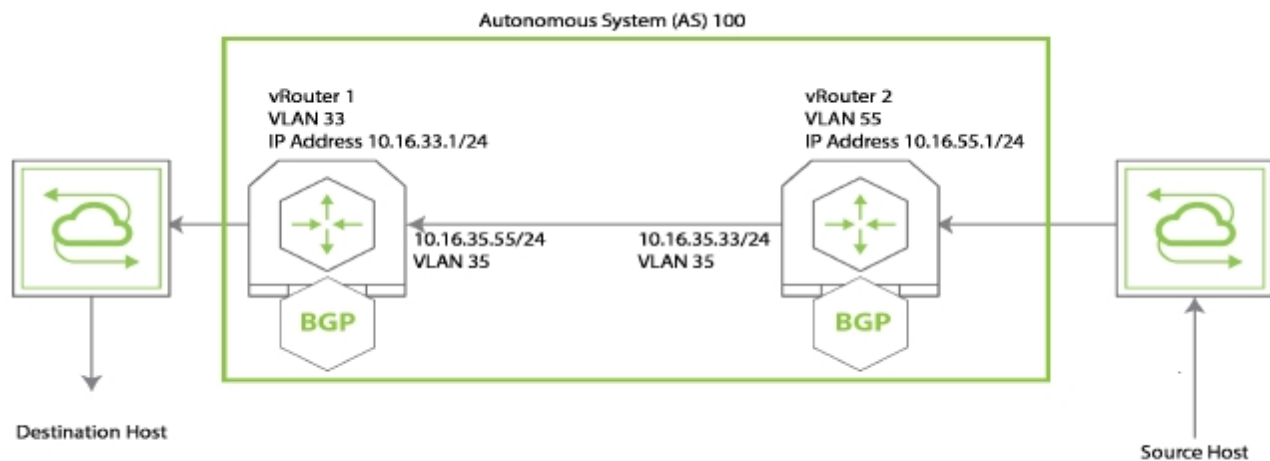
<code>ospf-redist-bgp-route-map vrouter route-map name</code>	Specify the route map for OSPF redistribution of BGP routes.
---	--

## Configuring BGP on a vRouter

Border Gateway Protocol (BGP) is a path-vector protocol and is the most commonly used routing protocol on the Internet. It advertises the paths required to reach a certain destination. BGP is also a protocol that sits on top of TCP, and is simpler than Open Shortest Path First (OSPF).

For example, in Figure 5-2, the network administrator wants to configure network traffic from the source host to reach the destination host. But when different VLANs are configured, the source host traffic is not aware of the route between the source host and the destination host. However, there is a VLAN that spans VLAN 33 and VLAN 55. You can solve this problem by configuring BGP in the same Autonomous System (AS) 100 that sends traffic over VLAN 35. This allows the source host to learn the route to the destination host.

Using a loopback address for peering is useful when there are multiple paths between the BGP peers which would otherwise tear down the BGP session if the physical interface used for establishing the session goes down. It also allows the vRouters running BGP with multiple links between vRouters to load balance over the available paths.



**Figure 5-2: Configuring BGP for Two VLANs**

This example assumes that you have two VLANs, VLAN33 and VLAN55. Also, that you have added ports to the configuration.

Begin by configuring vRouter1, a software vRouter, on VLAN 33 with the BGP information:

```
CLI (network-admin@Leaf1) > vrouter-create name vrouter1 fabricname-  
global router-type hardware bgp-as 100 bgp-redist-connected-metric none
```

Additional BGP parameters include the following:

- `bgp-redist-static-metric` – redistribute static BGP route metric number

- `bgp-redis-connected-metric` — redistribute connected BGP route metric
- `bgp-redis-rip-metric` — redistribute BGP into RIP process metric
- `bgp-redis-ospf-metric` — redistribute BGP into OSPF process metric
- `bgp-max-paths` — maximum number of BGP paths
- `bgp-ibgp-multipath` — allow the BGP vRouter to select multiple paths for load sharing.
- `bgp-bestpath-as-path` — allow BGP to use the best path for traffic forwarding.
- `bgp-dampening` | `no-bgp-dampening` — suppress flapping routes so they are not advertised.
- `bgp-stalepath-time` — how long a router waits before deleting stale routes after an end of record (EOR) message is received from the restarting router.
- `bgp-graceful-shutdown` | `no-bgp-graceful-shutdown` — how to configure BGP graceful shutdown (RFC8326)

Add the IP addresses and VLANs:

```
CLI (network-admin@Leaf1) > vrouter-interface-add vrouter-name vrouter1 ip
10.16.35.33/24 vlan 35
```

```
CLI (network-admin@Leaf1) > vrouter-interface-add vrouter-name vrouter1 ip
10.16.33.1/24 vlan 33
```

Add the BGP information:

```
CLI (network-admin@Leaf1) > vrouter-bgp-add vrouter-name vrouter1 neighbor
10.16.35.55 remote-as 100
```

```
CLI (network-admin@Leaf1) > vrouter-bgp-network-add vrouter-name vrouter1
network 10.16.33.0/24
```

Display the interface information for vrouter1:

```
CLI (network-admin@Leaf1) > vrouter-interface-show format all layout
vertical
```

```
vrouter-name:    vrouter1
nic:             eth1.33
ip:             10.9.100.100/16
assignment:     static
mac:            66:0e:94:30:c6:92
vlan:           33
vxlan:          0
if:             data
alias-on:
exclusive:      no
nic-config:     enable
nic-state:      up
```

```
secondary-macs:
vrouter-name:    vrouter1
nic:             eth2.33
ip:             192.168.42.11/24
assignment:      static
mac:            66:0e:94:30:25:5e
vlan:           33
vxlan:          0
if:             data
alias-on:
exclusive:       no
nic-config:      enable
nic-state:       up
secondary-macs:
```

If you want to filter IP hosts, you can add prefix lists to the BGP configuration. See [Configuring Prefix Lists for BGP and OSPF](#).

Then, configure vRouter2 on VLAN 55:

```
CLI (network-admin@Leaf1) > vrouter-create name vrouter2 fabricname-
global router-type hardware bgp-as 100 bgp-redist-connected-metric none
```

Add the IP addresses and VLANs:

```
CLI (network-admin@Leaf1) > vrouter-interface-add vrouter-name vrouter2 ip
10.16.35.55/24 vlan 35
```

```
CLI (network-admin@Leaf1) > vrouter-interface-add vrouter-name vrouter2 ip
10.16.55.1/24 vlan 55
```

Then add the BGP information:

```
CLI (network-admin@Leaf1) > vrouter-bgp-add vrouter-name vrouter2 neighbor
10.16.35.33 remote-as 100
```

```
CLI (network-admin@Leaf1) > vrouter-bgp-network-add vrouter-name vrouter2
network 10.16.55.0/24
```

And finally, add the loopback address:

```
CLI (network-admin@Leaf1) > vrouter-loopback-interface-add vrouter-name
vrouter1 index 5 ip 1.1.1.1
```

Display the vRouter BGP configuration:

```
CLI (network-admin@Leaf1) > vrouter-bgp-show format all layout vertical
```

```
vrouter-name:          vrouter33
```



```
ip: 10.16.35.55
neighbor: 10.16.35.55
remote-as: 100
next-hop-self: no
route-reflector-client: no
override-capability: no
soft-reconfig-inbound: no
max-prefix-warn-only: no
vrouter-name: vrouter33
ip: 10.16.33.0
network: 10.16.33.0/24
vrouter-name: vrouter55
ip: 10.16.35.33
neighbor: 10.16.35.33
remote-as: 100
next-hop-self: no
route-reflector-client: no
override-capability: no
soft-reconfig-inbound: no
max-prefix-warn-only: no
vrouter-name: vrouter55
ip: 10.16.55.0
network: 10.16.55.0/24
```

To reset BGP neighbors, use the `vrouter-bgp-neighbor-reset` command.

To display BGP neighbors, use the `vrouter-bgp-neighbor-show` command.

```
CLI (network-admin@Leaf1) > vrouter-bgp-neighbor-show
```

```
vrouter-name: vrouter1
neighbor: 10.9.100.201
ver: 4
remote-as: 100
msg_rcvd: 11
msg_sent: 19
tblver: 0
inQ: 0
outQ: 0
up/down: 00:54:04
state/pfxrcd: Connect
vrouter-name: vrouter2
neighbor: 10.9.100.101
ver: 4
remote-as: 100
msg_rcvd: 12
msg_sent: 18
tblver: 0
inQ: 0
outQ: 0
up/down: 00:53:37
```

state/pfxrcl: Connect

## Additional BGP Parameters

There are additional BGP parameters that you can use to optimize your BGP network. Add any of the following parameters:

- `ebgp-multihop` — a value for external BGP to accept or attempt BGP connections to external peers, not directly connected, on the network. This is a value between 1 and 255.
- `update-source vrouter` — the source IP address of BGP packets sent by the router. This parameter is required if you want BGP to perform peering over a loopback interface.
- `prefix-list-in` — specify a list of incoming prefixes for route redistribution.
- `prefix-list-out` — specify a list of outgoing prefixes for route redistribution.
- `override-capability` — override the result of capability negotiation with the local configuration. This parameter allows you to ignore a remote peer's capability value.
- `soft-reconfig-inbound` — defines the route refresh capability by allowing the local device to reset inbound routing tables dynamically by exchanging route refresh requests to supporting peers.
- `max-prefix` — allows you to specify the maximum number of IP prefixes to filter.
- `max-prefix-warn` — add a parameter to warn when the maximum number of prefixes is reached.

## Enabling or Disabling BGP SNMP MIBs

---

You can enable or disable SNMP MIBs for BGP configurations by using the command, `vrouter-create`, or, `vrouter-modify`:

```
CLI (network-admin@Leaf1) > vrouter-create name-string bgp-snmp|no-bgp-snmp  
bgp-snmp-notification|no-bgp-snmp-notification
```

## Configuring AS and AS Prepending on BGP

---

You can prepend one or more autonomous system (AS) numbers at the beginning of an AS path. The AS numbers are added at the beginning of the path after the actual AS number from which the route originates has been added to the path. Prepending an AS path makes a shorter AS path look longer and therefore less preferable to BGP.

The BGP best path algorithm determines how the best path to an autonomous system (AS) is selected. The AS path length determines the best path when all of the following conditions are met:

There are multiple potential routes to an AS.

- BGP has the lowest preference value, sometimes referred to as the administrative distance, of the available routes.
- The local preferences of the available routes are equal.

When these conditions are met, the AS path length is used as the tie breaker in the best path algorithm. When two or more routes exist to reach a particular prefix, BGP prefers the route with the shortest AS Path length.

If you have multi-homing to one or more service providers, you may prefer for incoming traffic take a particular path to reach your network. Perhaps you have two connections, but one costs less than the other. Or you might have one fast connection and another, much slower connection that you only want to use as a backup if your primary connection is down. AS path prepending is an easy method that you can use to influence inbound routing to your AS.

Netvisor One has new parameters for the `vrouter-route-map-*` commands:

```
CLI (network-admin@Leaf1) > vrouter-route-map-add
```

---

<code>as-path-prepend integer</code>	Specify a value between 1 and 4,294,967,295.
<code>as-path-prepend-last-as integer</code>	Specify a value between 1 and 10.
<code>as-path-exclude integer</code>	Specify a value between 1 and 4,294,967,295.

---

```
CLI (network-admin@Leaf1) > vrouter-route-map-show
```

---

<code>as-path-prepend integer</code>	Displays a value between 1 and 4,294,967,295.
<code>as-path-prepend-last-as integer</code>	Displays a value between 1 and 10.
<code>as-path-exclude integer</code>	Displays a value between 1 and 4,294,967,295.

---

## Configuring Border Gateway Protocol (BGP) Communities

A BGP community is a group of prefixes that share some common property and can be configured with the BGP community attribute. The BGP Community attribute is an optional transitive attribute of variable length. The attribute consists of a set of four octet values that specify a community. The community attribute values are encoded with an Autonomous System (AS) number in the first two octets, with the remaining two octets defined by the AS. A prefix can have more than one community attribute. A BGP speaker that sees multiple community attributes in a prefix can act based on one, some or all the attributes. A router has the option to add or modify a community attribute before the router passes the attribute on to other peers.

The local preference attribute is an indication to the AS which path is preferred in order to reach a certain network. When there are multiple paths to the same destination, the path with the higher preference is preferred (the default value of the local preference attribute is 100).

### Common Community Attributes

- **Standard** (well known) — These community attributes are 4 octets long, with well known values
  - Internet (0) — advertise these routes to all neighbors.
  - no-export (0xFFFFF01) — do not advertise to outside a BGP confederation boundary.
  - no-advertise (0xFFFFF02) — do not advertise to other BGP peers .
  - local-AS (0xFFFFF03) — do not advertise to external BGP peers.
- **Standard** - generic (AS:value) — These community attributes are also 4 octet long, but values can be really generic. The first 16-bit number is normally the AS number of the network that sets the community or looks for it, and the second number is one that conveys the intended information, for example: 65001:100.

For example to set the community attribute, **no-export**, to all route prefixes matching prefix **subnet100**, use the following syntax:

```
CLI (network-admin@Leaf1) > vrouter-route-map-add vrouter-name vr1 name  
rmap1 seq 10 action permit match-prefix subnet100 community-attribute no-  
export
```

To set the community attribute, **65002:200** to all route prefixes matching prefix subnet100, use the following syntax:

```
CLI (network-admin@Leaf1) > vrouter-route-map-add vrouter-name vr1 name  
peer vr2 action permit seq 20 match-prefix subnet99 community-attribute-  
generic 65002:200
```

### Community Lists

BGP community list is a user defined BGP communities attribute list. The BGP community list can be used for matching or manipulating BGP communities attribute in updates. This is used on the receive side of the BGP updates to match what is set in the received updates. Community lists can be used in route-map with match-community keyword to apply any policy on the receive side.

- **Standard** — Standard community list defines attribute which matches standard communities as defined above (well known or generic).

To set the community list permitting the community value **300** for AS **65002**, use the following syntax:

```
CLI (network-admin@Leaf1) > vrouter-community-list-add vrouter-name vr2
style standard name clist300 action permit community-attribute 65002:300
```

The Netvisor One commands for `vrouter-route-maps-*` support additional parameters for BGP communities:

```
CLI (network-admin@Leaf1) > vrouter-route-map-add
```

---

<code>match-community match-community-string</code>	Specify the community string to match. (BGP only)
<code>exact-match no-exact-match</code>	Specify if the community string is an exact match or not. (BGP only)
<code>community-attribute-generic community-attribute-generic-string</code>	Specify a generic community attribute such as AA:NN. (BGP only)
<code>additive no-additive</code>	Specify if a given community is appended to existing communities value.
<code>comm-list-del vrouter community-list name</code>	Specify if you want to remove community values from BGP community attributes.

---

New commands support creating BGP Communities:

```
CLI (network-admin@Leaf1) > vrouter-community-list-add
```

---

<code>vrouter-name name-string</code>	Specify a vRouter to add the community list.
Add the following community list parameters:	
<code>style standard</code>	Specify the style of the community list.
<code>name name-string</code>	Specify a name for the community list.
<code>action permit deny</code>	Specify the action for the community list.
<code>community-attribute community-attribute-string</code>	Specify the community attribute.

---

```
CLI (network-admin@Leaf1) > vrouter-community-list-remove
```

---

<code>vrouter-name <i>name-string</i></code>	Specify a vRouter to remove the community list.
--	---

Add the following community list parameters:

<code>style standard</code>	Specify the style of the community list.
-----------------------------	--

<code>name <i>name-string</i></code>	Specify a name for the community list.
--------------------------------------	--

<code>action permit deny</code>	Specify the action for the community list.
---------------------------------	--

<code>community-attribute <i>community-attribute-string</i></code>	Specify the community attribute.
--	----------------------------------

---

CLI (network-admin@Leaf1) > `vrouter-community-list-show`

---

<code>vrouter-name <i>name-string</i></code>	Displays the vRouter name.
--	----------------------------

Add the following community list parameters:

<code>style standard</code>	Displays the style of the community list.
-----------------------------	---

<code>name <i>name-string</i></code>	Displays a name for the community list.
--------------------------------------	---

<code>action permit deny</code>	Displays the action for the community list.
---------------------------------	---

<code>community-attribute <i>community-attribute-string</i></code>	Displays the community attribute.
--	-----------------------------------

---

## Configuring BGP Route Maps for Origin

---

Use the `vrouter-route-map` commands to define the conditions for redistributing routes from one routing protocol to another. The `route-map` command contains a list of `match` and `set` commands associated with it.

The `match` commands specify the match criteria: the conditions under which Netvisor ONE redistributes the current route-map command.

The `set` commands specify the set actions: the particular redistribution actions to perform if the criteria enforced by the `match` commands are met.

```
CLI (network-admin@Leaf1) > vrouter-route-map-modify vrouter-name vr2 name  
TEST6 seq 10 origin <tab>
```

---

<code>none</code>	Resets the origin to none.
<code>egp</code>	Specifies the route as a remote EGP route.
<code>igp</code>	Specifies the route as a local IGP route.
<code>incomplete</code>	Route is of unknown heritage.

---



## Configuring BGP Graceful Shutdown

The BGP Graceful Shutdown functionality enables the network administrators to gracefully shutdown all BGP sessions per vrouter between BGP peer(s). This feature does not shutdown the sessions, but notifies the neighbor peers that a maintenance event is occurring and that the traffic should be re-routed to another peer to avoid traffic black-holing. The *graceful, no traffic impact* behaviour of the feature depends on the topology and BGP setup in a network.

When a BGP speaker is rebooted, upgraded, or shut down during a maintenance window, the admin re-sends all the BGP routes (route refresh) to the neighbor peers by attaching a BGP GRACEFUL\_SHUTDOWN community to each prefix. Upon receiving the prefixes, the BGP neighbors should dynamically change the local preference associated to these subnets to a very low value, typically, LP=0. By changing the local preference to 0, the BGP neighbors select another path (i.e. next-hop) to reach the prefixes carrying the GRACEFUL\_SHUTDOWN community attribute. When the traffic is completely re-routed to alternate path, the BGP speaker can administratively shutdown all BGP sessions without experiencing traffic black-holing issue.

You must configure the vrouter and BGP neighbors prior to enabling the bgp-graceful-shutdown knob on the vrouter.

**Note:** This feature do not support a per-session graceful shutdown, but supports a per-vRouter graceful shutdown (all BGP sessions of a particular vRouter are affected).

To configure BGP graceful shutdown, use the following commands (this command enables you to modify several parameters of BGP and other protocols.):

```
CLI (network-admin@switch1) > vrouter-modify name name-string bgp-graceful-shutdown
```

<code>vrouter-create</code>	Allows you to create a hardware vRouter and specify the parameters for the vRouter.
<code>name <i>name-string</i></code>	Specify a name for the service configuration.
<code>bgp-graceful-shutdown   no-bgp-graceful-shutdown</code>	Specify to enable or disable BGP graceful shutdown (RFC8326).

Below is an example of a vRouter configuration enabling BGP graceful shutdown on **vr2**:

```
CLI (network-admin@switch1) > vrouter-modify name vr2 router-type hardware
bgp-as 65200 router-id 22.22.22.22 bgp-graceful-shutdown
```

To display the details, use the show command (Similar to other BGP knobs the new field is displayed only if the field is listed in the format statement or `format all` keyword is used):

```
CLI (network-admin@switch1) > vrouter-show format name,scope,router-
```

type,bgp-as,router-id,bgp-graceful-shutdown

name	scope	router-type	bgp-as	router-id	bgp-graceful-shutdown
vr1	fabric	hardware	65100	11.11.11.11	off
vr2	fabric	hardware	65200	22.22.22.22	on
vr3	local	hardware	65300	33.33.33.33	off

To disable BGP graceful shutdown knob, the configuration, use the command:

```
CLI (network-admin@switch1) > vrouter-modify name name-string no-bgp-graceful-shutdown
```

## Configuring BGP Unnumbered

The Border Gateway Protocol (BGP) is a path vector protocol used to exchange routing and reachability information. Traditionally, to exchange IPv4 prefixes, you have to configure explicit BGP sessions with the neighbor IP address and remote-AS information for each BGP peer, which can become cumbersome in large networks. This would also occupy significant address space as each BGP peer must have an IPv4 address.

By using RFC5549 (Advertising IPv4 Network Layer Reachability Information with an IPv6 Next Hop), Pluribus enables you to create BGP unnumbered sessions which are simpler to configure. Practically, we need not have an IPv4 address for every BGP-enabled interface and therefore it is called an unnumbered configuration. Instead of specifying an IPv4 neighbor address and the remote-AS number, we can provide the L3 port number and declare an eBGP session. BGP unnumbered uses the link-local IPv6 address as the next-hop IP address for both IPv4 and IPv6 prefixes.

In effect, the need for configuring IPv4 addresses on all BGP-enabled interfaces and the need for declaring an explicit ASN for the remote side is eliminated through BGP unnumbered.

**Note:** iBGP is not supported for unnumbered BGP sessions.

To configure a BGP unnumbered session, use the two following commands:

```
CLI (network-admin@Leaf1) > vrouter-interface-add vrouter-name <> l3-port
<> ipv6-unnumbered
```

<code>vrouter-interface-add</code>	Use this command to add an interface to a vRouter.
<code><b>vrouter-name</b> <i>name-string</i></code>	<b>Specify the name of the vRouter to which the interface is to be added.</b>
<code>vlan &lt;0..4095&gt;</code>	Specify the VLAN ID to which the interface has to be added. (optional). <b>Note:</b> This option is not supported while configuring BGP unnumbered.
Specify any of the following options:	
<code>ip ip-address</code>	Specify the IP address to be assigned to the interface.
<code>Netmask <i>netmask</i></code>	Specify the network mask for the IP address.
<code>Assignment [none   dhcp   dhcpv6]</code>	Specify one among the options as the type of IP address assignment.
<code>ip2 ip-address</code>	Specify the second IP address of the interface.

<code>netmask2 netmask</code>	Specify the network mask for the second IP address.
<code>assignment2 [none   dhcp   dhcpv6]</code>	Specify the type of assignment for the second IP address.
<code>linklocal ip-address</code>	Specify the IPv6 link local address.
<code>ipv6-unnumbered no-ipv6-unnumbered</code>	Specify either of the options to enable or disable IPv6 unnumbered on the interface.
<code>vnet vnet name</code>	Specify the VNET name assigned to the vRouter.
<code>bd bridge-domain name</code>	Specify the Interface bridge domain name.
<code>vlan-type [public   private]</code>	Specify the interface VLAN type.
<code>if [mgmt   data   span   span2   span3]</code>	Specify one among the options as the interface type.
<code>alias-on alias-on-string</code>	Specify an alias for the interface.
<code>exclusive   no-exclusive</code>	Specify if the interface is exclusive to the configuration. Exclusive means that other configurations cannot use the interface. Exclusive is specified when you configure the interface as a span interface and allows higher throughput through the interface.
<code>nic-enable   nic-disable</code>	Specify one of the options to enable or disable the NIC.
<code>pim   no-pim</code>	Specify either of the options to state if the VNIC is a Protocol Independent Multicast (PIM) interface or not.
<code>pim-dr-priority &lt;1..4294967295&gt;</code>	Specify the designated router priority for the PIM interface. Netvisor selects the vRouter interface with higher DR priority as the designated router.
<code>pim-cluster   no-pim-cluster</code>	Specify if the VNIC is on exclusive transit VLAN.
<code>fabric-nic   no-fabric-nic</code>	Specify if the VNIC is on a VLAN used for fabric setup or not.
<code>vrrp-id &lt;0..255&gt;</code>	Specify the VRRP ID.
<code>vrrp-primary vrrp-primary-string</code>	Specify the primary interface for VRRP failover.
<code>vrrp-priority &lt;0..254&gt;</code>	Specify the VRRP priority for the interface. This is a value between 0 (lowest) and 254 (highest).
<code>vrrp-adv-int &lt;300..40950&gt;</code>	Specify the VRRP advertisement interval in ms. The range is from 30 to 40950 with a default value of 1000.

<b>l3-port</b> <i>l3port-usable-port name</i>	<b>Specify the name of the Layer 3 port to be added to the vrouter.</b>
<i>secondary-macs secondary-macs-string</i>	Specify a secondary MAC address.
<i>mtu &lt;68..9398&gt;</i>	Specify a MTU value in bytes between 68 and 9216.
<i>if-nat-realm [internal   external]</i>	Specify the NAT interface realm as internal or external.
<i>priority-tag   no-priority-tag</i>	Specify either of the options to add a VLAN 0 priority tag on forwarded traffic or remove it.

Now, issue the second command to add a BGP neighbor to the vrouter and finish the BGP unnumbered configuration.

```
CLI (network-admin@Leaf1) > vrouter-bgp-add vrouter-name <> l3-port <>
[remote-as external]
```

<i>vrouter-bgp-add</i>	Use this command to add a BGP neighbor to a vrouter.
<i>vrouter-name name-string</i>	Specify the name of the vrouter to which the neighbor is being added.
Specify any of the following options:	
<i>neighbor ip-address</i>	Specify the IP address for BGP neighbor.
<b>l3-port</b> <i>l3port-usable-port name</i>	<b>Specify the L3 port name for the BGP session.</b>
<b>remote-as</b> <i>&lt;external internal -2..429496729&gt;</i>	<b>Supply this parameter to specify if the BGP session is external/internal or to provide an AS number</b>  <b>Note:</b> Specify <i>external</i> to configure a BGP unnumbered session. This parameter is redundant as an eBGP session is assumed for unnumbered BGP sessions.
<i>next-hop-self   no-next-hop-self</i>	Specify either of the options to set the next hop as self or to remove that configuration.
<i>password password-string</i>	Provide a password for MD5 BGP.
<i>ebgp-multihop &lt;1..255&gt;</i>	Specify the value for external BGP between 1 and 255.
<i>update-source vrouter loopback-interface ip</i>	Specify the source IP address of BGP packets sent by the router. This parameter is required if you want BGP to perform peering over a loopback

	interface.
<code>prefix-list-in vrouter <i>prefix-list name</i></code>	Specify the prefix list to filter inbound packets.
<code>prefix-list-out vrouter <i>prefix-list name</i></code>	Specify the prefix list to filter outbound packets.
<code>route-reflector-client no-route-reflector-client</code>	Specify if a route reflector client is used.
<code>override-capability no-override-capability</code>	Specify either of the options to enable or disable override capability.
<code>soft-reconfig-inbound no-soft-reconfig-inbound</code>	Specify either of the options to enable or disable soft reset to reconfigure inbound traffic.
<code>max-prefix <i>max-prefix-number</i></code>	Specify the maximum number of prefixes.
<code>max-prefix-warn-only no-max-prefix-warn-only</code>	Specify either of the options to enable or disable the warning if the maximum number of prefixes is exceeded.
<code>bfd no-bfd</code>	Specify either of the options to enable or disable BFD protocol support for fault detection.
<code>bfd-multihop no-bfd-multihop</code>	Specify either of the options to enable or disable the use of BFD multi-hop port for fault detection.
<code>multi-protocol [ipv4-unicast ipv6-unicast]</code>	Specify either of the options as a multi-protocol feature.
<code>weight [none   -1..65535]</code>	Specify the default weight value for the neighbor's routes either as <code>none</code> or as a value between 0 and 65535.
<code>default-originate no-default-originate</code>	Specify either of the options to enable or disable the announcing of default routes to the neighbor.
<code>neighbor-keepalive-interval &lt;0..65535&gt;</code>	Specify the BGP keepalive interval in seconds. The keepalive interval stipulates how often the the keepalive messages are sent.
<code>neighbor-holdtime &lt;0..65535&gt;</code>	Specify the BGP holdtime in seconds. The hold time specifies how long a router will wait for incoming BGP messages before it assumes the neighbor is dead.
<code>connect-retry-interval &lt;0..65535&gt;</code>	Specify the BGP connect retry interval in seconds.
<code>send-community no-send-community</code>	Specify either of the options to enable or disable the sending of any community attribute to the neighbor.
<code>route-map-in vrouter <i>route-map name</i></code>	Specify the name of the route map for incoming routes.

<code>route-map-out vrouter route-map name</code>	Specify the name of the route map for outgoing routes.
<code>allowas-in no-allowas-in</code>	Specify either of the options to allow/reject routes with local AS in AS_PATH.
<code>interface vrouter interface nic</code>	Specify the Interface to reach the neighbor.
<code>advertisement-interval &lt;0..600&gt;</code>	Specify the minimum interval between sending BGP routing updates.

These commands create an interface with a link-local IPv6 address and configures an eBGP session for the interface.

To view the configuration on the vrouter, use the command, `vrouter-interface-show`. For example:

```
CLI (network-admin@Leaf1) > vrouter-interface-show format nic,l3-port,vlan,ip,ipv6-unnumbered
```

vrouter-name	nic	l3-port	vlan	ip	ipv6-unnumbered
vr1	eth0.4092	17	4092	fe80::640e:94ff:feff:7bdc/64	yes

The BGP configuration can be displayed using the command:

```
CLI (network-admin@Leaf1) > vrouter-bgp-show format l3-port,nic,neighbor,remote-as
```

vrouter-name	l3-port	nic	remote-as
vr1	17	eth0.4092	external

Use the following command to view the relevant neighbor information:

```
CLI (network-admin@Leaf1) > vrouter-bgp-neighbor-show vrouter-name vr2 format neighbor,l3-port,nic,remote-as,up/down
```

vrouter-name	neighbor	l3-port	nic	remote-as	up/down
vr2		17	eth2.4092	65100	00:06:20
vr2	192.168.101.1			65100	00:02:48

The output shows that the neighbor field for the unnumbered interface does not have an IP address in it, while the l3-port and nic fields have information. It can also be noted that a BGP unnumbered session can co-exist with a numbered BGP session, but not on the same interface.

The Routing Information Database (RIB) and Forwarding Information Database (FIB) confirms that both IPv4 and IPv6 prefixes learned by BGP use IPv6 next-hop addresses:

```
CLI (network-admin@Leaf1) > vrouter-rib-routes-show format ip,prelen,nexthop,flags,vlan
```

ip	prelen	nexthop	flags	vlan
192.168.111.0	24	fe80::640e:94ff:feff:7bdc	in-hw	4092
5001:11:1::	48	fe80::640e:94ff:feff:7bdc	in-hw	4092

```
CLI (network-admin@Leaf1) > vrouter-fib-routes-show format ip,prelen,vlan,port,nexthop-
mac,egress-id
```

ip	prelen	vlan	port	nexthop-mac	egress-id
192.168.111.0	24	4092	17	66:0e:94:ff:7b:dc	100010
5001:11:1::	48	4092	17	66:0e:94:ff:7b:dc	100010

Unnumbered BGP neighbor parameters can be modified using the command `vrouter-bgp-modify`. For example:

```
CLI (network-admin@Leaf1) > vrouter-bgp-modify vrouter-name vr3 l3-port 19
multi-protocol ipv6-unicast
```

An L3 port can be removed from a BGP configuration using the command `vrouter-bgp-remove`. For example:

```
CLI (network-admin@Leaf1) > vrouter-bgp-remove vrouter-name vr3 l3-port 17
```

An L3 interface can also be removed from a vrouter using the command `vrouter-interface-remove`. For example:

```
CLI (network-admin@Leaf1) > vrouter-interface-remove vrouter-name vr3 l3-
port 20
```

Information regarding BGP neighbors can be reset using the command `vrouter-bgp-neighbor-reset`. For example:

```
CLI (network-admin@Leaf1) > vrouter-bgp-neighbor-reset vrouter-name vr3 l3-
port port 11
```

A BGP neighbor can be shut down using the command `vrouter-bgp-neighbor-shutdown`. For example:

```
CLI (network-admin@Leaf1) > vrouter-bgp-neighbor-shutdown vrouter-name vr3
l3-port port 26
```

A BGP neighbor can be brought back up after the shut down using the command `vrouter-bgp-neighbor-no-shutdown`. For example:

```
CLI (network-admin@Leaf1) > vrouter-bgp-neighbor-no-shutdown vrouter-name
vr3 l3-port 10
```



## Configuring Prefix Lists for BGP and OSPF

---

Prefix lists allow you to permit or deny host IP addresses from route distribution in BGP and OSPF configurations. To configure prefix lists for BGP, this example assumes that you have a vRouter configured for BGP, **vrouter-bgp**, and you want to deny the IP address, 172.26.0.0 with the netmask 255.255.0.0, sequence number 5, and minimum prefix length 17 bits:

```
CLI (network-admin@Leaf1) > vrouter-prefix-list-add vrouter-name vrouter-  
bgp name deny-bits action deny prefix 172.26.0.0 netmask 255.255.0.0 seq 5  
min-prefix-len 17
```

This prefix list rejects any subnets of 172.26.0.0/16 with prefixes 17 bits or longer. For example, the subnets 172.26.16.9/30 and 172.26.101.0/24 are rejected from route distribution.

The sequence number allows you to insert or remove new lines in a prefix list as well as at the beginning or end. It is recommended that you increment the sequence numbers by 10 so you can easily add or subtract lists from the configuration. See also:

- [Configuring Open Shortest Path First \(OSPF\)](#)
- [Configuring BGP on a vRouter](#)

## Configuring Bidirectional Forwarding Detection for IPv4 and IPv6

---

This feature adds bidirectional forwarding detection for IPv4 and IPv6 BGP neighbor and provides support for IPv6 BGP NBR reach-ability detection by using BFD protocol. When a BFD session goes from UP to DOWN, BFD informs BGP to bring the neighbor (NBR) down, until BFD returns to an UP state.

You create the BFD session by adding the `bfd` parameter to the vRouter configuration using the `bfd` parameter for the command, `vrouter-bgp-modify`. IPv6 BFD sessions for BGP NBRs are hosted in Netvisor One. The BFD session is started when you add the `bfd` parameter the BGP vRouter configuration.

```
CLI (network-admin@Leaf1) > vrouter-bgp-neighbor-show layout vertical
```

```
vrouter-name:          vr10
neighbor:              2002:100::2
ver:                  4
remote-as:             51000
msg_rcvd:              189
msg_sent:              193
up/down:
state/pfxrcd:          00:00:49 0
multi-protocol:        ipv6-unicast
```

Netvisor One supports IPv6 static route reach-ability detection using BFD protocol. Add IPv6 BFD session by specifying two end point IPv6 addresses, a source IPv6 address and a destination IPv6 address. The source IPv6 address must be a known local IPv 6 address. When a BFD session is up, Netvisor One assesses all the IPv6 static routes configured with a gateway or a BFD destination IPv6 address matching the destination IPv6 address of the BFD session. When a match is found, this static route is installed in Routing Information Database (RIB) and Forwarding Information Database (FIB).

```
CLI (network-admin@Leaf1) > vrouter-static-bfd-show layout vertical
```

```
vrouter-name:          vr10
src-ip:                2006:100::2
dst-ip:                2002:100::2
type:                  multi-hop
```

## Configuring BFD for OSPF Fault Detection

Bidirectional Forwarding Detection (BFD) can be used for OSPF fault detection. This feature provides fast failure detection when there is an intermediate device between two non-adjacent OSPF neighbors. That is, BFD provides fast failure detection between two nodes and notifies any protocol (OSPF) of this event to make it converge much faster than with default timers in protocols. Since OSPF hello timers may not be fast enough for detecting neighbor loss, you can use BFD to establish a BFD connection with the OSPF neighbor and bring an OSPF neighbor down as soon as BFD detects an issue.

Note that BFD is used for down detection only. This means that regular OSPF neighbor discovery and state machine transitions are not affected by enabling BFD.

You can enable BFD on all OSPF interfaces at the global level or on a specific interface. By default, BFD is disabled globally and on all interfaces. Individual interface configuration takes precedence over the global configuration.

**Note:** BFD is not supported for OSPFv6.

Setting the vrouter global OSPF level:

To enable OSPF BFD per vRouter on global OSPF level:

```
CLI (network-admin@Leaf1) > vrouter-modify name vrouter-name
```

vrouter-modify	Modify a vrouter
One or more of the following options	
ospf-bfd-all-if   no-ospf-bfd-all-if	Enables or disables BFD protocol for fault detection on all OSPF interface. Default is disabled

Setting the Interface (nic) OSPF level:

To enable OSPF BFD per interface :

```
CLI (network-admin@Leaf1) > vrouter-interface-config-add vrouter-name nic
```

vrouter-interface-config-add	Add an interface configuration to a vRouter VNIC name.
One or more of the following options	
nic vrouter interface nic	Specify the name of the VNIC.
[ospf-bfd] default   enable   disable	Enable BFD protocol support for OSPF fault detection

To modify OSPF BFD per interface:

```
CLI (network-admin@Leaf1) > vrouter-interface-config-modify vrouter-name
name nic nic ospf-bfd enable|disable|default
```

<code>vrouter-interface</code>	Modify an interface configuration to a vRouter
One or more of the following options:	Specify the name of the VNIC.
<code>ospf-bfd enable disable default</code>	Enables or disables the BFD protocol for fault detection on all OSPF interface. Default is disabled.

### Displaying the OSPF BFD Configuration State

To display the configuration state of OSPF BFD, use these show commands:

```
CLI (network-admin@Leaf1) > vrouter-show
```

```
CLI (network-admin@Leaf1) > vrouter-interface-show
```

**Note:** There are no changes to the commands: `vrouter-ospf-neighbor-show` and `vrouter-bfd-neighbor-show`

## Configuring Optimized BFD Path

To address BFD flaps, Netvisor ONE offers an optimized path for BFD control packets to reach a destination vRouter. BFD packets are relayed by the Netvisor ONE OS from the switch to the destination vRouter using a PCIe link which connects one of the switch ports to the host CPU. The BFD fastpath functionality alleviates BFD timeout/delay issues when other high bandwidth traffic is coming to the host CPU.

The optimized path option can be enabled or disabled using the command:

```
CLI (network-admin@leaf1) > system-settings-modify
```

<code>system-settings-modify</code>	Use this command to modify system settings.
Specify one or more of the following options:	
<code>optimize-arps   no-optimize-arps</code>	Enable or disable ARP optimization.
<code>lldp   no-lldp</code>	Enable or disable LLDP.
<code>policy-based-routing   no-policy-based-routing</code>	Enable or disable Policy-Based Routing (PBR). This enables flexible packet forwarding and routing through user-defined policies.
<code>optimize-nd   no-optimize-nd</code>	Enable or disable ND optimization.
<code>reactivate-mac   no-reactivate-mac</code>	Enable or disable reactivation of aged out MAC entries.
<code>reactivate-vxlan-tunnel-mac   no-reactivate-vxlan-tunnel-mac</code>	Enable or disable reactivation of MAC entries over VXLAN tunnels.
<code>manage-unknown-unicast   no-manage-unknown-unicast</code>	Enable or disable unknown unicast management.
<code>manage-broadcast   no-manage-broadcast</code>	Enable or disable broadcast management
<code>block-loops   no-block-loops</code>	Enable or disable loop blocking.
<code>auto-trunk   no-auto-trunk</code>	Enable or disable auto trunking.
<code>auto-host-bundle   no-auto-host-bundle</code>	Enable or disable auto host bundling.
<code>cluster-active-active-routing   no-cluster-active-active-routing</code>	Enable or disable active-active routing on a cluster.
<code>fast-route-download   no-fast-route-download</code>	Enable or disable fast route download from routesnoop.
<code>fast-interface-lookup   no-fast-interface-lookup</code>	Enable or disable fast router interface lookup.

<code>routing-over-vlags   no-routing-over-vlags</code>	Enable or disable routing to vLAGs from cluster links.
<code>source-mac-miss to-cpu   copy-to-cpu</code>	Specify either of the options as the unknown source MAC learn behavior.
<code>use-igmp-snoop-l2   use-igmp-snoop-l3</code>	Specify whether L2 or L3 tables are to be used for IGMP snooping.
<code>vle-tracking-timeout &lt;3..30&gt;</code>	Set a VLE tracking timeout as a value between 3 and 30s. The default timeout is 3s.
<code>pfc-buffer-limit pfc-buffer-limit-string</code>	Specify the percent of global system buffer space allowed for PFC.
<code>cosq-weight-auto   no-cosq-weight-auto</code>	Specify either of the options to enable or disable automatic weight assignment for CoS (Class of Service) queues based on min-guarantee configuration.
<code>lossless-mode   no-lossless-mode</code>	Enable or disable lossless mode.
<code>stagger-queries   no-stagger-queries</code>	Stagger igmp/mld snooping queries.
<code>host-refresh   no-host-refresh</code>	Enable or disable refreshing host ARP entries to keep L2 entries active.
<code>proxy-conn-retry   no-proxy-conn-retry</code>	Enable or disable proxy connection retry.
<code>proxy-conn-max-retry 0..10</code>	Set the maximum number of proxy connection retry attempts as a value between 0 and 10.
<code>proxy-conn-retry-interval 100..2000</code>	Set the number of milliseconds to wait between proxy connection retry attempts. This is a value between 100 and 2000.
<code>nvos-debug-logging   no-nvos-debug-logging</code>	Logging mode selection (Direct OR viz. nvlog demon)
<code>manage-l2-uuc-drop   no-manage-l2-uuc-drop</code>	Enable or disable L2 UUC (Unknown Unicast Drop) towards data port.
<code>xcvr-link-debug   no-xcvr-link-debug</code>	Enable or disable system debug to capture link information.
<b><code>fastpath-bfd   no-fastpath-bfd</code></b>	<b>Enable or disable BFD fastpath.</b>

For example, the command below enables BFD fastpath:

```
CLI (network-admin@leaf1) > system-settings-modify fastpath-bfd
```

This command creates a new vFlow which redirects all BFD packets to CPU through the PCIe link.

To see the status of BFD fastpath on a switch, use the command `system-settings-show`. For example,

the BFD fastpath status can be viewed exclusively using the command:

```
CLI (network-admin@leaf1) > system-settings-show format fastpath-bfd
```

```
fastpath-bfd: off
```

Netvisor ONE has the transmission queue `tx-class-inband-bfd` for sending out BFD packets. For reception, the `rx-bfd` queue is used.

To view the details of the transmission queue, use the command:

```
CLI (network-admin@leaf1) > nv-queue-stats-show name tx-class-inband-bfd
```

To view the receiver queue details, use the command:

```
CLI (network-admin@leaf1) > nv-queue-stats-show name rx-bfd
```

## Configuring Policy-Based Routing

Policy-Based Routing (PBR) enables flexible packet forwarding and routing through user defined policies. Unlike traditional routing based on destination IP address only, PBR allows you to define flexible routing policies based on other parameters such as source and destination IP addresses, IP protocol type, or source and destination L4 port numbers.

PBR policies are implemented with vFlow entries, which Netvisor ONE allocates in a dedicated (hardware) vFlow table, called *System-L3-L4-PBR*.

In addition, the PBR policy configuration process leverages the vFlow command syntax as explained later in this section (refer also to the *Configuring and Using vFlows* section for further details on the feature).

PBR routing policies are higher priority than static and dynamic routes. They can match packets based on all Layer 4 and Layer 3 packet fields, as supported by the vFlow configuration syntax.

Note that, if a PBR policy clause is matched but the next-hop is not resolved, the matching traffic is dropped until the next-hop gets resolved.

To enable PBR, use the following command:

```
CLI (network-admin@switch) > system-settings-modify policy-based-routing
```

Note: nvOSd must be restarted for this setting to take effect

To disable PBR, use the following command:

```
CLI (network-admin@switch) > system-settings-modify no-policy-based-routing
```

**Note: nvOSd must be restarted for this setting to take effect**

Use the following vflow command to configure a PBR policy. For details on configuring vFlows, see the *Configuring and Using vFlows* chapter.

```
CLI (network-admin@switch) > vflow-create name <policy-name> vrouter-name  
<vr-name> scope local [<match qualifiers>] action to-next-hop-ip action-to-  
next-hop-ip-value <ip-address> table-name System-L3-L4-PBR-1-0
```

**Note:** You can only specify the scope as `local`.

Use the following command to modify the PBR policy:

```
CLI (network-admin@switch) > vflow-modify name <policy-name> vrouter-name  
<vr-name> [<match qualifiers>] action to-next-hop-ip action-to-next-hop-ip-  
value <ip-address>
```



Use the following command to delete the policy:

```
CLI (network-admin@switch) > vflow-delete name <policy-name>
```

Below is an example of PBR policy creation:

```
CLI (network-admin@switch) > vflow-create name test_pbr scope local in-port
10 src-ip 192.168.1.1 src-ip-mask 255.255.255.0 vrouter-name vr1 action to-
next-hop-ip action-to-next-hop-ip-value 192.168.10.10
```

To view the configure policy, use the following command:

```
CLI (network-admin@switch)> vflow-show
```

```
switch:    spine1
name:      test_pbr
scope:     local
type:      pbr
in-port:   10
src-ip:    192.168.1.1/255.255.255.0
burst-size: auto
vrouter-name: vr1
precedence: default
action:    to-next-hop-ip
action-to-next-hop-ip-value: 192.168.10.10
enable:    enable
table-name: System-L3-L4-PBR-1-0
```

To modify this policy, the *vrouter name* and *action to-next-hop-ip* parameters are required in `vflow-modify` command to identify it is a *PBR vFlow entry* that is getting modified. For example, this command modifies the in-port value:

```
CLI (network-admin@switch) > vflow-modify name test_pbr in-port 20 vrouter-
name vr1 action to-next-hop-ip action-to-next-hop-ip-value 192.168.10.10
```

To display the vFlow table's usage and a specific PBR policy, use the following command sequence:

```
CLI (network-admin@switch) > vflow-table-show layout vertical
```

```
name:      Egress-Table-1-0
flow-max-per-group: 512
flow-used:  0
flow-tbl-slices: 2
capability: match-metadata
flow-profile: system
name:      System-L1-L4-Tun-1-0
flow-max-per-group: 2048
flow-used:  54
flow-tbl-slices: 2
capability: set-metadata
```

```
flow-profile: system
name:        System-VCAP-table-1-0
flow-max-per-group: 512
flow-used:    0
flow-tbl-slices: 1
capability:   none
flow-profile: system
name:        System-L3-L4-PBR-1-0
flow-max-per-group:
flow-used:
flow-tbl-slices:
capability:   set-metadata
flow-profile: system
```

```
CLI (network-admin@switch) > vflow-show name pbr_test layout vertical
```

```
name:        pbr_test
scope:       local
type:        pbr
src-ip:      10.10.10.1/255.255.255.0
burst-size:  auto
vrouter-name: vr1
precedence:  default
action:      to-next-hop-ip
action-to-next-hop-ip-value: 30.30.30.1
enable:      enable
table-name:   System-L3-L4-PBR-1-0
```

## Sending Network Traffic to an ECMP Group with PBR

When it is required to specify multiple next hops for redundancy purposes in Policy-Based Routing policies, it is possible to use *static ECMP groups*. They can be created with the `static-ecmp-group-create` command and then used in a vFlow PBR configuration to identify all the next hops.

You can add up to 16 next hops (NH) to an ECMP group.

Static ECMP groups can be defined with any of the three scopes: `local`, `cluster` or `fabric`. They can become active only if they are associated with a vRouter in the configuration. In other words, only if a static ECMP group is associated with a vRouter, Netvisor ONE creates an ECMP group entry in the hardware.

A static ECMP group can be associated with a vFlow PBR policy by using the action `to-ecmp-group` and the group's name as the action value for `action-to-ecmp-group-value`. For example:

```
CLI (network-admin@switch) > vflow-create name PBR_ECMP scope local src-ip
3.3.3.0/24 vlan 300 action to-ecmp-group action-to-ecmp-group-value
group_name vrouter-name vr-s2 table-name System-L3-L4-PBR-1-0
vflow-create: ecmp group group_name not created in hw
```

In the above case the vRouter did not exist hence the group was not programmed in hardware.

In addition, only if a Layer 3 entry is resolved and therefore is active as a given next hop, the associated egress ID is added to the ECMP group. Then, if a vFlow policy using the ECMP group is matched by some traffic, the hardware hashes (i.e., distributes) the traffic over the corresponding active next hops based on the Layer 3 and Layer 4 fields in the packets.

You can use the following command to create a static ECMP group associated to a vRouter:

```
CLI (network-admin@switch) > static-ecmp-group-create
```

<code>group-name</code> <i>group-name-string</i>	Specify an ECMP group name.
<code>scope</code> <code>local</code>   <code>cluster</code>   <code>fabric</code>	Specify the scope of the group.
<code>vrouter-name</code> <i>vrouter-name</i>	Specify the vRouter name.
<code>hash-type</code> <code>non-resilient</code>   <code>resilient</code>	Specify the ECMP hash type.

You can use the following command to delete a static ECMP group:

```
CLI (network-admin@switch) > static-ecmp-group-delete group-name group-name-string
```

Informational note: You cannot delete a static ECMP group while it is in use by any vFlow configuration.

You can use the following command to modify a static ECMP group:

```
CLI (network-admin@switch) > static-ecmp-group-modify group-name <group-name-string> vrouter-name <vrouter name> hash-type non-resilient/resilient
```

To display a static ECMP group's information you can use the command:

```
CLI (network-admin@switch) > static-ecmp-group-show
```

group-name group-name-string	Displays an ECMP group name.
scope local cluster fabric	Displays the scope of the group.
vrouter-name vrouter-name	Displays the vRouter name.
vrid vrid-number	Displays the vRouter ID.
hw-ecmp-id hw-ecmp-id-number	Displays the hardware ID.
hash-type non-resilient resilient	Displays the ecmp hash type.

```
CLI (network-admin@switch) > static-ecmp-group-show
```

```
group-name scope vrouter-name vrid hw-ecmp-id
-----
gr1          local              -1    -1
```

In the above example a vRouter is missing, hence the ECMP group is not active.

To add or remove a next hop to an ECMP group you can use:

```
CLI (network-admin@switch) > static-ecmp-group-nh-add
```

group-name group-name-string	Specify the name of the ECMP group.
ip ip-address	Specify the IP address for the next hop.

```
CLI (network-admin@switch) > static-ecmp-group-nh-remove
```

group-name group-name-string	Specify the name of the ECMP group.
ip ip-address	Specify the IP address for the next hop.

To show the next hop information you can use:

```
CLI (network-admin@switch) > static-ecmp-group-nh-show
```

group-name group-name-string	Displays the name of the ECMP group.
ip ip-address	Displays the IP address for the next hop.
vlan vlan-id	Displays the VLAN of the next hop.

egress-id egress-id-number	Displays the hardware egress ID.
----------------------------	----------------------------------

By default ECMP groups use a fixed hashing algorithm to distribute the traffic across multiple next hops. The advantage of this choice is that such algorithm is simple to implement in hardware and hence is widely available on all switch models.

However, when a link associated with a next hop goes down, the traffic is automatically re-distributed to adapt to the change in the number of paths: this action requires a complete remapping of the hash values thus resulting in unnecessary traffic disruption for certain flows.

Therefore, starting from Netvisor ONE release 5.1.1, on certain models only, a new more flexible hashing algorithm is supported. It is called *resilient hashing*, because it helps prevent unnecessary traffic disruption when the number of next hops changes.

The hash type can be specified as a parameter when a static ECMP group is created like so:

```
CLI (network-admin@switch) > static-ecmp-group-create group-name <name>
[hash-type non-resilient|resilient]
```

The default hash type is non-resilient. For example, two groups with two different hash types can be created with the following commands:

```
CLI (network-admin@switch) > static-ecmp-group-create group-name gr1 scope
fabric
```

```
CLI (network-admin@switch) > static-ecmp-group-nh-add group-name gr1 ip
2.2.2.2
```

```
CLI (network-admin@switch) > static-ecmp-group-create group-name gr2 scope
fabric hash-type resilient
```

```
CLI (network-admin@switch) > static-ecmp-group-nh-add group-name gr2 ip
3.3.3.3
```

```
CLI (network-admin@switch) > static-ecmp-group-show
```

group-name	scope	vrouter-name	vrid	hw-ecmp-id	hash-type
gr1	fabric	vr1	1	200001	<b>non-resilient</b>
gr2	fabric	vr1	1	200000	<b>resilient</b>

Informational note: Resilient hashing is not supported in the following switch models:

- Dell Z9100, Freedom F9532-C
- Dell S5048, Freedom F9572L-V

## Configuring Multicast Listener Discovery (MLD)

---

In IPv4, Layer 2 switches can use IGMP snooping to limit the flooding of multicast traffic by dynamically configuring Layer 2 interfaces so that multicast traffic is forwarded to only those interfaces associated with IP multicast address. In IPv6, MLD snooping performs a similar function. With MLD snooping, IPv6 multicast data is selectively forwarded to a list of ports that want to receive the data, instead of being flooded to all ports in a VLAN. This list is constructed by snooping IPv6 multicast control packets.

MLD is a protocol used by IPv6 multicast routers to discover the presence of multicast listeners (nodes configured to receive IPv6 multicast packets) on its directly attached links and to discover which multicast packets are of interest to neighboring nodes. MLD is derived from the IGMP protocol. MLD version 1 (MLDv1) is equivalent to IGMPv2, and MLD version 2 (MLDv2) is equivalent to IGMPv3. MLD is a sub-protocol of Internet Control Message Protocol version 6 (ICMPv6), and MLD messages are a subset of ICMPv6 messages, identified in IPv6 packets by a preceding Next Header value of 58.

The switch can snoop on both MLDv1 and MLDv2 protocol packets and bridge IPv6 multicast data based on destination IPv6 multicast MAC addresses. The switch can be configured to perform MLD snooping and IGMP snooping simultaneously.

To display MLD routers on the network, use the `mld-router-show` command:

```
CLI (network-admin@switch) > mld-router-show
```

```
group-ip:
node-ip:
vlan:
port:
source:
node-type:
expires:
```

The show output displays the following information:

- Multicast group IP address in IPv6 format
- Host node IP address in IPv6 format
- Host VLAN ID
- Port number
- Multicast traffic source IP address in IPv6 format
- Node type as host or router
- Expires as the age-out time

To display MLD group membership on the network, use the `mld-show` command:

```
CLI (network-admin@switch) > mld-show
```

```
group-ip:  
node-ip:  
vlan:  
port:  
source:  
node-type:  
expires:
```

The show output displays the following information:

- Multicast group IP address in IPv6 format
- Host node IP address in IPv6 format
- Host VLAN ID
- Port number
- Multicast traffic source IP address in IPv6 format
- Node type as host or router
- Expires as the age-out time

To enable or disable MLD snooping or modify the scope, use the `mld-snooping-modify` command:

```
CLI (network-admin@switch) > mld-snooping-modify
```

To modify the scope from fabric to local, use the following syntax:

```
CLI (network-admin@switch) > mld-snooping-modify scope fabric
```

To display information about MLD snooping, use the `mld-snooping-show` command.

## Configuring an IGMP Querier IP Address

You can configure an IGMP querier IP address for a VLAN or as a global IGMP querier. The IGMP querier sends IGMP General Query messages on the network.

If you do not specify a querier IP address, then Netvisor ONE uses 0.0.0.0 as the default value. There can be an unique querier IP for each VLAN, or you can configure the same Querier IP address for all the VLANs participating in IGMP snooping. The Querier IP address should have a local scope and every switch should have a unique Querier IP address.

With a valid source IP address on IGMP Query packets, Netvisor ONE adds the VLAN receiving the Query to an IGMP Snoop switch list, and now reflects in the `igmp-switches-show` output and the IGMP queries sent to the peer Switch as well. This solicits a report from the hosts listening on the peer switch.

Use these Netvisor One commands to configure an IGMP querier IP address.

<code>igmp-querier-ip-modify</code>	Modify IGMP Querier IP configuration
<code>querier-ip ip-address</code>	Specify the Snooping Querier IP address
<code>vlangs-on-querier-ip vlan-list</code>	Specify the VLAN map.
<code>igmp-querier-ip-show</code>	Display IGMP Querier IP config
<code>querier-ip is</code>	Specify the Snooping Querier IP address
<code>vlangs-on-querier-ip vlan-list</code>	VLAN MAP



## Configuring Multicast Listener Discovery (MLD) Snooping per VLAN

MLD snooping provides support per individual VLANs. In addition to current global enable and disable support, you can do the following using the `mld-snooping-modify` command:

- Specify a range of VLANs on which MLDv1 and MLDv2 messages are used. These two VLAN ranges are mutually exclusive. When a MLDv1 query is received on a VLAN configured for MLDv2, the MLDv2 query is moved to a VLAN configured with MLDv1 if there is only one multi-cast router for this VLAN. If a MLDv2 message is later received for the same VLAN, existing entries are removed and the supported version is changed to MLDv2.
- Specify VLAN ranges for snooping link-local addresses and ND Solicited Node addresses. You must include these VLANs in one MLDv1 and MLDv2 VLAN list.

Use these command options for the `mld-snooping-modify` command:

```
CLI (network-admin@Spine1) > mld-snooping-modify
```

<code>mld-snooping-modify</code>	Enables or disables Multicast Listener Discovery (MLD) snooping
One or more of the following options:	
<code>scope local fabric</code>	Specify the MLDsnooping scope - fabric or local
<code>enable disable</code>	Enable or disable MLD snooping
<code>mldv1-vlans vlan-list</code>	Specify the VLANs to enable snooping and use MLDv1 protocol. Default: none
<code>mldv2-vlans vlan-list</code>	Specify the VLANs on which to enable snooping and use MLDv2 protocol. Default 1 - 4092
<code>snoop-linklocal-vlans vlan-list</code>	Allow snooping of link-local groups(ff02::/16) on these vlans. Default 1 - 4092
<code>snoop-nd-vlans vlan-list</code>	Allow snooping of ND SN Multicast addresses (ff02::1:ff/104) on these vlans. Default 1 - 4092

The `mld-snooping-modify show format all` command displays the following sample output:

```
CLI (network-admin@Spine1) > mld-snooping-show format all
```

<code>switch:</code>	Name of Switch
<code>enable:</code>	no
<code>mldv1-vlans:</code>	none

---

mldv2-vlans:	1 - 4092
snoop-linklocal-vlans:	1 - 4092
snoop-nd-vlans:	1 - 4092
nvOS-managed-vlans:	none
interop-vl-vlans:	none
vlans:	1 - 4092

---

## Creating MLD Static Sources and Static Groups

---

To determine how to forward multicast traffic, a switch with MLD snooping enabled maintains information about the following interfaces in its multicast forwarding table:

- **Multicast-router interfaces** — These interfaces lead toward multicast routers or MLD queriers.
- **Group-member interfaces** — These interfaces lead toward hosts that are members of multicast groups.

The switch learns about these interfaces by monitoring MLD traffic. If an interface receives MLD queries, the switch adds the interface to the multicast forwarding table as a multicast-router interface. If an interface receives membership reports for a multicast group, the switch adds the interface to the multicast forwarding table as a group-member interface.

Table entries for interfaces that the switch learns about are subject to aging. For example, if a learned multicast-router interface does not receive MLD queries within a certain interval, the switch removes the entry for that interface from the multicast forwarding table.

You can create MLD static sources using IPv6 addresses and then create static groups using the static sources.

To create an MLD static source for IPv6 address, `ff02::1:ff11:111` as the group, and IPv6 `2001:db8::2:1` as the source on VLAN 25, port 44-45, use the following syntax:

```
CLI network-admin@switch > mld-static-source-create source-ip 2001:db8::2:1  
group-ip ff02::1:ff11:111 vlan 25 ports 44-45
```

The parameter, `ports`, is an optional parameter. You can delete an MLD static source, but you cannot modify the parameters.

To display MLD static sources, use the `mld-static-source-show` command.

To create an MLD static group for IPv6 address, `ff02::1:ff11:1111`, on VLAN 25, ports 44-45, use the `mld-static-group-create` command:

```
CLI network-admin@switch > mld-static-group-create group-ip  
ff02::1:ff11:1111 vlan 25 ports 44-45
```

You can delete an MLD static group, but you cannot modify the parameters. To display MLD static groups, use the `mld-static-group-show` command.

## Displaying MLD Statistics for a VLAN

---

To display MLD statistics for a VLAN, use the `mld-stats-show` command:

```
CLI network-admin@switch > mld-stats-show
```

```
switch:  
vlan:  
v1-queries:  
v2-queries:  
v1-member-reports:  
v1-done-group:  
v2-member-reports:  
queries-sent:  
drops:  
ignored:
```

## Configuring and Administering the Pluribus Fabric

---

Pluribus Networks offers a unique and highly differentiated approach to software-defined networking (SDN), called Adaptive Cloud Fabric. The distributed architecture enables organizations to build scalable private and public clouds with improved ease of management, reliability, security and performance. Pluribus Networks innovative Netvisor® ONE software virtualizes open networking hardware and builds a holistic, standard-based distributed network, referred to as a fabric, which provides improved management, automation, telemetry and resiliency.

This chapter provides information for understanding and using the Pluribus Networks Netvisor ONE command line interface (CLI) on a Netvisor ONE switch to configure and administer the Pluribus Fabric.

- 
- [Understanding the Netvisor® ONE Adaptive Cloud Fabric™](#)
    - [Understanding Fabric Transactions](#)
    - [Understanding Fabric Status, vPorts and Keepalives](#)
    - [Understanding Different Fabric Deployment Models](#)
    - [Guidelines and Limitations](#)
  - [Creating an Initial Fabric](#)
    - [About the Default Configuration](#)
    - [Displaying Fabric Instances](#)
    - [Adding Switches to an Existing Fabric](#)
  - [Configuring the Fabric Over the Management Interface](#)
    - [Displaying Fabric Nodes](#)
  - [Configuring Layer 4 Ports for Fabric Communication](#)
  - [Configuring a Fabric Over a Layer 3 Network](#)
  - [Connecting Multiple Fabric Nodes over Layer 3 Fabric](#)
  - [Troubleshooting the Fabric](#)
    - [Displaying the Transaction History](#)
    - [Keeping Transactions in Sync with Auto-Recovery](#)
    - [Rolling Back and Rolling Forward Transactions](#)
    - [Rolling Back the Configuration of the Fabric](#)
    - [Cluster Transaction Divergence](#)
    - [About the Fabric Default Parameters](#)
  - [Supported Releases](#)
-

- 
- [Displaying Fabric Information and Statistics](#)
-

## Understanding the Netvisor® ONE Adaptive Cloud Fabric™

Pluribus Networks offers a unique and highly differentiated approach to software-defined networking (SDN), called *Adaptive Cloud Fabric*. The distributed architecture enables organizations to build scalable private and public clouds that enjoy improved ease of management, reliability, security and performance.

Pluribus Networks' innovative Netvisor® ONE software *virtualizes open networking hardware* to build a holistic, standard-based *distributed network*, referred to as '*a fabric*', which provides improved management, automation, telemetry and resiliency.

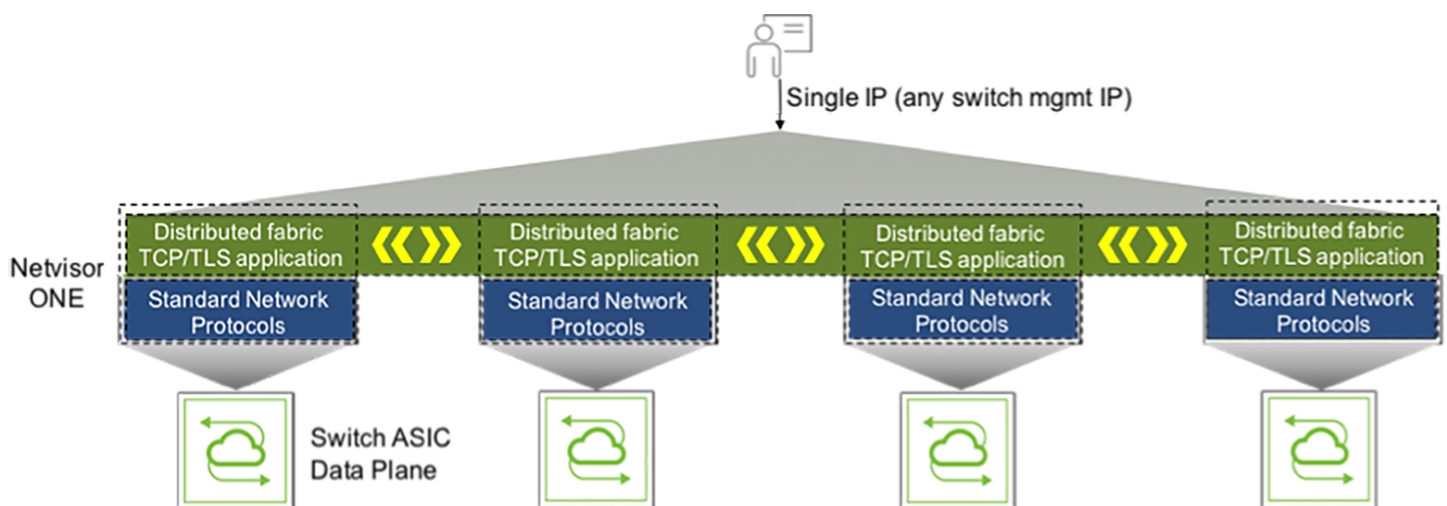
These enhancements are achieved thanks to the adoption of an advanced network virtualization paradigm.

In a Pluribus fabric every node behaves like a '*virtual module*' within a *logically-unified physically-distributed* '*virtual chassis*' which abstracts the *network topology*<sup>1</sup>. Each fabric node shares a common view of the network, including MAC and IP addresses, connections and application flows for management as well as redundancy purposes.

In addition, thanks to advanced embedded telemetry technology, the Netvisor ONE software provides *fabric-wide traffic visibility* to reveal network congestion issues and application performance bottlenecks so as to speed up troubleshooting, improve operational efficiency and strengthen security.

Pluribus Adaptive Cloud Fabric is a distributed application on top of a standard Layer 2/Layer 3 network that unifies the management plane for all the switches of the fabric (see figure below) and at the same time enhances the network control plane with the ability to automate and simplify several network functions.

**Note:** <sup>1</sup>The Adaptive Cloud Fabric supports any network topology, including ring, leaf-spine and multi-site.

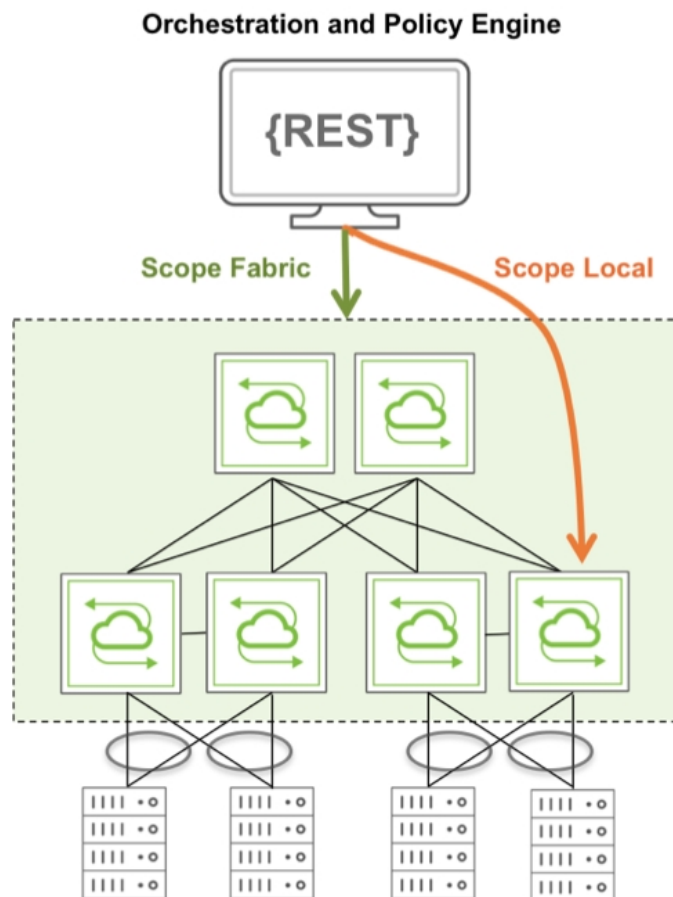


**Figure 6 - 1: Adaptive Cloud Fabric Simplified Management with Single IP to Configure and Monitor the Fabric**

This distributed control plane is *fully symmetrical* (or *peer-to-peer*) so that any node can act as a single point of management for the entire fabric. Therefore, *it does not require any controller entity* to guarantee the proper operation of the network and hence does not suffer from the intrinsic limits of a centralized control plane SDN architecture (such as *split brain* problems). At the same time, it is also able to inter-operate with a centralized orchestration system or controller through RESTful APIs or an OVSDB interface. This enables additional powerful management strategies when leveraging popular platforms such as Ansible or OpenDaylight.

Adaptive Cloud Fabric's advanced transactional model guarantees that device configuration is maintained consistent across fabric nodes and supports also rollback capabilities. Therefore, *a single point of provisioning* provides 'atomic' fabric-wide configuration with commands that can operate on a list of dispersed fabric devices (instead of simply on individual ones).

Since Adaptive Cloud Fabric does not inherently require special controllers to operate (i.e., uses a *controller-less, de-centralized* model), and since it leverages standard protocols (instead of proprietary technologies) it is fully interoperable with devices from other vendors.



**Figure 6 - 2: Network Orchestration with Adaptive Cloud Fabric**

Adaptive Cloud Fabric's open architecture offers users a far superior degree of interoperability compared to centralized SDN architectures, while at the same time providing support for powerful market-leading network-wide analytics.

While each node can belong to only one fabric, multiple fabric domains can be part of the same network so



as to implement more granular management domain segregation.

For example, in a typical leaf and spine data center configuration, it is often preferred to provision one management domain for all leaf switches and one separate domain for all spine switches, so as to preserve homogeneity as well as provisioning and monitoring simplicity within each functional domain.

The following picture (**Figure 6-3**) depicts an example of data center 'pod' topology comprising two management domains corresponding to two *separate fabric* instances: one for all the leaf switches and one for the two spine switches.



**Figure 6 - 3: Two Fabric Instances to Achieve Two Management Domains**

The Adaptive Cloud Fabric technology offers many additional benefits (described in more detail in the following chapters) that include more granular management segmentation for multi-tenancy, geographically dispersed data center interconnection (DCI), sophisticated security, etc.

## Understanding Fabric Transactions

---

The Adaptive Cloud Fabric uses *transactions* to synchronize configuration changes across the nodes of the fabric. Netvisor ONE records transactions as *atomic* operations that must either succeed and persist or fail and rollback, across the entire fabric. Transactions cannot be partially completed.

The fabric and Netvisor ONE adheres to the four standard transaction requirements: *atomicity, consistency, isolation and durability* (also collectively known as *ACID*).

Netvisor ONE does not require a fixed master node to coordinate all transactions across the fabric. Transactions start from the node where a command is run. This node is called the *originator node* or the *originator*, and coordinates the transactions with all the other fabric nodes.

Netvisor ONE commands originate from clients such as a CLI user, a RESTful API user or an external orchestration system. The commands are executed on a chosen switch, which becomes the originator node.

The originator first applies the configuration change specified by the command on the local node. If that fails, Netvisor rolls back any partial changes and then returns a failure message to the user. Only after the local change succeeds does the originator start the transaction. Netvisor ONE then atomically sends the configuration change such as create, delete, modify, add or remove commands to other fabric nodes.

Netvisor ONE transmits fabric transactions over a dedicated TCP socket, does not retain it and closes after each phase of the transaction. Transactions are encrypted using the TLS protocol.

All transactions are logged in a log file on a per scope basis in this location: `/var/nvOS/etc/<scope>`, where `<scope>` is `Local`, `Cluster`, or `Fabric`.

Scope defines the set of nodes participating in a transaction:

- Local — only the local node participates in the transaction.
- Cluster — only two redundant nodes participate in the transaction.
- Fabric — all nodes in the same fabric instance participate in the transaction.

For several commands, you can specify the scope of the intended action and therefore the scope of the ensuing transaction.

If a failure occurs on the fabric, transactions on certain nodes in the fabric can become out of sync. Once transactions become out of sync, no further transactions can be executed across the scope of local, fabric, or cluster.

You can verify the fabric node states with the command, `fabric-node-show`, and review the `fab-tid` values for matching values.

```
CLI (network-admin@switch) > fabric-node-show format name,fab-name,fab-  
tid,state,device-state,
```

name	fab-name	fab-tid	state	device-state
-----	-----	-----	-----	-----
pnswitch2	pnfabric	<b>2</b>	online	ok
pnswitch1	pnfabric	<b>2</b>	online	ok

The `state` column represents the communication status between members of the fabric and the `device-state` column represents the overall health of each switch. Also note that the fabric transaction ID (2 in this example) is consistent across all members of the fabric.

Mismatching `fab-tid` values for one or more nodes within the same fabric instance represent a corner case that the fabric control logic typically prevents. For more details, see the *Rolling Back and Rolling Forward Transactions* section for more details.

## Understanding Fabric Status Updates, vPorts and Keepalives

Instead of transactions, which are reserved for create, delete, modify, add or remove commands, Netvisor ONE uses status update messages to distribute information across the fabric.

Unlike transactions, Netvisor ONE does not strictly guarantee the status update data to be synchronized across the fabric.

Netvisor ONE sends the following states from the local node to the other nodes in the fabric:

- Node State
- Port State
- VLAN State
- Owned vPort State
- Layer 3 Entry State

Netvisor ONE uses vPort (or virtual port) as Layer 2 entries managed by the software and associated to ports where a node performs MAC address learning. vPorts contain information such as the MAC and IP address of a host, the VLAN and the connected port, the state, and other parameters.

In a fabric a node only sends status updates for vPorts that “is part of the fabric, which means that it sends updates only for the states of the hosts directly connected to that node. Netvisor ONE uses the term ‘owned vPorts’ to represent directly connected hosts.

You can display this information about directly connected hosts using the command, `vport-show`.

For example:

```
CLI (network-admin@switch) > vport-show
```

mac	vlan	ip	ports	state	migrate
00:00:4c:06:91:f8	514	10.81.114.21	61	active	9

In addition, Netvisor ONE sends status updates whenever a state changes on the fabric. For example, if a port goes down, or you create a new VLAN, the node with the port or VLAN sends a status update about the specific change.

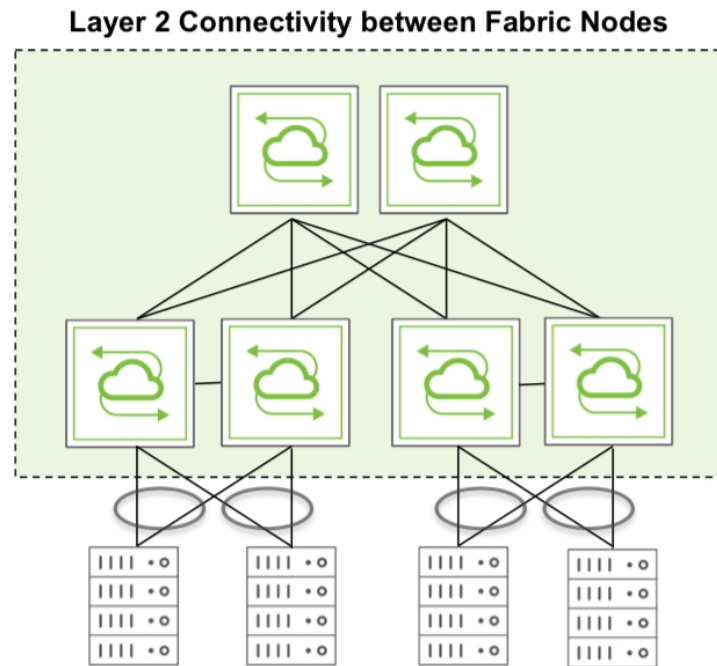
If sending the update message fails, Netvisor ONE attempts to resend it every 250ms.

With multiple nodes configured in the same fabric instance, switches send out special messages, called *fabric keepalives*, typically every 10 seconds to allow fabric peers to keep track of the online state of other nodes in the same fabric instance.

If a fabric keep-alive message is not received from a specific node for a period of time (30 s) equal to 3 times the transmission interval (10 s), the suddenly-gone-silent node's state is marked as `offline` by its neighbors. This inter-device connectivity/status check is performed with every other node within the same fabric instance.

## Understanding the Different Fabric Deployment Models

When you create a fabric instance, by default, Netvisor ONE uses Layer 2 communication over a configurable (user-selectable) VLAN for dedicated messages. The default behavior is the basic mode of operation and is referred to as *Fabric over L2* deployment model.



**Figure 6 - 4 Fabric over Layer 2 Deployment Model**

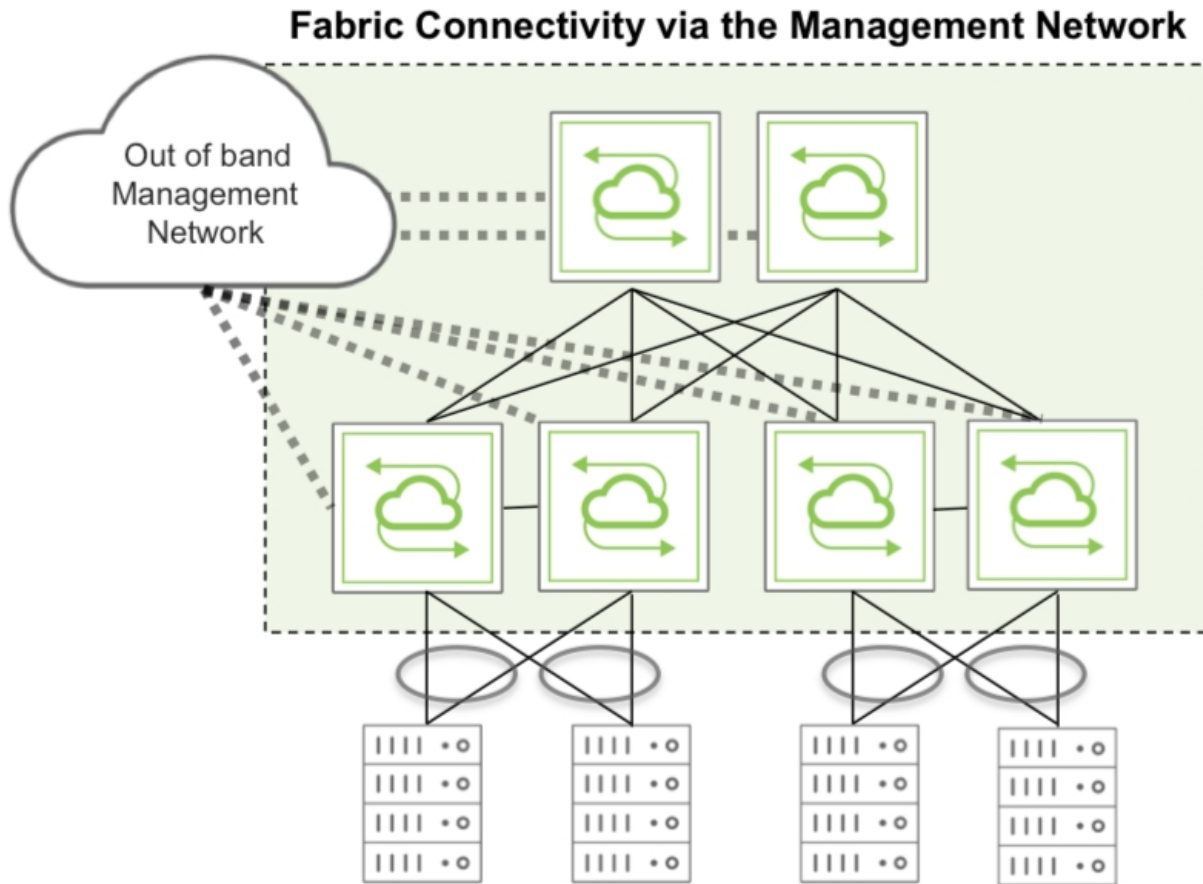
To perform fabric-related communication between nodes, Netvisor ONE uses different message groups called:

- fabric-network
- control-network

The fabric-network group consists of a number of messages such as transactions, fabric notifications, and proxy I/O messages such as API calls.

The control-network group consists of status updates, cluster synchronization messages, proxy I/O messages and forwarded packets. They may (but don't have to) be configured separately for improved message forwarding flexibility.

In certain network designs, a *dedicated management network* is deployed *in parallel* to the regular network, also referred to as *in-band* network, to interconnect nodes through management interfaces (see **Figure 6-5**).

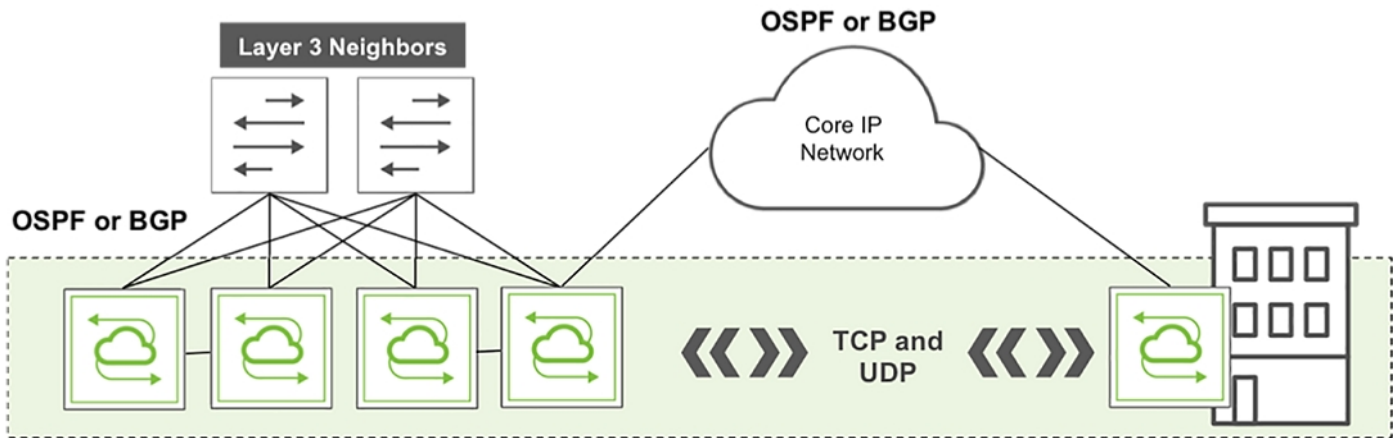


**Figure 6-5: Fabric Connectivity over the Management Network**

The *out-of-band* management network provides a robust alternate communication path between the nodes unaffected by user data traffic and user-related events.

In such designs a network administrator may leverage the dedicated management network for fabric communication. Therefore, Netvisor ONE allows the network administrator, using the `fabric-create` command, to select if a communication group uses the node management interface or the in-band interface. The fabric deployment model leveraging the management network for communication is called *Fabric over Management Interface*.

In addition to the two described network configurations, a third deployment model exists, *Fabric over L3*, and leverages a Layer 3 network running BGP or OSPF routing protocol to establish point-to-point fabric communication between nodes.



**Figure 6-6: Fabric over Layer 3 Deployment**

In the *Fabric over L3* deployment, multicast messages cannot be exchanged, as no Layer 2 adjacency exists between the nodes. Direct TCP or UDP communication must be established instead.

Therefore, you must first create the fabric on one node and then all the other nodes connected through Layer 3 can join the fabric one by one by specifying the IP address of the initial node using the command `fabric-join switch-ip`.

## Guidelines and Limitations

---

The following guidelines and limitations apply while configuring the fabric:

- Pluribus recommends, for security reasons, to change the VLAN ID used to create an in-band fabric over a Layer 2 or Layer 3 network as Netvisor ONE uses a default VLAN ID, VLAN 1.
- A fabric over the in-band network (Layer 2 or Layer 3) does not require a special MTU configuration.
- A fabric over management uses Layer 2 multicast, so be sure you do not filter multicast traffic on your management switches.
- Changing the IP address of a node joined to the fabric may cause the node to appear as offline for other fabric members. Therefore, before changing the IP address, unjoin the fabric, and then rejoin the fabric to avoid any inconsistencies.
- A Pluribus switch can only be part of one single fabric at a time.
- Netvisor ONE supports the following currently validated, tested number of fabric nodes in the same fabric:
  - 24 nodes for non-Pluribus switches
  - 32 nodes for Pluribus Networks switches
- When setting up a fabric, switches can only join the fabric one at a time. If you have a large fabric, you should consider this delay during the provisioning of the fabric.



## Creating an Initial Fabric

---

After completing the initial setup of a switch, you can create a new `fabric` instance to add the switch, or you can add the switch to an existing fabric.

When multiple switches join a fabric, the switches act as 'virtual modules' of a 'virtually distributed switch' with a single logical management plane. In this mode of operation, switches share status information and exchange commands based on the configured scope.

For example, any command with the scope, `fabric`, executes on each switch belonging to a shared fabric instance. This virtualized management paradigm significantly simplifies the network configuration and speeds up the deployment of complex networks.

A fabric instance can consist of just one individual switch, even though it is more common to have more than one switch, to ensure redundancy. For example, see [Figure 6- 3 Two Fabric Instances to Achieve Two Management Domains](#), where two redundant spine switches belong to a dedicated fabric instance.

Netvisor ONE continues to maintain the sharing of state and scope of a switch as long as the switch belongs to the fabric instance. When a switch leaves one fabric instance to join another one, the switch loses the synchronized fabric state and configuration of the first instance and learns the state and configuration of the second one.

To create a new fabric instance, use the following command:

```
CLI (network-admin@switch) > fabric-create name name-string
```

where *name-string* designates the name of the fabric.

Use the `password` option if you want to assign a password to the fabric instance creation process so that other switches are required to securely join the fabric only if the administrator knows the password.

```
CLI (network-admin@switch) > fabric-create name name-string  
password<return>
```

```
fabric password: hidden-password-string <return>  
confirm fabric password: hidden-password-string <return>
```

## About the Default Configuration

---

By default, Netvisor ONE creates a new fabric instance on VLAN 1.

However, VLAN 1 is often the default VLAN in most networks and therefore security best practices recommend using a non-default VLAN, whenever possible, for maximum robustness and error prevention.

To assign a non-default VLAN, for example, a VLAN ID between 2 and 4093, use the command:

```
CLI (network-admin@switch) > fabric-create name name-string vlan vlan-id
```

To change the VLAN ID of an existing fabric, use the command, `fabric-local-modify`.

Use the same command to change the fabric administration network, control plane network, and the network to send fabric advertisements.

```
CLI (network-admin@switch) > fabric-local-modify vlan vlan-id
```

**Note:** In order to change the VLAN number of an existing instance you must recreate it. Also, a switch can belong to only one fabric instance.

## Displaying Fabric Instances

---

To show all the fabric instances and their specific details, use the `fabric-show` command:

```
CLI network-admin@switch > fabric-show
```

name	id	vlan	fabric-network	control-network	tid	fabric-advertisement-network
Fabric1	b000707:59b6a9ef	0	mgmt	mgmt	48	inband-mgmt
Fabric2	90004eb:59b7da05	0	mgmt	mgmt	8	inband-mgmt

Netvisor ONE discovers all available fabric instances by sending out special multicast messages, called *global discoveries*, whenever a physical port becomes forwarding or on demand.

For example, when you execute a `fabric-show` command, Netvisor ONE sends discovery messages over in-band as well as over the management interface.

After receiving a global discovery message the receiving device responds with a *global keep-alive* message containing the required fabric and node information.

This local multicast-based discovery mechanism implies that direct Layer 2 connectivity exist between the discoverer and the polled switches.

## Adding Switches to an Existing Fabric

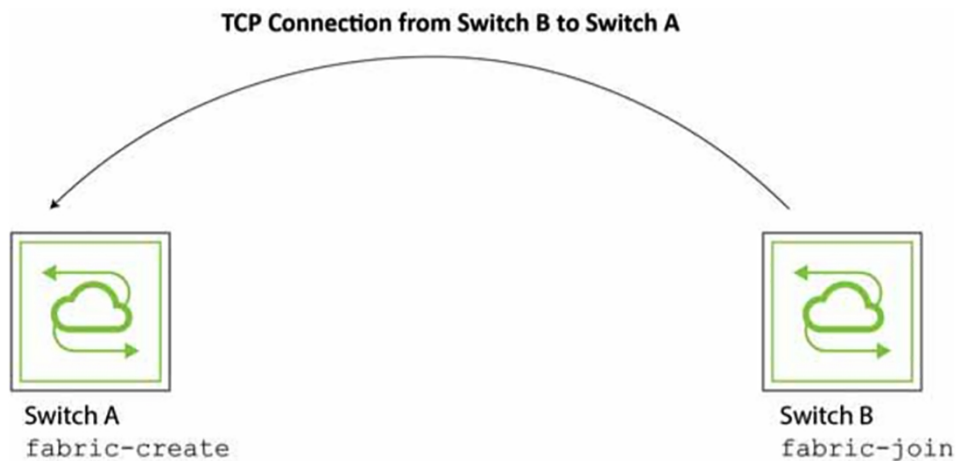
To add a switch to an available fabric instance, use the following command:

```
CLI (network-admin@switch) > fabric-join name name-string
```

For example:

```
CLI (network-admin@switch) > fabric-join name MyFabric
```

In case of directly connected switches as in **Figure 6-7**, switch B learns about available fabrics from switch A and then sends a message to join the selected fabric.



**Figure 6 -7 Directly Connected Switch Joining a Fabric**

Using the Tab key, Netvisor ONE displays all fabrics configured in the local network as options.

A switch joins the fabric either by using a discovered fabric name or by using a switch IP address. To join a remote fabric, use the `fabric-join` command with a remote switch IP address:

```
CLI (network-admin@switch) > fabric-join switch-ip 192.168.2.2  
Joined fabric MyFabric. Restarting nvOS...
```

In addition, the Netvisor ONE software can use the password set up during the fabric creation process to encrypt communication between the nodes in the fabric.

In such cases, when the switch joins a fabric instance from a node, you must type in the password to join it.

**Note:** Avoid creating fabrics with the same name to prevent conflicts.

Once a new switch joins an existing fabric, the new switch downloads all fabric configuration from the existing fabric switch and restarts the nvOSd (reboots).

After the switch is rebooted and is up and running, the new switch becomes part of the existing fabric.

## Configuring the Fabric Over the Management Interface

---

Fabric configuration information can be exchanged over the network through in-band communication.

However, occasional disruption to in-band traffic occurs due to factors such as network re-convergence, port flapping, power system transients, and other events.

Therefore, an alternative method is to configure fabric communication over the management interface for a dedicated communication channel.

This can be achieved while creating the fabric, for example:

```
CLI (network-admin@switch) > fabric-create name MyFabric fabric-network  
mgmt
```

When you create a fabric over the management interface, any other node joining the fabric inherits this setting. In other words, all nodes within the same fabric communicate through the same network type with fabric peers. You cannot have mixed fabric configurations using both management interfaces and in-band communication.

Therefore, Netvisor does not display fabrics over an incompatible network when you execute the `fabric-join` command. This prevents a switch from joining an incompatible fabric.

When you configure the fabric communication over the management interface, all fabric communication stays on the management network, except the following types of packets:

- Cluster synchronization-related messages and cluster keep-alive packets sent over the in-band interface.
- The fabric advertisements such as fabric keep-alive packets and global-discovery packets are controlled by `fabric-advertisement-network`, which is configured while creating or modifying a fabric.

While fabric-related communication such as transactions, notifications, file system replication messages, and other communications can be configured to be sent over the management network, for consistency, it is recommended to use the same management network for other purposes or communication types such as network status updates and forwarded packets, collectively referred to as control network and fabric advertisements.

The `fabric-create` command allows you to select the transmission medium for other traffic types using specific parameters named `control-network` and `fabric-advertisement-network`:

```
CLI (network-admin@switch) > fabric-create name MyFabric [fabric-network  
in-band|mgmt] [control-network in-band|mgmt] [fabric-advertisement-network  
inband-mgmt|inband-only|mgmt-only]
```

## Displaying Fabric Nodes

Netvisor ONE uses fabric keepalive packets to determine the state of each fabric node. To display the state, use the `fabric-node-show` command with the syntax:

```
CLI network-admin@switch > fabric-node-show [state offline| online|in-band-only-online|mgmt-only-online|fabric-joined|eula-required|setup-required|fabric-required|fresh-install]
```

Netvisor ONE supports monitoring and reporting on both management and in-band network, therefore the node state can be one of the following:

- `online` — reach-ability of node over both management and in-band interfaces
- `In-band-only-online` — reach-ability of node through in-band channel only
- `mgmt-only-online` — reach-ability of node through management network only
- `offline` — no reach-ability over either communication channel.

In this example, Netvisor ONE displays the `online` node state in the command output:

```
CLI (network-admin@switch) > fabric-node-show layout vertical
```

```
id:                167772208
name:              switch
fab-name:          MyFabric
fab-id:            a000030:5537b46c
cluster-id:        a000030:1
fab-mcast-ip:      ::
local-mac:         64:0e:94:28:00:8e
fabric-network:    in-band
mgmt-ip:           10.9.100.100/16
mgmt-mac:          64:0e:94:28:00:8f
mgmt-l3-port:      0
mgmt-secondary-macs:
in-band-ip:        192.168.42.10/24
in-band-mac:       64:0e:94:28:00:8e
in-band-l3-port:   0
in-band-secondary-macs:
fab-tid:           8
cluster-tid:       1
out-port:          0
version:           5.0.0-5000014540
state:             online
firmware-upgrade:  not-required
device-state:      ok
ports:            0
```

Also check the `fab-tid` value for consistency on each node. See the *Troubleshooting the Fabric* section for details.



