Inspur

CN12900 Series

INOS-CN Label Switching Configuration Guide

Inspur-Cisco Networking Technology Co.,Ltd. provides customers with comprehensive technical support and services. For any assistance, please contact our local office or company headquarters.
Website: http://www.inspur.com/
Technical Support Tel: 400-691-1766
Technical Support Email:icnt_service@inspur.com
Technical Document Support Email: icnt_service@inspur.com
Address: 1036 Langchao Road, Lixia District, Jinan City, Shandong Province
Postal code: 250101

Notice

Preface

Objectives

This guide describes main functions of the CN12900 Series. To have a quick grasp of the CN12900 Series, please read this manual carefully.

Versions

The following table lists the product versions related to this document.

Product name	Version
CN12900 Series	

Conventions

Symbol conventions

The symbols that may be found in this document are defined as follows.

Symbol	Description
Warning	Indicates a hazard with a medium or low level of risk which, if not avoided, could result in minor or moderate injury.
Caution	Indicates a potentially hazardous situation that, if not avoided, could cause equipment damage, data loss, and performance degradation, or unexpected results.
Note	Provides additional information to emphasize or supplement important points of the main text.
Tip	Indicates a tip that may help you solve a problem or save time.

General conventions

Convention	Description
Boldface	Names of files, directories, folders, and users are in **boldface**. For example, log in as user **root**.

Convention	Description
Italic	Book titles are in *italics*.
`Lucida Console`	Terminal display is in `Lucida Console`.

Command conventions

Convention	Description
Boldface	The keywords of a command line are in **boldface**.
Italic	Command arguments are in *italics*.
[]	Items (keywords or arguments) in square brackets [ ] are optional.
{ x | y | ... }	Alternative items are grouped in braces and separated by vertical bars. One is selected.
[x | y | ...]	Optional alternative items are grouped in square brackets and separated by vertical bars. One or none is selected.
{ x | y | ... } *	Alternative items are grouped in braces and separated by vertical bars. A minimum of one or a maximum of all can be selected.
[x | y | ...] *	The parameter before the & sign can be repeated 1 to n times.

GUI conventions

Convention	Description
Boldface	Buttons, menus, parameters, tabs, windows, and dialog titles are in **boldface**. For example, click **OK**.
>	Multi-level menus are in boldface and separated by the ">" signs. For example, choose **File** > **Create** > **Folder**.

Keyboard operation

Format	Description
Key	Press the key. For example, press **Enter** and press **Tab**.
Key 1+Key 2	Press the keys concurrently. For example, pressing **Ctrl+C** means the two keys should be pressed concurrently.
Key 1, Key 2	Press the keys in turn. For example, pressing **Alt**, **A** means the two keys should be pressed in turn.

Mouse operation

Action	Description
Click	Select and release the primary mouse button without moving the pointer.
Double-click	Press the primary mouse button twice continuously and quickly without moving the pointer.
Drag	Press and hold the primary mouse button and move the pointer to a certain position.

Change history

Updates between document versions are cumulative. Therefore, the latest document version contains all updates made to previous versions.

Issue 01 (2020-02-24)

Initial commercial release

Contents

Figure

Table

CHAPTER 1 New and Changed Information

This chapter provides release-specific information for each new and changed feature in the *Inspur CN12900 Series INOS-CN Label Switching Configuration Guide*.

1.1 New and Changed Information

This table summarizes the new and changed features for the *Inspur CN12900 Series INOS-CN Label Switching Configuration* Guide and tells you where they are documented.

Table 1 : New and Changed Features for Inspur INOS-CN

Feature	Description	Changed in Release	Where Documented
Initial Release		9.2(1i)	

CHAPTER 2 Configuring Static MPLS

This chapter contains information on how to configure static multiprotocol label switching (MPLS).

2.1 About Static MPLS

Generally, label switching routers (LSRs) use a label distribution protocol to dynamically learn the labels that they should use to label-switch packets. Examples of such protocols include:

- Label Distribution Protocol (LDP), the Internet Engineering Task Force (IETF) standard that is used to bind labels to network addresses
- Resource Reservation Protocol (RSVP), which is used to distribute labels for traffic engineering (TE)
- Border Gateway Protocol (BGP), which is used to distribute labels for MPLS virtual private networks (VPNs)

To use a learned label to label-switch packets, an LSR installs the label into its Label Forwarding Information Base (LFIB).

The static MPLS feature enables you to statically configure the following:

- The binding between a label and an IPv4 or IPv6 prefix
- The action corresponding to the binding between a label and an IPv4 or IPv6 prefix (label swap or pop)
- The contents of an LFIB cross-connect entry

2.1.1 Label Swap and Pop

As a labeled packet traverses the MPLS domain, the outermost label of the label stack is examined at each hop. Depending on the contents of the label, a swap or pop (dispose) operation is performed on the label stack. Forwarding decisions are made by performing an MPLS table lookup for the label carried in the packet header. The packet header does not need to be reevaluated during packet transit through the network. Because the label has a fixed length and is unstructured, the MPLS forwarding table lookup process is both straightforward and fast.

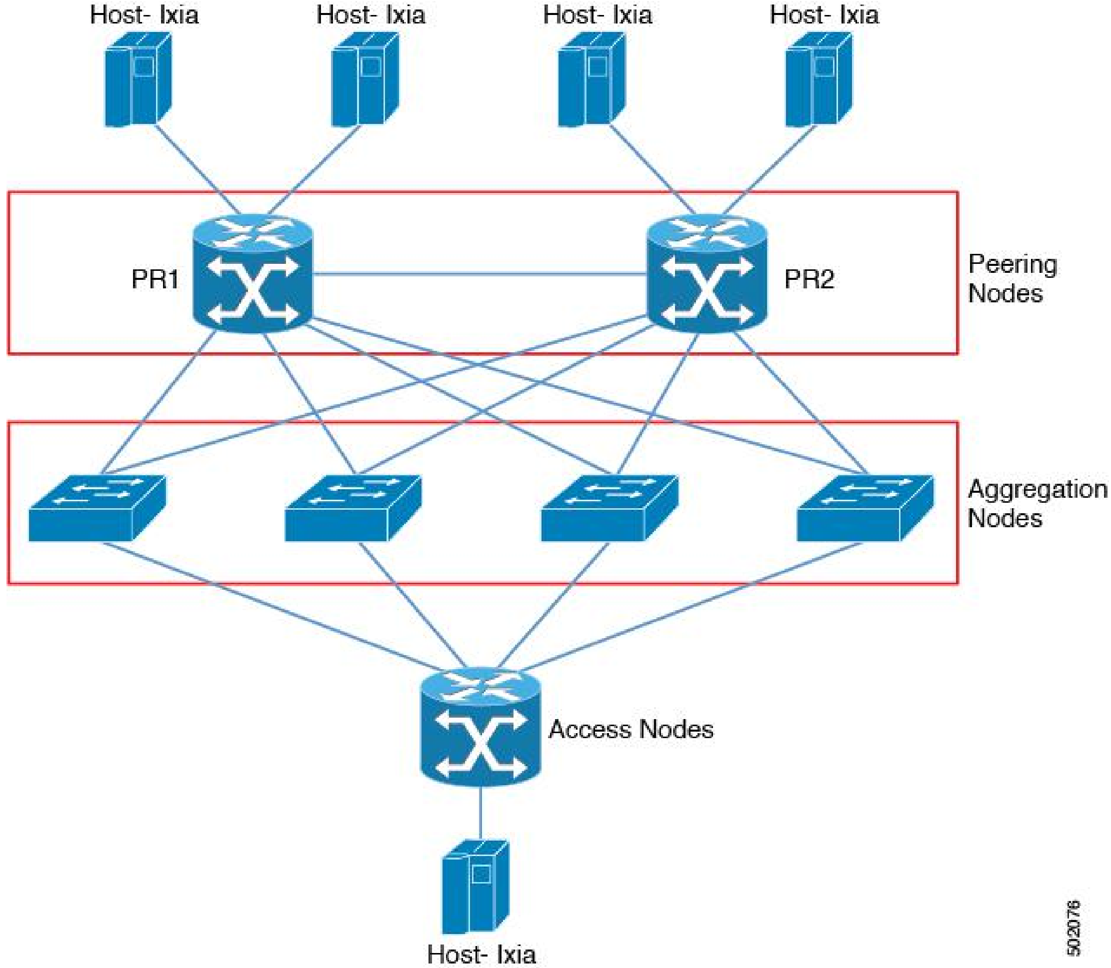
In a swap operation, the label is swapped with a new label, and the packet is forwarded to the next hop that is determined by the incoming label.

In a pop operation, the label is removed from the packet, which may reveal an inner label below. If the popped label was the last label on the label stack, the packet exits the MPLS domain. Typically, this process occurs at the egress LSR. A failure of the primary link in the aggregator reroutes the MPLS traffic to the backup link and results in a swap operation.

2.1.2 Static MPLS Topology

This diagram illustrates the static MPLS source routing topology. The access nodes perform the swap operation, and the aggregation nodes perform the pop operation for the primary path and the swap operation for the backup path.

Figure 1 : Static MPLS Topology



2.1.3 Benefits of Static MPLS

- Static bindings between labels and IPv4 or IPv6 prefixes can be configured to support MPLS hop-by-hop forwarding through neighbor routers that do not implement LDP label distribution.
- Static cross-connects can be configured to support MPLS label switched path (LSP) midpoints when neighbor routers do not implement either LDP or RSVP label distribution but do implement an MPLS forwarding path.

2.1.4 High Availability for Static MPLS

Inspur CN12900 Series switches support stateful switchovers (SSOs) for static MPLS. After an SSO, static MPLS returns to the state it was in previously.

Static MPLS supports zero traffic loss during SSO. MPLS static restarts are not supported.

2.2 Licensing Requirements for Static MPLS

The following table shows the licensing requirements for this feature:

Product	License Requirement

Inspur INOS-CN	Static MPLS requires no license. Any feature not included in a license package is bundled with the INOS-CN image and is provided at no extra charge to you.

2.3 Prerequisites for Static MPLS

Static MPLS has the following prerequisites:

- For Inspur CN12900 Series switches, you must configure the ACL TCAM region size for MPLS, save the configuration, and reload the switch. (For more information, see the "Using Templates to Configure ACL TCAM Region Sizes" and "Configuring ACL TCAM Region Sizes" sections in the Inspur CN12900 Series INOS-CN Security Configuration Guide).

2.4 Guidelines and Limitations for Static MPLS

Static MPLS has the following guidelines and limitations:

- Static MPLS is supported on Inspur CN12900 Series switches with the CN129-X636C-R, CN129-X636Q-R, or CN129-X6136YC-R line cards.
- Static MPLS, MPLS segment routing, and MPLS stripping cannot be enabled at the same time.
- Equal-cost multipath (ECMP) is not supported with label pop.
- Label pop and swap operations are supported, but label push operations are not.
- MPLS packets will be forwarded as long as the ingress label matches the configured label and the configured FEC (prefix) is in the routing table.

The device generally performs as a label switching router (LSR). It performs as a label edge router (LER) for penultimate hop popping, by installing the explicit null label as the out-label in the label FIB (LFIB) by an LSR before the packet is passed to an adjacent LER. This means that label switching router (LSR) unctions with 1 or more labels.

