

MPLS Support on VPC-SI for CUPS

- Revision History, on page 1
- Feature Description, on page 1
- How it Works, on page 2
- Monitoring and Troubleshooting, on page 12

Revision History

Note Revision history details are not provided for features introduced before release 21.24.

Revision Details	Release
First introduced	Pre 21.24

Feature Description

In the existing platforms (VPC-DI, ASR 5500), the boxer supports MPLS, which uses the underlying dataplane forwarder to switch MPLS traffic. In ASR 5500, the NP4c network processor generates and processes MPLS traffic while in VPC-DI, the IFTask generates and processes MPLS traffic.

The MPLS Support on VPC-SI for CUPS feature enables MPLS support on VPC-SI (SI-CUPS), which uses VPP as the dataplane forwarder.

VPP supports and provides multiple dataplane features that include the MPLS stack as a separate graph node. VPP generates labeled packets and simultaneously processes incoming labeled packets. This helps differentiate between different customer VRFs to support a large number of corporate APNs having different addressing models and requirements.

The MPLS Support on VPC-SI for CUPS feature supports the following functionalities:

- Uses the VPP MPLS stack to send the MPLS labeled packet.
- Uses the VPP MPLS stack to process the incoming labeled MPLS packet.

- Supports all existing MPLS configuration (VPC-DI, ASR 5500) and provides feature parity with new deployments using VPC-SI CUPS.
- Supports VPPCTL CLI commands to display NHLFE and ILM tables that are in VPP for debugging and comparing values with boxer configuration.

How it Works

This section briefly describes how the MPLS Support on VPC-SI for CUPS works.

In the current CUPs architecture, VPP forwarder provides its own MPLS stack, which supports all the existing functionalities for MPLS packet processing. The VPP MPLS stack is configured with the appropriate Next-Hop Label Forwarding Entry (NHLFE) and incoming label map (ILM) tables. This helps generate the MPLS packet on the egress with the correct MPLS header. It also processes the incoming MPLS packet and switches this packet based on the incoming labels to the appropriate next hop table identifier (VRF context of the subscriber) based on the incoming label.

The MPLS solution supports the following scenarios:

- MPLS-CE Connected to PE
- VPC-SI as a PE

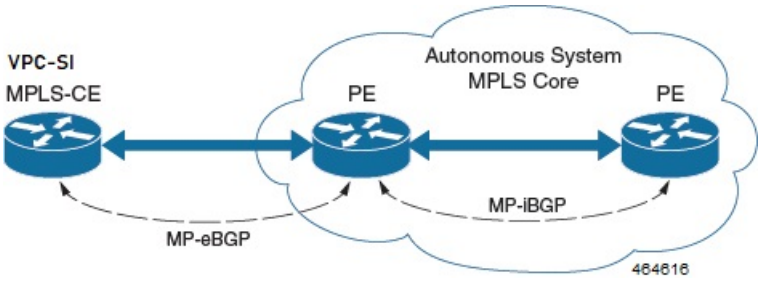
VPC-SI also supports VPNv6 as described in RFC 4659 – *BGP-MPLS IP Virtual Private Network (VPN) Extension for IPv6 VPN*.

MPLS-CE Connected to PE

The VPC-SI functions as an MPLS-CE (Customer Edge) network element connected to a Provider Edge (PE) Label Edge Router (LER), which in turn connects to the MPLS core as per RFC 4364.

The following figure illustrates the MPLS-CE to PE connection:

Figure 1: VPC-SI MPLS-CE to PE



The MPLS-CE functions like a PE router within its own Autonomous System (AS). It maintains Virtual Routing and Forwarding (VRF) routes and exchanges VPN route information with the PE through an MP-eBGP (Multi Protocol external BGP) session.

The PE is also configured with VRFs and exchanges VPN routes with other PEs in its AS through MP-iBGP (Multi Protocol internal BGP) connection and MPLS-CE through an MP-eBGP connection.