## Displaying Fabric Information and Statistics

To display information on the configured fabrics, use the `fabric-show` command:

```
CLI (network-admin@switch) > fabric-show
```

name	id	vlan	fabric-network	control-network	tid
Fabric1	a000030:5537b46c	3	in-band	in-band	365
Fabric2	6000210:566621ee	100	mgmt	in-band	5055

To display the information about the fabric instance of the local switch, use the `fabric-info` command:

```
CLI (network-admin@switch) > fabric-info format all layout vertical
```

```
name:                Fabric1
id:                  a000030:5537b46c
vlan:                3
fabric-network:      in-band
control-network:     in-band
tid:                 365
fabric-advertisement-network: inband-only
```

To display fabric statistics use the `fabric-stats-show` command:

```
CLI (network-admin@switch) > fabric-stats-show
```

switch	id	server	storage	VM	vlan	vxlan	tcp-syn	tcp-est	tcp-completed	tcp-bytes	udp-bytes	arp
-----	--	-----	-----	--	----	-----	-----	-----	-----	-----	-----	----
pubdev02	0	0	0	0	0	0	14.0k	5	40	125K	0	0
pubdev03	0	0	0	0	0	0	3.85K	3	24	110M	0	0

To display fabric statistics in vertical format, use the following command:

```
CLI (network-admin@switch) > fabric-stats-show format all layout vertical
```

```
switch:              sw45
id:                  0
servers:             0
storage:             0
VM:                  0
vlan:                0
vxlan:               0
tcp-syn:             0
tcp-est:             0
tcp-completed:       0
tcp-bytes:           0
udp-bytes:           0
arp:                 0
```



## Configuring Layer 4 Ports for Fabric Communication

Netvisor ONE receives fabric communication messages by using system vFlow entries to match traffic flows based on the IP packet protocol, TCP or UDP, and Layer 4 destination port.

Any existing TCP or UDP traffic across the network using the same Layer 4 destination port may match the vFlows and may potentially cause conflicts.

**Table 6-1:** Default Port Values for Netvisor ONE

Message Type	Layer 4 Port
fabric	23399
notify	23398
proxy	23397
fabric-keepalive	23394
filesystem-replication	23392
cluster-traffic-forwarding	23391
vport-statistics	23390
l2-encap	23389
igmp-encap	23388
icmpv6-encap	23387
arp-encap	23386
cluster-analytics	23385

To avoid conflicts with generic TCP or UDP traffic, configure an alternate Layer 4 port range for special messages using the command:

```
CLI (network-admin@switch) > fabric-comm-ports-modify
```

You can specify the value for the starting point of the range:

```
range-start 1024..65435 port range start
```

To display the current starting point and range values, use the following command:

```
CLI (network-admin@switch) > fabric-comm-ports-show
```

```
switch:                                pnswitch2
range-start:                           23300
fabric-port:                           23399
notify-port:                           23398
proxy-port:                            23397
fabric-keepalive-port:                 23394
```

<code>filesystem-replication-port:</code>	23392
<code>cluster-traffic-forwarding-port:</code>	23391
<code>vport-statistics-port:</code>	23390
<code>l2-encap-port:</code>	23389
<code>igmp-encap-port:</code>	23388
<code>icmpv6-encap-port:</code>	23387
<code>arp-encap-port:</code>	23386
<code>cluster-analytics-port:</code>	23385

When you modify the port range, you must configure each node in the fabric individually.

This change temporarily interrupts fabric communication until you have completed the configuration of each node with the same port range.

There is no loss of switched traffic if the interruption is brief.

Because application of this command prevents communication with other nodes, you must log into each node directly and separately apply the `fabric-comm-ports-modify` command.

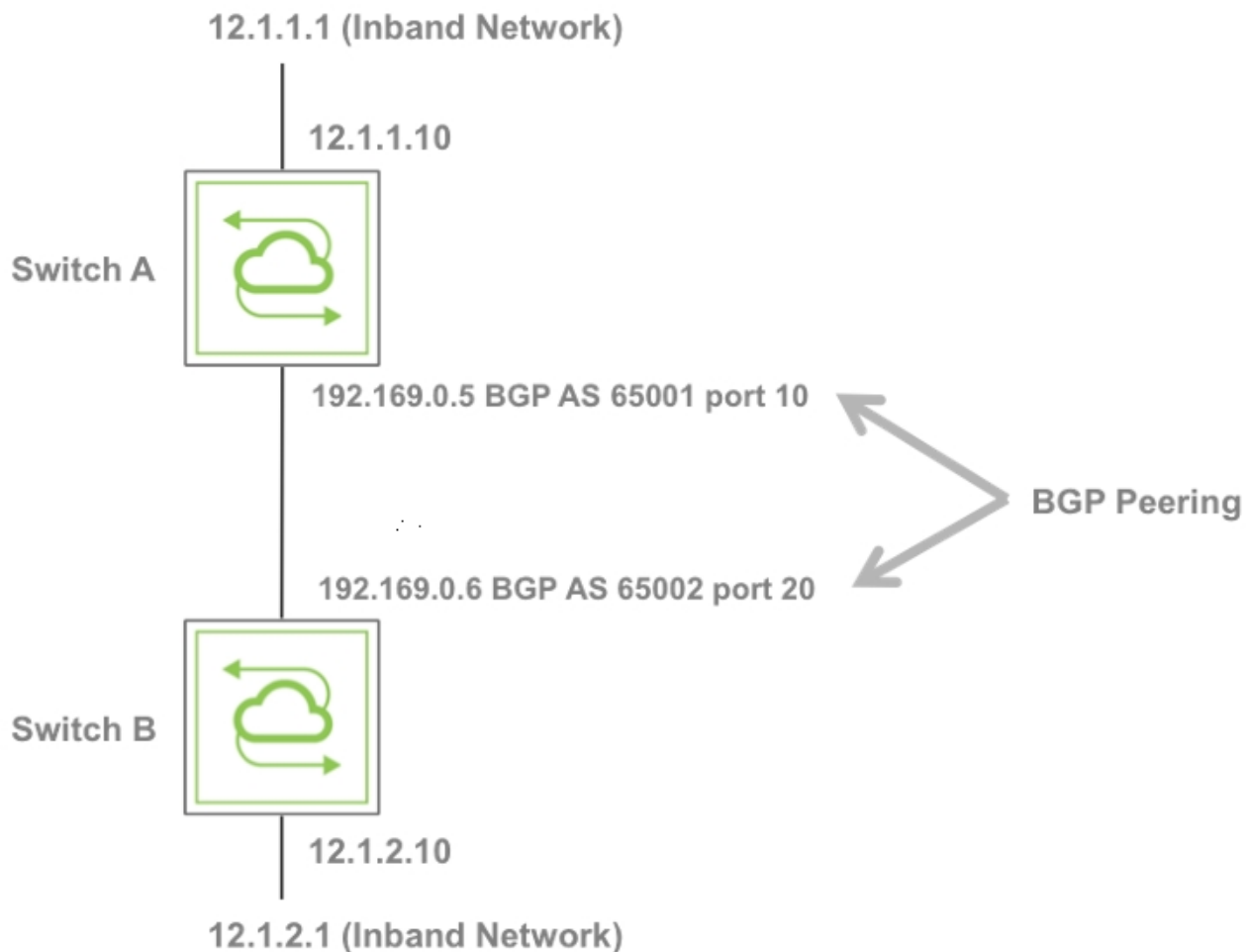
## Configuring a Fabric Over a Layer 3 Network

As described in previous sections, you can create a Netvisor ONE fabric over the management network or over in-band, i.e., between switches with direct (Layer 2) connectivity.

Another option is to establish Netvisor ONE fabric over a Layer 3 network. Netvisor ONE supports this capability over a Layer 3 network running either the *BGP* or the *OSPF routing protocol*.

Before you can configure this feature on your network, make sure you configure BGP or OSPF routing on your switches. For more information on Layer 3 configuration, refer to the *Configuring Layer 3 Features* section.

In a simple topology with a network of two BGP neighbors as in **Figure 6-8**. Netvisor ONE uses in-band communication to establish TCP sessions between the two switches to exchange fabric messages.



**Figure 6-8: Example Network of Two BGP Neighbors Topology**

To achieve the BGP-based configuration and establish fabric communication between the two switches, you must perform the following steps:

- 1) Modify the in-band IP in the same subnet of in-band-nic-ip address.
- 2) Create a vRouter on each switch.
- 3) Setup BGP peering and redistribute routes.
- 4) Setup up global static routes using the respective vRouter as the gateway. You must perform this step because the routes reside inside the vRouter table and each switch must have an appropriate route to reach the in-band network of another switch.
- 5) Add global static route on switch B to reach switch A in-band network with switch B in-band-nic-ip as the gateway IP address.
- 6) Then you can configure switch B to join the fabric previously created by switch A.

Use the following commands to accomplish the necessary configuration:

### Creating a vRouter for Fabric Communication

Use the `fabric-comm-vrouter-bgp-create` command to create a vRouter for fabric communication.

This command has numerous options including the following:

<code>name</code>	Name of the fabric communication vRouter.
<code>bgp-as</code>	BGP Autonomous System number of the vrouter from 1 to 4294967295.
<code>bgp-nic-vlan</code>	Interface VLAN number between 0 and 4095. You can enable BGP over VLAN interface also.
<code>bgp-redistribute</code>	BGP route redistribution. The default value is connected.
<code>bgp-max-paths</code>	Maximum BGP paths.
<code>bgp-ibgp-multipath</code>	Number of IBGP multipath connections.
<code>bgp-nic-ip</code>	IP address of the peering interface.
<code>bgp-nic-netmask netmask</code>	BGP NIC netmask.
<code>bgp-nic-linklocal ip-address</code>	IPv6 link local address.
<code>bgp-nic-vnet</code>	Interface VLAN vNET.
<code>bgp-nic-bd</code>	Interface bridge domain.
<code>bgp-nic-vlan-type</code>	Interface VLAN type.
<code>bgp-nic-l3-port</code>	Layer 3 port
<code>bgp-nic-mtu</code>	Interface MTU
<code>bgp-nic-if-nat-realm</code>	NAT interface realm
<code>in-band-nic-ip</code>	IP address range of the in-band management interface.
<code>remote-as</code>	BGP remote AS number from 1 to 4294967295

---

neighbor

IP address of the BGP neighbor.

---

The BGP protocol reserves Autonomous System (AS) numbers 64512–65534 for private use and the example below uses some of the private AS numbers.

This dedicated CLI performs the majority of the actions in the logical steps 1-4 described in the *Configuring a Fabric Over a Layer 3 Network* section above. However, it does not change the inband IP address or create the fabric.

After creating the fabric, the root switch, Switch A, does not have to specify the fabric network in the switch configuration, but the switch must create a route to the other inband network on the switch with the `fabric-in-band-network-create` command.

Switch B, a non-root switch, does need to specify the fabric network by using the `fabric-network` parameter:

See an example on how to establish fabric communication between two switches (switch A and switch B) detailed in steps 1 through 6 above.

### Switch A

```
CLI (network-admin@switch) > switch-setup-modify in-band-ip 12.1.1.1/24
```

```
CLI (network-admin@switch) > fabric-comm-vrouter-bgp-create name vrouter-a  
bgp-nic-l3-port 10 bgp-as 65001 bgpnic-ip 192.169.0.5/30 in-band-nic-ip  
12.1.1.10/24 remote-as 65002 neighbor 192.169.0.6 bgp-redistribute  
connected fabric-network 12.1.1.0/24
```

```
CLI (network-admin@switch) > fabric-in-band-network-create network  
12.1.2.1/24
```

### Switch B

```
CLI (network-admin@switch) > switch-setup-modify in-band-ip 12.1.2.1/24
```

```
CLI (network-admin@switch) > fabric-comm-vrouter-bgp-create name vrouter-b  
bgp-nic-l3-port 20 bgp-as 65002 bgpnic-ip 192.169.0.6/30 in-band-nic-ip  
12.1.2.10/24 remote-as 65001 neighbor 192.169.0.5 fabric-network  
12.1.1.0/24 bgp-redistribute connected fabric network 12.1.2.0/24
```

```
CLI (network-admin@switch) > switch-route-create network 12.1.1.0/24  
gateway-ip 12.1.2.10
```

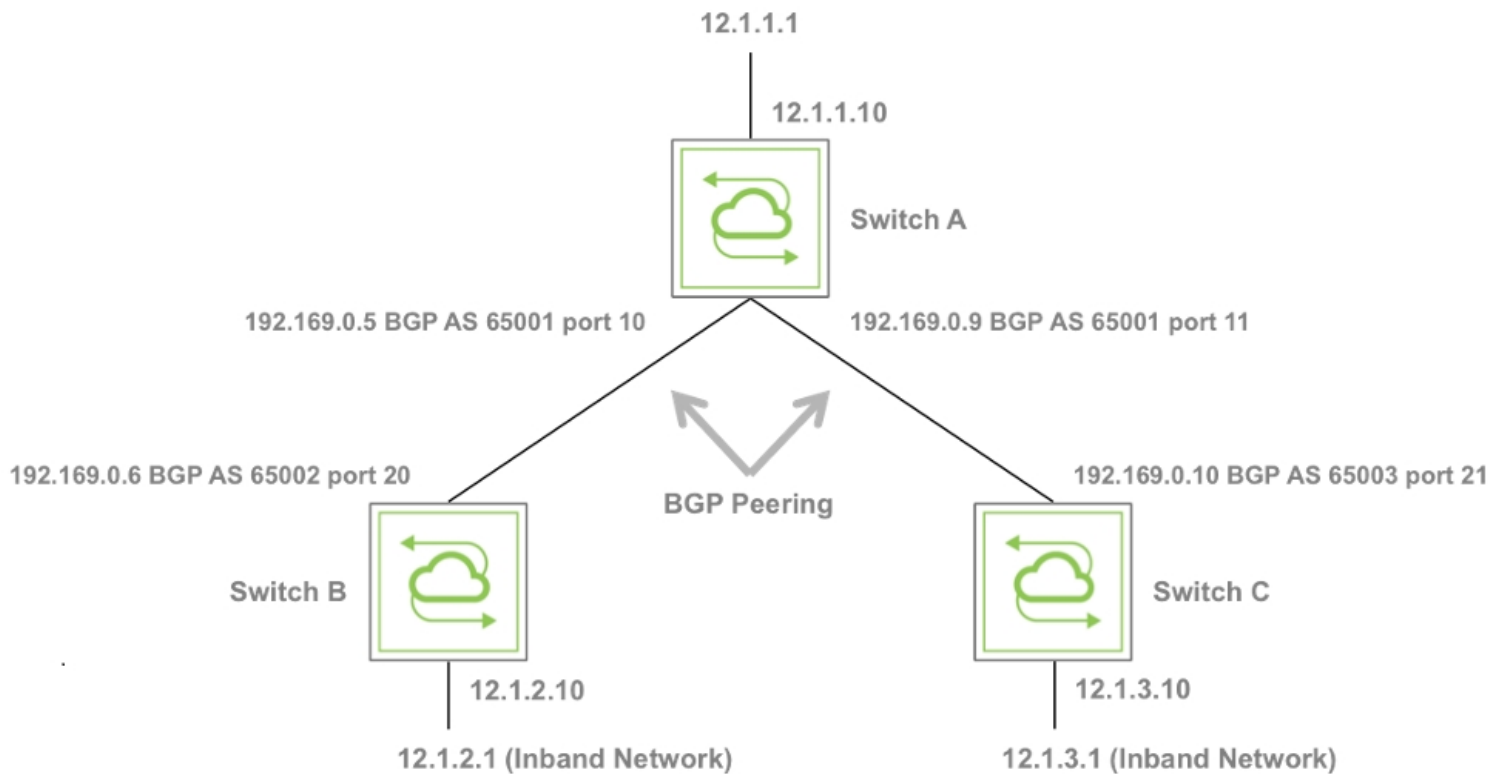
```
CLI (network-admin@switch) > fabric-join switch-ip 12.1.1.1
```

```
CLI (network-admin@switch) > Joined fabric myfabric. Restarting nvOS...
```

## Connecting Multiple Fabric Nodes over a Layer 3 Fabric

In a topology with more than two switches configured for BGP, the first peer for each switch can be created using the `fabric-comm-vrouter-bgp-create` command. You must create the other switches separately.

See **Figure 6-9** where a third switch, Switch C, is added to the previous two-switch topology.



**Figure 6-9: An Example of Fabric over Layer 3 Topology**

The configuration for this example follows the same scheme as the two switch topology. However, Netvisor ONE requires the following additional steps:

- 1) On Switch A, configure BGP peering with Switch C and create a static route to reach the inband network on Switch C. To extend this configuration with more than three-switch example, use this step on every switch with more than one BGP peer.

```
CLI (network-admin@switch) > vrouter-interface-add vrouter-name vrouter-a
ip 192.169.0.9/30 13-port 11
```

```
CLI (network-admin@switch) > vrouter-bgp-add vrouter-name vrouter-a
neighbor 192.169.0.10 remote-as 65003
```

```
CLI (network-admin@switch) > fabric-in-band-network-create network
12.1.3.0/30
```



2) On Switch C, use the `fabric-comm-vrouter-bgp-create` command and connect:

```
CLI (network-admin@switch) > switch-setup-modify in-band-ip 12.1.3.1/24
```

```
CLI (network-admin@switch) > fabric-comm-vrouter-bgp-create name vrouter-c  
bgp-nic-l3-port 21 bgp-as 65003 bgpnic-ip 192.169.0.10/30 in-band-nic-ip  
12.1.3.10/24 remote-as 65001neighbor 192.169.0.9 fabric-network 12.1.1.0/24  
bgp-redistribute connected fabric-network 12.1.3.0/24
```

```
CLI (network-admin@switch) > switch-route-create network 12.1.1.0/24  
gateway-ip 12.1.3.10
```

```
CLI (network-admin@switch) > switch-route-create network 12.1.2.0/24  
gateway-ip 12.1.3.10
```

```
CLI (network-admin@switch) > switch-route-create network 12.1.3.0/24  
gateway-ip 12.1.2.10
```

```
CLI (network-admin@switch) > fabric-join switch-ip 12.1.1.1
```

```
Joined fabric myfabric. Restarting nvOS...
```

## Troubleshooting the Fabric

---

- [Displaying the Transaction History](#)
- [Keeping Transactions in Sync with Auto-Recovery](#)
- [Rolling Back and Rolling Forward Transactions](#)
- [Rolling Back the Configuration of the Fabric](#)
- [How to Fix the Cluster Transaction Divergence Issue](#)
- [About the Fabric Default Parameters](#)

## Displaying the Transaction History

When configuring and administering the fabric, you can display the list of executed transactions by using the `transaction-show` command. For example in this case committed transaction 1 corresponds to a command in which a vRouter was created:

```
CLI (network-admin@switch) > transaction-show
```

```
start-time:    2020-05-14,15:38:34.622
tid:           1
state:         commit
command:       vrouter-create id b001b66:0 name vr1 scope fabric vnet test-
fab-2-global dedicated-vnet-service locati
on 1 storage-pool rpool zone-id b001b66:1 router-type hardware router-
ipstack frr hw-router-mac 66:0e:94:66:b3:0f clu
ster-active-active-routing hw-vrid 0 hw-vrrp-id -1 bgp-redis-static-metric
none bgp-redis-connected-metric none bgp
-redis-rip-metric none bgp-redis-ospf-metric none bgp-max-paths 1 bgp-
delayed-startup 1 bgp-update-delay-strict 0 b
gp-keepalive-interval 60 bgp-holdtime 180 bgp-global-nh-preference true
ospf-redis-static-metric none ospf-redis-st
atic-metric-type 2 ospf-redis-connected-metric none ospf-redis-connected-
metric-type 2 ospf-redis-rip-metric none
ospf-redis-rip-metric-type 2 ospf-redis-bgp-metric none ospf-redis-bgp-
metric-type 2 ospf-default-information none
  ospf-default-info-originate-metric none ospf-default-info-originate-
metric-type 2
undo-command:  vrouter-delete name vr1
-----
start-time:    2020-05-14,15:39:02.820
tid:           2
state:         commit
command:       vrouter-interface-add vrouter-name vr1 nic eth0.4092 ip
10.0.120.2 netmask 24 mac 66:0e:94:66:b3:0f vl
an 4092 vlan-type public public-vlan 4092 no-exclusive nic-enable no-pim
pim-dr-priority 1 no-pim-cluster no-fabric-n
ic vrrp-priority 100 vrrp-adv-int 1000 l3-port 45 mtu 1500 no-sriov-vf
mirror-traffic false no-priority-tag
undo-command:  vrouter-interface-remove vrouter-name vr1 nic eth0.4092
```

The Netvisor ONE transaction log file contains the list of commands corresponding to the executed transactions as well as the undo command(s) required to undo each transaction. Commands are shown in the `transaction-show` output for *informational purposes only*.

Each transaction can have a *local*, *cluster* or *fabric* scope. For example acceptance of the EULA is a transaction that is local to each device:

```
CLI (network-admin@switch) > transaction-show transaction-scope local
```

```

start-time:    2019-06-02,21:50:35.497
tid:           1
state:         commit
command:       switch-setup-modify eula-accepted true eula_timestamp 2019-
06-02,21:50:35
undo-command:  switch-setup-modify eula-accepted false eula_timestamp 1969-
12-31,16:00:00

```

On the other hand, for instance transactions of VLAN creation and deletion are usually (but not necessarily always) executed with a fabric scope. Other objects such as vNETs are created and deleted with a fabric scope too, as shown in the example below:

```

CLI (network-admin@switch) > transaction-show transaction-scope fabric

```

```

start-time:    2019-06-02,22:10:54.895
tid:           1
state:         remote-commit
command:       vlan-create id 100 type public vxlan-mode standard scope
fabric stats
undo-command:  vlan-delete id 100
-----

```

```

start-time:    2019-06-02,22:11:25.834
tid:           2
state:         remote-commit
command:       vnet-create id c00025b:1 name vn1 scope fabric vrg c00025b:0
vlan-type public vlans 5 config-admin create-vnet-mgr vnet-mgr-storage-pool
rpool vnet_mgr_id c00025b:0 vnet_mgr_zone_id c00025b:1 vnet_mgr_location 1
vrg_created_by_vnet true admin_role c00025b:400
undo-command:  vnet-delete name vn1
-----

```

```

start-time:    2019-06-02,22:11:36.907
tid:           3
state:         remote-commit
command:       vlan-delete id 100
undo-command:  vlan-create id 100 type public hw-vpn 0 hw-mcast-group 0
replicators "" repl-vtep :: in_hw false public-vlan 100 user_public_id 0
public_id_set_by_user false scope fabric description vlan-100 label "vlan
100" active yes stats uses_refcnt no refcnt 0 vrg 0:0 ports 1-69 untagged-
ports none send-ports 21,23,31-34,53,57,61,65,69,272-275 active-edge-ports
272 non-auto-ports none ports-specified false initialized true flags ""
-----

```

```

start-time:    2019-06-02,22:11:51.550
tid:           4
state:         remote-commit
command:       vnet-delete name vn1
undo-command:  vnet-create id c00025b:1 name vn1 scope fabric vrg c00025b:0
vlan-type public vlans 5 config-admin admin 40000 create-vnet-mgr vnet-mgr-

```

```
name vnl-mgr vnet-mgr-storage-pool rpool vnet_mgr_id c00025b:0
vnet_mgr_zone_id c00025b:1 vnet_mgr_location 1 vrg_created_by_vnet true
global false allow_admin_access false admin_role c00025b:400
```

In case of clusters (that is, pairs of redundant devices described in more detail later in this guide) certain transactions are applied to both nodes as if they were a single logical device.

You can choose the transaction information format for a node by typing the `transaction-show` command followed by the `format` parameter to choose a tabular or a vertical output. For example:

```
CLI (network-admin@switch) > transaction-show format all layout vertical

start-time:    03-19,13:46:42
end-time:      03-19,13:46:43
scope:         fabric
tid:           33
state:         remote-commit
command:       --unrecoverable-- vlan-delete id 22
undo-command:  --unrecoverable-- vlan-create id 22 nvid a000030:16 scope
fabric name vlan-22 active yes stats vrg 0:0 ports 1-72,128-129,255
untagged-ports none send-ports 31,41,47-48,51,65-66 active-edge-ports none
ports-specified false flags
-----

start-time:    09:36:09
end-time:      09:36:09
scope:         fabric
tid:           34
state:         remote-commit
command:       vlan-create id 35 scope fabric stats ports-specified true
```

Please note that, in this command's output, the `scope` parameter indicates which set of transactions are displayed, as each scope has an independent set of transactions associated with it. As shown in this example, Netvisor ONE uses `fabric` as the *default scope* unless another scope is specified.

Also note that you should not copy and paste commands and undo-commands from this output because they may include information that does not apply to a different context. As a matter of fact, as mentioned earlier, Netvisor ONE displays the fields for *informational purposes* only so as to allow you to see exactly what happens to the configuration when you roll forward or roll back the transaction ID.

Once you decide which node to modify and the transaction to roll forward or roll back to, you can use the `transaction-rollforward-to` or `transaction-rollback-to` commands to re-run the command (roll forward) or undo the command (roll back) on the node being *troubleshooted*, or even in a broader scope, as described in the following sections.

## Keeping Transactions in Sync with Auto-Recovery

When transactions are executed with a fabric scope, they get applied to all nodes that are part of the same fabric instance as the local node. This process requires coordination and synchronization across the nodes.

You can display the ID of the last executed transaction by using the `fabric-node-show` command:

```
CLI (network-admin@pnswitch1) > fabric-node-show format name,fab-name,fab-
tid,state,device-state,
```

name	fab-name	fab-tid	state	device-state
pnswitch2	pnfabric	<b>2</b>	online	ok
pnswitch1	pnfabric	<b>2</b>	online	ok

The ID is the same because the fabric has made sure that the same transaction be executed consistently on all online nodes.

What happens, though, if one of the nodes is temporarily offline and gets out of sync?

Netvisor ONE's fabric synchronization logic possesses an auto-recovery function (called auto-recover) that makes sure that the transactions get automatically resynchronized when they accidentally go out of sync. This capability is enabled by default:

```
CLI (network-admin@pnswitch1) > transaction-settings-show
```

```
switch: pnswitch1
allow-offline-cluster-nodes: on
auto-recover: on
auto-recover-retry-time: 5m
reserve-retry-maximum: 10
reserve-retry-interval-maximum(s): 8
```

In special scenarios, this capability can be turned off *on purpose* to force the configuration of a specific node to be rolled back to a previous state:

```
CLI (network-admin@pnswitch1) > transaction-settings-modify no-auto-recover
```

```
CLI (network-admin@pnswitch1) > transaction-settings-show
```

```
switch: pnswitch1
allow-offline-cluster-nodes: on
auto-recover: off
auto-recover-retry-time: 5m
reserve-retry-maximum: 10
reserve-retry-interval-maximum(s): 8
```

```
switch: pnswitch2
allow-offline-cluster-nodes: on
auto-recover: on
```

```
auto-recover-retry-time: 5m
reserve-retry-maximum: 10
reserve-retry-interval-maximum(s): 8
```

However, this change is local to the device where the command was issued. Therefore, in those cases in which it is necessary to apply this change to all nodes in the fabric, the `switch *` directive as shown in the following command can be used:

```
CLI (network-admin@pnswitch1) > switch * transaction-settings-modify no-
auto-recover
```

```
CLI (network-admin@pnswitch1) > transaction-settings-show
switch: pnswitch1
allow-offline-cluster-nodes: on
auto-recover: off
auto-recover-retry-time: 5m
reserve-retry-maximum: 10
reserve-retry-interval-maximum(s): 8
```

```
switch: pnswitch2
allow-offline-cluster-nodes: on
auto-recover: off
auto-recover-retry-time: 5m
reserve-retry-maximum: 10
reserve-retry-interval-maximum(s): 8
```

## Rolling Back and Rolling Forward Transactions

---

Netvisor ONE maintains a log file with the list of transactions with their respective *undo commands* to be able to revert back, when necessary, to a previous state, that is, to *roll back* one or more transactions starting from the latest one. On the other hand, the list of executed commands can be used to redo certain transactions, in other words to *roll forward* one or more transactions.

However, this is only desirable under special circumstances, because the *auto-recover* feature by default automatically makes sure that all nodes are synchronized to the latest transaction.

For example in case of rare conditions in which transactions diverge on different nodes (despite auto-recover), a roll back or roll forward action may be performed manually through the corresponding command.

However, the auto-recover function may need to be *temporarily disabled* on the affected node(s) to permit the desired action.

The `transaction-rollback-to` command is used to roll back to an earlier fabric transaction number. The `transaction-rollforward-to` command is instead used to roll forward to a subsequent fabric transaction number.

For instance, the fabric state gets accidentally out of sync according to the `fabric-node-show` command output with, say, a missing interface addition transaction:

```
CLI (network-admin@pnswitch1) > fabric-node-show format name,fab-name,fab-  
tid,state,device-state,
```

name	fab-name	fab-tid	state	device-state
pnswitch2	pnfabric	1	online	ok
pnswitch1	pnfabric	2	online	ok

Hence the state can be rolled back to a previously synced ID to restore fabric-wide (`scope fabric`) consistency:

```
CLI (network-admin@pnswitch1) > transaction-rollback-to scope fabric tid 1  
Warning: rolled back transactions are unrecoverable unless another fabric  
node has them. Proceed? [y/n] y
```

After successfully rolling back the transaction (i.e., no error message is printed on the console), the change completes and the transaction is removed from the transaction log.

Alternatively the state can be rolled forward to reattempt to successfully redo the previously failed fabric-wide interface addition:

```
CLI (network-admin@pnswitch1) > transaction-rollforward-to scope fabric tid  
2  
Added interface eth2.13
```

After successfully rolling forward a transaction (i.e., no error is printed on the console), the change



completes and the transaction log is updated.

If multiple nodes go out of sync, you must *recover each node separately*.

An alternative approach (*usually reserved to customer support for special cases*) is to try to force a roll back or roll forward action when the configuration is in sync but the transaction ID fails to get updated:

```
CLI (network-admin@pnswitch1) > transaction-rollforward-to scope fabric tid
2 ignore-error
Added interface eth2.13
```

In rare cases when you apply a configuration to the fabric and a node does not respond to the configuration, you may want to evict the node from the fabric to troubleshoot the problem on the specific device.

To evict a node, that node must be offline, otherwise the eviction command will fail. Then you can use the `fabric-node-evict` command to perform the eviction process like so:

```
CLI (network-admin@switch) > fabric-node-evict name pnswitch2
```

or

```
CLI (network-admin@switch) > fabric-node-evict id b000021:52a1b620
```

## Rolling Back the Configuration of the Fabric

---

In addition to troubleshooting rare conditions, the roll back action can be used in certain circumstances in which it is beneficial to *turn back time* on all the nodes in the same fabric instance.

As discussed in earlier sections, the fabric natively (i.e., by default) stays *synchronized automatically to the last transaction executed*. Therefore, in order to revert all switches to a previous state, it is necessary to temporarily disable this automatic function (auto-recover) like so:

```
CLI (network-admin@pnswitch1) > switch * transaction-settings-modify no-  
auto-recover
```

Then it is possible to perform a roll back action to a certain ID on all those switches. You can check the latest transaction ID with the command:

```
CLI (network-admin@pnswitch1) > fabric-node-show format name,fab-name,fab-  
tid,state,device-state,
```

name	fab-name	fab-tid	state	device-state
pnswitch1	myfabric	<b>1533</b>	online	ok
pnswitch2	myfabric	<b>1533</b>	online	ok

Let us say you want to roll back all nodes to transaction ID 1530. You can do that with the following command:

```
CLI (network-admin@pnswitch1) > switch * transaction-rollback-to  
transaction-scope fabric tid 1530  
Warning: rolled back transactions are unrecoverable unless another fabric  
node has them.  
Please confirm y/n (Default: n):y  
CLI (network-admin@pnswitch1) > fabric-node-show format name,fab-name,fab-  
tid,state,device-state,
```

name	fab-name	fab-tid	state	device-state
pnswitch1	myfabric	<b>1530</b>	online	ok
pnswitch2	myfabric	<b>1530</b>	online	ok

In this case, though, rolled back transactions are unrecoverable because the fabric logic is forced to *forget* about them on a per node basis where auto-recover was disabled. (So no roll forward is then possible).

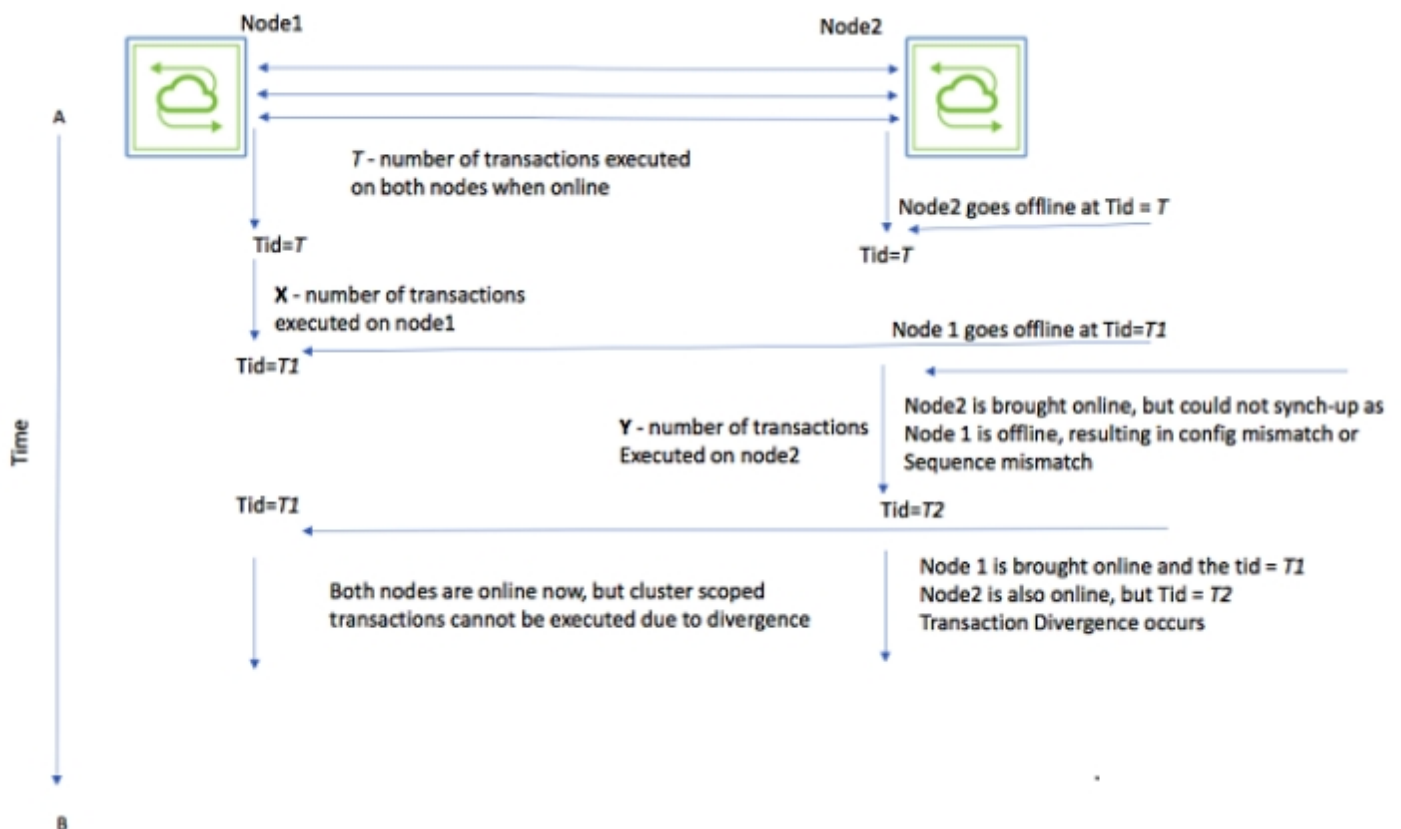
Once all the nodes have been rolled back to the chosen transaction, you can re-enable auto-recover to make sure that the fabric stays in sync automatically from that moment onward:

```
CLI (network-admin@pnswitch1) > switch * transaction-settings-modify auto-  
recover
```

## How to Fix the Cluster Transaction Divergence Issue

When a cluster transaction divergence occur, you cannot execute any further cluster-scoped configurations on either of the cluster nodes. With the release of Netvisor ONE version 5.1.2, we provide a solution to fix the cluster transaction divergence issue.

To ensure high availability, Netvisor ONE allows cluster scope transactions to proceed even when one of the peer nodes in a cluster goes offline (enabled, by default, by using the command: `transaction-settings-modify` with `allow-offline-cluster-nodes` parameter; which can be disabled, you can disable this option, if required) during configuration changes. However, this may cause a transaction divergence as explained in **Figure 6-10**. For example, consider a cluster with two nodes- node 1 and node2, both are online and configurations commands are executed on both nodes, and the transactions are synchronized until a certain time where the TID=T. After certain time, node2 goes offline, but transactions continue on node1 and the TID on node1 changes to T1. Now, both the nodes go offline and then node2 comes back online, and transactions continue on node2, where the TID changes to T2. Now, when node1 also comes back online, the transactions get diverged as illustrated in **Figure 6-10**.



**Figure 6-10: Cluster Transaction Divergence**

The two possible transaction divergence cases as illustrated in **Figure 6-10** include:

- Case 1: Even if  $X$  number of transactions and  $Y$  number of transactions are the same and is also executed in the same sequence ( $T1 = T2$ ), still there will be transaction divergence due to the change in cluster-change IDs.

- Case 2: If the X and Y transactions on the nodes are a separate (different) set of commands or is executed in different sequence, then also transaction divergence occurs.

The auto-recovery feature, which is enabled by default on Netvisor ONE, automatically rolls forward the transactions with the peers when the offline cluster node is brought up online when there is no transaction divergence. But, in the case of transaction divergence, you have to manually roll-back to a common transaction point for the auto-recovery feature to synchronize (for details, see the previous sections of *Troubleshooting the Fabric* section). As this is a manual process, it is error prone (because if you mistakenly roll-back to a different transaction point, then you have to repeat the process to get to the common transaction point).

To mitigate this issue, Netvisor ONE introduces a new command: `transaction-cluster-divergence-fix`. You can run this `local-scoped` command on any of the cluster nodes. However, Pluribus recommends to run this command on a slave node where the divergence occurred.

In case 1 above, when you run the `transaction-cluster-divergence-fix` command on a node, Netvisor ONE updates the change IDs of all diverged transaction IDs to match with the peer node.

In case 2 above, when you run the command on a node, Netvisor fetches the transaction details from the cluster peer and rolls-back the configuration to a common transaction ID on the node on which the command is executed. Then, the node re-synchronizes the configuration with the cluster peer configuration if the auto-recovery function (called `auto-recover`) that makes sure that the transactions get automatically re-synchronized is set to `ON` (This capability is enabled by default). For details on the auto-recovery function, see the [Keeping Transactions in Sync with Auto-Recovery](#) section.

**Note:** It is recommended to take a backup of the current configuration before issuing the `transaction-cluster-divergence-fix` command. Ensure that the fabric communication between the nodes in the cluster is enabled for this functionality to work.

To fix the cluster transaction divergence issue, use the command on the failed cluster node (recommended to run on a slave node):

```
CLI (network-admin@node_slave) > transaction-cluster-divergence-fix
Warning: this will download config from cluster peer and rollback to tid
before it is diverged. It is recommended to take config backup before
running this command.
Please confirm y/n (Default: n):y
```

## Related Commands

```
CLI (network-admin@pn-test-2) > transaction-settings-modify
```

---

Specify one or more of the following options:

---

`allow-offline-cluster-nodes` | `no-allow-offline-cluster-nodes`

---

Specify to allow transactions to proceed even if a cluster node is offline. By default, the `allow-offline-cluster-nodes` parameter is enabled on nvOS.

---

<code>auto-recover   no-auto-recover</code>	Specify to automatically recover missed transactions.
<code>auto-recover-retry-time duration: #d#h#m#s</code>	Specify the retry time for transaction to auto-recover.
<code>reserve-retry-maximum reserve-retry-maximum-number</code>	Specify the maximum number of retries for transaction reservation; 0 for infinite.
<code>reserve-retry-interval-maximum reserve-retry-interval-maximum-number (s)</code>	Specify the maximum number of seconds to wait between transaction reservation retries.

## About the Fabric Default Parameters

---

The following is a list of Netvisor default port numbers, multicast addresses, and communication information about the fabric:

### Internal keepalive message

- Multicast IP: 239.4.9.7
- UDP Destination Port: 23399
- Netvisor sends the packet from the CPU to the internal port to ensure that the CPU path to the switch is working and the internal port is up.

### Fabric keepalive message

- UDP Destination Port: 23394
- Point to point UDP fabric keepalives: if Netvisor cannot receive these messages, the fabric node may go into an offline state.

### Global discovery message

- Multicast IP: 239.4.9.3
- UDP destination port: 23399
- Each node periodically multi-casts a message about the fabric. This enables the `fabric-show` command on directly connected nodes, Layer 2-adjacent, to display available fabric instances and also enables the `fabric-join name name-string` command.

### Proxy commands

- TCP Destination Port: 23397 SSL
- Used for Netvisor ONE-to-Netvisor ONE communication. Used for internal purposes and also to implement commands executed on other switches from a local switch.

### Status propagation

- TCP Destination Port: 23398 SSL
- Port changes and vport changes propagated to other nodes in the fabric.

### TCP API clients

- TCP Destination Port: 23396 SSL
- API clients connect to this port. Disable using the `admin-service-modify if <mgmt/data> no-net-api` command.

### File System replication

- TCP Destination Port: 23392 SSL
- For ZFS send and ZFS receive messages when replicating file systems across the fabric.

### L2 ARP/DMAC miss/Broadcast encapsulation

- UDP Destination Port: 23389
- These are special VXLAN-encapsulated packets sent from one management CPU to another management CPU between two Layer 2 adjacent switches.

### L3 ARP/DMAC miss/Broadcast encapsulation

- UDP Destination Port: 23388
- These are special VXLAN-encapsulated packets sent from one management CPU to another management CPU between two switches interconnected at Layer 3.

#### vPORT status

- Multicast IP: 239.4.9.4
- UDP Destination Port: 23390
- vPort update messages associated to status changes to hypervisors or hosts in the fabric.

#### vFlow CPU packets

- UDP Destination Port: 23398
- These packets are sent point-to-point for vflow-snoop of a fabric-scoped vFlow

**Note:** All of these messages must not drop or become lost in order to keep a fabric healthy. However, the multicast messages do not propagate through routers and therefore Netvisor ONE does not rely on them for fabrics created over Layer 3 networks.

The `fabric-node-show` command displays information about internal data structures for each node in the fabric, including the node state.

For example:

```
CLI (network-admin@switch) > fabric-node-show
```

fab-name	mgmt-ip	in-band-ip	in-band-vlan-type	fab-tid	version	state
WDTest	10.36.10.45/24	10.0.1.1/24	public	7	5.0.0-5000014540	online

If Netvisor ONE does not receive any keepalives or other messages from a fabric node for about 30 seconds, then Netvisor marks the node as `offline`.

Anything that prevents keepalives or other kinds of messages from flowing freely between fabric nodes can cause problems for fabric connectivity.

## Supported Releases

Command/Parameter	Netvisor ONE Version
fabric-create fabric-show	Commands added in <i>version 1.2</i>
fabric-network control-network	Parameters added in <i>version 2.4</i>
fabric-advertisement-network	Parameter added in <i>version 2.4.1</i>
fabric-node-show fabric-join	Commands introduced in <i>version 1.2</i>
fabric-info fabric-unjoin	Commands introduced in <i>version 1.2.1</i>
transaction-show	Command introduced in <i>version 2.2.3</i>
fabric-comm-vrouter-bgp-create	Command introduced in <i>version 2.4.1</i>
fabric-comm-ports-show fabric-comm-ports-modify	Commands introduced in <i>version 3.0.0</i>

Please also refer to the *Pluribus Command Reference* document.

## Related Documentation

Refer to these sections of the Configuration Guides:

- [Implementing a Fabric Upgrade](#)
- [Configuring Layer 2 Features](#)
- [Configuring Layer 3 Features](#)
- [Configuring High Availability](#)
- [Configuring Virtual Networks \(vNETs\)](#)

for further information on concepts mentioned in this section (such as Layer 2, Layer 3, etc.)



## Configuring High Availability

---

This chapter provides information for understanding and using the Pluribus Networks Netvisor OONene command line interface (CLI) on a Netvisor ONE switch to configure High Availability.

---

- [Understanding the High Availability Feature in Netvisor ONE](#)
    - [Understanding Link Aggregation](#)
    - [Understanding the Link Aggregation Control Protocol \(LACP\)](#)
    - [Understanding Switch Clusters](#)
    - [About the Spanning Tree Protocol \(STP\) in Cluster Mode](#)
    - [About Split Brain](#)
    - [About Layer 3 Hardware Forwarding in a Cluster](#)
    - [Understanding vLAGs](#)
    - [Understanding Virtual Router Redundancy Protocol \(VRRP\)](#)
    - [About Cluster Active-Active Routing for IPv6 Addresses](#)
  - [Guidelines and Limitations](#)
  - [Configuring Static Trunking for Link Aggregation](#)
  - [Configuring Active-Standby Link Failover on Management Interfaces](#)
  - [Configuring Link Aggregation Control Protocol \(LACP\)](#)
  - [Configuring a Cluster](#)
  - [Configuring a vLAG](#)
    - [Modifying LACP Mode and Parameters on an Existing vLAG Configuration](#)
  - [Configuring the Cluster Slave Switch Bring-up Process](#)
  - [Configuring Active-Active vLAGs: a Step-by-Step Example](#)
  - [Understanding the vLAG Forwarding Rule](#)
  - [Configuring Asymmetric Routing over vLAGs](#)
  - [About Symmetric Routing over vLAGs](#)
  - [Configuring Active-Active vLAG Forwarding with Loopback Recirculation](#)
  - [Configuring Virtual Router Redundancy Protocol](#)
  - [Troubleshooting High Availability](#)
  - [Supported Releases](#)
  - [Related Documentation](#)
-

## Understanding the High Availability Feature in Netvisor ONE

---

High availability (HA) refers to the characteristic of a system (an individual device or a group of devices) to ensure that service availability (i.e., uptime) is higher than normal and hence downtime is minimized. In case of networking, this is achieved by increasing the resilience (i.e., fault tolerance) of the network so as to be able to provide and maintain an acceptable level of service during faults and challenges to normal operation.

High availability is generally achieved through multiple strategies that include:

- Redundancy of device components (such as power supplies and fans) to maximize the reliability of individual network nodes.
- Device and link redundancy to maximize end-to-end service continuity by eliminating single points of failure and guaranteeing fast re-convergence.
- Simplified operation as well as network automation and orchestration to minimize potential service-disrupting human errors.

Pluribus' Adaptive Cloud Fabric control plane provides intuitive and less error-prone mechanisms to automatically configure and manage a number of physical or logical components as a virtual functional super-entity referred to as a fabric). For details, see the [Configuring and Administering the Pluribus Fabric](#) chapter.

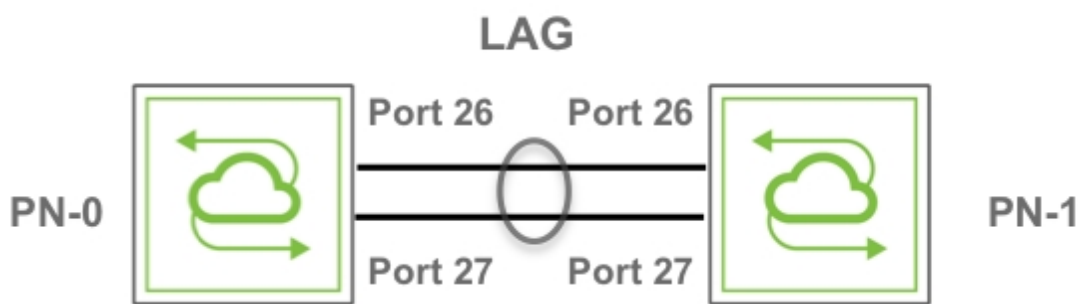
Moreover, as part of the redundant fabric architecture, the Netvisor ONE software supports additional advanced capabilities to increase network automation, resiliency and scalability, such as automatic link aggregation and switch clustering with vLAGs.

## Understanding Link Aggregation

Link aggregation is a standard technology used to combine multiple network connections in order to provide redundancy in case of single or multiple link failure. The augmented **data pipe** generated by aggregating N individual links behaves as a single logical high-bandwidth data path whose capacity can be up to N times the bandwidth of each individual link it comprises. Such logical connection is called a Link Aggregation Group (LAG).

Sometimes it is also called a (port) trunk, link bundle, or port channel as it **bundles** a number of connected physical ports together to implement a single augmented logical communication channel capable of traffic load sharing with fast re-convergence and redundancy.

Therefore, when the number of active bundled ports in a LAG changes for some reason (for example user configuration, hardware fault, etc.), traffic patterns dynamically adapt to reflect the re-balanced state of the LAG with minimal service disruption. Load splitting is achieved by hashing each packet's L2 through L4 headers (when present) to select a physical link to follow.



**Figure 7-1 - Link Aggregation Group (LAG)**

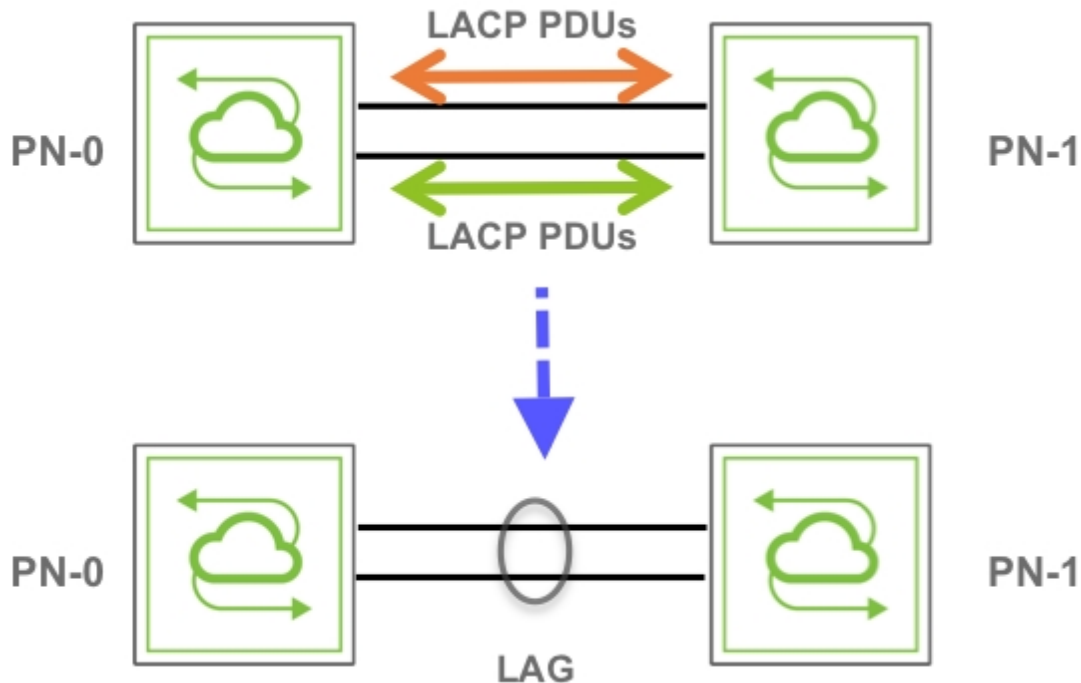
Other terms commonly used to describe this technology included with servers: network interface controller (NIC) bonding or NIC teaming.

Trunks can be either statically set up or dynamically created. In the former case the LAG creation is very fast but it lacks the automation and connectivity checks of the latter case.

For link aggregation the IEEE organization has defined a standard, vendor-independent implementation with the IEEE 802.3ad specification (later transferred to the IEEE 802.1AX-2008 document and superseded by 802.1AX-2014) and has also defined a standard dynamic protocol, called Link Aggregation Control Protocol (LACP), to automate this process.

## Understanding the Link Aggregation Control Protocol (LACP)

The IEEE 802.3ad Link Aggregation Control Protocol (LACP) supports the automatic creation of Ethernet LAGs by exchanging special LACP frames (called LACPDUs) down all the links that have the protocol enabled. If it finds an LACP-capable neighbor device (peer) on the other end of the links, it will negotiate the dynamic creation of a LAG comprising all the compatible links (only full-duplex point-to-point Ethernet links running at the same speed can be matched and grouped together).



**Figure 7-2 - Link Aggregation Control Protocol (LACP)**

The LACP dynamic negotiation can be configured in one of two modes: active or passive.

In active mode LACP always (unconditionally) sends frames along the configured links that are intended to be bundled together. In passive mode, instead, LACP does not initiate a conversation/negotiation until it hears from the peer (this is typically the default configuration). Hence, for the negotiation to start at least one of the peers needs to be switched to active mode by the network administrator.

Once the negotiation is complete and the configured number of ports is aggregated, LACP periodically transmits keepalive frames over a LAG to verify connectivity of each link member and to react (i.e., trigger a link failover) in case of detection of a malfunction. LACP can be configured with a slow or fast lacp-timeout value (90s vs 3s) to react more gracefully or more promptly to connectivity changes.

LACP therefore is very helpful with the function of creating and/or augmenting a LAG as it always performs the necessary validations of the added links (such as bidirectional connectivity).

The maximum number (N) of bundled ports in a LAG is typically (4 or) 8. On top of that a significant number of LAGs (L) may need to be provisioned in a large network design to achieve optimal scalability. Therefore, especially in complex deployments with many LAGs and several ports per LAG, the LACP dynamic function

plays a really crucial role both in simplifying the port bundling process and in making sure that such process runs smoothly and error free over a large number of network entities (N x L).

In addition, in case of link failure, both LACP and the hardware algorithms will then make sure that the re-convergence/failover process is prompt and minimally impactful for the traffic. (On the other hand, static LAG configuration would not guarantee an equally easy and effective behavior in all possible scenarios.)

In summary, the LACP logic performs the following functions on the switch:

- Maintains configuration and port state information to control link aggregation.
- Exchanges configuration and connectivity information with other peer devices.
- Attaches or detaches ports from LAGs based on the exchanged configuration and state information.

LACP frames are always exchanged between neighboring ports when in active mode, whereas in passive mode frames are exchanged as long as the peer is active. This allows the network admin to leverage the automated LACP negotiation logic to control the LAG bring-up or bring-down process by simply configuring one side of a trunk.

Off mode is instead used to disable the LACP function on a trunk when for example the network admin wants to manage the trunk bring up process manually (however that prevents any dynamic negotiation and link verification).

Furthermore, LACP employs two priority values to determine which ports to aggregate into an active LAG. If for example a hardware limitation prevents all configured compatible ports from aggregating, then the priority values will be used by LACP to select which ports to add and which ports to keep in standby mode.

LAG members in standby mode don't actively participate in traffic forwarding but can later be promoted to forwarding state to be used to replace a failed LAG member.

The LACP logic uses a system priority value, which is generated by the combination of the device MAC address with a user-configurable two-octet priority parameter. It also uses a port aggregation priority value, which is generated by the combination of the port number with a user-configurable two-octet priority parameter. Both priority values are used by LACP to dynamically select which port to aggregate in a bundle and which device (with higher priority) is entitled to that decision.

The values for both user-configurable priority parameters can vary from 1 to 65535. Their default configuration is 32768.

Just as their member links, LAGs are point-to-point logical connections, so they can only be as resilient as the networking nodes that they interconnect. In other words, if either peer device at the end of the LAG fails, the entire LAG goes down.

Therefore, in order to further improve the network's fault tolerance, it is of paramount importance to introduce multiple redundant traffic paths in the design (also known as multi-pathing).

This can be achieved in numerous ways at Layer 2 and/or at Layer 3. A very effective Layer 2 strategy to implement dual fully redundant paths with very fast traffic failover is to extend the capability of LAG across two distinct switches to create a virtual LAG device pair.

In Pluribus terminology this cross-chassis LAG capability is implemented with switch clusters.



## Understanding Switch Clusters

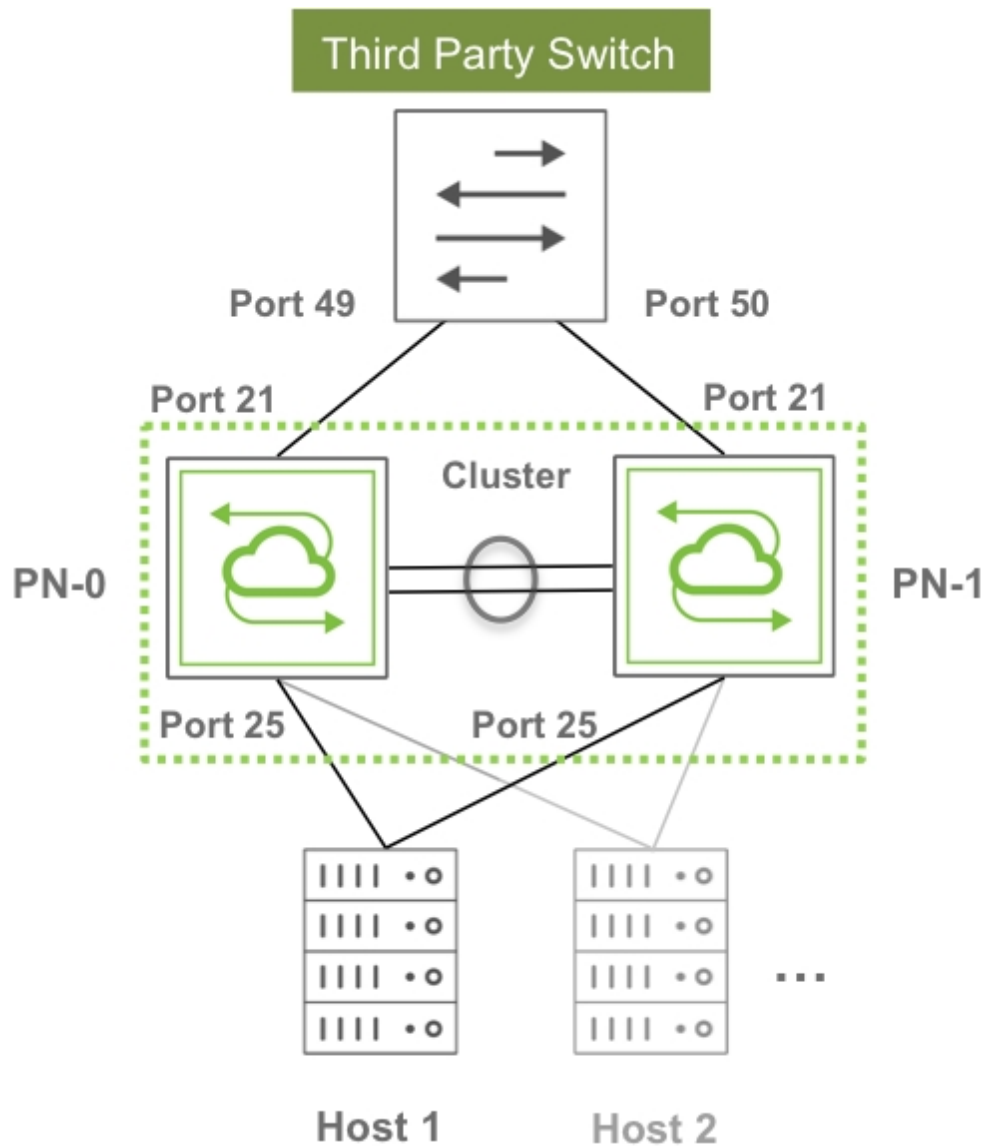
---

In general, in computer science the term **cluster** (also known as high-availability/HA cluster or fail-over cluster) is used to identify a group of devices that are functionally equivalent and structurally redundant so that they are able to provide continuity of service (without user intervention) when any component or an entire device fails. Therefore they are used for critical applications when all single points of failure must be eliminated.

Pluribus Networks builds upon this concept to bring end-to-end multi-pathing with advanced redundancy to high-performance networks where HA is a critical requirement, especially in case of mission-critical high-uptime data center and enterprise deployments.

For this reason Pluribus' clustering technology represents a key pillar of Pluribus' Adaptive Cloud Fabric architecture as well as a powerful tool for network designers. From a practical standpoint it allows customers to deploy in many critical points of the network redundant pairs of switches (simply referred to as switch clusters or clusters) with both upstream and downstream traffic load balancing and fast failover capabilities.

A common scenario is depicted in Figure 7-3 below.



**Figure 7-3 - Example of Requirement for Upstream and Downstream Path Redundancy**

As seen in the figure above, clusters can interconnect third-party switches as well as servers in a redundant fashion simply by using standard technologies (link bundling or NIC bonding) on those devices.

The key is that each redundant pair of Pluribus switches functions as a single logical entity capable of inter-switch state synchronization to guarantee proper Layer 2 operation as well as traffic load balancing and failover. To external devices this switch pair appears as a single virtual neighbor both for traffic forwarding and for protocol communication purposes.

As a matter of fact, in computer networking an alternate name that is sometimes used for a cluster is virtual chassis. And this latter designation is perhaps more useful to describe the concept: a pair of separate physical switches is joined in a virtual entity that behaves as if ports belonging to each device were unified within a common virtual chassis.

In Pluribus' advanced networking architecture clustering is a natural fit and represents a logical extension of the fabric. The requirement for clustering is in fact that the two switches be members of the same fabric



instance.

In addition, due to the asymmetric nature of certain interactions, when you create a cluster you must designate one cluster node as cluster-node-1 (that is, primary node) and the other as cluster-node-2 (that is, secondary node).

In order to set up a cluster, an active link between the primary and the secondary node must exist (as shown above in Figure 3). This is called a “cluster link” and can comprise one or more ports directly connecting the two cluster nodes together. If there are more than one port, they will be automatically bundled and the cluster link will effectively be a “cluster trunk” (that is, the LAG of those ports).

Using an intrinsically redundant cluster trunk for cluster node interconnection is the recommended best practice.

The aggregate bandwidth of such trunk should be such that no major traffic bottleneck is experienced in case of failover scenario and/or in the presence of single-homed upstream or downstream devices (while in the former case the cluster trunk is used as backup path only in case of failure, in the latter case a portion of the traffic is always forced to cross the cluster links). This bandwidth calculation varies depending on the network design, the traffic type, the number and requirements of the single-homed devices, etc.

VLAN 4094 is reserved and is used to carry cluster synchronization traffic. It is automatically added to the in-band interface port and to the cluster link when you create the cluster configuration. This is possible because Netvisor ONE detects cluster links using an extra data set sent in LLDP messages. Therefore, when a cluster link is detected through this mechanism, VLAN 4094 is automatically added to it.

Netvisor ONE performs cluster synchronization over the control network of the fabric. For the in-band interface, synchronization uses VLAN 4094 over the cluster links. For management, synchronization is performed over the management interface.

Each cluster node sends a synchronization message (with cluster state, versioning info, device uptime, etc.) to the peer node every 2 seconds. These messages function also as keepalives: so if three messages in a row are missed, the cluster pair goes offline (that is, symmetrically).

On the other hand, the process of going online for the cluster happens asymmetrically, for two main reasons: to avoid race conditions and to determine an additional important state of each node, the **master** or **slave** role, for protocol support purposes. Synchronization messages are thus exchanged and used to select the master and slave roles via an internal negotiation algorithm based on parameters like uptime.

When a cluster comes online, it triggers:

- A resend of all status updates to the peer node
- A re-synchronization of all vLAGs
- The transition of STP to run in cluster mode

Synchronization of vLAGs (see also the next sections) is used to make sure that they behave as a single logical entity across both nodes. This is achieved by synchronizing port state, as well as Layer 2 entries when needed.

vLAGs in a cluster can work in active-standby mode (where only the vLAG members on one node are active) or in active-active mode (where all the vLAG members are active and forwarding). The latter mode requires Layer 2 entry/vPort synchronization across nodes.

vLAG state synchronization typically happens from the primary node to the secondary node. However, the secondary node can synchronize a (local) port state when that port comes up. In case of necessity it can also request the primary node to (re)start the synchronization process.

Layer 2 entry synchronization is instead performed symmetrically by both nodes upon detection of a vPort change for performance reasons.

## About the Cluster Re-peer Process

When a cluster node dies or needs replacement, it is necessary to use another switch to substitute the failed/missing node to rebuild the cluster pair as soon as possible so as to restore network redundancy.

Initially, the `cluster-repeer` command was used to rebuild a cluster with a new switch node. Subsequently, as an enhancement, the rebuild process was integrated into the fabric join procedure. A cluster pair is now restored via the `fabric-join` command followed by the option:

<code>repeer-to-cluster-node</code>	Replace a dead cluster node by restoring against the existing cluster node.
-------------------------------------	---

Before adding a new cluster node, though, the node to be replaced must first be evicted from the fabric with the `fabric-node-evict` command.

Then, in order to be able to rebuild a cluster via the `fabric-join` command with the `repeer-to-cluster-node` option, the new node needs to be directly connected to the remaining cluster node that is still active.

The joining node that performs a `fabric-join` to repeer with the cluster's master node will initiate direct communication with it to get up-to-date fabric and cluster configuration files and transaction IDs.

After the joining node installs the new configuration, it transmits a fabric-join transaction to notify all fabric members that it has joined the fabric. In addition, it runs a cluster-update command which rebuilds the cluster pair with itself and the existing cluster peer as members. The existing cluster peer will also modify any cluster objects that may require an update as a consequence of the repeer procedure (for example, trunks).

Finally, the joining node restarts and the process is complete.

The aforementioned procedure implicitly applies to Layer 2-based fabric topologies where direct Layer 2 connectivity with the other fabric nodes is possible for message exchange.

In case of Layer 3 fabric designs, instead, a further enhancement has been introduced in Netvisor ONE release 5.1.0 to perform the repeer operation over a Layer 3 fabric interconnect.

A new `fabric-join` option has been added to allow the users to specify that a cluster pair needs to be rebuilt over a Layer 3 fabric configuration:

<code>over-l3</code>	Specify to establish fabric is over layer 3.
----------------------	--

In addition, a new `over-13` flag has been added to the output of the `fabric-info` command to indicate whether it's a Layer 2 or Layer 3 case.

## About the Spanning Tree Protocol (STP) in Cluster Mode

---

In cluster mode Netvisor ONE ensures that the two physical cluster nodes behave as one single logical spanning tree node.

It achieves this by automatically sharing across both devices the same STP bridge ID and STP priorities, and by using dedicated and disjoint STP port ID ranges: STP port IDs on the primary node are 0 to PORTS\_MAX - 1 whereas on the secondary node are PORTS\_MAX to (2\*PORTS\_MAX - 1). This happens regardless of the online/offline state of the cluster and thus guarantees protocol ID consistency/persistency and a seamless failover in case of node failure.

In addition, in case of active-active vLAGs (that is, dual-homed links), Netvisor ONE makes sure that they behave as single logical ports in the STP state machine. For this purpose, rather than creating a new port ID range for these logical ports, the master uses its local STP port ID as the logical port ID (which is preserved across failovers). Therefore, in case of single vLAG member link failure no STP re-convergence occurs. (Please note that with active-standby vLAGs a single link failure can cause an STP topology change.)

If an entire cluster node fails and therefore the cluster goes offline, STP will run the STP state machine locally and independently from the (down) peer. This is therefore called *independent mode*.

Then, when the cluster comes back online and needs to run in STP cluster mode, as mentioned in the previous section, the cluster synchronization process is employed to elect a master node (with the non-elected peer becoming the slave).

Only the master node runs the STP state machine, which encompasses all VLANs and all ports, both local (on the master switch) and remote on the slave switch (cluster links are excluded and are managed separately, as described in a subsequent paragraph).

The master's STP state machine, user configuration changes (of the STP mode, MSTP instances, bridge ID, etc.) and vLAG port states are mirrored to the slave node so that, if the master fails, the slave can seamlessly take over from where the master left off.

In conjunction with the ownership of the STP state machine, it's required for the master switch to manage the very important BPDU transmission function, even for any (single-homed) ports belonging to the slave switch. Not all the aggregate burden of this function falls on the master node, though: the slave switch provides some basic PDU relay support to the master to ensure proper load splitting of this function and hence optimal scalability when running in STP cluster mode. In addition, the slave relays to the master all BPDUs it may directly receive.

As for cluster links, they follow specific STP state transitions: when a cluster goes online, the cluster links start in STP blocked state (except for VLAN 4094 used for synchronization) to prevent loops. However, once the state from the newly elected master gets fully synchronized to its slave, the cluster links are put into STP forwarding state to provide a back-up forwarding path for traffic.

In special cases in which the transmission of the synchronization messages from the master to the slave fails (or times out), the cluster links are put back into the STP blocking state until synchronization can complete to avoid potential loops.

If the master node goes offline, the slave will continue running the STP state machine from the last mirrored state. If the former master then comes back online, it is then selected to become the slave and the state of the node that was already up is mirrored to it.

In this way, there is never any interruption in STP state preservation even after multiple failures of either node, as long as one node is always up.

## About Split Brain

---

If a node in a cluster fails to communicate with its peer (for three consecutive sync messages), the node sets the cluster to offline and functions in independent mode.

This transition is fine when an entire node breaks down, because the sole surviving node has to take up full responsibility to provide continuity of service on its own, without load splitting.

On the other hand, when it's the communication between the two nodes to break down (not an actual device) for an extended period of time, both nodes remain alive but isolated, and the cluster would have to be set to offline state. This condition is called split brain and is undesirable, because the two halves of the cluster work independently instead of in sync with each other, potentially causing network issues.

To avoid ending up with a split brain, that is, with a split cluster, Netvisor ONE can be configured to use a secondary communication channel to verify the health of both nodes. Then, if both nodes are found to be healthy but the inter-communication remains broken, the cluster is flagged as being affected by the split brain condition and is moved to a special state to prevent potential network issues.

To detect split brain, instead of going offline right away in case of broken communication on the primary channel, Netvisor ONE first sets the state of each node to remote-down while it uses a secondary communication channel to reach the peer node. A dedicated management network is often employed by various network designs, hence that can be leveraged as secondary communication channel. If each node can reach its peer through this network (determining that the nodes are indeed healthy but isolated), the master node will then transition to a new state called split-active while the slave node will be set to split-suspended state.

When a slave is in split-suspended state, it brings down its vLAG members to avoid split brain issues on the vLAGs. It also brings down any Layer 3 forwarding function to avoid incorrect routing issues. On the other hand, the master in split-active state will bring up any standby ports that are members of an active-standby vLAG to guarantee continuity of service of that vLAG.

When communication recovery occurs on the primary communication channel, the two nodes will try to re-sync by periodically sending messages to each other. As soon as mutual reach-ability is verified by the nodes, the state of the cluster is set to online.

On the other hand, if a node cannot communicate with its peer using both primary and secondary network, the cluster state is set to offline.

## About Layer 3 Hardware Forwarding in a Cluster

---

In a cluster, Layer 3 hardware forwarding takes place on both nodes running as a VRRP pair (see also the next sections) even if the local VRRP interface is in standby.

In symmetry of routing each node is expected to forward traffic in hardware on behalf of the peer for any packets destined to the VRRP virtual IP address. This can be achieved by synchronizing L3 entries, L3 adjacencies (vPorts) and vRouter state between the two peers.

For single-homed L3 interfaces, instead, nodes may act independently and may even have an L3 adjacency over the cluster link for failover purposes.

## Understanding vLAGs

---

The virtual chassis logical model can be effectively used to enable an advanced capability called multi-chassis link bundling/port channeling. This feature is also referred to as MC-LAG (Multi-Chassis Link Aggregation Group).

An MC-LAG is a cross-device link bundle (i.e., trunk). It typically has at least one port member on one switch and another port member on a backup switch. It can be facing downstream toward an end point (configured with a regular LAG on its end) or upstream toward another switch or router (also using a regular LAG or even an MC-LAG).

The IEEE 802.1AX-2008 standard for link aggregation (LAG) does not mention MC-LAG but does not preclude it, because the implementation of the required synchronization technology can be vendor-specific without affecting interoperability. Such additional under the hood (i.e., mostly transparent) synchronization technology performs advanced functions such as automatic MAC address learning synchronization across members, sharing of the same switch MAC and STP system ID, sharing of port state and information even during dynamic protocol negotiations, etc.

Pluribus' implementation of MCLAG is called vLAG (Virtual Link Aggregation Group) on a switch cluster. That in practice means that in order to be able to configure a vLAG a cluster pair needs to be set up first.

As mentioned in an earlier section, a necessary requirement for a cluster to be created is to provision a redundant **horizontal** connection between the two cluster members: in Pluribus parlance this process is called automatic LAG (link aggregation) of the intra-cluster trunk. As long as at least two ports on both cluster members are physically interconnected, the automatic LAG logic will take care of bundling them together and provisioning them as intra-cluster trunk.

The **cluster trunk** is just a regular LAG, but is the pivot around which various important functions revolve. First off, it provides a critical communication channel utilized for cluster synchronization activities as well as protocol exchanges. It is also instrumental for supporting the MC-LAG (a.k.a. vLAG) function on the cluster, as it can function as a backup forwarding path during failover scenarios. (Therefore it must be a fat enough data pipe to prevent potential bottlenecks in case of failures.)

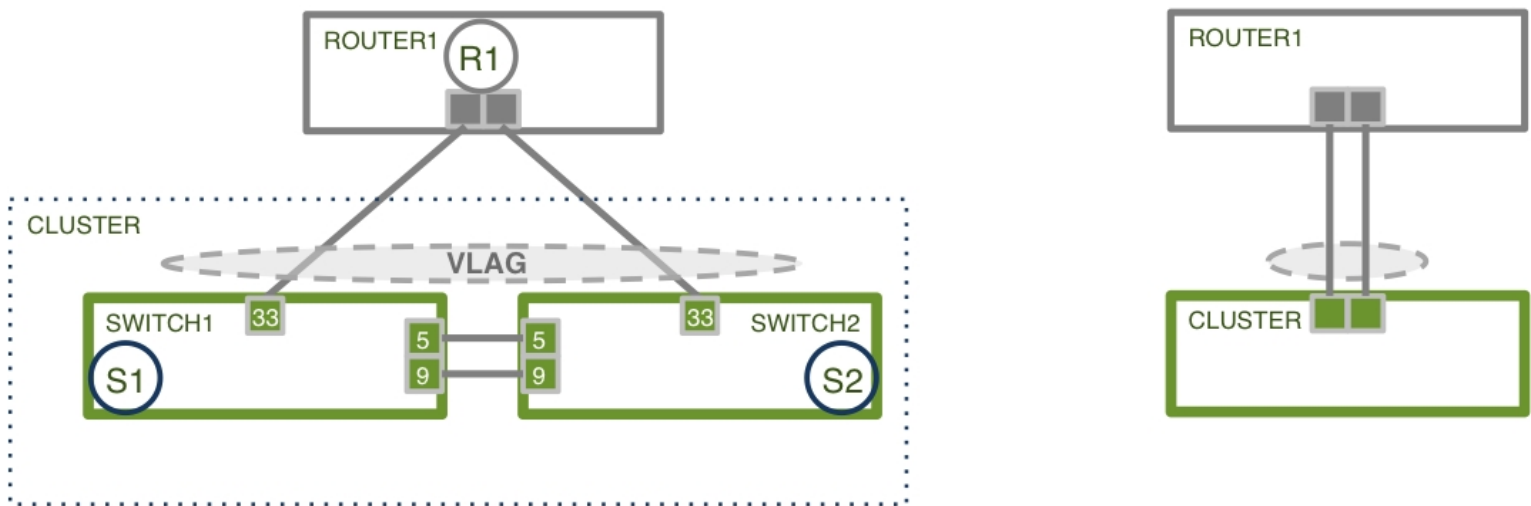
From a high level point of view, a vLAG is a single logical but physically redundant upstream or downstream connection across a pair of cluster switches (see also Figure 4 below). In other words, physically it's a triangular topology that includes the cluster trunk, while logically it appears just as if it were a point-to-point multi-link connection.

This property makes active-active vLAGs ideal for a number of Layer 2 or Layer 3 network designs, where standard port bundling/port channeling technologies (IEEE 802.3ad) can be used for high availability and very fast traffic steering (due to multi-pathing) in case of link failure. (See also the examples in the figures below.)

Using clusters with vLAGs for instance is a technique often chosen to avoid using the Spanning Tree Protocol (STP) for Layer 2 redundancy.

Figure 7-4 below shows how a vLAG is used to present a router/switch with a single logical connection instead of two separate ones. Therefore, in case of a Layer 2 configuration, loop prevention is not needed on that network segment (as no loop is present).





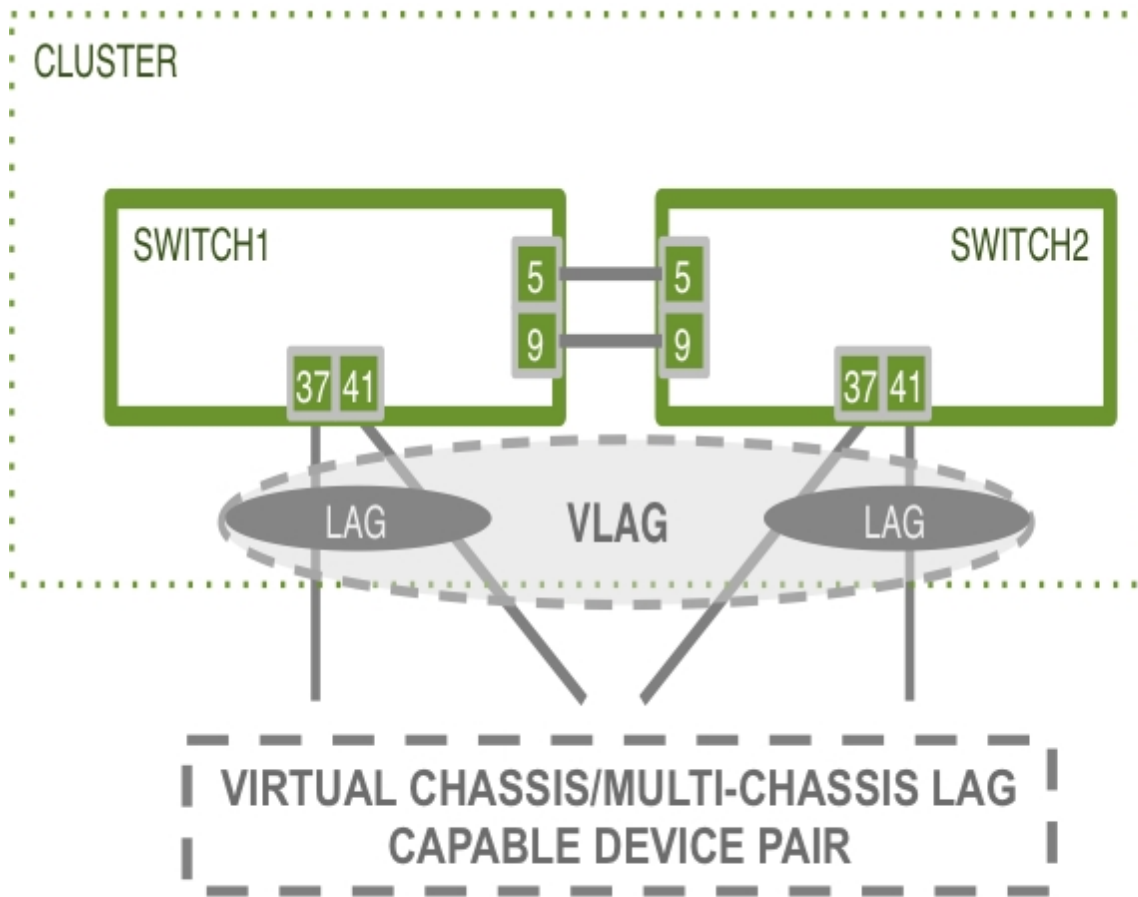
**Figure 7-4 - Logical View of a vLAG from an External Device**

Furthermore, the same strategy can be used downstream too to interconnect dual-homed servers to leaf switches.

In fact, as long as the neighboring device (whether a router, switch or server) supports the standard LACP protocol, port bundling negotiation is taken care of by the protocol itself automatically even when a vLAG is used on the Pluribus switches' side. (Note that even static trunking can be effectively used in these scenarios, especially when dynamic negotiation is not required.) This is by virtue of the interoperability characteristics built in to the Adaptive Cloud Fabric's control plane.

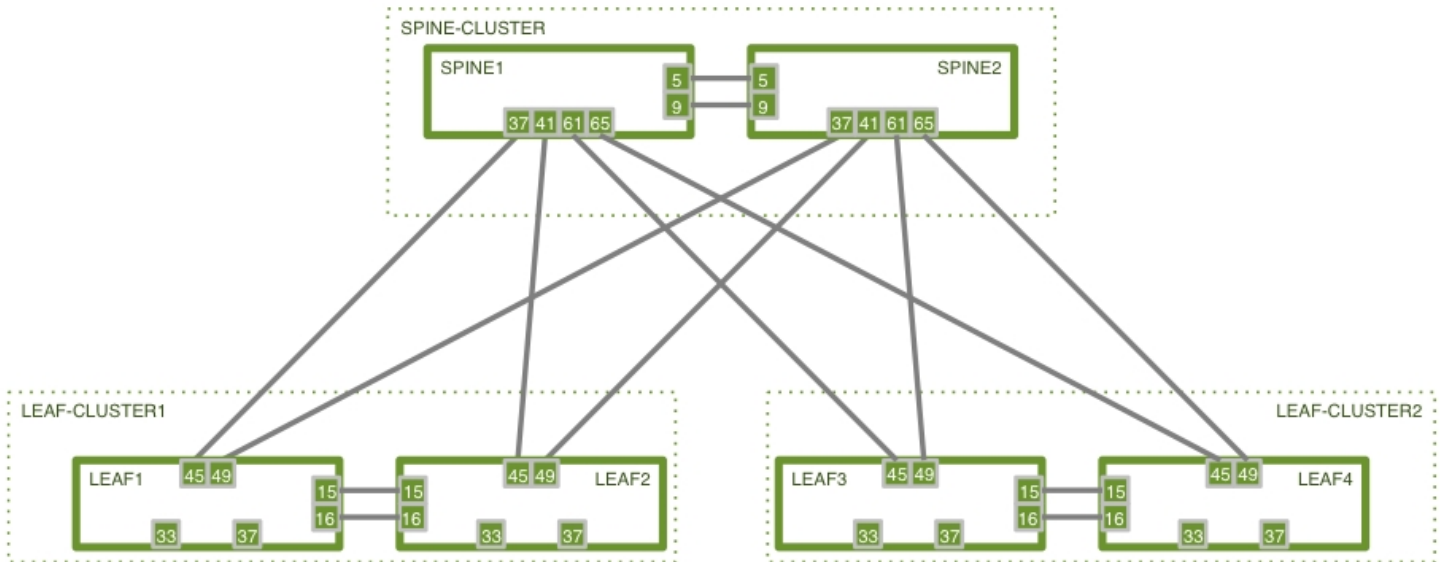
In general, switch clusters with vLAGs are extremely capable and flexible configurations so they can be employed in a number of network designs in conjunction with other standard technologies.

For example, as shown in Figure 7-5 below, vLAGs can connect to other vLAGs or to other third-party virtual chassis/multi-chassis LAG solutions. This creates a “four-way” bundle with the interconnection of two multi-chassis LAGs on opposite sides.



**Figure 7-5 - Four-way vLAG between a Pluribus Cluster Pair and Another Multi-chassis LAG**

Figure 7-6 below shows the common use case of interconnection of leaf and spine Pluribus clusters to provide high availability using the virtual chassis/four way vLAG technique.



**Figure 7-6 - Pluribus High Availability Cluster-based Spine/Leaf Design**

In these network designs using active-active vLAGs (and cluster links), to prevent potential Layer 2 forwarding loops Netvisor ONE automatically installs special internal forwarding rules that drop any packets that would egress a vLAG port if the ingress port is found to be the cluster link.

On top of all of the above, clusters and vLAGs can also be used in conjunction with the standard VRRP technology to provide redundant active-active first-hop Layer 3 gateways to multi-homed end-points.

This capability is particularly important in the context of DCI designs, when used in conjunction with another standard technology, VXLAN, as we will see in a subsequent chapter.

## Understanding Virtual Router Redundancy Protocol (VRRP)

---

The Virtual Router Redundancy Protocol (VRRP) is a standard high-availability protocol initially defined by the Internet Engineering Task Force (IETF) in RFC 2338, later superseded by RFC 5798 for version 3 of the protocol with support for both IPv4 and IPv6.

The goal is to eliminate the single point of failure inherent in the first hop router (default gateway) for the connected hosts. Therefore, when at least two potential first hop routers are deployed, VRRP can be used to dynamically assign responsibility for the function of “virtual next hop” to either of the active routers on the (V)LAN. This is done by implementing an election protocol to select a so-called master router out of the two (or more) VRRP-capable routers available.

A master router performs the function of virtual router, i.e., controls the IPv4 or IPv6 address(es) (called virtual addresses or VIPs) used as default gateway(s) by the hosts and forwards packets sent to these address(es).

Non-elected router(s) are called backup router(s). The VRRP logic supports dynamic failover in the forwarding responsibility to a backup router, if the master router becomes unavailable.

- For IPv4, the critical benefit provided by VRRP is a higher-availability default gateway function with a resilient VIP address.
- For IPv6 configurations in which standard IPv6 Neighbor Discovery mechanisms could potentially help with the selection of the default gateway, the key benefit provided by VRRP is to provide a fast failover and a resilient VIP address.

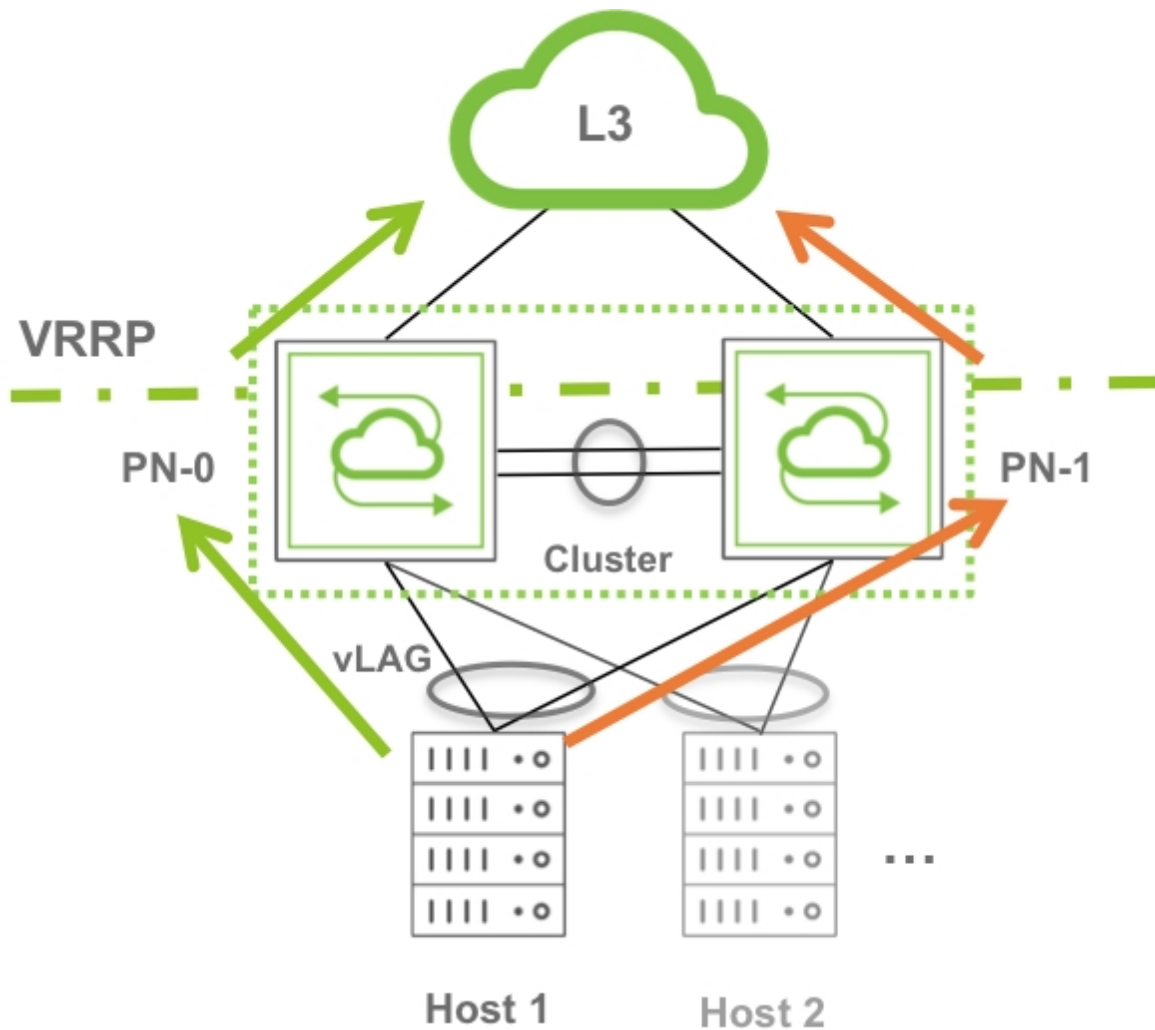
VRRP supports rapid transition from master to backup router in case of node failure. The master router sends VRRP advertisements every second to the backup(s). If the master router's advertisements are not received within a time window of three seconds, then a backup router is elected as the master. If the failed master router becomes active again, it can reclaim the role of master or allow the former backup to continue as the master router. The role depends on the value assigned to a parameter called VRRP priority.

VRRP routers are configured with a priority of between 1 and 254 and the router with the highest priority will be elected to be the master. The default priority is 100.

At Layer 2 a VRRP virtual router must use an address in the 00-00-5E-00-01-XX Media Access Control (MAC) address range. In particular, the last byte of the address (XX) corresponds to the Virtual Router Identifier (VRID), which is distinct for each virtual router in the network. A VRRP virtual router will reply with this special MAC address when an ARP request is sent for the virtual router's IPv4 address.

As we will see in subsequent sections, having a resilient VIP that survives device failures is critical for simpler and more effective high availability of advanced services.

In Pluribus' network designs VRRP can be used in conjunction with clusters and vLAG to combine fast switchover capabilities at Layer 2 with a redundant Layer 3 VIP for first hop routing toward the upstream part of the network (i.e., toward spine switches or other network devices).



**Figure 7-7 - Example of Use of VRRP in Conjunction with a Cluster**

In addition, Pluribus' implementation optimizes the performance of this technology combination by supporting active-active Layer 3 forwarding on both VRRP routers in a cluster pair.

In other words, when a VIP is configured, each router is expected to route in hardware on behalf of the other peer for any packets destined to the VIP.

## About Cluster Active-Active Routing for IPv6 Addresses

---

With cluster active-active routing two cluster peers act as forwarding proxies for each other's destination MAC addresses. This works for IPv4 as well as IPv6 traffic. But if the two routers try to communicate with each other, packets may not route correctly on the network.

In particular, in case of the Neighbor Discovery function IPv6 uses ICMPv6 packets (instead of L2 ARP packets, as in the case of IPv4). Therefore, when one cluster node PN-0 sends a Neighbor Solicitation message to its peer, the other node PN-1 responds with a Neighbor Advertisement message with the destination IP address of the requester (PN-0). When transmitted by PN-1 the Neighbor Advertisement message will have both the destination IP and the destination MAC address of the peer PN-0. Since both nodes act as forwarding proxies for each other, the destination MAC address of the peer will be matched by the hardware of PN-1 and the packet will be routed (back to the CPU of PN-1) without reaching its correct destination (i.e., PN-0). This behavior would therefore break the Neighbor Discovery function of IPv6.

To obviate this problem, Netvisor ONE adds a host route in hardware that matches the link-local IPv6 address of the cluster peer.

This host route entry properly routes packets such as Neighbor Advertisement messages to their destination.

## Guidelines and Limitations

---

For recommended cluster and vLAG design topologies, refer to the section “[About Symmetric Routing over vLAGs](#)”.

## Configuring Static Trunking for Link Aggregation

---

To statically configure a Link Aggregation Group (LAG), also known as trunk, you can use the `trunk-create` command.

For example, in order to aggregate the three links connected to ports 1, 2, 3 on a switch, use the following steps:

1) Create a trunk called trunk-1 on ports 1, 2, 3 by entering the following command:

```
CLI (network-admin@switch) > trunk-create name trunk-1 ports 1,2,3
```

2) To verify the result of the configuration, use the `trunk-show` command:

```
CLI (network-admin@switch) > trunk-show
```

name	port	speed	autoneg	jumbo
trunk-1	1-3	10g	off	off

3) You can modify the trunk configuration for example by removing port 2 from the group:

```
CLI (network-admin@switch) > trunk-modify name trunk-1 port 1,3
```

4) Verify the updated trunk configuration:

```
CLI (network-admin@switch) > trunk-show
```

name	port	speed	autoneg	jumbo
trunk-1	1,3	10g	off	off

Notice that the list of ports has changed from 1-3 to 1,3 indicating that port 2 is no longer a member of the trunk.

To remove the trunk from the switch, use the `trunk-delete` command:

```
CLI (network-admin@switch) > trunk-delete name trunk-1
```

Verify that the trunk configuration is removed again by using the `trunk-show` command.



## Configuring Active-Standby Link Failover on Management Interfaces

---

With switches that have dual management ports, you can aggregate them in a LAG with both of them being active and forwarding. This is useful when increased bandwidth is required for management purposes or when the upstream management network supports solutions like MG-LAG (aka virtual chassis) to remove single points of failure.

Alternatively, when it's desirable that only one of the two management interfaces be forwarding, the so-called “active-standby” or “active-backup” mode of operation is supported for LAG on management ports.