- Static MPLS supports up to 128 labels.
- The backup path is supported only for a single adjacency and not for ECMP.
- Inspur CN12900 Series switches support a limited backup path FRR convergence.
- The output for most of the MPLS commands can be generated in XML or JSON.
- VRFs, vPCs, and VXLAN are not supported with static MPLS.
- Subinterfaces are not supported for static MPLS.
- The Forwarding Equivalence Class (FEC) should exactly match routes in the routing table.
- When you configure fast reroute (backup), you can specify only the connected next hop (and not the recursive next hop) as the next-hop prefix in the backup configuration.
- When multiple FECs are sharing the backup (the same next-hop and interface), any change to the backup configuration requires a reconfiguration of all the other FECs that are sharing the backup configuration.
- When the backup path is active, the **show mpls switching labels** command will not show the out label/out interface/next hop and related statistics. You can use the **show forwarding mpls label** *label* **stats platform** command to check the statistics.
- If traffic ingresses or egresses on a non-default unit (where the default unit is unit0), the corresponding ULIB statistics will not be displayed in the output of the **show mpls switching labels** *low-label-value* [*high-label-value*] **detail** command. You can use the **show forwarding mpls label** *label* **stats platform** command to check the statistics.
- If the backup and primary paths are pointing to the same interface, the backup action swap takes precedence.
- Physical (Ethernet) and port channels are supported only for backup.

2.5 Configuring Static MPLS

2.5.1 Enabling Static MPLS

You must install and enable the MPLS feature set and then enable the MPLS static feature before you can configure MPLS static labels.

SUMMARY STEPS

1. **configure terminal**
2. [**no**] **install feature-set mpls**
3. [**no**] **feature-set mpls**
4. [**no**] **feature mpls static**
5. (Optional) **show feature-set**
6. (Optional) **show feature | inc mpls_static**

DETAILED STEPS

	Command or Action	Purpose

Step 1	**configure terminal**
	Example:
	`switch# configure terminal`
	`switch(config)#`
		Enters global configuration mode.

Step 2	[**no**] **install feature-set mpls** **Example:** `switch(config)# install feature-set mpls`	Installs the MPLS feature set. The **no** form of this command uninstalls the MPLS feature set.
Step 3	[**no**] **feature-set mpls** **Example:** `switch(config)# feature-set mpls`	Enables the MPLS feature set. The **no** form of this command disables the MPLS feature set.
Step 4	[**no**] **feature mpls static** **Example:** `switch(config)# feature mpls static`	Enables the static MPLS feature. The **no** form of this command disables the static MPLS feature.

Step 5	(Optional) **show feature-set**
	Example:
	`switch(config)# show feature-set     State`
	`Feature Set Name  ID`
	`--------------------  -------- --------`
	`mpls4    enabled`
		Displays the status of the MPLS feature set.

Step 6	(Optional) **show feature | inc mpls_static**
	Example:
	`switch(config)# show feature | inc`
	`mpls_static`
		Displays the status of static MPLS.

	Command or Action	Purpose
	`mpls_static 1 enabled`	

2.5.2 Reserving Labels for Static Assignment

You can reserve the labels that are to be statically assigned so that they are not dynamically assigned.

Before you begin

Ensure that the static MPLS feature is enabled.

SUMMARY STEPS

1. **configure terminal**
2. [**no**] **mpls label range** *min-value max-value* [**static** *min-static-value max-static-value*]
3. (Optional) **show mpls label range**
4. (Optional) **copy running-config startup-config**

DETAILED STEPS

	Command or Action	Purpose

Step 1	**configure terminal**
	Example:
	`switch# configure terminal`
	`switch(config)#`
		Enters global configuration mode.

Step 2	[**no**] **mpls label range** *min-value max-value* [**static**
	min-static-value max-static-value]
	Example:
	`switch(config)# mpls label range 17 99 static 100`
	`10000`
		Reserves a range of labels for static label assignment.
		The range for the minimum and maximum values is from
		16 to 471804.

Step 3	(Optional) **show mpls label range** **Example:** `switch(config)# show mpls label range`	Displays the label range that is configured for static MPLS.
Step 4	(Optional) **copy running-config startup-config** **Example:** `switch(config)# copy running-config startup-config`	Copies the running configuration to the startup configuration.

2.5.3 Configuring Static Label and Prefix Binding Using the Swap and Pop Operations

In a top-of-rack configuration, the outer label is swapped to the specified new label. The packet is forwarded to the next-hop address, which is auto-resolved by the new label.

In an aggregator configuration, the outer label is popped, and the packet with the remaining label is forwarded to the next-hop address. Pop operations are performed in the primary path, and swap operations are performed in the backup path.

Before you begin

Ensure that the static MPLS feature is enabled.

SUMMARY STEPS

1. **configure terminal**
2. **interface** *type slot/port*
3. [**no**] **mpls ip forwarding**
4. **mpls static configuration**
5. **address-family** {**ipv4** | **ipv6**} **unicast**
6. **local-label** *local-label-value* **prefix** *destination-prefix destination-prefix-mask*
7. **next-hop** {**auto-resolve** | *destination-ip-next-hop* **out-label implicit-null** | **backup** *local-egress-interface destination-ip-next-hop* **out-label** *output-label-value*}
8. (Optional) **copy running-config startup-config**

DETAILED STEPS

	Command or Action	Purpose

Step 1	**configure terminal**
	Example:
	`switch# configure terminal`
	`switch(config)#`
		Enters global configuration mode.

Step 2	**interface** *type slot/port*
	Example:
	`switch(config)# interface ethernet 2/2`
	`switch(config-if)#`
		Enters the interface configuration mode for the specified
		interface.

Step 3	[**no**] **mpls ip forwarding** **Example:** `switch(config-if)# mpls ip forwarding`	Enables MPLS on the specified interface. The **no** form of this command disables MPLS on the specified interface.

Step 4	**mpls static configuration**
	Example:
	`switch(config-if)# mpls static configuration`
	`switch(config-mpls-static)#`
		Enters MPLS static global configuration mode.

Step 5	**address-family** {**ipv4** | **ipv6**} **unicast**
	Example:
	`switch(config-mpls-static)# address-family ipv4`
	`unicast`
	`switch(config-mpls-static-af)#`
		Enters global address family configuration mode for the
		specified IPv4 or IPv6 address family.

Step 6	**local-label** *local-label-value* **prefix** *destination-*	Specifies static binding of incoming labels to IPv4

	Command or Action	Purpose
	prefix *destination-prefix-mask* **Example:** `switch(config-mpls-static-af)# local-label 2000 prefix 1.255.200.0 255.255.255.25 switch(config-mpls-static-af-lbl)#`	or IPv6 prefixes. The *local-label-value* is the range of the static MPLS label defined in the **mpls label range** command.
Step 7	**next-hop** {**auto-resolve** | *destination-ip-next-hop* **out-label implicit-null** | **backup** *local-egress-interface destination-ip-next-hop* **out-label** *output-label-value*} **Example:** `switch(config-mpls-static-af-lbl)# next-hop auto-resolve`	Specifies the next hop. These options are available: • **next-hop auto-resolve**—Use this option for label swap operations. • **next-hop** *destination-ip-next-hop* **out-label implicit-null**—Use this option for the primary path in label pop operations. • **next-hop backup** *local-egress-interface destination-ip-next-hop* **out-label** *output-label-value*—Use this option for the backup path in label pop operations.
Step 8	(Optional) **copy running-config startup-config** **Example:** `switch(config-mpls-static-af-lbl)# copy running-config startup-config`	Copies the running configuration to the startup configuration.

2.6 Verifying the Static MPLS Configuration

To display the static MPLS configuration, perform one of the following tasks:

Command	Purpose
show feature | inc mpls_static	Displays the status of static MPLS.
show feature-set	Displays the status of the MPLS feature set.
show ip route	Displays routes from the unicast Routing Information Base (RIB).
show mpls label range	Displays the label range that is configured for static MPLS.
show mpls static binding {**all** | **ipv4** | **ipv6**}	Displays the configured static prefix or label bindings.

Command	Purpose
show mpls switching [**detail**]	Displays MPLS switching information.

This example shows sample output for the **show mpls static binding all** command:

```
1.255.200.0/32: (vrf: default) Incoming label: 2000
  Outgoing labels:
      1.21.1.1  implicit-null
      backup 1.24.1.1 2001
2000:1:255:201::1/128: (vrf: default) Incoming label: 3000
  Outgoing labels:
      2000:1111:2121:1111:1111:1111:1111:1  implicit-null
      backup 2000:1:24:1::1 3001
```

This example shows sample output for the **show mpls switching detail** command:

```
VRF default
IPv4 FEC
 In-Label                        : 2000
 Out-Label stack                 : Pop Label
 FEC                             : 1.255.200.0/32
 Out interface                   : Po21
 Next hop                        : 1.21.1.1
 Input traffic statistics        : 0 packets, 0 bytes
 Output statistics per label     : 0 packets, 0 bytes
IPv6 FEC
 In-Label                        : 3000
 Out-Label stack                 : Pop Label
 FEC                             : 2000:1:255:201::1/128
 Out interface                   : port-channel21
 Next hop                        : 2000:1111:2121:1111:1111:1111:1111:1
 Input traffic statistics        : 0 packets, 0 bytes
 Output statistics per label     : 0 packets, 0 bytes
```

This example shows normal, XML, and JSON sample output for the **show mpls switching** command when the switch is configured with a static IPv4 prefix:

```
switch# show run mpls static | sec 'ipv4 unicast'
address-family ipv4 unicast
local-label 100 prefix 192.168.0.1 255.255.255.255 next-hop auto-resolve out-label 200
     switch# show mpls switching
     Legend:
     (P)=Protected, (F)=FRR active, (*)=more labels in stack.
     IPV4:                 FEC name           Out-Interface      Next-Hop
     In-Label   Out-Label
     VRF default 200       192.168.0.1/32     Eth1/23            1.12.23.2
     100

switch# show mpls switching | xml
<?xml version="1.0" encoding="ISO-8859-1"?> <nf:rpc-reply
xmlns:nf="urn:ietf:params:xml:ns:netconf:base:1.0"
 <nf:data>
```

```
<show>
 <mpls>
  <switching>
   <__XML__OPT_Cmd_ulib_show_switching_cmd_labels>
   <__XML__OPT_Cmd_ulib_show_switching_cmd_detail>
     <__XML__OPT_Cmd_ulib_show_switching_cmd_vrf>
      <__XML__OPT_Cmd_ulib_show_switching_cmd___readonly__>
       <__readonly__>
        <TABLE_vrf>
             <ROW_vrf>
              <vrf_name>default</vrf_name>
              <TABLE_inlabel> <ROW_inlabel>

                <in_label>100</in_label>
                <out_label_stack>200</out_label_stack>
                <ipv4_prefix>192.168.0.1/32</ipv4_prefix>
                <out_interface>Eth1/23</out_interface>
                <ipv4_next_hop>1.12.23.2</ipv4_next_hop>
                <nhlfe_p2p_flag> </nhlfe_p2p_flag>
               </ROW_inlabel>
              </TABLE_inlabel>
           </ROW_vrf>
           </TABLE_vrf>
           </__readonly__>
          </__XML__OPT_Cmd_ulib_show_switching_cmd___readonly__>
         </__XML__OPT_Cmd_ulib_show_switching_cmd_vrf>
        </__XML__OPT_Cmd_ulib_show_switching_cmd_detail>
       </__XML__OPT_Cmd_ulib_show_switching_cmd_labels>
      </switching>
     </mpls>
    </show>
   </nf:data>
  </nf:rpc-
  reply> ]]>]]>
```


```
switch# show mpls switching | json
```


```
{"TABLE_vrf": {"ROW_vrf": {"vrf_name": "default", "TABLE_inlabel":
{"ROW_inlabel
": {"in_label": "100", "out_label_stack": "200", "ipv4_prefix":
"192.168.0.1/32"
, "out_interface": "Eth1/23", "ipv4_next_hop": "1.12.23.2",
"nhlfe_p2p_flag": nu
ll}}}}}
```


2.7 Displaying Static MPLS Statistics

To monitor static MPLS statistics, perform one of the following tasks:

Command	Purpose
show forwarding [**ipv6**] **adjacency mpls stats**	Displays MPLS IPv4 or IPv6 adjacency statistics.
show forwarding mpls drop-stats	Displays the MPLS forwarding packet drop statistics.
show forwarding mpls ecmp [**module** *slot* | **platform**]	Displays the MPLS forwarding statistics for equal-cost multipath (ECMP).
show forwarding mpls label *label* **stats** [**platform**]	Displays MPLS label forwarding statistics.