The EBGP connection allows the PE to change next-hop IP addresses and labels in the routes learnt from IBGP peers before advertising them to the MPLS-CE. The MPLS-CE uses only MP-eBGP to advertise and

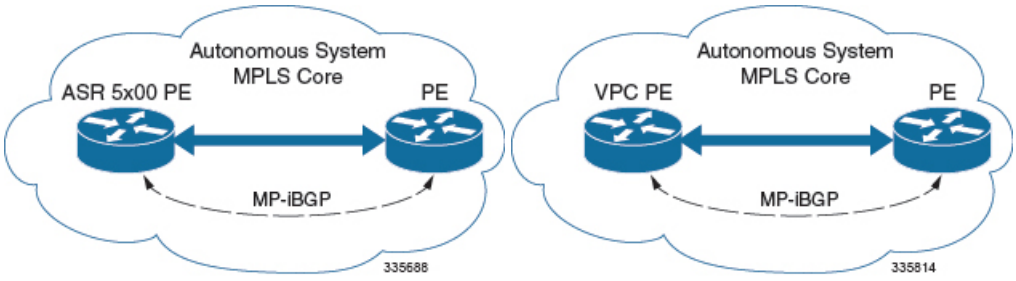
learn routes. Label Distribution Protocol (LDP) and Resource Reservation Protocol (RSVP) are not required because of direct-connect EBGP peering. The MPLS-CE pushes or pops a single label (learnt over the MP-eBGP connection) to or from the PE.

VPC-SI as a PE

Overview

In this scenario, the VPC-SI functions as a PE router sitting at the edge of the MPLS core. See the figure below.

Figure 2: VPC-SI as a PE



The VPC-SI eliminates the need for an ASBR or PE as shown in the first two scenarios. In this scenario, two main requirements are introduced: IBGP functionality and MPLS label distribution protocols.

The VPC-SI can be configured to add two labels:

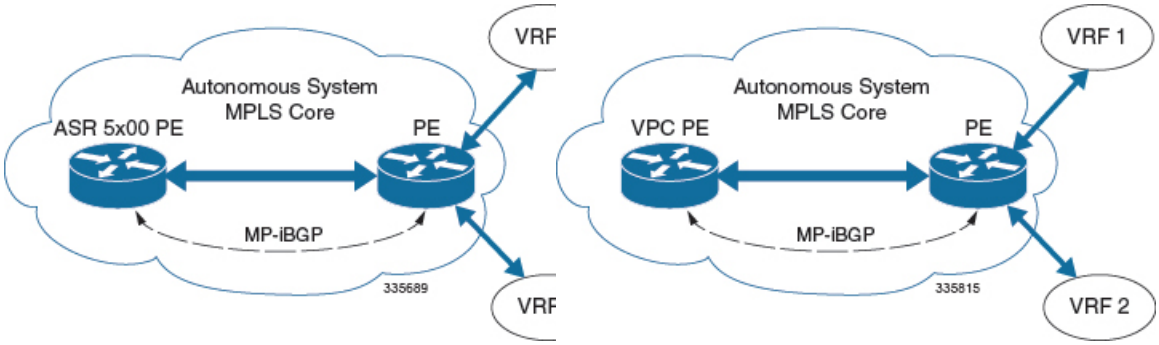
- an outer label learned from LDP or RSVP-TE (RSVP-Traffic Engineering)
- an inner label learned from MP-iBGP

This solution supports traffic engineering and QoS initiated via the VPC-SI.

Sample Configuration

In this example, VRFs are configured on the ASR 5500 PE and pools are associated with VRFs. The VPC-SI exchanges VPN routes with its IBGP peers (PE routers) and learns the MPLS paths to reach PEs via LDP. The VPC-SI forwards the packets to the next-hop with two labels – an inner label learned from PE and an outer label learned from the next hop IBGP neighbor.