In this case, the management interfaces form a LAG with only one of the two members active at any time. The other link is in standby mode, which means that it becomes forwarding only when the primary link goes down. When the primary link comes back up, it rejoins the LAG and the backup link returns to standby state.

Use the `switch-setup-modify` command to set up a management LAG.

Use the `active-standby` option to enable the active-backup mode.

```
CLI (network-admin@switch) > switch-setup-modify mgmt-lag <disable| enable  
| active-standby>
```

## Configuring Link Aggregation Control Protocol (LACP)

---

The Link Aggregation Control Protocol (LACP) is a dynamic LAG protocol that is controlled by user configuration. The negotiation process and the addition/removal of ports is controlled by parameters like LACP mode (active, passive and off) and LACP priorities.

To dynamically bring up a trunk the ports on at least one of the two peer LACP switches must be set to active mode.

For example use the following command to dynamically bundle ports 20-23:

```
CLI (network-admin@switch) > trunk-create name trunk23 ports 20-23 lacp-mode active
```

To modify the mode of a trunk running LACP, use the following command:

```
CLI (network-admin@switch) > trunk-modify name trunk23 lacp-mode passive
```

To modify a port configuration with a non default LACP priority, use the following command:

```
CLI (network-admin@switch) > port-config-modify port 33 lacp-priority 34
```

To enable or disable LACP on a switch, and/or to change the default system priority value, you can use the following command:

```
CLI (network-admin@switch) > lacp-modify enable|disable system-priority priority
```

The default system-priority value is 32768 with a range of 1 to 65535.

To display LACP information, use the following command:

```
CLI (network-admin@switch) > lacp-show
```

switch:	switch
enable:	yes
system-priority:	32768
systemid:	800640e942c007a

## Configuring a Cluster

---

To set up a cluster of two switches, say, `pleiades1` and `pleiades2`, you must first verify that they both belong to the same fabric instance, in this case `corp-fab`:

```
CLI (network-admin@switch) > fabric-node-show layout vertical
```

```
id: 184549641
name: pleiades1
fab-name: corp-fab
fab-id: b000109:5695af4f
cluster-id: 0:0
local-mac: 3a:7f:b1:43:8a:0f
fabric-network: in-band
control-network: in-band
mgmt-ip: 10.9.19.203/16
mgmt-mac: ec:f4:bb:fe:06:20
mgmt-secondary-macs:
in-band-ip: 192.168.168.203/24
in-band-mac: 3a:7f:b1:43:8a:0f
in-band-vlan: 0
in-band-secondary-macs:
fab-tid: 1
cluster-tid: 0
out-port: 0
version: 2.4.204009451, #47~14.04.1-Ubuntu
state: online
firmware-upgrade: not-required
device-state: ok
ports: 104
```

To create a cluster configuration, use the following command:

```
CLI (network-admin@switch) > cluster-create name cluster1 cluster-node-1
pleiades1 cluster-node-2 pleiades2
```

To verify the status of the cluster, use the `cluster-show` command:

```
CLI (network-admin@switch) > cluster-show
```

name	state	cluster-node-1	cluster-node-2
-----	-----	-----	-----
cluster1	online	pleiades1	pleiades2

To replace a failed cluster node, use the `fabric-join repeer-to-cluster-node` command.

However, you must evict the failed node from the fabric, and then run the `fabric-join repeer-to-cluster-node` command on an active node after replacing the failed node.

To display information about the cluster, use the `cluster-info` command:

```
CLI (network-admin@switch) > cluster-info format all layout vertical
```

```
name:                cluster1
state:               online
cluster-node-1:      pleiades1
cluster-node-2:      pleiades2
tid:                 2
mode:                master
ports:               69-71,129
remote-ports:        69-71,128
```

## Performing the Cluster Re-peer Process

---

In order to rebuild a cluster over a Layer 2 or Layer 3 fabric it's necessary to use the `fabric-join` command with the `repeer-to-cluster-node` option.

For example, if one wants to join `switch1` from `switch3` to rebuild a cluster pair (that previously lost `switch2`), the following commands can be used:

- In case of a Layer 2 fabric:

```
CLI (network-admin@switch3) > fabric-join repeer-to-cluster-node switch1
```

- In case of a Layer 3 fabric (using for example OSPF or BGP for interconnection), the command changes to:

```
CLI (network-admin@switch3) > fabric-join repeer-to-cluster-node switch1 ?
```

```
password          plain text password
abort-on-conflict  fabric join is aborted
delete-conflicts   conflicts are deleted
over-l3           fabric is over layer 3
```

```
CLI (network-admin@switch3) > fabric-join repeer-to-cluster-node switch1
over-l3
```

Joined fabric myFabric. Restarting nvOS...

Please enter username and password:

Username (network-admin):

Password:

Connected to Switch switch3; nvOS Identifier:0x900093c; Ver: 5.1.0-5010014758

You can display the information about the fabric instance of the local switch using the `fabric-info` command:

```
CLI (network-admin@switch3) > fabric-info format all layout vertical
```

```
name:                myFabric
id:                  a000030:5537b46c
vlan:                3
fabric-network:      in-band
control-network:     in-band
tid:                 365
fabric-advertisement-network: inband-only
over-l3:             true
```

## Configuring a vLAG

If you want to connect the two cluster nodes to an upstream switch or to a downstream host in a redundant fashion, you can configure a vLAG between the uplinks or the downlinks on the cluster nodes.

For example, if `switch1` has port 53 connected to an upstream switch and `switch2` has port 19 connected to the same upstream switch, create a vLAG by executing the `vlag-create` command on either of the cluster switches (`switch1` in this example):

```
CLI (network-admin@switch) > vlag-create name uplink port 53 [peer-switch switch2] peer-port 19
```

To verify the configuration, use the following command:

```
CLI (network-admin@switch) > vlag-show
```

name	cluster	mode	switch	port	peer-switch	peer-port	status	local-state	lacp-mode
----	-----	----	-----	----	-----	-----	-----	-----	-----
uplink	cluster1	active-acti	switch1	53	switch2	19	normal	enabled	active

**Note:** Before you can create a vLAG, you must configure the two switches in a cluster.

## Modifying LACP Mode and Parameters on an Existing vLAG Configuration

You can modify the LACP mode as part of an existing vLAG configuration using the following command to change parameters like LACP mode or timeout:

```
CLI (network-admin@switch) > vlag-modify
```

<code>name name-string</code>	Specify the vLAG name.
<code>failover-move-L2 failover-ignore-L2</code>	If you specify the parameter <code>failover-move-L2</code> , Netvisor sends gratuitous ARPs.
<code>lacp-mode off passive active</code>	Specify the LACP mode as off, passive or active.
<code>lacp-timeout slow fast</code>	Specify the LACP timeout as slow (90 seconds) or fast (3 seconds).
<code>lacp-fallback bundle individual</code>	Specify the LACP fallback mode as individual or bundled.
<code>lacp-fallback-timeout 30..60</code>	Specify the LACP fallback timeout in seconds. The default is 50 seconds.

Configuring a (v)LAG in LACP fallback mode *individual* (vs. *bundle*) allows it to operate as individual ports in the absence of proper LACP negotiation with a network peer. This configuration is useful with hosts with multiple network interfaces that for example need to boot from a server on the network prior to booting the operating system on the local hard drive (this is known as PXE boot, based on the Pre-Boot Execution Environment). In this scenario the hosts would have to use individual ports until they can boot the operating system and start using LACP.

Once LACP is running and port members receive LACP PDUs from their peers, then the port members of the configured vLAG can get bundled to operate as a trunk.

With this configuration, Netvisor ONE logically prepares the trunk on the switch but does not add any of the port members to it. The ports continue to operate individually until LACP PDUs are heard on them. At that point, the dynamic negotiation process is started, proper compatibility and communication is verified between the peers and the bundling process is carried out if all conditions are met. Only then all member ports cease to operate individually and are aggregated together as an operational trunk.

To give hosts enough time to PXE boot, a fallback timeout interval parameter is used (with a default value of 50 seconds). If no LACP PDUs are received for the number of seconds configured as the fallback timeout, the LACP logic checks if the LACP negotiation interval (`lacp-timeout`) has expired. If it has, then the port members fall back to individual mode. If it has not, another fallback timer is scheduled with a value equal to the fallback timeout.

For example:

- The LACP fallback timeout is set to 50 seconds and the LACP negotiation interval is set to 90 seconds (default).
- After 50 seconds, the fallback timer is rescheduled because the LACP negotiation interval has not

expired yet.

- After an additional 40 seconds (90 seconds total), the LACP negotiation interval expires.
- Another 10 seconds pass (100 seconds total) when the fallback timer expires, no LACP PDUs have been received (for example due to the host still booting), then the member ports fall back to individual ports.

Both the negotiation/bundling and the fallback cases described above represent the on-demand characteristic of LACP-based port aggregation process (applicable to both vLAGs and normal trunks).



## Configuring the Cluster Slave Switch Bring-up Process

**Note:** This feature is applied only during the initial start-up of the cluster slave switch.

When a (previously down) slave node comes back up and a cluster is thus restored to online state, it can take measurable time to change the hardware routing and Layer 2 tables. This set-up delay may cause some non-negligible traffic loss on the network if lots of hardware changes are performed in bulk.

Therefore, to minimize service disruption to a negligible interval, the bring-up process is performed in an optimized staggered fashion by incrementally restoring/updating the ports. (This mode can be changed as shown below.) Some delays are also added during the staggered bring-up to make sure that state transitions and synchronization complete before moving to the next set of changes. (Such delays are configurable as shown below.)

In order to minimize any traffic impact, all non-Layer 3 and non-vLAG ports are restored first. This allows the cluster links to activate and the cluster configuration and state to synchronize.

After synchronization is complete, Layer 3 ports are brought up first. After that, Netvisor ONE waits for a configurable `l3-to-vlan-interface-delay` before bringing up vRouter vlan interfaces. This is so that routing protocols can delay advertising directly-attached subnets upstream.

After Layer 3 ports are brought up, Netvisor ONE also waits for another configurable `l3-to-vlag-delay` to allow time for the routing protocols to converge and program the routes. This time defaults to 15 seconds. After that delay, the vLAG ports are brought up staggered. (Note that `l3-to-vlan-interface-delay` should be less than or equal to `l3-to-vlag-delay`.)

The `maximum-sync-delay` parameter controls the maximum time to wait for synchronization in the case where the cluster cannot synchronize information.

Note that, if the node coming up is a cluster's master, then no staggering and no Layer 3-to-vLAG delay is applied (because it's not beneficial).

With version 5.1.1, Netvisor ONE also supports the staggered bring up of vRouter VNICs. When configured, the vRouter VNICs, which are not configured on Layer3 ports, are brought up in a staggered manner during the nvOS boot-up on a cluster peer. You can specify the wait time (0-60000 ms) between NIC bring up.

By default, the vRouter interfaces for which the VLAN is up is brought up simultaneously. The bring up can be staggered by specifying a non-zero `vrouter-if-staggered-interval`. The staggered bring up process is helpful to reduce the traffic loss caused by the simultaneous bring up of all VNICs.

**Note:** The vRouter interfaces on Layer 3 ports are brought up first and is not staggered. Also, after bootup, the subsequent VLAN interfaces being down or up is not affected by the staggered configuration.

In most cases it is not necessary to change the default parameter values.

However, in certain scenarios to be able to further optimize the slave bring up process to match one's specific requirements, it is possible to modify the parameters by using the following command:

```
CLI (network-admin@switch) > cluster-bringup-modify
```

You can specify one or more of the following options:

<code>l3-port-bringup-mode staggered simultaneous</code>	Specify the Layer 3 port bring up mode during start up.
<code>l3-port-staggered-interval duration: #d#h#m#s</code>	Specify the interval between Layer 3 ports in Layer 3 staggered mode. This can be in days, hours, minutes, or seconds.
<code>vlag-port-bringup-mode staggered simultaneous</code>	Specify the vLAG port bring up mode during start up.
<code>vlag-port-staggered-interval duration: #d#h#m#s</code>	Specify the interval between vLAG ports in vLAG duration: This can be in days, hours, minutes, or seconds.
<code>maximum-sync-delay duration: #d#h#m#s</code>	Specify the maximum delay to wait for cluster to synchronize before starting Layer 3 or vLAG port bring up. This can be in days, hours, minutes, or seconds.
<code>l3-to-vlag-delay duration: #d#h#m#s</code>	Specify the delay between the last Layer 3 port and the first vLAG port bring up. This can be in days, hours, minutes, or seconds. The default value is 15 seconds.
<code>l3-to-vlan-interface-delay duration: #d#h#m#s</code>	Specify the delay between last L3 port and vRouter vlan interface bring up. This can be in days, hours, minutes, or seconds.
<code>port-defer-bringup-delay duration: #d#h#m#s</code>	Specify the global timer value to be used for port delay-bring up.
<code>port-defer-bringup-mode staggered simultaneous</code>	Specify the port defer bring up mode during start up.
<code>port-defer-bringup-staggered-interval duration: #d#h#m#s</code>	Specify the interval between ports in defer bring up mode.
<code>vrouter-if-bringup-mode staggered simultaneous</code>	Specify the vRouter VLAN interface bring up mode.
<code>vrouter-if-staggered-interval 0..60000</code>	Specify the interval in ms between vRouter VLAN interface bring up in staggered mode. The value ranges between 0-60000 ms.

To display the status of the cluster bring up process, use the `cluster-bringup-show` command:

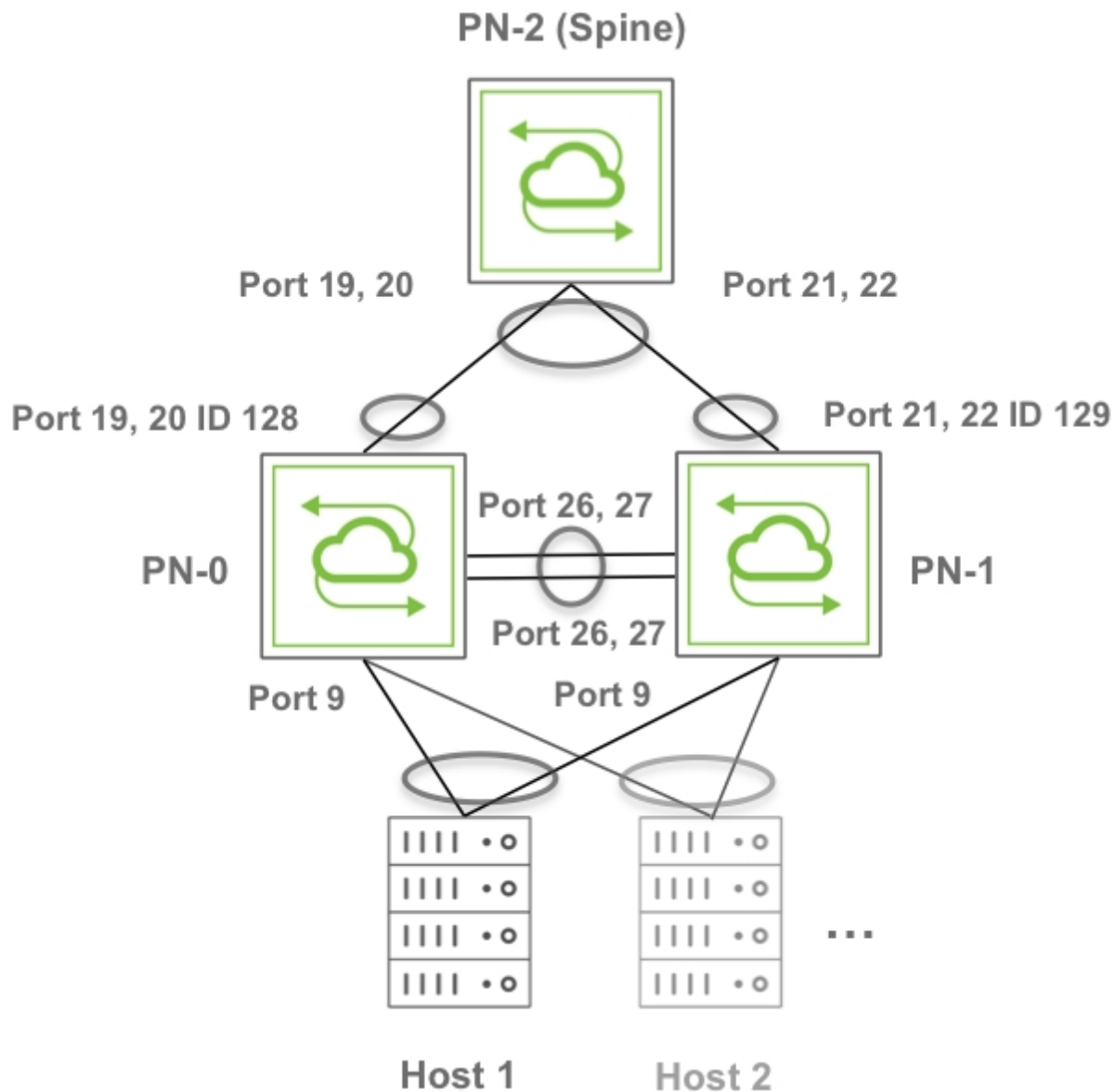
```
CLI (network-admin@switch) > cluster-bringup-show
```

```
state:                                ports-enabled
l3-port-bringup-mode:                 staggered
```

```
l3-port-staggered-interval:    3s
vlag-port-bringup-mode:        staggered
vlag-port-staggered-interval:  3s
maximum-sync-delay:            1m
l3-to-vlag-delay:              15s
l3-to-vlan-interface-delay:    0s
```

## Configuring Active-Active vLAGs: a Step-by-Step Example

**Note:** As displayed in **Figure 7-8**, there must be a physical connection between PN-0 and PN-1 before you can configure a vLAG.



**Figure 7-8 - Active-Active vLAGs Toward a Dual-Homed Host and a Spine Switch**

The sample topology in Figure: Active-Active vLAGs Toward a Dual-Homed Host and a Spine Switch above comprises three Pluribus switches that are part of the same fabric instance, fab-vLAG. The spine switch is set as RSTP root.

It's important to note that ports 19-20 on PN-0 and ports 21-22 on PN-1 are connected to PN-2 (Spine). Ports

26,27 interconnect PN-0 and PN-1 for the cluster link configuration required to set up a vLAG.

You can use the following steps to configure the above-pictured configuration with active-active vLAGs.

1) On the spine switch PN-2 use the following command:

```
CLI (network-admin@switch) > stp-modify bridge-priority 4096
```

2) Create the fabric and add the switches to it. On PN-2 use the `fabric-create` command:

```
CLI (network-admin@switch) > fabric-create name fab-VLAG
```

On PN-0 and PN-1 join the fabric:

```
CLI (network-admin@switch) > fabric-join name fab-VLAG
```

```
CLI (network-admin@switch) > fabric-join name fab-VLAG
```

3) Create VLAN connectivity from the spine switch all the way to the host. On PN-2 create for example VLAN 25 with scope fabric:

```
CLI (network-admin@switch) > vlan-create id 25 scope fabric
```

On PN-0 and PN-1 add VLAN 25 and untag the port connected to the host:

```
CLI (network-admin@switch) > vlan-port-add vlan-id 25 untagged ports 9
```

```
CLI (network-admin@switch) > vlan-port-add vlan-id 25 untagged ports 9
```

On PN-0 and PN-1 make the port connected to the host be an edge port:

```
CLI (network-admin@switch) > stp-port-modify port 9 edge
```

```
CLI (network-admin@switch) > stp-port-modify port 9 edge
```

4) Create a cluster configuration between PN-1 and PN-0. This creates the cluster link between ports 26,27. On PN-0 enter the `cluster-create` command:

```
CLI (network-admin@switch) > cluster-create name VLAG cluster-node-1 PN-0  
cluster-node-2 PN-1
```

5) In this example, for simplicity's sake we will use static trunks toward the spine switch to create a vLAG. (LACP's active mode can be used too, instead of off mode.) In this case you would first disable ports between PN-2 and PN-0 and then create a static trunk between them. Therefore, on PN-0 modify the ports facing PN-2 like so:

```
CLI (network-admin@switch) > port-config-modify port 19,20 disable
```

6) Create a two-port trunk between PN-0 and PN-2 with those ports:

```
CLI (network-admin@switch) > trunk-create name pn0-to-pn2 ports 19,20 lacp-  
mode off
```

```
CLI (network-admin@switch) > trunk-show format all layout vertical
```

```
switch:                PN-0  
trunk-id:               128  
name:                   pn0-to-pn2  
ports:                  none  
speed:                  disable  
egress-rate-limit:      unlimited  
autoneg:                 off  
jumbo:                   on  
enable:                  off  
lacp-mode:               off  
lacp-priority:           0  
lacp-timeout:            slow  
lacp-fallback:           bundle  
lacp-fallback-timeout:  50  
lacp-individual:         none  
stp-port-cost:           2000  
stp-port-priority:       128  
reflect:                 off  
edge-switch:            no  
pause:                   no  
description:               
loopback:                off  
receive-only:            on  
unknown-ucast-level:     %  
unknown-mcast-level:     %  
broadcast-level:         %  
lport:                   0  
rem-rswitch-port-mac:    00:00:00:00:00:00  
rswitch-default-vlan:    0  
status:                    
config:                    
trunk-hw-id:             0  
send-port:               4294967295  
routing:                  yes  
host-enable:             no
```

From the above output, you can verify the name and ID of the trunk configuration *pn0-to-pn2*. You need this information to create the vLAG.

7) On PN-1 repeat the same commands to create a static two-port trunk (LACP mode off) between PN-1 and PN-2. You would, therefore, disable ports between PN-2 and PN-1 and then create a static trunk between them. On PN-1 modify the ports facing PN-2 like so:

```
CLI (network-admin@switch) > port-config-modify port 21,22 disable
```

```
CLI (network-admin@switch) > trunk-create name pn1-to-pn2 ports 21,22 lacp-mode off
```

```
CLI (network-admin@switch) > trunk-show format all layout vertical
```

```
switch:                PN-1
intf:                  129
name:                  pn1-to-pn2
port:                  21-22
speed:                 10g
autoneg:               off
jumbo:                 off
enable:                off
lacp-mode:              off
lacp-priority:          32768
lacp-timeout:           slow
reflect:               off
edge-switch:           no
pause:                 no
description:
loopback:              off
mirror-only:           off
lport:                 0
rswitch-default-vlan:  0
port-mac-address:      06:60:00:02:10:80
status:
config:
send-port:             0
```

8) Create the vLAG from the bottom switches going upstream. Keep one side of the vLAG disabled while you configure this step. On PN-0 use the `vlag-create` command:

```
CLI (network-admin@switch) > vlag-create name to-spine port 128 peer-port 129 peer-switch PN-1 lacp-mode off mode active-active
```

On PN-2 create a normal 4-way trunk with the name `trunk-pn`:

```
CLI (network-admin@switch) > trunk-create name trunk-pn ports 19,20,21,22 lacp-mode off
```

9) Enable ports on all switches. On PN-2, PN-0 and PN-1 enter the `port-config-modify` command:

```
CLI (network-admin@switch) > port-config-modify port 19,20,21,22 enable
```

```
CLI (network-admin@switch) > port-config-modify port 19,20 enable
```

```
CLI (network-admin@switch) > port-config-modify port 21,22 enable
```

10) As a final step, create the server-facing active-active vLAG. In this case we will make it dynamic (LACP mode active). On PN-0 enter the vlag-create command:

```
CLI (network-admin@switch) > vlag-create name to-host port 9 peer-port 9  
peer-switch PN-1 lacp-mode active mode active-active
```

Display and verify the vLAG configuration information:

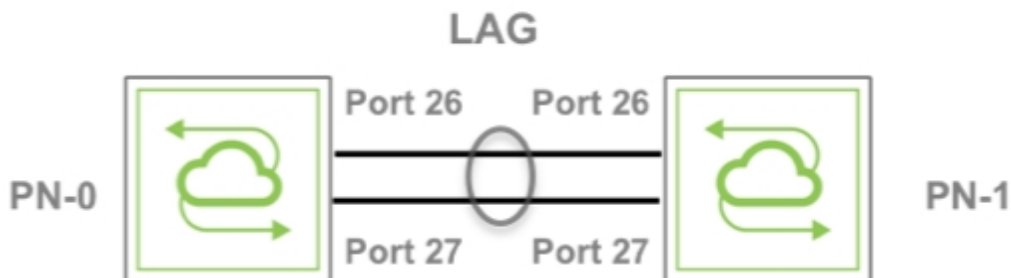
```
CLI (network-admin@switch) > vlag-show format all layout vertical
```

```
id:                a000024:0  
name:              to-host  
cluster:          VLAG  
mode:             active-active  
switch:           PN-0  
port:             9  
peer-switch:      PN-1  
peer-port:        9  
failover-move-L2: no  
status:           normal  
local-state:      enabled, up  
lacp-mode:        active  
lacp-timeout:     slow  
lacp-key:         26460  
lacp-system-id:   110013777969246
```



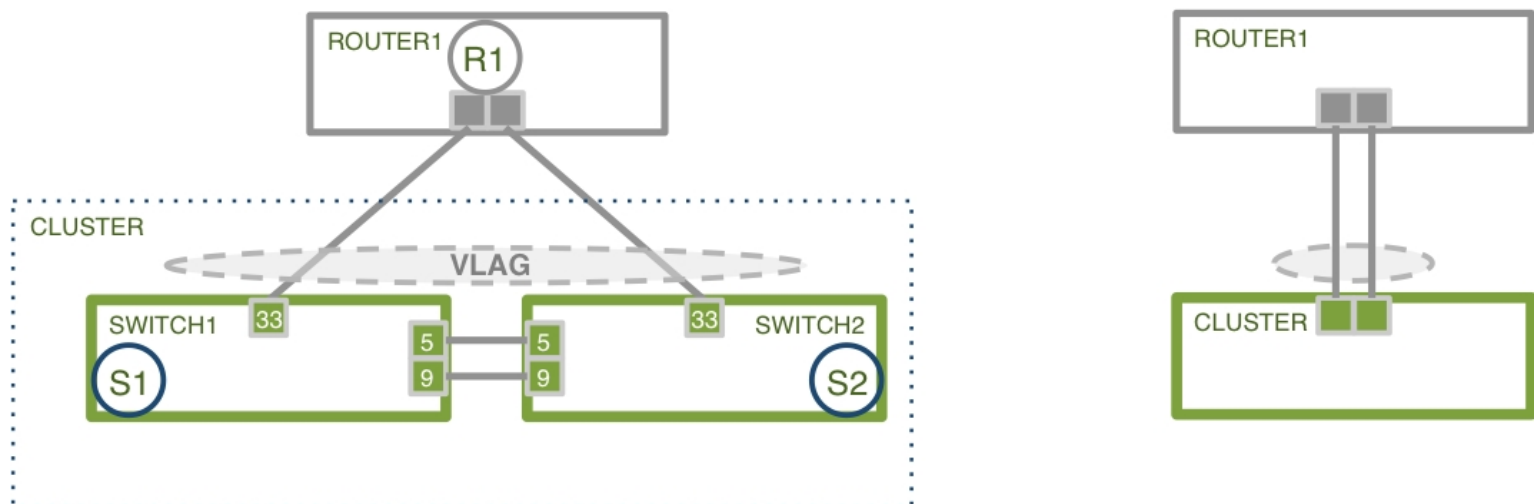
## Understanding LAG Path Selection and Load Balancing

For Layer 2 back-to-back connectivity, Pluribus Netvisor ONE software supports the standard link aggregation technology in order to combine multiple network connections into a logical pipe called a Link Aggregation Group (LAG), or ‘(port) trunk’, which can provide redundancy in case of single or multiple link failure.



**Figure4-1: Two-port LAG Example**

Netvisor ONE also supports link aggregation across two redundant chassis to implement multi-pathing without requiring the creation of Layer 2 loops and the use of the Spanning Tree protocol. This feature is called vLAG (Virtual Link Aggregation Group) on a *switch cluster*.



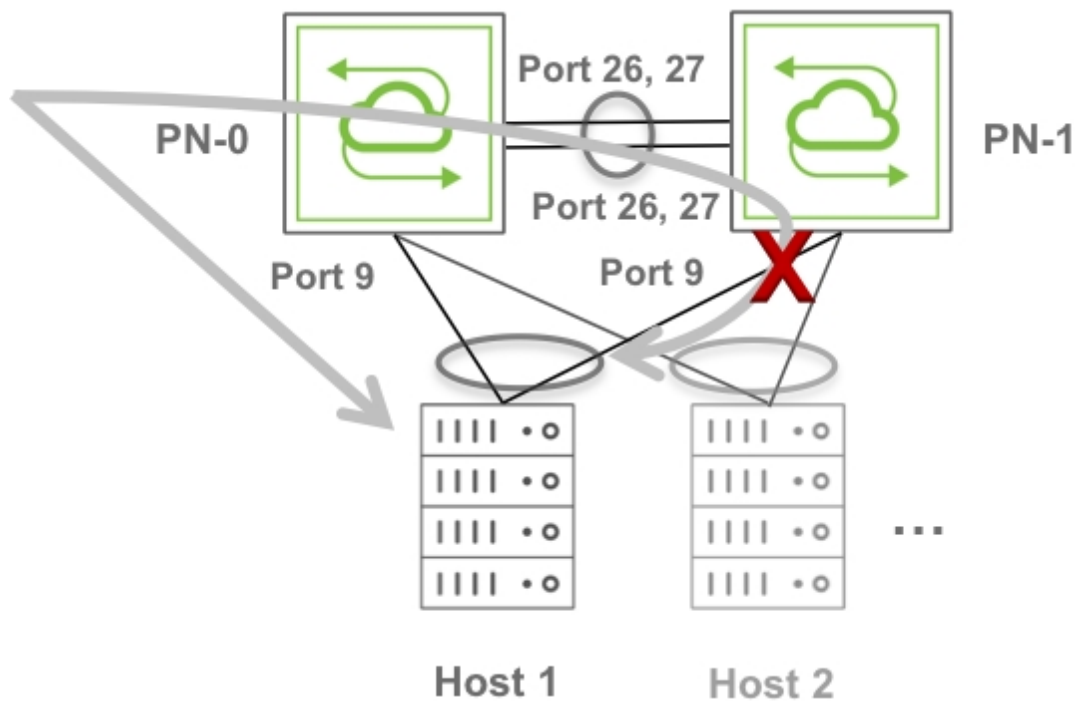
**Figure4-2: Two-port vLAG Example**

Both the aforementioned technologies perform path selection in hardware using a high-performance technology called *packet field hashing*.

What that means is that the hardware extracts a number of packet fields and with them performs a special calculation to generate a *hardware index*. Such index is then used to select an egress physical port for a (v) LAG.

## Understanding the vLAG Forwarding Rule

In order to prevent potential duplication of multicast or broadcast traffic, a forwarding rule is installed on all ports that are part of vLAGs to drop traffic whose ingress point is the cluster links. This rule is shown in the figure below:



**Figure 7-9 - vLAG Forwarding Rule**

The figure above shows a packet entering cluster switch PN-0 that needs to be sent out to Host 1 over the vLAG. To prevent duplication of traffic on the vLAG, any traffic traversing the cluster links and egressing on vLAG ports on cluster switch PN-1 is dropped.

## Configuring Asymmetric Routing over vLAGs

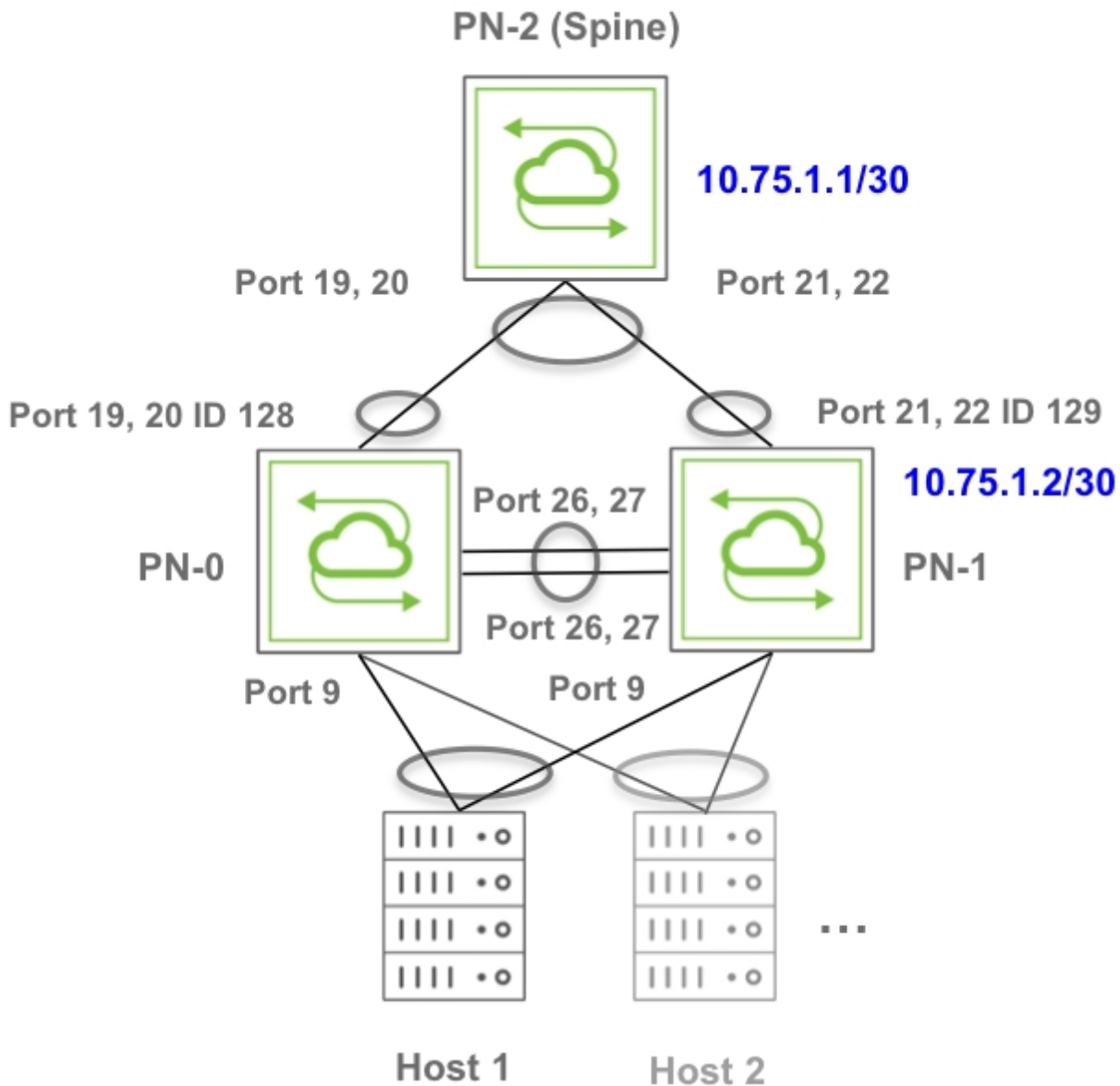
---

vLAGs are symmetric Layer 2 redundant paths that work best with symmetric traffic flows. In these cases the aforementioned vLAG forwarding rule works fine, as discussed in the next section.

In case of Layer 3 forwarding over vLAGs, there can exist certain corner-case network designs in which a routing protocol (e.g., BGP or OSPF) is used to peer over a vLAG in an asymmetric fashion.

Let's consider the example in Figure: Asymmetric Routing over a vLAG in which the network admin has chosen to configure BGP peering only with cluster member PN-1. The downstream hosts are dual-homed and are properly connected to the rest of the network via a default gateway (provided by VRRP).

PN-2 is connected via a vLAG as well to the cluster, however is configured with a /30 subnet that is only present on one cluster member (PN-1 in this example, with PN-2 peering with it via BGP and using the address 10.75.1.1/30).



**Figure 7-10 - Asymmetric Routing over a vLAG**

If Host 1 wants to talk to PN-2's 10.75.1.1 (or to another asymmetric route advertised by it), its traffic can go left or right depending on the vLAG load-balancing algorithm. If it goes right, PN-1 can route it to PN-2. However if the traffic goes left, PN-0 has no route it has learned locally to reach PN-2. Therefore, in the best case it will direct traffic over the cluster link (assuming there is some IGP running over it). However, PN-1 will not be able to send the packet to H1 due to the egress filtering rules enforced on vLAGs (see previous section).

For this reason asymmetric routing configurations should be avoided.

However a knob is provided as a last resort to bypass the cluster egress filtering rules in case the customer still wants to utilize this sub-optimal design, which is not recommended. In general, it is sub-optimal for two main reasons:

- Even when there is no failure (vLAG links are up), traffic is forced to cross the cluster links, thereby

potentially oversubscribing them.

- There is no redundancy at L3 level. When BGP peering to cluster member PN-1 fails, there is no backup path to direct the traffic to.

To enable support of this design a new configuration option is provided to allow packets crossing the cluster link to be routed without being dropped when egressing vLAGs.

To modify the default configuration use the `system-settings-modify routing-over-vlags|no-routing-over-vlags` command like so:

```
CLI (network-admin@switch) > system-settings-modify routing-over-vlags
```

You can use this command to verify the configuration change:

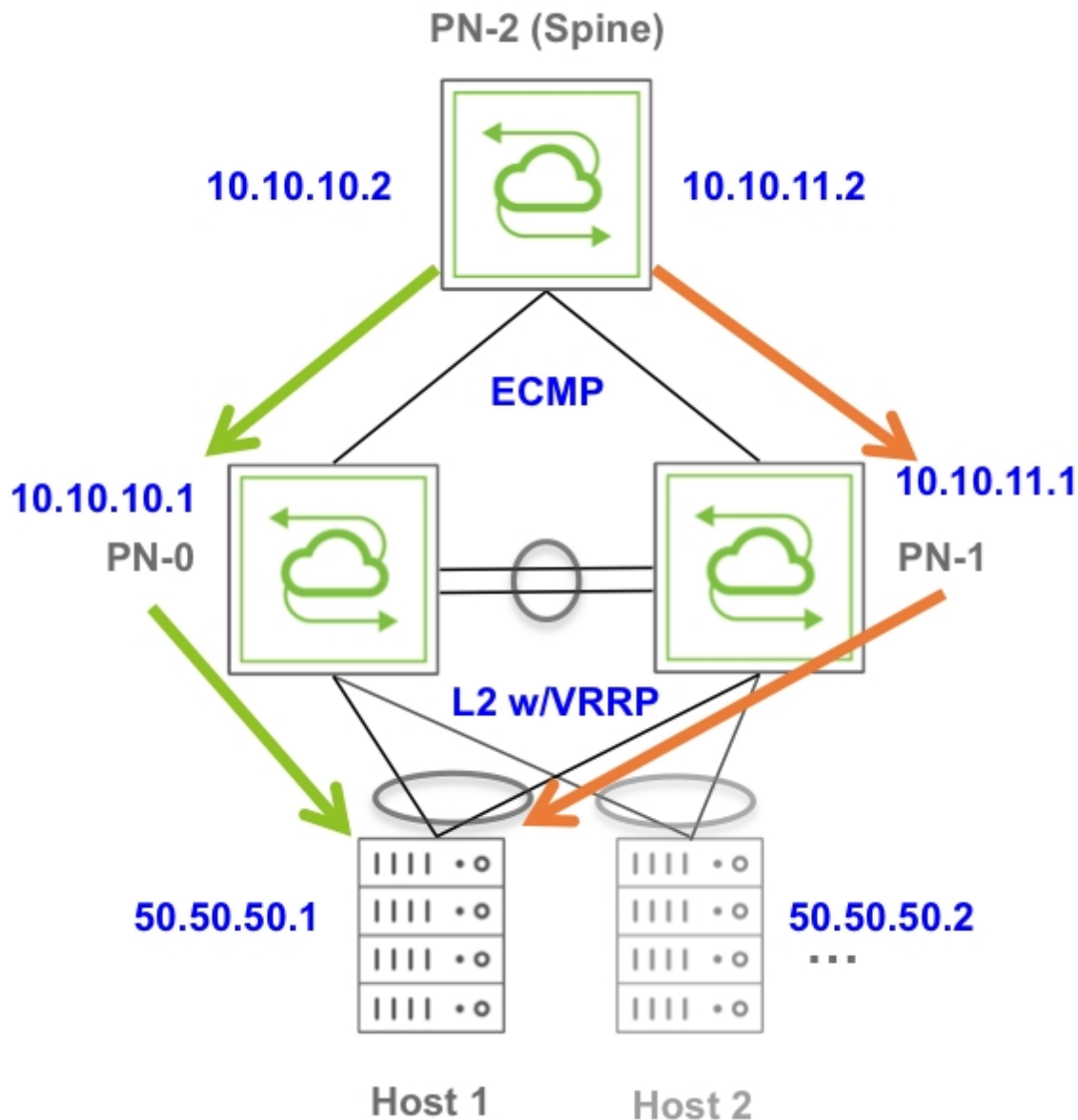
```
CLI (network-admin@switch) > system-settings-show
```

```
switch:                PN-1
routing-over-vlags:    on
switch:                vanquish4
routing-over-vlags:    off
switch:                vanquish3
routing-over-vlags:    off
```

## About Symmetric Routing over vLAGs

As stated above, vLAGs work best with symmetric traffic flows. In order to achieve that with routing, two main designs are recommended: VRRP + ECMP and symmetric VRRP.

The goal of either design is to distribute traffic equally across the different paths both upstream and downstream, making sure that the redundant cluster nodes can steer the traffic to its destination without having to rely upon the cluster links.

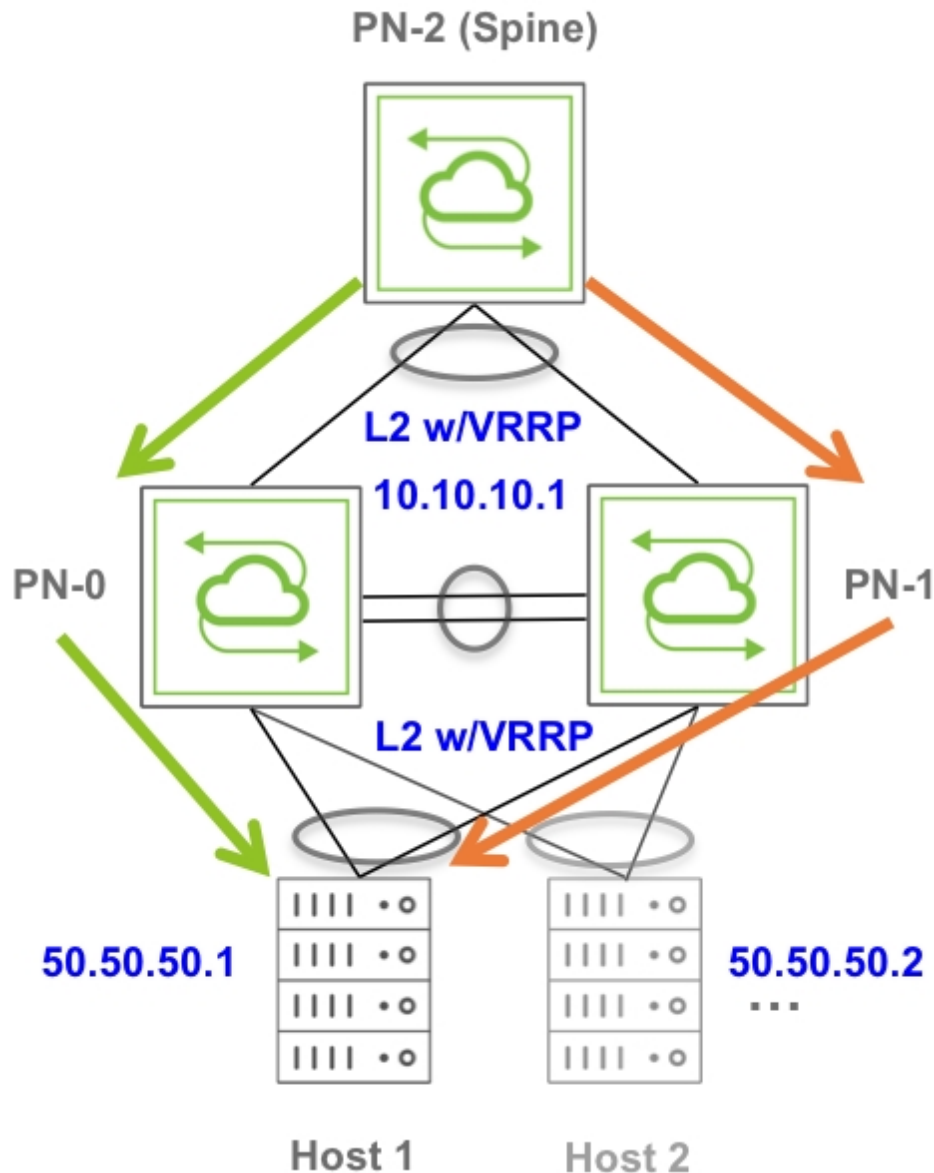


**Figure 7-11 - Symmetric Routing over a vLAG with ECMP and VRRP**

In this figure traffic is, for example, directed to and from host 50.50.50.1: downstream the spine router/switch has ECMP routes toward both cluster nodes, which have Layer 3 adjacencies to all the hosts, so either of them can properly steer traffic to 50.50.50.1. Upstream they implement the active-active

default gateway function via VRRP and support traffic load-balancing with vLAGs from the host(s) toward the spine.

Hence this design implements optimal Layer 3 forwarding both ways (without relying on the cluster links as active paths, only as backups). It also supports running Layer 3 routing protocols on the cluster switches.



**Figure 7-12 - Symmetric Routing over a vLAG with VRRP**

In this other scenario two cluster switches run vRouters with active-active VRRP in order to provide redundant Layer 3 next hops (using virtual IPs) to both upstream and downstream devices.

This design achieves symmetric Layer 3 forwarding purely via vLAG load-balancing and VRRP active-active forwarding. However, note that it does not lend itself to the use of dynamic routing protocols on vRouters because with VRRP routing adjacencies would only form on the vRouter acting as VRRP master, preventing the slave vRouter to process and install routes.

Netvisor ONE supports the active-active dual-forwarding logic by default with VRRP. However, if needed, you can disable it or re-enable it on a per vRouter basis with this command:

```
CLI (network-admin@switch) > vrouter-modify name vRouter-PN-0 cluster-
active-active-routing|no-cluster-active-active-routing
```

To display the configuration, use the vrouter-show command:

```
CLI (network-admin@switch) > vrouter-show format all layout vertical
```

```
switch:                PN-0
id:                    b000f1e:1
name:                  vRouter-PN-0
type:                  vrouter
scope:                 local
vnet:                  test
vnet-service:          dedicated
state:                 enabled
location:              sw45
zone-id:               b000f1e:2
template:              no
failover-action:       stop-old
router-type:           hardware
fabric-comm:           false
router-ipstack:        frr
hw-router-mac:         66:0e:94:1e:7a:6a
cluster-active-active-routing: disable
hw-vrid:               0
hw-vrrp-id:            -1
proto-multi:           none
proto-routing:         static,routesnoop
bgp-redis-static-metric: none
bgp-redis-connected-metric: none
bgp-redis-rip-metric:  none
bgp-redis-ospf-metric: none
bgp-dampening:         false
bgp-scantime(s):       60
bgp-delayed-startup(s): 1
bgp-keepalive-interval(s): 60
bgp-holdtime(s):       180
ospf-redis-static-metric: none
ospf-redis-static-metric-type: 2
ospf-redis-connected-metric: none
ospf-redis-connected-metric-type: 2
ospf-redis-rip-metric:  none
ospf-redis-rip-metric-type: 2
ospf-redis-bgp-metric:  none
ospf-redis-bgp-metric-type: 2
ospf-stub-router-on-startup: false
ospf-bfd-all-if:      no
ospf-default-information: none
```



```
ospf-default-info-originate-metric: none
ospf-default-info-originate-metric-type: 2
bgp-snmp: false
bgp-snmp-notification: false
ospf-snmp: false
ospf-snmp-notification: false
ospf6-snmp: false
ospf6-snmp-notification: false
ip-snmp: false
```

## Configuring Active-Active vLAG Forwarding with Loopback Recirculation

---

In network designs in which vLAG load-balancing with fast failover is required to work in conjunction with dynamic routing protocol peering, the aforementioned vLAG forwarding rule may cause traffic drops.

Therefore, for certain switch models with spare bandwidth an additional configuration option has been implemented.

Let's consider the example in Figure: Symmetric Routing over a vLAG with Loopback Recirculation below where a cluster pair uses standard OSPF peering over a Layer 2 domain represented by a redundant (v)LAG.

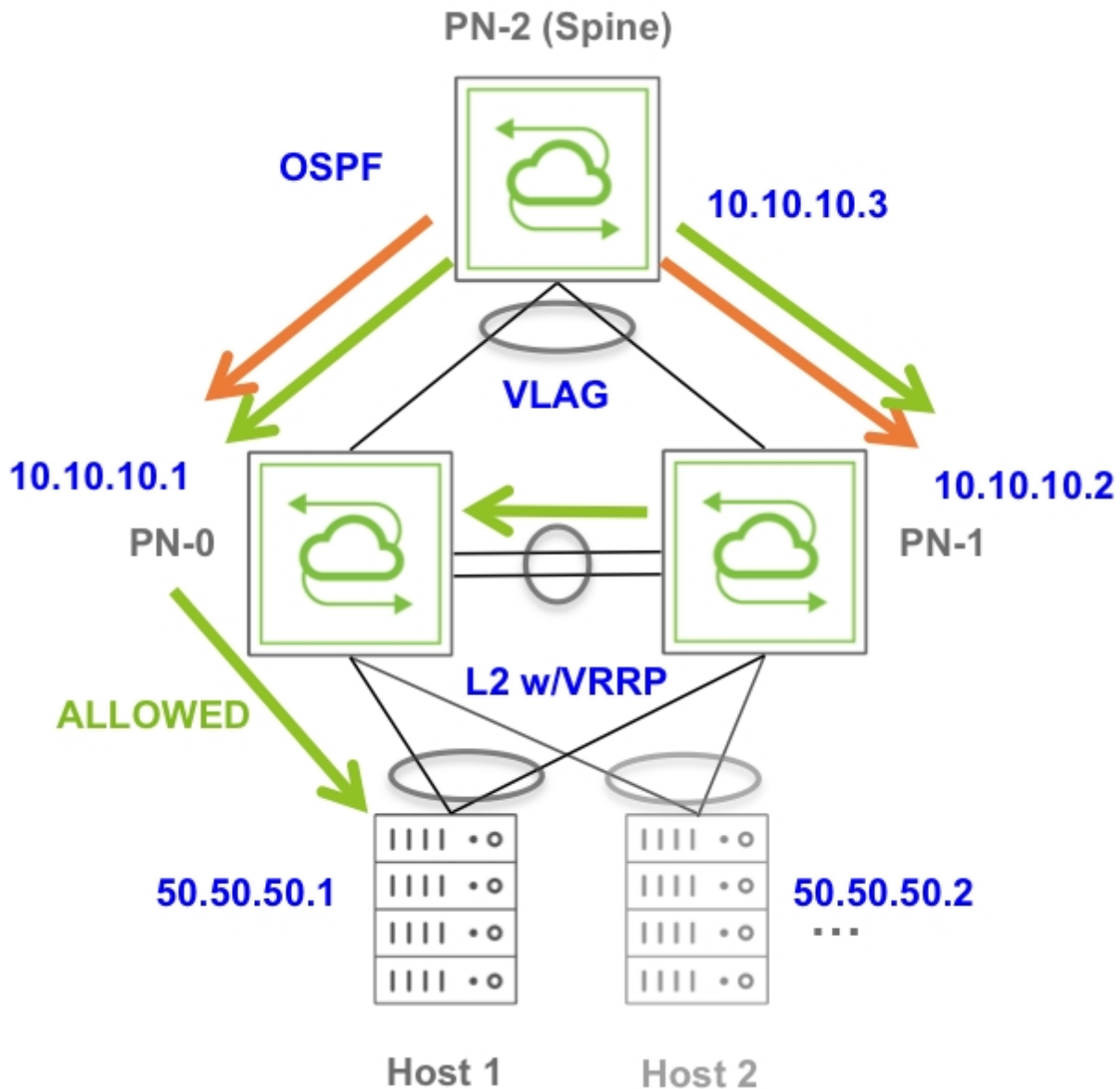
To the routing protocol running on PN-2 the (v)LAG looks like a share medium over which both cluster nodes, 10.10.10.1 and 10.10.10.2, are reachable. However, the (v)LAG performs traffic load-balancing: packets sent toward 10.10.10.1 can be steered to either PN-0 or PN-1; likewise, packets sent toward 10.10.10.2 can be steered to either PN-0 or PN-1. When packets are sent to the “wrong” next-hop, the latter will have to use the cluster links to steer the packets back to their “correct” destination.

However, packets traversing the cluster links that need to egress a vLAG (going downstream to reach one of the hosts) are dropped.

For this case, Netvisor ONE supports the configuration of internal loopback recirculation for routed packets entering the switch from cluster links in order to bypass the vLAG forwarding rule.

The approach consists in provisioning a set of physical ports in loopback mode without requiring external cabling. In particular, the Pluribus E28-Q model offers up to 12x10GE ports (that is, which are not exposed to users in the front panel) that can be configured in loopback mode for an additional forwarding capacity of 120 Gbps.

An internal forwarding rule can be manually installed on both cluster nodes to redirect the traffic to the internal loopback when it ingresses from the cluster links with a destination MAC address matching that of the next-hop. When a packet is recirculated by this rule over the loopback, it gets normally routed and hence is not subject to the vLAG forwarding rule (that would otherwise drop it).



**Figure 7-13 -Symmetric Routing over a vLAG with Loopback Recirculation**

Note that, when this option is configured, approximately 50% of the traffic traverses the cluster links, which is not ideal, hence it is not recommended as a general solution but can be useful in certain designs.

For each vRouter that needs redirection of the traffic to the internal loopback, first find the MAC addresses of the (two, in case of a cluster) corresponding interfaces using the `vrouter-interface-show` command:

```
CLI (network-admin@switch) > vrouter-interface-show vrouter-name UP1
```

```
vrouter-name    UP1
nic:            eth11.200
ip:             200.200.200.1/24
assignment:     static
mac:            66:0e:94:10:29:e1
```

```
vlan:    200
vxlan:   0
if:      data
vm-nic-type:  data
exclusive:  no
nic-config:  enable
nic-state:  up
```

In this case the interface MAC address is 66:0e:94:10:29:e1.

Then configure a loopback (with a single port or a multi-port trunk):

```
CLI (network-admin@switch) > port-config-modify port 89 loopback
```

Or for the trunk case:

```
CLI (network-admin@switch) > port-config-modify port 89-100 loopback
```

```
CLI (network-admin@switch) > trunk-create name loopbackUP ports 89-100
loopback
```

Then create the special forwarding rule using the vFlow command:

```
CLI (network-admin@switch) > vflow-create name LB_vflow scope local in-port
129 dst-mac 66:0e:94:10:29:e1 action set-dmac-to-port action-value 89
action-set-mac-value 66:0e:94:10:29:e1
```

Or for the trunk case:

```
CLI (network-admin@switch)> vflow-create name LB_vflow scope local in-port 129 dst-mac
66:0e:94:10:29:e1 action set-dmac-to-port action-value 130 action-set-mac-value 66:0e:94:10:29:e1
```

For `action-value` use the loopback port number or the loopback trunk ID (130, in this example, assigned during creation).

For a cluster pair, you must configure each node using the above configuration steps.

## Configuring Virtual Router Redundancy Protocol

---

The configuration of the VRRP protocol is tied to the creation and configuration of vRouters. It requires entering various parameters such as a VRRP priority, a VRRP ID to uniquely identify the virtual router, etc.

### Configuring VRRP Priority

The priority is a value used by the VRRP master election process.

The valid priority range is 1-254, where 1 is the lowest priority and 254 is the highest priority.

The default value is 100. Higher values indicate higher priority for the master router election, therefore a backup router (also called a slave) can be configured for example with a priority value lower than the default.

### Configuring the VRRP ID

A virtual router is identified by its virtual router identifier (VRID) and by a set of virtual IPv4 and/or IPv6 address(es).

Each virtual IPvX address is paired to a MAC address in the 00-00-5E-00-01-XX address range where the last byte of the address (XX) corresponds to the VRID.

The VRID is also used to tag and differentiate protocol messages exchanged by VRRP routers.

The virtual router identifier is a user-configurable parameter with a value between 1 and 255. There is no default value.

In the configuration this parameter has to be associated to a vRouter entity and to the VRRP interface, as shown in the example below.

### Example Configuration

In this example two switches, named switch1 and switch2, are going to share a subnet and VLAN over which to set up VRRP's virtual router function (with an ID of 10):

- VLAN 100 with IP address range 192.168.11.0/24

The corresponding vRouters are going to share a common vNET:

- The vrrp-router vNET with scope fabric

To configure VRRP, start with switch1 and create a vRouter that is associated with the aforementioned vNET and a VRRP ID of 10. Before configuring the vrouter-create command, you must create the corresponding vnet:

```
CLI (network-admin@switch) > vrouter-create name vrrp-rtr1 vnet vrrp-router  
router-type hardware hw-vrrp-id 10 enable
```

Add a vRouter interface that corresponds to the router's own real IP address:

```
CLI (network-admin@switch) > vrouter-interface-add vrouter-name vrrp-rtr1  
ip 192.168.11.3 netmask 24 vlan 100 [if data]
```

The above command will output a message such as:

```
Added interface eth0.100 with ifIndex 24
```

You can also use the `vrouter-interface-show` command to check the name of the newly created interface (eth0.100):

```
CLI (network-admin@switch) > vrouter-interface-show
```

```
format all layout vertical  
vrouter-name: vrrp-rtr1  
nic: eth0.100  
ip: 192.168.11.3/24  
assignment: static  
mac: 66:0e:94:dd:18:c4  
vlan: 100  
vxlan: 0  
if: data  
alias-on:  
exclusive: no  
nic-config: enable  
nic-state: up
```

Create the VRRP interface on the master switch with virtual IP 192.168.11.2, VRRP ID 10 and default priority (100):

```
CLI (network-admin@switch1) > vrouter-interface-add vrouter-name vrrp-rtr1  
ip 192.168.11.2 netmask 24 vlan 100 [if data] vrrp-id 10 vrrp-primary  
eth0.100 vrrp-priority 100
```

The above command will output a message such as:

```
Added interface eth1.100 with ifIndex 25
```

Then create a vRouter and an interface (with real IP 192.168.11.4) also on switch2:

```
CLI (network-admin@switch) > vrouter-create name vrrp-rtr2 vnet vrrp-router  
router-type hardware hw-vrrp-id 10 enable
```

```
CLI (network-admin@switch) > vrouter-interface-add vrouter-name vrrp-rtr2  
ip 192.168.11.4 netmask 24 vlan 100 [if data]
```

Use the `vrouter-interface-show` command to check the name of the newly created interface (eth3.100):

```
CLI (network-admin@switch2) > vrouter-interface-show format all layout
```

vertical

```
vrouter-name: vrrp-rtr2
nic: eth3.100
ip: 192.168.11.4/24
assignment: static
mac: 66:0e:94:21:a9:6c
vlan: 100
vxlan: 0
if: data
alias-on:
exclusive: no
nic-config: enable
nic-state: up
```

Create the VRRP interface for the backup switch with the same VIP 192.168.11.2, same VRRP ID (10) and a lower-than-default priority (say, 50):

```
CLI (network-admin@switch2) > vrouter-interface-add vrouter-name vrrp-rtr2
ip 192.168.11.2 netmask 24 vlan 100 [if data] vrrp-id 10 vrrp-primary
eth3.100 vrrp-priority 50
```

Display the information about the VRRP setup:

```
CLI (network-admin@switch2) > vrouter-interface-show format all layout
vertical
```

```
vrouter-name: vrrp-rtr1
nic: eth0.100
ip: 192.168.11.3/24
assignment: static
mac: 66:0e:94:dd:18:c4
vlan: 100
vxlan: 0
if: data
alias-on:
exclusive: no
nic-config: enable
nic-state: up
vrouter-name: vrrp-rtr1
nic: eth1.100
ip: 192.168.11.2/24
assignment: static
mac: 00:00:5e:00:01:0a
vlan: 100
vxlan: 0
if: data
alias-on:
exclusive: no
nic-config: enable
nic-state: up
```

```
vrrp-id: 10
vrrp-primary: eth0.100
vrrp-priority: 100
vrrp-state: master
vrouter-name: vrrp-rtr2
nic: eth3.100
ip: 192.168.11.4/24
assignment: static
mac: 66:0e:94:21:54:07
vlan: 100
vxlan: 0
if: data
alias-on:
exclusive: no
nic-config: enable
nic-state: up
vrouter-name: vrrp-rtr2
nic: eth4.100
ip: 192.168.11.2/24
assignment: static
mac: 00:00:5e:00:01:0a
vlan: 100
vxlan: 0
if: data
alias-on:
exclusive: no
nic-config: enable
nic-state: down
vrrp-id: 10
vrrp-primary: eth3.100
vrrp-priority: 50
vrrp-state: slave
```

When you intentionally disable the master's VRRP interface, the backup interface becomes the new master:

```
vrouter-name: vrrp-router2
nic: eth4.100
ip: 192.168.11.2/24
assignment: static
mac: 00:00:5e:00:01:0a
vlan: 100
vxlan: 0
if: data
alias-on:
exclusive: no
nic-config: enable
nic-state: up
vrrp-id: 10
vrrp-primary: eth3.100
vrrp-priority: 50
```



```
vrrp-state: master
```

When you re-enable the disabled interface on the former master, that interface becomes the master again, and the second interface returns to be a backup (slave):

```
vrouter-name: vrrp-router2  
nic: eth4.100  
ip: 192.168.11.2/24  
assignment: static  
mac: 00:00:5e:00:01:0a  
vlan: 100  
vxlan: 0  
if: data  
alias-on:  
exclusive: no  
nic-config: enable  
nic-state: down  
vrrp-id: 10  
vrrp-primary: eth3.100  
vrrp-priority: 50  
vrrp-state: slave
```

## Troubleshooting High Availability

A trunk (also called LAG or port channel) can be configured automatically or can be defined manually with or without the LACP protocol.

Due to their resiliency and traffic load balancing, trunks can be used for inter-switch communication within a cluster (auto-LAG) or for general network connectivity (user-configured LAG).

You can verify a trunk (LAG) status with the command `trunk-show`:

```
CLI (network-admin@switch) > trunk-show format
switch,name,ports,speed,lacp-mode,status
```

switch	name	ports	speed	lacp-mode	status
-----	-----	-----	-----	-----	-----
pns witch1	ports1-4	1-4	10g	off	up
pns witch1	ports5-8	5-8	10g	off	up
pns witch1	ports9-12	9-12	10g	off	up
pns witch1	ports13-16	13-16	10g	off	up

Trunks can be configured with or without LACP. The following example shows the LACP options available when creating a trunk:

```
CLI (network-admin@switch) > trunk-create name port1-4 ports 1,4 lacp-mode
```

Off	LACP is off
Passive	LACP passive mode
Active	LACP active mode

When LACP is set to active or passive mode it helps detect link and configuration changes, whereas in off mode it is up to the network admin to deal with the trunk bring up process and to avoid configuration mistakes/oversights (such as asymmetries in the configuration).

Clusters and vLAGs provide the underlying redundancy structure for network communications. You can check that a cluster and a vLAG are functioning properly with the following commands. First verify the cluster status (online or offline) with the command `cluster-show`:

```
CLI (network-admin@switch) > cluster-show
```

name	state	cluster-node-1	cluster-node-2	tid	ports	remote-ports
-----	----	-----	-----	---	-----	-----
pnclusterodd	online	pns witch1	pns witch3	15	4,36,128	4,36,129
pnclustereven	online	pns witch2	pns witch4	0	4,8,128	4,8,129

Then verify the vLAG status(es) with the command `vlag-show`:

```
CLI (network-admin@pns witch1) > vlag-show layout vertical
```

```
name:      pnvlag1
cluster:   pnclusterodd
mode:      active-active
switch:    pnswitch1
port:      trunk-to-plus
peer-switch: pnswitch3
peer-port: trunk-to-plus
status:    normal
local-state: enabled
lacp-mode: up  off
```

```
name:      pnvlag2
cluster:   pnclustereven
mode:      active-active
switch:    pnswitch2
port:      49
peer-switch: pnswitch4
peer-port: 18
status:    normal
local-state: enabled
lacp-mode: up  active
```

A vLAG is a logical entity that relies upon its port members (physical ports with an operational Layer 1 status) and upon the underlying cluster.

Therefore, first check that the vLAG status is normal and the state is “enabled,up”.

If there are problems with the vLAG, work back through the objects it depends on the cluster, and ultimately the physical ports and the cables.

## Supported Releases

---

Command/Parameter	Netvisor ONE Version
<i>trunk-create, trunk-modify</i> (with LACP parameters and loopback)	Commands with parameters first supported in version 2.1
<i>cluster-create</i>	Command added in version 1.2
<i>cluster-bringup-modify</i>	Command added in version 2.5.4
<i>l3-to-vlan-interface-delay</i>	Parameter introduced in version 2.6.2
<i>vlag-create, vlag-modify</i> (with parameters for mode <i>lacp-mode</i> , <i>active-standby</i> and <i>active-active</i> )	Commands with parameters first supported in version 2.2
<i>lacp-mode, lacp-fallback, and lacp-fallback-timeout</i>	Parameters introduced in version 2.4
<i>lacp-port-priority</i>	Parameter introduced in version 2.5
<i>hw-vrrp-id</i>	Parameter introduced in version 2.2 for <i>vrouter-create</i>
<i>vrrp-id, vrrp-priority, and vrrp-primary</i>	Parameters introduced in version 2.2 for <i>vrouter-interface-modify</i>

Refer to the Pluribus Networks Command Reference documents for more details on the commands.

## Related Documentation

---

For more information on concepts mentioned in this section (such as Layer 2, Layer 3, and the Fabric), refer to below topics from the Configuration Guide:

- [Configuring Layer 2 Features](#)
- [Configuring Layer 3 Features](#)
- [Configuring and Administering the Pluribus Fabric](#)

## Configuring VXLAN

---

This chapter provides information for understanding and using the Pluribus Networks Netvisor ONE command line interface (CLI) on a Netvisor ONE switch to configure VXLAN.

---

- [Understanding VXLAN](#)
  - [About VXLAN's Packet Format and Its Implications](#)
  - [About Pluribus' VXLAN Implementation and Benefits](#)
  - [About Unicast Fabric VRFs with Anycast Gateway](#)
  - [About IGMP Snooping Support with VXLAN](#)
  - [About Distributed Multicast Forwarding with Fabric VRFs](#)
  - [About Pluribus Open Networking Multi-site Fabric](#)
- [Guidelines and Limitations](#)
- [Configuring the VXLAN Underlay Network](#)
- [Configuring the Overlay: VTEP Interconnections and VNIs](#)
  - [Configuring the VXLAN Loopback Trunk](#)
  - [Checking VXLAN Recirculation's L2 and L3 Entries](#)
  - [Showing VXLAN Trunk Replication Counters](#)
  - [Displaying ECMP Load Balancing Info for VXLAN](#)
  - [Configuring VTEP Objects with Automatic Fabric Connections](#)
  - [Disabling VXLAN Termination](#)

- [Configuring Unicast Fabric VRFs with Anycast Gateway](#)
  - [Configuring IGMP Snooping with VXLAN](#)
  - [Configuring Multicast Fabric VRFs](#)
  - [Supported Releases](#)
  - [Related Documentation](#)
-

## Understanding VXLAN

---

In modern networks, Virtual eXtensible Local Area Network (VXLAN) has become a key technology enabler for advanced data center network designs such as those employed by cloud service providers and large enterprises. RFC 7348 is the IETF *informational* document that describes in detail the VXLAN encapsulation scheme.

Historically, the primary need for the introduction of a new encapsulation scheme originated in the data center where high server density on top of server virtualization placed increased demands on the available physical and logical resources of the network.

In particular, each virtual machine (VM) requires its own Media Access Control (MAC) address. Therefore, highly scalable data centers with N VMs running on M servers would require MAC address tables potentially larger ( $N \times M$ ) than what is available in a switched Ethernet network.

In addition, bare metal servers as well as VMs in a data center need to be grouped and isolated according to various management and security policies: this is usually achieved by assigning them to Virtual LANs (VLANs). However, the maximum number of VLANs is limited to 4094 and hence it's inadequate for the largest network designs, especially when *multi-tenancy* and *VLAN reuse* are required.

In fact, data centers are often required to host multiple tenants, which must have their own isolated management and data forwarding domains. This means that each tenant should be able to independently assign MAC addresses and VLAN IDs without causing resource conflicts in the physical network. In order to achieve this, an extra *layer of network virtualization* is required.

While various solutions for each aforementioned resource limitation/challenge have been proposed in the past, thanks to its flexibility and ample software and hardware support VXLAN has risen above all alternatives to become the *solution of choice* to address all of the above limitations. Moreover, its powerful UDP-based encapsulation has been leveraged to enable novel capabilities as well as to replace legacy solutions for highly scalable network designs (as further discussed in the paragraphs below).

One important use case is when virtualized environments require having Layer 2 forwarding capabilities scale across the entire data center or even between different data centers for efficient allocation of compute, network and storage resources. A solution that enables a network to scale across data centers is oftentimes referred to as *Data Center Interconnect (DCI)* and is an important high availability design.

In the DCI scenario traditional approaches that use the Spanning Tree Protocol (STP) or an MPLS-based technology (such as VPLS) for a loop-free topology are not always optimal or flexible enough. In particular, network designers typically prefer to use an *IP transport for the interconnection, redundancy and load balancing of resources and traffic*.

Nonetheless, despite the preference for a Layer 3-based interconnect, in VM-based environments there is often a need to preserve Layer 2-based connectivity for inter-VM communication. How can both models coexist then?

VXLAN is the answer to this question: it employs a UDP-based header format which can be used to route traffic within a physical Layer 3 network (also called an *underlay*) while its encapsulation capabilities can seamlessly *preserve and augment* Layer 2's inherent characteristics (such as MAC addresses and VLANs)



and communication rules.

From this point of view VXLAN marries the best of both worlds. Hence it can be used to implement a virtual Layer 2 network (a so-called *Layer 2 overlay*) on top of a traditional Layer 3 network design.

This capability is also an ideal fit for example for DCI deployments where a Layer 2 overlay is required to carry the Ethernet traffic to and from geographically dispersed VMs.

The approach of transporting Layer 2 traffic in a UDP-encapsulated format is akin to a logical 'tunneling function'.

However, VXLAN's informational RFC does not enforce any particular control-plane and data-plane scheme (like a tunnel-like model), nor does it limit the number of possible use cases. (It simply illustrates how to overcome certain limitations while suggesting one communication scheme for unicast traffic and one for multicast/broadcast/unknown unicast traffic).

In practical terms, then, VXLAN is primarily an encapsulation format. The way the control plane uses it varies from vendor to vendor, sometimes with fully proprietary implementations and sometimes with open software components based on common interoperability standards.

Despite this heterogeneity and flexibility of implementation, the VXLAN format has become widely popular: it is supported in hardware by most data center switches and it is also supported in software on virtualized servers running for example an open virtual switch.

Most hardware vendors however don't implement verbatim the entire RFC in particular with regard to the 'flood and learn' data plane mechanism described in it, because deemed not scalable enough.

The following sections will describe Pluribus Networks' innovative VXLAN integration with the *Adaptive Cloud Fabric* (ACF, or simply the *fabric*), the related optimizations and use cases, and their configuration.

## About VXLAN's Packet Format and Its Implications

From a pure encapsulation perspective, the VXLAN format has a few important general implications related to the specific fields that it adds as well as to the UDP encapsulation.

The VXLAN header is shown in **Figure 8-1** below.

### VXLAN Header:

```

+---+---+---+---+---+---+---+---+---+---+---+---+---+---+---+---+---+---+---+---+---+---+
|R|R|R|R|I|R|R|R|                               Reserved                               |
+---+---+---+---+---+---+---+---+---+---+---+---+---+---+---+---+---+---+---+---+---+---+
|                               VXLAN Network Identifier (VNI) |       Reserved       |
+---+---+---+---+---+---+---+---+---+---+---+---+---+---+---+---+---+---+---+---+---+---+
  
```

### Inner Ethernet Header:

```

+---+---+---+---+---+---+---+---+---+---+---+---+---+---+---+---+---+---+---+---+---+---+
|                               Inner Destination MAC Address                               |
+---+---+---+---+---+---+---+---+---+---+---+---+---+---+---+---+---+---+---+---+---+---+
| Inner Destination MAC Address | Inner Source MAC Address |
+---+---+---+---+---+---+---+---+---+---+---+---+---+---+---+---+---+---+---+---+---+---+
|                               Inner Source MAC Address                               |
+---+---+---+---+---+---+---+---+---+---+---+---+---+---+---+---+---+---+---+---+---+---+
| OptnlEthtype = C-Tag 802.1Q | Inner.VLAN Tag Information |
+---+---+---+---+---+---+---+---+---+---+---+---+---+---+---+---+---+---+---+---+---+---+
  
```

### Payload:

```

+---+---+---+---+---+---+---+---+---+---+---+---+---+---+---+---+---+---+---+---+---+---+
| Ethertype of Original Payload |
+---+---+---+---+---+---+---+---+---+---+---+---+---+---+---+---+---+---+---+---+---+---+
|                               Original Ethernet Payload                               |
|
| (Note that the original Ethernet Frame's FCS is not included) |
+---+---+---+---+---+---+---+---+---+---+---+---+---+---+---+---+---+---+---+---+---+---+
  
```

### Frame Check Sequence:

```

+---+---+---+---+---+---+---+---+---+---+---+---+---+---+---+---+---+---+---+---+---+---+
| New FCS (Frame Check Sequence) for Outer Ethernet Frame |
+---+---+---+---+---+---+---+---+---+---+---+---+---+---+---+---+---+---+---+---+---+---+
  
```

**Figure 8-1: VXLAN Header Added on Top of a Plain UDP Packet (Ethernet + IP + UDP)**

VXLAN requires the addition of a variable number of extra octets (i.e., overhead) to all the frames it encapsulates: as shown above, Inner Ethernet Header + Payload + FCS represent the Ethernet frame that originates in the underlay network from a host (virtualized or not). Such original frame is then encapsulated into an 8-octet VXLAN header, which is then encapsulated into a regular UDP (+ IP + 802.3) packet. An outer IEEE 802.1Q VLAN tag is optional and, when present, adds *four additional octets* to the total.

For example, in case of UDP over IPv4, this equates to a total of 50/54 additional octets of VXLAN encapsulation overhead. In case of UDP over IPv6 an additional 20 octets (compared to IPv4's case) have to be added due to the longer IPv6 header (for 70/74 bytes of total extra overhead).

## About the MTU Configuration

The first obvious implication is that, in order to make sure that variable-length VXLAN encapsulated packets can be successfully carried across a generic Ethernet underlay network, the *Maximum Transmission Unit (MTU)* size must be properly increased end-to-end on all the physical and logical interfaces connecting the network nodes.

In the IPv4 case the MTU needs to be raised to at least *1554 octets* (from the standard 1500 value). In the IPv6 case it needs to be set to at least *1574 octets*.

In an Ethernet network, usually this requirement is supported for both cases by enabling *jumbo frame forwarding* on the device interfaces (at Layer 2 and 3) that have to carry any VXLAN traffic.

## About the VXLAN IDs

The second important implication of the VXLAN packet format is that it includes a very useful 24-bit field called *VXLAN Network Identifier (VNI)*. Per the definition in RFC 7348:

*VXLAN Segment ID/VXLAN Network Identifier (VNI): this is a 24-bit value used to designate the individual VXLAN overlay network on which the communicating VMs are situated. VMs in different VXLAN overlay networks cannot communicate with each other.*

Traditional network segmentation at Layer 2 is implemented with IEEE 802.1Q VLANs to provide logical segmentation of the traffic in different broadcast domains. The use of VLANs, however, imposes a number of limiting factors, especially in large network designs, for example in large multi-tenant data centers.

By leveraging the VNI field as well as a number of VXLAN-based forwarding optimizations it is possible to provide the same Ethernet Layer 2 network services as VLANs but with greater extensibility and flexibility.

In particular, the 24-bit VNI field supports 16 million possible IDs, which is usually considered more than sufficient for the largest network designs.

When compared to VLANs, VNIs offer the following benefits:

- *Increased scalability* to address more Layer 2 segments as it is possible to map such segments to up to 16 million IDs that can coexist in the same administrative domain.

- *Flexible allocation* of segment IDs in multi-tenant environments, as it is possible to map each VLAN ID to different unique VNIs (that is, VLAN ID reuse is easily supported). Furthermore, VXLAN's UDP transport enables the *flexible interconnect* of data center pods across a generic IP underlay, even when the pods are geographically distributed for redundancy and load sharing purposes.
- *Standard routing technologies* (Layer 3 forwarding and equal-cost multipath (ECMP) load-balancing) are applied to VNI-based segments in the underlying IP network thanks to VXLAN's UDP-based encapsulation. This can translate into better network utilization, simpler interoperability in heterogeneous networks and greater scalability enabled by *hardware-based forwarding and replication*.

## About VXLAN's Layer 2 Extension

The third important implication of the VXLAN packet format is that it can be used to transport Layer 2 traffic from a source to a destination where both source and destination use a single MAC address each (or a limited number of addresses). That is why VXLAN is often likened to a 'tunnel' from a source to a destination that carries traffic from any number of end devices.

In more general terms, as we will see in the following, it's a *versatile Layer 3 forwarding path* from a source to a destination.

Both source and destination are called *VXLAN Tunnel End Points* (VTEPs) per the RFC.

One important challenge in today's virtualized environments is that there is increased demand on MAC address tables of switches that connect to servers. Instead of learning one MAC address per physical server, the switch now has to learn the MAC addresses of the individual VMs, and if the MAC address table overflows, the switch will stop learning new MAC addresses until idle entries age out.

With VXLAN's *end points using a single MAC address*, only that address gets exposed to and learned by adjacent transit switches. Therefore, there is no requirement for global learning of all MAC addresses anymore (like in a flat Layer 2 network), and *MAC address scalability* is only limited by the table size of the switch where the VTEP is configured.

Furthermore, the VNI identifies the scope of the inner MAC frame originated from each host. Therefore, it is possible to utilize overlapping MAC addresses across segments without conflict since, just like with VLANs, the traffic is logically isolated by using different identifiers (the VNIs in case of VXLAN segments).

The aforementioned advantageous implications bring even more benefits to network designers when combined with Pluribus Network's innovative *Adaptive Cloud Fabric* and its advanced feature set.

In the following sections we will describe Pluribus' distinctive implementation of the VXLAN technology, its forwarding optimizations and benefits, as well as its specific configuration steps.

## About Pluribus' VXLAN Implementation and Benefits

---

If a network designer were tasked with selecting the best and most desirable technology among a number of implementation options, he or she would be hard pressed not to go with an open and standard approach. That is simply because a standard technology can maximize interoperability and hence offer network designers a wealth of software and hardware choices.

In contrast to other closed proprietary approaches, which require significant investments in complex and rigid hardware, Pluribus Networks has chosen the user-friendly path of a *software-defined networking technology* based on the VXLAN industry standard, where the fabric and its features seamlessly integrate with the advanced capabilities enabled by the VXLAN transport. (For more details on Pluribus' fabric configuration, refer to the *Setting up and Administering the Pluribus Fabric* chapter.)

In particular, Pluribus has integrated the VXLAN implementation with the fabric's capability of distributed MAC learning and VLAN transport, its redundant path support through vLAGs, virtual addresses and vRouters, as well as with the fabric VRFs' distributed forwarding capability for maximum redundancy and scalability.

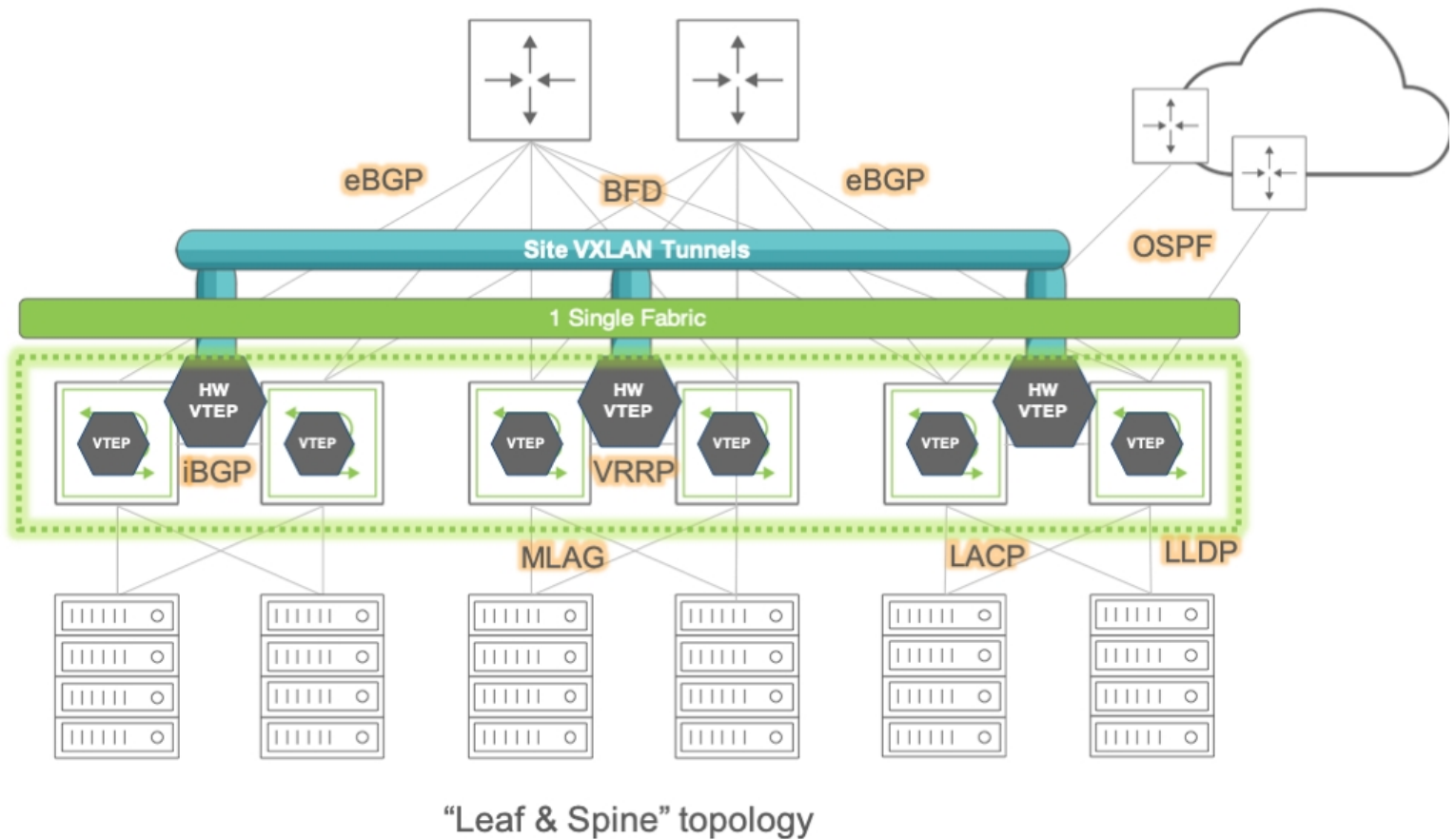
The power of the *Adaptive Cloud Fabric* distributed control plane has been leveraged to greatly augment and simplify VXLAN's basic capabilities without requiring complex proprietary (and potentially not interoperable) enhancements. *Fabric VRFs and distributed multicast forwarding* (described later in this section) are two examples of the powerful synergies that have been achieved.

As part of the Adaptive Cloud Fabric, two inter-related logical layers have to be designed and configured: the underlay and the overlay network.

Let's look at each one individually.

### About the Underlay Network's Architecture

For flexibility and openness reasons, Pluribus engineers have decided to support a spectrum of different options based on *standard protocols* for fabric underlay architectures: they can be built using Layer 2 only, or with a combination of Layer 2 and Layer 3, or only with Layer 3, as shown in **Figure 8-2** below.



**Figure 8-2: Pluribus Fabric Over Any Underlay Network**

As depicted above, network underlays typically include both path and device redundancy.

As explained in the *Configuring High Availability* chapter, either with Layer 2 or with Layer 3 designs Pluribus supports switch clustering and vLAGs in order to minimize reliance on the Spanning Tree Protocol (in case of Layer 2 networks) as well as to optimize redundancy and failover speed.

For Layer 3 underlay networks Pluribus also supports standard routing protocols for device interconnection (such as OSPF and BGP) as well as the VRRP protocol when default gateway redundancy is required.

In particular, having the possibility of leveraging an IP-based routed transport for intra-DC and inter-DC interconnection makes the entire architecture more open and leaner, with the option to use any switch vendor in the core (since any vendor nowadays can provide a broad, inter-operable and robust support of Layer 3 technologies).

In summary, for the network underlay the designer will have to decide which topology and forwarding scheme to use, which devices to deploy in each network location, and, as explained earlier, will have to enable jumbo frames wherever the VXLAN transport is required (typically it is required end-to-end for pervasive deployment models).

Pluribus Networks' open networking switches can be used as both leaf and spine switches, and can be connected to 3rd party switches, routers and appliances using just the aforementioned standard underlay technologies (whose configuration is described in detail in other chapters of this guide).

## About the VXLAN Fabric Overlay and End Points

Regarding the VXLAN transport's architecture and configuration, network designers have to decide where to place the VXLAN end points, i.e., the sources and the destinations of the VXLAN forwarding paths (referred to as *tunnels* in the RFC).

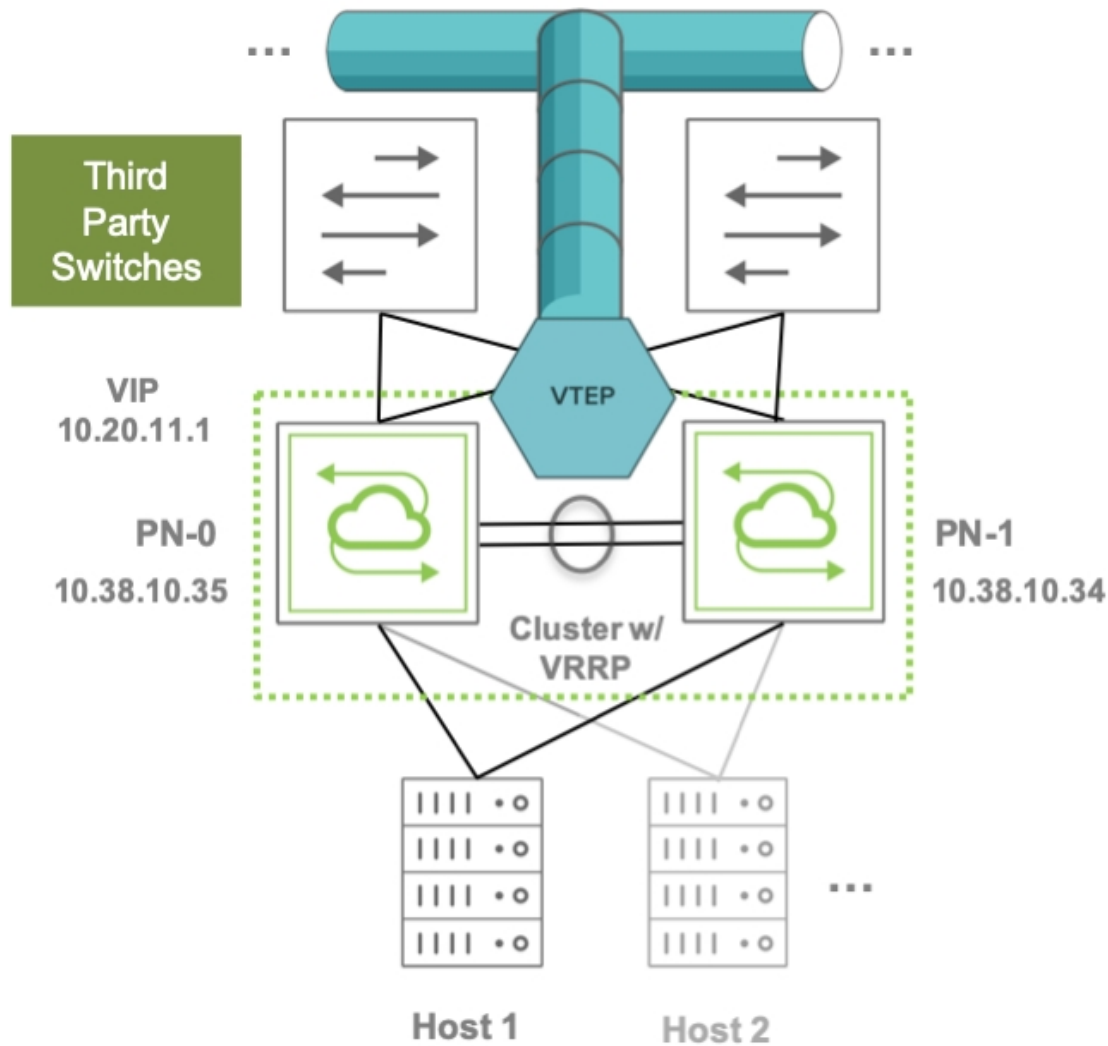
This decision depends on many different factors (such as level of software and hardware support, scalability, virtualization, traffic patterns, etc.) but the choice usually falls on the edge nodes of the fabric (whether hardware or software-based).

Redundancy can also be a key decision factor, as discussed in the next section.

A common scalable choice for end point (VTEP) placement is represented by the leaf switch(es), that is, the first-hop switches physically connected to the hosts (bare metal servers or VMs).

Furthermore, when configured in redundant pairs, leaf switches are the ideal locations for VTEP placement for improved service continuity, as shown in the figure below. (On the other hand, it is also possible that in certain other designs the VTEPs can be placed on spine switches, if so required by design prerequisites and/or network constraints.)





**Figure 8-3: VTEP Placement on Leaf Switch Clusters**

Pluribus switches support flexible VTEP placement. For example, VTEPs can be placed on pairs of Pluribus leaf switches that are connected to a third-party Layer 3 core network (as shown in the figure above), or they can be placed on pairs of Layer 3 Pluribus spine switches that interconnect two DC sites (or pods) over a generic IP-based network, or the design choice can even be a combination of the two above.

In any of these cases, the VXLAN traffic flows are terminated on the switches, which are responsible for encapsulating packets that arrive from servers or hosts on a Layer 2 network and transmitting them over to their destination. Similarly, the VXLAN packets arriving from the opposite direction are decapsulated and the inner packets are forwarded over to their destination end devices. The switches also collect traffic statistics, optimize ARP requests and MAC learning.

In addition to the above, Pluribus also supports scenarios in which VXLAN is not terminated on the switch itself, that is, in which the switch does not participate in the encapsulation and/or decapsulation of the VXLAN traffic.

In such case a host would typically implement a VTEP in software and therefore the switch would act mainly as an underlay node for the pass-through VXLAN traffic directed to that host.



On that VXLAN traffic, though, Netvisor ONE can still perform an important function:

Analytics collection: All TCP control packets are captured as well as ARP packets traversing the tunnel. These packets are used to build connection statistics and provide visibility as to which VXLAN nodes are on specific ports.

## About VTEP High Availability

A critical design requirement for various DC networks is to deploy the VXLAN transport in conjunction with a *high availability (HA) configuration* at the edge of the fabric. This guarantees path redundancy and service continuity in case of link or device failure at the edge.

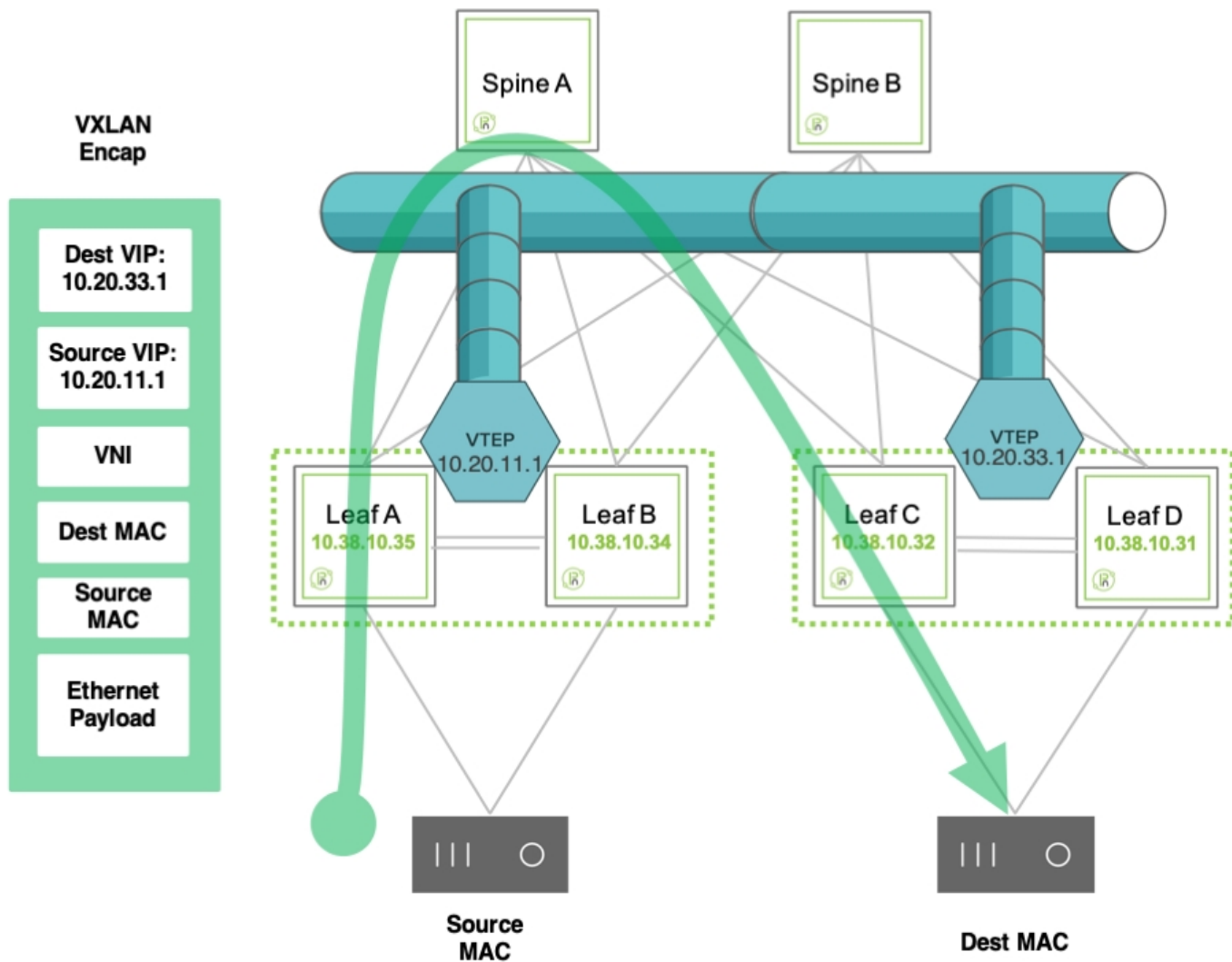
In VXLAN parlance edge redundancy support is known as *VXLAN Tunnel Endpoint High Availability (VTEP HA)*.