Command	Purpose
show mpls forwarding statistics [**interface** *type slot/port*]	Displays MPLS forwarding statistics.
show mpls switching labels *low-label-value* [*high-label-value*] [**detail**]	Displays the MPLS label switching statistics. The range for the label value is from 0 to 524286.

This example shows sample output for the **show forwarding adjacency mpls stats** command:

```
FEC             next-hop   interface  tx packets  tx bytes  Label info
--------------- ---------- ---------- ----------- --------- ----------
1.255.200.0/32  1.21.1.1   Po21       87388       10836236  POP 3
1.255.200.0/32  1.24.1.1   Po24       0           0         SWAP 2001
switch(config)#
switch(config)# show forwarding mpls drop-stats
Dropped packets : 73454
Dropped bytes : 9399304
```

This example shows sample output for the **show forwarding ipv6 adjacency mpls stats** command:

```
FEC                       next-hop        interface  tx packets  tx bytes  Label info
2000:1:255:201::1/128---------- 2000:1.21.1.1-------- ---------- ----------- --------- -----------
-----------               ------          Po21       46604       5778896   POP 3
2000:1:255:201::1/128  2000:1:24:1::1  Po24       0           0         SWAP 3001
```

This example shows sample output for the **show forwarding mpls label 2000 stats** command:

```
        +-----------      +-------------------+----------------+-------------+-------
Local-------- |Prefix     |FEC                |Next-Hop        |Interface    |Out
Label   |Table Id   |(Prefix/Tunnel id) |                |             |Label
        +-----------      +-------------------+----------------+-------------+-------
2000-------- |0x1        |1.255.200.0/32     |1.21.1.1        |Po21         |Pop Label
 HH: 100008, Refcount: 1
Input Pkts : 77129               Input Bytes : 9872512
Output Pkts: 77223               Output Bytes: 9575652
```

This example shows sample output for the **show mpls forwarding statistics** command:

```
MPLS software forwarding stats summary:
    Packets/Bytes sent           : 0/0
    Packets/Bytes received       : 0/0
    Packets/Bytes forwarded      : 0/0
    Packets/Bytes originated     : 0/0
    Packets/Bytes consumed       : 0/0
    Packets/Bytes input dropped  : 0/0
    Packets/Bytes output dropped : 0/0
```

2.8 Clearing Static MPLS Statistics

To clear the static MPLS statistics, perform these tasks:

Command	Purpose
clear forwarding [**ipv6**] **adjacency mpls stats**	Clears the MPLS IPv4 or IPv6 adjacency statistics.
clear forwarding mpls drop-stats	Clears the MPLS forwarding packet drop statistics.
clear forwarding mpls stats	Clears the ingress MPLS forwarding statistics.
clear mpls forwarding statistics	Clears the MPLS forwarding statistics.
clear mpls switching label statistics [**interface** *type slot/port*]	Clears the MPLS switching label statistics.

2.9 Configuration Examples for Static MPLS

This example shows how to reserve labels for static assignment:

```
switch# configure terminal
Enter configuration commands, one per line.  End with CNTL/Z.
switch(config)# mpls label range 17 99 static 100 0000
switch(config)# show mpls label range
Downstream Generic label region: Min/Max label: 17/99
Range for static labels: Min/Max Number: 100/10000
```

This example shows how to configure MPLS static label and IPv4 prefix binding in a top-of-rack configuration (swap configuration):

```
switch# configure terminal
Enter configuration commands, one per line.  End with CNTL/Z.
switch(config)# interface ethernet 1/1
switch(config-if)# mpls ip forwarding
switch(config-if)# mpls static configuration
switch(config-mpls-static)# address-family ipv4 unicast
switch(config-mpls-static-af)# local-label 2000 prefix 1.255.200.0/32
switch(config-mpls-static-af-lbl)# next-hop auto-resolve out-label 2000
```

This example shows how to configure MPLS static label and IPv6 prefix binding in a top-of-rack configuration (swap configuration):

```
switch# configure terminal
Enter configuration commands, one per line.  End with CNTL/Z.
switch(config)# interface ethernet 1/1
switch(config-if)# mpls ip forwarding
switch(config-if)# mpls static configuration
switch(config-mpls-static)# address-family ipv6 unicast
switch(config-mpls-static-af)# local-label 3001 prefix 2000:1:255:201::1/128
switch(config-mpls-static-af-lbl)# next-hop auto-resolve out-label 3001
```

This example shows how to configure MPLS static label and IPv4 prefix binding in an aggregator configuration (pop configuration):

```
switch# configure terminal
Enter configuration commands, one per line.  End with CNTL/Z.
```

```
switch(config)# interface ethernet 1/1
switch(config-if)# mpls ip forwarding
switch(config-if)# mpls static configuration
switch(config-mpls-static)# address-family ipv4 unicast
switch(config-mpls-static-af)# local-label 2000 prefix .255.200.0/32
switch(config-mpls-static-af-lbl)# next-hop 1.31.1.1 out-label implicit-null
switch(config-mpls-static-af-lbl)# next-hop backup Po34 1.34.1.1 out-label
2000
```

This example shows how to configure MPLS static label and IPv6 prefix binding in an aggregator configuration (pop configuration):

```
switch# configure terminal
Enter configuration commands, one per line.  End with CNTL/Z.
switch(config)# interface ethernet 1/1
switch(config-if)# mpls ip forwarding
switch(config-if)#     mpls     static     configuration
switch(config-mpls-static)# address-family ipv6 unicast
switch(config-mpls-static-af)#    local-label    3001    prefix    2000:1:255:201::1/128
switch(config-mpls-static-af-lbl)#   next-hop   2000:1:31:1::1   out-label   implicit-null
switch(config-mpls-static-af-lbl)# next-hop backup Po34 2000:1:34:1::1 out-label 3001
```

2.10 Additional References

2.10.1 Related Documents

Related Topic	Document Title
MPLS TCAM regions	See the *Using Templates to Configure ACL TCAM Region Sizes* section in the Inspur CN12900 Series INOS-CN Security Configuration Guide.

CHAPTER 3 Configuring MPLS Label Imposition

This chapter contains information on how to configure multiprotocol label switching (MPLS) label imposition.

3.1 About MPLS Label Imposition

An outgoing label stack having one or more labels can be statically provisioned using the MPLS Label Stack Imposition feature. The outgoing label stack is used in the following two types of statically configured MPLS bindings:

- Prefix and Label to Label Stack - Here an IP prefix or an incoming label is mapped to an outgoing stack, similar to static MPLS. An incoming prefix is mapped to out-label-stack for IP-only ingress traffic.
- Label to Label Stack - Here only an incoming label is mapped to an outgoing stack without any prefix.

The new MPLS binding types are implemented in the static MPLS component and are available only when the **feature mpls segment-routing** command is enabled.

If configured next-hops of MPLS label imposition are SR recursive next-hops (RNH), then they are resolved to actual next-hops using RIB. The outer label of the out-label stack is imposed automatically from the SR allocated labels.

ECMP is also supported by adding a number of path configurations.

3.2 Guidelines and Limitations for MPLS Label Imposition

MPLS label imposition has the following guidelines and limitations:

- MPLS label imposition is supported for the following:
- Inspur CN12900 Series switches with the CN129-X636C-R, CN129-X636Q-R, or CN129-X6136YC-R line cards.
- MPLS label imposition supports only IPv4.
- The maximum number of labels in an out-label stack is three for Inspur CN12900 Series switches. If you try to impose more labels, the trailing label is truncated automatically, and a syslog error message appears signaling to correct the configuration.
- Multicast is not supported for MPLS label imposition.
- Subinterfaces and port channels are not supported for MPLS label imposition.
- Prefixes and associated subnet masks learned from routing protocols (including from static routes) cannot be used as part of the label stack imposition policy.
- For label stack imposition verified scalability limits, see the Verified Scalability Guide for your device.

3.3 Configuring MPLS Label Imposition

3.3.1 Enabling MPLS Label Imposition

You must install and enable the MPLS feature set and then enable the MPLS segment routing feature before you can configure MPLS label imposition.

SUMMARY STEPS

1. **configure terminal**
2. [**no**] **install feature-set mpls**

3. [**no**] **feature-set mpls**
4. [**no**] **feature mpls segment-routing**
5. (Optional) **show feature-set**
6. (Optional) **show feature | grep segment-routing**

DETAILED STEPS

	Command or Action	Purpose

Step 1	**configure terminal**
	Example:
	`switch# configure terminal`
	`switch(config)#`
		Enters global configuration mode.

Step 2	[**no**] **install feature-set mpls** **Example:** `switch(config)# install feature-set mpls`	Installs the MPLS feature set. The **no** form of this command uninstalls the MPLS feature set.
Step 3	[**no**] **feature-set mpls** **Example:** `switch(config)# feature-set mpls`	Enables the MPLS feature set. The **no** form of this command disables the MPLS feature set.
Step 4	[**no**] **feature mpls segment-routing** **Example:** `switch(config)# feature mpls segment-routing`	Enables the MPLS segment routing feature. The **no** form of this command disables the MPLS segment routing feature.

Step 5	(Optional) **show feature-set**
	Example:
	`switch(config)# show feature-set`
	`Feature Set Name     ID       State`
	`-------------------- -------- ---------`
	`mpls                 4        enabled`
		Displays the status of the MPLS feature set.

Step 6	(Optional) **show feature | grep segment-routing**
	Example:
	`switch(config)# show feature | grep segment-routing`
	`segment-routing        1        enabled`
		Displays the status of MPLS segment routing.

3.3.2 Reserving Labels for MPLS Label Imposition

You can reserve the labels that are to be statically assigned. Dynamic label allocation is not supported.

Before you begin

Ensure that the MPLS segment routing feature is enabled.

SUMMARY STEPS

1. **configure terminal**
2. [**no**] **mpls label range** *min-value max-value* [**static** *min-static-value max-static-value*]
3. (Optional) **show mpls label range**
4. (Optional) **copy running-config startup-config**

DETAILED STEPS

	Command or Action	Purpose

Step 1	**configure terminal**
	Example:
	`switch# configure terminal`
	`switch(config)#`
		Enters global configuration mode.

Step 2	[**no**] **mpls label range** *min-value max-value* [**static** *min-static-value max-static-value*] **Example:** `switch(config)# mpls label range 17 99 static 100 10000`	Reserves a range of labels for static label assignment. The range for the minimum and maximum values is from 16 to 471804.
Step 3	(Optional) **show mpls label range** **Example:** `switch(config)# show mpls label range`	Displays the label range that is configured for static MPLS.
Step 4	(Optional) **copy running-config startup-config** **Example:** `switch(config)# copy running-config startup-config`	Copies the running configuration to the startup configuration.

3.3.3 Configuring MPLS Label Imposition

You can configure MPLS label imposition on the device.