Figure 3: Sample Configuration



```
mpls ip
  protocol ldp
    enable
  exit
exit

ip vrf vrf1
  mpls traffic-class copy
exit
ip vrf vrf2
  mpls traffic-class value 5
exit

router bgp 300
  ip vrf vrf1
    route-target export 300 1
      route-target import 300 1
      route-distinguisher 300 1
  exit
    ip vrf vrf2
      route-target export 300 2
        route-target import 300 2
        route-distinguisher 300 2
  exit

  router-id 209.165.201.1
  neighbor 209.165.200.225 remote-as 300
  neighbor 209.165.200.225 update-source node1_loopback

  address-family vpnv4
    neighbor 209.165.200.225 activate
      neighbor 209.165.200.225 send-community both
      neighbor 209.165.200.225 next-hop-self
  exit

  address-family ipv4 vrf vrf1
    redistribute connected
  exit

  address-family ipv4 vrf vrf2
    redistribute connected
exit

interface interface_to_internet
  ip address 209.165.200.224/27
  mpls ip
exit
router ospf
  network 209.165.201.0/27 area 209.165.201.5
exit
```

IPv6 Support for BGP MPLS VPNs

Overview

The VPC-SI supports VPNv6 as described in RFC 4659 – *BGP-MPLS IP Virtual Private Network (VPN) Extension for IPv6 VPN*.

An IPv6 VPN is connected over an IPv6 interface or sub-interface to the Service Provider (SP) backbone via a PE router. The site can be both IPv4 and IPv6 capable. Each VPNv6 has its own address space which means

a given address denotes different systems in different VPNs. This is achieved via a VPNv6 address-family which prepends a Route Distinguisher (RD) to the IP address.

A VPNv6 address is a 24-byte quantity beginning with an 8-byte RD and ending with a 16-byte IPv6 address. When a site is IPv4 and IPv6 capable, the same RD can be used for the advertisement of both IPv4 and IPv6 addresses.

The system appends RD to IPv6 routes and exchanges the labeled IPv6-RD using the VPNv6 address-family. The Address Family Identifier (AFI) and Subsequent Address Family Identifier (SAFI) fields for VPNv6 routes will be set to 2 and 128 respectively.

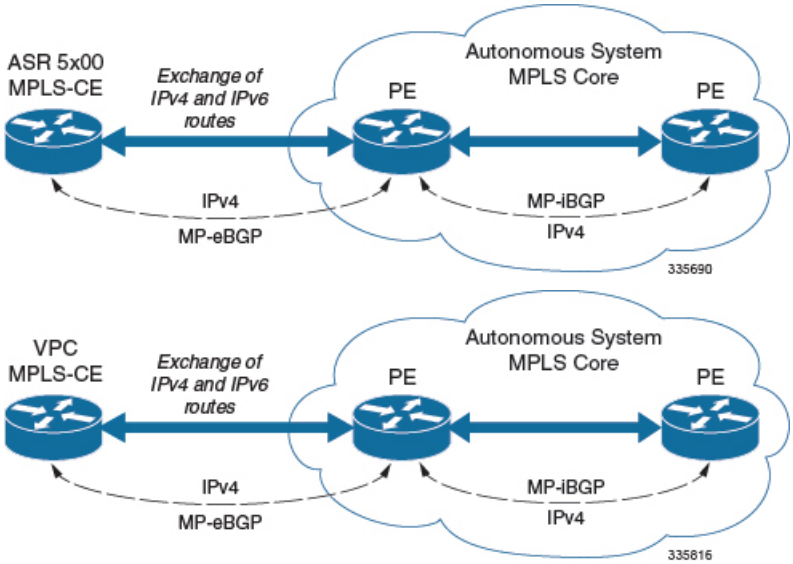
The IPv6 VPN traffic will be transported to the BGP speaker via IPv4 tunneling. The BGP speaker advertises to its peer a Next Hop Network Address field containing a VPN-IPv6 address whose 8-octet RD is set to zero and whose 16-octet IPv6 address is encoded as an IPv4-mapped IPv6 address (RFC 4291) containing the IPv4 address of the advertising router. It is assumed that only EBGP peering will be used to exchange VPNv6 routes.