Pluribus switches support VTEP HA through the combination of multiple features: in particular, Pluribus implements a VTEP HA switch pair by harnessing the power of switch clustering in conjunction with the vLAG technology and with the standard VRRP protocol (these features are described in detail in the *Configuring High Availability* chapter).

This feature combination enables the configuration of redundant active-active Layer 3 switches that function as VRRP peers (with mutual liveness checks) and that can provide a common *virtual IP (VIP)* address as active-active default gateway.

Thanks to the cluster synchronization function, a VTEP pair can act as a single logical VXLAN end point using a single shared VIP as source address. Similarly, a destination VXLAN end point can be reached by using its HA pair's common VIP as destination address.

This interconnection of virtual IPs is exemplified in the figure below.



**Figure 8-4: VIP Use for Inter-VTEP Communication**

Setting up multiple of the above logical pairs enables the creation of an overlay network of *interconnected VXLAN end points* based on virtual (not physical) addresses which offers *embedded physical as well as logical redundancy*.

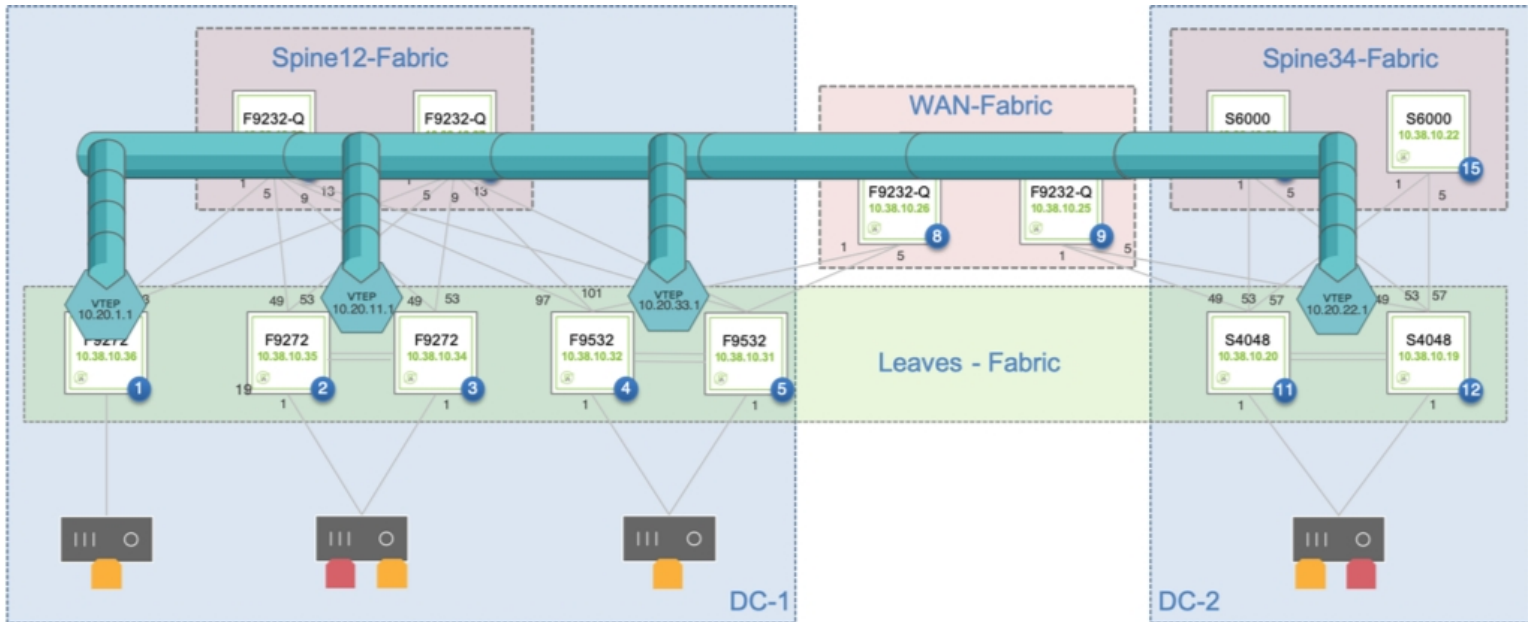
Configuration of these VXLAN pairs can be performed manually for each individual source and destination VTEP. However, Pluribus can further leverage the power of the Adaptive Cloud Fabric's distributed control plane to automate this tedious and potentially error prone process (for more details refer to the VTEP configuration section below).

Similarly, configuration of VLAN extensions over a VXLAN fabric requires a standard VLAN ID to VXLAN Network ID (VNI) mapping scheme. This ID mapping process too can be repetitive and error-prone, but with Pluribus' distributed control plane such global configuration can be propagated to all the nodes belonging to the same fabric instance, thus saving significant time and effort.

## About the VXLAN Fabric Overlay and its Forwarding Optimizations

A fabric overlay comprises a number of VXLAN end point pairs, as mentioned above, whose interconnection is used to transport end stations' traffic end-to-end based on their VLAN-to-VNI mappings.

A typical example of the logical overlay interconnect (of leaf switches in this instance) that VXLAN can be used to create over a physical underlay network is depicted in the figure below:



**Figure 8-5: Overlay Network Interconnecting Single and Redundant VTEPs**

Pluribus' forwarding logic and distributed control plane employ the standard *split horizon algorithm* to create a loop-free logical forwarding topology in the fabric overlay network so as to steer any VXLAN traffic over it without creating potential switching loops.

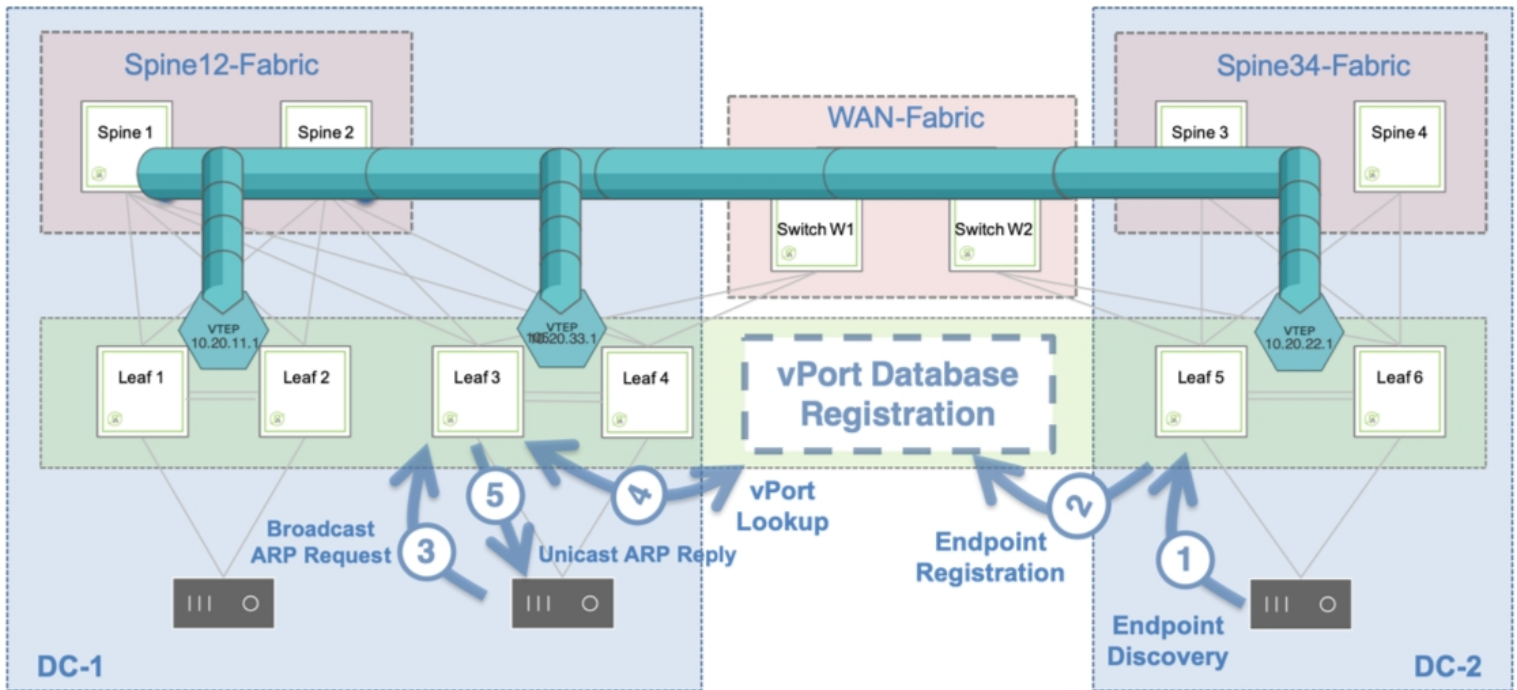
Furthermore, the implementation of the forwarding logic does not require complex address tracking and propagation mechanisms (such as IETF's MP-BGP-based EVPN solution). Pluribus instead uses its own *native address* registration and tracking technology, called *vPort database*, which is particularly useful to track host moves.

While a plain-vanilla hardware Layer 2 table is limited in its capacity by a switch's dedicated ASIC memory size, the Netvisor ONE software runs in the control plane processor's much larger DRAM memory space and hence is capable of tracking a large list of Layer 2 entries, much larger than what can fit into the limited space of a hardware table.

This logical software extension of the Layer 2 table is the vPort database, which can be considered as the *authoritative distributed endpoint directory and switching activity history book* of the entire Pluribus fabric.

In addition, the fabric uses vPort information in conjunction with the *ARP optimization technology* to help manage the dynamic process of binding MAC/IP address pairs.

Pluribus' ARP optimization technology harnesses the versatility of the vPort database to perform the dynamic endpoint discovery and ARP proxy functions that are required to support the mobility of end-points across the fabric, as described in the figure below.



**Figure 8-6: ARP Optimization Process**

In the figure above, on the right-hand side a host is discovered through the traffic exchange with its neighboring switch. As a consequence, its information is recorded in the vPort database.

On a remote switch (on the left-hand side of the figure) whenever someone sends a broadcast ARP request to learn about the MAC address associated with the IP address of the newly discovered host, an *ARP suppression function* is executed.

In other words, the ARP response comes *proxied* from the fabric based on the contents of the distributed vPort database, which is available on all fabric nodes. Therefore, there is no need for any direct ARP communication between requester and registered host, nor for any flooding of broadcast ARP packets.

Moreover, starting from release 2.6.0, Pluribus has added a further under-the-hood enhancement to the fabric's control plane logic that significantly reduces the need for the less-than-optimal 'flood and learn' scheme for unknown destination traffic.

This enhancement, called *vPort forwarding*, implements a more optimized forwarding logic thanks to the versatility of the vPort database: this logic is analogous to ARP optimization's use of the vPort database information to avoid unnecessary ARP flooding, but it's applied to the general case when the destination Layer 2 address of a generic packet can be successfully looked up in the distributed vPort database so as to avoid having to flood the packet to all its possible destinations.

## About ECMP Load Distribution for VXLAN Traffic

Equal-cost multi-path routing (ECMP) is a routing strategy where next-hop packet forwarding can occur over multiple paths on a per flow basis.

Hardware platforms usually perform a hash function on a number of packet fields to perform path selection and determine the load balancing traffic distribution.

In case of VXLAN-encapsulated traffic, for backward-compatibility purposes, the same *legacy* ECMP algorithm can be applied to VXLAN packets too without requiring deeper inspection.

This is because the VXLAN RFC recommends that the encapsulation function add some amount of *entropy* to a VXLAN packet by generating a variable source UDP port:

*Source Port: It is recommended that the UDP source port number be calculated using a hash of fields from the inner packet -- one example being a hash of the inner Ethernet frame's headers. This is to enable a level of entropy for the ECMP/load-balancing of the VM-to-VM traffic across the VXLAN overlay.*

In other words, that recommendation makes sure that some amount of variability is present when creating VXLAN connections so that they can be easily load-balanced using most ECMP functions implemented by networking platforms.

In particular, any ECMP algorithm that takes into account (also) the *Layer 4 source port* for path determination will be able to take advantage of the level of entropy added to the VXLAN packets as per the RFC.

In case of Netvisor ONE this is achieved *by default* by implementing a granular *7-tuple* hashing scheme that uses the following fields: source IP address, destination IP address, VLAN number, destination Layer 4 port, *source Layer 4 port*, IP protocol number, source physical port.

Other networking devices may use a simpler ECMP configuration referred to as *5-tuple hashing*, which includes just the source and destination IP addresses, the protocol type, and the source and destination Layer 4 ports for UDP or TCP flows.

Adding VXLAN traffic entropy and applying ECMP hashing are basic hardware functions supported by default on Pluribus switches. These functions are supported by most 3rd party switches and routers as well.

See also the *Displaying ECMP Load Balancing Info for VXLAN* section.

## About VXLAN Routing In and Out of Tunnels (RIOT)

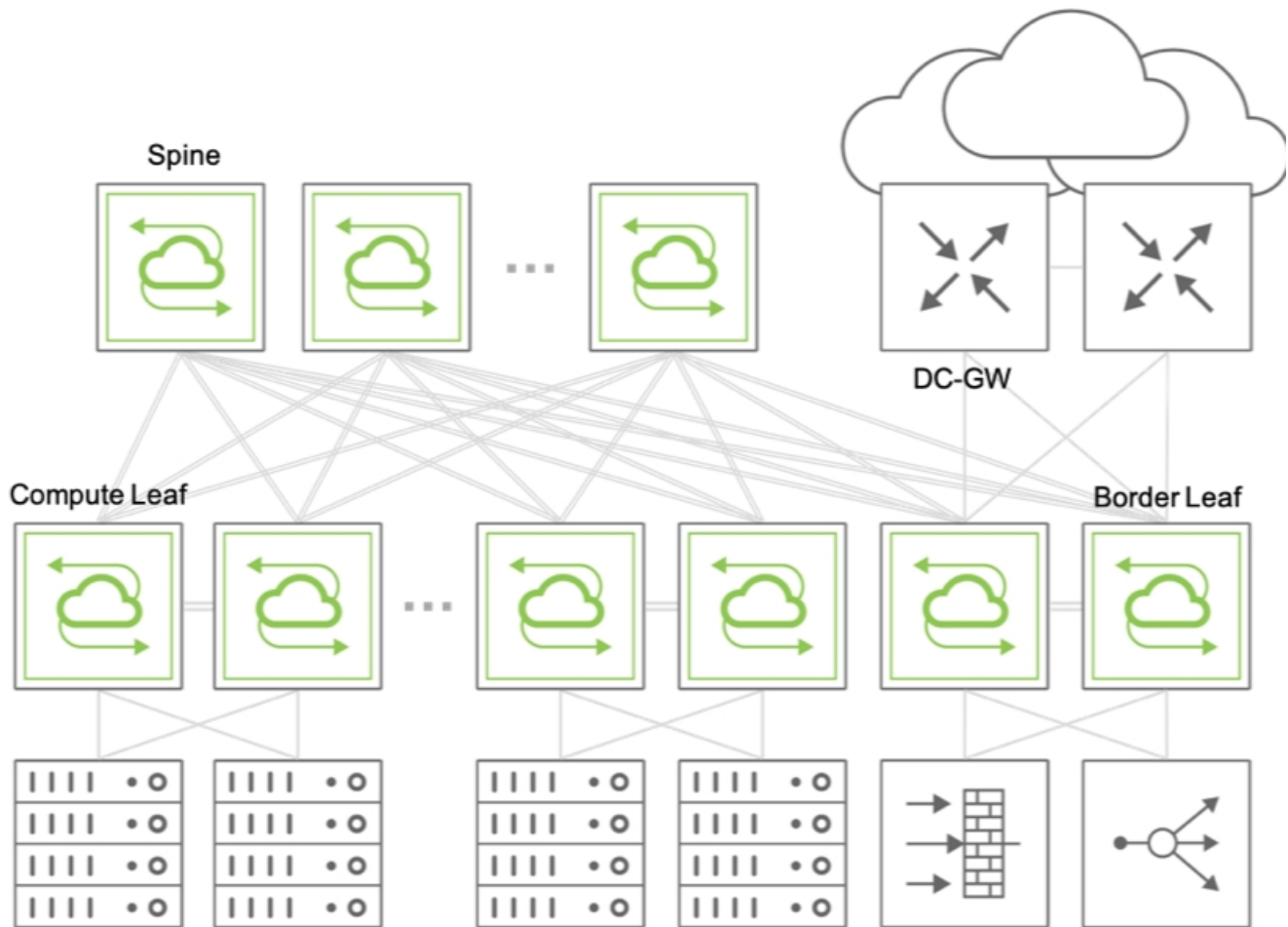
In most designs the VXLAN overlay network requires that overlay traffic be routed as well as bridged.

This is typically required for the host traffic to be routed across VNI-mapped VLANs and/or to reach the data center gateways, firewalls (or other L3/L4 appliances), so as to be forwarded to the Internet. The reverse path from the Internet also requires that the traffic be routed as well to reach the hosts.

This model is sometimes referred to as *centralized routing* for the VXLAN fabric (as opposed to the distributed model implemented with VRFs, as explained in a subsequent section). Pluribus also calls this model RIOT, which stands for *Routing In and Out of Tunnels*.

For hosts on different VLANs to communicate with each other and with the Internet, a cluster of *border leaf switches* is deployed for redundancy and load splitting purposes to become the overlay network's default

gateway. VLAN interfaces are configured as required by the network administrator on the cluster's vRouters (up to 32 vRouters are supported).



**Figure 8-7: Example of Fabric Design Using Centralized Routing**

Any fabric packets that need to be routed between two hosts in different VNI-mapped VLANs are sent to a centralized overlay vRouter and then VXLAN-encapsulated or -decapsulated depending on source and/or destination's host location.

When multiple sequential hardware operations are required, such as in the forwarding → decapsulation → routing sequence, or in the reverse routing → encapsulation → forwarding sequence (or also when any packet replication is needed), Pluribus leverages the hardware's capabilities to implement *multiple packet lookup and rewrite passes* using a technology called *recirculation* (or hardware loopback).

This technique is used whenever the hardware is incapable of applying all the necessary operations to certain traffic in a single instance (that is, in a single pass). Hence, such traffic will have to be recirculated at least once through the hardware forwarding engine.

Pluribus supports the recirculation technique with the dedicated `vxlan-loopback-trunk` configuration.



## About the VXLAN Loopback Trunk

Netvisor ONE uses the `vxlan-loopback-trunk` configuration command to set aside a number of (internal or front-panel) ports that are configured as a logical group (i.e., a dedicated port channel) for recirculation.

This special trunk link is auto-created and is reserved to be used to carve out the necessary amount of bandwidth required for the recirculation of packets.

Typically, the network administrator allocates a number of ports (e.g., 2 or more for redundancy purposes) to a VXLAN loopback trunk that is sufficient to provide an amount of aggregate bandwidth *in excess of* what is required for routing and/or replication of all Broadcast/Unknown Unicast/Multicast (BUM) VXLAN traffic.

From an operational standpoint, if a packet needs to be recirculated after decapsulation as part of the routing operation, Layer 2 entries for the vRouter MAC address or the VRRP MAC address on VNI-mapped VLANs are programmed to point to the `vxlan-loopback-trunk` link in hardware.

Therefore, to verify that the traffic can be recirculated, the network administrator can simply look at output of the `l2-table-show` command, which should display a special `vxlan-loopback` flag to indicate the recirculation-related hardware state (see also the Checking VXLAN Recirculation L2 and L3 Entries section below).

In summary, it is important to size the VXLAN loopback trunk appropriately and, in certain cases, it may be helpful to verify the operational state of the Layer 2 entries associated to it in the hardware tables.

When monitoring the fabric, it is also useful to periodically check the hardware recirculation counters (as discussed in the *Showing VXLAN Trunk Replication Counters* below) so that the network admin can stay abreast on the peak recirculation bandwidth needs of the various fabric switches.

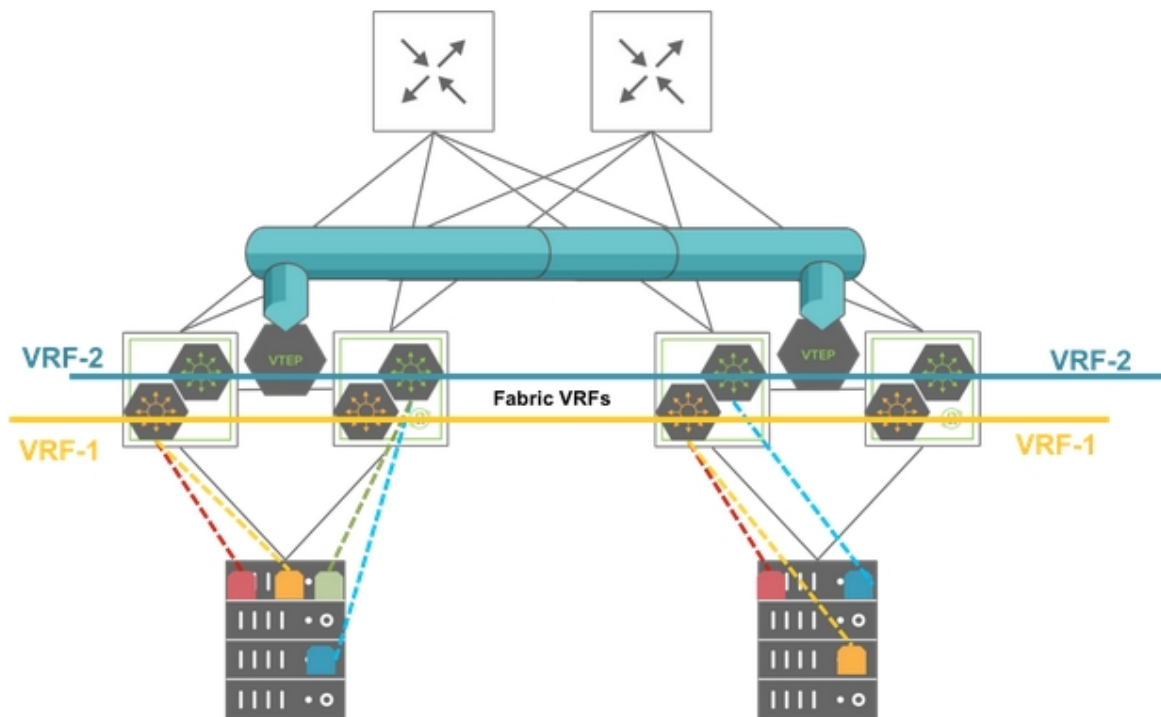
For sizing purposes, note that the VXLAN loopback infrastructure is used for routing traffic before VXLAN encapsulation and/or after VXLAN decapsulation. It is also used for bridging broadcast, unknown unicast or multicast (BUM, in short) traffic in the overlay network.

On the other hand, non-routed known unicast traffic is forwarded and encapsulated, or decapsulated and forwarded, in one single pass *without requiring the VXLAN loopback trunk*.

## About Unicast Fabric Virtual Routing and Forwarding (VRF) with Anycast Gateway

Netvisor ONE Adaptive Cloud Fabric adds Layer 3 segmentation to VXLAN interconnections with the support of VRF (Virtual Routing and Forwarding) instances, complementing the vRouter construct and offering a highly scalable distributed routing solution to network architects.

Netvisor supports VRF as a hardware technology allowing multiple routing spaces to coexist on the same distributed fabric architecture. Furthermore, with the addition of the Anycast Gateway functionality, the Adaptive Cloud Fabric enables distributed forwarding at the first hop router as well as intrinsic VM mobility capabilities across complex multi-site data center designs. This guarantees the maximum VRF scalability possible, limited only by the specific forwarding ASIC capabilities.



**Figure 8-8: East-West Traffic Segmentation with Multiple VRF Instances**

Netvisor Fabric VRFs have the following advantages:

- High scalability with support for a large number of VRF instances on a single fabric node (in the order of thousands depending on hardware capacity especially with newer ASICs and as an aggregate number fabric-wide).
- High performance distributed routing of East-West traffic at the Top-of-Rack (ToR) switch level. The distributed routing capability hosted on each leaf node avoids the need for hair pinning traffic to a centralized vRouter.
- Small forwarding state to manage on each node.



- Native redundancy without needing dedicated redundancy protocols (and potentially extra overhead).
- Dual stack support for IPv4 and IPv6 subnets.
- Simple fabric-wide configuration and management (typical provisioning overhead is proportional to:  $(\text{number\_of\_VRFs} + \text{number\_of\_VLANs} + \text{number\_of\_switches})$  instead of the industry average of up to  $(\text{number\_of\_VLANs} * \text{number\_of\_switches})$ ).
- IPv4 and IPv6 subnets can be automatically stretched to multiple locations without extra configuration.

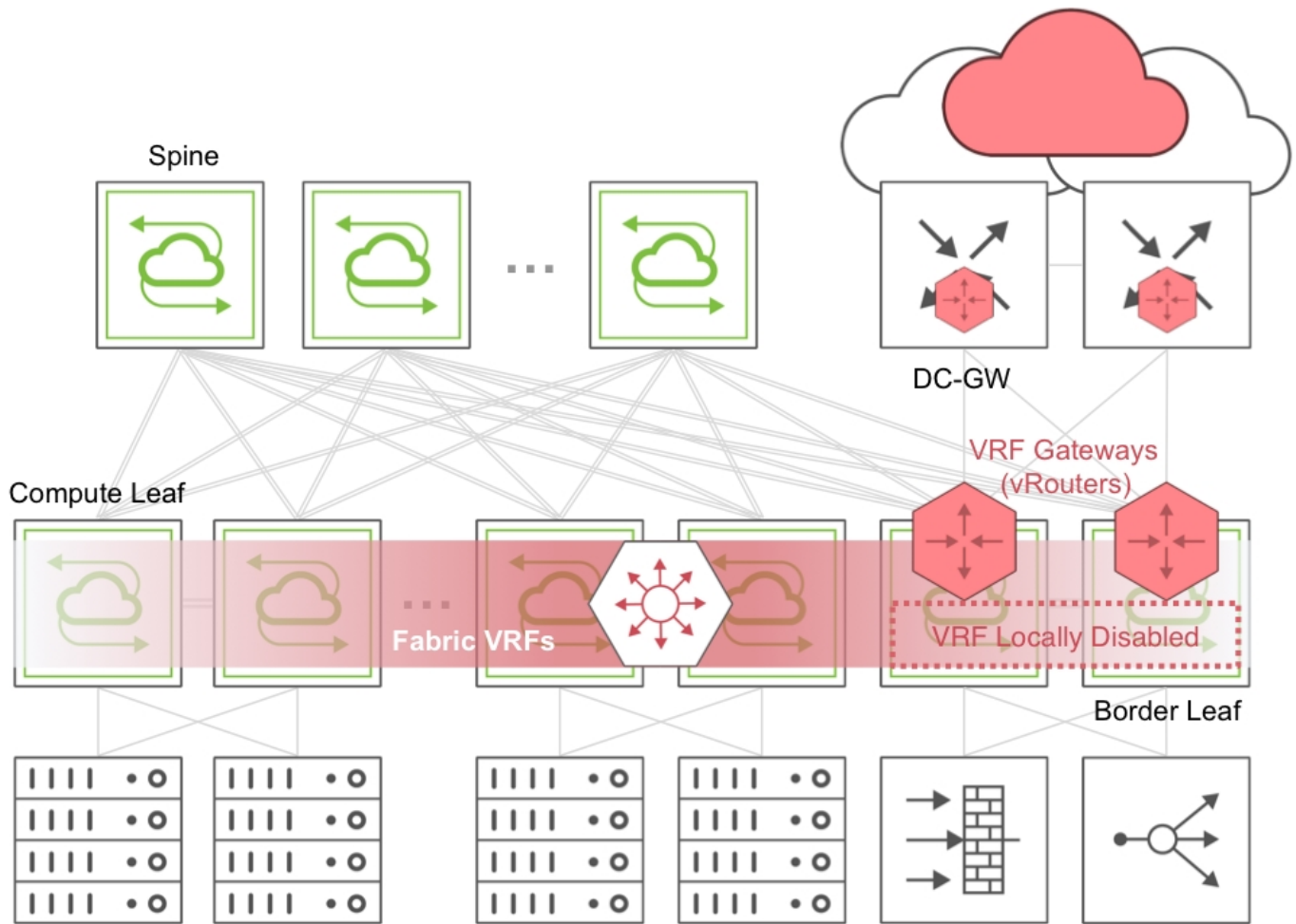
Fabric VRFs are lightweight distributed atomic constructs created without the need for a local vRouter and they do not currently support any routing protocols on VRF instances. This choice enables very high scalability and very low overhead in the management of the distributed segmentation and routing function.

You can connect fabric VRFs to third party VRF routers or gateways either directly using static routing or through a redundant group of border leaf switch(es) running the vRouter function mapping 1:1 to the Fabric VRF instances. In the latter case, border leaf switches can run any supported IGP protocol to interconnect with third party VRF routers or gateways.

For redundancy purposes, you can configure two VRF routers or gateways, sometimes referred to as *DC gateways*, can be configured per VRF (`vrf-gw` and `vrf-gw2`).

To be more precise, these two important configuration parameters represent two static default routes for northbound traffic. They can be quite flexible: after VRF global creation, they can be locally modified by using the `vrf-modify` command and allowing the implementation of different exit points for a VRF depending on the switch location.

In addition, static routing can be leveraged to augment them, for example, to install more than two routes or to change the VRF exit point for specific destination prefixes.



**Figure 8-9: East-West Traffic Segmentation with North-South VRF (DC) Gateways**

As part of the Fabric VRFs configuration, you can create IPv4 and IPv6 subnets, which are atomic objects in the Fabric data plane to associate to the VRF instances in order to implement distributed traffic segmentation.

In particular, Fabric leaf switches use subnet objects for management purposes to represent groups of directly connected hosts with a fabric wide scope across the VXLAN interconnect. Netvisor ONE also uses them to program subnet routes into the hardware to send Layer 3 packets corresponding to unresolved adjacencies to the software so that next-hop resolution through ARP requests can be performed. When a host responds to the ARP request(s), more specific Layer 2 and Layer 3 host entries are configured in the hardware so that end-to-end forwarding ensues.

In addition, Netvisor ONE supports the anycast gateway routing function for the Fabric VRFs to enable distributed first-hop routing, redundancy and mobility. This capability uses a dedicated virtual MAC address, called the anycast gateway MAC address, which gets associated with configurable anycast gateway IP addresses as part of the subnet object configuration.

The default MAC address for the anycast gateway function is 64:0e:94:40:00:02. It can be displayed with the `fabric-anycast-gateway-show` command. If necessary, you can also modify it using the `fabric-anycast-gateway-modify` command.

Furthermore, as a key VRF-aware service, Netvisor ONE supports end host address assignment through the

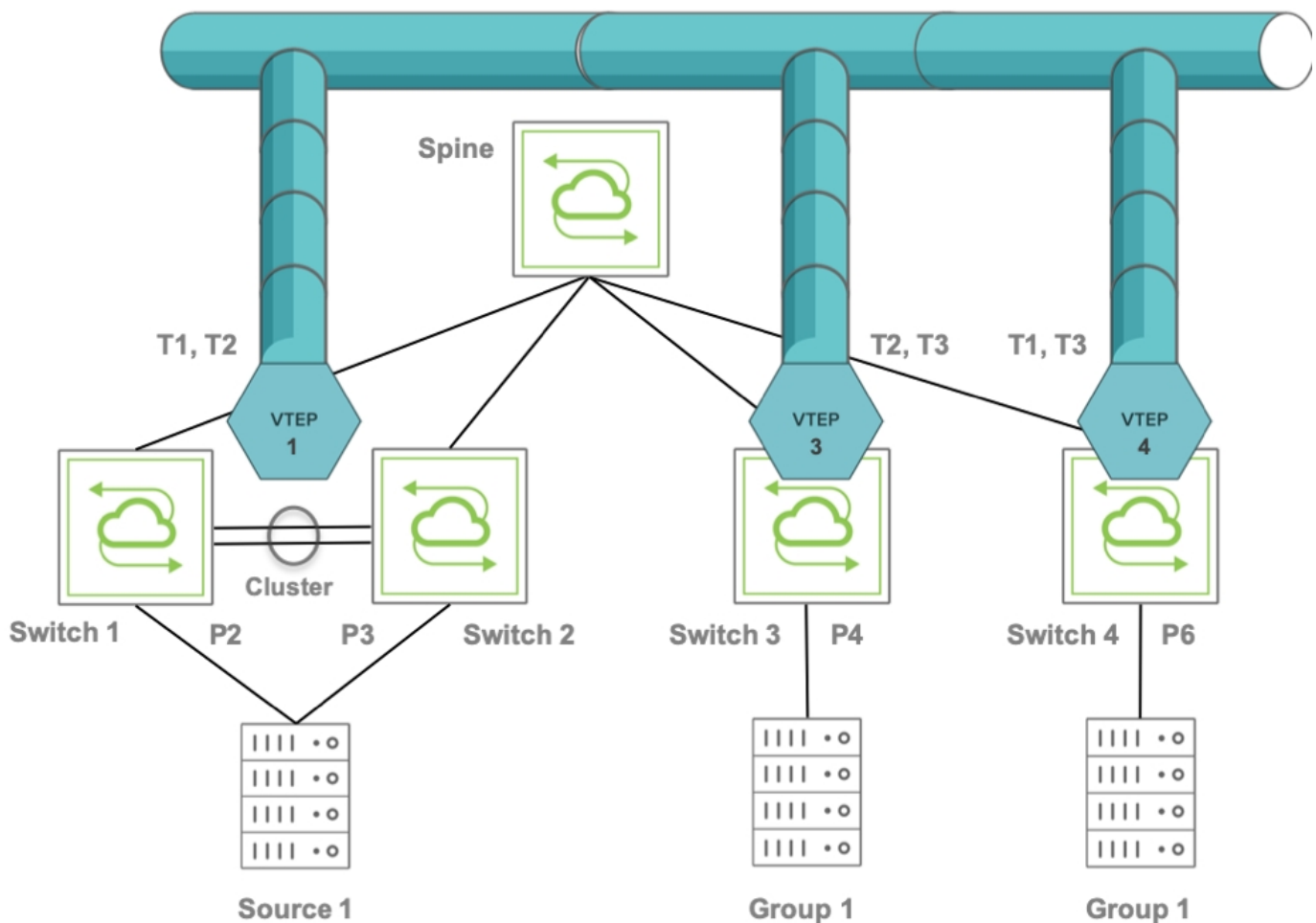
DHCP packet relay function for up to two DHCP servers.

## About IGMP Snooping Support with VXLAN

Starting with version 3.1.1, Netvisor ONE adds support for selective replication of (instead of always flooding) multicast traffic based on IGMP join messages received over ports and tunnels.

With this enhancement multicast traffic belonging to a group is forwarded only to member ports and relevant remote VTEPs.

This feature uses the head-end replication (HER) model for replication of packets to be sent over to remote VTEPs. Netvisor ONE also forwards IGMP join messages over VXLAN tunnels, for other fabric switches to see those messages and as a consequence build the group membership list accordingly.



**Figure 8-10: Example of Fabric Topology with Multicast Distribution**

The topology in the **Figure 8-10** includes Switch 1 and Switch 2 configured as a cluster pair that uses a common virtual IP address (VIP) as the source for two tunnels T1 and T2.

The cluster pair, Switch 1 and Switch 2, appears as one logical switch with a common VXLAN endpoint VTEP1. Two tunnels, T1 and T2, are created with the same local VIP toward the other two end points, VTEP3

and VTEP 4.

The spine switch hashes the traffic from Switch 3 or Switch 4 to the cluster pair, load balancing between Switch 1 and Switch 2.

The above topology includes Switches 1, 2, 3, and 4 with ports 2,3,4, and 6, as part of the same broadcast domain (say, VXLAN ID 10).

Initially, port 4 sends an IGMP join messages for the G1 multicast group. Hence Switch 3 adds local port 4 (P4) as an IGMP member for Layer 2 multicast group G1.

In addition, Switch 3 floods IGMP packets to the remote VTEPs 1 and 4. Hence the remote switches associated with those VTEPs receive IGMP packets and add G1 as Layer 2 multicast group.

On VTEP1's cluster the flooded IGMP join message is also forwarded (syncd) to the cluster peer through the out-of-band channel so that both cluster peers can see and program the same Layer 2 multicast group entry.

Next, P6 joins group G1 too, and the IGMP join packet is flooded to the remote VTEPs 1 and 3.

Now the group membership is:

- On both Switch 1 and Switch 2, VTEP 3 and 4
- On Switch 3, local port P4 and VTEP4
- On Switch 4, local port P6 and VTEP 3

Then, if source S1 sends multicast traffic on P3, that traffic matches the MAC address corresponding to G1's DMAC on VXLAN ID 10 in the Layer 2 table therefore the hardware bridges that traffic to the remote VTEPs 3 and 4. After receiving it, Switch 3 and Switch 4 check the Layer 2 table and forward the traffic to local receivers on P4 on Switch 3 and on P6 on Switch 4.

Refer to the *Configuring IGMP Snooping with VXLAN* section for the configuration details.

## About Distributed Multicast Forwarding with Fabric VRFs

---

**Note:** This feature is supported only on Dell S41xx-ON platforms.

By leveraging the distributed control plane of the Adaptive Cloud Fabric, Pluribus has extended unicast Fabric VRFs (as described above) to also include multicast routing support.

This enhancement is referred to as *Multicast Fabric VRFs with distributed routing support*.

It leverages Pluribus fabric's native capabilities to route multicast traffic without employing a multicast routing protocol (such as PIM) in the underlay or overlay networks.

In particular, Multicast Fabric VRFs are a *logical extension* of the IGMP Snooping feature described in the previous section.

That is because Multicast Fabric VRFs use techniques analogous to IGMP Snooping to handle and selectively forward multicast traffic.

In particular, with Multicast Fabric VRFs, the multicast forwarding logic is enhanced to also handle inter-VLAN forwarding and replication.

These are the lookup steps that are implemented.

First, multicast traffic may need to be locally bridged and/or remotely forwarded to multicast receivers in the same VXLAN ID-mapped VLAN on the other overlay nodes.

In addition, multicast receivers may be located locally in one or more distinct VLANs and hence multicast traffic may need to be routed and replicated to those VLANs as well.

Pluribus leverages Layer 2 hardware table entries to point to local ports and tunnels (represented by logical ports, as seen in the previous section). Those entries point to a loopback trunk to implement a *second pass lookup*. This pass corresponds to a hardware (\*, G, VLAN) L3 multicast lookup for any of the local receiver ports located in different output VLANs.

Furthermore, once a multicast packet reaches a remote VTEP in the overlay network, it is decapsulated and then L3 multicast entries route the traffic locally on that node. The decision to which VXLAN ID(s) to replicate the decapsulated packets is based on the IGMP snooping logic (which knows where all receivers attached to the node are).

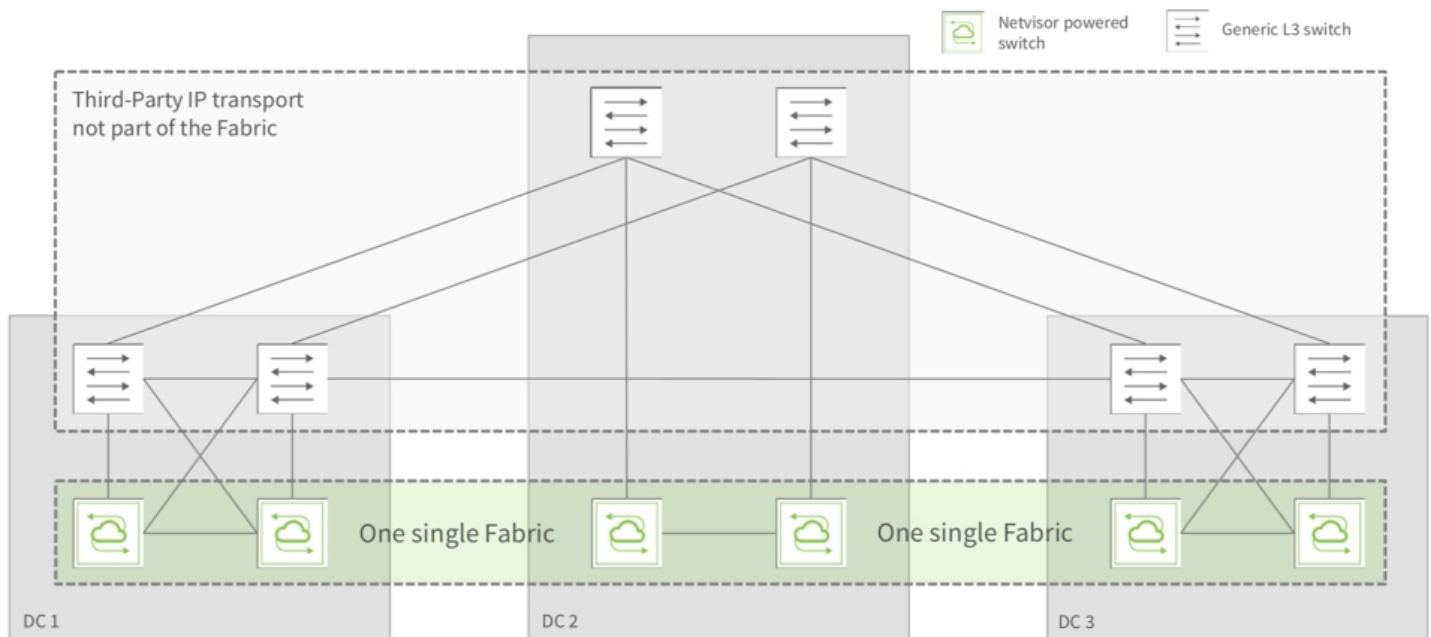
Refer to the *Configuring Multicast Fabric VRFs* section for the configuration details.

## About Pluribus Open Networking Multi-site Fabric

Data Center Interconnect (DCI) is the process of connecting two or more geographically distributed data center locations (also known as *multi-site fabric network designs*).

It is employed to ensure that data is exchanged consistently and in a timely manner among different locations for redundancy, high availability and load sharing purposes, to support a number of important design requirements that include: service resiliency (i.e., fault tolerance), performance and scale, disaster recovery, operational efficiency, local country regulations.

A *multi-site Adaptive Cloud Fabric* design is Pluribus' best-in-class standards-based solution to address customer requirements for multi-site data center connectivity. It leverages the VXLAN-based feature set to be able to extend across a generic IP transport core network, as depicted in the **Figure 8-11** below.



**Figure 8-11: Multisite Fabric Topology Over an Agnostic IP-routed Core**

Thanks to VXLAN, it boasts a number of very desirable characteristics: it's simple, very high performance, highly redundant and can scale up to support dozens of sites, while providing high availability (HA) with ~200 ms failover times.

It's flexible, multi-tenant, interoperable and topology agnostic: it works with any L3 core/underlay and uses sophisticated VXLAN-based Layer 2 extension and pseudo-wire technologies to achieve transparent inter-site communication with end-point tracking.

This enables the support for important use cases such as *VM mobility* for business continuity and geographical load sharing. In addition, it natively supports end-to-end visibility of client-server application traffic as well as of server-to-server communication flows.

In order to implement an Open DCI architecture that can scale to a multi-site design it is important to bring together all the foundational technologies discussed in the previous sections:

- Adaptive Cloud Fabric's distributed control plane
- Switch clusters with vLAGs
- Fabric traffic optimizations
- Virtual Networks (VNETs)-based network provisioning
- Analytics collection

Pluribus provides support for different underlay architectures as well as for different routing options (centralized or distributed) for the fabric overlay.

Distributed forwarding with *Fabric VRFs* represents the most scalable multi-site DC design option (with both unicast and multicast support), whose configuration steps are detailed in the following sections.



## Guidelines and Limitations

---

For most fabric designs the general recommendation is to implement a Layer 3 underlay for scalability and ease of interoperability: BGP and OSPF are solid and mature protocols that in a great number of scenarios offer the most effective way to design and run a fabric.

Whenever possible, for redundancy, scalability and optimal trunk load splitting it is recommended to allocate at least 50% extra bandwidth to VXLAN loopback trunks in excess of the peak aggregate bandwidth expected to be used by all the fabric traffic when recirculated.

These are some common limitations to consider when planning and configuring VXLAN deployments:

1. For RIOT centralized designs up to 32 vRouters are supported (the actual maximum number is platform dependent).
2. For Fabric VRF distributed designs the number of supported VRFs is in the order of thousands, depending on the specific switch model's hardware capacity.

## Configuring the VXLAN Underlay Network

---

Configuring the underlay network involves the configuration of (at least) the following base features:

- Layer 2
- Layer 3
- The fabric
- Clustering
- vLAGs
- Optionally (even though it's highly recommended), traffic analytics

Readers are referred to the respective configuration sections of each feature for more details.

In addition, on Pluribus switches jumbo frame support can be enabled on the appropriate interfaces to accept frames with an MTU larger than 1500 bytes (namely, with a 'jumbo' size). Refer to the *Enabling Jumbo Frame Support* section earlier in this guide for details.

This guarantees that all standard maximum size packets to be transmitted in the overlay network are not dropped by the underlay.

Moreover, starting from release 2.5.0, Netvisor ONE enforces a routing MTU size in hardware, which can be set with the `mtu` option when adding vRouter interfaces (see examples below).

Therefore, in order to avoid large frames to be dropped during Layer 3 forwarding or during encapsulation, in addition to enabling jumbo frames on physical interfaces, it is also important to *configure routed interfaces with the proper MTU settings* (typically between 1580 and 9398 bytes).

Changes to interface attributes (such as the MTU) are recommended to be applied in the initial phases of the underlay configuration. Moreover, changing the MTU on certain higher-level entities such as trunk ports may require more specific commands (such as `trunk-modify` instead of the basic `port-config-modify` command).

## Configuring the Overlay: VTEP Interconnections and VNIs

---

VTEPs can be configured as individual vRouter interfaces. However, as discussed in the *About VTEP High Availability* section, VTEPs are more commonly configured on switch pairs running VRRP to support redundant logical VIPs for VXLAN termination.

In this latter case, the first step is to create a VIP instead of a regular interface.

Both cases are exemplified below (a. and b.) in the list of steps required to set up the overlay:

1. First configure the underlay's vRouter interfaces, with the proper MTU:

- a) Create a vRouter and add a vRouter interface for each VTEP:

```
CLI (network-admin@switch) > vrouter-create name <vr-name> vnet <vnet-name> router-type hardware hw-vrrp-id <id>
```

```
CLI (network-admin@switch) > vrouter-interface-add vrouter-name <vr-name> ip <network/netmask> vlan <y> mtu <mtu>
```

- b) For VTEP HA instead add a vRouter interface using VRRP:

```
CLI (network-admin@switch) > vrouter-interface-add vrouter-name <vr-name> ip <network/netmask> vlan <y> vrrp-id <id> vrrp-primary <ethz.y> mtu <mtu>
```

<mtu> can be set for example to 1580 bytes (or more).

2. Once the VTEPs are created, configure the VTEP connections (also referred to as 'tunnels') from sources to destinations. On non-redundant switches, the tunnel is created with scope `local` whereas on redundant switches the tunnel is created with scope `cluster`:

```
CLI (network-admin@switch) > tunnel-create name <tunnel-name> local-ip <ip1> remote-ip <ip2> scope local vrouter-name <vr-name>
```

```
CLI (network-admin@switch) > tunnel-create name <tunnel-name> local-ip <vip1> remote-ip <vip2> scope cluster vrouter-name <vr-name> peer-vrouter-name <peer-vr-name>
```

3. Then create the mappings between VNIs and VLANs on the respective switches:

```
CLI (network-admin@switch) > vlan-create scope <scope> id <vlan-id> vxlan <vnid>
```

**Note:** A VLAN can be associated to a VNI when created on a VTEP HA pair with the `vlan-create scope cluster id <vlan-id> vxlan <vnid>` command. Also, the mappings can be set up

also with the `vlan-modify id <vlan-id> vxlan <vnid>` command *after VLAN creation*.

Then, to add ports to a VLAN created with `vlan-create` command, use the `vlan-port-add` command, for example:

```
CLI (network-admin@switch) > vlan-port-add vlan-id <vlan-id> ports <port numbers>
```

To delete a VLAN with its mapping, use the `vlan-delete` command.

Lastly, to display the information about a VLAN, for example to verify a VNI mapping and the list of ports added to it, use the `vlan-show` command:

```
CLI (network-admin@switch) > vlan-show id 70 format id, type, vxlan, scope, description, ports, untagged-ports layout vertical
```

```
id:      70
type:    public
vxlan:   70000
scope:   cluster
description:      vlan-70
ports:  0-2, 5-48, 50-52, 54-56, 63-70, 272-273, 275-276, 278-280, 397
untagged-ports:  none
```

#### 4. Add the required VNI mappings to the VXLAN connections:

```
CLI (network-admin@switch) > tunnel-vxlan-add name <tunnel-name> vxlan <vnid>
```

#### 5. For monitoring VXLAN specific states and statistics, use the following commands:

<code>vlan-show</code>	Displays the VXLAN ID associated with the VLAN ID.
<code>tunnel-show</code>	Displays the configured tunnel and the state.
<code>trunk-show</code>	Displays the port used for BUM traffic re-circulation.
<code>port-stats-show</code>	Displays statistics for each port.
<code>tunnel-stats-show</code>	Displays statistics for each tunnel.
<code>vxlan-stats-show</code>	Displays statistics for each VXLAN ID.

## Configuring the VXLAN Loopback Trunk

---

In order to perform routing on the overlay and to replicate Layer 2 broadcast, unknown unicast and multicast (BUM) traffic over multiple VXLAN tunnels on Pluribus switches, you must add ports to a `vxlan-loopback-trunk` using the following command:

```
CLI (network-admin@switch) > trunk-modify name vxlan-loopback-trunk ports  
<list of ports> jumbo
```

Where `<list of ports>` may vary based on the hardware architecture of the switch (on most models it is likely to be a list of front panel ports set aside for this use.)

Check the data sheet of your switch model to verify the number of internal ports as well as front-panel ports available and their respective speeds. Depending on the expected peak amount of routed and BUM traffic, you can use either N x 10G ports or two (or more) 40G ports.

The above command also prints out the following informational message:

```
!!!! Ports updated on vxlan-loopback-trunk. Ports added only support vxlan  
recirculation. Any features configured on these ports may not work. Please  
update the necessary config. Any ports removed need necessary adjustment on  
autoneg,jumbo,etc !!!
```

If you create a VLAN with associated VXLAN ID when `vxlan-loopback-trunk` is without any active ports, you will get this warning message:

```
CLI (network-admin@switch) > vlan-create id 100 scope local vxlan 1001000
```

```
!!!! Vlan 100 created, but there are no ports configured in vxlan-loopback-  
trunk. vxlan forwarding may not function correctly. !!!!
```

## Checking VXLAN Recirculation's L2 and L3 Entries

---

As discussed earlier, when implementing RIOT at least a recirculation pass is used. That requires that Layer 2 and Layer 3 entries be programmed appropriately to point to the loopback trunk.

With the `l2-table-show` command it's possible to verify that a specific VNI-mapped VLAN is configured to point to the VXLAN loopback trunk to forward and then encapsulate the upstream traffic at the ingress VTEP.

```
CLI (network-admin@switch) > l2-table-show vlan 200
```

```
mac:                00:00:5e:00:01:0a
vlan:                200
vxlan                10000
ip:                  2.2.2.2
ports:               69
state:               active,static,vxlan-loopback,router
hostname:            Spine1
peer-intf:           host-1
peer-state:
peer-owner-state:
status:
migrate:
```

When VTEP HA is implemented, the same command can be used to show that the VLAN is configured with VRRP and that it points to the VLAN loopback trunk. For example:

```
CLI (network-admin@Spine1) > l2-table-show vlan 200
```

```
mac:                00:00:5e:b9:01:b0
vlan:                200
vxlan                10000
ip:                  2.2.2.2
ports:               69
state:               active,static,vxlan-loopback,router,vrrp
hostname:            Spine1
peer-intf:           host-1
peer-state:          active,vrrp,vxlan-loopback
peer-owner-state:
status:
migrate:
```

Similarly, in order to decapsulate and route the VXLAN traffic originated from a source VTEP, at the destination VTEP at least two passes are required. Therefore, a Layer 3 entry is programmed to point to the `vxlan-loopback-trunk`.

The `l3-table-show` command can be used to verify that the hardware state is properly set with the `vxlan-loopback` flag:

```
CLI (network-admin@Spine4) > l3-table-show ip 3.3.3.2 format all
```

```
mac:                00:00:c0:00:07:75
ip:                  3.3.3.2
vlan:                200
public-vlan:         200
vxlan:               10000
rt-if:               eth5.200
state:               active,vxlan-loopback
egress-id:           100030
create-time:         16:46:20
last-seen:           17:25:09
hit:                 22
tunnel:              Spine1_Spine4
```

## Showing VXLAN Trunk Replication Counters

**Informational Note:** This command is only supported on **E68** and **E28** platforms.

It is possible to display statistics (incoming and outgoing bytes, packets, etc.) for VXLAN connections (simply called tunnels) that also include the amount of head-end replication (HER) applied to the VXLAN traffic.

Currently, Netvisor ONE uses `vxlan-loopback-trunk` port stats counters for HER replication statistics; therefore, they are cumulative not granular, but may provide a useful tool to measure the amount of `vxlan-loopback-trunk` bandwidth required in real world conditions.

The `tunnel-stats-show` command can be used to output all the statistics including the number of head-end replicated packets and the number of head-end replicated bytes. It can also be executed periodically with the `show-interval seconds-interval` option as shown in the example below:

```
CLI (network-admin@Spine1) > tunnel-stats-show show-interval 1 format all
```

time	tunnel-name	sw-inpkts	sw-opkts	sw-idrops	sw-odrops	counter	HER-pkts	HER-bytes
11:28:03	pip	8.41K	9	3.44K	0	0	4.37M	0
time	tunnel-name	sw-inpkts	sw-opkts	sw-idrops	sw-odrops	counter	HER-pkts	HER-bytes
11:28:04	pip	0	0	1	0	0	1	0
time	tunnel-name	sw-inpkts	sw-opkts	sw-idrops	sw-odrops	counter	HER-pkts	HER-bytes
11:28:05	pip	1	0	1	0	0	1	0
time	tunnel-name	sw-inpkts	sw-opkts	sw-idrops	sw-odrops	counter	HER-pkts	HER-bytes
11:28:06	pip	0	0	1	0	0	1	0
time	tunnel-name	sw-inpkts	sw-opkts	sw-idrops	sw-odrops	counter	HER-pkts	HER-bytes
11:28:07	pip	0	0	2	0	0	57.1K	0
time	tunnel-name	sw-inpkts	sw-opkts	sw-idrops	sw-odrops	counter	HER-pkts	HER-bytes
11:28:08	pip	0	0	1	0	0	76.2K	0
time	tunnel-name	sw-inpkts	sw-opkts	sw-idrops	sw-odrops	counter	HER-pkts	HER-bytes
11:28:10	pip	0	0	1	0	0	75.6K	0
time	tunnel-name	sw-inpkts	sw-opkts	sw-idrops	sw-odrops	counter	HER-pkts	HER-bytes
11:28:11	pip	0	0	1	0	0	76.6K	0
time	tunnel-name	sw-inpkts	sw-opkts	sw-idrops	sw-odrops	counter	HER-pkts	HER-bytes
11:28:12	pip	0	0	1	0	0	67.0K	0

**Figure 8-12: Wide Multi-Column Output of VXLAN Head End Replication Counters**



## Displaying ECMP Load Balancing Info for VXLAN

---

As discussed in the earlier sections, equal-cost multi-path (ECMP) load balancing defines a routing strategy in which next-hop packet forwarding to a single destination can occur over multiple paths.

In case of VTEP interconnections, it is possible to verify that multiple next hops are available with the `tunnel-show` command.

It is also possible to verify that ECMP is applied in hardware (provided multiple next hops exist) and to display the selection of the next hops by checking the RIB table with the `vrouter-rib-routes-show` command.

For example, the user has configured a tunnel called `leaf1toleaf2` with a local IP address `22.3.11.1` and a remote IP address `32.4.11.1`. The network design provides *two next hops* to route the VXLAN traffic from the local VTEP to the remote one: `192.178.0.6` and `192.178.0.2`.

The `tunnel-show` and `vrouter-rib-routes-show` commands can be used to display the tunnel's specific next hops as well as the route info for the remote end point (`32.4.11.0/24`) in the RIB table, as shown below:

```
CLI (network-admin@leaf1) > tunnel-show name leaf1toleaf2 ecmp-nexthops
layout vertical
```

```
scope:                local
name:                 leaf1toleaf2
type:                vxlan
vrouter-name:        scorpius-vxlan-01
local-ip:            17.60.13.1
remote-ip:           17.60.11.1
router-if:           eth0.3
next-hop:           192.178.0.6
next-hop-mac:        66:0e:94:f7:31:f0
nexthop-vlan:        4091
active:              yes
state:               ok
bfd:                 disabled
bfd-state:           not-replicator-vtep
route-info:          17.60.11.0/30
ports:               49
auto-tunnel:         static
scope:               local
name:                 leaf1toleaf2
type:                vxlan
vrouter-name:        scorpius-vxlan-01
local-ip:            17.60.13.1
remote-ip:           17.60.11.1
router-if:           eth0.3
next-hop:           192.178.0.2
```

```

next-hop-mac:      66:0e:94:b7:95:c3
nexthop-vlan:      4092
active:            yes
state:             ok
bfd:               disabled
bfd-state:         not-replicator-vtep
route-info:        17.60.11.0/30
ports:             272
auto-tunnel:       static
  
```

```
CLI (network-admin@leaf1) > vrouter-rib-routes-show ip 32.4.11.0
```

vrid	ip	prelen	number-of-nexthops	nexthop	flags	vlan	intf_ip	intf-id
0	17.60.11.0	30	2	192.178.0.6	ECMP,in-hw	4091	192.178.0.5	1
0	17.60.11.0	30	2	192.178.0.2	ECMP,in-hw	4092	192.178.0.1	0

## Configuring VTEP Objects with Automatic Fabric Connections

---

Based on the command sequence above, configuring a series of unidirectional end-point interconnections (even in a simple triangular topology) requires the network administrator to manually issue a long list of explicit (and hence tedious) `tunnel-create` commands.

Therefore, starting from release 2.6.0, Pluribus has introduced an additional user-friendly operational simplification element called a *logical VTEP configuration object* that leverages the intelligence of the fabric's distributed control plane to automate the aforementioned tedious connection setup process.

With this configuration paradigm, instead of having to create individual unidirectional connections between the end points (i.e., the `tunnel-create` model), a user can simply create a single VTEP object per endpoint with the `vtep-create` command and then map the required VXLAN identifiers to it.

This in turn triggers the *automatic creation* of all the required VXLAN connections in both directions between endpoints, yielding a *significant amount of configuration and time savings*.

An example of VTEP object configuration syntax is displayed below:

```
CLI (network-admin@switch) > vtep-create name <vtep-object-name> vrouter-  
name <vrouter name> ip <primary IP> virtual-ip <vip> [location <switch  
name>]
```

```
CLI (network-admin@switch) > vtep-vxlan-add name <vtep-object-name> vxlan  
<vnid1>
```

```
CLI (network-admin@switch) > vtep-vxlan-add name <vtep-object-name> vxlan  
<vnid2>
```

```
CLI (network-admin@switch) > vtep-vxlan-add name <vtep-object-name> vxlan  
<vnid3>
```

And so on...

VTEP objects can also be displayed or deleted with the commands `vtep-show` and `vtep-delete`, respectively.

## Disabling VXLAN Termination

---

It is possible, for security purposes, to disable VXLAN termination on certain ports that are not supposed to source VXLAN-encapsulated traffic.

This prevents any malicious host from generating VXLAN encapsulated packets that would normally be subject to (unwanted) VXLAN tunnel termination and subsequent forwarding. For example, the following command disables the termination on port 35:

```
CLI (network-admin@switch) > port-config-modify port 35 no-vxlan-termination
```

It can be re-enabled, if necessary, like so:

```
CLI (network-admin@switch) > port-config-modify port 35 vxlan-termination
```

The default settings are:

- vNETs with `vlan-type private` rely on the VXLAN functionality to implement their private characteristics. Therefore, when a port is configured to be a managed port with `vlan-type private`, VXLAN termination is disabled by default.
- Underlay ports have VXLAN termination on by default and can use the `port-config-modify` command to disable VXLAN termination as deemed to enforce port level security.

## Configuring Unicast Fabric VRFs with Anycast Gateway

The following commands are used for the configuration of VRF instances and of the associated VRF gateway (vrf-gw and vrf-gw2) IP addresses:

CLI (network-admin@switch) > vrf-create

name name-string	Specify a name for the VRF.
vnet vnet-name	Specify the name of the vNET to assign the VRF. If you only have a global vNET configured, omit this parameter.
scope local cluster fabric	Specify the scope for the VRF.
vrf-gw ip-address	Specify the gateway IP address.
vrf-gw2 ip-address	Specify the second gateway IP address.
vrf-gw-ipv6 ip-address	Specify the IPv6 gateway address.
vrf-gw2-ipv6 ip-address	Specify the second IPv6 gateway address.

CLI (network-admin@switch) > vrf-delete

name name-string	Specify VRF name that you want to delete.
vnet vnet-name	Specify the name of the vNET assigned to the VRF.

CLI (network-admin@switch) > vrf-modify

name name-string	Specify a name for the VRF.
vnet vnet-name	Specify the name of the vNET to assign the VRF.
scope local cluster fabric	Specify the scope for the VRF.
vrf-gw ip-address	Specify the gateway IP address.
vrf-gw2 ip-address	Specify the second gateway IP address.
vrf-gw-ipv6 ip-address	Specify the IPv6 gateway address.
vrf-gw2-ipv6 ip-address	Specify the second IPv6 gateway address.

CLI (network-admin@switch) > vrf-show

name name-string	Displays the name of the VRF.
------------------	-------------------------------

---

<code>vnet vnet-name</code>	Displays the name of the vNET assigned the VRF.
<code>scope local cluster fabric</code>	Displays the scope of the VRF.
<code>vrf-gw ip-address</code>	Displays the gateway IP address.
<code>vrf-gw2 ip-address</code>	Displays the second gateway IP address.
<code>vrf-gw-ipv6 ip-address</code>	Displays the IPv6 gateway address.
<code>vrf-gw2-ipv6 ip-address</code>	Displays the second IPv6 gateway address.

---

The following commands are used for the configuration of subnet objects for the associated anycast gateway addresses and the associated VNIs:

CLI (network-admin@switch) > subnet-create

---

<code>name name-string</code>	Specify the name of the subnet.
<code>scope local cluster fabric</code>	Specify the scope for the VRF.
<code>vnet vnet-name</code>	Specify the name of the vNET to assign the VRF.
<code>vxlan vxlan-id</code>	Specify the VXLAN ID to assign to the subnet.
<code>vrf vrf name</code>	Specify the VRF to which the subnet belongs to.
<code>network ip-address</code>	Specify the IPv4 network IP address.
<code>netmask netmask</code>	Specify the netmask for the IPv4 address.
<code>anycast-gw-ip ip-address</code>	Specify the anycast gateway IPv4 address for the subnet.
<code>network6 ip-address</code>	Specify the IPv6 subnet network address.
<code>netmask6 netmask</code>	Specify the IPv6 subnet netmask address.
<code>anycast-gw-ip6 ip-address</code>	Specify the anycast gateway IPv6 address for the subnet.
<code>packet-relay enable disable none</code>	Enable or disable the packet relay.
<code>forward-proto dhcp</code>	Specify the protocol type to forward the packets.
<code>forward-ip ip-address</code>	Specify the forwarding IPv4 address.
<code>forward-ip2 ip-address</code>	Specify the second forwarding IPv4 address.
<code>forward-ip6 ip-address</code>	Specify the forwarding IPv6 address.
<code>forward-ip6-2 ip-address</code>	Specify the second forwarding IPv6 address.
<code>enable disable</code>	Specify to enable/disable subnet routing.

---

CLI (network-admin@switch) > subnet-delete

<code>name name-string</code>	Specify the name of the subnet.
<code>vnet vnet-name</code>	Specify the name of the vNET to assign the VRF.
<code>vrf name-string</code>	Specify the VRF to assign the subnet.

CLI (network-admin@switch) > subnet-modify

<code>name name-string</code>	Specify the name of the subnet.
<code>vnet vnet-name</code>	Specify the name of the vNET to assign the VRF.
Specify one or more of the following options:	
<code>network ip-address</code>	Specify the IPv4 network IP address.
<code>netmask netmask</code>	Specify the netmask for the IPv4 address.
<code>anycast-gw-ip ip-address</code>	Specify the anycast gateway IPv4 address for the subnet.
<code>network6 ip-address</code>	Specify the IPv6 subnet network address.
<code>netmask6 netmask</code>	Specify the IPv6 subnet netmask address.
<code>anycast-gw-ip6 ip-address</code>	Specify the anycast gateway IPv6 address for the subnet.
<code>packet-relay enable disable none</code>	Enable or disable the packet relay.
<code>forward-proto dhcp</code>	Specify the protocol type to forward the packets.
<code>forward-ip ip-address</code>	Specify the forwarding IPv4 address.
<code>forward-ip2 ip-address</code>	Specify the second forwarding IPv4 address.
<code>forward-ip6 ip-address</code>	Specify the forwarding IPv6 address.
<code>forward-ip6-2 ip-address</code>	Specify the second forwarding IPv6 address.
<code>enable disable</code>	Specify to enable/disable subnet routing.

CLI (network-admin@switch) > subnet-show

<code>name name-string</code>	Displays the name of the subnet.
<code>scope local cluster fabric</code>	Displays the scope for the VRF.
<code>vnet vnet-name</code>	Displays the name of the vNET to assign the VRF.
<code>vlan vlan-id</code>	Displays the VLAN ID to assign to the subnet.
<code>vxlan vxlan-id</code>	Displays the VXLAN ID to assign to the subnet.
<code>vrf name-string</code>	Displays the VRF to assign the subnet.
<code>network ip-address</code>	Displays the network IPv4 address.
<code>netmask netmask</code>	Displays the netmask for the IPv4 address.

<code>anycast-gw-ip ip-address</code>	Displays the anycast gateway IPv4 address.
<code>network6 ip-address</code>	Displays the IPv6 subnet network address.
<code>netmask6 netmask</code>	Displays the IPv6 subnet netmask address.
<code>anycast-gw-ip6 ip-address</code>	Displays the anycast gateway IPv6 address for the subnet.
<code>linklocal ip-address</code>	Displays the IPv6 Link Local address.
<code>packet-relay enable disable none</code>	Displays the packet relay mode.
<code>forward-proto dhcp</code>	Displays the protocol type forwarding the packets.
<code>forward-ip ip-address</code>	Displays the forwarding IPv4 address.
<code>forward-ip2 ip-address</code>	Displays the second forwarding IPv4 address.
<code>forward-ip6 ip-address</code>	Displays the forwarding IPv6 address.
<code>forward-ip6-2 ip-address</code>	Displays the second forwarding IPv6 address.
<code>state init ok vxlan not found vxlan deactivated not-in-hw vrouter interface exists</code>	Displays the subnet state.
<code>hw-state no-hw-state</code>	Displays if there is a hardware state present.
<code>enable disable</code>	Displays the state of the subnet routing.
<code>format fields-to-display</code>	Display output using a specific parameter. Use all to display all possible output.
<code>parsable-delim character</code>	Display output formatted for machine parsing using a specified delimiter.
<code>sort-asc</code>	Display output in ascending order.
<code>sort-desc</code>	Display output in descending order.
<code>show dups</code>	Display duplicate entries in the output.
<code>layout vertical horizontal</code>	Format the output in a vertical or horizontal layout.
<code>show-interval seconds-interval</code>	Repeat the show command at a specified interval.
<code>show-headers no-show-headers</code>	Display column headers or not.
<code>limit-output number</code>	Limit the display output to a specific number of entries.
<code>count-output</code>	Display the number of entries in the output. This is useful with vRouter show commands.
<code>count-only</code>	Displays the number of entries only.
<code>unscaled</code>	Display full values in the output instead of scaled approximate values.
<code>raw-int-values</code>	Display integer values instead of mapped values



The following commands allow you to modify and display anycast gateway information on the fabric:

CLI (network-admin@switch) > fabric-anycast-mac-show

format fields-to-display	Display output using a specific parameter. Use all to display all possible output.
parsable-delim character	Display output formatted for machine parsing using a specified delimiter.
sort-asc	Display output in ascending order.
sort-desc	Display output in descending order.
show dups	Display duplicate entries in the output.
layout vertical horizontal	Format the output in a vertical or horizontal layout.
show-interval seconds-interval	Repeat the show command at a specified interval.
show-headers no-show-headers	Display column headers or not.
limit-output number	Limit the display output to a specific number of entries.
count-output	Display the number of entries in the output. This is useful with vRouter show commands.
count-only	Displays the number of entries only.
unscaled	Display full values in the output instead of scaled approximate values.
raw-int-values	Display integer values instead of mapped values

CLI (network-admin@switch) > fabric-anycast-mac-modify

mac mac-address	Modify the MAC address for anycast. The default MAC address is 64:0e:94:40:00:02.
-----------------	---

For example, the following vrf-create command can be used to create VRF-1:

CLI (network-admin@switch) > vrf-create name VRF-1 scope fabric

The vrf-create command can be issued to configure for instance 1000 VRFs on a single node, as shown in this output:

CLI (network-admin@switch) > vrf-show count-output

name	vnet	scope	anycast-mac	vrf-gw	vrf-gw2	active	hw-router-mac	hw-vrid
VRF-1	0:0	fabric	64:0e:94:40:00:02	::	::	no	00:00:00:00:00:00	-1
VRF_2	0:0	fabric	64:0e:94:40:00:02	::	::	yes	00:00:00:00:00:00	1

```

VRF_3    0:0  fabric 64:0e:94:40:00:02  ::  ::  yes  00:00:00:00:00:00  2
VRF_4    0:0  fabric 64:0e:94:40:00:02  ::  ::  yes  00:00:00:00:00:00  3
VRF_5    0:0  fabric 64:0e:94:40:00:02  ::  ::  yes  00:00:00:00:00:00  4
VRF_6    0:0  fabric 64:0e:94:40:00:02  ::  ::  yes  00:00:00:00:00:00  5
...
VRF_999  0:0  fabric 64:0e:94:40:00:02  ::  ::  yes  00:00:00:00:00:00  999

```

Count: 999

**Note:** The newer ASICs can support an even higher count. The maximum number is ASIC *limited*.

The following commands can be used to create two subnet objects associated with VRF-1 for East-West traffic segmentation:

```
CLI (network-admin@switch) > vlan-create id 12 vxlan 500012 scope fabric
ports none
```

```
CLI (network-admin@switch) > vlan-create id 13 vxlan 500013 scope fabric
ports none
```

```
CLI (network-admin@switch) > subnet-create name subnet-vxlan-500012 scope
fabric vxlan 500012 network 172.10.2.0/24 anycast-gw-ip 172.10.2.1 vrf VRF-
1
```

```
CLI (network-admin@switch) > subnet-create name subnet-vxlan-500013 scope
fabric vxlan 500013 network 172.10.3.0/24 anycast-gw-ip 172.10.3.1 vrf VRF-
1
```

Finally, the following commands can be used to create two smaller subnets (/29) to provide North-South reach-ability in and out of VRF-1 to/from VRF gateways 172.10.0.2 and 172.10.1.2:

```
CLI (network-admin@switch) > vlan-create id 10 vxlan 500010 scope fabric
ports none
```

```
CLI (network-admin@switch) > vlan-create id 11 vxlan 500011 scope fabric
ports none
```

```
CLI (network-admin@switch) > subnet-create name subnet-vxlan-500010 scope
fabric vxlan 500010 network 172.10.0.0/29 anycast-gw-ip 172.10.0.1 vrf VRF-
1
```

```
CLI (network-admin@switch) > subnet-create name subnet-vxlan-500011 scope
fabric vxlan 500011 network 172.10.1.0/29 anycast-gw-ip 172.10.1.1 vrf VRF-
1
```

**Note:** The scope of the VRF and subnet objects typically would be *fabric*; however, to cater to specific needs and designs it is also possible to configure *local VRFs* and *subnets* in certain cases.

The next step is to configure the VRF gateways for VRF-1:

```
CLI (network-admin@switch) > switch <switch_list> vrf-modify name VRF-1
vrf-gw 172.10.0.2 vrf-gw2 172.10.1.2
```



**Figure 8-13: Fabric VRFs with Border Leaves Connecting External Network**

In this example it is assumed that the connectivity is implemented with static routing on the DC gateways. To provide inbound reach-ability for VRF-1, the DC gateways must be provisioned with static routes for the VRF subnets receiving traffic from external networks, using the adjacent anycast gateway addresses as next-hop:

```
DC-Gateway-1# ip route vrf VRF-1 172.10.2.0/23 172.10.0.1
DC-Gateway-1# ip route vrf VRF-1 172.10.2.0/23 172.10.1.1
DC-Gateway-2# ip route vrf VRF-1 172.10.2.0/23 172.10.0.1
DC-Gateway-2# ip route vrf VRF-1 172.10.2.0/23 172.10.1.1
```

## Configuring IGMP Snooping with VXLAN

By snooping IGMP messages it is possible to determine the (local) port membership for multicast groups. It is also possible to include the logical ports associated with VXLAN tunnels and their remote VTEPs when IGMP messages are snooped on remote overlay network nodes.

The following command supports this feature:

```
CLI (network-admin@switch) > igmp-snooping-modify vxlan|no-vxlan
```

vxlan no-vxlan	Enable IGMP on VXLAN. Disabled by default.
----------------	--

```
CLI (network-admin@switch) > igmp-snooping-modify vxlan
```

```
CLI (network-admin@switch) > igmp-snooping-show
```

```
enable:                yes
vxlan:                 yes
enable-vlans:          1-4092
snoop-link local-vlans: none
```

To disable it:

```
CLI (network-admin@switch) > igmp-snooping-modify no-vxlan
```

```
CLI (network-admin@switch) > igmp-snooping-show
```

```
enable:                yes
vxlan:                 no
enable-vlans:          1-4092
snoop-linklocal-vlans: none
```

**Informational Note:** IGMP Snooping is enabled by default while the VXLAN option is disabled by default.

Let us consider an example: Assume that IGMP join messages for group 239.1.1.1 (from source 10.1.1.2) are received on a tunnel associated with VLAN 10 (with VNI 10), as shown in the command output below:

```
CLI (network-admin@switch) > vlan-show vxlan 10
```

id	type	vxlan	vxlan-type	replicators	scope	description	active	stats	ports	untagged-ports
active-edge-ports										
-----										
-----										
10	public	10	user	none	local	vlan-10	yes	no	9,41,69-72,253	9

Group IP 239.1.1.1 is associated to source IP 10.1.1.2 and its port membership list only contains the logical port ID (12755068416) associated with a VXLAN tunnel:

```
CLI (network-admin@switch) > igmp-show group-ip 239.1.1.1
```

group-ip	node-ip	vnet	vxlan	bd	vlan	port	source	node-type	expires(s)
239.1.1.1	10.1.1.2			10	12	<b>12755068416</b>	0.0.0.0	host	0

You can check the tunnel info (such as its associated VTEP IP addresses) corresponding to logical port **1275068416** with the following command:

```
CLI (network-admin@switch) > tunnel-show tunnelID 1275068416
```

```
scope:                local
name:                 auto-tunnel-70
type:                 vxlan
vrouter-name:         vr1
local-ip:             70.1.1.2
remote-ip:            80.1.1.2
router-if:            eth1.4092
next-hop:             70.1.1.1
next-hop-mac:         66:0e:94:70:61:7f
remote-switch:        0
active:               yes
state:                ok
bfd:                  disabled
bfd-state:            unknown
error:
route-info:           80.1.1.0/24
ports:                19
auto-tunnel:          auto
```

You can also verify that the L2 table contains the MAC address corresponding to group IP 239.1.1.1 (i.e., 01:00:5e:01:01:01):

```
CLI (network-admin@switch) > l2-table-hw-show mac 01:00:5e:01:01:01
```

mac	vlan	vxlan	ports	state	hw-flags	mc-index
<b>01:00:5e:01:01:01</b>	10	10	none	active,static,hit		201326595

## Configuring Multicast Fabric VRFs

**Note:** This feature is supported only on Dell S41xx-ON platforms.

The CLI commands to configure and verify distributed multicast forwarding with Multicast Fabric VRFs are described below:

```
CLI (network-admin@switch) > vrf-multicast-show
enable: yes
```

```
CLI (network-admin@switch) > vrf-multicast-modify
```

scope local fabric	Specify the scope - fabric or local
disable	Specify to disable flooding over tunnels in DM Forwarding
enable	Specify to enable flooding over tunnels in DM Forwarding

Enabling or disabling Multicast Fabric VRFs takes effect only after a reboot. For example, this command disables the feature:

```
CLI (network-admin@switch) > vrf-multicast-modify disable
System must be REBOOTED for this setting to take effect
```

A new command, `vrf-mroute-show`, can be used to show the L3 entries dynamically created to implement L3 multicast forwarding:

```
CLI (network-admin@switch) > vrf-mroute-show
```

srcip	group	vxlان	vlan	vrid	hw-group-id	send-vlan	ports	flags
0.0.0.0	224.1.1.1	10501	501	0	0x200098a	0	none	Active
0.0.0.0	<b>224.1.1.1</b>	<b>10501</b>	<b>501</b>	1	0x200098a	<b>506</b>	1	Active

This command can be used to check the list of multicast destination groups, incoming VLANs, outgoing ports and egress VLANs.

The above output shows (\*, 224.1.1.1, 501) with port 1 as outgoing port and 506 as egress VLAN. Note that this kind of L3 entry will only contain routed ports in the egress VLAN(s) (instead, a separate L2 entry will bridge in the ingress VLAN).

The `igmp-show` command offers a high-level view of the dynamic behavior of the feature (in this example, port 15 is the *loopback port*):

```
CLI (network-admin@switch) > igmp-show
```

group-ip	node-ip	vnet	vxlan	bd	vlan	port	source	node-type	expires(s)
<snip>									
239.1.1.1	::			10	15		0.0.0.0	host	0
239.1.1.1	10.1.1.2			10	12	12755068416	0.0.0.0	host	0

The first line is a Layer 2 entry for group 225.1.1.1 that corresponds to a logical port (1275068416), namely, to a VXLAN tunnel to replicate the multicast traffic to (as it has at least a receiver in the same VNI-mapped VLAN). More than one tunnel may be listed in this position.

The second line is dynamically programmed by the software to send a copy of the multicast traffic for group 225.1.1.1 in VLAN 10 to a special loopback port (see below) so that a second (routing) pass after recirculation can be performed. This means that there is a source sending traffic to the group on the ingress VLAN, therefore a L3 lookup can happen for local routing.

The third line (in the show output above) shows IGMP membership on port 19 in egress VLAN 30 (which needs to be routed to).

## Supported Releases

Command/Parameter	Netvisor ONE Version
<code>trunk-modify name vxlan-loopback-trunk</code>	Commands with parameters first supported in <i>version 2.4</i>
<code>tunnel-create</code>	Command added in <i>version 1.2</i> . The option, <code>cluster</code> , was added to scope and the parameters, <code>vrouter-name</code> and <code>peer-vrouter-name</code> , were added in <i>version 2.4</i> .
<code>tunnel-vxlan-add</code>	<i>In version 1.2 the command was introduced. In version 2.4 the parameter, vxlan, was added.</i>
<code>vtep-create, vtep-delete, vtep-show</code>	Commands introduced in <i>version 2.6.0</i> .
<code>vtep-vxlan-add, vtep-vxlan-remove, vtep-vxlan-show</code>	Commands introduced in <i>version 2.6.0</i> .
<code>vrf-create, vrf-delete, vrf-modify, vrf-show, subnet-create, subnet-delete, subnet-modify, subnet-show</code>	Commands introduced in <i>version 3.0.0</i>
<code>igmp-snooping-modify vxlan no-vxlan</code>	Parameter introduced in <i>version 3.1.1</i>
<code>vrf-multicast-modify, vrf-mroute-show</code>	Commands introduced in <i>version 5.1.0</i>
<code>vle-create, subnet-delete, subnet-modify, subnet-show</code>	Commands introduced in <i>version 2.5.4</i>
<code>vlan-create vxlan-mode standard transparent</code>	In <i>version 2.5.2</i> the parameter <code>vxlan-mode</code> was added.

Please also refer to the *Pluribus Command Reference* document.



## Related Documentation

---

Refer to these sections in the Configuration Guide for more details:

- [Configuring Layer 2 Features](#)
- [Configuring Layer 3 Features](#)
- [Setting up and Administering the Pluribus Fabric](#)
- [Configuring High Availability](#)

for further information on concepts mentioned in this section (such as Layer 2, Layer 3, the fabric, clusters, vLAGs, VRRP, etc.)

## Configuring Advanced Layer 2 Transport Services

---

This chapter provides information for understanding and using the Pluribus Networks Netvisor ONE command line interface (CLI) on a Netvisor ONE switch to configure Advanced Layer 2 Transport Services.

---

- [Understanding Virtual Link Extension \(vLE\)](#)
  - [Understanding VXLAN-based Bridge Domains for IEEE 802.1Q and QinQ Internet-working](#)
  - [Configuring Virtual Link Extension](#)
  - [Configuring Virtual Link Extension State Tracking](#)
  - [Configuring Virtual Link Extension Error Transparency](#)
  - [Configuring Virtual Link Extension Redundancy with Clusters](#)
  - [Showing Virtual Link Extension Between Ports on the Same Switch](#)
  - [Configuring Keep-Alive Time for Virtual Link Extension](#)
  - [Configuring VXLAN-based Bridge Domains for 802.1Q and QinQ Internetworking](#)
  - [Troubleshooting vLE](#)
  - [Guidelines and Limitations](#)
  - [Supported Releases](#)
  - [Related Documentation](#)
-

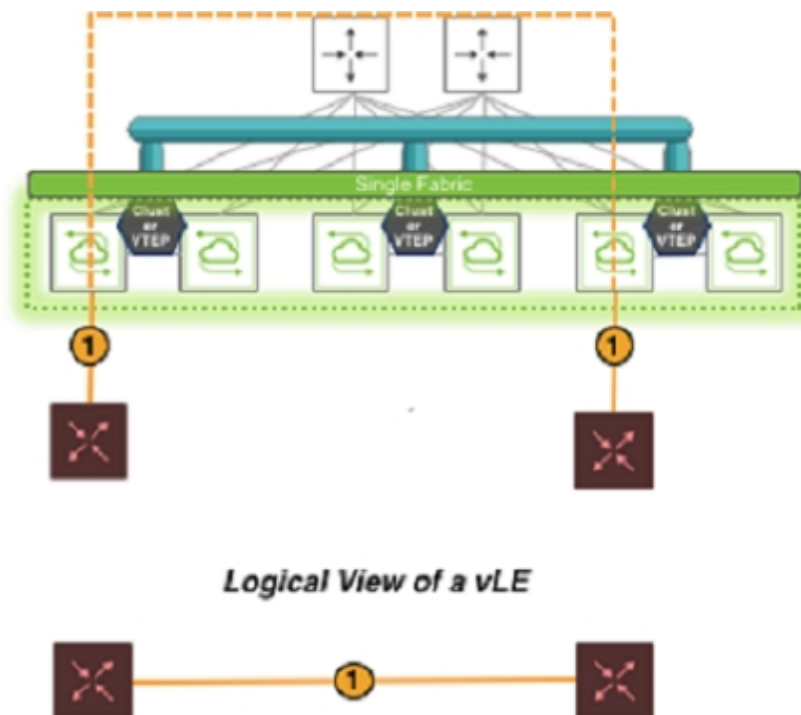
## Understanding Virtual Link Extension (vLE)

Pluribus Networks offers a highly flexible approach to software-defined networking (SDN), called *Adaptive Cloud Fabric*, which leverages a sophisticated distributed architecture in order to enable organizations to build scalable private and public clouds with the very desirable benefits of ease of management, reliability, security and performance.

With regard to ease of management and flexibility, the VXLAN technology described in the *Understanding VXLAN* chapter augments the Adaptive Cloud Fabric with a plethora of advanced capabilities. Among such features is the ability to natively transport traffic across the VXLAN fabric for end-to-end data center connectivity through overlay bridging and distributed routing.

Furthermore, in certain scenarios, network administrators have the requirement for the VXLAN fabric to be able to transport traffic *transparently* between two different ports as if interconnected through a *virtual pipe*: in other words, what comes in on one end goes out of the other end. This capability is often referred to as a ‘pseudo-wire’ in the literature. Pluribus calls it a *virtual wire*.

Pluribus Virtual Link Extension (vLE) implements a transparent point-to-point virtual link over the VXLAN fabric to emulate the connectivity of an actual wire interconnecting any two fabric ports selected by the user, as shown in **Figure 9-1** below.



**Figure 9-1: Virtual Link Extension over the VXLAN Fabric**

The key characteristic of a vLE is its ability to take what comes in on the local end of the virtual pipe and send it to the remote end. That includes regular data plane traffic as well as Layer 2 PDUs.

Hence, one application of Pluribus' Virtual Link Extension feature is the remote monitoring of traffic

between two fabric ports.

Another common use case is *lab automation*, i.e., the flexible interconnection of lab equipment through the fabric: with multiple vLEs the port interconnections can be dynamically (for example, programmatically) managed by the lab administrator through the creation of ad-hoc virtual pipes.

This type of use case is sometimes also referred to as '*IP Virtual Wire*' (as it runs on top of a VXLAN overlay) or '*Virtual Wire+*' to distinguish it from the regular (Layer 1) VirtualWire™ functionality (refer to the *Configuring VirtualWire Deployment Guide*). It is usually deployed with *error transparency* enabled (see below for more details).

From an implementation perspective, it should be noted that the Adaptive Cloud Fabric can transport multiple vLEs across a common high-performance VXLAN-based interconnect, thus enabling a very high degree of configuration flexibility and scalability. The main network constraint is the maximum bandwidth available on any of the shared physical links, which however can be scaled using higher speed Ethernet ports or trunks to avoid any chokepoints.

From a network design standpoint, in terms of ease of configuration, vLEs can use manual or automatic VXLAN tunnels to piggyback the mirrored traffic end-to-end and thus they enjoy the same wide-spanning benefits of any other VXLAN-based feature. The only constraint is that they cannot share the same VXLAN tunnels with other features.

Alternatively, the Pluribus UNUM™ management platform can be leveraged to very effectively perform the provisioning of the entire vLE deployment in very few steps. First, a Layer 3 fabric can be deployed from a UNUM Layer-3 Playbook. Then, *Manager --> Services --> Manage VLE* can be used to create and manage one or more vLEs through user-friendly configuration forms. (Refer to the UNUM Fabric Manager documentation for more details.)

Finally, programmatic creation and management of vLEs can be performed through REST APIs.

## About Virtual Link Extension State Tracking

Just as in the case of physical back-to-back Ethernet connections, it is usually desirable that if one end of a vLE goes down the peer end mirrors that state change.

This behavior is called *link state tracking*. With such symmetric behavior, link state changes can be propagated end-to-end and hence network nodes or endpoints are able to react to them.

In other words, when a vLE is created with tracking enabled between two physical ports of two switches, the vLE remains up as long as both physical ports are in the link up state.

When a vLE is created with tracking enabled between two trunk ports (a.k.a. port channels), the vLE stays up as long as at least one member port in the trunk is up and the remote trunk is also up. When the last trunk member goes down, the vLE is brought down. (Note that when you configure vLE tracking on a trunk port, you cannot configure tracking on individual trunk members.)

In the CLI link state tracking can be enabled/disabled over each vLE individually through the `tracking | no tracking` option. (In fact, having tracking enabled may not be required in all cases, such as for troubleshooting purposes.) Similarly, in the Pluribus UNUM management platform the `tracking | no tracking` option corresponds to a true/false configuration checkbox to be ticked by the user.

For more CLI configuration details refer to the *Configuring Virtual Link Extension State Tracking* section below. Refer to the UNUM Fabric Manager documentation for the specific pane and dashboard layout details for the link state tracking configuration checkbox.

The `node-1-port` and `node-2-port` are the names of the two vLE endpoints whose link state can be tracked. When `node-1-port` goes down the `vle-show` command can be used to check and display the overall vLE state, which is `local-down`. When `node-2-port` goes down, instead, the `vle-show` command displays the `remote-down` state.

If needed, as part of the vLE configuration, it is also possible to tweak the timing to be used by the tracking logic to detect a port down state and to trigger a (virtual) link down event.

For more details refer to the following sections.

## About Virtual Link Extension Between Two Ports of the Same Switch

In earlier versions of Netvisor ONE, the vLE implementation supported only one vLE endpoint per node. With version 5.1.1, both vLE endpoints can be part of the same node. This configuration is referred to as *local vLE*.

A vLE can have two ports in the same node only if the corresponding vLE's VXLAN ID is not already mapped to a tunnel. Similarly, a tunnel can be mapped to a vLE's VXLAN ID if there is at most one local port in the vLE. In the case of auto-tunnels, a VXLAN ID can be mapped to a VTEP only if there is at most one local port in the node associated with that VTEP.

Netvisor ONE allows the users to track the vLE port status when part of a local vLE. In such case, the `node-1` and `node-2` labels refer to the same switch. However, for consistency reasons, the `node-2-port` is still considered as '*remote port*' in the `vle-show` output, even though in actuality both the vLE ports are on the same switch.

Let us assume, for example, that port 30 (`node-2-port`) goes down, the vLE status is displayed as `remote-down` (see example below) so the user can understand exactly which port is causing the vLE to be down. Instead, if port 22 (`node-1-port`) is down, then the vLE status is displayed as `local-down`.

```
CLI (network-admin@switch) > vle-show layout vertical
```

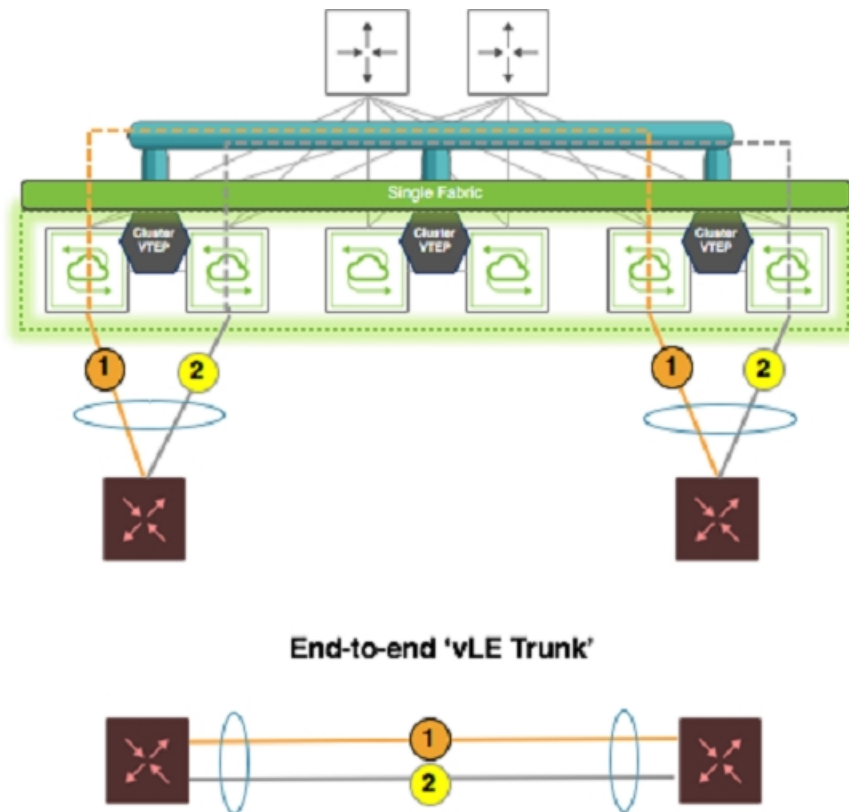
```
name:                vle1
node-1:              vanquish1
node-2:              vanquish1
node-1-port:         22
node-2-port:         30
status:              remote-down
tracking:             enabled
```

Use the `port-show` command to check if the `node-2-port` status is down or `vle-wait`. The `vle-wait` state indicates that the port is brought down due to remote port being down.