Before you begin

Ensure that the MPLS segment routing feature is enabled.
Set a static label range as follows: **mpls label range 16 16 static 17 50000**.

SUMMARY STEPS

1. **configure terminal**
2. **interface** *type slot/port*
3. [**no**] **mpls ip forwarding**
4. **mpls static configuration**
5. **address-family ipv4 unicast**
6. **lsp** *name*

7. **in-label** *value* **allocate policy** *prefix*
8. **forward**
9. **path** *number* **next-hop** *ip-address* **out-label-stack** *label-id label-id*
10. (Optional) **copy running-config startup-config**

DETAILED STEPS

	Command or Action	Purpose

Step 1	**configure terminal**
	Example:
	`switch# configure terminal`
	`switch(config)#`
		Enters global configuration mode.

Step 2	**interface** *type slot/port*
	Example:
	`switch(config)# interface ethernet 2/2`
	`switch(config-if)#`
		Enters the interface configuration mode for the specified
		interface.

Step 3	[**no**] **mpls ip forwarding** **Example:** `switch(config-if)# mpls ip forwarding`	Enables MPLS on the specified interface. The **no** form of this command disables MPLS on the specified interface.

Step 4	**mpls static configuration**
	Example:
	`switch(config-if)# mpls static configuration`
	`switch(config-mpls-static)#`
		Enters MPLS static global configuration mode.

Step 5	**address-family ipv4 unicast**
	Example:
	`switch(config-mpls-static)# address-family ipv4`
	`unicast`
	`switch(config-mpls-static-af)#`
		Enters global address family configuration mode for the
		specified IPv4 address family.

Step 6	**lsp** *name*
	Example:
	`switch(config-mpls-static-af)# lsp lsp1`
	`switch(config-mpls-static-lsp)#`
		Specifies a name for LSP.

Step 7	**in-label** *value* **allocate policy** *prefix*
	Example:
	`switch(config-mpls-static-lsp)# in-label 8100`
	`allocate policy 15.15.1.0/24`
	`switch(config-mpls-static-lsp-inlabel)#`
		Configures an in-label value and a prefix value (optional).

	Command or Action	Purpose

Step 8	**forward**
	Example:
	`switch(config-mpls-static-lsp-inlabel)# forward`
	`switch(config-mpls-static-lsp-inlabel-forw)#`
		Enters the forward mode.

Step 9	**path** *number* **next-hop** *ip-address* **out-label-stack** *label-id* *label-id* **Example:** `switch(config-mpls-static-lsp-inlabel-forw)# path 1 next-hop 13.13.13.13 out-label-stack 16 3000`	Specifies the path. The maximum number of supported paths is 32.
Step 10	(Optional) **copy running-config startup-config** **Example:** `switch(config-mpls-static-lsp-inlabel-forw)# copy running-config startup-config`	Copies the running configuration to the startup configuration.

3.4 Verifying the MPLS Label Imposition Configuration

To display the MPLS label imposition configuration, perform one of the following tasks:

Command	Purpose
show feature | grep segment-routing	Displays the status of MPLS label imposition.
show feature-set	Displays the status of the MPLS feature set.
show forwarding mpls label *label*	Displays MPLS label forwarding statistics for a particular label.
show mpls label range	Displays the label range that is configured for MPLS label imposition.
show mpls static binding {**all** | **ipv4**}	Displays the configured static prefix or label bindings.
show mpls switching [**detail**]	Displays MPLS label switching information.
show running-config mpls static	Displays the running static MPLS configuration.

This example shows sample output for the **show forwarding mpls label 8100** command:

```
slot 1
======= -----------
--------   +          +-------------------+----------------+------------+---------------------
Local|Prefix|FEC              |Next-Hop   |Interface | Out Label|Table Id |(Prefix/Tunnel
id)|Label-----------
8100--------   +          +-------------------+----------------+------------+---------------------|                  |
```

```
     |0x1    |25.25.0.0/16 |12.12.1.2|Po121      |3131 SWAP |
 17
"    |0x1    |25.25.0.0/16 |12.12.2.2|Eth1/51    |3131 SWAP |         |               |
 17
"    |0x1    |25.25.0.0/16 |12.12.3.2|Vlan122    |3131 SWAP |         |               |
 17
"    |0x1    |25.25.0.0/16 |12.12.4.2|Vlan123    |3131 SWAP |         |               |
 17
```

This example shows sample output for the **show mpls static binding all** command:

```
LI_TEST1 25.25.0.0/16: (vrf: default) Incoming label: 8100
LSP  Type:  POLICY
  Outgoing labels:
     (path 1) 12.12.1.2  3131,17
     (path 2) 12.12.2.2 3131,17
```

This example shows sample output for the **show mpls static binding all** command:

```
LI_TEST1 25.25.0.0/16: (vrf: default) Incoming label: 8100
LSP Type: POLICY
  Outgoing labels:
     (path 1) 12.12.1.2 3131,17
     (path 2) 12.12.2.2 3131,17



    (path  3)  12.12.3.2 3131,17
    (path  4)  12.12.4.2 3131,17
LI_TEST2 (vrf: default) Incoming label: 8200
LSP Type: XC
   Outgoing labels:
      (path 1)  12.12.3.2 3132,16
      (path 2)  12.12.4.2 3132,16
      (path 3)  12.12.1.2 3132,16
      (path 4)  12.12.2.2 3132,16
```

This example shows sample output for the **show mpls switching** command:

```
Legend:
(P)=Protected, (F)=FRR active, (*)=more labels in stack.
  Local      Out-Label  FEC                                           Out-Interface
  Next-Hop   3132       Label 8200
  8200                                       *
  12.12.3.2  3132       Label 8200
  8200                                       *
  12.12.4.2  3132       Label 8200
  8200                                       *
  12.12.1.2  3132       Label 8200
  8200                                       *
  12.12.2.2
  Local      Out-Label  FEC                                           Out-Interface
  Next-Hop   3131       Pol 25.25.0.0/16
```

```
8100                                                *
12.12.1.2  3131       Pol 25.25.0.0/16
8100                                                *
12.12.2.2  3131       Pol 25.25.0.0/16
8100                                                *
12.12.3.2  3131       Pol 25.25.0.0/16
8100                                                *
12.12.4.2
```

This example shows sample output for the **show running-config mpls static** command:

```
mpls static configuration
  address-family ipv4 unicast
    lsp LI_TEST2
      in-label 8100 allocate policy 25.25.0.0 255.255.0.0
        forward
          path 1 next-hop 12.12.1.2 out-label-stack 3131 17
          path 2 next-hop 12.12.2.2 out-label-stack 3131 17
          path 3 next-hop 12.12.3.2 out-label-stack 3131 17
          path 4 next-hop 12.12.4.2 out-label-stack 3131 17
```

This example shows sample output for the **show running-config mpls static all** command.

```
switch# show running-config mpls static all

!Command: show running-config mpls static all
!Time: Mon Oct 1 14:59:46 2018

version 9.2(1i)
logging level mpls static 5
mpls  static  configuration
address-family ipv4 unicast


lsp 9_label_stack_LPM
in-label    72000    allocate    policy    71.200.11.0
255.255.255.0 forward
path 1 next-hop 27.1.32.4 out-label-stack 21901 29701 27401 24501
25801 lsp 9_label_stack_LPM_01
in-label    72001    allocate    policy    72.201.1.1
255.255.255.255 lsp DRV-01
in-label    71011    allocate    policy    71.111.21.0
255.255.255.0 forward
path 1 next-hop 27.1.31.4 out-label-stack implicit-
null lsp DRV-02
in-label    71012    allocate    policy    71.111.22.0
255.255.255.0 forward
path  1  next-hop  8.8.8.8  out-label-stack
28901 lsp DRV-03
switch# show forwarding mpls label 72000

slot 1
=======
--------   +----------- +------------------+---------------- +------------- +-------
Local |Prefix |FEC |Next-Hop |Interface |Out
Label |Table Id |(Prefix/Tunnel id) | | |Label          +             +
72000--------   +----------- +------------------+----------------
```

```
       |0x1 |71.200.11.0/24 |27.1.32.4 |Eth1/21 |21901 SWAP
| | | | | | 29701
| | | | | | 27401
| | | | | | 24501
| | | | | | 25801
```

3.5 Displaying MPLS Label Imposition Statistics

To monitor MPLS label imposition statistics, perform one of the following tasks:

Command	Purpose
show forwarding [**ipv4**] **adjacency mpls stats**	Displays MPLS IPv4 adjacency statistics.
show forwarding mpls label *label* **stats** [**platform**]	Displays MPLS label forwarding statistics.
show mpls forwarding statistics [**interface** *type slot/port*]	Displays MPLS forwarding statistics.
show mpls switching labels *low-label-value* [*high-label-value*] [**detail**]	Displays MPLS label switching statistics. The range for the label value is from 0 to 524286.

This example shows sample output for the **show forwarding adjacency mpls stats** command:

```
FEC              next-hop   interface  tx packets  tx bytes  Label info
---------------- ---------- ---------- ----------- --------- ----------
1.255.200.0/32   1.21.1.1   Po21       87388       10836236  POP 3
1.255.200.0/32   1.24.1.1   Po24       0           0         SWAP 2001
switch(config)#
switch(config)# show forwarding mpls drop-stats
Dropped packets : 73454
Dropped bytes : 9399304
```

This example shows sample output for the **show forwarding ipv6 adjacency mpls stats** command:

```
FEC                       next-hop         interface  tx packets  tx bytes  Label info
2000:1:255:201::1/128---------- 2000:1.21.1.1-------- ---------- ----------- --------- -----------
-----------               ------           Po21       46604       5778896   POP 3
2000:1:255:201::1/128     2000:1:24:1::1   Po24       0           0         SWAP 3001
```

This example shows sample output for the **show forwarding mpls label 2000 stats** command:

```
        +-----------  +-------------------+----------------+-------------+-------
Local-------- |Prefix |FEC                |Next-Hop        |Interface    |Out
Label   |Table Id    |(Prefix/Tunnel id) |                |             |Label
        +-----------  +-------------------+----------------+-------------+-------
2000-------- |0x1     |1.255.200.0/32     |1.21.1.1        |Po21         |Pop Label
 HH: 100008, Refcount: 1
Input Pkts : 77129                 Input Bytes : 9872512
Output Pkts: 77223                 Output Bytes: 9575652
```

This example shows sample output for the **show mpls forwarding statistics** command:

```
MPLS software forwarding stats summary:
```

```
Packets/Bytes sent           : 0/0
Packets/Bytes received       : 0/0
Packets/Bytes forwarded      : 0/0
Packets/Bytes originated     : 0/0
Packets/Bytes consumed       : 0/0
Packets/Bytes input dropped  : 0/0
Packets/Bytes output dropped : 0/0
```

3.6 Clearing MPLS Label Imposition Statistics

To clear the MPLS label imposition statistics, perform these tasks:

Command	Purpose
clear forwarding [**ipv4**] **adjacency mpls stats**	Clears the MPLS IPv4 adjacency statistics.
clear forwarding mpls stats	Clears the ingress MPLS forwarding statistics.
clear mpls forwarding statistics	Clears the MPLS forwarding statistics.
clear mpls switching label statistics [**interface** *type slot/port*]	Clears the MPLS switching label statistics.