Support for VPN-IPv6 assumes the following:

- Dual Stack (IPv4/IPv6) routing
- IPv6 pools in VRFs
- BGP peering over a directly connected IPv4 interface

See the figure below.

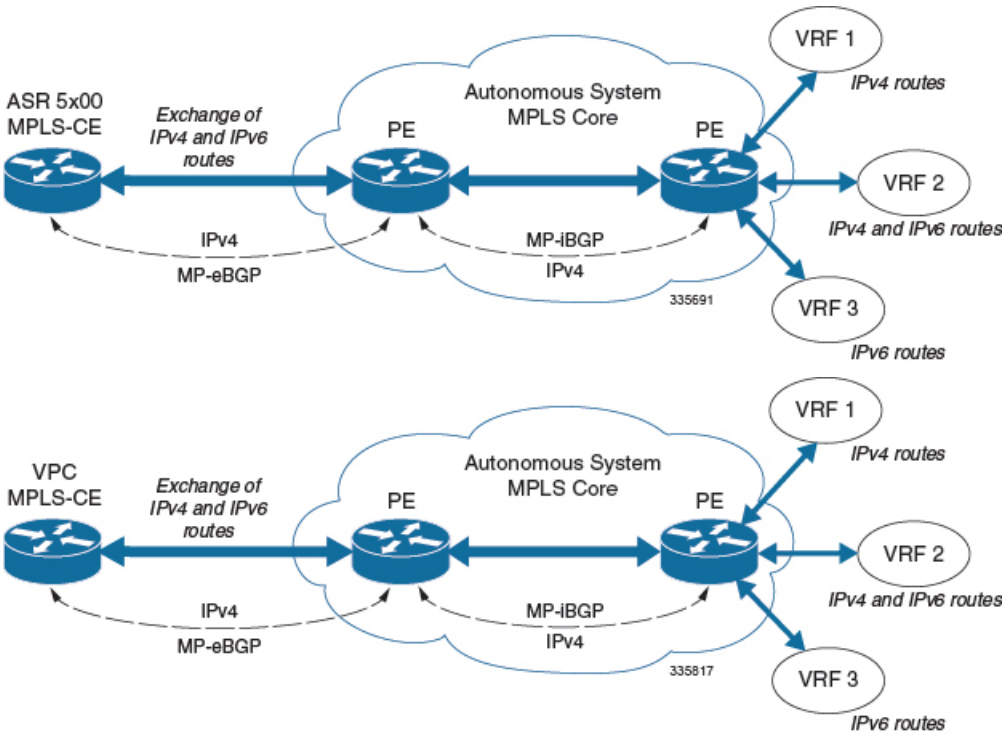
Figure 4: IPv6-RD Support for VPNv6



Sample Configuration

This example assumes three VRFs. VRF 1 has only IPv4 routes, VRF 2 has both IPv4 and IPv6 routes, and VRF 3 has only IPv6 routes.

Figure 5: VPNv6 Sample Configuration



Configure VRFs.

```
ip vrf vrf1
exit
ip vrf vrf2
exit
ip vrf vrf3
exit
```

Enable MPLS BGP forwarding.

```
mpls bgp forwarding
```

Configure pools.

```
ip pool vrf1-pool 209.165.200.230 255.255.255.224 private 0 vrf vrf1
exit
ip pool vrf2-pool 209.165.200.230 255.255.255.224 private 0 vrf vrf2

exit
ipv6 pool vrf2-v6pool prefix 2005:0101::/32 private 0 vrf vrf2
exit
ipv6 pool vrf3-v6pool prefix 2005:0101::/32 private 0 vrf vrf3
exit
```