## About Virtual Link Extension Redundancy with Clusters

In network designs using pseudo-wires, link redundancy can be achieved by creating trunks using (at least) pairs of pseudo-wires as shown below. Such trunks originate and terminate on the *end devices*.

That can be conveniently achieved in fabric designs that leverage pairs of leaf switches (called *clusters* in Pluribus parlance) to provide redundancy for Layer 2 and Layer 3 last-hop traffic: as shown in **Figure 9-2** below, when vLEs are configured, vLE redundancy can be implemented by setting up *one vLE per cluster switch* and by *leveraging port channeling or NIC teaming* (depicted with an ellipse in the diagram) on the end devices to implement end-to-end vLE trunks.



**Figure 9-2: Virtual Link Extension End-to-end Redundancy**

Leaf switches don't need to implement load balancing and failover for the vLE trunks as vLEs are in essence just end-to-end transparent pipes, which transport all the traffic and propagate port state changes through link state tracking. Hence, vLE-attached end devices can just use standard 802.3ad link bundling, static or dynamic with the LACP protocol, to manage the aggregation and the failover of the virtual wires.

vLE redundancy has the following characteristics:

- Failover between VTEPs used by vLEs is not required.
- Hence VTEPs associated with vLEs do not use a VRRP virtual IP address. They use a primary IP address,

instead, or the real IP address of a dedicated interface.

- L2 PDUs such as LACP messages on vLE transparent VLANs are not sent to a fabric node's management CPU, as they need to be passed transparently through the virtual wire.

In the example topology shown above in Figure 9-2, two generic switches are connected respectively to two cluster leaf nodes. There is no vLAG on such nodes as it is not required. Both generic switches are configured with the LACP protocol on their trunk links to run connectivity checks and to form the trunk dynamically. Hence, they are directly in charge of link redundancy and failover management, whereas the leaf nodes are responsible for *transparent end-to-end traffic transport*.

## About Virtual Link Extension Error Transparency

In lab automation deployments employing vLE it is usually required to not only transport valid data plane frames and PDUs but also to include potential malformed frames that are received on a vLE port.

Starting from Netvisor ONE release 5.1.1, with the optional configuration for *error transparency*, a vLE can be configured to transport also bad CRC frames as well as runts (i.e., undersized frames). Transported runt frames are padded with zeros up to a length of 64 bytes and their CRC is zeroed out, which can be used to recognize an undersized frame on the receiving end.

Error transparency is hardware-dependent (on newer ASICs) and thus works only on the following capable platforms:

- Dell Z9100, Freedom F9532-C
- Dell S4100 Series
- Dell S5200 Series
- Dell S4048T, Freedom F9372-X and F9372-T
- Dell S5048, Freedom F9572L-V

In particular, on these platforms undersized frames with a *minimum* length of 50 bytes are supported. Furthermore, full jumbo frame transport is also supported in Netvisor ONE version 5.1.1 by upping the maximum supported MTU size to 9398 bytes.

Refer to the *Enabling Jumbo Frame Support* section of this Configuration Guides for command details.

Error transparency is enabled globally to minimize the configuration steps required in lab automation setups. However, since it disables CRC checks even on inter-switch links, it can be turned off on a per port basis wherever it's not required.

## About Virtual Link Extension Traffic Analytics

Netvisor ONE does not make copies of vLE-transported control frames (such as LLDP, etc.) to the switch management CPU. Any inner VLAN tag, if present, is also preserved. This is achieved by installing a system vFlow entry called *Virtual-Link-Extend* with the highest priority of 15 with no action specified so that control frames are not terminated and sent to CPU.

In addition, to support *vLE traffic analytics*, a few additional system vFlow entries (named *System-vLE-x*, where x can be S or F or R) are installed with the same priority as the Virtual-Link-Extend entry in order to

copy TCP SYN/FIN/RST packets to the management CPU. This ensures that any SYN/FIN/RST packets carried by vLE can be used for *TCP flow analysis*.

The `vflow-show` command can be used to display such system entries used for transparency and flow analysis:

```
CLI (network-admin@switch) > vflow-show format name,scope,type,proto,tcp-flags,precedence,action,enable
```

name enable	scope	type	proto	tcp-flags	precedence	action
-----	-----	-----	-----	-----	-----	-----
-----						
System-vLE-S enable	local	system	tcp	syn	15	copy-to-cpu
System-vLE-F enable	local	system	tcp	fin	15	copy-to-cpu
System-vLE-R enable	local	system	tcp	rst	15	copy-to-cpu
Virtual-Link-Extend enable	local	system			15	none

```
CLI (network-admin@switch) > connection-show layout vertical
```

```
vnet:      100
vlan:      100
vxlan:     10100
src-ip:    20.20.20.1
dst-ip:    20.20.20.2
dst-port:  http
cur-state:  fin
syn-resends:  0
syn-ack-resends:  0
latency:   74.8us
obytes:    149
ibytes:    311
total-bytes:  460
age: 2h11m21s
```



## Understanding VXLAN-based Bridge Domains for IEEE 802.1Q and QinQ Internet-working

---

With the IEEE 802.1Q standard a network design is constrained to 4094 possible VLAN numbers. When more Layer 2 identifiers are needed, and a hierarchy of network entities can be established, the *VLAN stacking technology* (informally also called *QinQ*, after the name of Cisco's original implementation, also known as 802.1Q Tunneling) can be utilized to scale up to 16 million (4094 x 4094) identifiers using two (concatenated, i.e., 'stacked') VLAN tags instead of one.

The IEEE organization has standardized the VLAN stacking technology with the *Provider Bridges* standard, also known as *IEEE 802.1ad*, which was later incorporated into the IEEE 802.1Q-2011 revision of the LAN/MAN standard.

IEEE 802.1ad refers to providers of services, such as Transparent LAN Services in Metro networks. There are also a number of data center designs that can tap the scalability of the VLAN stacking technology: for example data center networks in which one tag (the so-called *outer tag*) is used to identify a data center customer, while the second tag (the so-called *inner tag*) is used to identify a customer service.

With this technology it is possible to identify up to 16 M services for up to 4K customers. The double-tagged traffic is oftentimes handed off to a provider (for example a cloud exchange provider) which can terminate it for end-to-end connectivity. Hence, this scheme can be used to deploy *hybrid cloud designs* in which private clouds integrate with external clouds.

In practice in such designs QinQ (i.e., VLAN stacking) augments the basic 802.1Q capabilities by massively scaling the number of usable network identifiers organized in a hierarchy.

Pluribus combines this technology with the VXLAN-based Adaptive Cloud Fabric to yield an even higher degree of scalability and flexibility. In fact, VXLAN IDs (a.k.a. VNIs) are 24-bit long (vs. 12-bits of VLAN IDs) and therefore are a perfect match for double-tagging Layer 2 network transports.

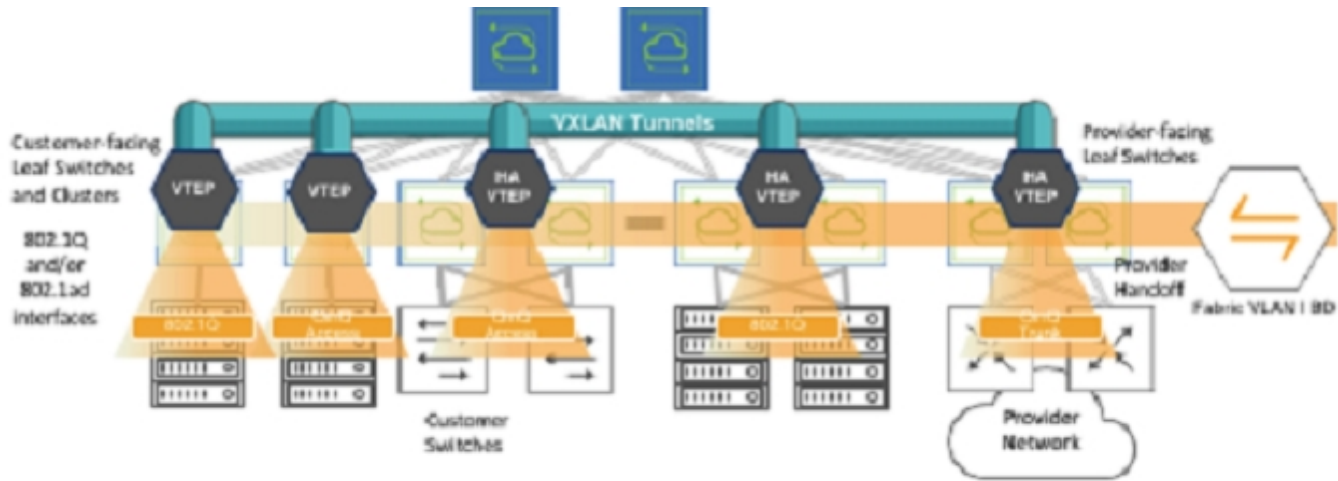
In addition, VLAN ID to VXLAN ID mappings enable a great degree of flexibility for network designers that Pluribus substantiates with a new type of configuration object called *VXLAN-based bridge domain* (BD).

VXLAN-based bridge domains will be supported in a future release of Netvisor ONE: they can be mapped to single or multiple 802.1Q VLANs as well as to dual IEEE 802.1ad VLAN tags, covering all possible design requirements.

The different available configuration models for VXLAN-based bridge domains are:

- *Single tag mapping*, in which an outer VLAN tag (for example, a customer ID) is mapped to a VNI on an IEEE 802.1ad port and the inner VLAN tags are preserved inside the VXLAN encapsulation. This type of mapping can be used on customer facing 'QinQ access' interfaces.
- *Double tag mapping*, in which an outer VLAN + inner VLAN tag pair is mapped to a VNI on an IEEE 802.1ad port (traffic is received double-tagged in ingress and is marked with two tags in egress after VXLAN decapsulation). This type of mapping can be used on multi-VLAN ports (sometimes called *QinQ trunks*) facing for example an external cloud provider.
- *Single 802.1Q tag mapping*, in which a single 802.1Q VLAN (or multiple 802.1Q VLANs) are mapped to a common VNI (for example, for inter-DC communication within the same customer's private cloud)

network).

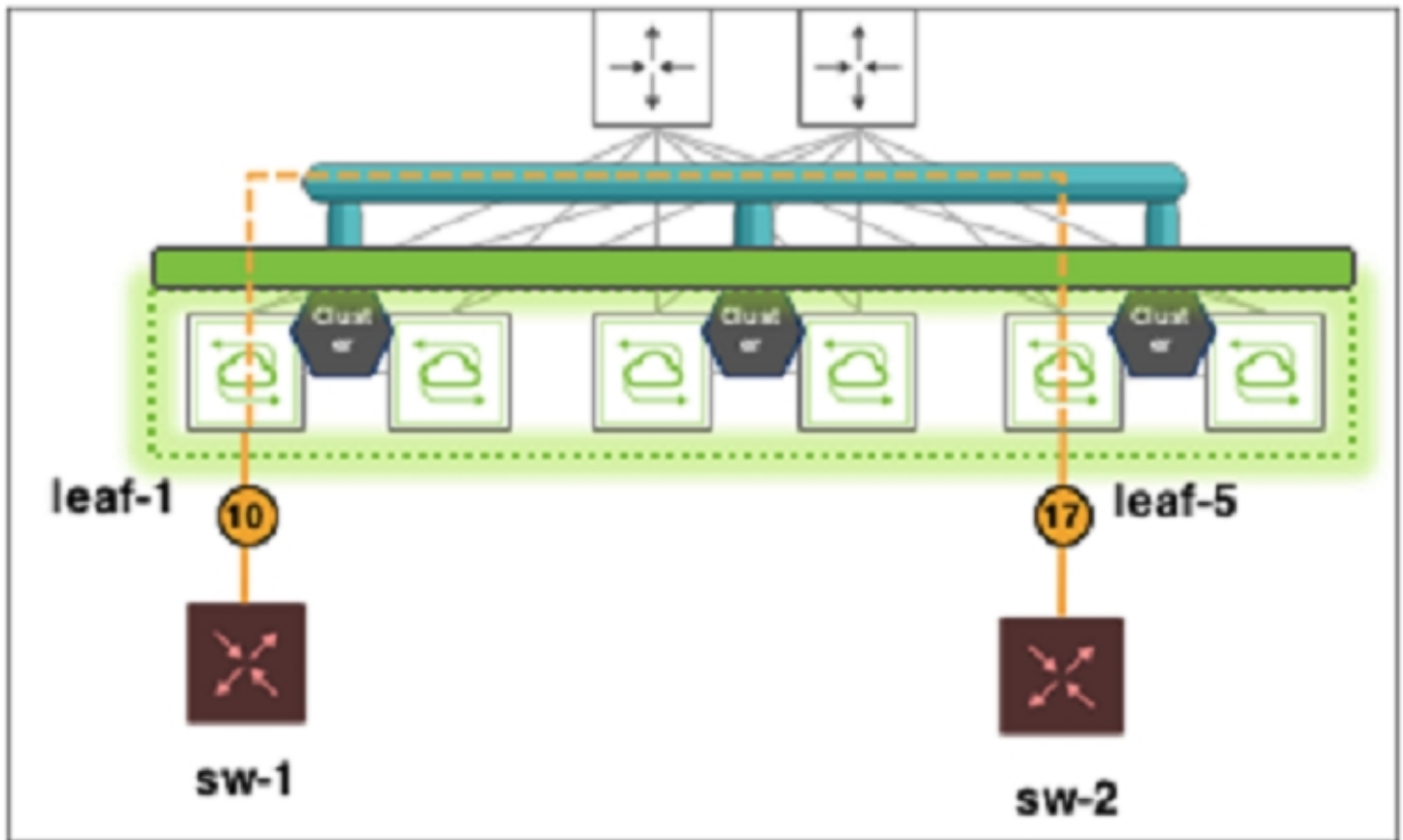


**Figure 9-3: Hierarchical Fabric Structure Using Bridge Domains for Intra- and Inter-DC Connectivity**

The various configuration models of bridge domains yield an unprecedented level of flexibility in terms of advanced Layer 2 transport services, with the ability of re-using VLANs across tenants, supporting QinQ hierarchies and handoff links, as well as aggregating multiple VLANs in the same overlay construct (thus supporting high-scale L2 tenant services).

## Configuring Virtual Link Extension

The vLE configuration revolves around the creation and transport over VXLAN of one or more transparent VLANs using the transparent option of the `vlan-create vxlan-mode standard|transparent` command. To explain how that is implemented, let's first refer to Figure A+4 below which shows an example with port 10 and 17 as ingress/egress vLE ports on two fabric leaf switches that are supposed to be connected transparently through a vLE pseudo-wire:



**Figure 9-4. Two Ports Interconnected Through a vLE Pseudo-wire**

Establishing such vLE pseudo-wire transport requires multiple configuration steps:

1. Configuration of jumbo frames on all the required ports (ingress and egress ports, as well as all inter-switch ports that are used to transport vLE traffic)
2. Creation of Layer 3 VLANs (or of routed ports) that are used as tunnel end points (with an MTU configuration suitable to route jumbo frames)
3. Set up of dedicated VXLAN connections. Typically, manual tunnels are created and set aside for vLE use.
4. Creation of a VXLAN-mapped *transparent* VLAN to transport the traffic over vLE
5. Addition of the transparent VLAN's VXLAN ID to the allocated tunnels.

Let us look at one configuration example for each of the above steps.

*Step 1* requires adding the jumbo option as part of the port configuration with the following command(s):

```
CLI (network-admin@leaf-1) > port-config-modify port 10 jumbo
```

In this example, as shown in the figure above, port 10 on leaf-1 is a vLE port. Jumbo (i.e., oversized) frame support should be enabled if mirroring of ingress frames with a (typical) MTU of 9216 bytes is required.

This option is usually already configured end-to-end on inter-switch links if a VXLAN transport is used in the fabric (refer to the *Configuring VXLAN* section).

Otherwise, the following commands can be used to enable the transport of oversized frames on inter-switch links (individual ports or trunks):

```
CLI (network-admin@leaf-1) > port-config-modify port <inter-switch port-  
list> jumbo
```

```
CLI (network-admin@leaf-1) > trunk-modify name <trunkname> jumbo
```

*Step 2* requires the creation of the *Layer 3 routing endpoints*, in this case VLAN interfaces (a.k.a. SVIs), on the switches where the ingress and egress vLE ports are located:

```
CLI (network-admin@leaf-1)> vlan-create id 1021 scope local description  
VLE-VTEP ports none
```

```
CLI (network-admin@leaf-1)> vrouter-interface-add vrouter-name Leaf-1 vlan  
1021 ip 10.21.1.1/30 mtu 9398
```

```
CLI (network-admin@leaf-5)> vlan-create id 1021 scope local description  
VLE-VTEP ports none
```

```
CLI (network-admin@leaf-5)> vrouter-interface-add vrouter-name Leaf-5 vlan  
1021 ip 10.21.7.1/30 mtu 9398
```

Then the VLAN(s) can be checked with the following command:

```
CLI (network-admin@leaf-1)> vlan-show id 1021
```

switch	id	scope	description	active
leaf-1	1021	local	VLE-VTEP	<b>yes</b>
leaf-5	1021	local	VLE-VTEP	<b>yes</b>

*Step 3* involves the creation of the *VXLAN tunnels*:

```
CLI (network-admin@leaf-1)> tunnel-create name VLE_L1_to_L5 scope local  
local-ip 10.21.1.1 remote-ip 10.21.7.1 vrouter-name Leaf-1
```

```
CLI (network-admin@leaf-5)> tunnel-create name VLE_L5_to_L1 scope local  
local-ip 10.21.7.1 remote-ip 10.21.1.1 vrouter-name Leaf-5
```

The tunnel creation can be checked with the following commands:

```
CLI (network-admin@leaf-1) > tunnel-show local-ip 10.21.1.1 format
switch,scope,name,type,vrouter-name,local-ip,remote-ip
```

switch	scope	name	type	vrouter-name	local-ip	remote-ip
leaf-1	local	VLE_L1_to_L5	vxlan	Leaf-1	10.21.1.1	10.21.7.1

```
CLI (network-admin@leaf-5) > tunnel-show local-ip 10.21.7.1 format
switch,scope,name,type,vrouter-name,local-ip,remote-ip
```

switch	scope	name	type	vrouter-name	local-ip	remote-ip
leaf-5	local	VLE_L5_to_L1	vxlan	Leaf-5	10.21.7.1	10.21.1.1

Step 4 requires the creation of a *transparent VLAN* per vLE:

```
CLI (network-admin@leaf-1)> vlan-create id 3001 scope local description
VLE-1 vxlan-mode transparent vxlan 3001000 ports 10
CLI (network-admin@leaf-5)> vlan-create id 3001 scope local description
VLE-1 vxlan-mode transparent vxlan 3001000 ports 17
```

The VLAN's creation can be checked with the following command:

```
CLI (network-admin@leaf-1)> vlan-show id 3001 format
switch,id,type,vxlan,scope,description,active,ports
```

switch	id	type	vxlan	scope	description	active	ports
leaf-1	3001	public	3001000	local	VLE-1	<b>yes</b>	10
leaf-5	3001	public	3001000	local	VLE-1	<b>yes</b>	17

Finally, in *step 5* the above VXLAN ID(s) of the transparent VLAN(s) are mapped to the tunnel(s):

```
CLI (network-admin@leaf-1)> tunnel-vxlan-add name VLE_L1_to_L5 vxlan
3001000
CLI (network-admin@leaf-5)> tunnel-vxlan-add name VLE_L5_to_L1 vxlan
3001000
```

The VXLAN tunnels and their mappings can be checked with the following command:

```
CLI (network-admin@leaf-1)> tunnel-vxlan-show vxlan 3001000
```

switch	name	vxlan
leaf-1	VLE_L1_to_L5	3001000
leaf-5	VLE_L5_to_L1	3001000

## Configuring Virtual Link Extension State Tracking

In addition to the above configuration steps, it is often common to also turn on state tracking in order to make sure that link state changes are mirrored:

```
CLI (network-admin@leaf-1)> vle-create name VLE-1 node-1 leaf-1 node-1-port 10 node-2 leaf-5 node-2-port 17 tracking
```

This configuration step requires specifying the two vLE nodes (leaf-1 and leaf-5) as well as their respective ports in addition to the tracking option by following this syntax:

vle-create	Create virtual link extension tracking
name name-string	Specify the VLE name
node-1 fabric-node name	Specify VLE node 1 name
node-2 fabric-node name	Specify VLE node 2 name
node-1-port node-1-port-number	Specify VLE node-1 port.
node-2-port node-2-port-number	Specify VLE node-2 port
[tracking no tracking]	Enable or disable tracking between VLE ports

The link state tracking configuration can be verified with the vle-show command like so:

```
CLI (network-admin@leaf-1)> vle-show
```

name	node-1	node-2	node-1-port	node-2-port	status	tracking
VLE-1	leaf-1	leaf-5	10	17	up	<b>enabled</b>

Furthermore, since a vLE transparently transmits PDUs over to the other end, it is possible to test state tracking by leveraging for example the LLDP protocol messages exchanged between two test switches (sw-1 and sw-2 in the figure above) connected to the vLE ports 10 and 17.

The two test switches exchange LLDP messages by default, so the LLDP peer relationship can just be shown with the command:

```
CLI (network-admin@sw-1) > lldp-show
```

switch	local-port	chassis-id	port-id	port-desc	sys-name
sw-1	1	0b0012c0	1	PN Switch Port(1)	sw-2

Then, if a vLE port is disabled (in this case, port 10), the peer relationship is lost and lldp-show does not display a neighbor anymore. The test switch's port also goes down thanks to vLE state tracking. Once the port is re-enabled, the neighboring relationship is re-established:

```
CLI (network-admin@leaf-5) > port-config-modify port 17 disable
```

```
CLI (network-admin@sw-1) > port-phy-show port 1
```

switch	port	state	speed	eth-mode	max-frame	learning	def-vlan
sw-1	1	<b>down</b>	10000	10Gbase-cr	1540	off	0

```
CLI (network-admin@sw-1) > lldp-show
```

```
CLI (network-admin@leaf-5) > port-config-modify port 17 enable
```

```
CLI (network-admin@leaf-5) > port-phy-show port 17
```

port	state	speed	eth-mode	max-frame	learning	def-vlan
17	<b>up</b>	10000	10G-SFI	1540	off	0

```
CLI (network-admin@sw-1) > port-phy-show port 1
```

switch	port	state	speed	eth-mode	max-frame	learning	def-vlan
sw-1	1	<b>up</b>	10000	10Gbase-cr	1540	off	1

```
CLI (network-admin@sw-1) > lldp-show
```

switch	local-port	chassis-id	port-id	port-desc	sys-name
sw-1	1	0b0012c0	1	PN Switch Port(1)	sw-2

To enable or disable tracking between existing vLE ports, use the `vle-modify` command:

```
CLI (network-admin@switch) > vle-modify name name-string tracking|no tracking
```

<code>vle-modify</code>	Modify virtual link extension tracking
<code>name name-string</code>	The vLE name to be modified
<code>tracking no tracking</code>	Enable or disable tracking between vLE ports

To delete a state tracking configuration, use the `vle-delete` command:

```
CLI (network-admin@switch) > vle-delete name name-string
```

## Configuring Virtual Link Extension Error Transparency

---

In order to be able to preserve bad CRC and undersized packets as part of the vLE transport, a new fabric-wide *error transparency* configuration option has been added:

```
CLI (network-admin@leaf-1) > vle-transparent-modify mode enable
```

```
CLI (network-admin@leaf-1) > vle-transparent-show
fabric: Transparent VLE mode is enabled
```

```
CLI (network-admin@leaf-1) > vle-transparent-modify mode disable
```

```
CLI (network-admin@leaf-1) > vle-transparent-show
fabric: Transparent VLE mode is disabled
```

However, since disabling CRC checks is not necessarily desirable on all links, some ports can be excluded (for example certain edge ports and inter-switch links) with a local scope command:

```
CLI (network-admin@leaf-1) > vle-transparent-modify mode enable
```

```
CLI (network-admin@leaf-1) > vle-transparent-port-remove port 18,45
```

```
CLI (network-admin@leaf-1) > vle-transparent-port-show
switch: leaf-1
leaf-1: ports removed from VLE transparent mode: 18,45
```

```
CLI (network-admin@leaf-1) > vle-transparent-port-add port 18,45
```

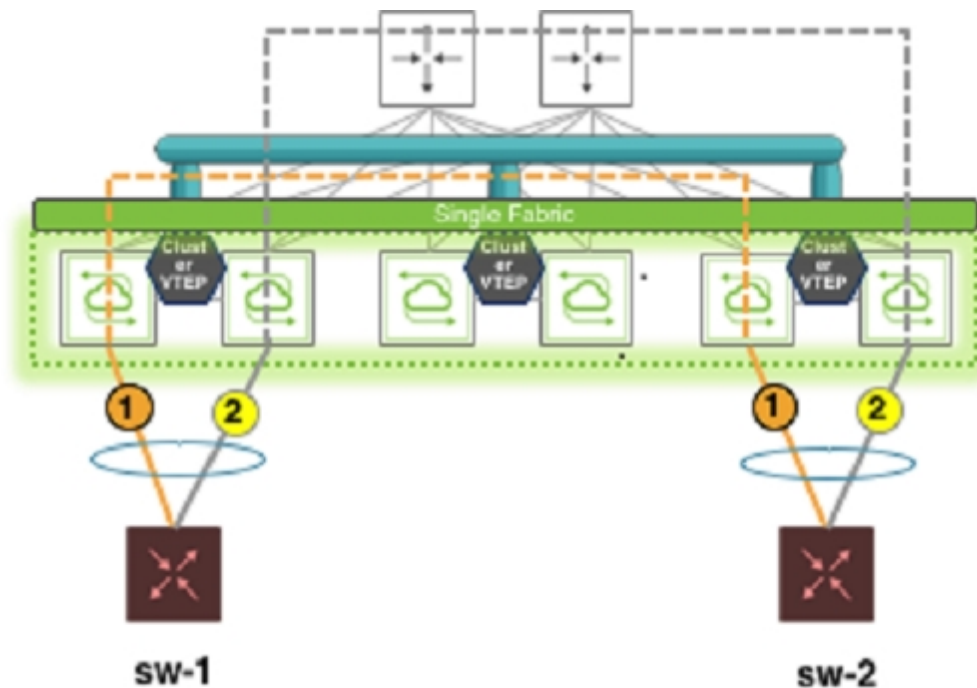
```
CLI (network-admin@leaf-1) > vle-transparent-port-show
switch: leaf-1
leaf-1: ports removed from VLE transparent mode: none
```

**Informational note:** Transparency exclusion should be applied to ports that are not in the end-to-end vLE transport path through the fabric, which needs to be based on supported switches that are capable of error transparency to preserve bad frames.



## Configuring Virtual Link Extension Redundancy with Clusters

The configuration of redundant vLE pseudo-wires requires the replication of the configuration steps described in the above sections. In addition, the endpoints (for example two switches as in the figure below) need to have link bundling enabled (static or dynamic LACP-based negotiation) in order to manage end-to-end path redundancy.



**Figure 9-5: Virtual Link Extension End-to-end Redundancy**

In this example, let us assume that both test switches, `sw-1` and `sw-2`, are configured with the LACP active negotiation option.

Then, every vLE configuration step needs to be replicated twice to create VLE-1 and VLE-2 (based on two transparent VLANs and four VTEPs):

1. Creation of Layer 3 VLANs that are used as tunnel end points
2. Set up of dedicated VXLAN connections
3. Creation of VXLAN-mapped *transparent VLANs*
4. Addition of the transparent VLAN's VXLAN ID to the allocated tunnels.

The distinctive configuration step is the creation of the VTEPs (step 2 above): VTEP HA is not required and not supported in case of vLE redundancy (namely, the virtual IPs cannot be used). The four VTEPs can be created instead with each cluster member's primary IP address like so:

```
CLI (network-admin@leaf-1) > tunnel-create scope local name VTEP1 vrouter-  
name vr-s1 local-ip 10.10.11.1 remote-ip 10.10.15.1
```

```
CLI (network-admin@leaf-2) > tunnel-create scope local name VTEP2 vrouter-  
name vr-s2 local-ip 20.20.22.1 remote-ip 20.20.26.1
```

```
CLI (network-admin@leaf-5) > tunnel-create scope local name VTEP3 vrouter-  
name vr-s3 local-ip 10.10.15.1 remote-ip 10.10.11.1
```

```
CLI (network-admin@leaf-6) > tunnel-create scope local name VTEP4 vrouter-  
name vr-s3 local-ip 20.20.26.1 remote-ip 20.20.22.1
```

Note that 10.10.11.1 and 10.10.15.1 are primary IP addresses used for VLE-1 on leaf-1 and leaf-5 respectively while 20.20.22.1 and 20.20.26.1 are primary IP addresses used for VLE-2 on leaf-2 and leaf-6.

The remaining four steps are the same as those described in the previous sections, replicated for both VLE-1 and VLE-2.

vLE tracking is also greatly beneficial in order to implement redundancy on Netvisor ONE nodes as it propagates link state changes and helps client side's LACP with timely link up/link down detection.

## Showing Virtual Link Extension Between Ports on the Same Switch

---

In certain network configurations, both vLE ports should be part of the same node. This case is supported by Netvisor ONE and is referred to as *local vLE*.

In the example below `node-1` and `node-2` refer to the same switch in the local vLE, in which port 30 is displayed as remote port in the `vle-show` output, even though both the vLE ports are on the same node.

Port 30 went down, therefore the vLE status is shown as `remote-down` (see command output below) so you can understand exactly which port caused the vLE to be down:

```
CLI (network-admin@switch) > vle-show
```

name	node-1	node-2	node-1-port	node-2-port	status	tracking
----	-----	-----	-----	-----	-----	-----
vle1	vanquish1	vanquish1	22	30	remote-down	enabled

You can use the `port-show` command to check if the `node-1-port` status is down or `vle-wait`. The `vle-wait` state indicates that the port was brought down due to the remote port being down.

## Configuring Keep-Alive Timeout for Virtual Link Extension

As part of the vLE link state tracking logic each node hosting a vLE endpoint sends *fabric fast keepalive* messages every second to its peer node that hosts the other endpoint.

A node considers its peer node down if a keepalive is not received from the remote node within 3 seconds (which is the default *vLE tracking timeout*). If that happens, the local port is brought down and the vLE state reflects both ports' down state.

However, in some deployments (for example with frequent link flaps), a three second timeout may not be enough to avoid false positives. Therefore, it can be modified with the following command:

```
CLI (network-admin@switch) > system-settings-modify vle-tracking-timeout seconds
```

You can configure a value *from 3 to 30 seconds*. (Note that the fast keepalive transmission frequency remains one second. Only the timeout value is adjusted to the configured value.) The configured value can be checked with the `system-settings-show` command.

**Note:** The default 3 second keepalive timeout is automatically configured when vLE link state tracking is enabled. When it is disabled, the standard fabric message timeout interval is 21 seconds as shown for the three leaf switches below that don't use tracking:

```
CLI (network-admin@switch) > fabric-node-show format name,keepalive-timeout
```

name	keepalive-timeout
leaf-1	21
leaf-2	21
leaf-3	21
switch	3

```
CLI (network-admin@switch) > system-settings-show format vle-tracking-timeout  
vle-tracking-timeout: 3
```

When tracking gets disabled, the fabric keepalive timeout goes back to 21 seconds:

```
CLI (network-admin@switch) > vle-modify name vLE no-tracking
```

```
CLI (network-admin@switch) > fabric-node-show format name,keepalive-timeout
```

name	keepalive-timeout
leaf-1	21
leaf-2	21
leaf-3	21
switch	21

```
CLI (network-admin@switch) > system-settings-show format vle-tracking-  
timeout,  
vle-tracking-timeout: 3
```

When it is necessary to change the administrative vLE keepalive timeout value, the fabric keepalive timeout reflects the new value when tracking is enabled, as shown below:

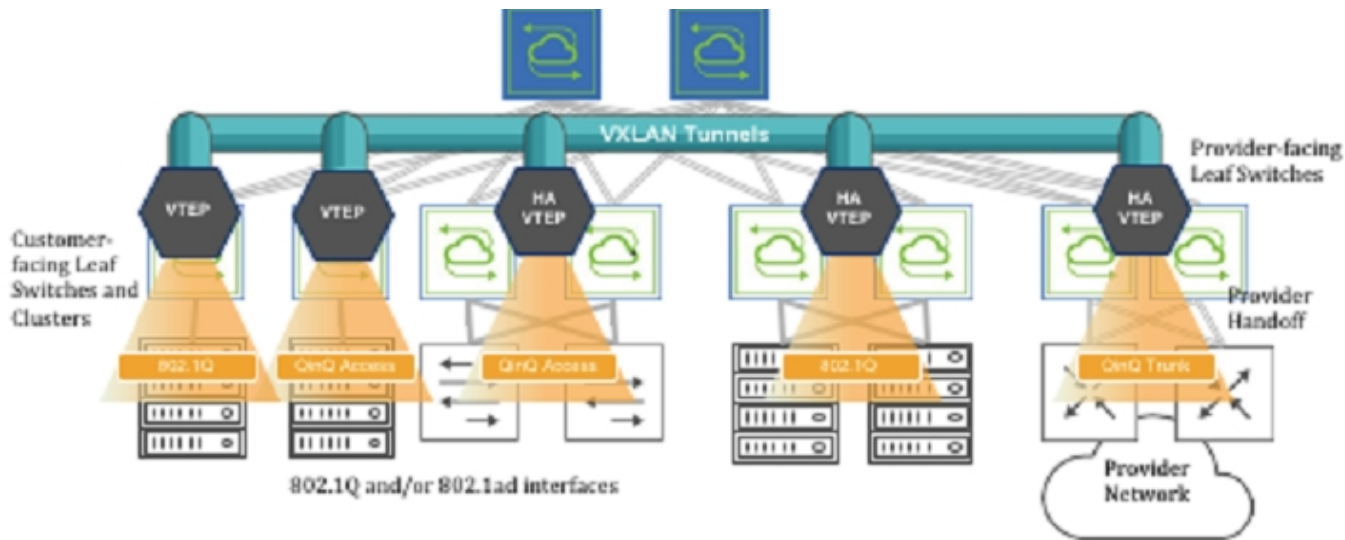
```
CLI (network-admin@switch) > system-settings-modify vle-tracking-timeout 30  
CLI (network-admin@switch) > fabric-node-show format name,keepalive-timeout
```

name	keepalive-timeout
leaf-1	21
leaf-2	21
leaf-3	21
switch	30

## Configuring VXLAN-based Bridge Domains for 802.1Q and QinQ Internetworking

Bridge domains' configuration process is based upon the deployment of a VXLAN transport as described in the *Configuring VXLAN* chapter of this guide.

Subsequently, switch ports can be selectively added to a BD to accept either 802.1Q frames or double-tagged 802.1ad frames based on the connected device(s) and/or network(s), as depicted in the **Figure 9-6**.



**Figure 9-6: VXLAN-based BD Internetworking**

Switch ports can be connected to either servers/hosts or switches/provider gateways as shown above.

In case of interconnection with an external cloud provider's gateway devices, as shown on the right hand side of **Figure 9-6**, the connectivity is based on double-tagged 802.1ad frames, as it is required to preserve all 16M (4094 x 4094) possible combinations of user IDs and service IDs.

On the other hand, connectivity to servers can be based on a single 802.1Q VLAN tag. This case is useful in private cloud networks for direct server-to-server connectivity (for example a Web services node that needs to communicate with a database node).

Alternatively, server NICs can be configured with both an outer VLAN and an inner VLAN ID. In such case, the inner VLAN represents for example the service implemented on such server. Certain network designs employ this double-tagging scheme to be able to scale the number of supported services to 16M. In this case, switch ports have to be configured in 802.1ad mode to preserve the inner tag.

The 802.1Q ports can also be used to connect to customer switches and traffic can be double tagged by the fabric leaf nodes themselves.

**Informational note:** BD ports can be configured in 802.1Q mode or in 802.1ad/QinQ mode but not in both on the same switch for the same BD.

Before adding ports to any of the aforementioned scenarios, the first basic configuration step is the creation of a VXLAN-based bridge domain entity with the command:

```
CLI (network-admin@leaf-5) > bridge-domain-create name bd1000 scope fabric
vxlan 10000 rsvd-vlan 4002
```

A bridge domain's scope can be `fabric`, `cluster`, or `local`.

The `bridge-domain-create` command uses a manually assigned VNI to uniquely identify the bridge domain at the hardware level.

Moreover, with `fabric` and `cluster` scope, it must set aside a VLAN ID as reserved for supporting *switch clusters*:

```
CLI (network-admin@leaf-5) > bridge-domain-create name bd1000 scope fabric
vxlan 10000 rsvd-vlan 4002
```

This command reserves VLAN 4002 on all *cluster switches* in the fabric for BD's internal use, so it can no longer be configured as a regular VLAN number. (The same requirement applies to `bridge-domain-create scope cluster` commands).

If for a particular pair of cluster switches the reserved VLAN has to be modified, on any one of the two switches you can use the following command:

```
CLI (network-admin@leaf-5) > bridge-domain-modify name bd1000 local-rsvd-
vlan 4006
```

Note that the above command requires both cluster switches to be rebooted to take effect. Before rebooting, both VLAN 4002 and 4006 are no longer available but only 4006 is reserved on the cluster switches after reboot.

Also note that in a cluster both switch members are required to use the same mode for a port, either QinQ or 802.1Q, for the same BD.

Bridge domains can also be created with a textual description to provide more context like so:

```
CLI (network-admin@leaf-5) > bridge-domain-create name bd1000 scope fabric
vxlan 10000 description business rsvd-vlan 4002
```

```
CLI (network-admin@leaf-5) > bridge-domain-show
```

name	scope	vxlan	auto-vxlan	description	ports	cluster-name
bd1000	fabric	10000	no	business	13	

The next configuration step is to set up the fabric overlay either with manual tunnel creation or with VTEP object configuration. The latter way is *preferred*.

In order to manually create a VXLAN tunnel, say, on cluster member leaf-5 in the **Figure 9-6**, the following commands can be used:

```
CLI (network-admin@leaf-5) > tunnel-create scope cluster name tunnel-5  
vrouter-name vr5 peer-vrouter-name vr1 local-ip 20.20.25.1 remote-ip  
20.20.27.1
```

```
CLI (network-admin@leaf-5) > trunk-modify name vxlan-loopback-trunk ports  
17 jumbo
```

The same steps are required on the other leaf switches to deploy an overlay comprising a full mesh of VXLAN-based interconnections.

Alternatively, VTEP objects can be configured so that a full mesh of VXLAN interconnections is brought up automatically without requiring manual configuration.

This is the command required to create a VTEP object on a non-clustered switch:

```
CLI (network-admin@leaf-1) > vtep-create name vtep1 vrouter-name vr1 ip  
20.20.25.1
```

which can be used to create each fabric termination point (vtep1, vtep2, etc. on each leaf switch).

In case of clustered switches using a common VIP, the following command pairs should be used:

```
CLI (network-admin@leaf-5) > vtep-create name vtep5 vrouter-name vr5 ip  
103.103.103.250 virtual-ip 103.103.103.10
```

```
CLI (network-admin@leaf-6) > vtep-create name vtep6 vrouter-name vr6 ip  
103.103.103.251 virtual-ip 103.103.103.10
```

The next configuration step is to associate the BD's VXLAN ID (e.g., 10000) to the newly created tunnel(s) or VTEP objects. For example, this command implements the association with a tunnel:

```
CLI (network-admin@leaf-1) > tunnel-vxlan-add name tunnel-1 vxlan 10000
```

whereas this command implements the association with a VTEP object:

```
CLI (network-admin@leaf-1) > vtep-vxlan-add name vtep1 vxlan 10000
```

Once the VXLAN overlay deployment is complete and the BD's VXLAN ID is associated to it, then it is possible to start adding switch ports.

**Informational note:** When a port is assigned to a BD (as in the examples below) it cannot be associated with a regular VLAN configured with the `vlan-create` command, and vice versa.

In order to add a port connected to the cloud provider network to the BD, you can use the following command that explicitly maps both inner and outer VLANs to the BD:

```
CLI (network-admin@leaf-7) > bridge-domain-port-add name bd1000 port 13
```



```
outer-vlan 13 inner-vlan 100
```

This configuration also means that a frame's MAC lookup in the Layer 2 table is performed including both the outer VLAN and the inner VLAN.

The same configuration can be used to add a port connected to an 802.1ad-configured server to the BD:

```
CLI (network-admin@leaf-2) > bridge-domain-port-add name bd1000 port 11
outer-vlan 13 inner-vlan 100
```

In order to add a port connected to an 802.1Q network to the BD, you can use the following command that explicitly maps only the outer VLAN to the BD (and preserves the inner VLAN):

```
CLI (network-admin@leaf-2) > bridge-domain-port-add name bd1000 port 15
outer-vlan 19
```

This configuration also means that a frame's MAC lookup in the Layer 2 table is performed including only the outer VLAN while the inner VLAN is ignored.

In order to add a port connected to an 802.1Q server to the BD, you can use the following command that explicitly associates one or more VLANs on the same switch port to the same BD:

```
CLI (network-admin@leaf-1) > bridge-domain-port-add name bd2100 port 10
vlangs 210,310
```

Note that multiple VLAN IDs on the same port can be part of the same bridge domain. However, VLAN IDs added to a BD cannot be reused with another BD on the same port. Also note that MAC addresses associated with these two VLAN IDs must be unique within the BD.

Finally, you should also configure all cluster ports/trunks and ports in QinQ mode to work with the 802.1ad TPID (i.e., 0x88A8) with the following command:

```
CLI (network-admin@switch) > port-config-modify port <port_number> allowed-
tpid q-in-q
```

Port removal from a BD and BD deletion follow the reverse order compared to the addition process described in the above steps.

Ports have to be removed first with the command:

```
CLI (network-admin@leaf-2) > bridge-domain-port-remove name bd1000 port 15
```

Then a BD's VXLAN ID may be manually removed from the associated tunnel(s):

```
CLI (network-admin@leaf-2) > tunnel-vxlan-remove name tunnel-2 vxlan 10000
```

or from the corresponding VTEP objects like so:

```
CLI (network-admin@leaf-2) > vtep-vxlan-remove name vtep2 vxlan 10000
```

Finally, the BD can be deleted with the following command:

```
CLI (network-admin@leaf-2) > bridge-domain-delete name bd1000
```

## Troubleshooting vLE

---

To troubleshoot vLE link state tracking, refer to the examples in the *Configuring Virtual Link Extension State Tracking* section. In particular, for both individual vLE pseudo-wires and redundant vLE trunks protocol messages can be leveraged to verify end-to-end connectivity (or lack thereof). LLDP messages (and LACP messages in case of trunks) show the neighboring port ID(s) and reflect the up/down connectivity status of the pseudo-wire(s). Therefore they are very valuable troubleshooting tools.

With vLE link state tracking enabled, a vLE pseudo-wire can go down (i.e., in vLE-wait state) when the remote end of the vLE goes down. However, in some cases, it may be possible that a “false positive” event occurs in which the local vLE port goes down even if the remote end is *physically up*. This could happen if the vLE tracking logic incorrectly believes the remote end to be unreachable. In such case, you can disable vLE tracking, which should bring the local port back up and allow you to check if the traffic is flowing through the vLE pseudo-wire correctly. Increasing the vLE tracking timeout can help to avoid false positives by giving more leeway to the link state tracking software, especially when experiencing frequent link flaps or CPU overload conditions (which could affect the tracking logic negatively).

## Guidelines and Limitations

---

This is a list of guidelines and limitations to keep in mind when configuring bridge domains:

- Currently on bridge domain ports the QoS default behavior is as follows:
  - Incoming 802.1Q frames' CoS value (a.k.a. PCP) is preserved along with the VLAN tag when transported in the VXLAN payload
  - Incoming 802.1ad frames' CoS value from the inner tag is preserved along with the inner VLAN tag when transported in the VXLAN payload
  - For outgoing 802.1Q frames after VXLAN decapsulation *CoS is set to value 0*
  - For outgoing 802.1ad frames after VXLAN decapsulation the CoS value carried in the VXLAN packet is preserved and becomes the CoS value of the inner tag. The outer tag instead *carries a CoS value of 0*.
- Currently on the Dell S5232-ON and Dell S5248-ON switch models the QinQ and untagged modes is *not* supported while the 802.1Q mode is allowed.
- On ports added to a BD, the STP protocol and the IGMP snooping features are not supported.
- Routing on BDs is not supported.
- vNETs are not supported with BDs
- The LLDP and LACP protocols are supported on BD ports configured in QinQ mode. The trunking and vLAG functionalities are supported too.
- The optimized ARP and optimized ND features cannot coexist with BDs. They should be set to off in the `system-setting-show` output. If the user attempts to enable the feature, a conflict message is printed as below:

```
CLI (network-admin@switch) > system-settings-modify optimize-arps
system-settings-modify: Cannot turn on optimized ARP as bridge-domain Q-in-Q is configured
```

When optimized ARP/ND is already configured before a port is added to a BD, a similar conflict message is printed:

```
CLI (network-admin@switch) > bridge-domain-port-add name bd1000 port 18
outer-vlan 18 inner-vlan 100
bridge-domain-port-add: Cannot add port in Q-in-Q mode to bridge-domain as
optimized ARP or ND configured
```

- The flowtrace function is not supported with VXLAN and BDs.
- The VLAN ID is not shown in the `l2-table-show` output when a L2 entry is associated to a cluster tunnel:

```
CLI (network-admin@leaf1) > l2-table-show bd bdl format
switch,mac,bd,vlan,vxlan,inrf,ports,tunnel
```

switch	mac	bd	vlan	vxlan	intf	ports	tunnel
leaf1	00:12:c0:80:1c:09	bd1	100	101001	274	82	
ursa-scale-leaf2	00:12:c0:80:1c:09	bd1	100	101001	274	82	
ursa-scale-leaf1	00:12:c0:80:40:52	bd1		101001			auto-tunnel-10.40.40.1_10.30.40.1
ursa-scale-leaf2	00:12:c0:80:40:52	bd1		101001			auto-tunnel-10.40.40.1_10.30.40.1
ursa-scale-leaf4	00:12:c0:80:40:52	bd1	100	101001	81	81	

- When configuring vLE redundancy, if LACP is used for end-to-end connectivity checks (which is usually preferable), a vLE trunk can be set up only with multiple individual virtual pipes using separate VTEPs (e.g., 8 in case of a 4-way vLE trunk). When using static trunks, instead, it is also possible (but generally less desirable) to set up, say, a 4-way vLE trunk with 2 2-way trunks as endpoints (which equates to a 4 x 2-port trunk configuration on leaf switches and a 4-way trunk configuration on end devices).

## Supported Releases

Command/Parameter	Netvisor ONE Version
<code>vxlan-mode standard transparent</code>	Parameter added in version 2.5.2
<code>vle-create, vle-delete, vle-modify, vle-show</code>	Command added in version 2.5.4
<code>vle-tracking-timeout</code>	Parameter added in version 3.0.0
<code>vle-transparent-modify, vle-transparent-show, vle-transparent-port-remove, vle-transparent-port-show, vle-transparent-port-add</code>	Commands introduced in version 5.1.1
<code>port-config-modify allowed-tpid q-in-q</code>	Parameter introduced in version 5.2.0
<code>bridge-domain-create, bridge-domain-rsvd-vlan-add, bridge-domain-port-add</code>	Commands introduced in version 5.2.0
<code>l2-table-show bd bd_name</code>	Parameter introduced in version 5.2.0

Please also refer to the Pluribus Command Reference document.

## Related Documentation

---

For further information on concepts mentioned in this section (such as Layer 2, Layer 3, etc.), refer to these sections of the Configuration Guide:

- [Configuring Switch Ports](#)
- [Configuring Layer 2 Features](#)
- [Configuring Layer 3 Features](#)
- [Configuring and Administering the Pluribus Fabric](#)
- [Configuring High Availability](#)
- [Configuring VXLAN](#)

Also, refer to the *VirtualWire Deployment* guide.

## Configuring and Using Network Management and Monitoring

---

This chapter provides information for understanding and using the Pluribus Networks Netvisor ONE command line interface (CLI) on a Netvisor ONE switch to Administer Switches and Fabrics.

---

- [Displaying System Statistics on a Switch](#)
  - [Understanding and Configuring Port Mirroring](#)
    - [Configuring Port Mirroring](#)
    - [Configuring Port Mirroring to a Remote Host](#)
  - [Configuring Logging](#)
    - [Sending Log Messages to Syslog Servers](#)
    - [Forwarding Log Files to an External Linux Server](#)
    - [Saving Diagnostic Files & Exporting to an External Server](#)
    - [Using Facility Codes with Log Messages](#)
    - [Displaying Log Counters Information](#)
    - [Viewing Log Events](#)
    - [Exceptions for Audit Logging](#)
  - [Understanding and Configuring SNMP](#)
-



## Displaying System Statistics on a Switch

---

You display system statistics on a server-switch using the `system-stats-show` command:

```
CLI (network-admin@Leaf1) > system-stats-show layout vertical
```

```
switch:           Leaf-ONVL
uptime:           1h22m26s
used-mem:          27%
used-swap:         0%
swap-scan:         0
cpu-user:          0%
cpu-sys:           1%
cpu-idle:          98%
```

The `swap-scan` output displays the number of scans performed on the swap. A nonzero number indicates that memory is paged from the physical memory (RAM) to virtual memory (disk or swap). A consistently high value indicates that all memory, both physical and virtual, is exhausted and the system may stop responding.

## Understanding and Configuring Port Mirroring

---

The port mirroring feature enables you to gain visibility into the traffic flowing through a network and assist in troubleshooting connectivity issues and monitoring the performance of network devices. By using the port mirroring feature, Netvisor ONE copies traffic from specific ports on a switch to other ports on the same switch or a different switch.

To analyze the traffic, you must configure a destination port for the mirrored traffic where the network analyzer tools are connected. For example, when an application needs access to data flowing through the switch, but does not want to impede the flow, the port mirroring feature can provide the configuration required by this application.

Netvisor ONE supports the following port mirroring functionalities:

- **Local Switched Port Analyzer (SPAN) mode:** This is a simple port mirroring configuration, which copies traffic from one or more source ports to one or more destination ports on the same switch.
- **Remote Switched Port Analyzer (RSPAN) mode:** An extension of SPAN, which copies packets from and between different switches by using a user-specified VLAN to carry the traffic. This feature enables remote monitoring of multiple switches.

## Configuring Port Mirroring

You can create a port mirror and configure the parameters using the `mirror-create` command.

CLI (network-admin@switch) > `mirror-create`

<code>mirror-create</code>	Create mirrored ports.
<code>name name-string</code>	Specify a mirror name.
Specify the following options:	
<code>direction [ingress egress bidirectional]</code>	Specify the direction of the traffic on the source port to be mirrored. Use this option to mirror the traffic that is received on source ports or traffic that leaves source ports, or both.
<code>out-port port-list</code>	Specify one or more outgoing traffic ports.
<code>out-trunk trunk name</code>	Specifying outgoing traffic trunk (link aggregation). <code>out-trunk</code> option load balances the outgoing traffic among trunk ports. You can either configure an <code>out-port</code> or an <code>out-trunk</code> .
<code>in-port port-list</code>	Specify one or more incoming traffic ports. The in-ports can overlap among other mirror instances.
<code>filtering [port vflow-and-port vflow-or-port]</code>	Specify the traffic filter policy. With <code>vflow-and-port</code> policy, only if a packet matches both the vFlow and the <code>in-port</code> for the mirror will it get mirrored. With <code>vflow-or-port</code> policy, the packet gets mirrored if it matches either the vFlow or the <code>in-port</code> of the mirror.
<code>enable disable</code>	Enable or disable the mirror. A mirror, once created, is enabled by default.
<code>other-egress-out [allow prevent]</code>	Specify to allow or prevent switching of other traffic to <code>out-port</code> . The default status is <code>prevent</code> .
<code>span-encap [none over-vlan]</code>	Specify the mirror encapsulation type. Specify <code>over-vlan</code> to enable RSPAN. The default is <code>none</code> .
<code>span-local-ip ip-address</code>	Specify the local IPv4 address.
<code>span-remote-ip ip-address</code>	Specify the remote IPv4 address.
<code>span-src-mac mac-address</code>	Specify the source MAC address for the mirror.
<code>span-dst-mac mac-address</code>	Specify the destination MAC address for the mirror.
<code>span-tagging-vlan vlan-id</code>	Specify the mirror SPAN tagging VLAN ID. This VLAN carries the traffic in RSPAN configuration.

```
span-tos 0..255
```

Specify the mirror SPAN Type of Service (ToS) as a value between 0 and 255.

```
nvie-mirror|no-nvie-mirror
```

Specify to mark/unmark this mirror as an NVIE mirror used to mirror traffic to NVIE virtual machines.

## Configuring Local SPAN

For Local SPAN, the in-port and out-port are on the same switch. For example:

```
CLI (network-admin@switch) > mirror-create name mir1 direction ingress in-
port 10 out-port 15
```

Netvisor ONE defines a mirror configuration, but does not add any traffic into that mirror. A sniffer tool like Wireshark can be used to capture and analyze the mirrored traffic at the destination port. You can modify a mirror configuration using the `mirror-modify` command. To view the details of a mirror configuration that you had created already, use the `mirror-show` command.

For example, if you had created the following mirror configuration:

```
CLI (network-admin@switch) > mirror-create name test direction
bidirectional out-port 10 in-port 15
```

The details of the configuration can be viewed using the command:

```
CLI (network-admin@switch) > mirror-show
```

name	direction	out-port	in-port	filtering	enable	other-egress-out	nvie-mirror
test	bidirectional	10	15	port	yes	prevent	false

To modify the above configuration, use the command:

```
CLI (network-admin@switch) > mirror-modify name test out-port 20
```

To view the modified configuration, use the command:

```
CLI (network-admin@switch) > mirror-show
```

name	direction	out-port	in-port	filtering	enable	other-egress-out	nvie-mirror
test	bidirectional	20	15	port	yes	prevent	false

Use the following command to modify a configuration and setup mirroring to send traffic from a range of data ports to multiple destination SPAN ports.

```
CLI (network-admin@switch) > mirror-modify mir25 in-port 1-5 out-port 50-52
```

To disable the configuration, use the following command:

```
CLI (network-admin@switch) > mirror-modify mir25 in-port 1-5 out-port 50
```

disable

By default, a port configured as out-port of a mirror only functions as egress port for mirrored traffic. The out-port does not allow transit traffic to flow through which, in certain cases, can lead to traffic black holing. To overcome this problem, the out-port may be configured to allow other egress traffic. For example:

```
CLI (network-admin@switch1) > mirror-create name mir20 direction ingress  
in-port 81 out-port 86 other-egress-out allow
```

## Configuring Multiple Port Mirrors

Netvisor ONE supports the creation of multiple mirrors based on the platforms used and at a time, you can configure up to four mirrors on a switch.

For example:

```
CLI (network-admin@switch) > mirror-create name rule1 in-port 1,2 out-  
port 50,53 span-encap over-vlan span-tagging-vlan 50
```

```
CLI (network-admin@switch) > mirror-create name rule2 in-port 3,4 out-  
port 51-53 span-encap over-vlan span-tagging-vlan 50
```

```
CLI (network-admin@switch) > mirror-create name rule3 in-port 5,6 out-  
port 51,52 span-encap over-vlan span-tagging-vlan 50
```

```
CLI (network-admin@switch) > mirror-create name rule4 in-port 7,8 out-  
port 53 span-encap over-vlan span-tagging-vlan 50
```

## Configuring Remote Port Mirroring

---

### Configuring VLAN-based Remote Port Mirroring

Remote port mirroring copies traffic between different switches and the mirrored traffic is carried over a specified VLAN. This functionality is also known as RSPAN.

To configure remote port mirroring over a VLAN, you must first create a mirror instance on the source switch (switch1 in the below CLI). You must also specify `over-vlan` as encapsulation scheme. For example:

```
CLI (network-admin@switch1) > mirror-create name mir2 direction ingress in-  
port 81 out-port 86 other-egress-out allow span-encap over-vlan span-  
tagging-vlan 200
```

This command tags all ingress packets on port 81 with VLAN 200 and these packets are sent out on port 86. If VLAN 200 is not configured on switch2, the packets are dropped at port 3. Thus, VLAN 200 must be configured on port 3 and port 10 of switch2 as shown below.

```
CLI (network-admin@switch2) > vlan-create id 200 scope local ports 3,10
```

The alternate method is to configure a mirror on switch2 with a VLAN 200 tagging.

For example:

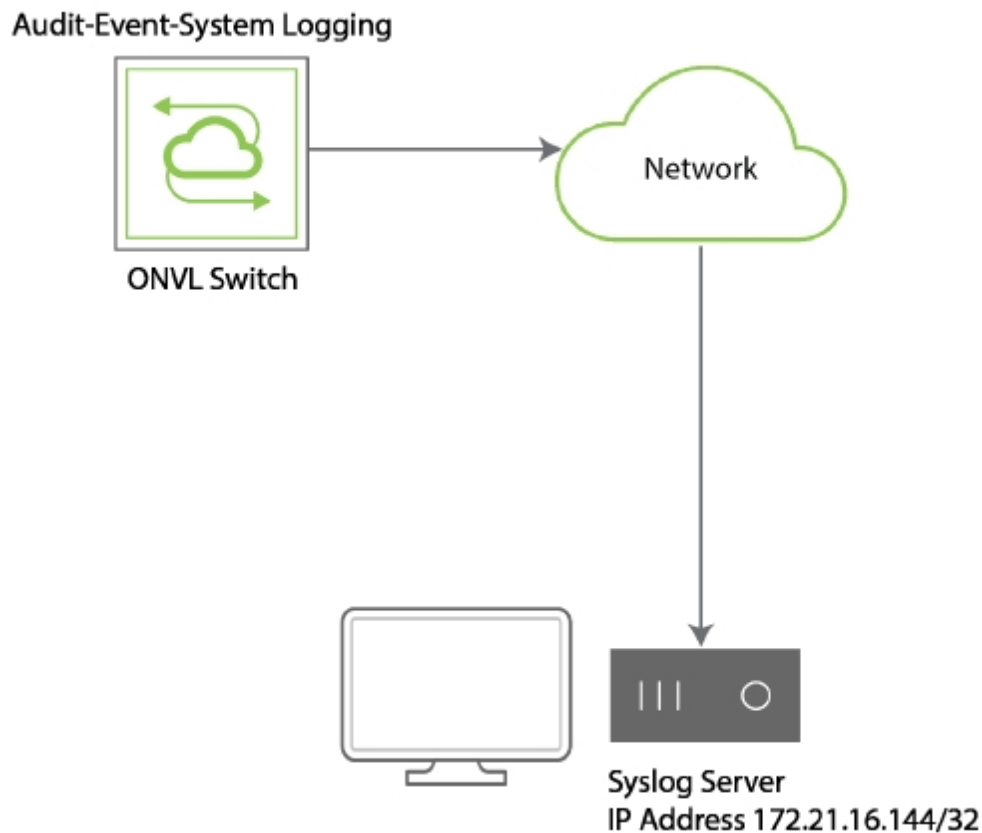
```
CLI (network-admin@switch2) > mirror-create name mir3 in-port 3 out-port 10  
span-encap over-vlan span-tagging-vlan 200
```

With this configuration, the mirrored packets can be analyzed at port 10 of switch2 by using a packet analyzer tool.

**Note:** If the path for the mirrored packets to reach the destination include multiple switches, the SPAN tagging VLAN must be configured on all the intermediate switches.

## Configuring Logging

Netvisor One logs all important activities that occur on the switch and fabrics created on them. Logging is enabled by default and is viewable using the CLI. You can also configure system logging to send syslog-formatted messages to other servers configured to receive them as part of centralized logging and monitoring.



**Figure 10-1 - Netvisor One Switch with Syslog Server**

The following types of activities are logged:

Log Type	Description
Event	<p>Records action observed or performed by switches. Each Event type can be enabled or disabled. Events are collected on a best effort basis. If events occur too rapidly to be recorded, the event log is annotated with the number of events lost.</p> <p>The following are examples of event types:</p> <ul style="list-style-type: none"> <li>• Port state changes</li> <li>• TCP connections</li> </ul>

Audit	<ul style="list-style-type: none"> <li>• STP port changes</li> </ul> <p>When an administrative change to the configuration is made, an audit log is recorded. An audit log consists of the command and parameters along with the success or failure indication. When a command fails, an error message is also recorded.</p> <p>The system log records error conditions and conditions of interest.</p> <p>There are four levels in the system log:</p>
System	<ul style="list-style-type: none"> <li>• critical</li> <li>• error</li> <li>• warn</li> <li>• note</li> </ul>
Error	<p>The error log records messages on standard error output, describing the last error encountered.</p>

Each log message includes the following information:

- Category - event, audit, or system
- Timestamp within a microsecond
- Process name and process ID of the process producing the message
- Unique message name
- Unique five digit numerical message code
- Message: additional message-specific parameters and explanation

A log message may include optional parameters, including associated VLAN, VXLAN, or switch port. An audit log message includes additional information:

- User
- Process ID
- Client IP of the remote computer issuing the command

An event log also includes the event type.

The maximum number of repeated messages detected by Netvisor ONE is ten (10). After five seconds, if Netvisor ONE detects repeated messages, then the log prints "Last X message(s) repeated Y time(s)". If the log message detects "X" and "Y" as both 1, then Netvisor ONE prints the message rather than "Last 1 message(s) repeated 1 time(s)". Netvisor ONE prints the log events after a five (5) second delay.



To view event logs using the CLI, enter the following command:

```
CLI (network-admin@Leaf1) > log-event-show
```

```
category time name code event-type port message
event 2013-06-04,13:12:18.304740 port_up 62 port 62 up
event 2013-06-04,13:12:18.304740 port_up 62 port 50 up
event 2013-06-04,13:12:18.304740 port_up 62 port 10 up
...
```

To view audit log entries, enter the following command:

```
CLI (network-admin@Leaf1) > log-audit-show
```

```
category time name code user message
-----
audit 2013-06-04,13:12:18.304740 command 1101 network-admin Command create id=b000011:! name=1
scope=fabric vrg=b000011:0 vlans=100 _mgr_id=b00001
audit 2013-06-04,13:12:18.304740 command 1101 network-admin Command create vrouter id=b000011:! name=1
scope=fabric vrg=b000011:0 vlans=100 _mgr_id=b00001
```

To view system log entries, use the following command:

```
CLI (network-admin@Leaf1) > log-system-show
```

```
time: 2013-09-17, 06:28:09.351514-07:00
name: 11006
level: warn
time: 2013-09-17, 11:28:09.351514-07:00
name: 11006
level: warn
time: 2013-09-17, 13:28:09.351514-07:00
name: 11006
level: warn
```

Currently, accessing system log information may require assistance from TAC to retrieve the logs from Netvisor One. To enable log auditing in Netvisor One, use the following command:

```
CLI (network-admin@Leaf1) > log-admin-audit-modify enable|disable
```

To display auditing status, use the following command:

```
CLI (network-admin@Leaf1) > log-admin-audit-show
```

## Modifying and Displaying Log Event Settings

By default, only system and port events are logged. Other logging is possible, and you can add other events using the `log-event-settings-modify` command. You can modify the way Netvisor One logs events by using the `log-event-settings-modify` command to remove or add log events.

For instance to remove logging of STP events, use the following command:

```
CLI (network-admin@Leaf1) > log-event-settings-modify no-stp
```

To display log event settings information, use the `log-event-settings-show` command.

## Sending Log Messages to Syslog Servers

---

To configure the switch to send all log messages to a syslog server with an IP address of 172.16.21.67, use the following command:

```
CLI (network-admin@Leaf1) > admin-syslog-create name log-all scope fabric  
host 172.16.21.76
```

To display the configuration use the `admin-syslog-show` command:

```
CLI (network-admin@Leaf1) > admin-syslog-show
```

name	scope	host	port	message-format
----	-----	-----	----	-----
log-all	fabric	172.16.21.67	514	legacy

To specify sending the syslog messages in structured format, per RFC5424, add the `message-format` option to the configuration.

```
CLI (network-admin@Leaf1) > admin-syslog-modify name log-all message-format  
structured
```

You can also modify the port that the service listens on to another port. More than one syslog listening service can be configured and appropriate syslog messages are sent to each one.

By default, all log messages are forwarded to syslog servers. To filter the log messages, use the `msg-level` option to specify the severity or other options:

```
CLI (network-admin@Leaf1) > admin-syslog-match-add syslog-name log-all name  
critical-msgs msg-level critical
```

You can modify syslog matching using the `admin-syslog-match-modify` command, or remove matching criteria using the `admin-syslog-match-remove` command.

To display the configuration, use the `show` command:

```
CLI (network-admin@Leaf1) > admin-syslog-match-show
```

syslog-name	msg-level	name
-----	-----	-----
log-all	critical	critical-msgs

## Forwarding Log Files to an External Linux Server

**Note:** Netvisor ONE supports only one external server for TCP-TLS export. The UDP syslog export can be done to more than one server.

You can forward log files to an external Linux server and encrypt them using Transport Layer Security (TLS) over Transmission Control Protocol (TCP).

The command, `admin-syslog-create` can be used to configure exporting logs using TCP-TLS. You can perform the below steps to configure exporting of logs:

```
CLI (network-admin@Leaf1) > admin-syslog-create name audit-logs scope local
host 172.16.21.33 transport tcp-tls port 10514
```

You can create TLS certificates using the following command:

```
CLI (network-admin@Leaf1) > syslog-tls-cert-request-create country US state
CA city Palo Alto organization QA organizational-unit engineering common-
name pluribusnetworks.com
```

This command creates a Certificate Signing Request (CSR) and places it in the directory `/sftp/export` used by Netvisor One. You must copy and the CSR to the CA server and sign it.

To import the signed certificate to Netvisor One, you must copy the certificate and the `ca.pem` file to `/sftp/import` directory in Netvisor One.

Use the following command to import the files:

```
CLI (network-admin@Leaf1) > syslog-tls-cert-import file-ca ca.pem file-cert
my-cert.pem
```

To enable TLS-TCP logging export, use the following syntax:

```
CLI (network-admin@Leaf1)>admin-syslog-create name audit-logs scope local
host 172.16.21.33 transport tcp-tls port 10514
```

To display the export information, use the `admin-syslog-show` command:

```
CLI (network-admin@Leaf1) > admin-syslog-show
```

switch	name	scope	host	port	transport	message-format
-----	-----	-----	-----	-----	-----	-----
leaf-pst-1	MYTLS	local	172.21.16.33	10514	tcp-tls	legacy

### Other new commands

<code>syslog-tls-cert-clear</code>	Clears the certificates
------------------------------------	-------------------------

syslog-tls-cert-request-show  
syslog-tls-cert-show  
config

Displays the certificate information  
Displays syslog TSL import certificate

## Saving Diagnostic Files and Exporting to an External Server

---

- 1) Use the `save-diags export-sftp` command.
- 2) The signed \*.tar file is saved to `/sftp/export` directory in Netvisor One.
- 3) Enable SFTP on the switch using the `admin-sftp-modify-enable` command.
- 4) Copy the file to the external server using SFTP to the Netvisor One switch.

## Using Facility Codes with Log Messages

---

Netvisor ONE labels log messages with a facility code indicating the area of the software that generated the log message.

Netvisor One uses the following facility codes by default:

- `Log_Daemon` for events and system messages
- `Log_AUDIT` for audit messages

The following severity levels are used by default:

- `Log_INFO` for events and audit messages
- `Log_Critical` = critical
- `Log_ERROR` = error
- `Log_WARNING` = warn
- `Log_NOTICE` = note

You can override the default values by configuring matches for each syslog configuration which allows Netvisor One to translate log messages into fields that the syslog servers understand.

## Displaying Log Counters Information

---

You can display information about the number of events that have occurred on the network by using the `log-system-counters-show` command:

```
CLI (network-admin@Leaf1) > log-system-counters-show layout vertical
```

```
switch:      pleiades24
critical:    0
error:       0
warn:       1061
note:       9
```

To reset the log counters, use the `log-system-counters-reset` command.

### Formatting and Filtering of Logging Messages

Netvisor ONE allows many options for filtering and formatting of log messages returned by these commands. Use the <tab> completion method and ? to explore them.

You can access the log files using SFTP, `switch-ip:/sftp/ONVL/logs` and NFS, `/net/switch-name/ONVL/logs`, if you have enabled the services.

Many systems support a syslog facility for sending or receiving log messages.

The infrastructure can send messages to syslog servers using either RFC 5424 (Structure) or RFC 3164 (legacy) formats.



## Viewing Log Events

---

For information about specific log events and their meaning, see the *Log Message Reference Guide*.

A log message consists of common parameters separated by spaces and a colon (:), and optional parameters such as key and value pairs, another colon, and then the log-specific message.

To view event logs using the CLI, enter the following command:

```
CLI (network-admin@Leaf1) > log-event-show
```

```
category:      event
time:          2014-07-17,07:37:17.466173-07:00
switch:        pleiades24
program:        nvOSd
pid:           6344
name:           mac_ip_changed
code:           11023
event-type:     port
                : global-default
port:          65
vlan:          200
message:        ip address change: mac=50:33:a5:e0:7f:fd ip=172.16.23.7
category:      event
time:          2014-07-17,07:37:50.109133-07:00
switch:        pleiades24
program:        nvOSd
pid:           6344
name:           mac_ip_changed
code:           11023
event-type:     port
                : vlb-web-svr
port:          65
vlan:          200
message:        ip address change: mac=50:33:a5:e0:7f:fd ip=172.16.23.1
category:      event
time:          2017-05-05,07:42:17.418349-07:00...
```

To view audit log entries, enter the following command:

```
CLI (network-admin@Leaf1) > log-audit-show layout vertical
```

```
category:      audit
time:          2017-04-01,14:56:40.763626-07:00
name:          user_command
code:          11001
user:          network-admin
message:        Command "vlan-create id 25
category:      audit
```

```
time:      2017-04-01,14:56:40.765839-07:00
name:      logout
code:      11100
user:      network-admin
message:   logout
category:  audit
time:      2017-04-01,14:56:40.847912-07:00
name:      login
code:      11099
user:      network-admin
message:   login
category:  audit
time:      2017-04-01,14:56:40.888363-07:00
name:      logout
code:      11100
...
```

To view system log entries, use the following command:

```
CLI (network-admin@Leaf1) > log-system-show
```

```
time:      2013-09-17, 06:28:09.351514-07:00
name:      11006
level:     warn
time:      2013-09-17, 11:28:09.351514-07:00
name:      11006
level:     warn
time:      2013-09-17, 13:28:09.351514-07:00
name:      11006
level:     warn
```

## Exceptions for Audit Logging

Use the commands `log-audit-exception-create`, `log-audit-exception-delete`, and `log-audit-exception-show` are used to control which CLI, shell and vtysh commands are subject to auditing.

If Netvisor ONE subjects a command to auditing, Netvisor ONE logs the command in the audit log and sends it to the TACACS+ server as authorization and accounting messages.

CLI (network-admin@Spine1) > `log-audit-exception-create`

	Create an audit logging exception.
<code>cli shell vtysh</code>	Specify the type of audit exception.
<code>pattern <i>pattern-string</i></code>	Specify a regular expression to match exceptions.
<code>any read-only read-write</code>	Specify the access type to match exceptions.
<code>scope local fabric</code>	Specify the scope of exceptions.

CLI (network-admin@Spine1) > `log-audit-exception-delete`

	Delete an audit logging exception.
<code>cli shell vtysh</code>	Specify the type of audit exception.
<code>pattern <i>pattern-string</i></code>	Specify a regular expression to match exceptions.
<code>any read-only read-write</code>	Specify the access type to match exceptions.

CLI (network-admin@Spine1) > `log-audit-exception-show`

	Display audit logging exceptions.
<code>cli shell vtysh</code>	Display the type of audit exception.
<code>pattern <i>pattern-string</i></code>	Display a regular expression to match exceptions.
<code>any read-only read-write</code>	Display the access type to match exceptions.
<code>scope local fabric</code>	Display the scope of exceptions.

By default, Netvisor ONE audits every command except for read-only CLI commands and `^/usr/bin/nvmore` which is the pager for the Netvisor ONE CLI:

CLI (network-admin@switch) > `log-audit-exception-show`

```
switch type  pattern                access  scope
```

```
-----
switch cli                                read-only local
switch shell ^/usr/bin/nvmore any        local
```

To enable auditing of ALL CLI commands, you can delete the `cli/read-only` exception:

```
CLI (network-admin@switch) > log-audit-exception-delete cli read-only
```

## Modifying User Roles

You can add privileges to a user by adding new parameters available for roles. To add shell access to a user's role, use the following syntax:

```
CLI (network-admin@switch) > role-create
```

<code>name</code> <i>name-string</i>	Specify a name for the user role.
<code>scope</code> <code>local fabric</code>	Specify a scope for the user role.
One or more of the following options:	
<code>access</code> <code>read-only read-write</code>	Specify the type of access for the user role. The default is read-write.
<code>running-config</code> <code> no-running-config</code>	Specify if the user role allows access to the switch running configuration.
<code>shell</code> <code> no-shell</code>	Specify if the user role allows access to the shell.
<code>sudo</code> <code> no-sudo</code>	Specify if the user role allows the sudo command.

The new parameters are also available for the `role-modify` command.

## Understanding and Configuring SNMP

---

This chapter provides information for understanding and configuring SNMP services on Netvisor One switches using the Pluribus Networks Netvisor One command line interface (CLI) .

---

- [Overview](#)
  - [Configuring SNMP](#)
  - [Creating SNMP Communities on SNMP V1 and V2](#)
  - [Creating SNMP Users on SNMPv3](#)
  - [Modifying the SNMP Engine ID](#)
  - [Configuring Thresholds on SNMP Processes](#)
  - [Supported SNMP MIBs](#)
  - [Routing MIBs](#)
  - [Enabling SNMP Traps](#)
  - [Using Additional SNMP commands](#)
  - [Supported SNMP Notifications \(Traps\)](#)
  - [Additional MIBs Supported in Netvisor ONE](#)
  - [Sample CLI outputs](#)
  - [Related Documentation](#)
-

## Overview - Understanding and Configuring SNMP

---

Simple Network Management Protocol (SNMP) is an application-layer protocol used to manage and monitor network devices and functions. SNMP provides a common language for network devices to relay management information between network elements and monitor the health of network devices such as routers, switches, access points, and even devices such as UPS, printers etc. SNMP is part of Transmission Control Protocol/Internet Protocol (TCP/IP) protocol suite and comes bundled with SNMP agent. The SNMP agents are configured and enabled to communicate with the network management system (NMS).

## Configuring SNMP

Netvisor ONE supports SNMP v1, v2, and v3 and the implementation is based on net-SNMP, where the SNMP daemon runs as a service. When an external browser or SNMP collector contacts the SNMP daemon on Netvisor, the SNMP daemon queries the nvOSd for details. However, the SNMP daemons (snmpd or nvOS\_snmpd) can hog the CPU and memory of the system causing *out of memory* and kill other processes such as nvOSd. Configuring the CPU and memory-limit thresholds mitigates this issue. For more details, see Configuring Thresholds on SNMP Processes.

Netvisor ONE uses SNMP for monitoring the networks only, that is, you can use SNMP service for read-only MIB support and does not provide support for managing the network. The following sections explain how to configure access controls and enable SNMP on a switch.

### Activating SNMP Service

By default, the SNMP service is disabled on Netvisor ONE. You can enable the SNMP service by using the admin-service-modify command. Netvisor ONE retains the status of the SNMP service (enabled or disabled) when you upgrade from one release version to another version.

To enable SNMP service, use the command:

```
CLI (network-admin@switch) admin-service-modify if if-string snmp|no snmp
```

<i>if if-string</i>	Specify the administrative service interface. The options are mgmt or data.
<i>snmp no snmp</i>	Specify to enable or disable SNMP.

To enable SNMP service through management interface, use the command:

```
CLI (network-admin@switch) admin-service-modify if mgmt snmp
```

To enable SNMP service on data interface, use the command:

```
CLI (network-admin@switch) admin-service-modify if data snmp
```

The above commands launch the daemon, sub-agents, and opens the firewall so that remote queries can reach the daemon. Use the no-snmp parameter for disabling SNMP service on the administrative interface:

```
CLI (network-admin@switch) admin-service-modify if mgmt no-snmp
```

For using SNMP services, you must configure access controls either by creating SNMP communities and/or by creating SNMP users.

## Creating SNMP Communities on SNMP V1 and V2

SNMPv1 and v2 protocol uses communities as a method of controlling access to information. A community consists of the community string and community type. You can create a community using the following command:

```
CLI (network-admin@switch) > snmp-community-create community-string
community-string-string community-type read-only|read-write
```

community-string <i>community-string-string</i>	Specify a community name
community-type read-only read-write	Specify the community type having read-only or read-write privileges

For example, to create a SNMP community string named, **pluribus**, with read-only privileges, use the following command:

```
CLI (network-admin@switch) > snmp-community-create community-string
pluribus community-type read-only
```

In Netvisor ONE, the snmp-show command enables SNMP walk internally on specific MIB tables or MIB elements. You can run SNMP walk from any host where SNMP software is enabled. To display the details of a Pluribus custom MIB, pnFabricTable, for example, use the command:

```
CLI (network-admin@switch) > snmp-show community-string pluribus name
pnFabricTable show-type walk
```

switch	name	value
-----	-----	-----
switch-name	FbIndex.100663455	Gauge32: 100663455
switch-name	NodeName.100663455	STRING: switch.
switch-name	FabricName.100663455	STRING: switch.
switch-name	NodeState.100663455	Gauge32: 1

To modify the SNMP community, **pluribus**, to read-write, use the following command:

```
CLI (network-admin@switch) > snmp-community-modify community-string
pluribus community-type read-write
```

To display information about the SNMP community, **pluribus**, use the following command:

```
CLI (network-admin@switch) > snmp-community-show community-string pluribus
```

switch	community-string	community-type
-----	-----	-----
switch	pluribus	read-write

To delete the SNMP community, **pluribus**, use the following command:



```
CLI (network-admin@switch) > snmp-community-delete community-string  
pluribus
```

## Creating SNMP Users on SNMPv3

---

SNMPv3 protocol supports the creation of users and optionally allows the usage of authentication and encryption. Netvisor ONE supports SHA or MD5 as authentication protocols and DES as the encryption algorithm. The default authentication protocol is SHA, however, Netvisor allows you to change the authentication protocol to MD5 by using the CLI.

You can also create a user without providing the authentication and privilege password options. For example,

```
CLI (network-admin@switch) > snmp-user-create user-name name-string auth  
priv
```

To create a user by providing the authentication and privilege passwords for encryption, use the following command. You must provide a password for authentication (*auth-password*) and encryption (*priv-password*):

```
CLI (network-admin@switch) > snmp-user-create user-name user-name-string  
auth-password auth-password-string [auth|no-auth] [auth-hash md5|sha] priv-  
password priv-password-string [priv|no-priv]
```

To create the user, *pluribus*, with authentication password **mOnk3ys\$**, and authentication hash as SHA1, use the following command:

```
CLI (network-admin@switch) > snmp-user-create user-name pluribus auth auth-  
hash sha
```

```
auth password: *****  
confirm password: *****
```

The password should have at least eight (8) characters and can be a combination of letters, numbers, and special characters. To modify the SNMP user and add privilege with the password, **b33h!v3#**, use the following command:

```
CLI (network-admin@switch) > snmp-user-modify user-name pluribus auth-  
password auth priv-password priv  
priv-password priv  
auth password: *****  
confirm password: *****  
priv password: *****  
confirm password: *****
```

To display information about the SNMP user created earlier, use the following command:

```
CLI (network-admin@switch) > snmp-user-show user-name pluribus
```

user-name	auth	auth-hash	priv
-----	----	-----	----
pluribus	yes	sha	yes

Create another user with user name, pluribus2 and authentication hash as MD5:

```
CLI (network-admin@switch) > snmp-user-create user-name pluribus2 auth
auth-password priv priv-password auth-hash md5
```

```
auth password:*****
confirm auth password:*****
priv password:*****
confirm priv password:*****
```

To display the details, use the following command:

```
CLI (network-admin@switch) > snmp-user-show
```

switch	user-name	auth	auth-hash	priv
switch	pluribus1	yes	sha	yes
switch	pluribus2	yes	md5	yes

To delete the SNMP user, use the `snmp-user-delete` command:

```
CLI (network-admin@switch) > snmp-user-delete user-name
```

After you create the SNMP user, you must grant permission to view the SNMP objects by using the View Access Control Model (VACM). To grant permission, use the command:

```
CLI (network-admin@switch) > snmp-vacm-create user-name snmp-user user-name
user-type [rouser|rwuser] oid-restrict oid-restrict-string [auth|no-auth]
[priv|no-priv]
```

The parameter, `oid-restrict`, is an optional argument and specifies a MIB sub-tree with a restricted view. In other words, if you specify an OID, you can only see that OID and the descendants in the tree.

For example, using the `snmp-vacm-create` command can restrict a particular user, `snmp-user` in accessing a specified OID. For example, to restrict access to `sysContact` OID, use the command:

```
CLI (network-admin@switch) > snmp-vacm-create user-name snmp-user user-type
rouser oid-restrict sysContact no-auth no-priv
```

To modify the VACM configuration of the user and to change no authentication to authentication, use the following command:

```
CLI (network-admin@switch) > snmp-vacm-modify user-name snmp-user user-type
rouser auth
```

To display information about the VACM configuration, use the `snmp-vacm-show` command:

```
CLI (network-admin@switch) > snmp-vacm-show
```

user-type	user-name	oid-restrict	view	auth	priv
rouser	snmp-user	sysContact		no	no

To delete the VACM of the user from the SNMP configuration, use the `snmp-vacm-delete` command:

```
CLI (network-admin@switch) > snmp-vacm-delete user-name snmp-user
```

## Modifying the SNMP Engine ID

Netvisor ONE allows you to modify the SNMP Engine ID and retrieve previous SNMP agent information for a switch that is no longer in use. The SNMP Engine ID is used only by SNMPv3 entities for uniquely identifying the V3 entries.

If you remove a switch due to an RMA or other reasons from the network, you can modify the SNMP Engine ID to use the old SNMP Engine ID. This enables Netvisor ONE to query and maintain the same history records for the new switch. The SNMP engine ID is a unique string of characters specific to a switch and identifies the device for administrative purposes.

After modifying the SNMP engine ID, you must recreate the SNMP user. To modify the SNMP engine ID, use the command:

```
CLI (network-admin@switch) > snmp-engineid-modify engineid <string>
```

---

engineid string

---

Specify the 28 character unique ID for the SNMP engine.

---

For example, to modify the SNMP engine ID, use the steps below:

- View the current SNMP engine ID by using the command:

```
CLI (network-admin@switch) > running-config-show | grep snmp
```

```
snmp-engineid-modify engineid 0x80001f8880077f7820da49395a00000000
```

- To modify the above SNMP engine ID to 0x80001f8880077f7820da49395a00000001, use the command:

```
CLI (network-admin@switch) > snmp-engineid-modify engineid
0x80001f8880077f7820da49395a00000001
```

```
Warning: All SNMP users will be erased.
```

```
Please confirm y/n (Default: n):y
```

```
Modified snmp engineID, Deleted all SNMP users.Please re-create SNMP
users.
```

- View the modified SNMP engine ID using the command:

```
CLI (network-admin@switch) > running-config-show | grep snmp
```

```
snmp-engineid-modify engineid 0x80001f8880077f7820da49395a00000001
```

To display the SNMP engine IDs configured on the switches in a fabric, use the command:

```
CLI (network-admin@aries-vxlan-01) > snmp-engineid-show
```

```
switch:    aries-vxlan-01
```

```
engineid: 0x80001f8880f3ac9a7e4b3fb15c00000000  
switch:   ursa-vxlan-01  
engineid: 0x80001f888030aae147f240b15c00000000  
switch:   ursa-vxlan-02  
engineid: 0x80001f8880cae2b0662f40b15c00000000
```

To display the SNMP engine ID of a local switch, use the command:

```
CLI (network-admin@switch-03*) > snmp-engineid-show  
  
engineid: 0x80001f888054a353110aefe65c00000000
```

The asterisk (\*) along with switch name in the above command indicates a specific local switch.

## Configuring Thresholds on SNMP Processes

When an external browser or SNMP collector contacts the SNMP daemon on Netvisor, the SNMP daemon queries the nvOSd for details. However, the SNMP daemons (snmpd or nvOS\_snmpd) can hog the CPU and memory of the system causing out of memory and kill processes such as nvOSd. Configuring the CPU and memory-limit thresholds mitigates this issue.

Netvisor ONE enables you to set thresholds for CPU and memory of snmpd and nvOS\_snmpd processes on Linux platforms. To configure the threshold values for CPU and memory, use the `snmp-limit-modify` command:

```
CLI (network-admin@switch01*) > snmp-limit-modify
```

Specify one or more of the following options:	
<code>cpu-limit</code> <i>cpu-limit-string</i>	Specify the maximum cpu usage of SNMP services in percentage.
<code>mem-limit</code> <i>mem-limit-number</i>	Specify the Maximum memory usage of SNMP services in bytes (+K,M,G or T).

To view the thresholds for CPU and memory, use the `snmp-limit-show` command:

```
CLI (network-admin@switch01*) > snmp-limit-show
```

```
cpu-limit: 0%
mem-limit: 0
```

The value zero (0) indicates that no threshold is configured and supports unlimited usage of CPU and memory.

To set the thresholds, for example, of 50% for CPU and 1 G for memory using the command,

```
CLI (network-admin@switch01*) > snmp-limit-modify mem-limit 1G cpu-limit 50
```

```
CLI (network-admin@draco01*) > snmp-limit-show
cpu-limit: 50%
mem-limit: 1G
```

## Supported SNMP MIBs

---

Netvisor One supports several MIBs and the MIBs run as a sub-agent to SNMP daemon (master) and uses AgentX protocol for communicating between the SNMP daemon and sub-agent.

When you issue an `snmp-show` command, the SNMP daemon receives the SNMP walk request and transfers the call to a sub-agent based on the SNMP OID of the request. The sub-agent populates the SNMP container to Netvisor ONE and displays the data. Netvisor ONE supports the following MIBs:

- IfTable (IF-MIB)
- IfXTable (IF-MIB)
- EntPhySensorTable (ENTITY-SENSOR-MIB)
- BGP MIB. The following BGPv4 tables are supported:
  - bgpPeerTable
  - bgp4PathAttrTable
- OSPF MIB. The following OSPF tables are supported:
  - ospfGeneralGroupTable
  - ospfAreaTable
  - ospfLsdbTable
  - ospfIfTable
  - ospfExtLsdbTable
  - ospfNbrTable
  - ospfIfMetricTable
- OSPFv3 MIB Tables. The following OSPFv3 tables are supported:
  - ospfv3GeneralGroupTable
  - ospfv3AreaTable
  - ospfv3NbrTable
  - ospfv3IfTable
  - ospfv3AreaLsdbTable
  - ospfv3AsLsdbTable
  - ospfv3LinkLsdbTable
- IEEE8021-Q-BRIDGE-MIB (ieee8021QBridgePortVlanStatisticsTable)
- IEEE8021-Q-BRIDGE-MIB (ieee8021QBridgeVlanStaticTable)
- IEEE8021-SPANNING-TREE-MIB (ieee8021SpanningTreePortTable)
- IEEE8021-SPANNING-TREE-MIB (ieee8021SpanningTreeTable)
- IP-MIB (ipv4InterfaceTable, ipv6InterfaceTable)
- VRRP-MIB (vrrpOperTable)
- PN-VRRP-MIB (supports IPv6 VRRP state change)
- Pluribus Enterprise PN-HR-MIB (pnHrCPUTable, pnHRMemTable)
- Pluribus Enterprise PN-FABRIC-MIB (pnFabricTable)
- PN-COS-MIB (pnCosStatsTable, pnCosBezelStatsTable)

The above MIBs consolidate into a single sub-agent, **nvOS\_snmpd** agent, which communicates with the master SNMP daemon and serves requests to the MIBs. All the MIBs except the routing MIBs are part of this **nvOS\_snmpd** agent.



## Routing MIBs

For the routing MIBs, OSPF and BGP, Netvisor ONE hosts the routing protocols inside vRouter. Use the commands `vrouter-create` and `vrouter-modify` to enable the routing MIBs. You can also enable the SNMP notifications for OSPF and BGP protocols while enabling the routing MIBs.

The enable Routing MIBs feature is supported only on single vRouter on a *fabric*. You cannot enable Routing MIBs for all the switches in same fabric.

To enable or disable the routing MIBs, BGP and OSPF on the vRouter, use the command:

```
CLI (network-admin@switch) > vrouter-create name name-string
```

Specify one or more options:	
<code>bgp-snmp   no-bgp-snm</code>	Enable or disable BGP SNMP MIB.
<code>bgp-snmp-notification   no-bgp-snmp-notification</code>	Enable or disable BGP SNMP notifications.
<code>ospf-snmp   no-ospf-snmp</code>	Enable or disable OSPF SNMP MIB.
<code>ospf-snmp-notification   no-ospf-snmp-notification</code>	Enable or disable OSPF SNMP notifications.
<code>ospf6-snmp   no-ospf6-snmp</code>	Enable or disable OSPFv3 SNMP MIB.
<code>ospf6-snmp-notification   no-ospf6-snmp-notification</code>	Enable or disable OSPFv3 SNMP notifications
<code>ip-snmp   no-ip-snmp</code>	Enable or disable ipForward SNMP MIB.

For example, to create a vrouter, **vr1** and to enable the routing MIBs and notifications on **vr1**, use the command:

```
CLI (network-admin@switch) > vrouter-create vr1 bgp-snmp bgp-snmp-notification ospf-snmp ospf-snmp-notification
```

Use the `vrouter-modify` command to modify or disable the SNMP notifications:

```
CLI (network-admin@switch) > vrouter-modify vr1 bgp-snmp no-bgp-snmp-notification ospf-snmp no-ospf-snmp-notification
```

To run SNMP Walk on the supported switches, you must ensure:

- The SNMP service is enabled on the switch by using the `admin-service-show` command.
- The community-string is configured correctly on the switch using the command, `snmp-community-create community-string <community-string-name>`. For more details, see the *Creating SNMP Communities* section.
- To get traps for particular SNMP trap receiver or NMS software, configure the SNMP trap sink host using the command, `snmp-trap-sink-create community <community-string-name> type TRAP_TYPE_V2C_TRAP dest-host <destination-host>`. For more details, see the *Enabling SNMP Traps* section.



## Enabling SNMP Traps

To enable SNMP traps, you must:

1. Setup the destination host. Use separate commands for SNMPv1 or SNMPv2c and SNMPv3 to setup the destination host.
2. Enable the SNMP traps to be sent.

To setup the destination host for SNMPv1 or SNMPv2c, use the command:

```
CLI (network-admin@switch) > snmp-trap-sink-create community community-string [type TRAP_TYPE_V1_TRAP|TRAP_TYPE_V2C_TRAP|TRAP_TYPE_V2_INFORM]
dest-host dest-host-string [dest-port dest-port-number]
```

<code>community <i>community-string</i></code>	Specify the community type.
<code>[type TRAP_TYPE_V1_TRAP  TRAP_TYPE_V2C_TRAP  TRAP_TYPE_V2_INFORM]</code>	Specify the trap type. The default trap is: TRAP_TYPE_V2C_TRAP.
<code>dest-host <i>dest-host-string</i></code>	Specify the destination host.
<code>[dest-port <i>dest-port-number</i>]</code>	Specify the destination port. The default port number is 162.

For example, to create a community *test1*, with destination host as *lyra10.pluribusnetworks.com* and trap type as *TRAP\_TYPE\_V1\_TRAP*, use the command:

```
CLI (network-admin@switch) > snmp-trap-sink-create community test1 dest-host lyra10.pluribusnetworks.com type TRAP_TYPE_V1_TRAP
```

This command sends the SNMPv1 traps to a receiver on the host *lyra10* with the community string *test1*.

To setup the destination host for SNMPv3 trap receiver, use the command:

```
CLI (network-admin@switch) > snmp-v3-trap-sink-create user-name user-name-string [type TRAP_TYPE_V3_TRAP|TRAP_TYPE_V3_INFORM] dest-host dest-host-string [dest-port dest-port-number]
```

<code>user-name <i>user-name-string</i></code>	Specify the user name for the SNMPv3 trap.
<code>[type TRAP_TYPE_V3_TRAP  TRAP_TYPE_V3_INFORM]</code>	Specify the SNMPv3 trap type.
<code>dest-host <i>dest-host-string</i></code>	Specify the destination host.
<code>[dest-port <i>dest-port-number</i>]</code>	Specify the destination port number. The default port number is 162

For example, to configure SNMPv3 traps for a user: *test2*, destination host as *lyra10.pluribusnetworks.com* and with authentication enabled, use the command:

```
CLI (network-admin@switch) > snmp-v3-trap-sink-create user-name test2 dest-host lyra10.pluribusnetworks.com auth
```

This command configures the traps to be sent to *lyra10* with the username *test2*, engine id *0xABCDEF* and with authentication enabled. You will be prompted for an authentication password with this command.