3.7 Configuration Examples for MPLS Label Imposition

This example shows how to configure MPLS label imposition by allocating a prefix and an incoming-label to out-label-stack binding:

```
switch(config-if)# mpls static configuration
switch(config-mpls-static)# address-family ipv4
unicast
switch(config-mpls-static-af)# lsp LI_TEST1
switch(config-mpls-static-lsp)# in-label 8100 allocate policy 25.25.0.0/16
switch(config-mpls-static-lsp-inlabel)# forward
switch(config-mpls-static-lsp-inlabel-forw)# path 1 next-hop 12.12.1.2 out-label-
stack 3131
 17
switch(config-mpls-static-lsp-inlabel-forw)# path 2 next-hop 12.12.2.2 out-label-stack
3131
 17
switch(config-mpls-static-lsp-inlabel-forw)# path 3 next-hop 12.12.3.2 out-label-stack
3131
 17
switch(config-mpls-static-lsp-inlabel-forw)# path 4 next-hop 12.12.4.2 out-label-stack
3131
 17
```

To remove a next-hop, you can use

```
no path 1
```

To remove the named lsp, you can use

```
no lsp LI_TEST1
```

This example shows how to configure MPLS label imposition by allocating an incoming-label to out-label-stack binding (no prefix) :

```
switch(config-if)# mpls static configuration
switch(config-mpls-static)# address-family ipv4
unicast
switch(config-mpls-static-af)# lsp LI_TEST1
switch(config-mpls-static-lsp)# in-label 8200
allocate switch(config-mpls-static-lsp-inlabel)#
forward
switch(config-mpls-static-lsp-inlabel-forw)# path 1 next-hop 12.12.3.2 out-label-stack 3132
 16
switch(config-mpls-static-lsp-inlabel-forw)# path 2 next-hop 12.12.4.2 out-label-stack 3132
 16
switch(config-mpls-static-lsp-inlabel-forw)# path 3 next-hop 12.12.1.2 out-label-stack 3132
 16
switch(config-mpls-static-lsp-inlabel-forw)# path 4 next-hop 12.12.2.2 out-label-stack 3132
 16
```

CHAPTER 4 Configuring MPLS Layer 3 VPNs

This chapter describes how to configure Multiprotocol Label Switching (MPLS) Layer 3 Virtual Private Networks (VPNs) on Inspur CN12904 and CN12908 switches.

4.1 Information About MPLS Layer 3 VPNs

An MPLS Layer 3 VPN consists of a set of sites that are interconnected by an MPLS provider core network. At each customer site, one or more customer edge (CE) routers or Layer 2 switches attach to one or more provider edge (PE) routers. This section includes the following topics:

- MPLS Layer 3 VPN Definition
- How an MPLS Layer 3 VPN Works
- Components of MPLS Layer 3 VPNs
- Hub-and-Spoke Topology
- OSPF Sham-Link Support for MPLS VPN

4.1.1 MPLS Layer 3 VPN Definition

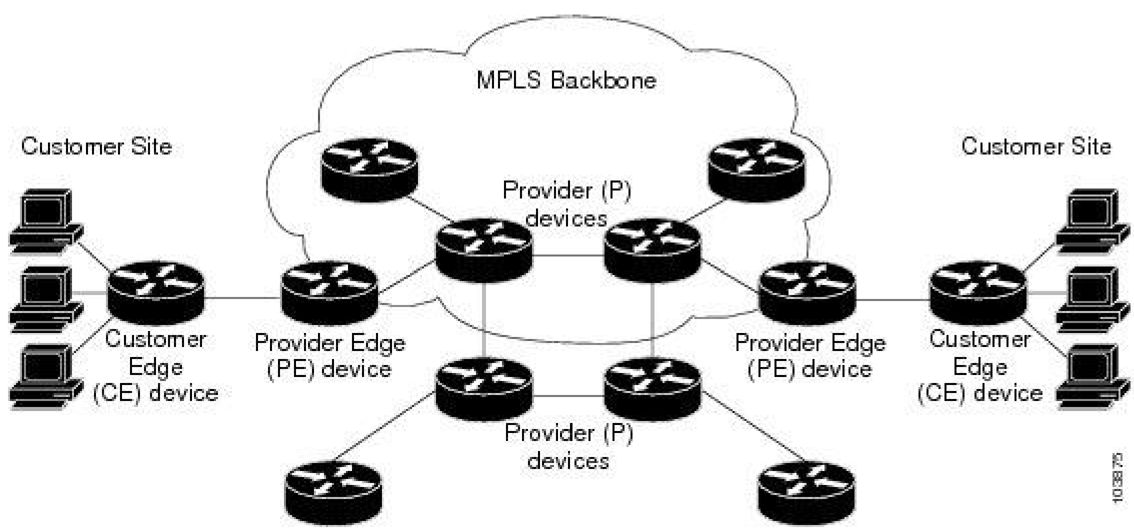
MPLS-based Layer 3 VPNs are based on a peer model that enables the provider and the customer to exchange Layer 3 routing information. The provider relays the data between the customer sites without direct customer involvement.

When you add a new site to an MPLS Layer 3 VPN, you must update the provider edge router that provides services to the customer site.

MPLS Layer 3 VPNs include the following components:

- Provider (P) router—A router in the core of the provider network. P routers run MPLS switching and do not attach VPN labels (an MPLS label in each route assigned by the PE router) to routed packets.
- Provider edge (PE) router—A router that attaches the VPN label to incoming packets that are based on the interface or subinterface on which they are received. A PE router attaches directly to a CE router.
- Customer edge (CE) router—An edge router on the network of the provider that connects to the PE router on the network. A CE router must interface with a PE router.

Figure 2： Basic MPLS Layer 3 VPN Terminology



4.1.2 How an MPLS Layer 3 VPN Works

MPLS Layer 3 VPN functionality is enabled at the edge of an MPLS network. The PE router performs the following tasks:

- Exchanges routing updates with the CE router
- Translates the CE routing information into VPN routes
- Exchanges Layer 3 VPN routes with other PE routers through the Multiprotocol Border Gateway Protocol (MP-BGP)

4.1.3 Components of MPLS Layer 3 VPNs

An MPLS-based Layer 3 VPN network has three components:

1. VPN route target communities—A VPN route target community is a list of all members of a Layer 3 VPN community. You must configure the VPN route targets for each Layer 3 VPN community member.
2. Multiprotocol BGP peering of VPN community PE routers—Multiprotocol BGP propagates VRF reachability information to all members of a VPN community. You must configure Multiprotocol BGP peering in all PE routers within a VPN community.
3. MPLS forwarding—MPLS transports all traffic between all VPN community members across a VPN enterprise or service provider network.

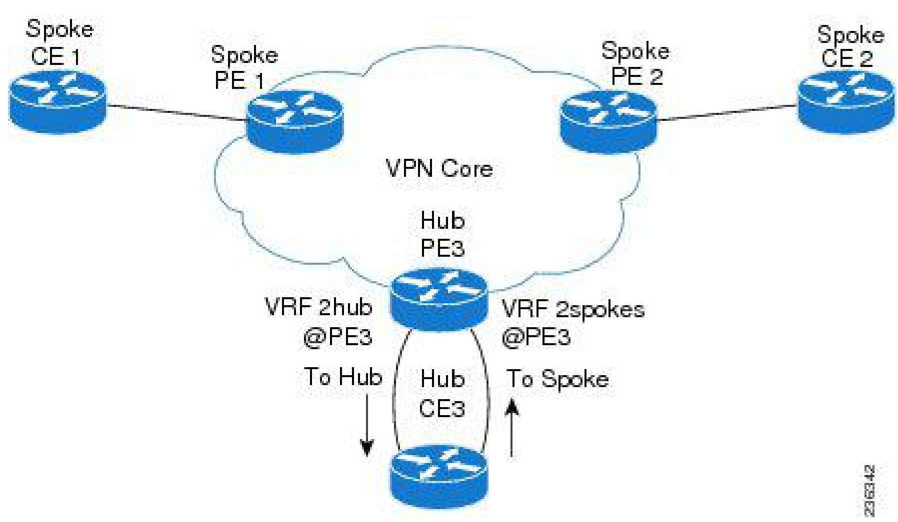
A one-to-one relationship does not necessarily exist between customer sites and VPNs. A site can be a member of multiple VPNs. However, a site can associate with only one VRF. A customer-site VRF contains all the routes that are available to the site from the VPNs of which it is a member.

4.1.4 Hub-and-Spoke Topology

A hub-and-spoke topology prevents local connectivity between subscribers at the spoke provider edge (PE) routers and ensures that a hub site provides subscriber connectivity. Any sites that connect to the same PE router must forward intersite traffic using the hub site. This topology ensures that the routing at the spoke sites moves from the access-side interface to the network-side interface or from the network-side interface to the access-side interface but never from the access-side interface to the access-side interface. A hub-and-spoke topology allows you to maintain access restrictions between sites.

A hub-and-spoke topology prevents situations where the PE router locally switches the spokes without passing the traffic through the hub site. This topology prevents subscribers from directly connecting to each other. A hub-and-spoke topology does not require one VRF for each spoke.

Figure 3：Hub-and-Spoke Topology



As shown in the figure, a hub-and-spoke topology is typically set up with a hub PE that is configured with two VRFs:

- VRF 2hub with a dedicated link connected to the hub customer edge (CE)
- VRF 2spokes with another dedicated link connected to the hub CE.

Interior Gateway Protocol (IGP) or external BGP (eBGP) sessions are usually set up through the hub PE-CE links. The VRF 2hub imports all the exported route targets from all the spoke PEs. The hub CE learns all routes from the spoke sites and readvertises them back to the VRF 2spoke of the hub PE. The VRF 2spoke exports all these routes to the spoke PEs.

If you use eBGP between the hub PE and hub CE, you must allow duplicate autonomous system (AS) numbers in the path which is normally prohibited. You can configure the router to allow this duplicate AS number at the neighbor of VRF 2spokes of the hub PE and also for VPN address family neighbors at all the spoke PEs. In addition, you must disable the peer AS number check at the hub CE when distributing routes to the neighbor at VRF 2spokes of the hub PE.

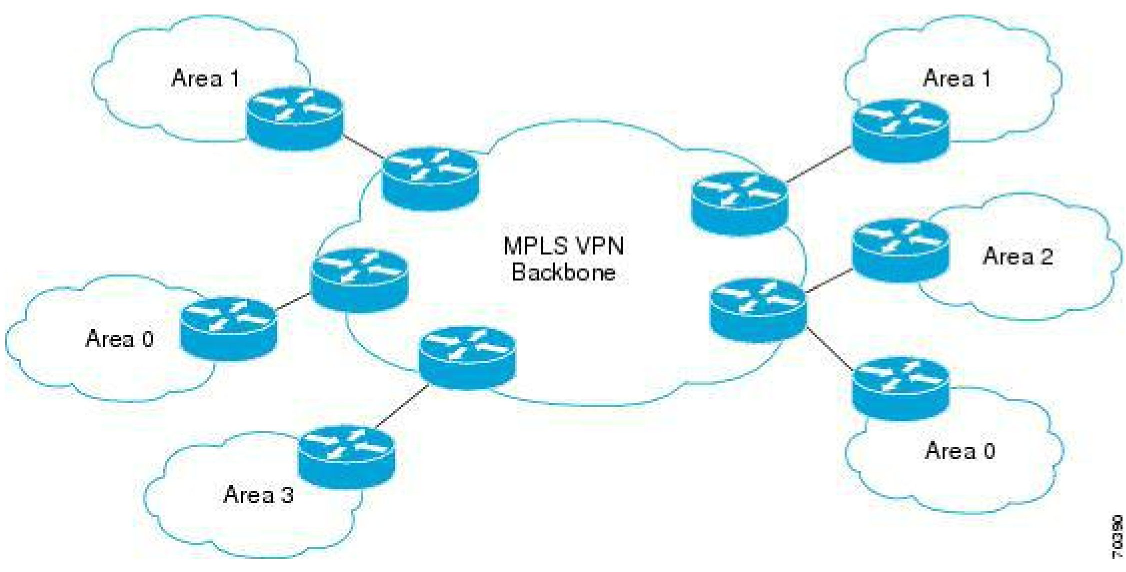
4.1.5 OSPF Sham-Link Support for MPLS VPN

In a Multiprotocol Label Switching (MPLS) VPN configuration, you can use the Open Shortest Path First (OSPF) protocol to connect customer edge (CE) devices to service provider edge (PE) devices in the VPN backbone. Many customers run OSPF as their intrasite routing protocol, subscribe to a VPN service, and want to exchange routing information between their sites using OSPF (during migration or on a permanent basis) over an MPLS VPN backbone.