Configure interfaces.

```
interface ce_interface_to_rtr
  ip address 209.165.200.226 255.255.255.224
exit
interface ce_v6_interface
  ip address 2009:0101:0101:0101::1/96
exit
interface ce_loopback loopback
  ip address 209.165.200.227 255.255.255.255
```

```
exit
interface vrf1-loop loopback
  ip vrf forwarding vrf1
  ip address 209.165.200.228 255.255.255.255
exit
interface vrf2-loop loopback
  ip vrf forwarding vrf2
  ip address 209.165.200.229 255.255.255.255
exit
interface vrf2-v6loop loopback
  ip vrf forwarding vrf2
  ip address 2005:0202:0101::1/128
exit
interface vrf3-v6loop loopback
  ip vrf forwarding vrf3
  ip address 2005:0303:0101::1/128
exit
```

Configure BGP along with address families and redistribution rules.

```
router bgp 800
  router-id 209.165.200.225
neighbor 209.165.200.240 remote-as 1003
  neighbor 209.165.200.240 activate
address-family vpnv4
  neighbor 209.165.200.240 activate
  neighbor 209.165.200.240 send-community both
exit
address-family vpnv6
  neighbor 209.165.200.240 activate
  neighbor 209.165.200.240 send-community both
exit
ip vrf vrf1
  route-distinguisher 800 1
  route-target export 800 1
  route-target import 800 1
exit
address-family ipv4 vrf vrf1
  redistribute connected
  redistribute static
exit
ip vrf vrf2
  route-distinguisher 800 2
  route-target export 800 2
  route-target import 800 2
exit
address-family ipv4 vrf vrf2
  redistribute connected
  redistribute static
exit
address-family ipv6 vrf vrf2
  redistribute connected
  redistribute static
exit
ip vrf vrf3
  route-distinguisher 800 3
  route-target export 800 3
  route-target import 800 3
exit
address-family ipv6 vrf   vrf3
  redistribute connected
  redistribute static
exit
```

Configure APNs.

```
apn walmart51.com
  selection-mode sent-by-ms
  accounting-mode none
  aaa group walmart-group
  authentication pap 1 chap 2 allow-noauth
  ip context-name Gi_ce
  ip address pool name vrf1-pool
exit
apn amazon51.com
  selection-mode sent-by-ms
  accounting-mode none
  aaa group amazon-group
  authentication pap 1 chap 2 allow-noauth
  ip context-name Gi_ce
  ip address pool name vrf2-pool
  ipv6 address prefix-pool vrf2-v6pool
exit
apn apple51.com
  selection-mode sent-by-ms
  accounting-mode none
  aaa group apple-group
  authentication pap 1 chap 2 allow-noauthip context-name Gi_ce
  ipv6 address prefix-pool vrf3-v6pool
exit
aaa-group amazon-group
  radius ip vrf vrf2
aaa group default
exit
gtpp group default
exit
ip igmp profile default
exit
```

Bind physical interfaces with the port.

VPN-Related CLI Commands

VPN-related features and functions are supported across several CLI command modes. The following tables identify commands associated with configuration and monitoring of VPN-related functions.

Table 1: VPN-Related Configuration Commands

CLI Mode	Command	Description
BGP Address-Family (IPv4/IPv6) Configuration Mode	**neighbor** *ip_address* **activate**	Enables the exchange of routing information with a peer router.
BGP Address-Family (IPv4/IPv6) Configuration Mode	**neighbor** *ip_address* **send community { both | extended | standard }**	Sends the community attributes to a peer router (neighbor).
BGP Address-Family (IPv4/IPv6) Configuration Mode	**redistribute connected**	Redistributes routes into BGP from another protocol as BGP neighbors.
BGP Address-Family (VPNv4) Configuration Mode	**neighbor** *ip_address* **activate**	Enables the exchange of routing information with a peer router.