After configuring the destination, you must enable the type of traps to be sent using the command:

```
CLI (network-admin@switch) > snmp-trap-enable-modify
```

Specify one or more of the following options:	
<code>link-up-down   no-link-up-down</code>	Specify the port link state as up or down.
<code>interface-up-down   no-interface-up-down</code>	Specify the vrouter interface status.
<code>default-monitors   no-default-monitors</code>	Specify the default monitoring.
<code>physical-sensors   no-physical-sensors</code>	Specify the temperature, fan speed.
<code>low-disk-space   no-low-disk-space</code>	Specify the allowed low disk space.
<code>low-disk-space-threshold low-disk-space-threshold-string</code>	Specify the Threshold value of low-disk-space in %.
<code>system-usage   no-system-usage</code>	Specify the Memory and CPU usage.
<code>high-system-usage-threshold high-system-usage-threshold-string</code>	Specify the Threshold value of system-usage in %.
<code>login-failure   no-login-failure</code>	Specify failure on Incorrect passwords during login.
<code>cluster-tr-diverge   no-cluster-tr-diverge</code>	Specify when the cluster transaction lists have diverged.
<code>lacp-status   no-lacp-status</code>	Specify the LACP status.
<code>vport-modified   no-vport-modified</code>	Specify when vPort modification happens.
<code>stp-port-modified   no-stp-port-modified</code>	Specify when STP port is modified.
<code>mirror-to-cpu   no-mirror-to-cpu</code>	Specify when Mirror to CPU configured.
<code>stp-port-state-failed   no-stp-port-state-failed</code>	Specify when STP Port State Failed.
<code>link-congestion-detected   no-link-congestion-detected</code>	Specify when Congestion detected at port.

<code>fabric-node-state-changed   no-fabric-node-state-changed</code>	Specify when Fabric Node State Changed.
<code>stp-new-root   no-stp-new-root</code>	Specify when New STP Root Detected.
<code>stp-topology-changed   no-stp-topology-changed</code>	Specify when STP Topology Changed
<code>vrrp-new-master   no-vrrp-new-master</code>	Specify when VRRP Master Change.
<code>disable-start-stop   no-disable-start-stop</code>	Specify when Disable Start/Stop Traps.
<code>cert-expiry   no-cert-expiry</code>	Specify whether to monitor the expiry of the Netvisor ONE certificates ( shipped with the switches) or not.

The above command enables the notifications to be sent out when a link changes the state to up or down.

Further, You can enable to collect system log messages and SNMP trap notifications for the following events when:

- Physical Interface is Up/Down
- Data/vRouter Interface is Up/Down
- CPU usage is higher than 70%
- Memory Utilization is higher than 70%
- Disk Utilization is when higher than 70%
- STP port is modified
- vPort is modified
- LAG/vLAG status changes
- Mirror to CPU vFlow is configured

## Using Additional SNMP commands

You can use the following additional SNMP commands to check the details of SNMP configurations and notifications on the Pluribus switches:

- `snmp-trap-enable-modify` — Use to enable notifications about link conditions and common system errors.

CLI (network-admin@switch) > `snmp-trap-enable-modify`

Specify one or more of the following options:	
<code>link-up-down   no-link-up-down</code>	Specify if you want to enable a link for up or down trap
<code>interface-up-down   no-interface-up-down</code>	Specifies the vrouter interface status as up or down.
<code>default-monitors   no-default-monitors</code>	Specify if you want to enable a default monitoring trap.
<code>physical-sensors   no-physical-sensors</code>	Specify if you want to enable a physical sensor trap for temperature, fan speed, etc
<code>low-disk-space   no-low-disk-space</code>	Specify if you want to enable a low disk space trap.
<code>low-disk-space-threshold low-disk-space-threshold-string</code>	Specify the threshold value of low-disk space in percentage (%).
<code>system-usage   no-system-usage</code>	Specify if you want to enable a system usage trap.
<code>high-system-usage-threshold high-system-usage-threshold-string</code>	Specify if you want to enable high system usage threshold value trap in percentage (%).
<code>login-failure   no-login-failure</code>	Specify if you want to enable login failure trap.
<code>cluster-tr-diverge   no-cluster-tr-diverge</code>	Specify if the cluster transaction list have diverged or not.
<code>lACP-status   no-lACP-status</code>	Specify if you want to enable a LACP trap.
<code>vport-modified   no-vport-modified</code>	Specify if you want a enable a trap when a vPort is modified.
<code>stp-port-modified   no-stp-port-modified</code>	Specify if STP port is modified or not.
<code>mirror-to-cpu   no-mirror-to-cpu</code>	Specify whether the trap should be mirrored to CPU.
<code>stp-port- state-failed   no-stp-port-state-failed</code>	Specify if you want to enable a trap when the STP port state is failed.

<code>link-congestion-detected   no-link-congestion-detected</code>	Specify if you want to enable a trap when a link congestion state is detected.
<code>fabric-node-state-changed   no-fabric-node-state-changed</code>	Specify if you want a trap sent when a fabric node state changes.
<code>stp-new-root   no-stp-new-root</code>	Specifies if new STP root is detected.
<code>stp-topology-changed   no-stp-topology-changed</code>	Specifies if STP topology has changed or not.
<code>vrrp-new-master   no-vrrp-new-master</code>	Specifies if VRRP master has changed or not.
<code>disable-start-stop   no-disable-start-stop</code>	Specifies if the start/stop traps are disabled or not.

- `snmp-trap-enable-show` — Displays enabled SNMP traps.

CLI (network-admin@switch) > `snmp-trap-enable-show`

The above command displays the information about the SNMP traps on the switch. Specify one or more of the formatting options to display the output in a desired format.

- `snmp-trap-sink-create` — Use to specify a SNMPv1 or v2 trap receiver.

CLI (network-admin@switch) > `snmp-trap-sink-create community community-string [type TRAP_TYPE_V1_TRAP | TRAP_TYPE_V2C_TRAP | TRAP_TYPE_V2_INFORM] dest-host dest-host-string [dest-port dest-port-number]`

<code>community <i>community-string</i></code>	Specify the community type
<code>[type TRAP_TYPE_V1_TRAP   TRAP_TYPE_V2C_TRAP   TRAP_TYPE_V2_INFORM]</code>	Specify the trap type. The default trap is: TRAP_TYPE_V2C_TRAP
<code>dest-host <i>dest-host-string</i></code>	Specify the destination host.
<code>[dest-port <i>dest-port-number</i>]</code>	Specify the destination port. The default port number is 162.

- `snmp-trap-sink-delete` — Delete SNMP trap sinks

CLI (network-admin@switch) > `snmp-trap-sink-delete community community-string dest-host dest-host-string [dest-port dest-port-number]`

The above command deletes the SNMP trap sink for the specified community. For details about the command parameters, see the CLI table for `snmp-trap-sink-create` command.

- `snmp-trap-sink-show` — Display SNMP trap sinks.

CLI (network-admin@switch) > `snmp-trap-sink-show community community-string`

```
[type TRAP_TYPE_V1_TRAP|TRAP_TYPE_V2C_TRAP|TRAP_TYPE_V2_INFORM] dest-host
dest-host-string [dest-port dest-port-number]
```

The above command displays the SNMP trap sinks. Specify one or more of the formatting options to display the output in a desired format.

- `snmp-v3-trap-sink-create` — Use to specify a SNMPv3 trap receiver.

```
CLI (network-admin@switch) > snmp-v3-trap-sink-create user-name user-name-
string [type TRAP_TYPE_V3_TRAP|TRAP_TYPE_V3_INFORM] dest-host dest-host-
string [dest-port dest-port-number]
```

<code>user-name user-name-string</code>	Specify the user name for the SNMPv3 trap.
<code>[type TRAP_TYPE_V3_TRAP  TRAP_TYPE_V3_INFORM]</code>	Specify the SNMPv3 trap type.
<code>dest-host dest-host-string</code>	Specify the destination host
<code>[dest-port dest-port-number]</code>	Specify the destination port number. The default port number is 162.

The above command enables you to create an SNMPv3 trap receiver.

- `snmp-v3-trap-sink-delete` — Use to delete a SNMPv3 trap receiver.

```
CLI (network-admin@switch) > snmp-v3-trap-sink-delete user-name user-name-
string dest-host dest-host-string [dest-port dest-port-number]
```

The above command deletes the SNMPv3 trap sink for the specified user. For details about the command parameters, see the CLI table for `snmp-v3-trap-sink-create` command.

- `snmp-v3-trap-sink-show` — Used to display a SNMPv3 trap receiver.

```
CLI (network-admin@switch) > snmp-v3-trap-sink-show user-name user-name-
string dest-host dest-host-string [dest-port dest-port-number]
```

The above command displays the SNMPv3 trap receivers for the specified options (see table below). Specify one or more of the formatting options to display the output in a desired format.

Specify one of or more options:	
<code>user-name user-name-string</code>	Displays traps for the specified user name.
<code>engine-id engine-id-string</code>	Displays the traps for the specified engine ID.
<code>[type TRAP_TYPE_V3_TRAP  TRAP_TYPE_V3_INFORM]</code>	Displays the traps for th specified trap type.
<code>dest-host dest-host-string</code>	Displays the traps for the destination host.



<code>dest-port dest-port-number</code>	Displays the traps for the specified destination port.
<code>auth no-auth</code>	Displays the authentication details.
<code>priv no-priv</code>	Displays the privilege details.

- `port-stats-snmp-show` – To display the accumulated port statistics for SNMP.

```
CLI (network-admin@switch) > port-stats-snmp-show port port-list port-
clear-count port-clear-count-number port-clear-time #d#h#m#s
port_clear_time high resolution time: #ns all-ports
```

<code>port port-list</code>	Specifies the port number or port list.
<code>port-clear-count port-clear-count-number</code>	Specifies the number of times <code>port-stats-clear</code> or <code>trunk-stats-clear</code> commands were issued on a port or trunk.
<code>port-clear-time #d#h#m#s</code>	Specifies the time since last <code>port-clear</code> or <code>trunk-clear</code> commands were issued. It is initialized with time at <code>nvOS_init</code> by default.
<code>port_clear_time high resolution time: #ns</code>	Specifies the high-resolution time-stamp of the last <code>port-stats-clear</code> command.
<code>all-ports</code>	A flag to specify if you want to view the stats of all ports on the switch. The default view contains only enabled ports.

- `snmp-system-modify` – Modify the system information for the SNMP configuration.

```
CLI (network-admin@switch) > snmp-system-modify syscontact syscontact-
string syslocation syslocation-string
```

Specify one of more options:	
<code>syscontact syscontact-string</code>	Specify or change the SNMP system contact.
<code>syslocation syslocation-string</code>	Specify or change the SNMP system location.

The above command enables you to modify the system details such as the system contact or location for the SNMP configuration.

- `snmp-system-show` – Displays the system details for the SNMP configurations.

```
CLI (network-admin@switch) > snmp-system-show
```

The above command displays the changed or modified system information for the SNMP configuration. Specify one or more of the formatting options to display the output in a desired format.

## Supported SNMP Notifications (Traps)

Netvisor ONE supports two types of SNMP notifications (a.k.a traps): event-based and message-based traps.

Event-based traps are the traps generated by the SNMP daemon based on specific events when an OID value changes. For example, when there is a change in the *link-up-down* or a *low-disk-space*. Message-based traps are triggered based on messages logged in the local logging mechanism. For example, a *login-failure* Trap is triggered when a login failure message is saved in the */var/log/auth.log* log file. **Table 1** explains the supported SNMP notifications.

**Table: Details of Supported SNMP Notifications**

Trap Name	Description	Trap Type	Trigger
link-up-down	Port link is up or down	Event-based	If enabled, SNMP generates a trap when a port is up or down.
default-monitors	Use default SNMP monitoring		
physical-sensors	Physical sensors are enabled		If enabled, SNMP generates a trap for physical sensors such as power supplies and fans.
low-disk-space	Monitors for low-disk-space	Event-based	If enabled, SNMP generates a trap if disk space is lower than threshold. SNMP checks the output of the command, <code>storage-pool-show</code> .
low-disk-space-threshold	The threshold value of low-disk-space in percentage	Event-based	SNMP generates the trap with low-disk-space.
system-usage	Monitors memory & CPU usage	Event-based	If enabled, SNMP generates a trap if system-usage[Total CPU-sys + user] space is greater than threshold. SNMP checks the output of the command, <code>system-stats-show</code> . The parameter <code>cpu-total</code> displays the threshold value.
system-usage-threshold	The threshold value of system-usage in percentage	Event-based	SNMP generates the system usage trap.
login-failure	Monitors login failures	Message-based	If enabled, SNMP generates a trap when user login with wrong password.
lACP-status	Monitors LACP enable or disable	Message-based	If enabled, SNMP generates a trap when the LACP state changes from enable to disabled or vice versa.

vport-modified	Monitors vPort modifications	Message-based	If enabled, SNMP generates a trap when vPort modifications occur on the switch.
stp-port-modified	Monitors STP port status	Message-based	If enabled, SNMP generates a trap when STP port state is modified using the command, <code>switch-local stp-port-modify port 1 &lt;block   edge   bpdu   root-guard&gt;</code>
stp-port-state-failed	Monitors STP port state failures	Message-based	If enabled, SNMP generates a trap when STP port state is modified using the command, <code>switch-local stp-port-modify port 128 edge bpdu-guard.</code>
mirror-to-cpu	Monitors mirror-to-cpu configuration	Message-based	If enabled, SNMP generates a trap when created a vflow using the command, <code>vflow-create name mirror scope local action copy-to-cpu</code> and also generates a trap for <code>perror.log</code> .
link-congestion-detected	Monitors congestion drop at port	Message-based	If enabled, SNMP generates a trap indicating a link is congested.
fabric-node-state-changed	Monitors fabric node states	Message-based	If enabled, SNMP generates a trap when the a fabric node changes state.
ospfIfStateChange	Monitors OSPF interface states	Event-based	If enabled, this notification is triggered when interface state changes from DR to Down or vice versa. Originator for this trap is by changing router-id.
ospfNbrStateChange	Monitors OSPF NBR states	Event-based	If enabled, this notification is triggered when neighbor state changes from DR to Down or vice versa. Originator for this trap is designated router on broadcast networks.
bgpEstablished	Monitors BGP NBR state	Event-based	If enabled, this notification is triggered when the BGP FSM enters the ESTABLISHED State. Originator for this trap is to bring up BGP session between two BGP Peers.
bgpBackwardTransition	Monitors BGP NBR state transition	Event-based	If enabled, this notification is triggered when the BGP FSM moves from higher number

			to lower numbered state. Originator for this trap when BGP state changes from active to idle (higher state to lower state).
stp-new-root	Monitors new STP root	Event-based	If enabled, SNMP generates a trap to monitor a new root for STP.
stp-topology-changed	Monitors STP topology change	Event-based	If enabled, SNMP generates a trap to monitor topology changes for STP.
interface-up-down	Monitors vRouter interfaces	Event-based	If enabled, SNMP generates a trap for an interface with the state up or down.
disable-start-stop	Monitors disable traps for start and stop	Event-based	If enabled, a restart of SNMP service from CLI does not send any traps.
fabric-node-state-changed	Monitors fabric node states	Event-based	If enabled, SNMP generates a trap to monitor fabric node state changes.
system-usage	Monitors memory & CPU usage	Event-based	If enabled, SNMP generates a trap to monitor memory and CPU changes.
vrrp-new-master	Monitors VRRP master changes	Event-based	If enabled, SNMP generates a trap to monitor VRRP master state changes.
ospfv3IfStateChange	Monitors OSPF Interface state changes	Event-based	If enabled, this notification is triggered when interface state changes from DR to Down or vice versa.
ospfv3NbrStateChange	Monitors OSPF neighbor state changes	Event-based	If enabled, this notification is triggered when neighbor state changes from DR to Down or vice versa. Originator for this trap is designated router on broadcast networks.
cluster-tr-diverge	Monitors Cluster Transaction list for divergence	Message-based	If enabled, this notification is triggered when Transaction Diverge message is generated in perror.log.
cert-expiry	Monitors expiry of Switch Certificate	Message-based	If enabled, this notification gets triggered when switch-certificate expiring in xx number of days in /nvOS/log/event.log. You can control value of the number of days by using the <code>cert-expiration-alert-modify &lt;days-before-expiration&gt;</code> command.



## Additional MIBs Supported in Netvisor ONE

MIB	Description
Entity-Sensor	<p>This module defines Entity MIB extensions for physical sensors.</p> <p>The Entity Sensor MIB contains a single group called, <b>entitySensorValueGroup</b>, which allows objects to convey the current value and status of a physical sensor. Click <a href="#">here</a> for the RFC details.</p> <hr/> <p><b>entPhySensorOperStatus</b> — Object identifies the current operational status of the sensor (as it is known to the agent). For example,</p> <p>entPhySensorUnitsDisplay.1    STRING: Temp Outlet. &gt;60  entPhySensorUnitsDisplay.2    STRING: Temp Inlet.  ...  entPhySensorUnitsDisplay.22    STRING: PS1 Status.</p>
Host-Resources (For CPU Utilization)	<p>Use this MIB in managing host systems. Click <a href="#">here</a> for the RFC details.</p> <ul style="list-style-type: none"> <li>To monitor the CPU utilization of the switch, use the custom MIB: <b>PN-HR-MIB: PnHrCpuTable</b></li> <li>To monitor the fabric node status change, use the custom MIB: <b>PN-FABRIC-MIB: PnFabricTable</b></li> </ul>
IF-MIB, Entity-Sensor-MIB	<p>The MIB module to describe generic objects for network interface sub-layers.</p> <p>This MIB is an updated version of the ifTable for MIB-II, and incorporates the extensions defined in <a href="#">RFC 1229</a>.</p>
PN-VRRP-MIB	<p>This MIB addresses the information collected with the VRRP master change trap and supports IPv6 VRRP state change.</p>

## Sample CLI outputs

To display the SNMP details for the Pluribus custom MIB, pnFabricTable, having the community-string: pluribus, use the command:

```
CLI (network-admin@aquila-m) > snmp-show community-string pluribus name
pnFabricTable show-type walk
```

switch	name	value
-----	-----	-----
aquila-m	FbIndex.100663455	Gauge32: 100663455
aquila-m	FbIndex.167772211	Gauge32: 167772211
aquila-m	NodeName.100663455	STRING: aquila.
aquila-m	NodeName.167772211	STRING: aquila-m.
aquila-m	FabricName.100663455	STRING: aquila.
aquila-m	FabricName.167772211	STRING: aquila.
aquila-m	NodeState.100663455	Gauge32: 1
aquila-m	NodeState.167772211	Gauge32: 1

To display the SNMP details for the Pluribus custom MIB, pnHRCpuTable, having the community-string: pluribus, use the command:

```
CLI (network-admin@aquila-m) > snmp-show community-string pluribus name
pnHrCpuTable show-type walk
```

switch	name	value
-----	-----	-----
aquila-m	pnHrCpuIdx.0	INTEGER: 0
aquila-m	pnHrCpuUshr.0	INTEGER: 0
aquila-m	pnHrCpuSys.0	INTEGER: 0
aquila-m	pnHrCpuIdle.0	INTEGER: 99
aquila-m	pnHrCpuTotal.0	INTEGER: 0

To display the system details including the cpu-total output, use the command:

```
CLI (network-admin@aquila-m) > system-stats-show
```

switch	uptime	used-mem	used-swap	paging	cpu-user	cpu-sys	cpu-idle
-----	-----	-----	-----	-----	-----	-----	-----
aquila-m	8m15s	26%	10%	0	0%	0%	100%

For all details, include the parameter format all:

```
CLI (network-admin@aquila-m) > system-stats-show format all
```

uptime	used-mem	used-mem-val	used-swap	used-swap-val	paging	cpu-user	cpu-sys	cpu-total	cpu-idle
-----	-----	-----	-----	-----	-----	-----	-----	-----	-----
8m29s	26%	17.2G	10%	7.79G	0	0%	0%	0%	99%

To display the accumulated port statistics for SNMP, use the port-stats-snmp-show command:

```
CLI (network-admin@switch) > port-stats-snmp-show
```

port	ibits	iUpkts	iBpkts	iMpks	iCongDrops	ierrs	obits	oUpkts	oBpkts	oMpks	oCongDrops	oerrs	mtu-errs
0	7.71K	12	0	0	0	0	7.53M	9.53K	0	0	0	0	
9	7.33K	0	6	6	0	0	3.81M	0	85	4.74K	0	0	
10	0	0	0	0	0	0	3.73M	0	90	4.63K	0	0	

To display the accumulated port statistics for SNMP in a vertical layout, use the command:

```
CLI (network-admin@dorado-ext-03) > port-stats-snmp-show format all layout vertical
```

```
switch:          dorado-ext-03
community-string: pluribus
community-type:  read-only
switch:          dorado-ext-03
community-string: pluribus1
community-type:  read-only
```

To display the modified system information, for example, if you modified the system contact to bob@xyz.com with location as brisbane, use the snmp-system-show command:

```
CLI (network-admin@switch-04) > snmp-system-show
```

```
switch:          switch-04
syscontact:      bob@xyz.com
syslocation:     brisbane
```

You can also view the details using SNMP walk.



## Related Documentation

---

For more information on concepts mentioned in the SNMP chapter, refer to below documents:

- [Net-SNMP](#)
- [RFC2790](#)
- [RFC1229](#)
- [RFC3433](#)

## Configuring and Using vFlows

---

This chapter provides information for understanding, configuring, and using the vFlow feature in Netvisor ONE.

---

- [Understanding vFlows and vFlow Objects](#)
  - [Configuring the Administrative Scope and State](#)
  - [Implementing the vFlow Policies](#)
  - [Filtering of Traffic Flows](#)
    - [Configuring vFlows with User Defined Fields \(UDFs\)](#)
  - [Forwarding Action in vFlow Filters](#)
  - [Configuring vFlow Filters](#)
  - [Refreshing vFlow Level Statistics for Long-lived Connections](#)
  - [Commands and Parameters in vFlow](#)
  - [Guidelines and Limitations](#)
  - [Use Cases in vFlow](#)
    - [Supporting TCP Parameters using vFlows](#)
    - [Configuring Burst Size in vFlow for Maximum Bandwidth](#)
    - [Configuring Bandwidth Sharing for a Single VLAN](#)
    - [Enhancing the vFlow Capability to Match Forwarding Type and Packet Resolution in ASIC](#)
  - [Using Application Flows and Statistics](#)
    - [Displaying Statistics for vFlows for all Switches in the Fabric](#)
    - [Understanding vFlow Statistics](#)
  - [Examples and Use Cases for Network Monitoring and Security](#)
    - [Using vFlows to Disable Communication for Security Monitoring](#)
    - [Configuring vFlows to Filter Packets on Management Interfaces](#)
    - [Configuring vFlow for Analytics](#)
-

## Understanding vFlows and vFlow Objects

---

The vFlow functionality in Netvisor ONE is a unique Pluribus feature, which defines fabric-wide policies (using match conditions) to facilitate the manipulation and redirection of traffic flows using physical or logical filtering methods (using action parameters) at line rate. Netvisor ONE implements vFlow objects in hardware that have no impact on the forwarding performance of the switch.

The vFlows can be applied to traffic flows regardless of the forwarding method or provisioning construct employed. As such, vFlow objects can be implemented for bridging, routing and extended bridging operations and also for transparent forwarding services such as Virtual Wire and Virtual Link extension (vLE).

The vFlows can also be viewed as Access Control Lists (ACL) with advanced capabilities.

The vFlow functionality offers a versatile, programmable, and distributed method for implementing security access control policies, security service insertion, flow monitoring and telemetry, quality of service, and optimized flow-based forwarding.

In Netvisor ONE the vFlow filters operate at wirespeed without any performance degradation because the vflow actions and filtering are applied in the ASIC pipeline at line rate, which ensures no latency or performance degradation.

The vFlow object enables you to:

- Configure traffic filtering based on L2, L3, and L4 layer parameters
- Configure traffic filtering based on action parameters such as blocking and forwarding traffic
- Configure vFlows to copy packets to CPU, packet mirroring, packet classification, traffic metering and bandwidth guarantee
- Gathering statistics for evaluation and analytics

At a high level, vFlow feature supports the following actions, which can be configured using the CLI commands:

- Creating a vFlow Object

```
CLI (network-admin@switch-1) > vflow-create name <vflow-name> scope [local|fabric] {specify one or more parameters} {specify any action}
```

- Modifying an existing vFlow Object

```
CLI (network-admin@switch-1) > vflow-modify name <vflow-name> {specify one or more parameters}
```

- Deleting an existing vFlow Object

```
CLI (network-admin@switch-1) > vflow-delete name <vflow-name>
```

- Displaying the applicable actions and use cases for a selected vFlow Object

```
CLI (network-admin@switch-1) > vflow-show {specify one or more parameters}
```

These actions are explained in detail in the subsequent sections for configuring vFlow objects with specific parameters.

## Elements of a vFlow

Netvisor ONE identifies a vFlow object by a unique name and is composed of the following elements:

- Administrative scope and state
- Implementation stage
- Traffic flow filter
- Forwarding action

## Configuring the Administrative Scope and State

The administrative *state* of a vFlow object determines if you enable or disable the corresponding flow policy in the switch hardware, as defined by mutually exclusive keywords `enable` and `no-enable`. By default, Netvisor ONE enables newly created vFlow objects.

The administrative scope defines the set of switches in the fabric where you create the vFlow object, which is controlled by the keyword `scope`, and can be either `fabric` or `local` scope. The administrative parameters for a `vflow-create` command are:

```
CLI (network-admin@switch-1) > vflow-create name <vflow-name> scope
[fabric|local] [enable|no-enable]
```

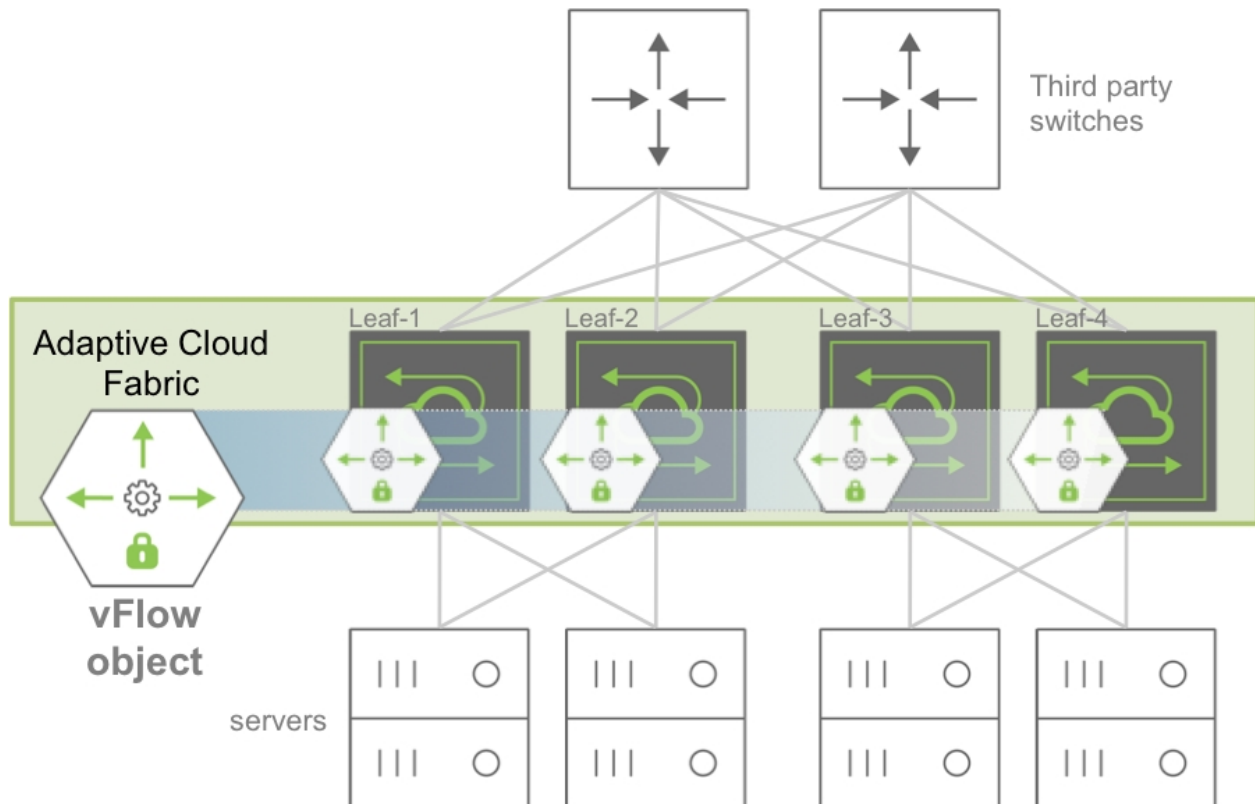
<code>name</code>	The vFlow object's unique identifier
<code>scope [fabric local]</code>	Defines the scope of the vFlow object. Once a Vflow object is created using either the local or the fabric scope, you cannot modify the scope of the vFlow object later. To modify, you must delete the vFlow object and create a new one.
<code>enable no-enable</code>	Enables or disables the flow policy in hardware. By default, Netvisor enables the vFlow objects. You can disable the vflow policy using the <code>no-enable</code> parameter.
<code>{parameters}</code>	<p>Specify one or more of the parameters</p> <ul style="list-style-type: none"> <li>• Table name</li> <li>• Filtering parameters</li> <li>• Action parameters</li> <li>• Flow class</li> </ul> <p>For details, see the <i>Filtering of Traffic Flows</i>, <i>Forwarding Action in vFlow Filters</i>, and <i>Commands and Parameters Applicable to vFlow Traffic</i> sections. Also, see the <i>Command Reference Guides</i></p>

**Note:** You can specify the hardware table name while creating a vFlow object, however, if not specified, Netvisor ONE uses the default table, **System-L1-L4-Tun-1-0**.

### Fabric Scope

A fabric-scoped vFlow is a single managed object distributed across all switches that are part of the Adaptive Cloud Fabric in Netvisor ONE. To create a fabric scoped vFlow object, for example, use the command:

```
CLI (network-admin@switch-1) > vflow-create name example_fabric_scope scope
fabric enable {parameters}
```



**Figure 10-1: Fabric Scoped vFlow Object Example**

**Figure 10-1** illustrates a fabric scoped vFlow object topology, where a single vFlow object is created on all four switches: Leaf-1, Leaf-2, Leaf-3, and Leaf-4 that are part of the Adaptive Cloud Fabric. The switches in the adaptive cloud fabric are also connected to multiple servers and other third party switches. In this scenario, the fabric-scoped vFlow can be modified concurrently on all switches of the fabric with a single CLI or API command, by referencing the unique name. For example, the below command disables the previously created vFlow object, **example\_fabric\_scope** for the entire fabric, where Netvisor ONE does not delete the object, but uninstalls the object from the hardware tables.

For example:

```
CLI (network-admin@switch-1) > vflow-modify name example_fabric_scope scope
fabric no-enable
```

## Local Scope

A local-scoped vFlow is an object defined and instantiated on one single switch. To create a locally scoped vFlow, for example, use the following command:

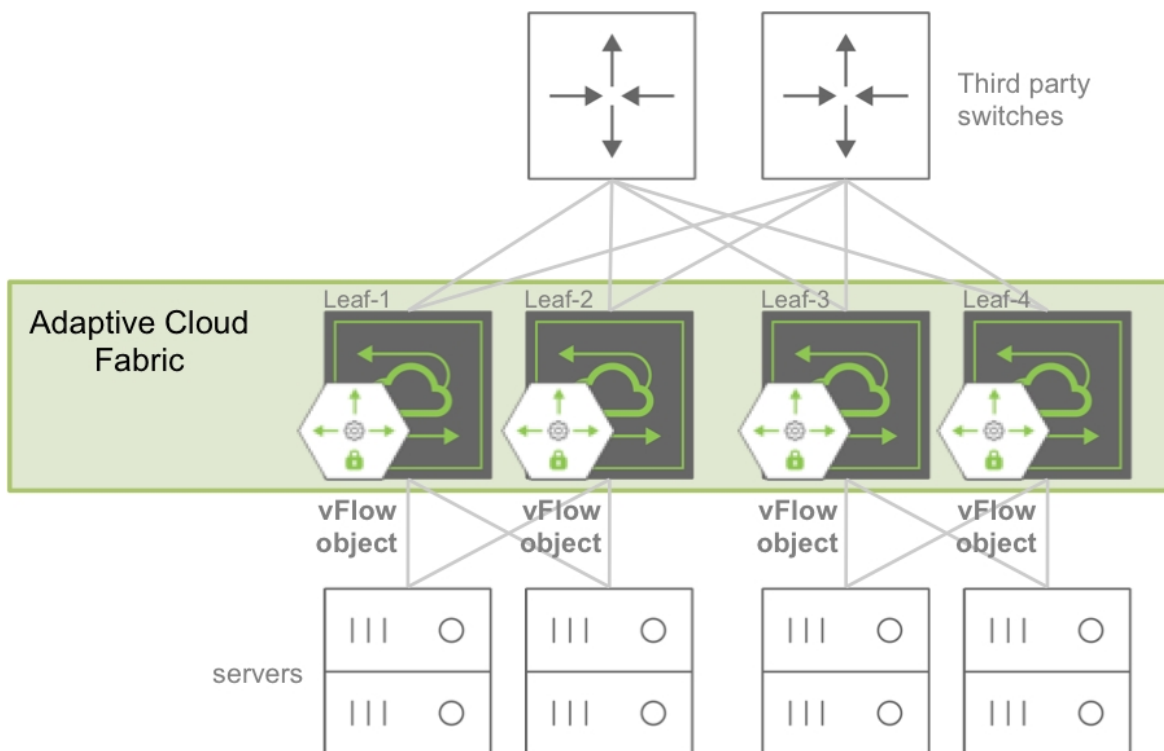
```
CLI (network-admin@switch-1) > vflow-create name example_local_scope scope
local enable {parameters}
```

Netvisor ONE allows you to apply or modify the same vFlow policy on multiple switches concurrently using a single CLI or API command by including the `switch` keyword followed by the list of individual switches or switch groups. Below is an example on creating a vFlow object on four switches, leaf-1, leaf-2, leaf-3, and leaf-4:

```
CLI (network-admin@leaf-1) > switch leaf-1,leaf-2 \ vflow-create name
example_local_scope scope local
```

```
CLI (network-admin@leaf-1) > switch leaf-3,leaf-4 \ vflow-create name
example_local_scope scope local
```

The above commands create the same vFlow object, `example_local_scope` on the four switches, leaf-1, leaf-2, leaf-3, and leaf-4 (see **Figure 10-2**).



**Figure 10-2: Local Scoped vFlow Object Example**

You can now modify or delete the vFlow objects on individual switches as explained in the example below:

To disable the vFlow object, `example_local_scope` on the switch, leaf-1, use the command:

```
CLI (network-admin@leaf-1) > switch leaf-1 vflow-modify name
example_local_scope no-enable
```

To delete the vFlow object, `example_local_scope` on the switch, leaf-2, use the command:

```
CLI (network-admin@leaf-1) > switch leaf-2 vflow-delete name
example_local_scope
```





## Implementing the vFlow Policies

Netvisor ONE allows you to apply multiple policies in parallel or in series to a particular traffic flow by providing the vFlow construct with two main attributes to control the sequential order of execution relative to other vFlows such as the hardware table and precedence.

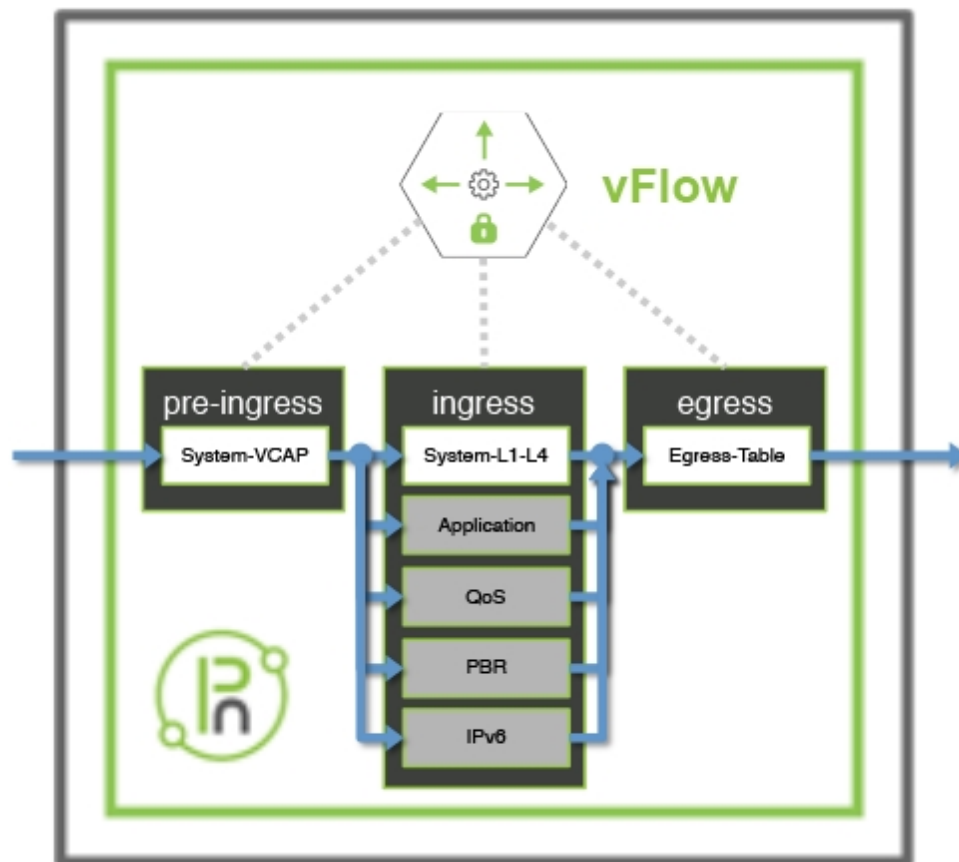
The following command keywords enable this functionality:

- `table-name` – hardware vflow table name
- `precedence` – processing priority value

### Hardware Table

Netvisor ONE provides multiple filter tables along the internal flow hardware data path. However, by default, the vFlow is installed in the ingress filter table, but allows you to optionally implement the vFlow in any other available table, although flow filtering, manipulation, and redirection capabilities may become limited.

**Figure 10-3** describes the available hardware tables with the corresponding vFlow table names and how the tables are concatenated, allowing both cascading and parallel execution policies.



**Figure 10-3: Concatenation of vFlow Hardware Tables**

Additionally, Figure 3 highlights the data-path forwarding stage for each filter table, where some tables are

always enabled (displayed in white), while some tables require manual enabling (displayed in grey) such as the Application, QoS, PBR, and IPv6 tables. Use the `table-name` keyword to install or program the vFlow in the specified hardware table.

**Table 10-1: Hardware Filter Tables with Descriptions**

Hardware Filter Tables	Description
System-VCAP	Where the system VCAP policies are defined at the pre-ingress stage
System-L1-L4	Where the system ingress traffic filtering policies are defined for L2, L3, and L4 packet parameters at the ingress or ICAP table. All system rules are defined in ICAP
Egress-Table	Where the system egress policies are defined at the egress or ECAP table. Supports drop and forward actions.
Application Table	Where the user application level policies are defined.
QoS Table	Where the ACL policies are defined
PBR Table	Where the policy based routing policies are defined. For details, see the <a href="#">Configuring Policy-Based Routing</a> section.
IPv6 Table	Where IPv6 policies are defined

You can view the configurable hardware tables by using the command:

```
CLI (network-admin@leaf-1) > vflow-table-profile-show
```

profile	hw-tbl	enable	flow-capacity	flow-slices-needed	flow-slices-used	comment
system	switch-main	enable	768	2	3	System-L1-L4-flows
application	switch-main	disable	0	1	0	User-Application
qos	switch-main	disable	0	1	0	QoS
ipv6	switch-main	disable	0	1	0	IPv6
pbr	switch-main	enable	0	0	0	PBR

**Note:** The capacity and availability of the hardware tables vary between switch models.

The optional tables (in grey in **Figure 10-3**) are disabled by default. You can enable optional tables with the `vflow-table-profile-modify` command. For example, enable the qos table using the command:

```
CLI (network-admin@leaf-1) > vflow-table-profile-modify profile qos enable
hw-tbl switch-main
```

You must reboot the switch or restart the nvOSd service for the settings to take effect. When you enable

optional hardware tables, Netvisor ONE allocates a minimum number of entries in the order of 256 vFlow objects (the number of vflow objects varies based on the platform and the type of the table). For maximum vFlow scalability, enable hardware tables only when necessary. You can monitor the resource consumption of active hardware tables with the following command:

```
CLI (network-admin@leaf-1) > vflow-table-show
```

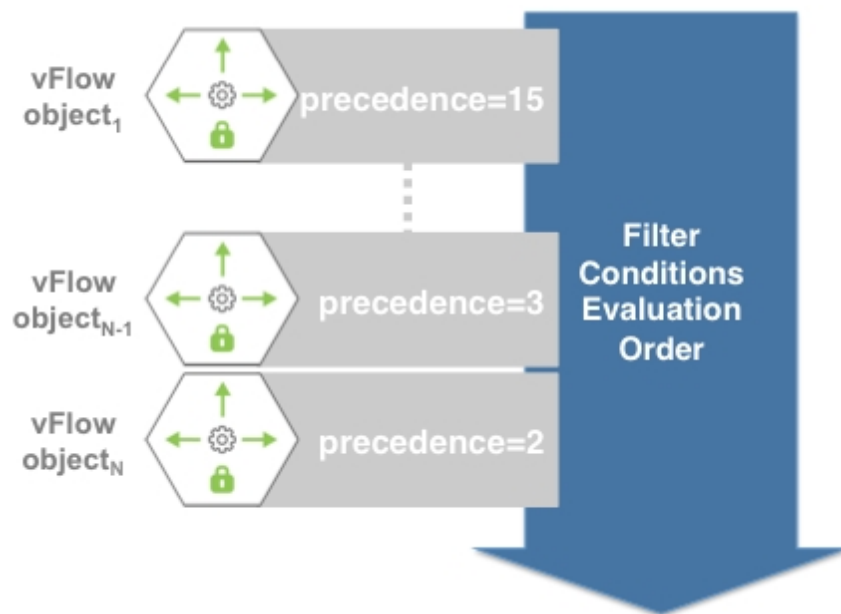
name	flow-max-per-group	flow-used	flow-tbl-slices	metadata	flow-profile
Egress-Table-1-0	512	0	1	match	system
System-L1-L4-Tun-1-0	2048	49	2	set,match	system
System-VCAP-table-1-0	512	0	1	set	system

## Precedence

When you implement two or more vFlow objects within the same hardware table, it may be necessary to enforce a particular evaluation order. Use the keyword `precedence` to enforce the evaluation order as Netvisor ONE executes vFlows with higher precedence value first. See a sample configuration below:

**Figure 10-4** displays the precedence or evaluation order for different vFlow objects. When a flow matches two or more vFlows with the same precedence, the corresponding vFlow actions are merged and executed together. When you create the vFlow, Netvisor ONE validates that the new object is consistent and can be merged with objects with the same precedence.

The precedence value is within a numerical range of 2 and 15, with 2 as the default value. You cannot configure the evaluation order or precedence value beyond 15.



**Figure 10-4: Evaluation Order for vFlow Objects with Different Precedence**

When you create multiple vflow objects within the same hardware table without specifying the precedence value (default value being 2), Netvisor displays an error message about the vFlow conflicts. For example,

- Create a vFlow:

```
CLI (network-admin@Leaf1) > vflow-create name example_vflow1 scope fabric  
bw flow-class meter bw-max 2g
```

- Create a second vFlow:

```
CLI (network-admin@Leaf1) > vflow-create name example_vflow2 scope fabric  
bw flow-class meter bw-max 5g src-ip 192.168.20.1
```

vflow-create: Flow conflicts with Flow example\_vflow1, ID68: specify fields to make flows mutually exclusive or change the flow precedence

The error message is generated because the vFlow configurations conflict with each other. To differentiate between the two vflows, assign a different precedence to example\_vflow2:

```
CLI (network-admin@Leaf1) > vflow-create name example_vflow2 scope fabric  
bw flow-class meter bw-max 5g src-ip 192.168.20.1 precedence 5
```

## Managing Traffic Classes with vFlow

The vFlow classes indicate the priority assigned to a packet within a switch for internal processing and prioritization and specifies a service type: traffic metering or traffic shaping, bandwidth guarantee. Netvisor ONE supports two types of vFlow classes:

- System Flow Classes
  - Metered flow class, where the traffic is not allowed to exceed a set rate.
  - Guaranteed bandwidth flow class, where the vFlow object guarantees a certain bandwidth and the switch priority is 9.
  - Lossless flow class, where drop action is unavailable and the switch priority is 10.
- User Defined Flow Classes
  - Flow classes created by users with priorities between 1 and 8
  - Used for traffic metering
  - Used for traffic shaping and bandwidth guarantee

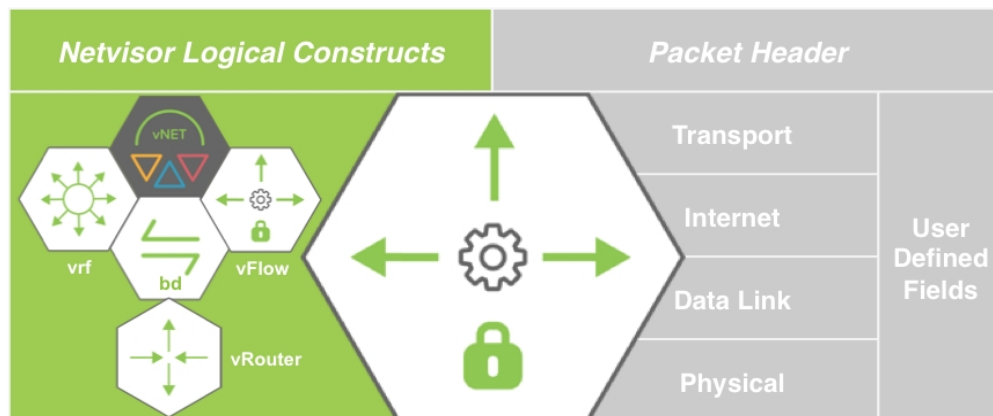
## Filtering of Traffic Flows

In the context of vFlows, filtering describes the traffic characteristics on which the network administrator intends to undertake a certain action. You can define the filter through one of the arbitrary set of matching qualifiers as illustrated in **Figure 10-5** and **Figure 10-6**:

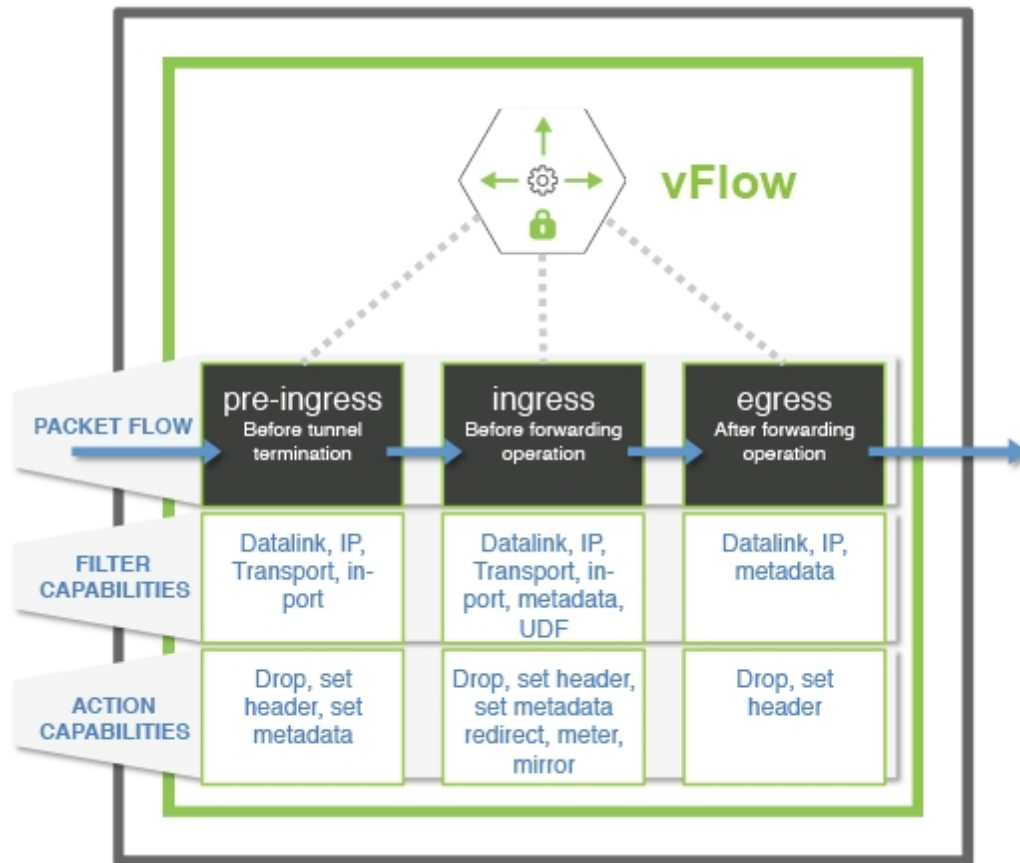
- Operating at an OSI or Internet Protocol layer, which is based on the packet header content

OR

- Based on Netvisor ONE logical constructs, such as vNET, vFlow metadata, or vRouter.



**Figure 10-5: vFlow Filtering Using Packet Header Fields and Netvisor Logical Constructs**



**Figure 10-6: vFlow Tables - Packet Forwarding Stage and Filter or Action Capabilities**

To define the vFlow filter using the components of OSI layer, use the following keywords or parameters in the `vflow-create` command:

### Filtering Qualifiers at Physical Layer

- `in-port port-list` — ingress physical interface or LAG interface identifiers. Netvisor ONE accepts values as a single value, a dash-separated value range, or comma-separated list of values and ranges.
- `out-port port-list` — egress physical interface or LAG interface identifier. Netvisor ONE accepts values as a single value, a dash-separated value range, or comma-separated list of values and ranges. This out-port is applicable for Egress\_table.

### Filtering Qualifiers at Data Link Layer

- `packet-res` — packet resolution in the ASIC. The resolution types can be L2-unicast, L2-unknown-unicast, L2-multicast, L2-unknown-multicast, L2-broadcast. For more details, see the [Enhancing the vFlow Capability to Match Forwarding Type and Packet Resolution in ASIC](#) section.
- `fwding-type` — ASIC forwarding type: VLAN, VXLAN, or VLE
- `vlan` — VLAN (Virtual LAN) identifier (IEEE 802.1q). Range is from 0 through 4095.

- `vxlan` — VxLAN Network Identifier or VNI
- `vxlan-ether-type` — Ethernet type such IPv4, ARP, WAKE, RARP, VLAN, IPv6, LACP, MPLS-uni, MPLS-multi, Jumbo, dot1X, AOE, Q-in-Q, LLDP, MACSEC, ECP, PTP, FCOE, FCOE-init, or Q-in-Q-old
- `vxlan-proto` — Protocol type for the VXLAN. Includes TCP, UDP, ICMP, IGMP, IP, ICMPv6
- `src-mac` — Source MAC address
- `src-mac-mask` — Mask for source MAC address
- `dst-mac` — Destination MAC address for the vFlow
- `dst-mac-mask` — Mask for destination MAC address
- `vlan-pri` — Class of Service (CoS) or VLAN priority (IEEE 802.1p), ranges from 0 through 7.

### **Filtering Qualifiers at Internet Layer**

- `src-ip` — Source IP address for the vFlow
- `src-ip-mask` — Mask for source IP address
- `dst-ip` — Destination IP address
- `dst-ip-mask` — Mask destination IP address
- `ttl` — Packet time-to-live
- `proto` — Layer 3 protocol for the vFlow
- `dscp` — 6-bit Differentiated Services Code Point (DSCP) for Quality of Service (QoS), in the range of 0 to 63
- `dscp-start` — Start value for DSCP
- `dscp-end` — End value for DSCP
- `tos` — Type of Service (ToS) value for Quality of Service (QoS)
- `tos-start` — Start value for Type of Service (ToS) range
- `tos-end` — End value for Type of Service (ToS) range

### **Filtering Qualifiers at Transport Layer**

- `src-port` — Source transport port
- `src-port-mask` — Mask for source transport port
- `src-port-end` — Ending port for a range of source ports. Use this qualifier along with `src-port` to specify a range of ports. This option is mutually exclusive with `src-port-mask`
- `dst-port` — Destination transport port

- `dst-port-mask` — Mask for destination transport port
- `dst-port-end` — Ending port for a range of destination ports. Use this qualifier along with `dst-port` to specify a range of ports. This option is mutually exclusive with `dst-port-mask`
- `tcp-flags` — Comma-separated list of TCP control flag values

### Filtering Qualifiers at User Defined Fields (UDFs)

- `udf-name[1-3]` — Reference to a User Defined Field (UDF) object, and defines advanced multi-layer filtering. Up to 3 objects are supported.
- `udf-data[1-3]` — Data value applied to the corresponding UDF object
- `udf-data[1-3]-mask` — Mask value applied to the corresponding UDF object

For details on configuring vFlows with UDFs, see the [Configuring vFlows with user Defined Fields](#) section.

### Netvisor ONE Logical Filtering

To define the vFlow filter using the Netvisor ONE logical constructs, use the following keywords or parameters in the `vflow-create` command:

- `metadata` — Metadata tag value assigned to packets along the internal hardware forwarding path, which can be used to correlate different vFlow objects operating at different ingress and egress stages
- `bridge-domain` — Logical abstraction of data-link learning and forwarding domain, implemented using a combination of physical interfaces, VLANs and VNI
- `vrouter-name` — Reference to an Internet layer VRF context defined on a local vRouter
- `vnet` — Virtual Network (vNET) value, used for identifying traffic belonging to a logical network partition for multi-tenancy and network segmentation purposes.



## Configuring vFlows with User Defined Fields (UDFs)

Netvisor allows you to define policy filters through one of the arbitrary set of matching qualifiers as explained in the [Filtering of Traffic Flows](#) section. One of the qualifier is the User Defined Field (UDF).

A UDF can match up to 128 bytes of a packet starting from the first byte of the packet. The length of the match can be from 1 to 4 bytes. Hardware with a Trident chip supports the creation of 8 UDF IDs. Each id can match a 2 byte portion of a packet. Creating a UDF with a length of 3 or 4 bytes requires 2 UDF IDs whereas a UDF with length of 1 or 2 bytes required 1 UDF id. The length specified for each UDF determines the total number of UDFs supported by Netvisor One. If you specify a length of 3 or 4 bytes, a maximum of 4 UDFs can be created. If you specify a length of 1 or 2 bytes, a maximum of 8 UDFs can be created.

**Limitation:** UDF offset range supported for UDF header packet-start type ranges from 0-63. This limitation is applicable for all NRU03 platforms.

A UDF adds a qualifier to the vFlow group, and you should create all UDFs before creating any vFlows. This feature is disabled by default, and you can enable it by using the following command:

```
CLI(network-admin@Spine1) > vflow-settings-modify enable-user-defined-flow|no-enable-user-defined-flow
```

<code>vflow-settings-modify</code>	Use this command to update a user vflow setting
Specify one of more of the following options	
<code>enable-user-defined-flow no-enable-user-defined-flow</code>	Specify to enable or disable the user defined flows.
<code>vxlan-analytics/no-vxlan-analytics</code>	Specify to enable or disable VXLAN analytics.
<code>inflight-vxlan-analytics/no-inflight-vxlan-analytics</code>	<p><b>Note:</b> You must disable VXLAN analytics before enabling the longlived tcp connection</p> <p>Specify to enable or disable the inflight VXLAN analytics.</p> <p><b>Note:</b> You must disable inflight VXLAN analytics before enabling the longlived tcp connection.</p> <p>Specify to enable or disable the long-lived TCP connection statistics.</p>
<code>longlived-tcp-conn-stats no-longlived-tcp-conn-stats</code>	<p><b>Note:</b> You must enable the user-defined-flow before enabling the longlived tcp connection statistics.</p>

To enable the user defined vflow, use the command:

```
CLI(network-admin@Spine1) > vflow-settings-modify enable-user-defined-flow
```

To disable the feature, use the command:

```
CLI(network-admin@Spine1) > vflow-settings-modify no-enable-user-defined-flow
```

**Note:** Reboot Netvisor OS for the changes (enable or disable commands) to take effect on the platform.

The command, `udf-create`, adds the qualifier to the UDF group in the hardware. This allocates UDF IDs based on the length. The command, `vflow-create`, has parameter fields to provide the data and mask to be matched by the vFlow. You can create vFlows with either one or two UDFs.

You cannot modify a UDF after adding it to a vFlow. You must delete the vFlow, modify the UDF, and re-create the vFlow with the modified UDF.

## New Commands for UDF

To create a new UDF, use the following command:

```
CLI(network-admin@Spine1) > udf-create name udf1 scope local offset 10
length 2 header packet-start
```

<code>udf-create</code>	Create the UDF qualifier list
<code>name <i>name-string</i></code>	Create the UDF name
<code>scope local fabric</code>	Scope for the UDF
<code>offset <i>number-bytes</i></code>	The offset in bytes. This is a value between 1 and 128.
<code>length <i>number-bytes</i></code>	The length in bytes. This is a value between 1 and 4 bytes.
<code>header [packet-start 13-outer 13-inner 14-outer 14-inner]</code>	The header from where offset is calculated.

To delete an UDF command:

```
CLI(network-admin@Spine1) > udf-delete name udf1
```

<code>udf-delete</code>	Delete UDF qualifier list
<code>name <i>name-string</i></code>	The name of the UDF to delete.

To modify an existing UDF command:

```
CLI(network-admin@Spine1) > udf-modify name udf1 scope local offset 20
length 4 header packet-start
```

<code>udf-modify</code>	Modify UDF qualifier list
<code>name name-string</code>	The name of the UDF to modify.
One or more of the following options:	
<code>offset number-bytes</code>	The offset in bytes. This is a value between 1 and 128.
<code>length number-bytes</code>	The length in bytes. This is a value between 1 and 4 bytes.
<code>header packet-start l3-outer l3-inner l4-outer l4-inner</code>	The header from where offset is calculated.

CLI(network-admin@Spine1) > `udf-show`

switch	name	scope	offset	length	header
spine1	u1	local	20	4	packet-start
spine1	u2	local	24	4	packet-start

<code>switch</code>	Displays the name of the switch
<code>udf-show</code>	Displays the UDF qualifier list
<code>name name-string</code>	Displays the UDF name
<code>scope local fabric</code>	Displays the scope for the UDF
<code>offset number-bytes</code>	Displays the offset in bytes. This is a value between 1 and 128.
<code>length number-bytes</code>	Displays the length in bytes. This is a value between 1 and 4 bytes.
<code>header packet-start l3-outer l3-inner l4-outer l4-inner</code>	Displays the header from where the offset is calculated.

The command, `vflow-create`, has the following additional parameters:

<code>udf-name1 udf-name</code>	Specify the name of the UDF.
<code>udf-data1 udf-data1-number</code>	Specify UDF data1q with the format 0xa0a0a01
<code>udf-data1-mask udf-data1-mask-number</code>	Specify the mask for udf-data with the format 0xffffffff.
<code>udf-name2 udf-name</code>	Specify the name of the UDF.
<code>udf-data2 udf-data2-number</code>	Specify UDF data2 with the format 0xa0a0a01
<code>udf-data2-mask udf-data2-mask-number</code>	Specify the mask for udf-data with the format 0xffffffff.

For example, to create a vflow with UDF parameters, use the command:

```
CLI(network-admin@Spine1) > vflow-create name udf1 scope local udf-name1
udf1 udf-data 0x0a0a0a01 udf-data-mask1 0xffffffff udf-name2 udf2 udf-data2
0x0a0a1400 udf-data-mask2 0xffffffff00
```

```
CLI(network-admin@Spine1) > vflow-show
```

name	scope	type	precedence	udf-name1	udf-data1	udf-data-mask1	udf-name2	udf-data2	udf-data-mask2
udf1	local	vflow	default	udf1	0xa0a0a01	0xffffffff	udf2	0xa0a1400	0xffffffff00

## Forwarding Action in vFlow Filters

The forwarding action defines the switch behavior for traffic that matches with the vFlow filter. Depending on the use case, the possible actions for the traffic flow include:

- Dropping of filtered traffic
- Regular or customized forwarding of filtered traffic
- Redirection or replication of filtered traffic to a switch processor, a physical port, or to an IP address destination.

### Access Control

To provide security access control policies that operate on high-speed distributed networks, the switch, which employs low-cost high-capacity technology like ASIC Ternary Content Addressable Memory (TCAM), is the ideal point of insertion. With Pluribus Adaptive Cloud Fabric, high-performance security access control using a **blacklist** or **whitelist** approach can be implemented consistently across the entire network using fabric-wide vFlow policies.

Netvisor ONE provides several vFlow actions for securing the network traffic with Access Control Lists (ACL) as described in the table :

**Table 10-2: vFlow Actions Supported in Netvisor ONE**

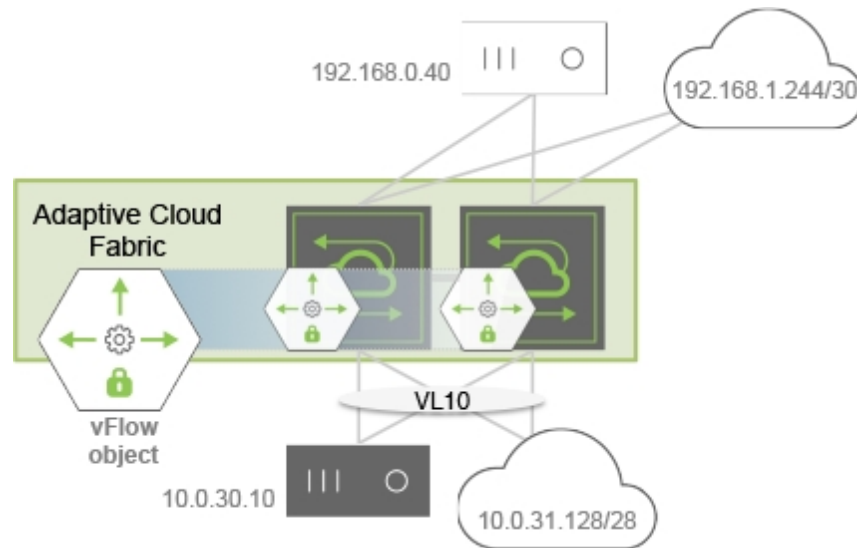
Action	Description
<code>action none</code>	Traffic is forwarded (permit filtered traffic) This is the default action.
<code>action drop</code>	Traffic matching the filter is removed from traffic path (block filtered traffic)
<code>action to-port</code>	Traffic matching the filter is forwarded to specified ports
<code>action to-cpu</code>	Traffic matching the filter is sent to CPU only
<code>action trap</code>	Traffic matching the filter is trapped to CPU
<code>action copy-to-cpu</code>	Traffic matching the filter is forwarded normally and copied to CPU, this action does not affect the traffic policy, but creates a copy of the policy to the CPU
<code>action copy-to-port</code>	Traffic matching the filter is forwarded normally and copied to port
<code>action check</code>	Traffic matching the filter is verified
<code>action setvlan</code>	VLAN ID is set for the traffic matching the filter
<code>action add-outer-vlan</code>	New outer VLAN tag is added to the traffic matching the filter
<code>action set tpid</code>	Traffic matching the filter is tagged with the provided TPID
<code>action to-port-set-vlan</code>	Traffic matching the filter is sent to provided port on specified VLAN

action tunnel-pkt	Tunnel frame is added for incoming traffic
action set-tunnel-id	Tunnel ID is set for the traffic matching the filter
action to-span	Traffic matching the filter is forwarded to the span ports
action cpu-rx	Stress test for cpu-rx
action cpu-rx-tx	Stress test for cpu-rx-tx
action set-metadata	Metadata is set for the traffic matching the filter
action set-dscp	DSCP value is set for the traffic matching the filter
action decap	Decap is set for the traffic matching the filter
action set-dmac	Destination MAC is set for traffic matching the filter
action to-next-hop-ip	The next hop IP address for traffic redirection
action set-dmac-to-port	Destination MAC is set and it is forwarded to port specified
action to-ports-and-cpu	Traffic matching the filter is forwarded to ports and CPU
action set-vlan-pri	Set VLAN priority for traffic matching the filter
action tcp-seq-offset	TCP sequence offset is set for traffic matching the filter
action tcp-ack-offset	TCP acknowledgment offset is set for traffic matching the filter
action l3-to-cpu-switch	Redirects Layer3 packets to CPU
action set-smac	Source MAC is set for the traffic matching the filter
action drop-cancel-trap	Traffic matching the flow is dropped and not trapped
action to-ecmp-group	ECMP group for traffic redirection
action redirect-to-vrouter	Redirect packets to vrouter
set-dscp	DSCP value is set for the traffic matching the filter
decap	Decap is set for the traffic matching the filter
set-dmac	Destination MAC is set for traffic matching the filter
to-next-hop-ip	The next hop IP address for traffic redirection
set-dmac-to-port	Destination MAC is set and it is forwarded to port specified
to-ports-and-cpu	Traffic matching the filter is forwarded to ports and CPU
set-vlan-pri	Set VLAN priority for traffic matching the filter
tcp-seq-offset	TCP sequence offset is set for traffic matching the filter
tcp-ack-offset	TCP ack offset is set for traffic matching the filter
l3-to-cpu-switch	Redirects l3 packets to CPU
set-smac	Source MAC is set for the traffic matching the filter
drop-cancel-trap	Traffic matching the flow is dropped and not trapped
to-ecmp-group	ECMP group for traffic redirection
redirect-to-vrouter	Redirect packets to vrouter

Typically the action resolution process gathers the actions from each slice match and decides on a collective action. However, when there is an action conflict, the collective action is not possible and the drop action takes higher priority. With the implementation of virtual slide mapping feature, action resolution is handled by the precedence or priority value.

## Understanding Blacklist Policy

You can implement access control policies using a blacklist approach, that is, approving all traffic by default and explicitly restricting certain traffic categories.



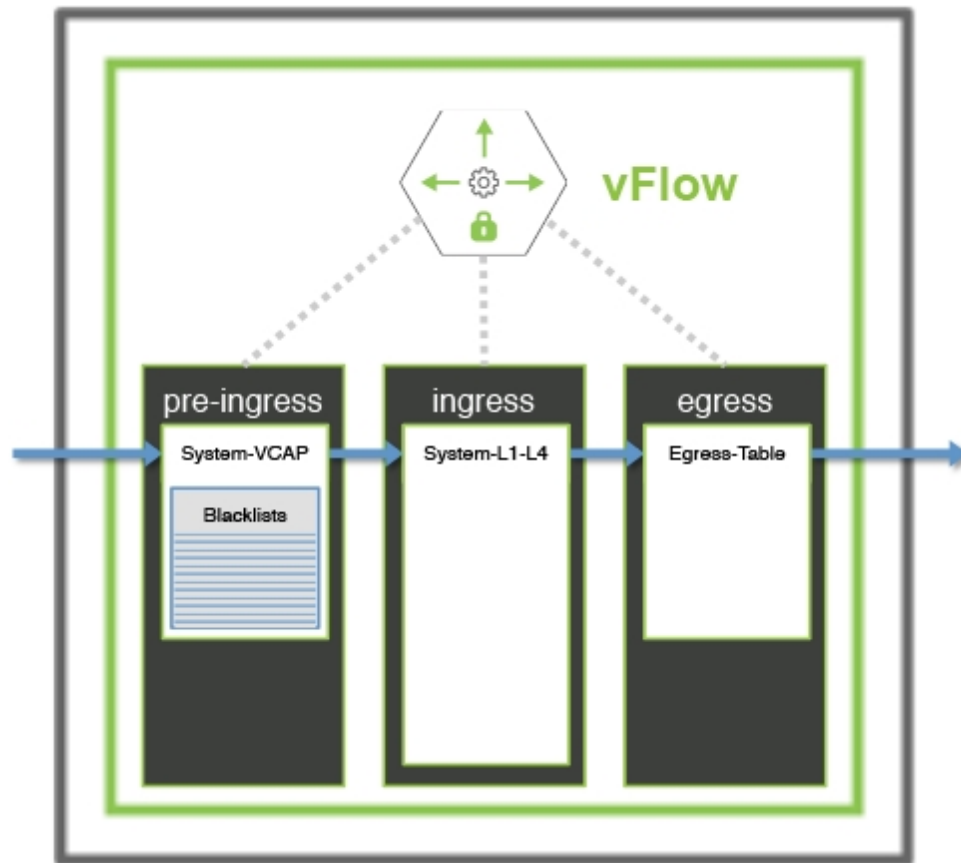
**Figure 10-7 - Blacklist Topology Example**

For example, as shown in **Figure 10-7**, you want to block HTTP traffic ingressing the switches in VLAN 10 going from the black host to the white host, while permitting all other traffic. Use the following parameters to configure a vFlow for this purpose:

- vFlow name: ACL01
- Filter: VLAN 10
- TCP port: 80
- Source IP address: 10.0.30.10
- Destination IP address: 192.168.0.40
- Forwarding Action: drop

```
CLI (network-admin@switch) > vflow-create name ACL01 scope fabric vlan 10
src-ip 10.0.30.10 dst-ip 192.168.0.40 proto tcp dst-port 80 action drop
```

For maximum ACL scalability, the blacklist policy can be also implemented using the System-VCAP table (see **Figure 10-8**) and provide additional capacity based on the switch models. In this approach, Netvisor ONE drops the blacklisted traffic at the pre-ingress stage without utilizing the resources in the default ingress table.



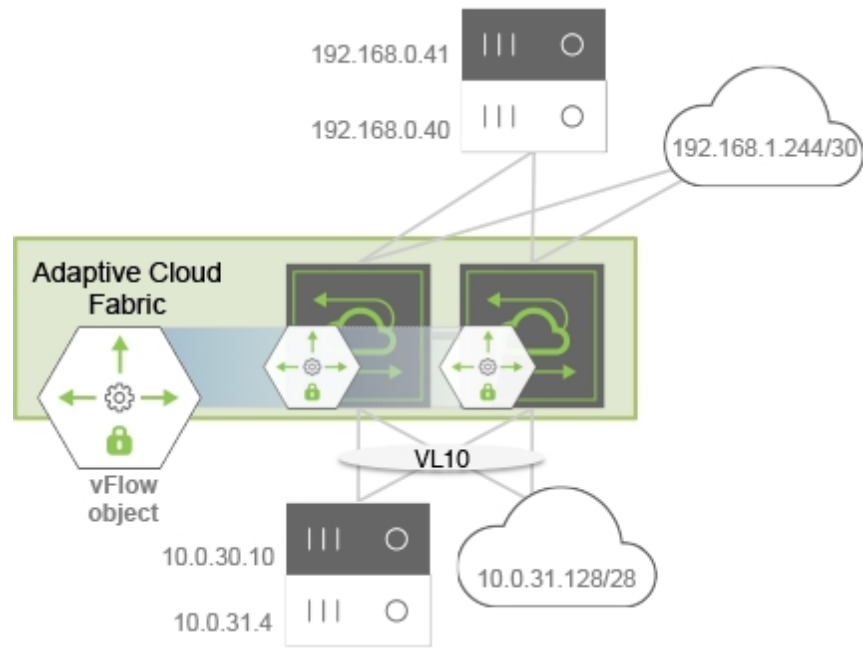
**Figure 10-8: Scaling the Blacklist ACLs using a Pre-Ingress Filter**

## Understanding Whitelist Policy

Contrary to the blacklist policy, the Whitelist model explicitly defines network traffic that is allowed and uses a default permit policy. **Figure10- 9** illustrates an example where you want to deny default traffic and approve traffic ingressing the switches on VLAN 10 for the following conditions:

- Traffic between the black hosts
- Traffic between the two subnets (clouds)
- Secure Shell (SSH) traffic between the white hosts where the host below (10.0.31.4) acts as a responder.





**Figure 10-9: Whitelist Topology Example**

To implement this policy, create four fabric-scoped vFlow objects, with the last one (**vFlow ACL04**) having the least (default) precedence and by using the following parameters:

#### 1. **vFlow ACL01**

- Filter: VLAN 10
- Source IP address: 10.0.30.10
- Destination IP address: 192.168.0.41
- Forwarding Action: none
- Precedence: 4

```
CLI (network-admin@switch) > vflow-create name ACL01 scope fabric vlan 10
precedence 4 src-ip 10.0.30.10 dst-ip 192.168.0.41
```

#### 2. **vFlow ACL02**

- Filter: VLAN 10
- Source IP address: 10.0.31.128/28
- Destination IP address: 192.168.1.144/30
- Forwarding Action: none
- Precedence: 4

```
CLI (network-admin@switch) > vflow-create name ACL02 scope fabric vlan 10
```

```
precedence 4 src-ip 10.0.31.128 src-ip-mask 255.255.255.250 dst-ip  
192.168.1.144 dst-ip-mask 255.255.255.252
```

### 3. vFlow ACL03

- Filter: VLAN 10
- TCP port: 22
- TCP flags: ACK
- Source IP address: 10.0.31.4
- Destination IP address: 192.168.0.40
- Forwarding Action: none
- Precedence: 4

```
CLI (network-admin@switch) > vflow-create name ACL03 scope fabric vlan 10  
precedence 4 src-ip 10.0.31.4 dst-ip 192.168.0.40 proto tcp dst-port 22  
tcp-flags ack
```

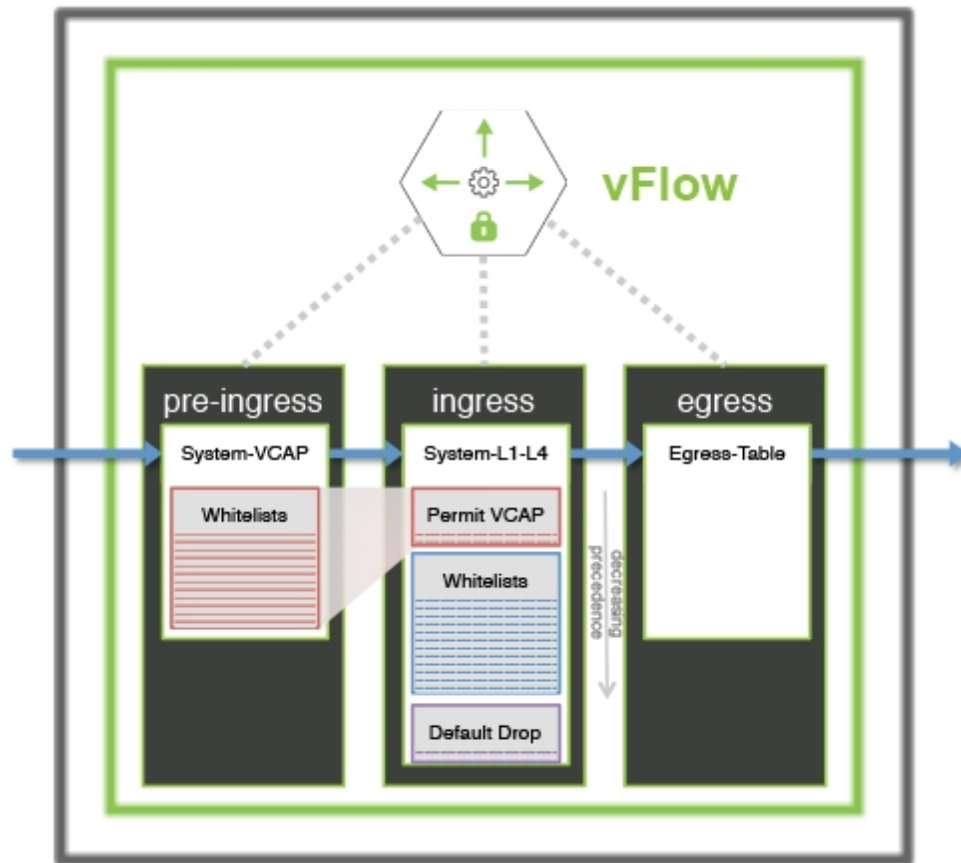
### 4. vFlow ACL04

- Filter: VLAN 10
- Forwarding Action: drop
- Precedence: 2 (default)

```
CLI (network-admin@switch) > vflow-create name ACL04 scope fabric vlan 10  
action drop
```

For maximum ACL scalability, you can implement the whitelist policy using the System-VCAP table and provide additional capacity for vFlow objects based on the switch models.

In this approach, Netvisor ONE examines Whitelisted traffic and labels the traffic with an arbitrary metadata value at the pre-ingress stage, without consuming resources in the default ingress table. During next ingress stage, Netvisor ONE uses a single vFlow object with highest precedence to permit traffic already allowed at pre-ingress stage. You can also build the same Whitelist policy using both System-VCAP and the default System-L1-L4 tables as shown in **Figure 10-10**.



**Figure 10-10: Scaling Whitelist ACLs Using a Pre-ingress Filter**

## Configuring vFlow Filters

---

A vFlow filter, in conjunction with a port mirror, gives granular control over the traffic that is mirrored through SPAN or RSPAN configurations. By configuring a vFlow with a mirror, you can select the traffic you need for analysis with precision.

To create a vFlow-mirror, you should configure a port mirror first. For details, see the *Configuring Port Mirroring* section in the *Configuring and Using Network Management and Monitoring* chapter.

For example:

```
CLI (network-admin@switch) > mirror-create name mir11 out-port 70 in-port 12 span-encap over-vlan span-tagging-vlan 300
```

To create the corresponding vFlow, use the `vflow-create` command:

```
CLI (network-admin@switch) > vflow-create name span1 scope local src-ip 100.1.1.200 precedence default action none mirror mir11
```

The `vflow-create` command allows numerous filtering policies. Refer to the previous sections of this chapter, *Configuring and Using vFlows* for more information.

A logical combination of a port mirror and a vFlow-based one can be configured using the `filtering` parameter in the `vflow-create` command.

- Use the `port` option to consider only the parameters configured in the `mirror-create` command for filtering the traffic.
- Use the `vflow-or-port` option to mirror traffic that meets either the vFlow or the mirror constraints. With this option, packets that match either the vFlow policy or the `in-port` parameter of the mirror get mirrored.
- Use the `vflow-and-port` option to mirror traffic that meets both the vFlow and the mirror constraints. With this option, only packets that match both the vFlow policy and the `in-port` parameter of the mirror get mirrored.

For example:

```
CLI (network-admin@switch) > mirror-create name mir5 out-port 80 in-port 40 filtering vflow-and-port span-encap over-vlan span-tagging-vlan 300
```

```
CLI (network-admin@switch) > vflow-create name flow1 scope local tos 112 action none mirror mir5
```

With the above configuration, only the packets that ingress on port 40 of switch1 with a ToS value of 112 are mirrored.

## Refreshing vFlow Level Statistics for Long-lived Connections

Prior to Netvisor ONE 5.1.1 release, the connection analytics displayed the connection statistics (incoming bytes, outgoing bytes, total bytes, and the age of the connection) only after the connection is completed, which worked well for short-timed connections. However, for long-lived TCP connections, the connection statistics was unreliable. This was due to the fact that the parameters involved were calculated with reference to the TCP sequence numbers, which, for a long-lived connection, always wrapped around.

To eliminate inaccuracies in long-lived TCP connection statistics, the TCP data packets with sequence numbers that are about to wrap around are sent to the CPU. This is implemented by defining a new vflow rule (policy). Netvisor ONE provides an option to enable this functionality through the `vflow-settings-modify` command. When the feature is active, the `connection-stats-show` and `connection-show` commands show accurate outputs for long-lived TCP connections.

### Note:

- To enable *long-lived TCP connection statistics*, you must first enable the *user-defined-flow* knob.
- Disable *vxlan-analytics* before enabling the *long-lived TCP connection statistics* knob.
- You cannot enable *long-lived TCP connection statistics* knob if the *inflight-vxlan-analytics* is enabled or vice-versa.

**Note:** You must restart nvOSd when you enable or disable the long-lived TCP connection statistics knob.

Use the `vflow-settings-modify` command to enable long-lived TCP connection statistics:

```
CLI (network-admin@switch) > vflow-settings-modify
```

<code>vflow-settings-modify</code>	Use this command to update a user vflow setting.
Specify one or more of the following options:	
<code>enable-user-defined-flow no-enable-user-defined-flow</code>	Specify to enable or disable the user defined flows.  <b>Note:</b> You must enable the user-defined-flow before enabling the longlived tcp connection statistics.
<code>vxlan-analytics no-vxlan-analytics</code>	Specify to enable or disable VXLAN analytics.  <b>Note:</b> You must disable VXLAN analytics before enabling the longlived tcp connection
<code>inflight-vxlan-analytics no-inflight-vxlan-analytics</code>	Specify to enable or disable the inflight VXLAN analytics.  <b>Note:</b> You must disable inflight VXLAN analytics

	before enabling the longlived tcp connection.
<pre>longlived-tcp-conn-stats   no- longlived-tcp-conn-stats</pre>	Specify to enable or disable the long-lived TCP connection statistics.

For example, to enable the long-lived TCP connection statistics, use the commands below:

```
CLI (network-admin@Leaf1) > vflow-settings-modify enable-user-defined-flow
```

```
CLI (network-admin@Leaf1) > vflow-settings-modify no-vxlan-analytics no-
inflight-vxlan-analytics
```

```
CLI (network-admin@Leaf1) > vflow-settings-modify longlived-tcp-conn-stats
```

To view the user vflow settings, use the command `vflow-settings-show`. For example, after enabling long-lived TCP connection statistics, the typical output would be:

```
CLI (network-admin@Leaf1) > vflow-settings-show
```

```
enable-user-defined-flow: on
vxlan-analytics:         off
inflight-vxlan-analytics: off
longlived-tcp-conn-stats: on
```

To view the connection statistics, use the show commands:

```
CLI (network-admin@Leaf1) > connection-stats-show
```

vlan	ip	port	iconns	oconns	ibytes	obytes	total-bytes
100	132.10.3.152	32	402617119		813G	809G	1.58T
100	132.10.3.113	32	402439803		822G	818G	1.60T
100	132.10.3.191	32	402379008		828G	822G	1.61T
100	132.10.3.160	32	402531295		828G	824G	1.61T
100	132.10.3.147	32	402620992		833G	829G	1.62T
100	132.10.3.131	32	402466573		840G	836G	1.64T

```
CLI (network-admin@Leaf1) > connection-show
```

vlan	src-ip	dst-ip	dst-port	cur-state	latency	obytes	ibytes	total-bytes	age
100	132.10.3.2	132.10.3.127	http	fin	67.9us	198	188	386	1s
100	132.10.3.2	132.10.3.186	http	fin	62.3us	198	188	386	1s
100	132.10.3.2	132.10.3.153	http	fin	511us	198	188	386	1s
100	132.10.3.2	132.10.3.205	http	fin	66.4us	198	188	386	1s
100	132.10.3.1	132.10.3.160	http	fin	305us	198	188	386	1s

## Commands and Parameters Applicable to vFlow Traffic

---

The vFlow feature includes several commands and keyword parameters to configure and monitor various vFlow actions. Some of the important commands and parameters are explained in this section.

- To create a new vFlow object, use the command:

```
CLI (network-admin@switch-1) > vflow-create name <vflow-name> scope [local|fabric] {parameters}
```

- To modify an existing vFlow object,

```
CLI (network-admin@switch-1) > vflow-modify name <vflow-name> {parameters}
```

- To delete an existing vFlow object,

```
CLI (network-admin@switch-1) > vflow-delete name <vflow-name>
```

- To display existing vFlow objects,

```
CLI (network-admin@switch-1) > vflow-show {parameters}
```

The key parameters required to create a vFlow object are categorized as:

- Scope (2)
  - local OR
  - fabric
- Tables (4)
  - <blank>
  - table-name Egress-Table-1-0
  - table-name System-L1-L4-Tun-1-0
  - table-name System-VCAP-table-1-0
- Match Conditions: Specify one or more match conditions:
  - bridge-domain name
  - vlan
  - vnet
  - in-port <port>
  - out-port <port>
  - ether-type <type>
  - src-mac <mac> src-mac-mask <mask>
  - dst-mac <mac> dst-mac-mask <mask>
  - src-ip <ip> src-ip-mask <mask>
  - dst-ip <ip> dst-ip-mask <mask>
  - src-port <port> proto <tcp|udp>
  - dst-port <port> proto <tcp|udp>
  - src-port <port> src-port-mask <mask>

- dst-port <port> dst-port-mask <mask>
- src-port <port> src-port-end <port>
- dst-port <port> dst-port-end <port>
- dscp-start <start value> dscp-end <end value>
- tos-start <start value> tos-end <end value>
- tos <tos>
- vlan-prio <802.1p priority>
- ttl <ttl>
- proto <IP proto>
- ingress-tunnel <tunnel>
- udf-name1 <name> udf-data1 <data> udf-data1-mask <mask>
- udf-name2 <name> udf-data2 <data> udf-data2-mask <mask>
- udf-name3 <name> udf-data3 <data> udf-data3-mask <mask>
- Special Actions
  - <blank>
  - bw-min <bw>
  - bw-max <bw>
  - burst-size <size>
  - precedence <IP precedence>
  - process-mirror
  - no-process-mirror
  - log-stats stats-interval <sec> dur <sec>
  - set-src <ip> set-src-port <port>
  - set-dst <ip> set-dst-port <port>
  - log-stats
  - no-log-stats
  - transient
  - no-transient
  - enable
  - no-enable
  - log-packets
  - no-log-packets
- Specific Actions with **Action** Keyword
  - action none
  - action drop
  - action to-port action-to-ports-value <port>
  - action to-cpu
  - action trap
  - action copy-to-cpu
  - action copy-to-port
  - action check
  - action setvlan action-value <vlan>
  - action add-outer-vlan
  - action set-tpid
  - action to-port-set-vlan
  - action tunnel-pkt
  - action set-tunnel-id
  - action to-span
  - action cpu-rx



- action cpu-rx-tx
- action set-metadata
- action set-dscp
- action decap
- action set-dmac
- action to-next-hop-ip
- action set-dmac-to-port
- action to-ports-and-cpu
- action set-vlan-pri
- action tcp-seq-offset
- action tcp-ack-offset
- action l3-to-cpu-switch
- action set-smac
- action drop-cancel-trap
- action to-ecmp-group
- action redirect-to-vrouter
- Flow Class
  - flow-class meter
  - flow-class guaranteed\_bw
  - flow-class lossless
  - flow-class class0
  - flow-class class1
  - flow-class class2
  - flow-class class3
  - flow-class class4
  - flow-class class5
  - flow-class class6
  - flow-class class7
  - flow-class class8
  - flow-class control
- Bandwidth parameters
  - bw-min <min>
  - bw-max <max>

## Guidelines and Limitations

---

When you configure the vFlow policies, it is recommended to follow the guidelines and limitation mentioned in this section.

**Guideline:** While specifying the `burst-size` in the `vflow-create` and `vflow-modify` commands, ensure that:

- The `burst-size` is large enough to handle the maximum transmission unit (MTU) size of the packets.
- The `burst-size` limit should not be set lower than 10 times the MTU size of the traffic on the interface to be policed.  
If the configured `burst-size` value is less than ten times the MTU size, then Netvisor ONE takes the default burst-size value for configuration. If you configure the burst-value higher than ten times the MTU size, then the configured value is accepted.

## Examples and Use Cases in vFlow

---

- [Supporting TCP Parameters using vFlows](#)
- [Configuring Burst Size in vFlow for Maximum Bandwidth](#)
- [Configuring Bandwidth Sharing for a Single VLAN](#)
- [Enhancing the vFlow Capability to Match Forwarding Type and Packet Resolution in ASIC](#)

## Supporting TCP Parameters using vFlows

---

Packet Broker requires the ability to create flows based on TCP control bits in a packet. The commands, `vflow-create` and `vflow-modify` have a new option `tcp-flags`. The supported TCP control bits include FIN, SYN, RST, PUSH, ACK, and URG.

Setting the ACK bit is supported only if it is combined with other TCP bits such as SYN and FIN and not as a single parameter.

Only `to-port` and `mirror` actions are supported by vFlow with `tcp-flags` filter. The actions added for vFlows with `tcp-flags` configured are `mirror-to-port`.

If analytics is enabled, then `copy-to-cpu` are also applied on the same vFlow. Also, these flows are created with a precedence of 3 or above.

System vFlows are created with precedence 2 so that analytics can also work even with these vFlows.

To create a vFlow for the default system table, use the following syntax:

```
CLI (network-admin@Spine1) > vflow-create name Redirect-TCP-Reset tcp-flags RST action to-port
```

```
CLI (network-admin@Spine1) > vflow-create name Redirect-TCP-ECN-Capable tcp-flags ECN,RST action to-port
```

```
CLI (network-admin@Spine1) > vflow-create name Mirror-TCP-Finished tcp-flags FIN action mirror
```

You can use the `vflow-table-show` command to display vFlow tables:

```
CLI (network-admin@Spine1) > vflow-table-show format all layout vertical
```

```
switch:      Spine1
name:        Egress-Table-1-0
id:          a0000d7:1
flow-max:    1024
flow-used:   0
flow-tbl-slices: 1
capability:  match-metadata
flow-tbl-bank: Egress
flow-profile: system
switch:      Spine1
name:        Decap-Table-1-0
id:          a0000d7:2
flow-max:    1024
flow-used:   0
flow-tbl-slices: 2
capability:  none
flow-tbl-bank: Match-Metadata
```

```
flow-profile:      vxlan
switch:            tac-f64-sw5
name:              OpenFlow-L2-L3-1-0
id:                a0000d7:3
flow-max:          1024
flow-used:         0
flow-tbl-slices:   7
capability:        none
flow-tbl-bank:     Match-Metadata
flow-profile:      openflow
```

## Configuring Burst Size in vFlow for Maximum Bandwidth

---

The `vflow-create` and `vflow-modify` commands support a configurable burst-size parameter. This feature enables you to specify different burst-sizes for different types of metered traffic. For example, you can configure higher burst levels for a metered application that may produce bursty traffic patterns when you click on it, such as a media-rich Web page link.

This feature defaults to burst-size auto, which auto-calculates the burst size based on the maximum bandwidth settings for the vFlow. You can configure a burst-size number between 0 through 134MB.

The command syntax is:

```
CLI (network-admin@switch) > vflow-create name name-string scope local |  
fabric in-port port-list bw-max bw-max-number burst-size number
```

For example, to create a vFlow with a burst size of 12 MB, use the following syntax:

```
CLI (network-admin@switch) > vflow-create name flow1 scope local in-port 12  
bw-max 5G burst-size 12M
```

## Configuring Bandwidth Sharing for a Single VLAN

---

In some instances, you may want to configure bandwidth sharing for a single VLAN with different IP addresses or subnets.

To do this, you must create a VRG with the required bandwidth:

```
CLI (network-admin@Leaf1) > vrg-create name admin-vrg vlans 100 data-bw-  
min 1g data-bw-max 2g scope fabric
```

You have now created a VRG with the guaranteed bandwidth of 1 Gbps and limited to a maximum of 2 Gbps. Now, create a vFlow for each IP address:

```
CLI (network-admin@Leaf1) > vflow-create name vfl-1 scope fabric vlan 100  
src-ip 1.1.1.1
```

```
CLI (network-admin@Leaf1) > vflow-create name vfl-2 scope fabric vlan 100  
src-ip 2.2.2.2
```

```
CLI (network-admin@Leaf1) > vflow-create name vfl-3 scope fabric vlan 100  
src-ip 3.3.3.3
```

```
CLI (network-admin@Leaf1) > vflow-create name vfl-4 scope fabric vlan 100  
src-ip 4.4.4.4
```

In this example, the specified IP addresses each have a guaranteed bandwidth between 1 Gbps and 2 Gbps.

If you want to specify a subnet, 100.100.100.0/28, and VLAN 53 with maximum bandwidth of 50 Mbps, use the following syntax:

```
CLI (network-admin@Leaf1) > vrg-create name vrg-custom scope fabric data-  
bw-min 50M data-bw-max 50M vlan 53
```

```
CLI (network-admin@Leaf1) > vflow-create name vfl-cust scope fabric src-ip  
100.100.100.0 src-ip-mask 255.255.255.240 vlan 53
```

However, later on, you found that sixteen IP addresses were not enough and you needed an additional 8 with the subnet, 101.101.101.8/29 that require the same bandwidth as the previous subnet. Use the following syntax:

```
CLI (network-admin@Leaf1) > vflow-create name vfl-cust-2 scope fabric src-  
ip 101.101.101.8 src-ip-mask 255.255.255.248 vlan 53
```

You now have two vFlows on VLAN 53.

Then, you discover that 50 Mbps is not sufficient to support the network traffic affected by the vFlow, and you want to upgrade to 80 Mbps:

```
CLI (network-admin@Leaf1) > vrg-modify name vrg-custom data-bw-min 80M
```

data-bw-max 80M



## Enhancing the vFlow Capability to Match Forwarding Type and Packet Resolution in ASIC

On certain platforms, where the VXLAN routing is supported using recirculation of packets by leveraging the vxlan-loopback-trunk parameter, the Layer 2 entries for route RMAC address, VRRP MAC address on VXLAN VLAN, or the Virtual Forwarder Interface (VFI) are programmed to point to vxlan-loopback-trunk ports in the hardware. As a result, any Layer 2 unicast packets destined for route RMAC address or the programmed VFIs do not reach the vRouter. Netvisor ONE allows you to mitigate this problem by enabling you to create vFlow objects and specify the desired policy.