The benefits of the OSPF sham-link support for MPLS VPN are as follows:

- Client site connection across the MPLS VPN Backbone—A sham link ensures that OSPF client sites that share a backdoor link can communicate over the MPLS VPN backbone and participate in VPN services.
- Flexible routing in an MPLS VPN configuration—In an MPLS VPN configuration, the OSPF cost that is configured with a sham link allows you to decide if OSPF client site traffic is routed over a backdoor link or through the VPN backbone.

The figure below shows an example of how VPN client sites that run OSPF can connect over an MPLS VPN backbone.



When you use OSPF to connect PE and CE devices, all routing information learned from a VPN site is placed in the VPN routing and forwarding (VRF) instance that is associated with the incoming interface. The PE devices that attach to the VPN use the Border Gateway Protocol (BGP) to distribute VPN routes to each other. A CE device can learn the routes to other sites in the VPN by peering with its attached PE device. The MPLS VPN super backbone provides an additional level of routing hierarchy to interconnect the VPN sites that are running OSPF.

When OSPF routes are propagated over the MPLS VPN backbone, additional information about the prefix in the form of BGP extended communities (route type, domain ID extended communities) is appended to the BGP update. This community information is used by the receiving PE device to decide the type of link-state advertisement (LSA) to be generated when the BGP route is redistributed to the OSPF PE-CE process. In this way, internal OSPF routes that belong to the same VPN and are advertised over the VPN backbone are seen as interarea routes on the remote sites.

4.2 Prerequisites for MPLS Layer 3 VPNs

MPLS Layer 3 VPNs has the following prerequisites:

- Ensure that you have configured MPLS and Label Distribution Protocol (LDP) in your network. All routers in the core, including the PE routers, must be able to support MPLS forwarding.
- Ensure that you have installed the correct license for MPLS and any other features you will be using with MPLS.

4.3 Guidelines and Limitations for MPLS Layer 3 VPNs

MPLS Layer 3 VPNs have the following configuration guidelines and limitations:

- You can configure MPLS Layer 3 VPN on Inspur CN12904 and CN12908 switches with the CN129-X636C-R, CN129-X636Q-R, and CN129-X6136YC-R line cards.
- You must enable MPLS IP forwarding on interfaces where the forwarding decisions are made based on the labels of incoming packets. If a VPN label is allocated by per prefix mode, MPLS IP forwarding must be enabled on the link between PE and CE.
- Because of the hardware limitation on the trap resolution, on Inspur CN12904 and CN12908 switches with the CN129-X636C-R, CN129-X636Q-R, and CN129-X6136YC-R line cards, uRPF may not be applied on supervisor bound packets via inband.
- On Inspur CN12904 and CN12908 switches with the CN129-X636C-R, CN129-X636Q-R, and CN129-X6136YC-R line cards, RACL is applied only to routed traffic so that the bridge traffic does not hit RACL. This applies to multicast OSPF control traffic.
- On Inspur CN12904 and CN12908 switches with the CN129-X636C-R, CN129-X636Q-R, and CN129-X6136YC-R line cards, Control Packets with Explicit-NULL label is not prioritized when sending to SUP. This may result in control protocols flapping when explicit-NULL is configured.
- Per-label statistics at a scale of 500K is not supported on Inspur CN12904 and CN12908 switches with the CN129-X636C-R, CN129-X636Q-R, and CN129-X6136YC-R line cards because of the hardware limitation.
- ARP scaling on Inspur CN12904 and CN12908 switches with the CN129-X636C-R, CN129-X636Q-R, and CN129-X6136YC-R line cards are limited to 64K if all the 64K MACs are different. This limitation also applies if there are several Equal Cost Multiple Paths (ECMP) configured on the interface.
- Packets with MPLS Explicit-NULL may not be parsed correctly with default line card profile.
- MPLS Layer 3 VPNs support the following CE-PE routing protocols:
- BGP (IPv4 and IPv6)
- Enhanced Interior Gateway Protocol (EIGRP) (IPv4)
- Open Shortest Path First (OSPFv2)
- Routing Information Protocol (RIPv2)

Set statements in an import route map are ignored.

- The BGP minimum route advertisement interval (MRAI) value for all iBGP and eBGP sessions is zero and is not configurable.
- In a high scale setup with many BGP routes getting redistributed into EIGRP, modify the EIGRP signal timer

to ensure that the EIGRP convergence time is higher than the BGP convergence time. This process allows all the BGP routes to be redistributed into EIGRP, before EIGRP signals convergence.
- When OSPF is used as a protocol between PE and CE devices, the OSPF metric is preserved when routes are advertised over the VPN backbone. The metric is used on the remote PE devices to select the correct route. Do not modify the metric value when OSPF is redistributed to BGP and when BGP is redistributed to OSPF. If you modify the metric value, routing loops might occur.
- On Inspur CN12904 and CN12908 switches with the CN129-X636C-R, CN129-X636Q-R, and CN129-X6136YC-R line cards, MPLS Traffic Engineering (RSVP) is not available.
- On Inspur CN12904 and CN12908 switches with the CN129-X636C-R, CN129-X636Q-R, and CN129-X6136YC-R line cards, only a few LDP sessions can be configured for usage with MPLS Layer 3 VPN.

4.4 Default Settings for MPLS Layer 3 VPNs

Table 2：Default MPLS Layer 3 VPN Parameters

Parameters	**Default**
L3VPN feature	Disabled
L3VPN SNMP notifications	Disabled
allowas-in (for a hub-and-spoke topology)	0
disable-peer-as-check (for a hub-and-spoke topology)	Disabled

4.5 Configuring MPLS Layer 3 VPNs

4.5.1 Configuring the Core Network

Assessing the Needs of MPLS Layer 3 VPN Customers

You can identify the core network topology so that it can best serve MPLS Layer 3 VPN customers.
- Identify the size of the network:
- Identify the following to determine the number of routers and ports you need:
- How many customers do you need to support?
- How many VPNs are needed per customer?
- How many virtual routing and forwarding instances are there for each VPN?
- Determine which routing protocols you need in the core network.
- Determine if you need MPLS VPN high availability support.
- Configure the routing protocols in the core network.
- Determine if you need BGP load sharing and redundant paths in the MPLS Layer 3 VPN core.

Configuring MPLS in the Core

To enable MPLS on all routers in the core, you must configure a label distribution protocol. You can use either of the following as a label distribution protocol:
- MPLS Label Distribution Protocol (LDP).
- MPLS Traffic Engineering Resource Reservation Protocol (RSVP).

Configuring Multiprotocol BGP on the PE Routers and Route Reflectors

You can configure multiprotocol BGP connectivity on the PE routers and route reflectors.

Before you begin

- Ensure that graceful restart is enabled on all routers for BGP and LDP.

SUMMARY STEPS

1. **configure terminal**
2. **feature bgp**
3. **feature-set mpls**
4. **feature-set mpls l3vpn**
5. **router bgp** *as - number*
6. **router-id** *ip-address*
7. **neighbor** *ip-address***remote-as** *as-number*
8. **address-family { vpnv4 | vpnv6 } unicast**
9. **send-community extended**
10. **show bgp { vpnv4 | vpnv6 } unicast neighbors**
11. **copy running-config startup-config**

DETAILED STEPS

	Command or Action	Purpose

Step 1	**configure terminal**
	Example:
	`switch# configure terminal`
	`switch(config)#`
		Enters global configuration mode.

Step 2	**feature bgp**
	Example:
	`switch(config)# feature bgp`
	`switch(config)#`
		Enables the BGP feature.

Step 3	**feature-set mpls**
	Example:
	`switch(config)# feature-set mpls`
	`switch(config)#`
		Enables the MPLS feature-set.

Step 4	**feature-set mpls l3vpn**
	Example:
	`switch(config)# feature-set mpls l3vpn`
	`switch(config)#`
		Enables the MPLS Layer 3 VPN feature.

Step 5	**router bgp** *as - number* **Example:** `switch(config)# router bgp 1.1`	Configures a BGP routing process and enters router configuration mode. The as-number argument indicates the number of an autonomous system that identifies the router to other BGP routers and tags the routing information. The AS number can be a 16-bit integer or a

	Command or Action	Purpose
		32-bit integer in the form of a higher 16-bit decimal number and a lower 16-bit decimal number in xx.xx format.
Step 6	**router-id** *ip-address* **Example:** `switch(config-router)# router-id 192.0.2.255`	(Optional) Configures the BGP router ID. This IP address identifies this BGP speaker. This command triggers an automatic notification and session reset for the BGP neighbor sessions. .

Step 7	**neighbor** *ip-address***remote-as** *as-number*
	
	Example:
	
	`switch(config-router)# neighbor 209.165.201.1`
	`remote-as 1.1`
	`switch(config-router-neighbor)#`
		Adds an entry to the iBGP neighbor table. The ip-address
		argument specifies the IP address of the neighbor in dotted
		decimal notation.

Step 8	**address-family { vpnv4 | vpnv6 } unicast**
	
	Example:
	
	`switch(config-router-neighbor)# address-family`
	`vpnv4 unicast`
	`switch(config-router-neighbor-af)#`
		Enters address family configuration mode for configuring
		routing sessions, such as BGP, that use standard VPNv4
		or VPNv6 address prefixes.

Step 9	**send-community extended**
	
	Example:
	
	`switch(config-router-neighbor-af)# send-community`
	` extended`
		Specifies that a communities attribute should be sent to a
		BGP neighbor.

Step 10	**show bgp { vpnv4 | vpnv6 } unicast neighbors**
	
	Example:
	
	`switch(config-router-neighbor-af)# show bgp vpnv4`
	` unicast neighbors`
		(Optional) Displays information about BGP neighbors.

Step 11	**copy running-config startup-config** **Example:** `switch(config-router-vrf)# copy running-`	(Optional) Copies the running configuration to the startup configuration.

	Command or Action	Purpose
	`config` `startup-config`	

4.5.2 Connecting the MPLS VPN Customers

Defining VRFs on the PE Routers to Enable Customer Connectivity

You must create VRFs on the PE routers to enable customer connectivity. You configure route targets to control which IP prefixes are imported into the customer VPN site and which IP prefixes are exported to the BGP network. You can optionally use an import or export route map to provide more fine-grained control over the IP prefixes that are imported into the customer VPN site or exported out of the VPN site. You can use a route map to filter routes that are eligible for import or export in a VRF, based on the route target extended community attributes of the route. The route map might, for example, deny access to selected routes from a community that is on the import route target list.

SUMMARY STEPS

1. **configure terminal**
2. **feature-set mpls**
3. **feature-set mpls l3vpn**
4. **vrf context** *vrf-name*
5. **rd** *route-distinguisher*
6. **address-family { ipv4 | ipv6 } unicast**
7. **route-target** { **import** | **export** } *route-target-ext-community* **}**
8. **maximum routes** *max-routes* [**threshold** *value*] [**reinstall**]
9. **import** [**vrf default** *max-prefix*] **map** *route-map*
10. **show vrf** *vrf-name*
11. **copy running-config startup-config**

DETAILED STEPS

	Command or Action	Purpose

Step 1	**configure terminal**
	Example:
	`switch# configure terminal`
	`switch(config)#`
		Enters global configuration mode.