CLI Mode	Command	Description
BGP Address-Family (VPNv4) Configuration Mode	**neighbor** *ip_address* **send community { both | extended | standard }**	Sends the extended-community attribute to a peer router. In VPN, route-distinguisher and route-target are encoded in the BGP extended-community. This command enables sending of BGP routes with extended community to a neighbor.
BGP Address-Family (VRF) Configuration Mode	**neighbor** *ip_address* **activate**	Enables the exchange of routing information with a peer router.
BGP Address-Family (VRF) Configuration Mode	**neighbor** *ip_address* **send community { both | extended | standard }**	Sends the extended-community attribute to a peer router. In VPN, route-distinguisher and route-target are encoded in the BGP extended-community. This command enables sending of BGP routes with extended community to a neighbor.
BGP Address-Family (VRF) Configuration Mode	**redistribute connected**	Redistributes routes into BGP from another protocol as BGP neighbors.
BGP Configuration Mode	**address-family { ipv4 vrf** *vrf_name* **| vpnv4 }**	Enables the exchange of IPv4 VRF routing information. There is a different mode for each address-family.
BGP Configuration Mode	**address-family { ipv6 vrf** *vrf_name* **| vpnv6 }**	Configures a VPNv6 address family and IPv6 VRF routing in BGP.
BGP Configuration Mode	**ip vrf** *vrf_name*	Adds a VRF to BGP and switches to the VRF Configuration mode to allow configuration of BGP attributes for the VRF.
BGP IP VRF Configuration Mode	**route-distinguisher {** *as_value* **|** *ip_address* **}** *rd_value*	Assigns a Route Distinguisher (RD) for the VRF. The RD value must be a unique value on the router for each VRF.
BGP IP VRF Configuration Mode	**route-target { both | import | export } {** *as_value* **|** *ip_address* **}** *rt_value*	Adds a list of import and export route-target extended communities to the VRF.

CLI Mode	Command	Description
Context Configuration Mode	**ip pool** *pool_name addr_range* **vrf** *vrf_name* **[mpls-label input** *inlabel1* **output** *outlabel1 outlabel2* **]**	Configures a pool into the specified VRF. This parameter must be specified with the Next-Hop parameter. *inlabel1* is the MPLS label that identifies inbound traffic destined for this pool. *outlabel1* and *outlabel2* specify the MPLS labels to be added to packets sent for subscribers from this pool.
Context Configuration Mode	**ip vrf** *vrf_name*	Creates a VRF and assigns a VRF-ID. A VRF is created in the router.
Context Configuration Mode	**ipv6 pool** *pool_name* **vrf** *vrf_name*	Associates the pool with that VRF. **Note:** By default the configured ipv6 pool will be associated with the global routing domain.
Context Configuration Mode	**mpls bgp forwarding**	Globally enables MPLS Border Gateway Protocol (BGP) forwarding.
Context Configuration Mode	**mpls exp** *value*	Sets the default behavior as Best Effort using a zero value in the 3-bit MPLS EXP header. This value applies to all the VRFs in the context. The default behavior is to copy the DSCP value of mobile subscriber traffic to the EXP header, if there is no explicit configuration for DSCP to EXP (via the **mpls map-dscp-to-exp dscp** *n* **exp** *m* command). **mpls exp** disables the default behavior and sets the EXP value to the configured *value*.
Context Configuration Mode	**mpls ip**	Globally enables the MPLS forwarding of IPv4 packets along normally routed paths.
Context Configuration Mode	**radius change-authorize-nas-ip ip_address** *ip_address* **{ encrypted | key }** *value* **port** *port_num* **mpls input** *inlabel* **output** *outlabel1 outlabel2*	Configures COA traffic to use the specified MPLS labels. *inlabel* identifies inbound COA traffic. *outlabel1* and *outlabel2* specify the MPLS labels to be added to the COA response. *outlabel1* is the inner output label; *outlabel2* is the outer output label.
Ethernet Interface Configuration Mode	**mpls ip**	Enables dynamic MPLS forwarding of IP packets on this interface.