To create the vFlow object and to enable the match forwarding and packet resolution capability, use the command:

```
CLI (network-admin@switch) > vflow-create name <name-string> scope [local|
fabric] in-port <port-list> fwding-type [vlan|vxlan|vle] packet-res [l2-
unicast|l2-unknown-unicast|l2-multicast|l2-unknown-multicast|l2-broadcast]
action [none|drop|to-port|to-cpu|trap|copy-to-cpu|copy-to-port|check|
setvlan|add-outer-vlan|set-tpid|to-port-set-vlan|tunnel-pkt|set-tunnel-id|
to-span|cpu-rx|cpu-rx-tx|set-metadata|set-dscp|decap|set-dmac|to-next-hop-
ip|set-dmac-to-port|to-ports-and-cpu|set-vlan-pri|tcp-seq-offset|tcp-ack-
offset|l3-to-cpu-switch|set-smac|drop-cancel-trap|to-ecmp-group|redirect-
to-vrouter]
```

name <name-string>	Specify the name of the vFlow object.
scope [local fabric]	Specify if the scope of the vFlow object is local or fabric
in-port <port-list>	Specify the incoming port for the vFlow object
fwding-type [vlan vxlan vle]	Specify the ASIC forwarding type
packet-res [l2-unicast l2-unknown-unicast l2-multicast l2-unknown-multicast l2-broadcast]	Specify the packet resolution in the ASIC
action [none drop to-port to-cpu trap copy-to-cpu copy-to-port check setvlan add-outer-vlan set-tpid to-port-set-vlan tunnel-pkt set-tunnel-id to-span cpu-rx cpu-rx-tx set-metadata set-dscp decap set-dmac to-next-hop-ip set-dmac-to-port to-ports-and-cpu set-vlan-pri tcp-seq-offset tcp-ack-offset l3-to-cpu-switch set-smac drop-cancel-trap to-ecmp-group redirect-to-vrouter]	Specify the forwarding action to apply to the vFlow object

For example, to create a vFlow object: *vflow1*, scope: *local*, in-port number (port number of vxlan-loopback-trunk): *397*, with forwarding type: *vxlan*, packet resolution in ASIC as *l2-unicast* and forwarding action to be applied to the vFlow object as *redirect-to-vrouter*, use the command:

```
CLI (network-admin@switch) > vflow-create name vflow1 scope local in-port  
397 fwding-type vxlan packet-res l2-unicast action redirect-to-vrouter
```

In the above example, port **397** is the port number of the vxlan-loopback-trunk and the action **redirect-to-vrouter** redirects the packets unmodified to data port.

To view the details, use the command:

```
CLI (network-admin@switch) > vflow-show
```

name	scope	type	in-port	burst-size	precedence	action	packet-res	fwding-type	enable	table-name
vflow1	local	vflow	397	auto	13	redirect-to-vrouter	l2-unicast	vxlan	enable	System-L1-L4-Tun-1-0

## Using Application Flows and Statistics

---

- [Displaying Statistics for vFlows for all Switches in the Fabric](#)
- [Understanding vFlow Statistics](#)

## Displaying Statistics for vFlows for all Switches in the Fabric

To create vFlows across the entire fabric, configure the vFlow with the scope `fabric` and `stats enable` option. By using these parameters, you can enable statistics for the flow on all switches that are members of the fabric and you can display the statistics for any switch in the fabric.

For example, to create a vFlow for VLAN1 with the scope `fabric`, use the following command:

```
CLI (network-admin@Leaf1) > vflow-create name example_fabric_flow1 scope
fabric log-stats enable vlan 1
```

To display the statistics for the new vFlow for a switch in the fabric, use the following syntax:

```
CLI (network-admin@Leaf1) > switch switch-name vflow-stats-show name
example_fabric_flow1
```

name	packets	bytes	cpu-packets	cpu-bytes
-----	-----	-----	-----	-----
example_fabric_flow1	51.4K	13.8M	50.1K	13.1M

If you omit the switch name, all vFlow statistics for the fabric are displayed.

name	packets	bytes	cpu-packets	cpu-bytes
-----	-----	-----	-----	-----
example_fabric_flow1	1.32K	305K	1.29K	291K
example_fabric_flow1	910	256K	884	243K

## Understanding vFlow Statistics

The virtual network-based flows- vflows, display statistics for packet traffic flows on a switch and across the fabric. The vFlows are very powerful and provide many features such as quality of service (QoS), traffic shaping, packet redirect, drop actions, mirror, and capture.

A vFlow can be configured to store log statistics to a file accessible to clients using NFS and SFTP. If statistics logging is enabled, Netvisor ONE periodically polls the switch for the most recent statistics for each flow and saves the statistics to an exported file. Netvisor ONE also saves individual statistics received from other switches in the fabric and combines the statistics from all switches to record aggregate statistics for the entire fabric.

The switch consists of two components, the switch and the server. vFlows with operations such as drop executes within the switch component. Some vFlows operations for QoS take place in the switch component, while others operate within the co-processor by directing pertinent traffic to the co-processor.

There, the traffic is managed and then sent back to the switch component. Other actions such as copy-to-cpu sends the match traffic to the server component where the traffic is managed and then forwards packets for delivery. In general, the details are managed by Netvisor ONE including fabric scope commands that cause all switches within a fabric to participate in an operation and then sends the compiled results to the CLI or to log files.

Before you can access the files, you must enable NFS or SFTP access to the log files by using the `admin-service-modify` command.

```
CLI (network-admin@switch-1) > vflow-share-show
```

vnet	enable	share-path
----	-----	-----
fab1-global	no	switch-1://fab1-global
fab1-global	no	switch-1://fab1-global
fab1-global	no	switch-1:///fab1-global
fab1-global	no	switch-1://fab1-global
fab1-global	no	switch-1://fab1-global

```
CLI (network-admin@switch) > vflow-share-modify fab1-global enable
```

```
CLI (network-admin@switch) > vflow-share-show
```

vnet	enable	share-path
-----	-----	-----
fab1-global	yes	switch-1://fab1-global
fab1-global	yes	switch-1://fab1-global
fab1-global	no	switch-1://fab1-global

fabl-global	no	switch-1://fabl-global
fabl-global	no	switch-1://fabl-global

You can then access the statistics log files using NFS in the following locations:

For the switch scope, the files are located in: /net/switch-name//-name/flow/flow-name/switch/switch-name/stats

For the fabric scope, the files are located in: /net/switch-name//-name/flow/flow-name/fabric/stats

To create a vFlow for example, Host-Agent-Discover, and measure statistics, enter the following command:

```
CLI (network-admin@switch) > vflow-create name Host-Agent-Discover scope
local system
```

To view all vFlows currently tracked by the switch or fabric, use the vflow-show command:

```
CLI (network-admin@switch) > vflow-show
```

```
switch:           pleiades24
name:             Host-Agent-Discover
scope:           local
type:            system
dst-ip:          224.4.9.6
precedence:      2
action:          copy-to-cpu
switch:         pleiades24
name:           DHCP-client
scope:         local
type:         system
in-port:      1-68
src-port:     68
proto:        udp
precedence:   2
action:       copy-to-cpu
switch:      pleiades24
name:       Host-Agent-Discover
scope:     local
type:     system
dst-ip:   224.4.9.6
precedence: 2
action:   copy-to-cpu
switch:  pleiades24
name:   DHCP-client
scope: local
type:  system
in-port: 1-68
src-port: 68
proto:  udp
```

```
precedence:      2
action:          copy-to-cpu
```

From the information displayed in the output, you can review the switch, the name of the vFlow, scope, type of vFlow, destination IP address, precedence, and action for the vFlow.

To display statistics for all vFlows, use the `vflow-stats-show` command:

```
CLI (network-admin@switch) > vflow-stats-show
```

name	packets	bytes	cpu-packets	cpu-bytes
-----	-----	-----	-----	-----
IGMP-Flow	368K	23.0M	392K	23.0M
LLDP-Flow	82.9K	26.3M	82.9K	26.0M
Host-Agent	17.8K	1.11M	0	0
ECP	0	0	0	0

To monitor statistics of a vFlow and update every 10 seconds, use the following syntax:

```
CLI (network-admin@switch) > vflow-stats-show name flow1 show-diff-interval
10
```

To log persistent records of flow statistics, use the logging parameter and collect statistics every 10 seconds:

```
CLI (network-admin@switch) > vflow-create name monitor-flow scope local
ether-type arp stats log stats-interval 5
```

You can display the statistics logs for the new flow using the `vflow-stats-show` command.

**Note:** Conflicting vFlows - Multiple vFlows can be active at once, but cannot apply them at the same time. You can use the precedence parameter to set the order of the vFlows. If you set the precedence to a higher value (0 - 10 with 0 as the lowest precedence), the vFlow has a higher precedence than those with lower values. If you are seeing error messages about vFlow conflicts, try adding a precedence value to new or existing vFlows.

## Examples and Use Cases for Network Monitoring and Security

---

- [Using vFlows to Disable Communication for Security Monitoring](#)
- [Configuring vFlows to Filter Packets on Management Interfaces](#)
- [Configuring vFlow for Analytics](#)



## Using vFlows to Disable Communication for Security Monitoring

---

You can use vFlows to control the traffic by specifying the communications that are not allowed in a switch or a fabric. Use the following steps to create a vFlow as a firewall:

Define a VLAN and destination IP-based flow and specify that the flow is dropped by the switch, with statistics monitoring enabled:

```
CLI (network-admin@Leaf1) > vflow-create name vflow10 scope local vlan 99
dst-ip 172.168.24.1 action drop stats enable
```

Display the statistics for the new flow above as the traffic is dropped:

```
CLI (network-admin@Leaf1) > vflow-stats-show name vflow10 show-diff-
interval 5
```

switch	name	packets	bytes	cpu-packets	cpu-bytes
-----	----	-----	-----	-----	-----
Leaf1	vflow10	864	116K	0	0
Leaf1	vflow10	5	936K	0	0

There are many options available for creating vFlows, and vFlows can be used to shape traffic, capture statistics, capture flow metadata, capture packets, or manage communications. The options include:

- vlan
- in-port
- out-port
- ether-type
- src-mac
- src-mac-mask
- dst-mac
- dst-mac-mask
- src-ip
- src-ip-mask
- dst-ip
- dst-ip-mask
- src-port
- dst-port
- dscp
- tos
- proto
- flow-class
- uplink-ports
- bw-min
- bw-max
- precedence
- action

- action-value
- no-mirror
- mirror
- no-process-mirror
- process-mirror
- no-log-packets
- log-packets
- packet-log-max
- stats
- stats-interval
- duration
- no-transient
- transient
- vxlan
- vxlan-ether-type
- vxlan-proto

## Configuring vFlows to Filter Packets on Management Interfaces

The Pluribus Networks switches support administrative services and protocols such as SSH, HTTP, SSL, ICMP, etc. (for all supported protocols, see the show command output below). Management vflow feature enables the use of IPTables to support filtering based on filter parameters on management interfaces traffic.

The management traffic on Pluribus switches are handled in two ways:

- Out-of band management interface traffic: Uses IPTables to perform kernel based filtering
- In-band management interface traffic: Uses vflow based programming approach

In dual stack networks, both IPv4 and IPv6 filters can be used on the management port (in-band/out-of-band). By default, the management traffic allows all SSH, NFS, SNMP traffic and denies all web traffic as displayed in the command output below:

```
CLI (network-admin@spine1) > admin-service-show
```

if	ssh	nfs	web	web-ssl	web-ssl- port	web-port	snmp	net-api	icmp
----	----	----	-----	-----	-----	-----	-----	-----	-----
mgmt	on	on	off	off	443	80	on	on	on
data	on	on	off	off	443	80	on	on	on

This feature uses the existing vflow commands to add filters on the out-of-band and in-band management interfaces that are specific for these administrative services. The vflow rules uses precedence numbering to maintain the order of filters and helps in enforcing rules at specific locations in the IPTables. However, when you configure vflow rules, make sure that the vflow rules do not have a conflict with the system rules because the system rules may take precedence over the user configured vflow rules.

While configuring the vflow rules, be aware of the following configuration considerations:

- The parameter if is used to configure management vflows.
- The vflow rules support only permit and drop actions.
- The order of the configuration aligns with the order in which the rules are programmed. However, the user can re-arrange the rules using precedence.
- The vflow rules take precedence in both IPTables and TCAM are:
- By default, the vflow rules have a precedence value of four (4).
- Implicit drop priority is always lower than the user configured management vflows
- IPTables filter is added such that it precedes the existing system rule.
- The following are the applicable scaling numbers:

- For in-band traffic: the egress TCAM table limitation of 256 entries or as per hardware limits.
- For out-of-band traffic: The IPTables scale limitation is applied.

For example, create a vflow with the following parameters, use the command:

```
CLI (network-admin@Spine1) > vflow-create name <mgmt_flow> if <mgmt|data>
scope <local|fabric> src-ip <IP> src-mask <MASK> dstip <IP> dst-mask <MASK>
proto <num_or_name> src-port <src-port-number> dst-port <dst-port-number>
action <permit|drop> precedence <num>
```

name	Name of the vFlow that you are creating
if	Specify the vflow administrative service as management or data
scope	Specify the scope as local or fabric
src-ip	Specify the source IP Address
src-mask	Specify the source IP address mask
dstip	Specify the destination IP address
dst-mask	Specify the destination IP mask
proto	Specify the name or number of the protocol
src-port	Specify the Layer 3 protocol source port for the vFlow
dst-port	Specify the Layer 3 protocol destination port for the vFlow
action	Specify the action, whether to drop the packet or allow/permit the flow of packet
precedence	Specify the traffic priority value. The default values range between 2 and 15.

To delete a vflow, use the command:

```
CLI (network-admin@spine1) > vflow-delete name <mgmt_flow>
```

To modify the vflow rule, use the command:

```
CLI (network-admin@spine1) > vflow-modify name <mgmt_flow> if <mgmt|data>
src-ip <IP> src-mask <MASK> dstip <IP> dst-mask <MASK>
proto <num_or_name> src-port <num> dst-port <num> action <permit|drop>
precedence <num>
```

To display the configured vflow rules from the IPTables, use the command:

```
CLI (network-admin@spine1) > vflow-mgmt-show name <string>
```

The following example displays an In-band filter configured in Egress Content Aware Processing (ECAP) TCAM on two IPV4 addresses, where the vflow filters are applied to block the ssh connection from the source IP address, 10.10.10.19 whereas the ssh connection is allowed from the IP address, 10.10.10.20:

```
CLI (network-admin@spine1) > switch-local vflow-show
```

name	scope	type	in-port	src-ip	dst-port	precedence	action	enable
fdata	local	vflow	73	10.10.10.20	22	4	none	enable
fdata1	local	vflow	73	10.10.10.19	22	4	drop	enable
tcp_22	local	vflow	73		22	default	drop	enable

To display the examples for out-of-band management filters.

CLI (network-admin@spine1) > vflow-mgmt-show

name	scope	type	src-ip	dst-port	precedence	action	enable
data1	local	iptables	153.1.1.120 /255.255.25 5.255	22	15		enable
implicitv4_ drop_tcp_22 _vmgmt0	local	iptables		22	15	drop	enable
mgmt_ipv4	local	iptables	2.1.1.1		default	none	enable
implicitv4_ drop_icmp_v mgmt0	local	iptables		0	15	drop	enable
mgmt1_ipv6	local	iptables	2000::2/fff f:ffff:ffff :ffff::		default	none	enable
mgmt_ipv6	local	iptables	2000::1/fff f:ffff:ffff :ffff::		default	none	enable
implicitv6_ drop_ipv6- icmp_vmgmt0	local	iptables		0	15	drop	enable

To display the packets and byte count from the IPTables, use the command:

CLI (network-admin@spine1) > vflow-mgmt-stats-show name <string>

CLI (network-admin@spine1) > vflow-mgmt-stats-show

switch	name	pkts	bytes
-----	-----	-----	-----
spine1	data1	0	0
spine1	implicitv4_drop_tcp_2 2_vmgmt0	16	976
spine1	mgmt_ipv4	0	0
spine1	implicitv4_drop_icmp_ vmgmt0	29	2.38K
spine1	mgmt1_ipv6	0	0
spine13	mgmt_ipv6	0	0

```
spine1          implicitv6_drop_ipv6-  0
                icmp_vmgmt0
```

To clear all the IPTable rules, use the command:

```
CLI (network-admin@spine1) > vflow-mgmt-stats-clear name <string>
```

The following example displays an In-band filter configured in Egress Content Aware Processing (ECAP) TCAM on two IPV4 addresses, where the vflow filters are applied to block the ssh connection from the source IP address, 10.10.10.19 whereas the ssh connection is allowed from the IP address, 10.10.10.20:

```
CLI (network-admin@spine1) > switch-local vflow-show
```

name	scope	type	in-port	src-ip	dst-port	preceden ce	action	enable	if
fdata	local	vflow	73	10.10.10 .20	22	4	none	enable	data
fdata1	local	vflow	73	10.10.10 .19	22	4	drop	enable	data
tcp_22	local	vflow	73		22	default	drop	enable	data

## Configuring vFlow for Analytics

---

A vFlow can be used to capture packets for analysis, and you can determine if the vFlow captures packets across the fabric or on a single switch. Packets are captured by forwarding them from the data plane of the switch to the control plane.

A flow that directs packets to the switch CPU can be configured to save packets to a file by enabling the log-packets parameter. The file is written using a libcap compatible format so that programs like TCPdump and Wireshark can be used to read the file. The file is exported to clients using NFS or SFTP.

Packet capture data is available with switch or fabric scope. The pcap files are stored over NFS in the following locations:

- `/net/<ServerSw_Name>/ONVL/global/flow/<Flow_Name>/switch/<Switch_Name>/pcap`
- `/net/<ServerSw_Name>/ONVL//<_Name>/flow/<Flow_Name>/switch/<Switch_Name>/pcap`
- `/net/<ServerSw_Name>/ONVL/global/flow/<Flow_Name>/fabric/pcap`
- `/net/<ServerSw_Name>/ONVL//<_Name>/flow/<Flow_Name>/fabric/pcap`

Snooping only works if you use the parameters, `copy-to-cpu` or `to-cpu`.

The `copy-to-cpu` parameter ensures that the data plane forwards the packets and sends a copy to the CPU. Use this parameter if you want traffic to flow through the switch.

The `to-cpu` parameter doesn't forward packets and interrupts traffic on the switch. To snoop all application flow packets of protocol type TCP, enter the following CLI commands at the prompt:

```
CLI (network-admin@Leaf1) > vflow-create name snoop_all scope local proto tcp action copy-to-cpu
```

Then use the following command to display the output:

```
CLI (network-admin@Leaf1) > vflow-snoop
```

```
switch: pleiades24, flow: snoop_all, port: 65, size: 66, time: 20:07:15.03867188
```

```
smac: 64:0e:94:28:00:fa, dmac: 64:0e:94:2c:00:7a, etype: ip
```

```
sip: 192.168.2.51, dip: 192.168.2.31, proto: tcp
```

```
sport: 42120, dport: 33399
```

```
switch: pleiades24, flow: snoop_all, port: 65, size: 184, time: 20:07:15.03882961
```

```
smac: 64:0e:94:28:00:fa, dmac: 64:0e:94:2c:00:7a, etype: ip
```

```
sip: 192.168.2.51, dip: 192.168.2.31, proto: tcp
```

```
sport: 42120, dport: 33399
```

```
switch: pleiades24, flow: snoop_all, port: 43, size: 66, time:
20:07:15.03893740
smac: 64:0e:94:2c:00:7a, dmac: 64:0e:94:28:00:fa, etype: ip
sip: 192.168.2.31, dip: 192.168.2.51, proto: tcp
sport: 33399, dport: 42120
```

To restrict the flows captured to TCP port 22, SSH traffic, create the following vFlow:

```
CLI (network-admin@Leaf1) > vflow-create name snoop_ssh scope local action
copy-to-cpu src-port 22 proto tcp vflow-add-filter name snoop_ssh
```

Then use the `vflow-snoop` command to display the results:

```
switch: pleiades24, flow: snoop_ssh, port: 41, size: 230, time:
10:56:57.05785917 src-mac: 00:15:17:ea:f8:70, dst-mac: f4:6d:04:0e:77:60,
etype: ip src-ip: 10.9.11.18, dst-ip: 10.9.10.65, proto: tcp src-port: 22,
dst-port: 62356
switch: pleiades24, flow: snoop_ssh, port: 41, size: 118, time:
10:56:57.05922560 src-mac: 00:15:17:ea:f8:70, dst-mac: f4:6d:04:0e:77:60,
etype: ip src-ip: 10.9.11.18, dst-ip: 10.9.10.65, proto: tcp src-port: 22,
dst-port: 62356
```

The optional parameter `vflow-add-filter` restricts the output of the `vflow-snoop` command to the packets matching the `snoop_ssh` flow definition.

To capture traffic packets for a flow across the entire fabric, you create a flow with the scope of fabric. To copy the packets to a pcap file, add the `log-packets` option:

```
CLI (network-admin@Leaf1) > vflow-create name fab_snoop_all scope fabric
action copy-to-cpu port 22 log-packets yes
```

If you enable `log-packets`, the separate pcap files for all switches are available on any switch. In addition a consolidated pcap file is available that aggregates the packets from all switches in the entire fabric.

## Support for IPv6 Addresses and vFlow Configurations

You must modify the vFlow table profile using the new command, `vflow-table-profile-modify`:

```
CLI (network-admin@Leaf1) > vflow-table-profile-modify profile ipv6 hw-tbl
switch-main
```

You must reboot the switch in order for the settings to take effect. To ensure that the profile is available after rebooting, use the `vflow-table-show` command:

```
CLI (network-admin@Leaf1) > vflow-table-show
```

name	flow-max-per-group	flow-used	flow-tbl-slices	capability	flow-profile
-----	-----	-----	-----	-----	-----
Egress-Table-1-0	256	0	2	match-metadata	system



Egress-Table-v6-1-0	256	0	1	none	egress-v6
IPv6-Table-1-0	1536	0	1	none	ipv6
System-L1-L4-Tun-1-0	1536	57	2	set-metadata	system
System-VCAP-table-1-0	512	1	1	none	system

## Configuring Network Security

---

This chapter provides information for understanding and using the Pluribus Networks Netvisor ONE command line interface (CLI) on a Netvisor ONE switch to configure Network Security.

---

- [About Port Isolation](#)
  - [Creating and Implementing Access Control Lists \(ACLs\)](#)
  - [Using and Configuring MAC ACLs](#)
  - [Using and Configuring IP ACLs](#)
  - [Support for DHCP Snooping](#)
  - [Support for Router Advertisement \(RA\) Guard](#)
  - [Configuring Control Plane Traffic Protection \(CPTP\)](#)
  - [Using Wireshark to Analyze Packets in Real Time](#)
  - [Examples and Use cases for QoS](#)
-

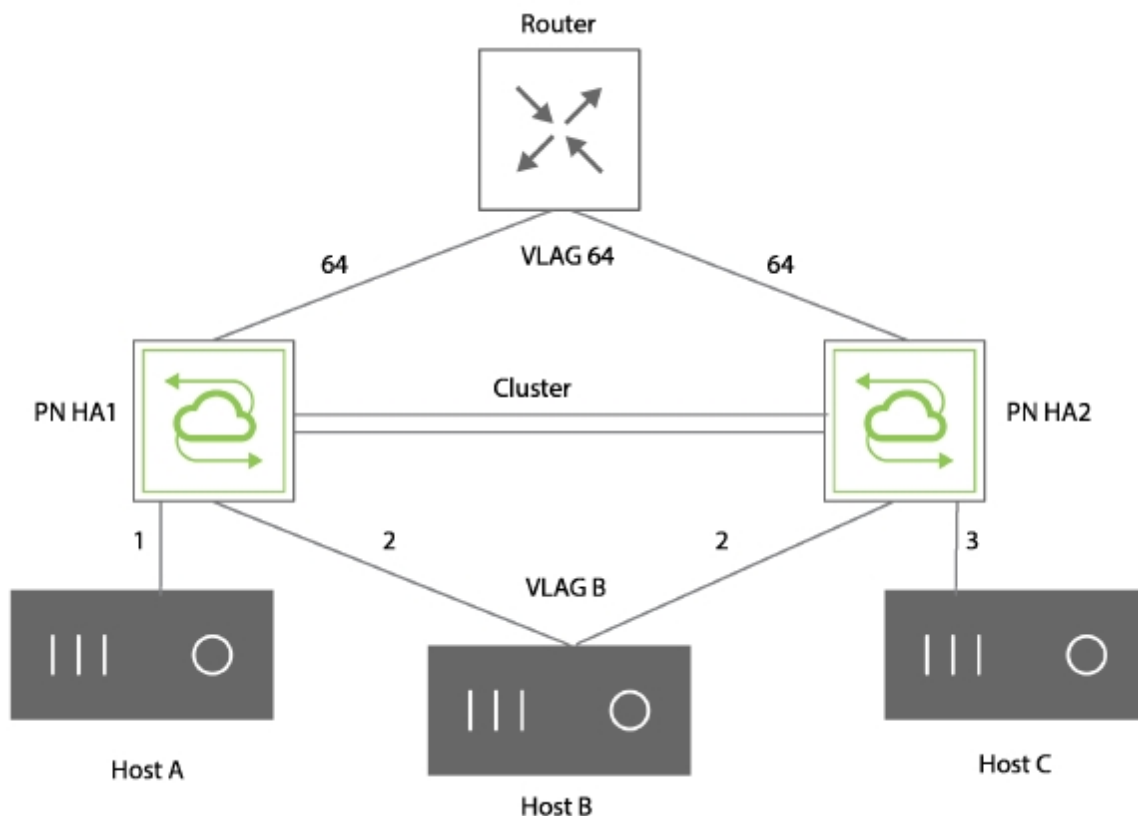
## About Port Isolation

Port Isolation prevents local switching among ports on a Netvisor ONE switch or on a pair of Netvisor ONE switches configured as a cluster. With Port Isolation, Netvisor ONE disables direct communication with hosts part of same Layer 2 domain connected to isolated ports or to mutually learn the other MAC address. Communication between these hosts occurs through a Layer 3 device. Use this feature to secure bridged east-west traffic through a firewall.

When using this feature on ports within a cluster, you must configure the port-link state association rules between the uplink ports and the downlink isolated ports.

### Example Configuration

In a typical scenario, as shown in the **Figure 12-1** below, ports 1, 2, and 3 are configured as isolated ports so that the hosts attached to these ports cannot communicate with each other directly, but only through the upstream firewall or router that is connected to port 64.



**Figure 12-1 - Port Isolation Scenario**

As shown in the figure, create the configuration as follows:

#### PN-HA1

```
CLI (network-admin@Leaf1) > port-config-modify port 1 no-local-switching
```

```
CLI (network-admin@Leaf1) > port-config-modify port 2 no-local switching
```

## **PN-HA2**

```
CLI (network-admin@Leaf1) > port-config-modify port 2 no-local-switching
```

```
CLI (network-admin@Leaf1) > port-config-modify port 3 no-local-switching
```

Typically, you configure the upstream router or firewall to perform local proxy ARPs and/or NDP proxy and respond to all ARP requests and/or Neighbor Solicitations coming from isolated hosts.

To avoid interfering with local proxy ARPs and NDP proxy, disable ARP and ND Optimization as follows:

```
CLI (network-admin@Leaf1) > system-settings-modify no-optimize-arps
```

```
CLI (network-admin@Leaf1) > system-settings-modify no-optimize-nd
```

## Configuring Port Isolation

To configure Port Isolation, use the following steps:

- 1) Configure the isolated ports. In this example, ports 1 and 2:

```
CLI (network-admin@Leaf1) > port-config-modify port 1,2 no-local-switching
```

- 2) Optionally, configure the port link state association. A port association is required to match the link state of downlink isolated ports with the one of uplink ports. When all uplink ports are down, downlink isolated ports are administratively disabled until one of the uplinks becomes operational again. In this example, the port association name is PA, uplink (master), ports value is 64, and isolated downlink (slave) ports value are 1,2.

```
CLI (network-admin@Leaf1) > port-association-create-name PA master-ports 64
slave-ports 1,2 policy any-master
```

- 3) Optionally, disable ARP and ND optimization.

```
CLI (network-admin@Leaf1) > system-settings-modify no-optimize-arps
```

```
CLI (network-admin@Leaf1) > system-settings-modify no-optimize-nd
```

This feature uses the command `no-local-switching` for the `port-config-modify` command. To configure one or more isolated ports:

```
CLI (network-admin@Leaf1) > port-config-modify port port-list no-local-switching
```

To view ports that are impacted by the `no-local-switching` command, use the `port-egress-show` command:

switch	port	egress	rx-only	active- active- vlags	loopback	mir- prevent-out	no-local- switching- out
-----	----	-----	-----	-----	-----	-----	-----
				-		-	--
1	0-72	none	none	none	none	none	none
2	0-72	none	none	none	none	none	none
3	0-72	none	none	none	none	none	none
4	0-72	none	none	none	none	none	none
5	0-4,11-72	none	none	none	none	none	5-10
6	0-4,11-72	none	none	none	none	none	5-10
7	0-4,11-72	none	none	none	none	none	5-10
8	0-4,11-72	none	none	none	none	none	5-10

The following Port Isolation options for the `trunk-create`, `trunk-modify`, and `trunk-`

show commands are as follows:

```
CLI (network-admin@Leaf1) > trunk-create
```

---

trunk-create	Create a trunk configuration for link aggregation
One or more of the following options:	
local-switching no-local-switching	Specify no-local-switching if you do not want the port to bridge traffic to another no-local-switching port.

---

```
CLI (network-admin@Leaf1) > trunk-modify
```

---

trunk-modify	Modify a trunk configuration for link aggregation
One or more of the following options:	
reflect norelect	Specify if physical port reflection is enabled or not.

---

```
CLI (network-admin@Leaf1) > trunk-show
```

---

trunk-show	Display trunk configuration
One or more of the following options:	
reflect norelect	Displays if physical port reflection is enabled or not.

---

## Creating and Implementing Access Control Lists (ACLs)

---

Access Control Lists (ACLs) allow you to configure basic traffic filtering for IP addresses and MAC addresses. The ACL controls if routed packets are forwarded or blocked on the network. The packet is examined by the switch and then determines if the packet is forwarded or dropped based on the criteria configured in the ACLs. nvOS supports Layer 2 (MAC) or Layer 3 (IP) ACLs.

ACL criteria can be based on source or destination addresses or the protocol type. nvOS supports UDP, TCP, IGMP, and IP protocols.

You can use ACLs to restrict contents of routing updates or provide traffic flow control. ACLs can allow one host to access part of your network and prevent another host from accessing the same area. You can also use ACLs to decide what types of traffic are forwarded or blocked.

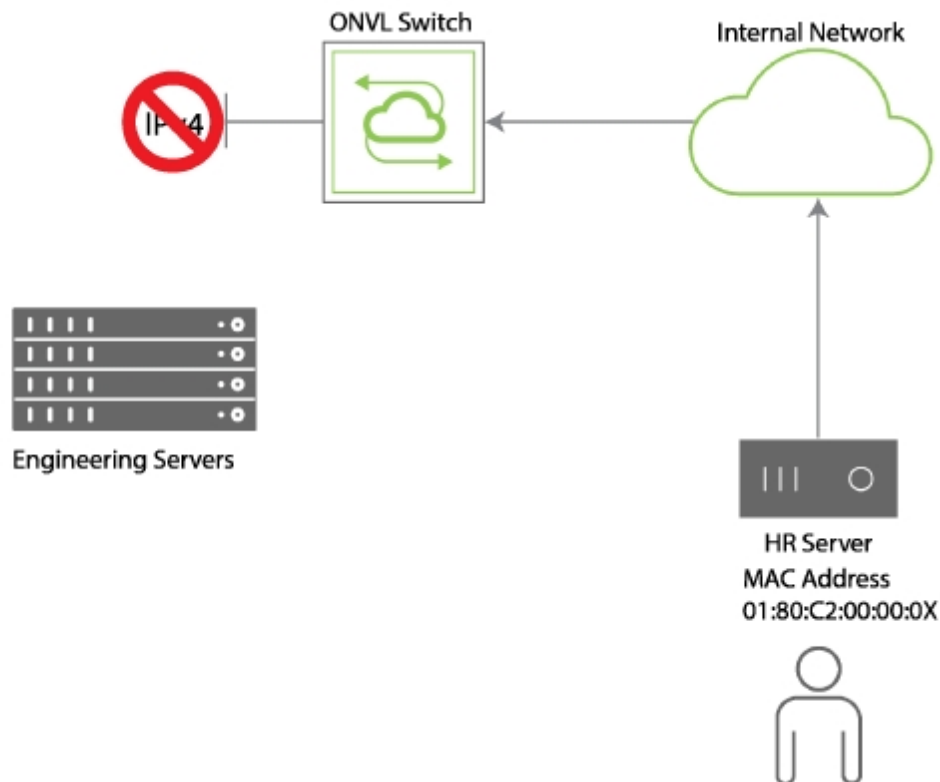
If you need more background on ACLs and using them on your network, refer to the many networking resources available.

## Using and Configuring MAC ACLs

### Using MAC ACLs to Deny Network Traffic

You can create ACLs based on MAC addresses to deny network traffic from a specific source. MAC addresses are Layer 2 protocols and most often assigned by the hardware manufacturer.

The **Figure 12-2** below shows an example of a MAC address and Ethernet type that you want to block from the network.



**Figure 12-2 - MAC ACL Blocking Access**

### Configuring a MAC ACL to Deny Network Traffic

To deny IPv4 network traffic from MAC address, 01:80:c2:00:00:0X, for the scope fabric, create the MAC ACL, deny-MAC, using the following syntax:

```
CLI (network-admin@Leaf1) > acl-mac-create name deny-mac action deny src-
mac 01:80:c2:00:00:0X ether-type ipv4
scope fabric
```

To review the configuration, use the `acl-mac-show` command:

```
CLI (network-admin@Leaf1) >acl-mac-show name deny-mac layout vertical
```



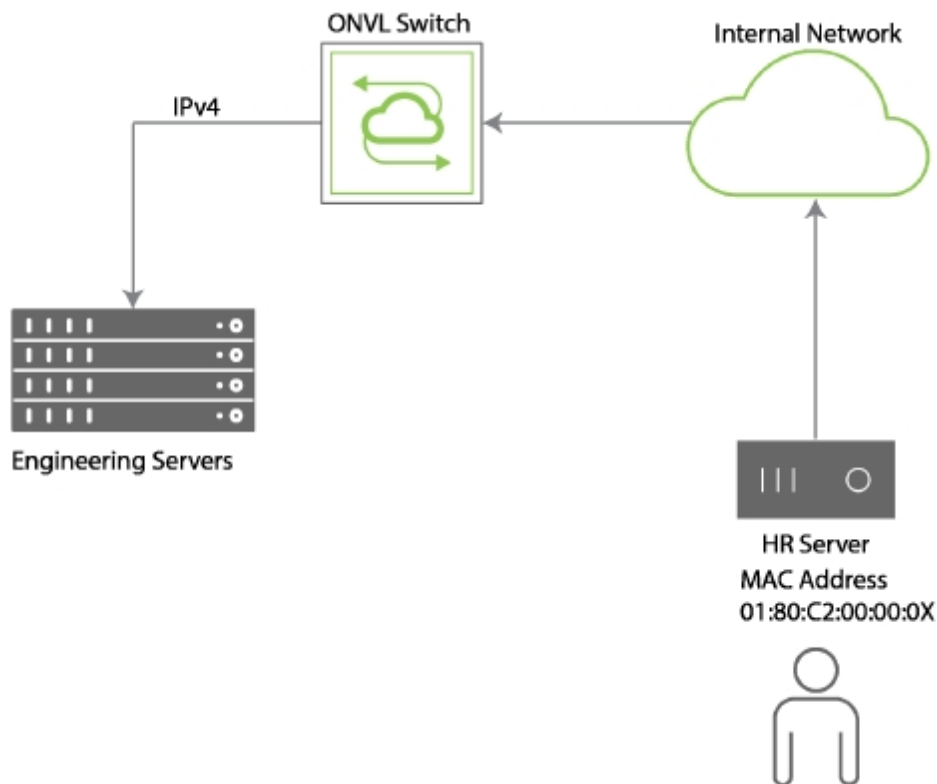
```

name:          deny-mac
id:            b000015:12
action:        deny
src-mac:       01:80:c2:00:00:0X
dst-mac:       00:00:00:00:00:00
dst-mac-mask:  aa:aa:aa:aa:aa:aa
ether-type:    ipv4
vlan:          0
scope:         fabric
port:          0

```

## Using MAC ACLs to Allow Network Traffic

So now that you've blocked the MAC address, let's reverse the scenario and allow IPv4 network traffic from the MAC address to the network.



**Figure 12-3 - MAC ACL Allowing Access**

See Configuring a MAC ACL to Allow Network Traffic to review the example configuration.

## Configuring a MAC ACL to Allow Network Traffic

To allow IPv4 network traffic from MAC address, 01:80:c2:00:00:0X, for the scope fabric, create the MAC ACL, allow-MAC, using the following syntax:

```
CLI (network-admin@Leaf1) > acl-mac-create name allow-mac action permit
```

```
src-mac 01:80:c2:00:00:0X ether-type ipv4 scope fabric
```

To review the configuration, use the `acl-mac-show` command:

```
CLI (network-admin@Leaf1) > acl-mac-show name deny-mac layout vertical
```

```
name:          deny-mac
id:            b000015:12
action:        deny
src-mac:       01:80:c2:00:00:0X
dst-mac:       00:00:00:00:00:00
dst-mac-mask:  aa:aa:aa:aa:aa:aa
ether-type:    ipv4
vlan:          0
scope:         fabric
port:          0
```

To delete the ACL configuration, use the `acl-mac-delete` command.

To modify the ACL configuration, use the `acl-mac-modify` command.

### Configuring a MAC ACL to Deny Network Traffic

To deny IPv4 network traffic from MAC address, 01:80:c2:00:00:0X, for the scope fabric, create the MAC ACL, `deny-MAC`, using the following syntax:

```
CLI (network-admin@Leaf1) > acl-mac-create name deny-mac action deny src-
mac 01:80:c2:00:00:0X ether-type ipv4 scope fabric
```

To review the configuration, use the `acl-mac-show` command:

```
CLI (network-admin@Leaf1) > acl-mac-show name deny-mac layout vertical
```

```
name:          deny-mac
id:            b000015:12
action:        deny
src-mac:       01:80:c2:00:00:0X
dst-mac:       00:00:00:00:00:00
dst-mac-mask:  aa:aa:aa:aa:aa:aa
ether-type:    ipv4
vlan:          0
scope:         fabric
port:          0
```

## Using and Configuring IP ACLs

---

### Configuring IP ACLs

From **Figure 11-4** [Network Example - IP ACL for Internal Servers](#), the following information is available:

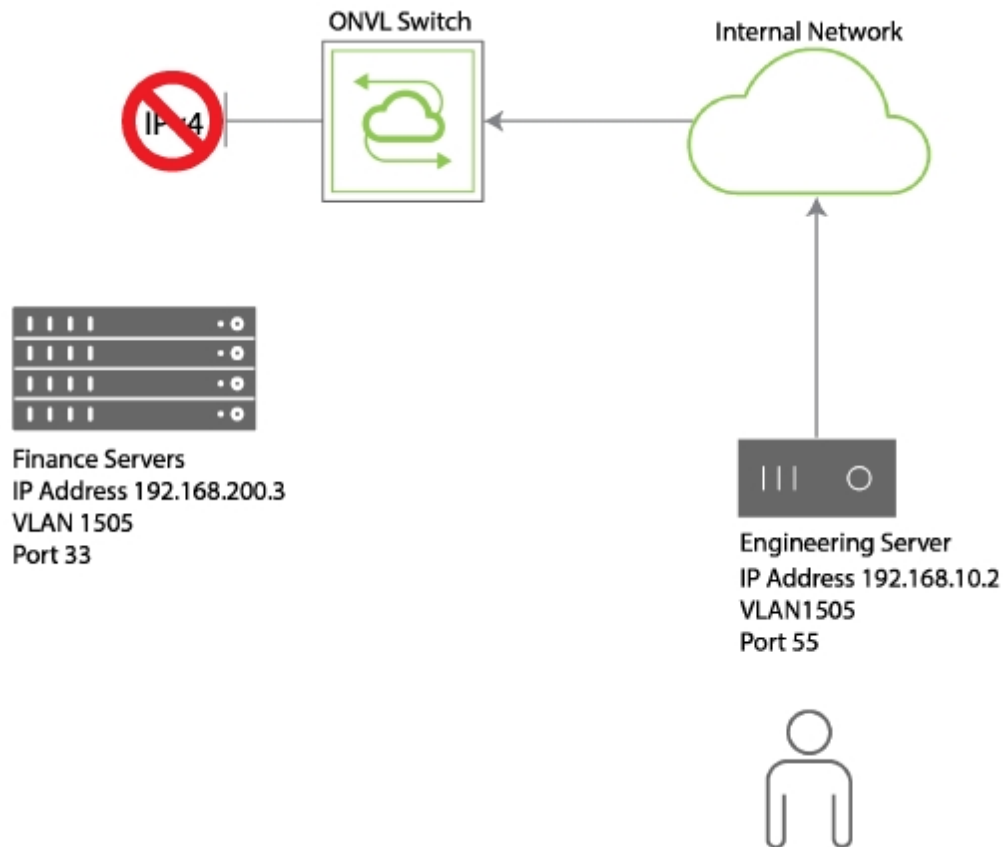
- Source IP address
- Source netmask
- Destination IP address
- Destination netmask
- Type of protocol to deny - IP
- Ports
- VLAN

### Using a Deny IP ACL to Block Network Traffic

In this example, a network is shown with a Finance server on one part of the network, and an Engineering server on another part.

You want to block the Engineering server from the Finance server in order to protect company sensitive information.

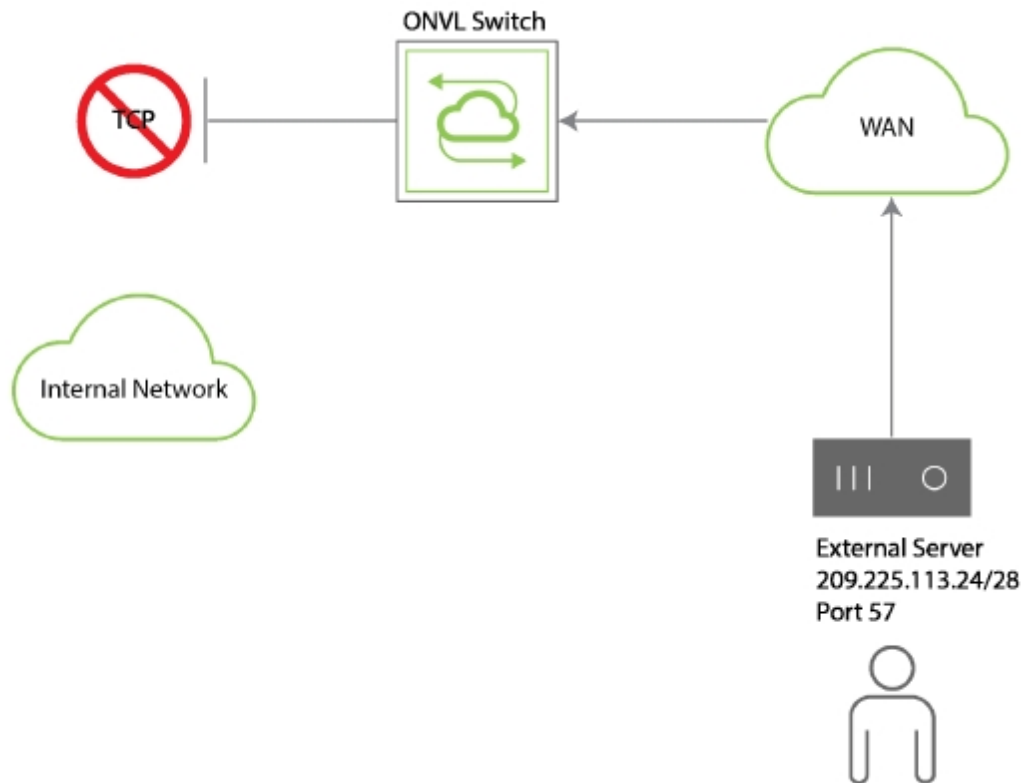
See [Configuring an Internal Deny ACL](#) to review the configuration sample.



**Figure 12-4 - Network Example - IP ACL for Internal Servers**

Or, you may discover that an external source is attempting to access your network, and ping your servers for IP addresses.

You can use an ACL to block the specific source using an IP ACL.



**Figure 12-5 - IP ACL Blocking External Access**

See [Configuring an External Deny ACL](#) to review the configuration example.

## Using IP ACLs to Allow Network Traffic

In the same manner, you can allow specific traffic to a destination such as the external server as shown in Figure 2 - IP ACL Blocking External Access.

To allow HTTP traffic to 209.225.113.24, see [Configuring an External Allow IP ACL](#) to review the configuration example.

## Configuring an Internal Deny ACL

Let's configure the ACL for denying traffic from the Engineering server to the Finance server and name the ACL, deny-finance:

```
CLI (network-admin@Leaf1) > acl-ip-create name deny-finance action deny
scope local src-ip 192.168.10.2 src-ip-mask 24 dst-ip 192.168.200.3 dst-ip-
netmask 24 proto ip src-port 55 dst-port 33 vlan 1505
```

To review the configuration, use the `acl-ip-show` command:

```
CLI (network-admin@Leaf1) > acl-ip-show name deny-hr layout vertical
```

```
name:          deny-ip
id:            b00011:20
```

```
action:          deny
proto:           ip
src-ip:          192.168.10.2/24
src-port:        55
dst-ip:          192.168.200.3/24
dst-port:        33
vlan:            1505
scope:           local
port:            0
```

Now, when you attempt to access the Finance server from the Engineering server, the packets are dropped.

## Configuring an External Deny ACL

From Figure 2 IP ACL Blocking External Access, you can see the following information:

- IP Address
- Port Number

To configure an ACL to deny traffic from the external server, use the `acl-ip-create` command to create an ACL named `deny-external`:

```
CLI (network-admin@Leaf1) > acl-ip-create name deny-external scope fabric
src-ip 209.255.113.24/28
```

To review the configuration, use the `acl-ip-show` command:

```
CLI (network-admin@Leaf1) > acl-ip-show name deny-external layout vertical
```

```
name:            deny-external
id:              b000022:20
action:          deny
proto:           tcp
src-ip:          209.225.113.24/28
src-port:        0
dst-ip:          ::/0
dst-port:        0
vlan:            0
scope:           fabric
port:            0
```

## Configuring an External Allow IP ACL

To allow HTTP traffic to the external server, 209.225.113.24 with a netmask of 255.255.255.240 and a scope of `fabric`, you can create an IP ACL called `allow-http` using the following syntax:

```
CLI (network-admin@Leaf1) > acl-ip-create name allow-http permit scope
```

```
fabric src-ip 0.0.0.0. src-ip-mask 255.255.255.255 dst-ip 209.225.113.24  
dst-ip-mask 255.255.255.240 protocol tcp dst-port 57
```

To review the configuration, use the `acl-ip-show` command:

```
CLI (network-admin@Leaf1) > >acl-ip-show name allow-http layout vertical
```

```
name:                allow-http  
id:                  b000025:20  
action:              allow  
proto:               tcp  
src-ip:               0.0.0.0/255.255.255.255  
src-port:             0  
dst-ip:               209.225.113.24/28  
dst-port:             57  
vlan:                 0  
scope:                fabric  
port:                 0
```

To delete the ACL configuration, use the `acl-ip-delete` command.

To modify the ACL configuration, use the `acl-ip-modify` command.

## Support for DHCP Snooping

Netvisor ONE supports DHCP snooping as a security feature allowing the network to avoid denial-of-service (DoS) attacks from rogue DHCP servers. You define trusted ports to connect to the known DHCP servers. DHCP snooping also maintains a mapping table for current assignments.

In a DHCP packet flow, there are the following packet types:

- DHCPDISCOVER/DHCPREQUEST — Packets from the DHCP client to server (UDP dest-port = 67)
- DHCPOFFER/DHCPACK — Packets from the DHCP Server to client (UDP dest-port = 68)

Netvisor One must snoop the DHCP packets in order to leverage this feature, and achieves this by installing a copy-to-cpu vFlow with the parameter, bw-max, to set packet rate limits.

- DHCP-client-vflow — Packets with UDP dest-port=67, copy-to-cpu
- DHCP-server-vflow — Packets with UDP dest-port=68, copy-to-cpu

A trusted port is a port receiving the DHCP server messages from a trusted DHCP server. Any DHCP server message, such as OFFER/ACKNOWLEDGE, received from trusted ports are valid. Ports not specifically configured as trusted are untrusted ports.

Netvisor One drops any DHCP server message received from an untrusted port, and ensures that a rogue DHCP server cannot assign IP addresses to devices on your network.

Enable DHCP snooping and specify the list of trusted server ports using the following set of commands:

```
CLI (network-admin@Spine1) > dhcp-filter-create name name-string trusted-ports port-list
```

<i>name name-string</i>	Specify a name for the filter.
<i>trusted-ports port-list</i>	Specify a list of trusted ports.

```
CLI (network-admin@Spine1) > dhcp-filter-modify name name-string trusted-ports port-list
```

<i>name name-string</i>	Specify the name of the filter to modify.
<i>trusted-ports port-list</i>	Specify a list of trusted ports.

```
CLI network-admin@Spine1) > dhcp-filter-delete name name-string
```

<i>name name-string</i>	Specify the name of the filter to delete.
-------------------------	---

```
CLI (network-admin@Spine1) > dhcp-filter-show name name-string trusted-
```



```
ports port-list vlan vlan-list
```

---

<code>name name-string</code>	Displays the name of the filter.
<code>trusted-ports port-list</code>	Specify a list of trusted ports.
<code>vlan vlan-list</code>	Displays a list of VLANs.

---

In order to drop the packets from rogue DHCP servers, connected through untrusted ports, Netvisor One has a new system vFlow, DHCP-LOG-DROP.

The vFlow sends the packets to the CPU, to track the untrusted server messages, and then drop the untrusted DHCP server packets. This is set to a higher precedence than the DHCP trusted ports vFlow. The vFlow includes the untrusted port list for the ingress port.

Untrusted ports typically connect to hosts where DHCP clients can send messages, and Netvisor One ensures the DHCP messages are rate limited using `dhcp` CPU class.

All the DHCP messages use the `dhcp` CPU class. The existing command for `cpu-class-modify` is used:

```
CLI (network-admin@Spine1) > cpu-class-modify name dhcp rate-limit rate-limit-number
```

The show output for the command, `dhcp-lease-show`, has two new parameters to display trusted and rogue DHCP servers:

```
CLI (network-admin@Spine1) > dhcp-lease-show trusted-server|no-trusted-server
```

```
CLI (network-admin@Spine1) > dhcp-lease-show
```

```

ip                mac                port vnet vlan db-state  server server-ip  server-port trusted-server
last-msg
-----
6053:23a7:0:0:200:: 00:12:c0:80:1f:b8  9          1    unknown  10.1.1.100 65          no          offer

```

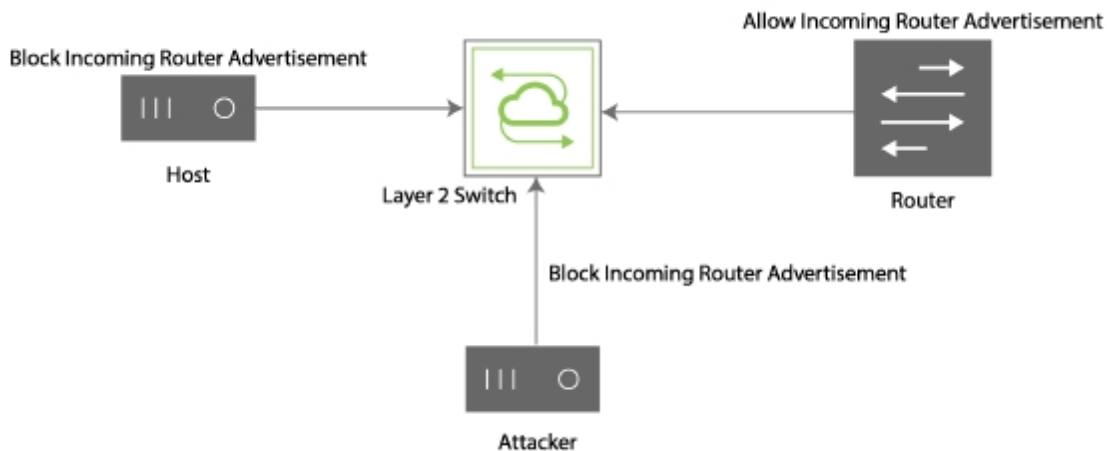
Log messages indicate the presence of an unknown or rogue DHCP servers:

```
DHCP server message received from untrusted port=<x> server-ip=<ip-addr>
```

## Support for Router Advertisement (RA) Guard

The IPv6 RA Guard feature provides support for allowing the network administrator to block or reject unwanted or rogue RA guard messages arriving at the network device platform. RAs are used by devices to announce themselves on the link. The IPv6 RA Guard feature analyzes these RAs and filters out RAs sent by unauthorized devices. In host mode, all RA and router redirect messages are not allowed on the port. The RA Guard feature compares configuration information on the Layer 2 (L2) device with the information found in the received RA frame. Once the L2 device has validated the content of the RA frame and router redirect frame against the configuration, it forwards the RA to the unicast or multicast destination. If the RA frame content is not validated, the RA is dropped.

**Note:** Internal ports and cluster ports are not blacklisted.



**Figure 12-6 - Route Advertisement (RA) Configuration**

In **Figure 12-6**, the Layer 2 device receives a RA from the router and floods the RA on the ports. The attacker host, attempting to gain control over the network, sends a misleading RA with different prefixes, link-local or global IP addresses. The host assumes the attacker to be the router, based on priority or arrival order. When you configure RA Guard, you can disallow any RAs sent from ports connected to host ports using RA policies. The RA sent by the router, the source IP address, from the port and prefixes are whitelisted by the policies defined by the configuration.

To configure the RA Guard feature, follow these steps:

- 1) Create an access list using the command, `access-list-create`.
- 2) Create a prefix list using the command, `prefix-list-create`.
- 3) Create the IPv6 security profile using the command, `ipv6security-raguard-create`.

This creates two vFlows for RA Guard:

- One vFlow drops RAs sent by devices with the role `host` as assigned using the `ipv6security-raguard-create` command.
- The second vFlow sends RAs to the CPU on qualified ports or VLANs with the action, `to-cpu`, and the device role as `router`.

The RAs are received and examined and the necessary action is taken based on the access and prefix lists or port and VLAN policies. The RA is now accepted and flooded back to all ports.

There are new commands to support this feature:

```
CLI (network-admin@Spine1) > access-list-create
```

<code>name <i>name-string</i></code>	Specify a name for the access list.
<code>scope <i>scope</i></code>	Specify if the scope is local or fabric.

```
CLI (network-admin@Spine1) > access-list-delete name name-string
```

<code>name <i>name-string</i></code>	Specify the name for the access list to delete.
<code>scope <i>scope</i></code>	Specify if the scope is local or fabric.

```
CLI (network-admin@Spine1) > access-list-show
```

```
switch  name scope
-----  ---  -----
spine1  test  local
```

```
CLI (network-admin@Spine1) > access-list-ip-add
```

<code>name <i>name-string</i></code>	Specify a name for the access list.
<code>ip <i>ip-address</i></code>	Specify the IP address for the access list.

```
CLI (network-admin@Spine1) > access-list-ip-delete name name-string ip ip-address
```

```
CLI (network-admin@Spine1) > access-list-ip-show
```

```
switch  name ip
```

```
-----
spinel  test 1.1.1.4
```

```
CLI (network-admin@Spinel) > prefix-list-create
```

---

<code>name <i>name-string</i></code>	Specify a name for the prefix list.
<code>scope <i>scope</i></code>	Specify if the scope is local or fabric.

---

```
CLI (network-admin@Spinel) > prefix-list-delete name name-string
```

```
CLI (network-admin@Spinel) > prefix-list-show
```

---

<code>name <i>name-string</i></code>	Displays the name for the prefix list.
<code>scope <i>scope</i></code>	Displays if the scope is local or fabric.

---

```
CLI (network-admin@Spinel) > prefix-list-network-add
```

---

<code>name <i>name-string</i></code>	Specify the name for the prefix network list.
<code>network <i>ip-address</i></code>	Specify the IP address for the network.
<code>netmask <i>netmask</i></code>	Specify the netmask.

---

```
CLI (network-admin@Spinel) > prefix-list-network-delete name name-string
```

```
CLI (network-admin@Spinel) > prefix-list-network-show
```

---

<code>name <i>name-string</i></code>	Displays the name for the prefix network list.
<code>network <i>ip-address</i></code>	Displays the IP address for the network.
<code>netmask <i>netmask</i></code>	Displays the netmask.

---

```
CLI (network-admin@Spinel) > ipv6security-raguard-create
```

---

<code>name <i>name-string</i></code>	Specify the RA policy name.
--------------------------------------	-----------------------------

---

---

<code>device host router</code>	Specify the type of device as host or router.
<code>router-priority low medium high</code>	Specify the router priority as low, medium, or high.
<code>access-list <i>name-string</i></code>	Specify the access list name.
<code>prefix-list <i>name-string</i></code>	Specify the prefix list name.

---

```
CLI (network-admin@Spine1) > ipv6security-raguard-delete
```

---

<code>name <i>name-string</i></code>	Specify the RA policy name.
--------------------------------------	-----------------------------

---

```
CLI (network-admin@Spine1) > ipv6security-raguard-modify
```

---

<code>name <i>name-string</i></code>	Specify the RA policy name.
<code>device host router</code>	Specify the type of device as host or router.
<code>router-priority low medium high</code>	Specify the router priority as low, medium, or high.
<code>access-list <i>name-string</i></code>	Specify the access list name.
<code>prefix-list <i>name-string</i></code>	Specify the prefix list name.

---

```
CLI (network-admin@Spine1) > ipv6security-raguard-show
```

---

<code>name <i>name-string</i></code>	Displays the RA policy name.
<code>device host router</code>	Displays the type of device as host or router.
<code>router-priority low medium high</code>	Displays the router priority as low, medium, or high.
<code>access-list <i>name-string</i></code>	Displays the access list name.
<code>prefix-list <i>name-string</i></code>	Displays the prefix list name.

---

```
CLI (network-admin@Spine1) > ipv6security-raguard-port-add
```

---

<code>name <i>name-string</i></code>	Specify the name of the RA Guard policy to add ports.
<code>ports <i>port-list</i></code>	Specify the list of ports to add to the policy.

---

```
CLI (network-admin@Spine1) > ipv6security-raguard-port-remove
```

---

<code>name <i>name-string</i></code>	Specify the name of the RA Guard policy to remove ports.
<code>ports <i>port-list</i></code>	Specify the list of ports to remove from the policy.

---

```
CLI (network-admin@Spine1) > ipv6security-raguard-port-show
```

---

<code>name <i>name-string</i></code>	Displays the name of the RA Guard policy.
<code>ports <i>port-list</i></code>	Displays the list of ports.

---

```
CLI (network-admin@Spine1) > ipv6security-raguard-vlan-add
```

---

<code>name <i>name-string</i></code>	Specify the name of the RA Guard policy to add VLANs.
<code>vlan <i>vlan-id</i></code>	Specify the VLANs to add to the policy.

---

```
CLI (network-admin@Spine1) > ipv6security-raguard-vlan-remove
```

---

<code>name <i>name-string</i></code>	Specify the name of the RA Guard policy to remove VLANs.
<code>vlan <i>vlan-id</i></code>	Specify the VLANs to remove from the policy.

---

```
CLI (network-admin@Spine1) > ipv6security-raguard-vlan-show
```

---

<code>name <i>name-string</i></code>	Displays the name of the RA Guard policy to add VLANs.
<code>vlan <i>vlan-id</i></code>	Displays the VLANs to add to the policy.

---

## Configuring Control Plane Traffic Protection (CPTP)

---

### About the Importance of Hardening of Network Infrastructure

The network infrastructure is often a prime target for malicious attacks because of the possibility of inflicting the most amount of damage to as many devices as possible (in the worst case scenario, to bring down the entire network and, with it, all the attached devices).

Other reasons to target the network may be to attempt to redirect (for example, through unintended flooding) and snoop traffic to learn about clear-text information and find out possible other weaknesses that can lead to further malicious actions.

Last but not least, network instability sometimes caused by mis-configurations or failures may result in an excessive amount of traffic that can put a strain on the network infrastructure and further exacerbate its stability problems.

Therefore, first and foremost, it is of foundational importance to apply robust control to the traffic that reaches the network devices and to implement appropriate protections against any potentially disruptive traffic.

In particular, this section focuses specifically on the hardware-based protection of the network *control plane*. Other complementary mechanisms, such as hardware-based security and QoS policies described in detail in other topics can be applied to the data plane too for comprehensive network hardening.

### About Control Plane Traffic Protection

In any network device there exists a management entity (typically a CPU) that is in charge of communication exchanges with other networking devices as well as of interactions with a portion of the traffic coming from the rest of the network (the so-called *data plane*). In general, all the traffic that is natively directed or purposefully redirected to such management processor is commonly referred to as *control plane traffic*.

When the amount of any class or classes of traffic belonging to the control plane becomes abnormal--e.g., due to a Denial of Service (DoS) attack attempt--then the network device needs to take some containment action.

Hardware-based queuing and rate limiting are two common techniques employed to implement CPU protections, with *different* levels of granularity and control depending on the switch model's hardware capabilities.

### About Port-based Control Plane Protection

Certain switch models use an internal Ethernet port to transport a portion of the control plane traffic to the management CPU. This special type of interface is referred to as a *rear-facing interface*. The list of models with a rear facing interface includes the following platforms:

- **Edgecore:** AS7716-32X, AS7316-54XS, AS7712-32X, AS7312-54XS, AS7716-32X
- **Pluribus switches:** F9532-C, F9572-V, F9532C-XL-R
- **Ericsson:** NSU, NRU01, NRU02, NRU03

For this case 8 queues are available for control plane traffic segregation and rate-limiting, which Netvisor ONE leverages to protect the CPU from anomalous traffic.

By default, mission-critical control plane traffic is split across 7 weighted queues based on common network management requirements.

In addition, queue 0 is the default 'catch-all' queue that corresponds to all the control plane traffic not specifically segregated into one of the other seven queues.

Any of the eight CPU queues is configured with a default maximum transmission rate suitable to protect the control plane from overloading. If needed, the default rate values may be modified by the network administrator to match specific design requirements.

## About Advanced Control Plane Protection

Pluribus Networks has implemented support for *Advanced Control Plane Traffic Protection (CPTP) with Auto-Quarantine*. This feature is supported on the CPU *inband* interface of the Freedom and Edgecore data center platforms, as well as of Dell's Open Networking Switches.

This very granular capability allows the control plane's processing path to be protected against both misbehaving and malicious devices (compromised end-points, rogue network nodes, etc.) that may start pumping an abnormal amount of control plane traffic. In Pluribus parlance, this is also referred to as *CPU hog protection*.

Advanced CPTP operates over 43 independent queues (from 0 to 42) in order to be able to provide separation and granular control over different types of control plane traffic classes. It can be enabled with the `system-settings-modify cpu-class-enable` command.

Note: The current default setting is `no-cpu-class-enable`. To change it to `cpu-class-enable` requires a subsequent *system reboot* for the setting to take effect.

Note: CPTP on the inband interface performs CPU traffic classification and queuing in hardware, therefore there is no performance penalty in enabling this feature. CPTP queuing supports round-robin scheduling and rate limiting of individual CPU traffic classes, in addition to guaranteeing minimum buffer space allocation to each class.

CPU resources are protected by segregating into separate queues the following types of traffic by default: various standard network control packets, cluster communication messages, fabric updates as well as regular flooded traffic, packets required for MAC learning and copy-to-cpu packets, analytics, etc.

In addition, custom traffic classes can be added by configuring user-defined CPU policies (for example, for troubleshooting purposes).



**Note:** Traffic flows that end up sharing a user-defined CPU queue will compete with each other for bandwidth. It is therefore recommended to configure queue-sharing only for traffic that does not constantly compete for CPU time under identical circumstances. Whenever possible, competing classes should be assigned to different queues.

## About Auto-Quarantine

Advanced CPTP can be very granular with its innovative *auto-quarantine* (a.k.a. CPU hog protection) mechanism. As a user, you can enable the CPU hog protection capability (through the `cpu-class-modify` command) for the following protocols (in Pluribus' parlance also called *CPU classes*): OSPF, BGP, BFD, LACP, STP, ARP, VRRP, and LLDP.

When auto-quarantine is enabled, the Netvisor ONE software monitors control plane packets arriving at the CPU on a per-source-device basis. Traffic from a source device that is deemed to be consuming too much bandwidth (as per user-configurable `rate-limit` value) is redirected to a dedicated per-protocol `quarantine` queue by installing a hardware policy entry.

At the same time a syslog alert is displayed and the offending source device's subsequent activity is monitored. Quarantine state is left automatically only when the traffic activity returns below acceptable limits for a pre-configured `timeout` time; then a corresponding syslog is displayed. You can view messages using the `log-event-show event-type system` command.

Only certain system-defined protocol queues support hog protection. For these CPU classes the user can choose to enable the CPU hog protection capability, or to select the `enable-and-drop` option. In the latter case, all traffic from the quarantined source with the assigned protocol is dropped during ingress.

Since hardware resources are limited, it is also possible to specify a threshold for the maximum number of acceptable CPU hog violators per port. When reached, such threshold causes that class's auto-quarantine hardware policy to become per switch-port (i.e., less granular) instead of being per-port per-offender.

## Configuring Port-based Control Plane Traffic Protection

Certain switch models make use of an internal *rear-facing interface* for CPU communication in addition to a special `control-port`. Other models use the `control-port` only.

For all these cases, by default 8 queues are available for control plane traffic segregation and rate-limiting on a per internal port basis. The corresponding eight default packet rates (pps) can be displayed with the following command:

```
CLI (network-admin@switch) > port-cos-rate-setting-show layout-vertical
```

```
switch:          switch
port:            control-port
ports:           0
cos0-rate(pps):  5000
cos1-rate(pps):  5000
cos2-rate(pps):  5000
cos3-rate(pps):  5000
cos4-rate(pps):  5000
cos5-rate(pps):  5000
cos6-rate(pps):  5000
cos7-rate(pps):  5000
```

Internal (rear facing) data and span ports can be present in a system to carry control plane traffic to the CPU, each using 8 separate queues and rates, as shown below in condensed form:

```
CLI (network-admin@switch) > port-cos-rate-setting-show
```

port	ports	cos0-rate(pps)	cos1-rate(pps)	cos2-rate(pps)	...	cos6-rate(pps)	cos7-rate(pps)
control-port	0	100000	100000	100000	...	100000	100000
data-port	117	100000	100000	100000	...	100000	100000
span-ports	118	100000	100000	100000	...	100000	100000

It is possible to modify the default rate settings in packets per second using the `port-cos-rate-setting-modify` command:

```
CLI (network-admin@switch) > port-cos-rate-setting-modify ?
```

<code>port-cos-rate-settings-modify</code>	Update the port cos rate limit
<code>port control-port   data-port   span-ports</code>	port
Specify at least one of the following options	
<code>cos0-rate unlimited   0..10000000</code>	cos0 rate limit (pps)
<code>cos1-rate unlimited   0..10000000</code>	cos1 rate limit (pps)
<code>cos2-rate unlimited   0..10000000</code>	cos2 rate limit (pps)

cos3-rate unlimited 0..10000000	cos3 rate limit (pps)
cos4-rate unlimited 0..10000000	cos4 rate limit (pps)
cos5-rate unlimited 0..10000000	cos5 rate limit (pps)
cos6-rate unlimited 0..10000000	cos6 rate limit (pps)
cos7-rate unlimited 0..10000000	cos7 rate limit (pps)

In addition, to show the per-queue *traffic* statistics you can issue the following command:

```
CLI (network-admin@switch) > port-cos-stats-show port 0 layout vertical
```

```
switch:      switch
time:        11:59:15
port:        0
cos0-out:    58.8M
cos0-drops:  180M
cos1-out:    58.8M
cos1-drops:  185M
cos2-out:    0
cos2-drops:  0
cos3-out:    0
cos3-drops:  0
cos4-out:    0
cos4-drops:  0
cos5-out:    0
cos5-drops:  0
cos6-out:    65.5M
cos6-drops:  1.06G
cos7-out:    483K
cos7-drops:  493M
```

To clear the queue statistics on the internal ports, use the `port-cos-stats-clear` command.

## Configuring Advanced Control Plane Traffic Protection

To configure this feature, you must first enable it using the `system-settings-modify` command. The command syntax is:

```
CLI (network-admin@switch) > system-settings-modify cpu-class-enable|no-cpu-class-enable
```

After you enable Advanced Control Plane Traffic Protection (with the `cpu-class-enable` option), Netvisor ONE prompts you to restart the switch.

**Note:** The alternative 8-queue mode described in the previous section is applied to the main control plane communication channel when `system-settings-modify` is set to `no-cpu-class-enable`.

To show the pre-configured Advanced Control Plane Traffic Protection classes, you can use the `cpu-class-show` command:

```
CLI (network-admin@switch) > cpu-class-show format all count-output
```

name	scope	rate-limit	hog-protect	hog-protect-support	queue
-----	-----	-----	-----	-----	-----
dmac-miss	local	1000	disable	none	1
smac-miss	local	1000	disable	none	2
l3-miss	local	1000	disable	none	3
ttl1	local	1000	disable	none	4
stp	local	1000	disable	supported	5
lacp	local	1000	disable	supported	6
system-d	local	1000	disable	none	7
igmp	local	1000	disable	supported	8
bcast	local	1000	disable	none	9
icmpv6	local	1000	disable	supported	10
tcp-analytics	local	1000	disable	none	11
kpalv	local	1000	disable	none	12
ecp	local	1000	disable	none	13
arp	local	3000	disable	supported	14
lldp	local	1000	disable	supported	15
dhcp	local	1000	disable	none	16
pim	local	1000	disable	supported	17
local-subnet	local	1000	disable	supported	18
bgp	local	1000	disable	supported	19
ospf	local	1000	disable	supported	20
bfd	local	1000	disable	supported	21
vrrp	local	1000	disable	supported	22
control	local	3000	disable	none	23
dhcp-log-drop	local	1000	disable	none	24
http-rest	local	3000	disable	none	25
vport-messages	local	1000	disable	supported	26
hog-arp	local	100	disable	none	27
hog-ospf	local	100	disable	none	28
hog-bgp	local	100	disable	none	29

hog-bfd	local	100	disable	none	30
hog-lacp	local	100	disable	none	31
hog-stp	local	100	disable	none	32
hog-vrrp	local	100	disable	none	33
hog-lldp	local	100	disable	none	34
hog-local-subnet	local	100	disable	none	35
hog-igmp	local	100	disable	none	36
hog-pim	local	100	disable	none	37
hog-icmpv6	local	100	disable	none	38
hog-vport-messages	local	100	disable	none	39
Count: 39					

This command shows the different categories of control plane traffic that get protected by this feature (for example, `smac-miss` and `dmac-miss` for MAC address learning as part of the vPort database entry creation; or `stp`, `lacp`, and `lldp` for the Layer 2 protocol classes, etc.). It also shows the respective default `rate-limit` values (in packets per second), the queue numbers (0-42, where some queue numbers are *unused by default*) and also whether or not each class supports *auto-quarantine* (`hog-protect-support`).

Auto-quarantine queues are labeled with a special name *hog-<class name>*, such as: `hog-arp`, `hog-ospf`, `hog-bgp`, `hog-bfd`, `hog-lacp`, `hog-stp`, `hog-vrrp`, `hog-lldp`, `hog-local-subnet`, `hog-igmp`, `hog-pim`, `hog-icmpv6`.

**Note:** Starting from Netvisor ONE release 5.1.0 two new queues, one for CPU-bound REST API traffic (TCP port 80 and 443) and another for vPort database-related messages (UDP port 23398), are added with the names: `http-rest` and `vport-messages`. The default `rate-limit` values are set to 3000 pps and 1000 pps respectively. An auto-quarantine queue is added for the latter: `hog-vport-messages`.

Furthermore, starting from Netvisor ONE release 5.1.0 the default `rate-limit` values for `arp` and `control` have been conservatively lowered to 3000. When upgrading to this release, existing user configuration changes will be honored; however, in the absence of user modified values, the old default values will be replaced with the new more conservative ones.

**Note:** The total number of CPU classes available for CPTP is limited by the hardware. In case of conflict, system-created CPU classes are prioritized over user-defined ones at bootup. Given that, if all available classes are used up, some user-defined classes will not persist across an upgrade if more system classes are added in the new release. In such cases, users should account for any (potential) CPTP system class differences between releases while planning an upgrade.

Settings of pre-configured system classes (except the catch-all class 0) can be modified with the following command:

```
CLI (network-admin@switch) > cpu-class-modify
```

<code>cpu-class-modify</code>	Modify a CPU class.
<code>name name-string</code>	Specify the name of the CPU class.
Specify one of more of the following options	
<code>rate-limit rate-limit-number</code>	Specify the cap for the rate limit.

---

<code>hog-protect</code>	<code>disable</code>   <code>enable</code>   <code>enable-</code>
	<code>and-drop</code>

---

Specify if you want to enable, enable and drop packets, or disable hog protection.

The Class 0 rate instead can be configured using the following command:

```
CLI (network-admin@switch) > port-cos-rate-setting-modify port control-port  
cos0-rate rate
```

**Note:** Starting from Netvisor ONE release 5.1.0 the default cos0-rate value is set to 3000 pps automatically when Advanced CPTP is enabled.

## Configuring User-Defined Traffic Classes

These commands are available to configure custom CPU classes that are added to the default list (shown in previous section) in order to address special user requirements:

To add a CPU class, use the command:

```
CLI (network-admin@switch) > cpu-class-create
```

name	Specify a name for the CPU class.
scope local fabric	Specify the scope as local or fabric.
rate-limit rate-limit-number	Specify the cap for the rate limit.
hog-protect disable enable enable-and-drop	Specify if you want to enable, enable and drop packets, or disable hog protection.

The `cpu-class-create` command can be used to allocate new CPU traffic queues for special cases that are identified by user-configurable policies.

For instance, when the culprit of a high CPU utilization issue is being investigated in connection with an errant traffic flow (e.g., FTP), the Pluribus TAC team may recommend configuring a user-defined traffic class with an associated policy to protect the CPU.

In this example a new class `TAC-class` is created and associated to a rate of 100 pps and automatically allocated to free queue 40:

```
CLI (network-admin@switch) > cpu-class-create name TAC-class scope local
rate-limit 100
```

```
CLI (network-admin@switch) > cpu-class-show name TAC-class
```

name	scope	rate-limit	hog-protect	hog-protect-support	queue
TAC-class	local	100	disable	none	40

TAC also requests to create/modify a vFlow policy to direct FTP packets to the new CPU class (`TAC-class`) queue. vFlow match criteria can be freely chosen to match the specific rate limiting need of the user provided they include a `to-cpu` or `copy-to-cpu` action for packets:

```
CLI (network-admin@switch) > vflow-create name myflow scope local in-port
10 proto ftp action to-cpu cpu-class TAC-class
```

```
CLI (network-admin@switch) > vflow-show
```

name	scope	type	in-port	burst-size	precedence	action	enable	cpu-class
------	-------	------	---------	------------	------------	--------	--------	-----------

```
-----
myflow local vflow 10 auto default to-cpu enable TAC-class
```

**Note:** CPTP's vFlow match criteria can be freely chosen to match the specific rate limiting need of the user provided they include a `to-cpu` or `copy-to-cpu` action for packets.

After issue resolution, the user can delete a previously created custom CPU class if no longer needed:

```
CLI (network-admin@switch) > cpu-class-delete
```

name	Specify the name of the CPU class to delete.
------	--

Or you can modify the CPU class after creating it:

```
CLI (network-admin@switch) > cpu-class-modify
```

name	Specify the name of the CPU class.
rate-limit rate-limit-number	Specify the cap for the rate limit.
hog-protect disable enable enable-and-drop	Specify if you want to enable, enable and drop packets, or disable hog protection.

**Informational Note:** You cannot modify the scope of the CPU class. If you want to change the scope, you must delete the existing CPU class and create a new CPU class with the correct scope.



## Configuring Other Advanced CPTP Settings

Various additional settings are available to tweak the behavior of Control Plane Traffic Protection to one's needs (e.g., the threshold for the maximum number of acceptable CPU hog violators per port).

It is possible to show the values of the settings with the following command:

```
CLI (network-admin@switch) > cpu-class-settings-show
```

```
switch:                switch
hog-checker-interval(ms): 100
hog-max-violators-per-port: 50
hog-warning-threshold:    5
hog-violator-timeout(s):  20
```

In this example, the timeout for a violator (which is no longer misbehaving) to be removed from quarantine is 20 seconds. The maximum number of violators per port is set to 50. These settings can be modified with the following command:

```
CLI (network-admin@switch) > cpu-class-settings-modify
```

<code>hog-checker-interval <i>hog-checker-interval-number</i> (ms)</code>	Specify the hog checking interval in milliseconds.
<code>hog-max-violators-per-port <i>hog-max-violators-per-port-number</i></code>	Specify the maximum number of hog violators per port.
<code>hog-warning-threshold <i>hog-warning-threshold-number</i></code>	Specify the number of warnings at which host becomes hog violator.
<code>hog-violator-timeout <i>hog-violator-timeout-number</i> (s)</code>	Specify the timeout before restoring the hog violator to normal queue after an idle state.

## Showing and Clearing CPTP Statistics

You can view the CPTP statistics, including class 0s by using the following command:

```
CLI (network-admin@switch) > cpu-class-stats-show
```

<code>name</code>	Specify the name of the CPU class to clear statistics.
<code>cos cos-number</code>	Specify the CoS value for the CPU class.

Or to clear them:

```
CLI (network-admin@switch) > cpu-class-stats-clear
```

<code>name</code>	Specify the name of the CPU class to clear statistics.
<code>cos cos-number</code>	Clear the CoS value for the CPU class.
<code>hw-out-pkts hw-out-pkts-number</code>	Clear the hardware transmitted packet count.
<code>hw-drop-pkts hw-drop-pkts-number</code>	Clear the number of hardware dropped packets.
<code>sw-pkts sw-pkts-number</code>	Clear the number of packets processed in software.
<code>sw-drops-pkts sw-drops-pkts-number</code>	Clear the number of packets dropped in software because the queue is full.
<code>hog-violations hog-violations-number</code>	Clear the number of hog protection host violations and moved to separate queue.
<code>hog-warnings hog-warnings-number</code>	Clear the number of hog protection delegated bandwidth warnings.
<code>hog-hosts-in hog-hosts-in-number</code>	Clear the number of added hosts for hog protection.
<code>hog-hosts-out hog-hosts-out-number</code>	Clear the number of hosts removed from hog protection.
<code>hog-max-hosts-drops hog-max-hosts-drops-number</code>	Clear the number of dropped hosts with hog protection because the maximum number of hosts is reached.

A handy command to periodically check the CPU traffic statistics is the following:

```
CLI (network-admin@switch) > cpu-class-stats-show show-diff-interval 1
```

which refreshes the statistics every second on screen, and thus can be used to observe traffic spikes.

In case of an auto-quarantined traffic source, the violator's information and related statistics can be displayed with the following commands:

CLI (network-admin@switch) > hog-violator-show

<code>mac mac-address</code>	Displays the hog violator MAC address.
<code>vlan vlan-id</code>	Displays the hog violator VLAN ID.
<code>vxlan vxlan-id</code>	Displays the hog violator VXLAN ID.
<code>port port-number</code>	Displays the hog violator ingress port.
<code>cpu-class cpu-class-string</code>	Displays the hog violator original class.
<code>hog-cpu-class hog-cpu-class-string</code>	Displays the hog violator hog queue CPU class.
<code>created date/time: yyyy-mm-ddTHH:mm:ss</code>	Displays the time and date when hog violator is created.
<code>vflow vflow-string</code>	Displays the redirect vFlow.
<code>vflow2 vflow-string</code>	Displays the redirect vFlow 2.
<code>vflow3 vflow-string</code>	Displays the redirect vFlow 3.

CLI (network-admin@switch) > hog-violator-stats-show

<code>time date/time: yyyy-mm-ddTHH:mm:ss</code>	Displays the time and date to start statistics collection.
<code>start-time date/time: yyyy-mm-ddTHH:mm:ss</code>	Displays the start time of the statistics collection.
<code>end-time date/time: yyyy-mm-ddTHH:mm:ss</code>	Displays the end time of the statistics collection.
<code>duration duration: #d#h#m#s</code>	Displays the duration of statistics collection.
<code>interval duration: #d#h#m#s</code>	Displays the interval between statistics collection.
<code>since-start</code>	Displays the statistics collection since the start time.
<code>older-than duration: #d#h#m#s</code>	Displays the statistics collection older than the time.
<code>within-last duration: #d#h#m#s</code>	Displays the statistics collection within a specified time period.
<code>name vflow-name</code>	Displays the name of the vFlow.
<code>vnet vnet-name</code>	Displays the VNET name of the vFlow.

id	Displays the ID assigned to the vFlow.
----	--

## Examples and Use cases for QoS

---

- [Configuring DSCP to CoS Mapping](#)
- [Applying CoS Queue Mapping based on Re-Marked DSCP in vFlow](#)

## Configuring DSCP to CoS Mapping

---

Netvisor One supports creating Quality of Service (QoS) maps that configure hardware based mapping of Differentiated Services Code Point (DSCP) value in a received IP header to a Cost of Service (CoS) priority. This helps in prioritizing traffic based on DSCP markings by using the appropriate egress CoS queues to send packets out.

Netvisor One sets the DSCP value to the 6 upper bits in the 8-bit ToS field of an IP header. Details about the specific values and the proposed traffic disposition can be found in these RFCs :

- RFC 2474 (DS Fields Definitions)
- RFC 2475 (DiffServ architecture)
- RFC 2597 (AF PHB Group)
- RFC 2780 (IANA Allocation Guidelines)

A quick summary of DSCP in Netvisor One:

- DSCP values range from 0 to 63, while packet priorities map to 8 CoS values or priority queues.
- Standards (IANA) include specific values in their guidelines. These values are used by different vendors to facilitate inter-connectivity.
- Class selector code points (CS0 through CS7, multiples of 8) are backwards compatible with IP ToS values. These values also serve as base selectors for other values.
- Assured Forwarding (AF) code points have 4 priority classes, each class has three code points indicating the drop precedence.

Class1: AF11/12/13 (DSCP 10, 12, 14)

Class2: AF21/22/23 (DSCP 18, 20, 22)

Class3: AF31/32/33 (DSCP 26, 28, 30)

Class4: AF41/42/43 (DSCP 34, 36, 38)

- 0 is best effort (CoS 0, default)
- 46 is an Expedited Forwarding (EF) code point, indicating critical traffic.

There are new commands to support this feature:

```
CLI (network-admin@Spine1) > dscp-map-create
```

---

<code>dscp-map-create</code>	Create a DSCP priority mapping table with default DSCP to priority mappings.
<code>name name-string</code>	Create a name for the DSCP map

---

```
CLI (network-admin@Spine1) > dscp-map-delete
```

---

<code>dscp-map-delete</code>	Delete a DSCP priority mapping table.
<code>name name-string</code>	The name of the DSCP map to delete.

---

```
CLI (network-admin@Spine1) > dscp-map-show
```

---

<code>dscp-map-show</code>	Display a DSCP priority mapping table
<code>name name-string</code>	Display the name of the DSCP map.

---

This command displays output only if there are maps configured.

```
CLI (network-admin@Spine1) > dscp-map-pri-map-modify
```

---

<code>dscp-map-pri-map-modify</code>	Update priority mappings in tables.
<code>dscp-map selector:</code> <code>name name-string</code>	Specify the name for the DSCP map to modify.
the following pri-map arguments:	
<code>pri number</code>	Specify a CoS priority from 0 to 7.
<code>dsmmap number-list</code>	Specify a DSCP value(s) as a single value, comma separated list, or a number range.

---

```
CLI (network-admin@Spine1) > dscp-map-pri-map-show
```

---

<code>dscp-map-pri-map-show</code>	Display priority mappings in tables.
<code>dscp-map selector:</code> <code>name name-string</code>	Display the name of the DSCP map.
the following pri-map arguments:	
<code>pri number</code>	Display a CoS priority from 0 to 7.
<code>dsmmap number-list</code>	Display a DSCP value(s) as a single value, comma separated list, or a number range.

---

The `dscp-map-pri-map-show` displays output only if there are maps configured.

The default values are listed in the following `dscp-map-pri-map-show` output:

```
CLI (network-admin@Spine1) > dscp-map-pri-map-show name dscp-map1
```

switch	name	pri	dsmap
Spine1	ds2	0	none
Spine1	ds2	1	8,10,12,14
Spine1	ds2	2	16,18,20,22
Spine1	ds2	3	24,26,28,30
Spine1	ds2	4	32,34,36,38
Spine1	ds2	5	40
Spine1	ds2	6	48
Spine1	ds2	7	56

The command, `port-config-modify`, has a new parameter, `dscp-map map-name | none` to support this feature.

Using the option `none` deletes or cancels a DSCP map previously configured on the port.



## Applying CoS Queue Mapping based on Re-Marked DSCP in vFlow

Currently, Netvisor ONE allows a vFlow to mark or re-mark matched packets with a DSCP value on egress. Netvisor ONE does not prioritize this traffic in terms of the egress port CoS queue selected for transmit. Another feature, *Enabling DSCP to Priority and CoS Mappings* introduces the ability to create DSCP QoS maps and apply to ports, but the maps apply to ingress packets. This feature introduces the ability to prioritize traffic based on the remarked DSCP value in a vFlow.

Netvisor ONE enables you to create named DSCP maps as independent objects, and applies the maps to ingress ports for prioritization of packets based on the DSCP markings. In this feature, you can apply the same maps in a vFlow. QoS maps can be applied to ports, but not to Flow Processor entries corresponding to vFlows. This implementation does the prioritization explicitly, since flows can be configured with CoSQ values.

The implementation has the following features:

- Verify the DSCP map named in the vFlow exists.
- Determine the priority and CoS for the DSCP value assigned to the vFlow.
- Apply this CoS value to the Flow Processor entry in hardware.
- Reconfigure CoS in the flow when the vFlow DSCP setting changes.
- Prevent deleting a DSCP map in use by a vFlow.
- Update the CoS setting of vFlows using the DSCP map when the DSCP map priority settings are updated.

You can specify the name of a DSCP map in the `vflow-create` command:

```
dscp-map dscp-map name | none
```

Specify the DSCP map to apply on the flow.  
Please reapply if map priorities are updated.

## Configuring and Using Network Telemetry

---

This chapter provides information for understanding and using the Pluribus Networks Netvisor ONE command line interface (CLI) on a Netvisor ONE switch to configure Network Telemetry.

---

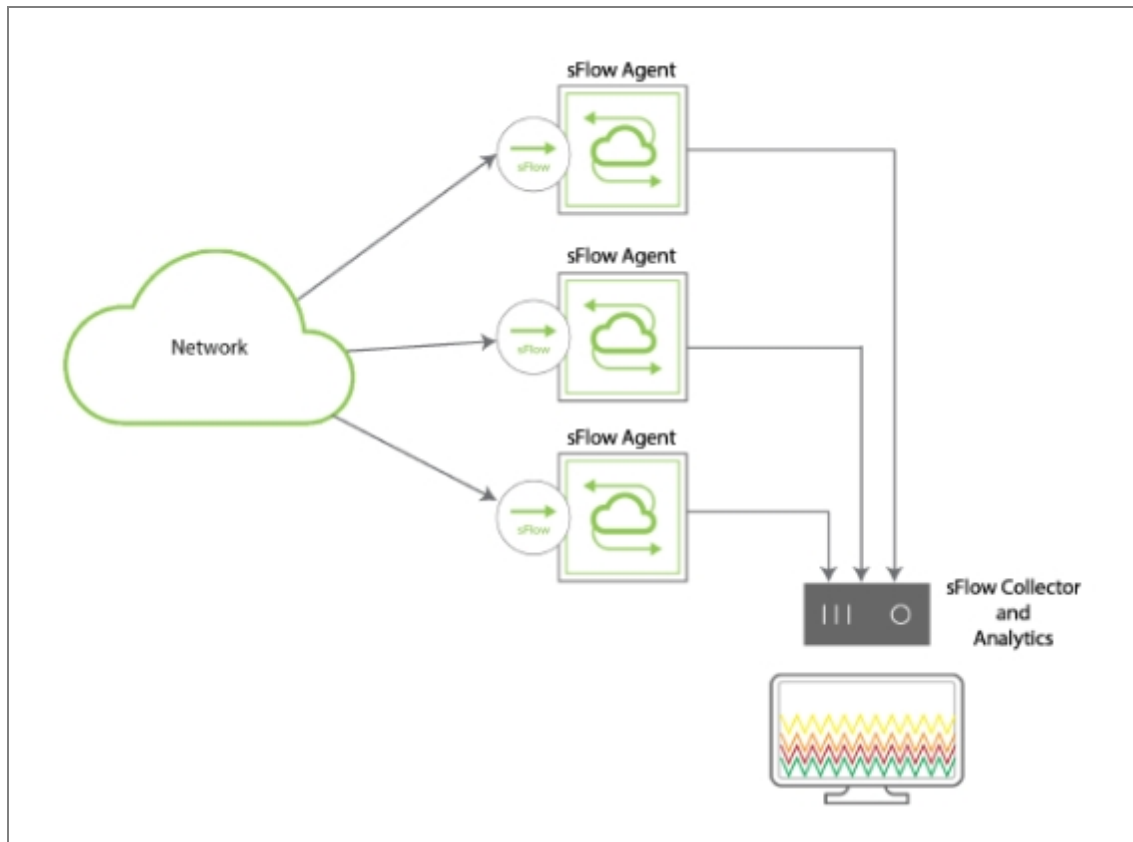
- [About sFlow](#)
  - [Guidelines While Configuring sFlow](#)
  - [Configuring the sFlow Collector](#)
  - [Configuring sFlow Agents on the Network](#)
  - [Adding or Removing Additional Ports to sFlow](#)
  - [Using Wireshark to Analyze Packets in Real Time](#)
  - [Analyzing Live Traffic Using Wireshark](#)
-

## About sFlow

As businesses rely on network services for mission critical applications, small changes in network usage can impact network performance and reliability. These changes can impact a business' ability to conduct important business functions, which can increase the cost of maintaining network services.

sFlow (also known as 'sampled flow'), is an industry standard monitoring feature that provides visibility into network usage and active routes on the network. The sFlow feature provides the data required to effectively control and manage network usage and supports application-level traffic flows at wire-speed on all physical interfaces. This capability ensures that network services provide a competitive edge to the businesses.

In Netvisor ONE, the sFlow monitoring system has two main components: the sFlow collector and sFlow agent. The sFlow agent runs on Pluribus switches for sampling of packets and are sent to the sFlow collector for further processing, see **Figure 13-1** below.



**Figure 13-1 - Sample Topology of sFlow Monitoring System**

The sFlow monitoring system supports two types of samplers: (i) Ingress sFlow sampler and (ii) Egress sFlow sampler. You can configure both or one of the sampler types on a port in Pluribus switches to enable sampling of IN and OUT packets simultaneously. That is, the Pluribus switches support sFlow at port level, allowing you to configure the sample threshold rates using the Netvisor ONE CLI commands.

A few examples of sFlow applications include the following:

- Detecting, diagnosing, and fixing network problems
- Real-time congestion management
- Understanding application mixes such as HTTP, Web, DNS
- Accounting usage for billing
- Audit trail analysis to identify unauthorized network activity and trace sources of Denial of Service (DoS) attacks
- Route profiling and optimizing peers
- Trending and capacity planning

sFlow is an open source sampling tool providing constant traffic flow information on all enabled interfaces simultaneously. sFlow data is sent to a collector that formats the data into charts and graphs while recording and identifying trends on the network. You can use this information for troubleshooting a network, performing diagnostics, and analyzing the data.

## Packet Sampling

Packet sampling is used to characterize the network traffic. If the sFlow agent on the switch is configured for packet sampling, the agent takes copies of random samples of forwarded packets within the CPU and forwards the headers and some of the payload data of the sample packet to the sFlow collector at regular intervals. You can specify the number of packets to sample from the total packets and this is called the *sample rate*. On an average, one in every **N** packets is captured and analyzed. Random sampling prevents the synchronization of periodic traffic patterns.

The packets are stored and sent to the sFlow collector at an interval that you can configure on the switch. This is called the *polling interval*. You can apply sampling to ingress and egress frames independently. The rate at which the sFlow agent sends datagram depends on the sampling rate, the traffic rate, and the configured maximum datagram size.

You can sample different types of packets such as frames sent to the CPU or interfaces of the switch, routed packets, flooded packets, and multicast packets.

However, the following packet types are not sampled by sFlow:

- LACP frames
- LLDP frames
- STP RPDUs
- IGMP packets
- Ethernet PAUSE frames
- Frames with CRC errors
- PIM\_HELLO packets

- Packets dropped by ACLs/vFlows
- Packets dropped as a result of VLAN violations
- Routed packets with IP options or MTU violations

To enable sFlow on your switch, you must configure the following (described in subsequent sections):

- sFlow Collector
- Add Exporter license (see UNUM documentation for details on adding the license)
- sFlow Agent

## Guidelines While Configuring sFlow

---

Here are some guidelines to consider while configuring the sFlow monitoring feature:

- The sFlow monitoring is a *local* scoped feature.
- The sFlow agent is a thread that runs continuously and gets the signals of sFlow packets, which is sent to sFlow collector.
- The default sample rate for sFlow configuration is 4096.
- The sFlow agent ID is usually a switch management IP address and indicates the originator or sample frames provider.
- The sFlow monitoring feature supports only IPv4 sampling.
- The sFlow collector configuration requires a collector IP address and collector ports.
- The sFlow collector configuration can be *local* or *fabric* scoped feature.
- You can configure multiple collector configurations, however, the first or the latest available collector is selected for sending the sampled packets.