Step 2	**feature-set mpls**
	Example:
	`switch(config)# feature-set mpls`
	`switch(config)#`
		Enables the MPLS feature-set.

Step 3	**feature-set mpls l3vpn**
	Example:
	`switch(config)# feature-set mpls l3vpn`
	`switch(config)#`
		Enables the MPLS Layer 3 VPN feature.

Step 4	**vrf context** *vrf-name* **Example:**	Defines the VPN routing instance by assigning a VRF name and enters VRF configuration mode. The vrf-

	Command or Action	Purpose

	`switch(config)# vrf context vpn1`
	
	`switch(config-vrf)#`
		name
		argument is any case-sensitive, alphanumeric string up to
		32 characters.

Step 5	**rd** *route-distinguisher*
	
	Example:
	
	`switch(config-vrf)# rd 1.2:1`
	
	`switch(config-vrf)#`
		Configures the route distinguisher. The route-distinguisher
		argument adds an 8-byte value to an IPv4 prefix to create
		a VPN IPv4 prefix. You can enter an RD in either of these
		formats:
		• 16-bit or 32-bit AS number: your 32-bit number, for
		example, 1.2:3
		• 32-bit IP address: your 16-bit number, for example,
		192.0.2.1:1

Step 6	**address-family { ipv4 | ipv6 } unicast**
	
	Example:
	
	`switch(config-vrf)# address-family ipv4 unicast`
	`switch(config-vrf-af-ipv4)#`
		Specifies the IPv4 address family type and enters address
		family configuration mode.

Step 7	**route-target** { **import** | **export** }
	
	route-target-ext-community **}**
	Example:
	
	`switch(config-vrf-af-ipv4)# route-target import`
	`1.0:1`
		Specifies a route-target extended community for a VRF
		as follows:
		• The import keyword imports routing information from
		the target VPN extended community.
		• The export keyword exports routing information to
		the target VPN extended community.
		• The route-target-ext-community argument adds the
		route-target extended community attributes to the
		VRF's list of import or export route-target extended
		communities. You can enter the
		route-target-ext-community argument in either of
		these formats:
		• 16-bit or 32-bit AS number: your 32-bit number,

	Command or Action	Purpose
		for example, 1.2:3 • 32-bit IP address: your 16-bit number, for example, 192.0.2.1:1
Step 8	**maximum routes** *max-routes* [**threshold** *value*] [**reinstall**] **Example:** `switch(config-vrf-af-ipv4)# maximum routes 10000`	(Optional) Configures the maximum number of routes that can be stored in the VRF route table. The max-routes range is from 1 to 4294967295. The threshold value range is from 1 to 100.
Step 9	**import** [**vrf default** *max-prefix*] **map** *route-map* **Example:** `switch(config-vrf-af-ipv4)# import vrf default map vpn1-route-map`	(Optional) Configures an import policy for a VRF to import prefixes from the default VRF as follows: • The max-prefix range is from 1 to 2147483647. The default is 1000 prefixes. • The route-map argument specifies the route map to be used as an import route map for the VRF and can be any case-sensitive, alphanumeric string up to 63 characters.
Step 10	**show vrf** *vrf-name* **Example:** `switch(config-vrf-af-ipv4)# show vrf vpn1`	(Optional) Displays information about a VRF. The vrf-name argument is any case-sensitive, alphanumeric string up to 32 characters.
Step 11	**copy running-config startup-config** **Example:** `switch(config-router-vrf)# copy running-config startup-config`	(Optional) Copies the running configuration to the startup configuration.

Configuring VRF Interfaces on PE Routers for Each VPN Customer

You can associate a virtual routing and forwarding instance (VRF) with an interface or subinterface on the PE routers.

SUMMARY STEPS

1. **configure terminal**
2. **interface** *type number*

3. **vrf member** *vrf-name*
4. **show vrf** *vrf-name* **interface**
5. **copy running-config startup-config**

DETAILED STEPS

	Command or Action	Purpose

Step 1	**configure terminal**
	Example:
	`switch# configure terminal`
	`switch(config)#`
		Enters global configuration mode.

Step 2	**interface** *type number*
	Example:
	`switch(config)# interface Ethernet 5/0`
	`switch(config-if)#`
		Specifies the interface to configure and enters interface
		configuration mode as follows:
		• The type argument specifies the type of interface to
		be configured.
		• The number argument specifies the port, connector,
		or interface card number.

Step 3	**vrf member** *vrf-name* **Example:** `switch(config-if)# vrf member vpn1`	Associates a VRF with the specified interface or subinterface. The vrf-name argument is the name assigned to a VRF.
Step 4	**show vrf** *vrf-name* **interface** **Example:** `switch(config-if)# show vrf vpn1 interface`	(Optional) Displays information about interfaces associated with a VRF. The vrf-name argument is any case-sensitive alphanumeric string up to 32 characters.
Step 5	**copy running-config startup-config** **Example:** `switch(config-router-vrf)# copy running-config startup-config`	(Optional) Copies the running configuration to the startup configuration.

Configuring Routing Protocols Between the PE and CE Routers

Configuring Static or Directly Connected Routes Between the PE and CE Routers

You can configure the PE router for PE-to-CE routing sessions that use static routes.

SUMMARY STEPS

1. **configure terminal**
2. **vrf context** *vrf-name*
3. { **ip ipv6** } **route** *prefix nexthop*

4. **address-family { ipv4 | ipv6 } unicast**
5. **feature bgp** *as - number*
6. **router bgp** *as - number*
7. **vrf** *vrf-name*
8. **address-family { ipv4 | ipv6 } unicast**
9. **redistribute static route-map** *map-name*
10. **redistribute direct route-map** *map-name*
11. **show { ipv4 | ipv6 } route vrf** *vrf-name*
12. **copy running-config startup-config**

DETAILED STEPS

	Command or Action	Purpose

Step 1	**configure terminal**
	Example:
	`switch# configure terminal`
	`switch(config)#`
		Enters global configuration mode.

Step 2	**vrf context** *vrf-name*
	Example:
	`switch(config)# vrf context vpn1`
	`switch(config-vrf)#`
		Defines the VPN routing instance by assigning a VRF
		name and enters VRF configuration mode. The vrf-name
		argument is any case-sensitive, alphanumeric string up to
		32 characters.

Step 3	{ **ip ipv6** } **route** *prefix nexthop*
	Example:
	`switch(config-vrf)# ip route 192.0.2.1/28 ethernet`
	`2/1`
		Defines static route parameters for every PE-to-CE session.
		The prefix and nexthop are as follows:
		• IPv4—in dotted decimal notation
		• IPv6—in hex format.

Step 4	**address-family { ipv4 | ipv6 } unicast**
	Example:
	`switch(config-vrf)# address-family ipv4 unicast`
	`switch(config-vrf-af)#`
		Specifies the IPv4 address family type and enters address
		family configuration mode.

Step 5	**feature bgp** *as - number*
	Example:
	`switch(config-vrf-af)# feature bgp`
	`switch(config)#`
		Enables the BGP feature.

Step 6	**router bgp** *as - number* **Example:**	Configures a BGP routing process and enters router configuration mode. The as-number argument indicates

	Command or Action	Purpose
	`switch(config)# router bgp 1.1`	the number of an autonomous system that identifies the router to other BGP routers and tags the routing information. The AS number can be a 16-bit integer or a 32-bit integer in the form of a higher 16-bit decimal number and a lower 16-bit decimal number in xx.xx format.

Step 7	**vrf** *vrf-name*
	Example:
	`switch(config-router)# vrf vpn1`
	`switch(config--router-vrf)#`
		Associates the BGP process with a VRF.
		The vrf-name argument is any case-sensitive, alphanumeric
		string up to 32 characters.

Step 8	**address-family { ipv4 | ipv6 } unicast**
	Example:
	`switch(config-vrf)# address-family ipv4 unicast`
	`switch(config-vrf-af)#`
		Specifies the IPv4 address family type and enters address
		family configuration mode.

Step 9	**redistribute static route-map** *map-name* **Example:** `switch(config-router-vrf-af)# redistribute static route-map StaticMap`	Redistributes static routes into BGP. The map-name can be any case-sensitive, alphanumeric string up to 63 characters.
Step 10	**redistribute direct route-map** *map-name* **Example:** `switch(config-router-vrf-af)# redistribute direct route-map StaticMap`	Redistributes directly connected routes into BGP. The map-name can be any case-sensitive, alphanumeric string up to 63 characters.
Step 11	**show { ipv4 | ipv6 } route vrf** *vrf-name* **Example:** `switch(config-router-vrf-af)# show ip ipv4 route vrf vpn1`	(Optional) Displays information about routes. The vrf-name argument is any case-sensitive, alphanumeric string up to 32 characters.
Step 12	**copy running-config startup-config** **Example:**	(Optional) Copies the running configuration to the startup configuration.

	Command or Action	Purpose
	`switch(config-router-vrf)# copy running-config startup-config`	

Configuring BGP as the Routing Protocol Between the PE and CE Routers

You can use eBGP to configure the PE router for PE-to-CE routing sessions.

SUMMARY STEPS

1. **configure terminal**
2. **feature bgp**
3. **router bgp** *as - number*
4. **vrf** *vrf-name*
5. **neighbor** *ip-address***remote-as** *as-number*
6. **address-family { ipv4 | ipv6 } unicast**
7. **show bgp { vpnv4 | vpnv6 } unicast neighbors vrf** *vrf-name*
8. **copy running-config startup-config**

DETAILED STEPS

	Command or Action	Purpose

Step 1	**configure terminal**
	Example:
	`switch# configure terminal`
	`switch(config)#`
		Enters global configuration mode.

Step 2	**feature bgp**
	Example:
	`switch(config)# feature bgp`
	`switch(config)#`
		Enables the BGP feature.

Step 3	**router bgp** *as - number*
	Example:
	`switch(config)# router bgp 1.1`
	`switch(config-router)#`
		Configures a BGP routing process and enters router configuration mode.
		The as-number argument indicates the number of an autonomous system that identifies the router to other BGP routers and tags the routing information passed along. The AS number can be a 16-bit integer or a 32-bit integer in the form of a higher 16-bit decimal number and a lower 16-bit decimal number in xx.xx format.

Step 4	**vrf** *vrf-name* **Example:** `switch(config-router)# vrf vpn1`	Associates the BGP process with a VRF. The vrf-name argument is any case-sensitive, alphanumeric string up to 32 characters.

	Command or Action	Purpose
	`switch(config--router-vrf)#`	

Step 5	**neighbor** *ip-address***remote-as** *as-number*
	
	Example:
	`switch(config-router)# neighbor 209.165.201.1`
	`remote-as 1.1`
	`switch(config-router-neighbor)#`
		Adds an entry to the iBGP neighbor table. The ip-address
		argument specifies the IP address of the neighbor in dotted
		decimal notation. The as-number argument specifies the
		autonomous system to which the neighbor belongs.

Step 6	**address-family { ipv4 | ipv6 } unicast**
	
	Example:
	`switch(config-vrf)# address-family ipv4`
	`unicast`
	`switch(config-vrf-af)#`
		Enters address family configuration mode for configuring
		routing sessions, such as BGP, that use standard IPv4 or
		IPv6 address prefixes.