CLI Mode	Command	Description
Exec Mode	**clear ip bgp peer**	Clears BGP sessions.
Exec Mode	**lsp-ping** *ip_prefix_FEC*	Checks MPLS Label-Switched Path (LSP) connectivity for the specified forwarding equivalence class (FEC). It must be followed by an IPv4 or IPv6 FEC prefix.
Exec Mode	**lsp-traceroute** *ip_prefix_FEC*	Discovers MPLS LSP routes that packets actually take when traveling to their destinations. It must be followed by an IPv4 or IPv6 FEC prefix.
IP VRF Context Configuration Mode	**mpls map-dscp-to-exp dscp** *dscp_bit_value* **exp** *exp_bit_value*	Maps the final differentiated services code point (DSCP) bit value in the IP packet header to the final Experimental (EXP) bit value in the MPLS header for incoming traffic.
IP VRF Context Configuration Mode	**mpls map-exp-to-dscp exp** *exp_bit_value* **dscp** *dscp_bit_value*	Maps the incoming EXP bit value in the MPLS header to the internal DSCP bit value in IP packet headers for outgoing traffic.
MPLS-IP Configuration Mode	**protocol ldp**	Creates the MPLS protocol family configuration modes, or configures an existing protocol and enters the MPLS-LDP Configuration Mode in the current context. This command configures the protocol parameters for the MPLS protocol family.
MPLS-LDP Configuration Mode	**advertise-labels { explicit-null | implicit-null }**	Configure advertisement of Implicit NULL or Explicit NULL label for all the prefixes advertised by the system in this context.
MPLS-LDP Configuration Mode	**discovery { hello { hello-interval** *seconds* **| hold-interval** *seconds* **} | transport-address** *ip_address* **}**	Configures the Label Distribution Protocol (LDP) neighbor discovery parameters.
MPLS-LDP Configuration Mode	**enable**	Enables Label Distribution Protocol (LDP).
MPLS-LDP Configuration Mode	**router-id** *ip_address*	Configures the LDP Router ID.
MPLS-LDP Configuration Mode	**session timers { hold-interval** *seconds* **| keepalive-interval** *seconds* **}**	Configures the LDP session parameters.

Table 2: VPN-Related Monitoring Commands

CLI Mode	Command	Description
Exec Mode show Commands	**show ip bgp neighbors**	Displays information regarding BGP neighbors.
Exec Mode show Commands	**show ip bgp vpnv4 { all | route-distinguisher | vrf }**	Displays all VPNv4 routing data, routing data for a VRF or a route-distinguisher.
Exec Mode show Commands	**show ip bgp vpnv6**	Displays contents of VPNv6 routing table.
Exec Mode show Commands	**show ip bgp vpnv6 { all | route-distinguisher | vrf }**	Displays all VPNv6 routing data, routing data for a VRF or a route-distinguisher.
Exec Mode show Commands	**show ip pool**	Displays pool details including the configured VRF.
Exec Mode show Commands	**show mpls cross-connect**	Displays MPLS cross-connect information. MPLS tunnel cross-connects between interfaces and Label-Switched Paths (LSPs) connect two distant interface circuits of the same type via MPLS tunnels that use LSPs as the conduit.
Exec Mode show Commands	**show mpls ftn [vrf** *vrf_name*	Displays MPLS FEC-to-NHLFE (FTN) table information.
Exec Mode show Commands	**show mpls ftn [vrf** *vrf_name* **]**	Displays contents of the MPLS FTN table for a specified VRF.
Exec Mode show Commands	**show mpls ilm**	Displays MPLS Incoming Label Map (ILM) table information.
Exec Mode show Commands	**show mpls ldp**	Displays the MPLS LDP information.
Exec Mode show Commands	**show mpls nexthop-label-forwarding-entry**	Displays MPLS Next-Hop Label Forwarding Entry (NHLFE) table information.

Monitoring and Troubleshooting

This section provides information regarding the CLI command to monitor and troubleshoot the feature.

Show Commands and Outputs

This section provides information regarding show commands and their outputs in support of this feature.

show mpls fn vpp

The output of this CLI command contains the following new field for the MPLS Support on VPC-SI for CUPS feature:

- vpp
  - all-vrf
  - summary
  - vrf



Note This new field enables viewing of the VPP dataplane values that are confiigured in the VPP dataplane forwarder. This show command is used for debugging along with the existing debug commands.