## Configuring the sFlow Collector

You must configure sFlow collector before configuring the sFlow agents. The sFlow collector receives sFlow datagrams from the sFlow agents.

To configure a sFlow collector, use the command:

```
CLI (network-admin@switch) > sflow-collector-create collector-ip ip-address
collector-port collector-port-number name name-string scope local|fabric
```

<code>collector-ip ip-address</code>	Specify the sFlow collector IP address.
<code>collector-port collector-port-number</code>	Specify the sFlow collector port.
<code>name name-string</code>	Specify a name for the sFlow.
<code>scope local fabric</code>	Specify if the scope is local or fabric.

Below is an example configuring an sFlow collector with IP address **10.10.10.12**, on port **2055**. The collector name is **sf\_collector**, and the scope is **fabric**.

**Note:** If the scope is fabric, then additional switches that join the fabric receive the sFlow collector configuration. If the scope is local, then the sFlow collector is configured only on the local switch.

```
CLI (network-admin@switch) > sflow-collector-create collector-ip
10.10.10.12 collector-port 2055 name sf_collector scope fabric
```

**Note:** You can add as many collectors as needed for your configuration.

To delete an already configured sFlow collector, use the command:

```
CLI (network-admin@switch) > sflow-collector-delete name name-string
```

To view the details of the existing sFlow collectors, use the `sflow-collector-show` command specifying one or more of the following options:

```
CLI (network-admin@switch) > sflow-collector-show
```

Specify one of more of the following options:	
<code>collector-ip ip-address</code>	Displays the sFlow collector IP address.
<code>collector-port collector-port-number</code>	Displays the sFlow collector port.

<code>name</code> <i>name-string</i>	Displays the sFlow name.
<code>scope</code> <code>local</code>   <code>fabric</code>	Displays the scope of the sFlow collector.

For example, to view the details of the previously configured sFlow collector,

```
CLI (network-admin@switch) > sflow-collector-show
```

collector-ip	collector-port	name	scope
10.10.10.12	2055	sf_collector	fabric
10.13.26.75	6343	sflow_collect	local



## Configuring sFlow Agents on the Network

You must configure and enable sFlow on each switch that you want to use for monitoring the network traffic.

To configure or enable sFlow agent on the network, use the command:

```
CLI (network-admin@switch) > sflow-create name name-string [type ingress|
egress] [sample-type raw|cooked] ports port-list [sample-rate sample-rate-
number] [sample-interval sample-interval-number] [trunc-length trunc-
length-number] [vlan 0..4095] [agent-id ip-address]
```

<code>name</code> <i>name-string</i>	Specify a name for the sFlow.
<code>[type ingress egress]</code>	Specify whether the sFlow type is for ingress or egress traffic.
<code>[sample-type raw cooked]</code>	Specify the sFlow sample type, recommended sample type is raw.
<code>ports</code> <i>port-list</i>	Specify the sFlow ports.
<code>[sample-rate sample-rate-number]</code>	Specify the sFlow sample rate. The default value is 4096.
<code>[sample-interval sample-interval-number]</code>	Specify the sampling interval in seconds (1 sample in <b>N</b> seconds). The default value is 60 seconds.
<code>[trunc-length trunc-length-number]</code>	Specify the truncated length of the sFlow sample (sample packet size).
<code>[vlan 0..4095]</code>	Specify the VLAN ID. The VLAN option is configurable only on some hardware platforms. An error message is displayed if the platform does not support VLAN configuration for sFlow (see example below).
<code>[agent-id ip-address]</code>	Specify the local IP address.

Below is an example configuration on each switch for enabling the sFlow agent, **net-monitor**, on ingress ports **57-59**, sample type **raw**, sample-rate **4096**, sample interval **5** seconds (1 sample in 5 seconds), trunc-length **160** bytes:

```
CLI (network-admin@switch) > sflow-create name net-monitor type ingress
sample-type raw ports 57-59 sample-rate 4096 sample-interval 5 trunc-length
160 vlan 10
```

**Note:** Some platforms do not support VLAN configuration for sFlow. An error message is displayed when you configure an sFlow with VLAN configuration. For example:

```
CLI (network-admin@switch) > sflow-create name net-monitor type ingress
ports 10 vlan 10
sflow-create: This hardware platform does not support vlan specification
for sflow
```

To delete an existing sFlow agent, use the command:

```
CLI (network-admin@switch) > sflow-delete name name-string
```

To view the sFlow agent details, use the command:

```
CLI (network-admin@switch) > sflow-show
```

Specify one or more of the options:	
<code>name <i>name-string</i></code>	Displays the name for the sFlow.
<code>[type ingress egress]</code>	Displays whether the sFlow type is for ingress or egress traffic.
<code>[sample-type raw cooked]</code>	Displays the sFlow sample type.
<code>ports <i>port-list</i></code>	Displays the sFlow ports.
<code>[sample-rate <i>sample-rate-number</i>]</code>	Displays the sFlow sample rate.
<code>[sample-interval <i>sample-interval-number</i>]</code>	Displays the sampling interval
<code>[trunc-length <i>trunc-length-number</i>]</code>	Displays the truncated length of the sFlow sample.
<code>[vnet <i>vnet-name</i>]</code>	Displays the VNET name for the VLAN.
<code>[bd <i>bridge-domain name</i>]</code>	Displays the sFlow bridge domain name
<code>[vlan <i>0..4095</i>]</code>	Displays the VLAN ID.
<code>[agent-id <i>ip-address</i>]</code>	Displays the local IP address.

For example, to view the details of a previously configured sFlow agent by using the command,

```
CLI (network-admin@switch) > sflow-show layout vertical
```

```
name:      net-monitor
type:      ingress
sample-type:  raw
ports:     57-59
sample-rate:  4096
sample-interval:  5
trunc-length: 160
vnet:
bd:
vlan:
```

```
agent id: 10.1.1.243  
sample-pkt-cnt:      2845  
sample-drops:  0
```

## Adding or Removing Additional Ports to sFlow Configuration

Based on your sampling requirements, you can add additional ports to the sFlow agent on each switch by using the command,

```
CLI (network-admin@switch) > sflow-port-add sflow-name name-string switch
switch-name ports port-list
```

Specify the sflow selector:	
<i>sflow-name name-string</i>	Specify the sFlow name.
Specify the following port arguments:	
<i>[switch switch-name]</i>	Specify the switch name. This is a hidden option.
<i>[ports port-list]</i>	Specify the ports or range of ports.

For example, to add the ports, **61-62**, to the sFlow configuration, use the following command on each switch:

```
CLI (network-admin@switch) > sflow-port-add sflow-name net-monitor ports
61-62
```

To remove the additional ports from the sFlow configuration, use the `sflow-port-remove` command. For example,

```
CLI (network-admin@switch) > sflow-port-remove sflow-name net-monitor
ports 61-62
```

## Using Wireshark to Analyze Packets in Real Time

To use Wireshark to interactively analyze packets in real time, you need to capture a packet traffic flow, either on a specific switch or across the entire fabric using the scope option. Include the `log-packets` option to send packets to the associated pcap files, for example:

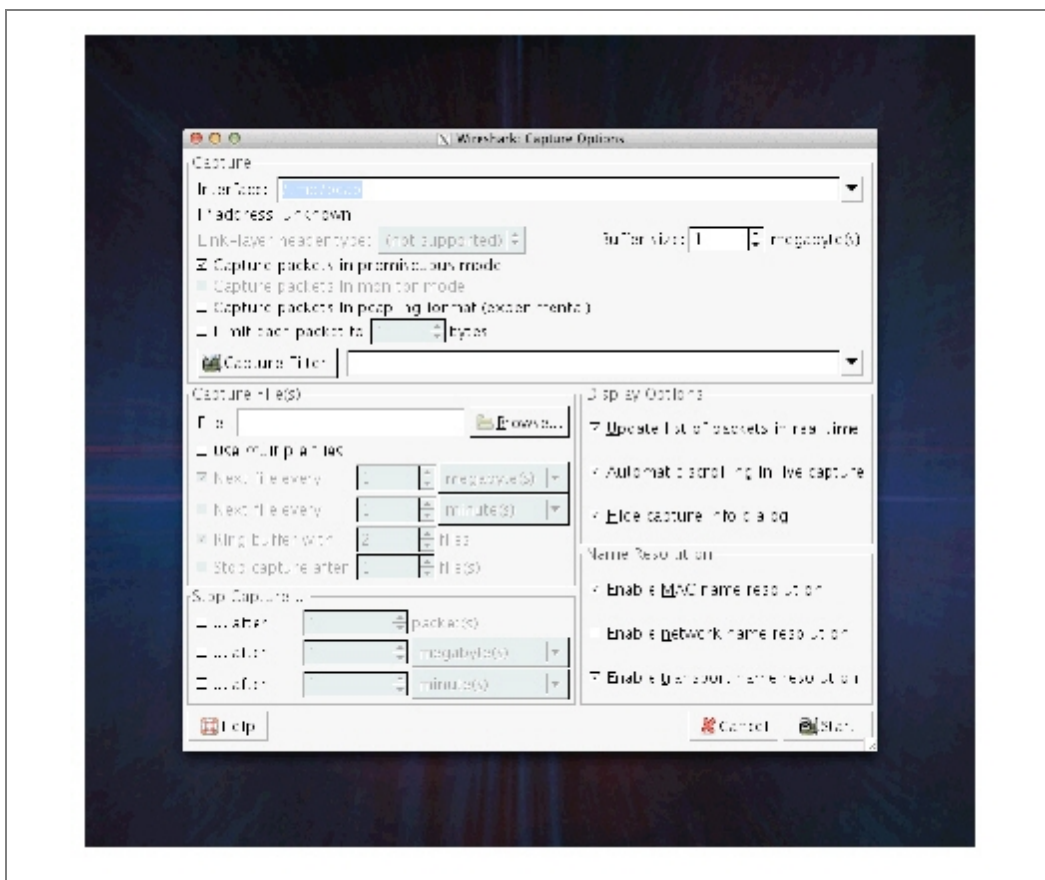
```
CLI (network-admin@Leaf1) > vflow-snoop scope fabric src-ip 112.168.3.105
action copy-to-cpu log-packets
```

Next, create a fifo on the host running Wireshark.

```
mkfifo /tmp/pcap
```

Start Wireshark, and select **Options** from the **Capture** menu.

Enter the fifo path that you created in the Interface field: `/tmp/pcap`



**Figure 12-8 - Wireshark Interface**

### Wireshark Capture Options

Use `tail` to copy the pcap file to the FIFO:

```
tail +0f \  
/net/ServerSw_Name//global/flow/Flow_Name/switch/Switch_Name/pcap/tmp/pcap
```

You need to substitute *ServerSw\_Name*, *Flow\_Name* and *Switch\_Name* to match your environment.

Live capture continues until the packet capture file is rotated. By default, the maximum packet capture file size is 10MB but it is configurable with the `packet-log-max` option of the `vflow-create` and `vflow-modify` commands.

**Note:** The `mkfifo` command used in this task is a standard feature of Linux-like operating systems, including MacOS. For Windows platforms, you may need to install the GNU CoreUtils package available at <http://gnuwin32.sourceforge.net/packages/coreutils.htm>.

## Analyzing Live Traffic Using Wireshark

---

Wireshark is a well known network protocol analyzer and one of many applications used for network protocol analysis.

Wireshark can interactively browse packet data from a live network or from a previously save pcap file.

**Note:** You can download Wireshark from <http://www.wireshark.org>

To use Wireshark to decode a previously saved packet flow capture file, export the file from the switch and analyze it with Wireshark.

**Note:** The path to a Netvisor One switch pcap file

is: `/net/<ServerSw_Name>/ONVL/global/flow/<Flow_Name>/<Switch_Name>/pcap`

## Configuring TACACS+

---

This chapter provides information for understanding and using the Pluribus Networks Netvisor ONE command line interface (CLI) on a Netvisor ONE switch to configure TACACS+.

---

- [Understanding TACACS+](#)
  - [Configuring TACACS+](#)
  - [Creating User Roles](#)
  - [Exceptions for Audit Logging](#)
-



## Understanding TACACS+

---

### About TACACS+

Terminal Access Controller Access Control System (TACACS+) is an Authentication, Authorization, and Accounting (AAA) protocol that provides a centralized database to use for authentication and authorization. This protocol uses a client server approach by which the client queries a server and the server replies with a pass or fail for authentication. The communication between the client and server uses TCP as the transport protocol, and requires a secret key.

Netvisor ONE allows you to configure external TACACS+ servers for authentication, authorization, and accounting of sessions. You can configure any number of TACACS+ servers, and each server may be configured to handle any combination of authentication, session authorization, command authorization, session accounting, and command accounting.

#### Note :

- The default *network-admin* account is exempt from all TACACS+ integration as a fail-safe account for sites without TACACS+ and also to allow access to Pluribus Networks facilities if TACACS+ is unavailable or unreachable. The admin account requires a code from Pluribus Networks Customer Advocacy to login. Because you can access the shell through the CLI, the administrator's account is rarely needed.
- The Pluribus account password is the same as the network-admin password and when the network-admin password is changed, the Pluribus account password also changes. The login shell for the Pluribus account is *nvauditsh* and the same can be used to enable auditing for commands run from the admin account. Logins on both types of accounts are disabled if a TACACS+ server with the parameter `authen-local` is active.

When the TACACS+ server is configured and is working correctly, then the :

- Local users from */etc/passwd* or */var/nvOS/etc/\*/user\_config.xml* are not allowed to login
- Authorization and Accounting for shell commands are enabled
- Authorization and Accounting for FRR commands are enabled
- Ability to drop into the shell from the CLI commands are allowed

You can configure TACACS+ by using the `aaa-tacacs-create` command and then, you can login to the switch and get CLI access using an account configured on the specified TACACS+ server.

The parameter `authen-local` controls if a TACACS+ server is used to authenticate local accounts or not. Netvisor ONE tracks errors between Netvisor ONE and the TACACS+ server and if a communication error occurs, then local authentication is allowed until communication with the TACACS+ server is recovered. This functionality enables the recovery of the system during an issue in the TACACS+ server. To allow a local account to authenticate, you must configure all active `aaa-tacacs` instances with `no-authen-local` option.

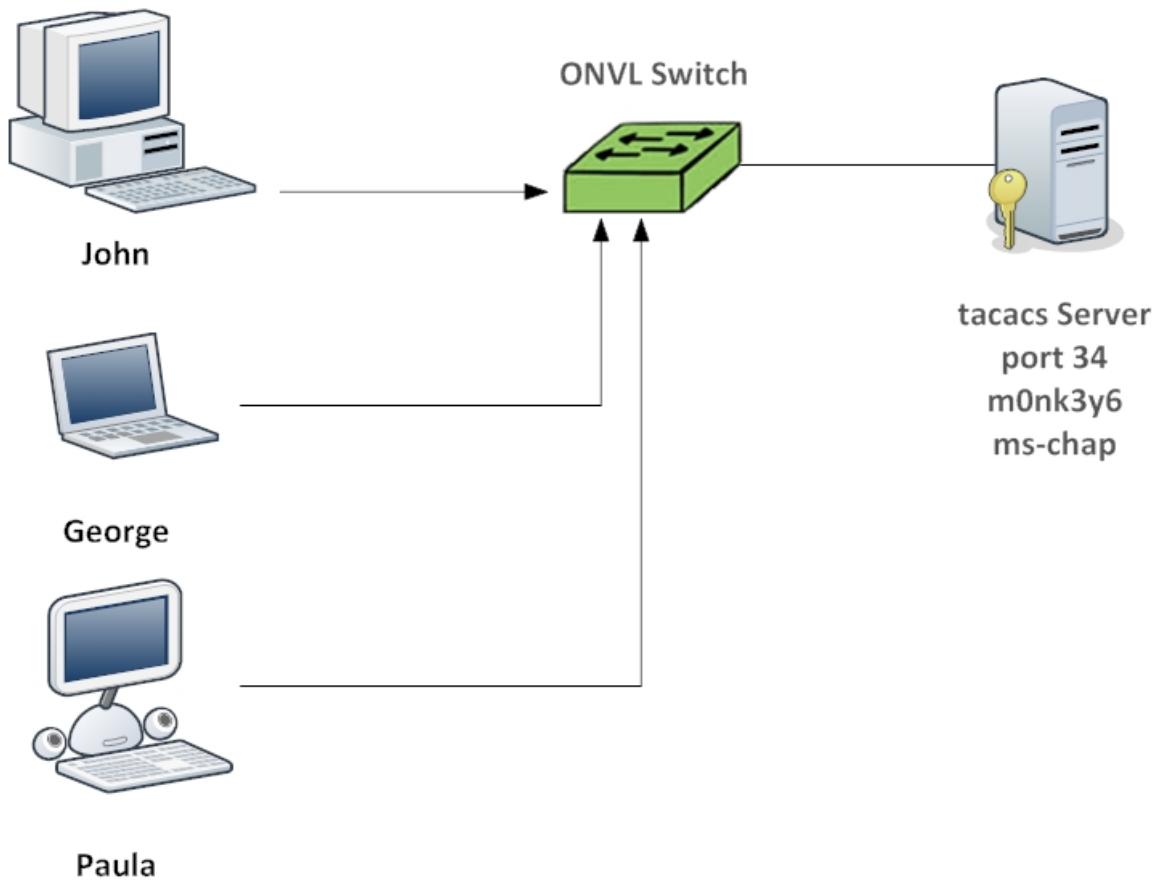
Local accounts include the accounts configured using the commands, `user-create` and `user-modify`, and the accounts such as `admin` and `pluribus`. Logins for both types of accounts are disabled if a TACACS+ server is set with `authen-local` as active.

The `vttysh` command is used to configure FRR for authorization and accounting of FRR commands.

The authentication protocols supported for TACACS+ are:

- PAP
- CHAP
- MS-CHAP

**Figure 13-1** illustrates a simple TACACS+ implementation.



**Figure 14-1: TACACS+ Implementation**

## Configuring TACACS+

To configure or create TACACS+ access on a switch, use the command:

```
CLI (network-admin@switch-1) > aaa-tacacs-create name name-string scope
local|fabric server server-string [port port-number] [secret secret-
string] [timeout timeout-number] [priority priority-number] [authen|no-
authen] [authen-local|no-authen-local] [authen-method pap|chap|ms-chap]
[sess-acct|no-sess-acct] [cmd-acct|no-cmd-acct] [acct-local|no-acct-local]
[sess-author|no-sess-author] [cmd-author|no-cmd-author] [author-local|no-
author-local] [author-local|no-author-local] [service service-string]
[service-shell service-shell-string] [service-vtysh service-vtysh-string]
```

<code>name <i>name-string</i></code>	Specify the name for TACACS+ config
<code>scope local fabric</code>	Specify the scope of TACACS+
<code>server <i>server-string</i></code>	Specify the TACACS+ server string
Specify one of more of the following options	
<code>[port <i>port-number</i>]</code>	Specify the TACACS+ communication port
<code>[secret <i>secret-string</i>]</code>	Specify the shared secret for TACACS+
<code>[timeout <i>timeout-number</i>]</code>	Specify the number of seconds before communication times out
<code>[priority <i>priority-number</i>]</code>	Specify the priority for TACACS+
<code>[authen no-authen]</code>	Specify whether to use authentication or no authentication
<code>[authen-local no-authen-local]</code>	Specify if the authentication overrides local users
<code>[authen-method <i>pap chap ms-chap</i>]</code>	Specify the authentication methods: PAP, CHAP (default), MS-CHAP
<code>[<i>sess-acct no-sess-acct</i>]</code>	Specify the session accounting
<code>[<i>cmd-acct no-cmd-acct</i>]</code>	Specify the command accounting
<code>[<i>acct-local no-acct-local</i>]</code>	Specify the accounting for local users
<code>[<i>sess-author no-sess-author</i>]</code>	Specify the authorization sessions
<code>[<i>cmd-author no-cmd-author</i>]</code>	Specify the command authorization
<code>[<i>author-local no-author-local</i>]</code>	Specify the authorization for local users
<code>[<i>service service-string</i>]</code>	Specify the service name used for TACACS+ requests sent from Netvisor ONE to the TACACS+ server for commands run at the Netvisor CLI and REST APIs. The default value is <i>shell</i> .

<code>[service-shell service-shell-string]</code>	Specify the TACACS+ service name string for <i>shell</i> commands
<code>[service-vtysh service-vtysh-string]</code>	Specify the TACACS+ service name string for <i>vttysh</i> commands

For example, to create TACACS+ account, `tac` having scope `local` with *no local authentication* privilege, use the command:

```
CLI (network-admin@switch) > aaa-tacacs-create name tac scope local authen-local
```

To modify the authentication access, use the command:

```
CLI (network-admin@switch) > aaa-tacacs-modify name tac no-authen-local
```

For a local account to authenticate, all the active `aaa-tacacs` instances must be configured with `no-authen-local` parameter.

Use the parameters `author-local` and `acct-local` to indicate if authorization and accounting messages for locally authenticated accounts should be sent to the TACACS+ server. For example,

```
CLI (network-admin@switch) > aaa-tacacs-modify name tac [author-local|no-author-local]
```

```
CLI (network-admin@switch) > aaa-tacacs-modify name tac [acct-local|no-acct-local]
```

To specify the *service* in authorization and accounting messages for *shell* and *vttysh* commands, use:

```
CLI (network-admin@switch) > aaa-tacacs-modify name tac \
service-shell unix-shell
```

```
CLI (network-admin@switch) > aaa-tacacs-modify name tac \
service-vtysh vtysh-shell
```

If *service-shell* or *service-vtysh* is not specified, then the value of the service option is used.

To delete a specified (for example, `tac`) TACACS+ configuration, use the `aaa-tacacs-delete` command:

```
CLI (network-admin@switch) > aaa-tacacs-delete name tac
```

To display the status of the TACACS server, use the `aaa-tacacs-status` command:

```
CLI (network-admin@switch) > aaa-tacacs-show name tac
```

## Creating User Roles

The TACACS+ server determines what role a user has by returning a role attribute. The roles include network-admin for full access and read-only-network-admin users who can run only the show commands.

You can create users on the switch and assign roles(based on role created or the default role of network-admin. To create a new user role, use the command:

```
CLI (network-admin@switch-1) > role-create name <name-string> scope local |
fabric access read-only|read-write
```

name <name-string>	Specify the role name
scope local fabric	Specify if the scope is local or fabric
Specify any of the following options:	
access read-only read-write	Specify the type of access, the default is read-write
running-config no-running-config	Specify to display running configuration of switch
shell no-shell	Specify to allow shell command
sudo no-sudo	Specify to allow sudo from shell

For example, to create a user, newrole, with scope, local and allowing shell command, use the command:

```
CLI (network-admin@switch-1) > role-create name newrole scope local shell
```

To display the configuration details, use the command:

```
CLI (network-admin@switch-1) > role-show
```

name	scope	vnet-access	access	running-config	shell	sudo
network-admin	local	all	read-write	permit	deny	deny
read-only-network-admin	local	all	read-only	deny	deny	deny
newrole	local	all	read-write	deny	permit	deny

To delete a role, use the command:

```
CLI (network-admin@switch-1) > role-delete name <name-string>
```

To modify a role, use the command:

```
CLI (network-admin@switch-1) > role-modify name <name-string> access read-
only|read-write running-config|no-running-config shell|no-shell sudo|no-
sudo delete-from-users
```

You can create a user and apply an initial role by using the command:

```
CLI (network-admin@switch-1) > user-create name <name-string> scope local |
fabric initial-role <role-name>
```

For example, to create a user, user2 with scope local and initial role as network-admin:

```
CLI (network-admin@switch-1) > user-create name user2 scope local initial-  
role newrole
```

```
CLI (network-admin@switch-1) > shell  
shell: shell access denied by role
```

The above error message indicates that shell access is denied because the network-admin does not have shell access configured. Hence, when you create a new user, you must assign a new role to the user so as to enable shell access. For example,

```
CLI (network-admin@switch-1) > user-create name user2 scope local initial-  
role newrole
```

Exit the configuration and login back to the switch as user2, for shell access to be enabled:

```
jenkins@pn-jenkins2.pluribusnetworks.com:~$ ssh user2@switch-1  
Warning: Permanently added 'switch-1,10.14.16.44' (ECDSA) to the list of  
known hosts.  
* Welcome to Pluribus Networks Inc. Netvisor(R). This is a monitored  
system.      *  
*              ACCESS RESTRICTED TO AUTHORIZED USERS ONLY  
*  
* By using the Netvisor(R) CLI, you agree to the terms of the Pluribus  
Networks *  
* End User License Agreement (EULA). The EULA can be accessed via  
*              *  
* http://www.pluribusnetworks.com/eula or by using the command "eula-show"  
*              *  
user2@switch-1's password:  
Netvisor OS Command Line Interface 5.1  
Connected to Switch switch-1; nvOS Identifier:0xb0013dc; Ver: 5.1.1-  
5010115002  
CLI (user2@switch-1) > shell  
user2@switch-1:~$  
user2@switch-1:~$ exit  
exit  
CLI (user2@switch-1) >  
CLI (user2@switch-1) > exit  
Connection to switch-1 closed.
```

To delete a user, use the command (add the parameter forcefully to delete a user forcefully):

```
CLI (network-admin@switch-1) > user-delete name <name-string> scope local  
initial-role newrole
```

To modify a user, use the command:

```
CLI (network-admin@switch-1) > user-modify name <name-string> password
```

`<password-string>`

To set a user password, use the command:

```
CLI (network-admin@switch-1) > user-password-set <name-string> scope local
uid uid-number type netvisor|unix|tacacs|web-token|mfg server aaa-tacacs
name initial-role <role-name>
```

<code>name &lt;name-string&gt;</code>	Specify the role name
Specify any of the following options:	
<code>scope local fabric</code>	Specify if the scope is local or fabric
<code>uid uid-number</code>	Specify the user ID
<code>type netvisor unix tacacs web-token mfg</code>	Specify the user type
<code>server aaa-tacacs name</code>	Specify the TACACS server
<code>initial-role &lt;role-name&gt;</code>	Specify the initial role for the user

## Exceptions for Audit Logging

The commands `log-audit-exception-create`, `log-audit-exception-delete`, and `log-audit-exception-show` are used to control which CLI, shell and vtysh commands are subject to auditing. If a command is subject to auditing, the command is logged in the audit log and sent to the TACACS+ server as authorization and accounting messages.

To create an audit logging exception, use the command:

```
CLI (network-admin@switch) > log-audit-exception-create cli|shell|vtysh
[pattern pattern-string] [any|read-only|read-write] scope local|fabric
```

<code>cli shell vtysh</code>	Specify the type of audit exception
<code>[pattern pattern-string]</code>	Specify the regular expression to match exceptions
<code>[any read-only read-write]</code>	Specify the access type to match exceptions
<code>scope local fabric</code>	Specify the scope ( <i>local</i> or <i>fabric</i> ) for exception

To delete an audit logging exception, use the command:

```
CLI (network-admin@switch) > log-audit-exception-delete cli|shell|vtysh
[pattern pattern-string] [any|read-only|read-write]
```

To display the audit logging exception, use the command:

```
CLI (network-admin@switch) > log-audit-exception-show cli|shell|vtysh
[pattern pattern-string] [any|read-only|read-write] scope local|fabric
```

By default, every command is audited except for *read-only CLI commands* and the shell command `^/usr/bin/nvmore`, which is the pager used by `nvOS_cli`:

```
CLI (network-admin@switch) > log-audit-exception-show
```

switch	type	pattern	access	scope
switch	cli		read-only	local
switch	shell	^/usr/bin/nvmore	any	local

To enable auditing of ALL CLI commands, you can delete the *read-only/CLI* exception:

```
CLI (network-admin@switch) > log-audit-exception-delete cli read-only
```



## Configuring Virtual Networks

---

This chapter provides information for understanding and using the Pluribus Networks Netvisor ONE command line interface (CLI) on a Netvisor ONE switch to configure Virtual Networks.

---

- [Understanding Virtual Networks \(vNETs\)](#)
- [Creating a Virtual Network](#)
- [Specifying the Type of vNET Interface](#)
- [Configuring vNET High Availability \(HA\)](#)
- [Related Documentation](#)

For more details, see the *Deploying Virtual Networks with Virtual Netvisor* guide.

---

## Understanding Virtual Networks (vNETs)

---

Pluribus Networks supports various network virtualization capabilities as part of a highly flexible approach to software-defined networking (SDN), called *Adaptive Cloud Fabric*.

Pluribus' fabric addresses all the most common network design requirements, including scalability, redundancy, predictable growth capability, fast convergence in case of a failure event, etc. Such requirements also include *multi-tenancy support*.

Therefore, in the data center, by leveraging Netvisor ONE's virtualization features, network designers can implement a variety of multi-tenancy models such as Infrastructure as a Service (IaaS) or Network as a Service (NaaS).

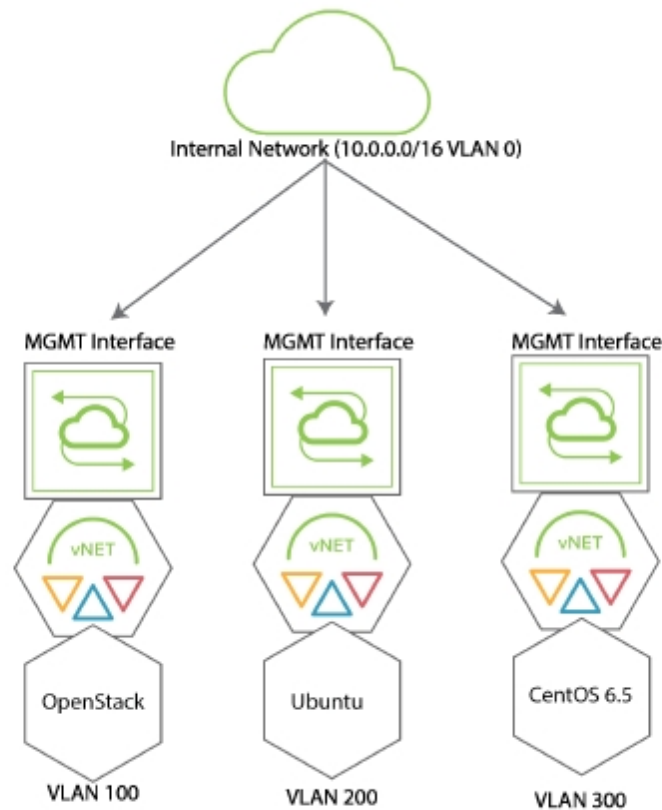
To support multiple tenants, the fabric's data plane segmentation technologies include standard features (such as VLANs) as well as advanced virtualization features such as VXLAN and distributed VRFs, to be able to deploy an open, interoperable, high-capacity and high-scale multi-tenant network. (Refer to the *Configuring VXLAN* section for more details on VXLAN and distributed VRFs).

In addition to data plane segmentation, fabric virtualization also comprises the capability of separating tenants into *isolated management domains*. Pluribus calls this capability *Virtual Networks (vNETs)* in short).

A vNET is an abstract control plane resource that is implemented globally across the fabric to identify a tenant's domain. By using vNETs, you can segregate a physical fabric into many logical domains, each with separate resources, network services, and Quality of Service (QoS) guarantees. vNETs therefore allow the network administrator to completely separate the provisioning of multiple tenants within the network.

Each vNET has a single point of management. As the fabric administrator, you can create VNETs and assign ownership of each vNET to individuals with responsibility for managing those resources.

You can create separate user names and passwords for each *vNET manager*. Using the separate vNET administration credentials, the vNET admin can use Secure Shell (SSH) to connect to the vNET manager and access a subset of the Netvisor ONE CLI commands to manage a specific vNET. This way, multiple tenants can share a fabric with each managing a vNET with security, traffic, and resource protection from other VNETs.



**Figure 15-1 - Using VNETs with Netvisor ONE**

VNETs are very flexible and can be used to create complex network architectures. For example, a Pluribus Networks switch, or a fabric of switches, can be used to create multiple tenant environments in an OpenStack deployment. In **Figure 14-1** above there are three VNETs, each one with a management interface and a data interface. Each vNET is assigned an IP address pool used for DHCP assignment of IP addresses to each node, server, or OS component.

Underlying each vNET is the *vNET manager*. Each vNET manager runs in a zone. When services are created for a vNET they occupy the same zone on a switch. This is called a shared service and it is the default when creating services. However, each zone can only support a single instance of a service. If a second service instance is needed for a vNET, then it needs to occupy a separate zone. This is called a dedicated service. In most cases, you can create services as shared resources unless you specifically want to create a dedicated service.

When a fabric is created, a vNET is also automatically created and named `fabric-name-global`. This vNET initially owns all resources within the fabric. As new vNETs are created, resources are moved from the default vNET to the new vNETs. Global services remain in the default vNET unless assigned to a specific vNET.

## Creating a Virtual Network (vNET)

---

To separate resources, including switch ports, IP addresses, VLANs, and VXLAN IDs, into separate management spaces, create (at least) a vNET and associate the desired resources to it (see below for an example of configuration).

The vNET creation is performed with the `vnet-create` command followed by a list of required parameters. Subsequently, you can configure a separate vNET administrator to manage each newly created management domain.

**Note:** You cannot create another vNET inside of a vNET.

The purpose of vNET objects is to provide independent network domains whose administrators are able to manage a set of dedicated resources without having to involve the fabric administrator, within the constraints that the fabric administrators defines.

Access control is performed based on the scope of a vNET, which includes a number of dedicated ports on which it's possible to apply certain commands/use certain dedicated resources.

For example, the applicable resources include three categories of commands/entities, listed below:

Layer 1 commands (i.e., port-related commands):

- o `port-show` (shows vNET ports but not internal/cluster ports)
- o `port-phy-show`
- o `port-config-modify/show`
- o `bezel-portmap-show`

Layer 2 commands (i.e., VLAN/LACP/STP/vLAG commands):

- o `port-lACP-modify/show`
- o `vlag-create/modify/delete`
- o `trunk-create/modify/delete`
- o `stp-port-modify/show/stp-portevent-show`
- o A VNET admin is not allowed to change the native VLAN on a shared port with the `port-vlan-add` command.

Layer 3 commands (i.e., vRouter commands):

- o `static-ecmp-group-show/static-ecmp-group-nh-show`
- o `vrouter-ping/traceroute`

Note that if any of the above commands are run on a port not in the scope of a vNET, a *No permission for port 'port = %d'* message is displayed, for example like so:

```
CLI (network-admin@switch) > vlag-create name vl1 port 99 peer-port 99
vlag-create: No permission for port 'port = 99'
```

Let us take a concrete example of vNET creation and see how the above commands behave within it:

```
CLI (network-admin@switch) > vnet-create name vn1 scope cluster vlan-type
private public-vlans 2000-2099 num-private-vlans 10 vxlans 10000100-
10000109 managed-ports 9,17 shared-ports 18 shared-port-vlans 105-109
```

Creating vn1-mgr zone, please wait...

With this command the fabric administrator creates a vNET as a dedicated domain comprising a number of managed and shared ports as well as private and public VLANs and VXLAN IDs. In other words, the fabric admin is partitioning the resources to provide a dedicated and *restricted* view on the network.

The following examples show how a vNET's view gets constrained for each command example:

```
CLI (network-admin@switch) > port-show
```

switch	port	bezel-port	status	config
switch	9	9	up,vlan-up	fd,10g
switch	17	17	disabled	fd,10g
switch	18	18	disabled	fd,10g

In this example the vNET admin can only show the managed and shared ports chosen as part of the vNET creation process, out of all the front panel ports.

```
CLI (network-admin@aquarius00) > port-phy-show
```

switch	port	state	speed	eth-mode	max-frame	learning	def-vlan
switch	9	up	10000	10Gbase-cr	1540	off	0
switch	17	down	10000	10Gbase-cr	1540	off	1
switch	18	down	10000	10Gbase-cr	1540	off	0

Also, the front panel and PHY information is constrained to the ports selected as part of the vNET creation process. Port configuration gets constrained too:

```
CLI (network-admin@aquarius00) > port-config-show format port,enable,
```

```
port enable
----
9      on
17     off
```

Regarding the Layer 2 configuration, these commands also get a constrained view:

```
CLI (network-admin@switch) > port-vlan-add port 9 untagged-vlan 45
port-vlan-add: No permission to modify untagged-vlan field
```

As shown, untagged VLANs are prevented from being changed.

The vLAGs can be created only using accessible ports (as port 9 in the example below):

```
CLI (network-admin@switch) > vlag-create name vl1 port 9 peer-port 9
CLI (network-admin@switch) > port-lacp-show layout vertical
```

```
switch:    switch
port:      9
name:      vl1
port-type: vlag
mode:      passive
timeout:   slow
system-id: 66:0e:94:b6:ab:01
lacp-key:  36285
system-priority: 32768
port-priority: 32768
aggregatable: yes
sync:      yes
coll:      yes
dist:      no
defaulted: yes
expired:   no
port-state: 0x5c
```

whereas inaccessible ports are blocked in the configuration:

```
CLI (network-admin@aquarius00) > vlag-create name vl1 port 99 peer-port 99
vlag-create: No permission for port 'port = 99'
```

Similarly, a VLAN trunk can be created (and then deleted) using accessible ports 9 and 10 like so:

```
CLI (network-admin@switch) > trunk-create name t ports 9-10
trunk 273 defer-bringup set to 1 based on first port 9
Created trunk t, id 273
```

```
CLI (network-admin@switch) > trunk-show format name,trunk-id,ports
```

name	trunk-id	ports
t	274	9-10
vxlan-loopback-trunk	397	

```
CLI (network-admin@switch) > trunk-delete name t
```

Furthermore, spanning tree (STP) commands are limited to accessible ports only:

```
CLI (network-admin@switch) > stp-port-show port 10
```

switch	port	block	filter	edge	bpdu-guard	root-guard	priority	cost
switch	<b>10</b>	on	off	no	no	no	128	2000

```
CLI (network-admin@switch) > stp-port-modify port 53 cost 10000
stp-port-modify: No permission over ports 53
```

```
CLI (network-admin@switch) > stp-port-event-show
```

switch	time	port	vlan	instance	count	initial-state	other-state	final-state
switch	01:13:52	17	1,4094	0	3	Disabled	Disabled	Forwarding
switch	01:15:42	9	1	0	1	Disabled	Disabled	Discarding
switch	01:16:02	9	1	0	1	Discarding	Disabled	Learning
switch	01:16:12	9	1	0	1	Learning	Disabled	Forwarding
switch	01:17:53	17	1	0	1	Forwarding	Disabled	Disabled
switch	01:17:53	9	1	0	1	Forwarding	Disabled	Disabled
switch	01:29:00	17	4094	0	1	Disabled	Disabled	Forwarding

The vRouter commands get constrained too (to accessible VLANs and interfaces) like so:

```
CLI (network-admin@switch) > vrouter-create name vr1 vnet vn1 router-type
hardware
Creating vr1 zone, please wait...
vrouter created
```

```
CLI (network-admin@switch) > vrouter-interface-add vrouter-name vr1 ip
192.168.99.13/24 vlan 100
Added interface eth0.100 with ifIndex 159
```

```
CLI (network-admin@switch) > vrouter-interface-show format vrouter-
name,ip,vnet,vlan,vlan-type,nic-state,mtu
```

vrouter-name	nic	ip	vnet	vlan	vlan-type	nic-state	mtu
vr1	eth0.100	192.168.99.13/24	vn1	100	private	up	1500

```
CLI (network-admin@switch) > vrouter-ping vrouter-name vr1 host-ip
192.168.99.13
```

```
PING 192.168.99.13 (192.168.99.13) 56(84) bytes of data.
64 bytes from 192.168.99.13: icmp_seq=1 ttl=64 time=0.066 ms
64 bytes from 192.168.99.13: icmp_seq=2 ttl=64 time=0.071 ms
64 bytes from 192.168.99.13: icmp_seq=3 ttl=64 time=0.063 ms
^C
--- 192.168.99.13 ping statistics ---
```

## Specifying the Type of vNET Interface

---

The `mgmt`, `data`, and `span` keywords used in different commands specify the path used to connect to the network service. For example, to specify an out-of-band connection to a management interface of a vNET, the interface is specified using the `mgmt` keyword. If in-band access to that management interface of the vNET is required, then the `data` or `span` keywords are used in the specific command.

The keywords, `data` and `span`, are essentially equivalent but apply to two separate paths. To maximize throughput between the server and the switch components, it is recommended to use both. The `data` keyword applies to port 65, and the `span` keyword applies to port 66.

Each vNET can have one or more isolating zones and network services are applied to each zone. Network services have their own zone or share the zone with the vNET manager which is the zone that the vNET user logs into to manage the vNET. In shared zones, the network interfaces are available to all network services in the shared zones, regardless of the service that created the network interface.

**Note:** This is an important concept as you can use service commands such as `v1b-interface-add` to add an interface or you can use `vnet-manager-interface-add` to add interfaces to a vNET. If you want the service to be specific to a vNET as a dedicated service, then add the interfaces using the `service-interface-add` commands.



## Configuring vNET High Availability (HA)

vNET HA provides high availability for switch access through a vNET manager. The vNET manager is a zone, typically with one IP interface, which allows a vNET administrator to log into and administer a vNET using the CLI or a REST API.

Netvisor ONE provides HA functionality by allowing you to create multiple vNET managers. Netvisor ONE enables access to the vNET managers when you add either standard or VRRP VIP interfaces to the vNET managers.

Without vNET HA, vNET administrators have only a single point of access where the vNET manager zone runs on a particular switch. If that switch fails, the administrator cannot log into the fabric and administer the vNET.

This feature provides the following:

- Create or delete a vNET manager zone.
- The `vnet-manager-interface` command accepts VRRP interfaces. This allows you to create VRRP interfaces on vNET manager zones
- The `vnet-create` command provides an option to not create a vNET manager zone when creating a vNET.
- Copy and paste SSH host keys used on different vNET managers. This is needed when using a VRRP VIP to access the vNET managers to avoid SSH host key violations if the VIP fails over to the standby vNET manager. Install these keys on the client machine used to connect to the vNET managers.

### Creating a vNET Manager Zone

To create additional vNET manager zones, use the `vnet-manager-create` command.

```
CLI (network-admin@Leaf1) > vnet-manager-create name name-string vnet vnet-name [disable|enable][location fabric-node name][storage-pool storage-pool name]
```

<code>vnet-manager-create</code>	Creates a vNET manager
<code>name name-string</code>	Specify the name of service configuration.
<code>vnet vnet-name</code>	Specify the vNET assigned to the service.
any of the following options:	
<code>disable enable</code>	Specify to enable or disable the service.
<code>location fabric-node name</code>	Specify the location of the service.

---

<code>storage-pool storage-pool name</code>	Specify the storage pool assigned to the service.
---	---

---

## Deleting a vNET Manager Zone

To delete a vNET manager zone, use the `vnet-manager-delete` command.

```
CLI (network-admin@Leaf1) > vnet-manager-delete name name-string
```

Specify the name of service configuration.

## Copying and Pasting SSH Keys

To output SSH host keys to copy and paste to `~/ .ssh/known_hosts` file of the client host, use the `vnet-manager-ssh-host-key-show` command.

This allows you to SSH to any vNET manager zone and avoid issues with invalid key hosts.

```
CLI (network-admin@Leaf1) > vnet-manager-ssh-host-key-show [vnet vnet name]
```

---

<code>vnet-manager-ssh-host-key-show</code>	Displays the vNET Manager host keys to copy and past to: <code>~/ .ssh/known_hosts</code> .
<code>name name-string</code>	Displays the name of service configuration.

---

## vNET Manager Command Options

This feature uses a new option `[no-]create-vnet-mgr` which controls whether to create a vNET manager.

The default behavior is creating a vNET manager as this is the current behavior of creating a vNET manager as part of `vnet-create`.

```
CLI (network-admin@Leaf1) > no-create-vnet-mgr
```

---

<code>vnet-create</code>	Creates a virtual network (vNET)
<code>name name-string</code>	Specify the vNET name.
<code>scope local cluster fabric</code>	Specify the vNET scope as local, cluster, or fabric.
<code>create-vnet-mgr no-create-vnet-mgr</code>	Create or not create a vNET manager service.

---

## VRRP Interfaces Option

This feature now accepts options for VRRP interfaces. This allows you to create VRRP interfaces on vNET manager zones.

```
CLI (network-admin@Leaf1) > vnet-manager-interface-add vnet-manager-name
name-string [vrrp-id 0,,255][vrrp-primary vrrp-primary-string][vrrp-
```

priority 0..254][vrrp-adv-int 300..40950]

---

vnet-manager-interface-add	Adds an interface to a vNET manager.
vnet-manager-name <i>name-string</i>	Specify the name of service configuration.
vrrp-id 0..255	Specify the ID assigned to VRRP.
vrrp-primary vrrp-primary-string	Specify the VRRP primary interface.
vrrp-priority 0..254	Specify the VRRP priority for the interface.
vrrp-adv-int 300..40950	Specify the VRRP Advertisement Interval in milliseconds. The minimum interval is 300ms and the maximum interval is 40950ms. The default interval is 1000ms.

---

## Related Documentation

---

For further information on concepts mentioned in this section (such as Layer 2, Layer 3, etc.), refer to these sections of the Configuration Guides:

- [Configuring Switch Ports](#)
- [Configuring Layer 2 Features](#)
- [Configuring Layer 3 Features](#)
- [Setting up and Administering the Pluribus Fabric](#)
- [Configuring High Availability](#)
- [Configuring VXLAN](#)

## Overview Open Virtual Switch

---

Open vSwitch is a virtual switch that enables network automation, while supporting standard management interfaces and protocols, like NetFlow. Open vSwitch also supports distribution across multiple physical servers.

In an Open vSwitch implementation, a database server and a switch daemon are used. The OVSDB protocol is used in a control cluster, along with other managers and controllers, to supply configuration information to the switch database server. Controllers use OpenFlow to identify details of the packet flows through the switch. Each switch may receive directions from multiple managers and controllers, and each manager and controller can direct multiple switches.

---

- [Configuring OVSDB with Netvisor One](#)
  - [Using OpenSSL TLS Certificates for OVSDB and other Services](#)
  - [Open Virtual Switch Database \(OVSDB\) Error Reporting](#)
-

## Configuring OVSDb with Netvisor ONE

---

There are a number of steps required to configure OVSDb using Netvisor One.

### 1. Configure the vNET with the number of private VLANs, VXLANs, and managed ports:

```
CLI (network-admin@Leaf1) > vnet-create name name-string vlans vlan-range num-private-vlans integer vxlans vxlan-id managed-ports port-list
```

### 2. Configure the underlay network:

```
CLI (network-admin@Leaf1) > vrouter-create name-string vnet name-string router type hardware
```

```
CLI (network-admin@Leaf1) > vrouter-interface-add vrouter-name name-string vlan vlan-id ip ip-address netmask netmask
```

### 3. Configure the tunnel:

```
CLI (network-admin@Leaf1) > vnet-tunnel-network-add name name-string network ip-address netmask netmask
```

```
CLI (network-admin@Leaf1) > trunk-modify name name-string trunk-id trunk-id port port-list
```

### 4. Create the SSL/TLS certificate:

```
CLI (network-admin@Leaf1) > cert-create name name-string country country-string state state-string city city-string organization organization-string organizational-unit organizational-unit-string common-name common-name-string container zone name-string
```

### 5. Configure OVSDb:

```
CLI (network-admin@Leaf1) > openvswitch-create name name-string vnet name-string global-vtep tunnel-ip ip-address dedicated-service cert-name name-string
```

```
CLI (network-admin@Leaf1) > openvswitch-interface-add ovs-name name-string ip ip-address netmask netmask if data|mgmt vlan vlan-id
```

Configuring the interface as data or management depends on the location of the controller, on the data network or the management network.

If the controller is on a Layer 3 network several hops away, use openvswitch-modify to configure a gateway IP address. This is required in order for the configuration to work properly.

## 6. Add the hardware VTEP manager:

```
CLI (network-admin@Leaf1) > opensvswitch-hwvtep-manager-add name name-string manager-type odl|nsx connection-method ssl ip ip-address username username-string password password-string port port-number
```

Netvisor automatically establishes the VXLAN tunnel between the local and remote hardware and software VTEPs.

If you connect to VMware NSX controllers, you must use SSL or TLS to securely connect with the hardware VTEP.

## Using OpenSSL TLS Certificates for OVSDB and other Services

---

This feature provides a common Transport Layer Socket (TLS) within Netvisor One that you can use for any service such as the Open vSwitch Database Management Protocol (OVSDB) or a Web service. TLS is needed for any SSL connection to a Netvisor One service. For OVSDB, it is needed to connect to a controller using SSL.

For HTTPS communication between a REST API client and the Tomcat application server which is running a switch, you need to configure and deploy a server certificate in a Tomcat server.

You can create one common certificate for all Netvisor One services or create multiple named certificates. Each service can use a different certificate identified by name or container name or zone.

The Certificate facility keeps track of certificate use by using various applications. It notifies the applications when a certificate is updated and it also prevents a certificate from being deleted if an application is using it.

There are two ways to generate certificates:

- Self-signed certificate
- Certificate signed by a Certificate Authority (CA)

### Self-signed Certificate

If you want to generate a self-signed certificate use the `cert-create` command. This command creates a server certificate and self-signs it.

### Certificate signed by a Certificate Authority (CA)

If you want to generate a certificate that is signed by a CA, follow these steps:

1. Create a certificate signing request.
2. Export the certificate signing request and send it to the CA administrator.
3. Import the certificate received from CA administrator against the right certificate signing request.
4. Import the intermediate root and root certificate to the switch, if not done already.

### CLI Commands

These commands allow you to manage TLS certificates. These commands are also available for REST API.

To create a server certificate that self-signs, use the `cert-create` command:

```
CLI (network-admin@Leaf1) > cert-create country country-string state state-string city city-string organization organization-string organizational-unit organization-unit-string common-name common-name-string name name-string [container/zone name]
```

---

<code>cert-create</code>	Creates a server certificate and self-sign.
--------------------------	---



---

<code>country</code> <i>country-string</i>	Specify a country name (two letter code).
<code>state</code> <i>state-string</i>	Specify a state or province name.
<code>city</code> <i>city-string</i>	Specify a city name.
<code>organization</code> <i>organization-string</i>	Specify an organization name.
<code>organizational-unit</code> <i>organizational-unit-string</i>	Specify an organizational unit name.
<code>common-name</code> <i>common-name-string</i>	Specify a common name.
<code>name</code> <i>name-string</i>	Specify a certificate name.
any of the following options:	
<code>container</code> <i>zone name</i>	Specify a certificate zone name.

---

To delete a certificate, use the `cert-delete` command:

```
CLI (network-admin@Leaf1) > cert-delete name name-string [container/zone name]
```

---

<code>cert-delete</code>	Deletes a certificate.
<code>name</code> <i>name-string</i>	Specify a country name (two letter code).
any of the following options:	
<code>container</code> <i>zone name</i>	Specify a certificate zone name.

---

To import a CA certificate file, use the `cert-import` command:

```
CLI (network-admin@Leaf1) > cert-import name name-string file-ca file-ca-string [container zone name][file-inter file-inter-string]
```

---

<code>cert-import</code>	Imports certificates from the SFTP directory.
<code>name</code> <i>name-string</i>	Specify a certificate name.
<code>file-ca</code> <i>file-ca-string</i>	Specify the name of CA certificate file.
<code>file-server</code> <i>file-server-string</i>	Specify the name of server certificate file.
any of the following options:	
<code>container</code> <i>zone name</i>	Specify a certificate zone name.
any of the following options:	
<code>file-inter</code> <i>file-inter-string</i>	Specify the name of intermediate CA certificate file.

---

To import a server certificate file, use the `cert-import` command:

```
CLI (network-admin@Leaf1) > cert-import name name-string file-server file-
server-string [container zone name][file-ca file-ca-string]file-inter file-
inter-string]
```

---

<code>cert-import</code>	Imports certificates from SFTP directory.
<code>name name-string</code>	Specify the certificate name.
<code>file-server file-server-string</code>	Specify the name of server certificate file.
at least one of the following options:	
<code>container zone name</code>	Specify a certificate zone name.
any of the following options:	
<code>file-ca file-ca-string</code>	Specify the name of the CA certificate file.
<code>file-inter file-inter-string</code>	Specify the name of the intermediate CA certificate file.

---

To create a certificate signing request, use the `cert-request-create` command:

```
CLI (network-admin@Leaf1) > cert-request-create name name-string
[container/zone name]
```

---

<code>cert-request-create</code>	Create a certificate signing request from an existing server certificate.
<code>name name-string</code>	Specify the certificate name.
at least one of the following options:	
<code>container zone name</code>	Specify a certificate zone container name.

---

To display a certificate signing request, use the `cert-request-show` command:

```
CLI (network-admin@Leaf1) > cert-request-show name name-string
[container/zone name]cert-request cert-request-name
```

```
-----BEGIN CERTIFICATE REQUEST-----
MIICnDCCAYQCAQEwVzELMAkGA1UEBhMCdXMxCzAJBgNVBAGMAmNhMQswCQYDVQQH
DAJtcdELMAkGA1UECgwCcGwxDALBgNVBASMBGVuZ2cxEjAQBgNVBAMMCXBsdXJp
YnVzMTCCASIwDQYJKoZIhvcNAQEBBQADggEPADCCAQoCggEBAMrE6Jowg0VKUw2M
NlL8vp1N8dYE/UL5pvu8FKYWgWg7tC2fjHunZCI0XmssFtZysQul/r9nk+edA5tt
0zIWRmqTB60wnWmzl6uGymeAsc9OSm0ZHFc9zZfUxKjRM/nldOri3Pw/rODbCjM9
qw05hsvZc/cl03ajYFrjlyMlKDIiPWltd1VTpc5TL6wCwnDM697Yb9oQ0cbLKTDl
w5AjQSgJK29rLul8ptAZXIUkeendpE4MCYrl6Hd+ziOJHXncj65MJyfANTZMrtGD
IJD3m+JsKZt882vMw3AZ3C9WEuE0OZrbabGBHqVKARik2qFhu2bGjlbuj/M6TOF5
Jj1WROUCAwEAAaAAMA0GCSqGSIB3DQEBBQUAA4IBAQCCh1YhXRNwkwmmw3FVH4H0Xi
rczy0FkyHkdSbIUif+6n3qgroRpBpcEdrx8fREyiw8hLUks9OcUlT+nSshsWIitI7
R5dcFlyo5HUVjqQQVmlSq3j4fM9XE8y8KRMZ3mfLXRTmuFPxbBuE3ZGj1BSLnBgK
ODqHF1gVa4u7l9mO3TRXczLQiAPaw38/kxEwkh4erJp4jjXf8K0h9JMGvYONYWeI
1PbiZpjIWDLNBg6sKqqrPAXEAjzGNMgNPIMXRepmEmnC/BaLVA04noZran8LRLNp
```

```
Id41o3TnlXiAodF/Mc7H5fIlhYf0YzWDSfz3PNufn6Dusu5M2ma7jtWlEdBW8huH
-----END CERTIFICATE REQUEST-----
```

To display certificates, use the `cert-show` command:

```
CLI (network-admin@Leaf1) > cert-show [cert-type ca|intermediate|server]
[subject subject-string] [issuer issuer-string] [serial-number serial-
number-number] [valid-from valid-from-string] [valid-to valid-to-string]
[country country-string] [state state-string] [city city-string]
[organization organization-string] [organizational-unit organizational-
unit-string] [common-name common-name-string] [ name name-string]
[container/zone name]
```

name	used-by	cert-type	container	subject
cert3		ca		/C=us/ST=ca/L=mp/O=pl/OU=engg/CN=pluribus1
cert3		server		/C=us/ST=ca/L=mp/O=pl/OU=engg/CN=pluribus1
cert1 ovs		ca		/C=US/ST=CA/L=PA/O=ovs/OU=ou/CN=Pluribus
cert1 ovs		server		/C=US/ST=CA/L=PA/O=ovs/OU=ou/CN=Pluribus

## Configuring OpenvSwitch for Certificates

The following `openvswitch-create` and `openvswitch-modify` command options allow you to specify a certificate name when creating an OpenvSwitch configuration.

```
CLI (network-admin@Leaf1) > openvswitch-create name name-string [cert-name
cert-name string[ [ca-cert-name ca-cert-name-string] [cert-location none|
global|container]
```

<code>openvswitch create</code>	Create an OpenvSwitch configuration.
any of the following options:	
<code>cert-name cert-name-string</code>	Specify the certificate name for SSL connections.
<code>ca-cert-name ca-cert-name-string</code>	Specify the CA Certificate name for SSL connection.
<code>cert-location none global container</code>	Specify the certificate location - global or within the container.

```
CLI (network-admin@Leaf1) > openvswitch-modify name name-string [cert-name
cert-name string[ [ca-cert-name ca-cert-name-string] [cert-location none|
global|container]
```

<code>cert-name cert-name-string</code>	Specify the certificate name for SSL connections.
<code>ca-cert-name ca-cert-name-string</code>	Specify the certificate name for SSL connections.
<code>cert-location none global container</code>	Specify the certificate location - global or within the container.



## Open Virtual Switch Database (OVSDB) Error Reporting

---

Netvisor One now supports an error-reporting mechanism for OVSDB and VTEPs. When an error occurs, Netvisor One sends a schema change to the OVSDB controller.

As more functionality is added to Netvisor One for OVSDB, OVSDB error reporting adds new errors to support the new functionality.

## Open Source Acknowledgments

---

The Pluribus Networks Netvisor One Command Line Interface (CLI) used the following Open Source Software:

### **bean.js**

(bean.js):

bean.js - copyright Jacob Thornton 2011

<https://github.com/fat/bean>

MIT License

special thanks to:

dean edwards: <http://dean.edwards.name/>

dperini: <https://github.com/dperini/nwevents>

the entire mootools team: [github.com/mootools/mootools-core](https://github.com/mootools/mootools-core)

Bonzo: DOM Utility (c) Dustin Diaz 2011

<https://github.com/ded/bonzo>

License MIT

### **d3v2**

(d3v2):

Copyright (c) 2012, Michael Bostock

All rights reserved.

Redistribution and use in source and binary forms, with or without modification, are permitted provided that the following conditions are met:

Redistributions of source code must retain the above copyright notice, this list of conditions and the following disclaimer.

Redistributions in binary form must reproduce the above copyright notice, this list of conditions and the following disclaimer in the documentation and/or other materials provided with the distribution.

The name Michael Bostock may not be used to endorse or promote products derived from this software without specific prior written permission.

THIS SOFTWARE IS PROVIDED BY THE COPYRIGHT HOLDERS AND CONTRIBUTORS "AS IS" AND ANY EXPRESS OR IMPLIED WARRANTIES, INCLUDING, BUT NOT LIMITED TO, THE IMPLIED WARRANTIES OF MERCHANTABILITY AND FITNESS FOR A PARTICULAR PURPOSE ARE DISCLAIMED. IN NO EVENT SHALL MICHAEL BOSTOCK BE LIABLE FOR ANY DIRECT, INDIRECT, INCIDENTAL, SPECIAL, EXEMPLARY, OR CONSEQUENTIAL DAMAGES (INCLUDING, BUT NOT LIMITED TO, PROCUREMENT OF SUBSTITUTE GOODS OR SERVICES; LOSS OF USE, DATA, OR PROFITS; OR BUSINESS INTERRUPTION) HOWEVER CAUSED AND ON ANY THEORY OF LIABILITY, WHETHER IN CONTRACT, STRICT LIABILITY, OR TORT (INCLUDING NEGLIGENCE OR OTHERWISE) ARISING IN ANY WAY OUT OF THE USE OF THIS SOFTWARE, EVEN IF ADVISED OF THE POSSIBILITY OF SUCH DAMAGE.

### **d3v2**

(d3v2):

## TERMS OF USE - EASING EQUATIONS

Open source under the BSD License.  
Copyright 2001 Robert Penner  
All rights reserved.

Redistribution and use in source and binary forms, with or without modification, are permitted provided that the following conditions are met:

Redistributions of source code must retain the above copyright notice, this list of conditions and the following disclaimer.

Redistributions in binary form must reproduce the above copyright notice, this list of conditions and the following disclaimer in the documentation \* and/or other materials provided with the distribution.

Neither the name of the author nor the names of contributors may be used to endorse or promote products derived from this software without specific \* prior written permission.

THIS SOFTWARE IS PROVIDED BY THE COPYRIGHT HOLDERS AND CONTRIBUTORS "AS IS"\* AND ANY EXPRESS OR IMPLIED WARRANTIES, INCLUDING, BUT NOT LIMITED TO, THE IMPLIED WARRANTIES OF MERCHANTABILITY AND FITNESS FOR A PARTICULAR PURPOSE ARE DISCLAIMED. IN NO EVENT SHALL THE COPYRIGHT OWNER OR CONTRIBUTORS BE LIABLE FOR ANY DIRECT, INDIRECT, INCIDENTAL, SPECIAL, EXEMPLARY, OR CONSEQUENTIAL DAMAGES (INCLUDING, BUT NOT LIMITED TO, PROCUREMENT OF SUBSTITUTE GOODS OR SERVICES; LOSS OF USE, DATA, OR PROFITS; OR BUSINESS INTERRUPTION) HOWEVER CAUSED AND ON ANY THEORY OF LIABILITY, WHETHER IN CONTRACT, STRICT LIABILITY, OR TORT (INCLUDING NEGLIGENCE OR OTHERWISE) ARISING IN ANY WAY OUT OF THE USE OF THIS SOFTWARE, EVEN IF ADVISED OF THE POSSIBILITY OF SUCH DAMAGE.

## DataTables

(DataTables):

@summary DataTables  
@description Paginate, search and sort HTML tables  
@version 1.9.4  
@file jquery.dataTables.js  
@author Allan Jardine ([www.sprymedia.co.uk](http://www.sprymedia.co.uk))  
@contact [www.sprymedia.co.uk/contact](http://www.sprymedia.co.uk/contact)  
@copyright Copyright 2008-2012 Allan Jardine, all rights reserved.

This source file is free software, under either the GPL v2 license or a BSD style license, available at:

[http://datatables.net/license\\_gpl2](http://datatables.net/license_gpl2)

[http://datatables.net/license\\_bsd](http://datatables.net/license_bsd)

This source file is distributed in the hope that it will be useful, but WITHOUT ANY WARRANTY; without even the implied warranty of MERCHANTABILITY or FITNESS FOR A PARTICULAR PURPOSE. See the license files for details.

For details please refer to: <http://www.datatables.net>

## **(Envision)**

Envision.js

(c) 2012 Carl Sutherland, Humble Software

Distributed under the MIT License

Source: <http://www.github.com/HumbleSoftware/envisionjs>

Homepage: <http://www.humblesoftware.com/envision>

## **excanvas**

(excanvas.js):

Filament Group modification note:

This version of excanvas is modified to support lazy loading of this file. More info here:

<http://pipwerks.com/2009/03/12/lazy-loading-excanvasjs/>

Copyright 2006 Google Inc.

Licensed under the Apache License, Version 2.0 (the "License");

you may not use this file except in compliance with the License.

You may obtain a copy of the License at: <http://www.apache.org/licenses/LICENSE-2.0>

Unless required by applicable law or agreed to in writing, software distributed under the License is distributed on an "AS IS" BASIS, WITHOUT WARRANTIES OR CONDITIONS OF ANY KIND, either express or implied.

See the License for the specific language governing permissions and limitations under the License.

## **FloodLight**

Copyright 2012, Big Switch Networks, Inc.

Originally created by David Erickson, Stanford University

Licensed under the Apache License, Version 2.0 (the "License"); you may not use this file except in compliance with the License. You may obtain a copy of the License at:

<http://www.apache.org/licenses/LICENSE-2.0>

Unless required by applicable law or agreed to in writing, software distributed under the License is distributed on an "AS IS" BASIS, WITHOUT WARRANTIES OR CONDITIONS OF ANY KIND, either express or implied. See the License for the specific language governing permissions and limitations under the License

## **Flotr2**

Flotr2 (c) 2012 Carl Sutherland

MIT License

Special thanks to:

Flotr: <http://code.google.com/p/flotr/> (fork)

Flot: <https://github.com/flot/flot> (original fork)

## **g.Raphael**

g.Raphael 0.5 - Charting library, based on Raphael

Copyright (c) 2009 Dmitry Baranovskiy (<http://g.raphaeljs.com>)

Licensed under the MIT (<http://www.opensource.org/licenses/mit-license.php>) license.

## **GRUB**



## GRUB -- GRand Unified Bootloader

Copyright (C) 1999,2000,2001,2002,2004 Free Software Foundation, Inc.

This program is free software; you can redistribute it and/or modify it under the terms of the GNU General Public License as published by the Free Software Foundation; either version 2 of the License, or (at your option) any later version.

This program is distributed in the hope that it will be useful, but WITHOUT ANY WARRANTY; without even the implied warranty of MERCHANTABILITY or FITNESS FOR A PARTICULAR PURPOSE. See the GNU General Public License for more details.

You should have received a copy of the GNU General Public License along with this program; if not, write to the Free Software Foundation, Inc., 675 Mass Ave, Cambridge, MA 02139, USA.

## GSON

Copyright (C) 2008 Google Inc.

Licensed under the Apache License, Version 2.0 (the "License"); you may not use this file except in compliance with the License.

You may obtain a copy of the License at: <http://www.apache.org/licenses/LICENSE-2.0>

Unless required by applicable law or agreed to in writing, software distributed under the License is distributed on an "AS IS" BASIS, WITHOUT WARRANTIES OR CONDITIONS OF ANY KIND, either express or implied.

See the License for the specific language governing permissions and limitations under the License.

## DNS/DHCP

Copyright (c) 2004-2013 by Internet Systems Consortium, Inc. ("ISC")

Copyright (c) 1995-2003 by Internet Software Consortium

Permission to use, copy, modify, and distribute this software for any purpose with or without fee is hereby granted, provided that the above copyright notice and this permission notice appear in all copies.

THE SOFTWARE IS PROVIDED "AS IS" AND ISC DISCLAIMS ALL WARRANTIES WITH REGARD TO THIS SOFTWARE INCLUDING ALL IMPLIED WARRANTIES OF MERCHANTABILITY AND FITNESS. IN NO EVENT SHALL ISC BE LIABLE FOR ANY SPECIAL, DIRECT, INDIRECT, OR CONSEQUENTIAL DAMAGES OR ANY DAMAGES WHATSOEVER RESULTING FROM LOSS OF USE, DATA OR PROFITS, WHETHER IN AN ACTION OF CONTRACT, NEGLIGENCE OR OTHER TORTIOUS ACTION, ARISING OUT OF OR IN CONNECTION WITH THE USE OR PERFORMANCE OF THIS SOFTWARE.

Internet Systems Consortium, Inc.

950 Charter Street

Redwood City, CA 94063

<[info@isc.org](mailto:info@isc.org)>

<https://www.isc.org>

## JIT

Copyright (c) 2011 Sencha Inc. - Author: Nicolas Garcia Belmonte (<http://philogb.github.com/>)

Permission is hereby granted, free of charge, to any person obtaining a copy of this software and associated documentation files (the "Software"), to deal in the Software without restriction, including without limitation the rights to use, copy, modify, merge, publish, distribute, sublicense, and/or sell copies of the

Software, and to permit persons to whom the Software is furnished to do so, subject to the following conditions:

The above copyright notice and this permission notice shall be included in all copies or substantial portions of the Software.

THE SOFTWARE IS PROVIDED "AS IS", WITHOUT WARRANTY OF ANY KIND, EXPRESS OR IMPLIED, INCLUDING BUT NOT LIMITED TO THE WARRANTIES OF MERCHANTABILITY, FITNESS FOR A PARTICULAR PURPOSE AND NONINFRINGEMENT. IN NO EVENT SHALL THE AUTHORS OR COPYRIGHT HOLDERS BE LIABLE FOR ANY CLAIM, DAMAGES OR OTHER LIABILITY, WHETHER IN AN ACTION OF CONTRACT, TORT OR OTHERWISE, ARISING FROM, OUT OF OR IN CONNECTION WITH THE SOFTWARE OR THE USE OR OTHER DEALINGS IN THE SOFTWARE.

### **jquery.js**

jQuery JavaScript Library v1.4.3

<http://jquery.com/>

Copyright 2010, John Resig

Dual licensed under the MIT or GPL Version 2 licenses.

<http://jquery.org/license>

Includes Sizzle.js

<http://sizzlejs.com/>

Copyright 2010, The Dojo Foundation

Released under the MIT, BSD, and GPL Licenses.

Date: Thu Oct 14 23:10:06 2010 -0400

### **jQuery UI**

Copyright (c) 2011 Paul Bakaus, <http://jqueryui.com/>

This software consists of voluntary contributions made by many individuals (AUTHORS.txt,

<http://jqueryui.com/about>) For exact contribution history, see the revision history and logs, available at

<http://jquery-ui.googlecode.com/svn/>

Permission is hereby granted, free of charge, to any person obtaining a copy of this software and associated documentation files (the "Software"), to deal in the Software without restriction, including without limitation the rights to use, copy, modify, merge, publish, distribute, sublicense, and/or sell copies of the Software, and to permit persons to whom the Software is furnished to do so, subject to the following conditions:

The above copyright notice and this permission notice shall be included in all copies or substantial portions of the Software.

THE SOFTWARE IS PROVIDED "AS IS", WITHOUT WARRANTY OF ANY KIND, EXPRESS OR IMPLIED, INCLUDING BUT NOT LIMITED TO THE WARRANTIES OF MERCHANTABILITY, FITNESS FOR A PARTICULAR PURPOSE AND NONINFRINGEMENT. IN NO EVENT SHALL THE AUTHORS OR COPYRIGHT HOLDERS BE LIABLE FOR ANY CLAIM, DAMAGES OR OTHER LIABILITY, WHETHER IN AN ACTION OF CONTRACT, TORT OR OTHERWISE, ARISING FROM, OUT OF OR IN CONNECTION WITH THE SOFTWARE OR THE USE OR OTHER DEALINGS IN THE SOFTWARE.

### **jquery.cookie.js**

jQuery Cookie plugin

Copyright (c) 2010 Klaus Hartl (stilbuero.de)

Dual licensed under the MIT and GPL licenses:

<http://www.opensource.org/licenses/mit-license.php>

<http://www.gnu.org/licenses/gpl.html>

### **jquery.hotkeys.js**

jQuery Hotkeys Plugin

Copyright 2010, John Resig

Dual licensed under the MIT or GPL Version 2 licenses.

Based upon the plugin by Tzury Bar Yochay:

<http://github.com/tzuryby/hotkeys>

Original idea by:

Binny V A, [http://www.openjs.com/scripts/events/keyboard\\_shortcuts/](http://www.openjs.com/scripts/events/keyboard_shortcuts/)

### **jquery.validate.min.js**

Query Validation Plugin 1.8.1

<http://bassistance.de/jquery-plugins/jquery-plugin-validation/>

<http://docs.jquery.com/Plugins/Validation>

Copyright (c) 2006 - 2011 Jörn Zaefferer

Dual licensed under the MIT and GPL licenses:

<http://www.opensource.org/licenses/mit-license.php>

<http://www.gnu.org/licenses/gpl.html>

### **JSTL**

DO NOT ALTER OR REMOVE COPYRIGHT NOTICES OR THIS HEADER.

Copyright (c) 1997-2010 Oracle and/or its affiliates. All rights reserved.

The contents of this file are subject to the terms of either the GNU General Public License Version 2 only ("GPL") or the Common Development and Distribution License("CDDL") (collectively, the "License"). You

may not use this file except in compliance with the License. You can obtain a copy of the License at [https://glassfish.dev.java.net/public/CDDL+GPL\\_1\\_1.html](https://glassfish.dev.java.net/public/CDDL+GPL_1_1.html) or packager/legal/LICENSE.txt. See the License for the specific language governing permissions and limitations under the License.

When distributing the software, include this License Header Notice in each file and include the License file at packager/legal/LICENSE.txt.

GPL Classpath Exception:

Oracle designates this particular file as subject to the "Classpath" exception as provided by Oracle in the GPL Version 2 section of the License file that accompanied this code.

Modifications:

If applicable, add the following below the License Header, with the fields enclosed by brackets [] replaced by your own identifying information:

"Portions Copyright [year] [name of copyright owner]"

Contributor(s):

If you wish your version of this file to be governed by only the CDDL or only the GPL Version 2, indicate your decision by adding "[Contributor] elects to include this software in this distribution under the [CDDL or GPL Version 2] license." If you don't indicate a single choice of license, a recipient has the option to distribute your version of this file under either the CDDL, the GPL Version 2 or to extend the choice of license to its

licensees as provided above. However, if you add GPL Version 2 code and therefore, elected the GPL Version 2 license, then the option applies only if the new code is made subject to such option by the copyright holder.

This file incorporates work covered by the following copyright and permission notice:

Copyright 2004 The Apache Software Foundation

Licensed under the Apache License, Version 2.0 (the "License");

you may not use this file except in compliance with the License.

You may obtain a copy of the License at: <http://www.apache.org/licenses/LICENSE-2.0>

Unless required by applicable law or agreed to in writing, software distributed under the License is distributed on an "AS IS" BASIS, WITHOUT WARRANTIES OR CONDITIONS OF ANY KIND, either express or implied.

See the License for the specific language governing permissions and limitations under the License.

## **jstree**

jsTree 1.0-rc3

<http://jstree.com/>

Copyright (c) 2010 Ivan Bozhanov (vakata.com)

Licensed same as jquery - under the terms of either the MIT License or the GPL Version 2 License

<http://www.opensource.org/licenses/mit-license.php>

<http://www.gnu.org/licenses/gpl.html>

\$Date: 2011-02-09 01:17:14 +0200 (ср, 09 ф евр 2011) \$

\$Revision: 236 \$

Start Copyright text (libedit 3.0)

Copyright (c) 1992, 1993

The Regents of the University of California. All rights reserved.

This code is derived from software contributed to Berkeley by Christos Zoulas of Cornell University.

Redistribution and use in source and binary forms, with or without modification, are permitted provided that the following conditions are met:

1. Redistributions of source code must retain the above copyright notice, this list of conditions and the following disclaimer.
2. Redistributions in binary form must reproduce the above copyright notice, this list of conditions and the following disclaimer in the documentation and/or other materials provided with the distribution.
3. Neither the name of the University nor the names of its contributors may be used to endorse or promote products derived from this software without specific prior written permission.

THIS SOFTWARE IS PROVIDED BY THE REGENTS AND CONTRIBUTORS "AS IS" AND ANY EXPRESS OR IMPLIED WARRANTIES, INCLUDING, BUT NOT LIMITED TO, THE IMPLIED WARRANTIES OF MERCHANTABILITY AND FITNESS FOR A PARTICULAR PURPOSE ARE DISCLAIMED. IN NO EVENT SHALL THE REGENTS OR CONTRIBUTORS BE LIABLE FOR ANY DIRECT, INDIRECT, INCIDENTAL, SPECIAL, EXEMPLARY, OR CONSEQUENTIAL DAMAGES (INCLUDING, BUT NOT LIMITED TO, PROCUREMENT OF SUBSTITUTE GOODS OR SERVICES; LOSS OF USE, DATA, OR PROFITS; OR BUSINESS INTERRUPTION) HOWEVER CAUSED AND ON ANY THEORY OF LIABILITY, WHETHER IN CONTRACT, STRICT LIABILITY, OR TORT (INCLUDING NEGLIGENCE OR OTHERWISE) ARISING IN ANY WAY OUT OF THE USE OF THIS SOFTWARE, EVEN IF ADVISED OF THE POSSIBILITY OF SUCH DAMAGE.

## **log4j**

Licensed to the Apache Software Foundation (ASF) under one or more contributor license agreements. See the NOTICE file distributed with this work for additional information regarding copyright ownership. The ASF licenses this file to You under the Apache License, Version 2.0 (the "License"); you may not use this file except in compliance with the License. You may obtain a copy of the License at:

<http://www.apache.org/licenses/LICENSE-2.0>

Unless required by applicable law or agreed to in writing, software distributed under the License is distributed on an "AS IS" BASIS, WITHOUT WARRANTIES OR CONDITIONS OF ANY KIND, either express or implied.

See the License for the specific language governing permissions and limitations under the License.

### **(pciutils-3.1.10):**

The PCI Utilities -- Declarations

Copyright (c) 1997--2008 Martin Mares <[mj@ucw.cz](mailto:mj@ucw.cz)>

Can be freely distributed and used under the terms of the GNU GPL.

### **qtip 2.0**

qTip2 - Pretty powerful tooltips - v2.0.0 - 2012-10-03

<http://craigsworks.com/projects/qtip2/>

Copyright (c) 2012 Craig Michael Thompson; Licensed MIT, GPL

### **raphael 2.1.0**

Raphaël 2.1.0 - JavaScript Vector Library

Copyright © 2008-2012 Dmitry Baranovskiy (<http://raphaeljs.com>)

Copyright © 2008-2012 Sencha Labs (<http://sencha.com>)

Licensed under the MIT (<http://raphaeljs.com/license.html>) license.

Copyright (c) 2013 Adobe Systems Incorporated. All rights reserved.

Licensed under the Apache License, Version 2.0 (the "License"); you may not use this file except in compliance with the License.

You may obtain a copy of the License at : <http://www.apache.org/licenses/LICENSE-2.0>

Unless required by applicable law or agreed to in writing, software distributed under the License is distributed on an "AS IS" BASIS, WITHOUT WARRANTIES OR CONDITIONS OF ANY KIND, either express or implied.

See the License for the specific language governing permissions and limitations under the License.

Eve 0.4.2 - JavaScript Events Library

Author Dmitry Baranovskiy (<http://dmitry.baranovskiy.com/>)

### **Rickshaw v1.1.2**

Adapted from <https://github.com/Jakobo/PTClass> \*/

Copyright (c) 2005-2010 Sam Stephenson

Permission is hereby granted, free of charge, to any person obtaining a copy of this software and associated documentation files (the "Software"), to deal in the Software without restriction, including without limitation the rights to use, copy, modify, merge, publish, distribute, sublicense, and/or sell copies of the Software, and to permit persons to whom the Software is furnished to do so, subject to the following conditions:

THE SOFTWARE IS PROVIDED "AS IS", WITHOUT WARRANTY OF ANY KIND, EXPRESS OR IMPLIED, INCLUDING BUT NOT LIMITED TO THE WARRANTIES OF MERCHANTABILITY, FITNESS FOR A PARTICULAR PURPOSE AND NONINFRINGEMENT. IN NO EVENT SHALL THE AUTHORS OR COPYRIGHT HOLDERS BE LIABLE FOR ANY CLAIM, DAMAGES OR OTHER LIABILITY, WHETHER IN AN ACTION OF CONTRACT, TORT OR OTHERWISE, ARISING FROM, OUT OF OR IN CONNECTION WITH THE SOFTWARE OR THE USE OR OTHER DEALINGS IN THE SOFTWARE.

Based on Alex Arnell's inheritance implementation.

section: Language

class Class

Manages Prototype's class-based OOP system.

Refer to Prototype's web site for a [tutorial on classes and inheritance](<http://prototypejs.org/learn/class-inheritance>).

### **science.js 1.7.0**

Copyright (c) 2011, Jason Davies

All rights reserved.

Redistribution and use in source and binary forms, with or without modification, are permitted provided that the following conditions are met:

Redistributions of source code must retain the above copyright notice, this list of conditions and the following disclaimer.

Redistributions in binary form must reproduce the above copyright notice, this list of conditions and the following disclaimer in the documentation and/or other materials provided with the distribution.

The name Jason Davies may not be used to endorse or promote products derived from this software without specific prior written permission.

THIS SOFTWARE IS PROVIDED BY THE COPYRIGHT HOLDERS AND CONTRIBUTORS "AS IS" AND ANY EXPRESS OR IMPLIED WARRANTIES, INCLUDING, BUT NOT LIMITED TO, THE IMPLIED WARRANTIES OF MERCHANTABILITY AND FITNESS FOR A PARTICULAR PURPOSE ARE DISCLAIMED. IN NO EVENT SHALL JASON DAVIES BE LIABLE FOR ANY DIRECT, INDIRECT, INCIDENTAL, SPECIAL, EXEMPLARY, OR CONSEQUENTIAL DAMAGES (INCLUDING, BUT NOT LIMITED TO, PROCUREMENT OF SUBSTITUTE GOODS OR SERVICES; LOSS OF USE, DATA, OR PROFITS; OR BUSINESS INTERRUPTION) HOWEVER CAUSED AND ON ANY THEORY OF LIABILITY, WHETHER IN CONTRACT, STRICT LIABILITY, OR TORT (INCLUDING NEGLIGENCE OR OTHERWISE) ARISING IN ANY WAY OUT OF THE USE OF THIS SOFTWARE, EVEN IF ADVISED OF THE POSSIBILITY OF SUCH DAMAGE.

### **sizzle**

Copyright (c) 2009, John Resig

All rights reserved.

Redistribution and use in source and binary forms, with or without modification, are permitted provided that the following conditions are met:



Redistributions of source code must retain the above copyright notice, this list of conditions and the following disclaimer.

Redistributions in binary form must reproduce the above copyright notice, this list of conditions and the following disclaimer in the documentation and/or other materials provided with the distribution.

Neither the name of the <organization> nor the names of its contributors may be used to endorse or promote products derived from this software without specific prior written permission.

THIS SOFTWARE IS PROVIDED BY John Resig "AS IS" AND ANY EXPRESS OR IMPLIED WARRANTIES, INCLUDING, BUT NOT LIMITED TO, THE IMPLIED WARRANTIES OF MERCHANTABILITY AND FITNESS FOR A PARTICULAR PURPOSE ARE DISCLAIMED. IN NO EVENT SHALL <copyright holder> BE LIABLE FOR ANY DIRECT, INDIRECT, INCIDENTAL, SPECIAL, EXEMPLARY, OR CONSEQUENTIAL DAMAGES (INCLUDING, BUT NOT LIMITED TO, PROCUREMENT OF SUBSTITUTE GOODS OR SERVICES; LOSS OF USE, DATA, OR PROFITS; OR BUSINESS INTERRUPTION) HOWEVER CAUSED AND ON ANY THEORY OF LIABILITY, WHETHER IN CONTRACT, STRICT LIABILITY, OR TORT (INCLUDING NEGLIGENCE OR OTHERWISE) ARISING IN ANY WAY OUT OF THE USE OF THIS SOFTWARE, EVEN IF ADVISED OF THE POSSIBILITY OF SUCH DAMAGE.

#### **tcl 8.5.9**

This software is copyrighted by the Regents of the University of California, Sun Microsystems, Inc., Scriptics Corporation, ActiveState Corporation and other parties. The following terms apply to all files associated with the software unless explicitly disclaimed in individual files.

The authors hereby grant permission to use, copy, modify, distribute, and license this software and its documentation for any purpose, provided that existing copyright notices are retained in all copies and that this notice is included verbatim in any distributions. No written agreement, license, or royalty fee is required for any of the authorized uses. Modifications to this software may be copyrighted by their authors and need not follow the licensing terms described here, provided that the new terms are clearly indicated on the first page of each file where they apply.

IN NO EVENT SHALL THE AUTHORS OR DISTRIBUTORS BE LIABLE TO ANY PARTY FOR DIRECT, INDIRECT, SPECIAL, INCIDENTAL, OR CONSEQUENTIAL DAMAGES ARISING OUT OF THE USE OF THIS SOFTWARE, ITS DOCUMENTATION, OR ANY DERIVATIVES THEREOF, EVEN IF THE AUTHORS HAVE BEEN ADVISED OF THE POSSIBILITY OF SUCH DAMAGE.

THE AUTHORS AND DISTRIBUTORS SPECIFICALLY DISCLAIM ANY WARRANTIES, INCLUDING, BUT NOT LIMITED TO, THE IMPLIED WARRANTIES OF MERCHANTABILITY, FITNESS FOR A PARTICULAR PURPOSE, AND NON-INFRINGEMENT. THIS SOFTWARE IS PROVIDED ON AN "AS IS" BASIS, AND THE AUTHORS AND DISTRIBUTORS HAVE NO OBLIGATION TO PROVIDE MAINTENANCE, SUPPORT, UPDATES, ENHANCEMENTS, OR MODIFICATIONS.

GOVERNMENT USE: If you are acquiring this software on behalf of the U.S. government, the Government shall have only "Restricted Rights" in the software and related documentation as defined in the Federal Acquisition Regulations (FARs) in Clause 52.227.19 (c) (2). If you are acquiring the software on behalf of the Department of Defense, the software shall be classified as "Commercial Computer Software" and the Government shall have only "Restricted Rights" as defined in Clause 252.227-7013 (b) (3) of DFARs.

Notwithstanding the foregoing, the authors grant the U.S. Government and others acting in its behalf permission to use and distribute the software in accordance with the terms specified in this license.

#### **tcllib 1.13**

This software is copyrighted by Ajuba Solutions and other parties.

The following terms apply to all files associated with the software unless explicitly disclaimed in individual files.

The authors hereby grant permission to use, copy, modify, distribute, and license this software and its documentation for any purpose, provided that existing copyright notices are retained in all copies and that this notice is included verbatim in any distributions. No written agreement, license, or royalty fee is required for any of the authorized uses.

Modifications to this software may be copyrighted by their authors and need not follow the licensing terms described here, provided that the new terms are clearly indicated on the first page of each file where they apply.

IN NO EVENT SHALL THE AUTHORS OR DISTRIBUTORS BE LIABLE TO ANY PARTY FOR DIRECT, INDIRECT, SPECIAL, INCIDENTAL, OR CONSEQUENTIAL DAMAGES ARISING OUT OF THE USE OF THIS SOFTWARE, ITS DOCUMENTATION, OR ANY DERIVATIVES THEREOF, EVEN IF THE AUTHORS HAVE BEEN ADVISED OF THE POSSIBILITY OF SUCH DAMAGE.

THE AUTHORS AND DISTRIBUTORS SPECIFICALLY DISCLAIM ANY WARRANTIES, INCLUDING, BUT NOT LIMITED TO, THE IMPLIED WARRANTIES OF MERCHANTABILITY, FITNESS FOR A PARTICULAR PURPOSE, AND NON-INFRINGEMENT. THIS SOFTWARE IS PROVIDED ON AN "AS IS" BASIS, AND THE AUTHORS AND DISTRIBUTORS HAVE NO OBLIGATION TO PROVIDE MAINTENANCE, SUPPORT, UPDATES, ENHANCEMENTS, OR MODIFICATIONS.

GOVERNMENT USE: If you are acquiring this software on behalf of the U.S. government, the Government shall have only "Restricted Rights" in the software and related documentation as defined in the Federal Acquisition Regulations (FARs) in Clause 52.227.19 (c) (2). If you are acquiring the software on behalf of the Department of Defense, the software shall be classified as "Commercial Computer Software" and the Government shall have only "Restricted Rights" as defined in Clause 252.227-7013 (c) (1) of DFARs.

Notwithstanding the foregoing, the authors grant the U.S. Government and others acting in its behalf permission to use and distribute the software in accordance with the terms specified in this license.

## **tcldreadline 2.1.0**

Copyright (c) 1998 - 2000, Johannes Zellner <[johannes@zellner.org](mailto:johannes@zellner.org)>

All rights reserved.

Redistribution and use in source and binary forms, with or without modification, are permitted provided that the following conditions are met:

Redistributions of source code must retain the above copyright notice, this list of conditions and the following disclaimer.

Redistributions in binary form must reproduce the above copyright notice, this list of conditions and the following disclaimer in the documentation and/or other materials provided with the distribution.

Neither the name of Johannes Zellner nor the names of contributors to this software may be used to endorse or promote products derived from this software without specific prior written permission.

THIS SOFTWARE IS PROVIDED BY THE COPYRIGHT HOLDERS AND CONTRIBUTORS "AS IS" AND ANY



EXPRESS OR IMPLIED WARRANTIES, INCLUDING, BUT NOT LIMITED TO, THE IMPLIED WARRANTIES OF MERCHANTABILITY AND FITNESS FOR A PARTICULAR PURPOSE ARE DISCLAIMED. IN NO EVENT SHALL THE REGENTS OR CONTRIBUTORS BE LIABLE FOR ANY DIRECT, INDIRECT, INCIDENTAL, SPECIAL, EXEMPLARY, OR CONSEQUENTIAL DAMAGES (INCLUDING, BUT NOT LIMITED TO, PROCUREMENT OF SUBSTITUTE GOODS OR SERVICES; LOSS OF USE, DATA, OR PROFITS; OR BUSINESS INTERRUPTION) HOWEVER CAUSED AND ON ANY THEORY OF LIABILITY, WHETHER IN CONTRACT, STRICT LIABILITY, OR TORT (INCLUDING NEGLIGENCE OR OTHERWISE) ARISING IN ANY WAY OUT OF THE USE OF THIS SOFTWARE, EVEN IF ADVISED OF THE POSSIBILITY OF SUCH DAMAGE.

## **UI Widgets**

### **The MIT License**

Copyright (c) 2010 Filament Group, Inc

Permission is hereby granted, free of charge, to any person obtaining a copy of this software and associated documentation files (the "Software"), to deal in the Software without restriction, including without limitation the rights to use, copy, modify, merge, publish, distribute, sublicense, and/or sell copies of the Software, and to permit persons to whom the Software is furnished to do so, subject to the following conditions:

The above copyright notice and this permission notice shall be included in all copies or substantial portions of the Software.

THE SOFTWARE IS PROVIDED "AS IS", WITHOUT WARRANTY OF ANY KIND, EXPRESS OR IMPLIED, INCLUDING BUT NOT LIMITED TO THE WARRANTIES OF MERCHANTABILITY, FITNESS FOR A PARTICULAR PURPOSE AND NONINFRINGEMENT. IN NO EVENT SHALL THE AUTHORS OR COPYRIGHT HOLDERS BE LIABLE FOR ANY CLAIM, DAMAGES OR OTHER LIABILITY, WHETHER IN AN ACTION OF CONTRACT, TORT OR OTHERWISE, ARISING FROM, OUT OF OR IN CONNECTION WITH THE SOFTWARE OR THE USE OR OTHER DEALINGS IN THE SOFTWARE.

## **Underscore.js v1.1.7**

### **Underscore.js 1.1.7**

(c) 2011 Jeremy Ashkenas, DocumentCloud Inc.

Underscore is freely distributable under the MIT license.

Portions of Underscore are inspired or borrowed from Prototype, Oliver Steele's Functional, and John Resig's Micro-Templating.

For all details and documentation: <http://documentcloud.github.com/underscore>

## **Underscore.js v1.1.7**

Xelerated version (hxps 2.5.3 and earlier):

Copyright 2008-2012 (c) Xelerated AB.

This program may be used and/or copied only with the written permission from Xelerated AB, or in accordance with the terms and conditions stipulated in the agreement/contract under which the program has been supplied.

All rights reserved.

## **Underscore.js v1.1.7**

Marvell version (hxps 2.6 and above):

(c), Copyright 2008-2013, Marvell International Ltd. (Marvell)

This code contains confidential information of Marvell.

No rights are granted herein under any patent, mask work right or copyright of Marvell or any third party. Marvell reserves the right at its sole discretion to request that this code be immediately returned to Marvell.

This code is provided "as is". Marvell makes no warranties, expressed, implied or otherwise, regarding its accuracy, completeness or performance.

### **Oracle Solaris Open Source Projects**

<http://solaris.java.net/license.html>

## About Pluribus Networks

---

Pluribus Networks delivers an open, controllerless software-defined network fabric for modern data centers, multi-site data centers and distributed cloud edge environments.

The Linux-based Netvisor® ONE operating system and the Adaptive Cloud Fabric™ have been purpose-built to deliver radically simplified networking and comprehensive visibility along with white box economics by leveraging hardware from our partners Celestica, Dell EMC, and Edgecore, as well as Pluribus' own Freedom™ Series of switches.

The Adaptive Cloud Fabric provides a fully automated underlay and virtualized overlay with comprehensive visibility and brownfield interoperability and is optimized to deliver rich and highly secure per-tenant services across data center sites with simple operations having no single point of failure.

Further simplifying network operations is Pluribus UNUM™, an agile, multi-functional web management portal that provides a rich graphical user interface to manage the Adaptive Cloud Fabric. UNUM has two key modules - UNUM Fabric Manager for provisioning and management of the fabric and UNUM Insight Analytics to quickly examine billions of flows traversing the fabric to ensure quality and performance.

Pluribus is deployed in more than 275 customers worldwide, including the 4G and 5G mobile cores of more than 75 Tier 1 service providers delivering mission-critical traffic across the data center for hundreds of millions of connected devices. Pluribus is networking, simplified.

For additional information contact Pluribus Networks at [info@pluribusnetworks.com](mailto:info@pluribusnetworks.com), or visit [www.pluribusnetworks.com](http://www.pluribusnetworks.com).

Follow us on Twitter [@pluribusnet](https://twitter.com/pluribusnet) or on LinkedIn at <https://www.linkedin.com/company/pluribus-networks/>.

### Corporate Headquarters

Pluribus Networks, Inc.  
5201 Great America Parkway, Suite 422  
Santa Clara, CA 95054  
1-855-438-8638 / +1-650-289-4717

### India Office

Pluribus Networks India Private Limited  
Indique Brigade Square, 4th Floor  
21, Cambridge Road  
Bangalore 560008

Document Version - June 2020