Step 7	**show bgp { vpnv4 | vpnv6 } unicast neighbors vrf**
	
	vrf-name
	
	Example:
	`switch(config-router-neighbor-af)# show bgp`
	`vpnv4`
	` unicast neighbors`
		(Optional) Displays information about BGP neighbors. The
		vrf-name argument is any case-sensitive alphanumeric string
		up to 32 characters.

Step 8	**copy running-config startup-config**
	
	Example:
	`switch(config-router-vrf)# copy running-`
	`config`
	`startup-config`
		(Optional) Copies the running configuration to the startup
		configuration.

Configuring RIPv2 Between the PE and CE Routers

You can use RIP to configure the PE router for PE-to-CE routing sessions.

SUMMARY STEPS

1. **configure terminal**
2. **feature rip**
3. **router rip** *instance-tag*
4. **vrf** *vrf-name*
5. **address-family ipv4 unicast**
6. **redistribute** { **bgp** as | **direct** | { **egrip** | **ospf** | **rip** } *instance-tag* | static } **route-map** *map-name vrf-name*
7. **show ip rip vrf** *vrf-name*
8. **copy running-config startup-config**

DETAILED STEPS

	Command or Action	Purpose

Step 1	**configure terminal**
	Example:
	`switch# configure terminal`
	`switch(config)#`
		Enters global configuration mode.

Step 2	**feature rip**
	Example:
	`switch(config)# feature rip`
	`switch(config)#`
		Enables the RIP feature.

Step 3	**router rip** *instance-tag* **Example:** `switch(config)# router rip Test1`	Enables RIP and enters router configuration mode. The instance-tag can be any case-sensitive, alphanumeric string up to 20 characters.

Step 4	**vrf** *vrf-name*
	Example:
	`switch(config-router)# vrf vpn1`
	`switch(config--router-vrf)#`
		Associates the RIP process with a VRF.
		The vrf-name argument is any case-sensitive, alphanumeric
		string up to 32 characters.

Step 5	**address-family ipv4 unicast**
	Example:
	`switch(config-router-vrf)# address-family ipv4 unicast`
	`switch(config-router-vrf-af)#`
		Specifies the address family type and enters address family
		configuration mode.

Step 6	**redistribute** { **bgp** as | **direct** | { **egrip** | **ospf** | **rip** } *instance-tag* | static } **route-map** *map-name vrf-name* **Example:** `sswitch(config-router-vrf-af)# show ip rip vrf vpn1`	Redistributes routes from one routing domain into another routing domain. The as number can be a 16-bit integer or a 32-bit integer in the form of a higher 16-bit decimal number and a lower 16-bit decimal number in xx.xx format. The instance-tag can be any case-sensitive alphanumeric string up to 20 characters
Step 7	**show ip rip vrf** *vrf-name*	(Optional) Displays information about RIP.

	Command or Action	Purpose
	Example: `sswitch(config-router-vrf-af)# show ip rip vrf vpn1`	The vrf-name argument is any case-sensitive, alphanumeric string up to 32 characters.
Step 8	**copy running-config startup-config** **Example:** `switch(config-router-vrf)# copy running-config startup-config`	(Optional) Copies the running configuration to the startup configuration.

Configuring OSPF Between the PE and CE Routers

You can use OSPFv2 to configure the PE router for PE-to-CE routing sessions. You can optionally create an OSPF sham link if you have OSPF back door links that are not part of the MPLS network.

SUMMARY STEPS

1. **configure terminal**
2. **feature ospf**
3. **router ospf** *instance-tag*
4. **vrf** *vrf-name*
5. **area** area-id **sham-link** *source-address destination-address*
7. **address-family { ipv4 | ipv6 } unicast**
8. **redistribute** { **bgp** as | **direct** | { **egrip** | **ospf** | **rip** } *instance-tag* | static } **route-map** *map-name*
9. **autonomous-system** *as-number*
11. **show ip egrip vrf** *vrf-name*
12. **copy running-config startup-config**

DETAILED STEPS

	Command or Action	Purpose

Step 1	**configure terminal**
	Example:
	`switch# configure terminal`
	`switch(config)#`
		Enters global configuration mode.

Step 2	**feature ospf**
	Example:
	`switch(config)# feature ospf`
	`switch(config)#`
		Enables the OSPF feature.

Step 3	**router ospf** *instance-tag* **Example:** `switch(config)# router ospf Test1`	Enables OSPF and enters router configuration mode. The instance-tag can be any case-sensitive, alphanumeric string up to 20 characters.
Step 4	**vrf** *vrf-name*	Enters router VRF configuration mode.

	Command or Action	Purpose

	Example:
	`switch(config-router)# vrf vpn1`
	`switch(config--router-vrf)#`
		The vrf-name argument is any case-sensitive, alphanumeric
		string up to 32 characters.

Step 5	**area** area-id **sham-link** *source-address destination-address*
	Example:
	`switch(config-router-vrf)# area 1 sham-link`
	`10.2.1.1 10.2.1.2`
		(Optional) Configures the sham link on the PE interface
		within a specified OSPF area and with the loopback
		interfaces specified by the IP addresses as endpoints.
		You must configure the sham link at both PE endpoints.

Step 7	**address-family { ipv4 | ipv6 } unicast**
	Example:
	`switch(config-router)# address-family ipv4 unicast`
	`switch(config-router-vrf-af)#`
		Specifies the address family type and enters address family
		configuration mode.

Step 8	**redistribute** { **bgp** as | **direct** | { **egrip** | **ospf** | **rip** }
	instance-tag | static } **route-map** *map-name*
	Example:
	`switch(config-router-vrf-af)# redistribute bgp`
	`1.0 route-map BGPMap`
		Redistributes BGP into the EIGRP.
		The autonomous system number of the BGP network is
		configured in this step. BGP must be redistributed into
		EIGRP for the CE site to accept the BGP routes that carry
		the EIGRP information. A metric must also be specified
		for the BGP network.
		The map-name can be any case-sensitive, alphanumeric
		string up to 63 characters.

Step 9	**autonomous-system** *as-number*
	Example:
	`switch(config-router-vrf-af)#`
	`autonomous-system 1.3`
		(Optional) Specifies the autonomous system number for
		this address family for the customer site.
		The as-number argument indicates the number of an
		autonomous system that identifies the router to other BGP
		routers and tags the routing information passed along. The
		AS number can be a 16-bit integer or a 32-bit integer in
		the form of a higher 16-bit decimal number and a lower

	Command or Action	Purpose
		16-bit decimal number in xx.xx format.
Step 10	**show ip egrip vrf** *vrf-name* **Example:** `switch(config-router-vrf-af)# show ipv4 eigrp vrf vpn1`	(Optional) Displays information about EIGRP in this VRF. The vrf-name can be any case-sensitive, alphanumeric string up to 32 characters
Step 11	**copy running-config startup-config** **Example:** `switch(config-router-vrf)# copy running-config startup-config`	(Optional) Copies the running configuration to the startup configuration.

Configuring EIGRP Between the PE and CE Routers

You can configure the PE router to use Enhanced Interior Gateway Routing Protocol (EIGRP) between the PE and CE routers to transparently connect EIGRP customer networks through an MPLS-enabled BGP core network so that EIGRP routes are redistributed through the VPN across the BGP network as internal BGP (iBGP) routes.

Before you begin

You must configure BGP in the network core.

SUMMARY STEPS

1. **configure terminal**
2. **feature egrip**
3. **router egrip** *instance-tag*
4. **vrf** *vrf-name*
5. **address-family ipv4 unicast**
6. **redistribute** { **bgp** as-number**route-map** *map-name*
7. **show ip ospf** *instance-tag* **vrf** *vrf-name*
8. **copy running-config startup-config**

DETAILED STEPS

	Command or Action	Purpose

Step 1	**configure terminal**
	Example:
	`switch# configure terminal`
	`switch(config)#`
		Enters global configuration mode.

Step 2	**feature egrip**
	Example:
	`switch(config)# feature egrip`
	`switch(config)#`
		Enables the EGRIP feature.

	Command or Action	Purpose
Step 3	**router egrip** *instance-tag* **Example:** `switch(config)# router egrip Test1`	Configures an EIGRP instance and enters router configuration mode. The instance-tag can be any case-sensitive, alphanumeric string up to 20 characters.

Step 4	**vrf** *vrf-name*
	Example:
	`switch(config-router)# vrf vpn1`
	`switch(config--router-vrf)#`
		Enters router VRF configuration mode.
		The vrf-name argument is any case-sensitive, alphanumeric
		string up to 32 characters.

Step 5	**address-family ipv4 unicast**
	Example:
	`switch(config-router-vrf)# address-family ipv4 unicast`
	`switch(config-router-vrf-af)#`
		(Optional) Enters address family configuration mode for
		configuring routing sessions that use standard IPv4 address
		prefixes.

Step 6	**redistribute** { **bgp** as-number**route-map** *map-name* **Example:** `sswitch(config-router-vrf-af)# show ip rip vrf vpn1`	Redistributes routes from one routing domain into another routing domain. The as number can be a 16-bit integer or a 32-bit integer in the form of a higher 16-bit decimal number and a lower 16-bit decimal number in xx.xx format. The instance-tag can be any case-sensitive alphanumeric string up to 20 characters
Step 7	**show ip ospf** *instance-tag* **vrf** *vrf-name* **Example:** `switch(config-router-vrf-af)# show ip rip vrf vpn1`	(Optional) Displays information about OSPF.
Step 8	**copy running-config startup-config** **Example:** `switch(config-router-vrf)# copy running-config startup-config`	(Optional) Copies the running configuration to the startup configuration.

Configuring PE-CE Redistribution in BGP for the MPLS VPN

You must configure BGP to distribute the PE-CE routing protocol on every PE router that provides MPLS Layer 3 VPN services if the PE-CE protocol is not BGP.

SUMMARY STEPS

1. **configure terminal**
2. **feature bgp**
3. **router bgp** *instance-tag*
4. **router id** *ip-address*
5. **router id** *ip-address* **remote-as** *as-number*
6. **update-source loopback [0 | 1]**
7. **address-family { ipv4 | ipv6 } unicast**
8. **send-community extended**
9. **vrf** *vrf-name*
10. **address-family { ipv4 | ipv6 } unicast**
11. **redistribute** { **direct** | { **egrip** | **ospfv3** | **ospfv3** |**rip** } *instance-tag* | static } **route-map** *map-name*
12. **show bgp { ipv4 | ipv6 } unicast vrf** *vrf-name*
13. **copy running-config startup-config**

DETAILED STEPS

	Command or Action	Purpose

Step 1	**configure terminal**
	Example:
	`switch# configure terminal`
	`switch(config)#`
		Enters global configuration mode.

Step 2	**feature bgp**
	Example:
	`switch(config)# feature bgp`
	`switch(config)#`
		Enables the BGP feature.

Step 3	**router bgp** *instance-tag*
	Example:
	`switch(config)# router bgp 1.1`
	`switch(config-router)#`
		Configures a BGP routing process and enters router configuration mode. The as-number argument indicates
		the number of an autonomous system that identifies the
		router to other BGP routers and tags the routing information passed along. The AS number can be a 16-bit
		integer or a 32-bit integer in the form of a higher 16-bit
		decimal number and a lower 16-bit decimal number in
		xx.xx format.

Step 4	**router id** *ip-address* **Example:** `switch(config-router)# router-id 192.0.2.255 1`	(Optional) Configures the BGP router ID. This IP address identifies this BGP speaker. This command triggers an automatic notification and session reset for the BGP neighbor sessions.

	Command or Action	Purpose
	switch(config-router)#	

Step 5	**router id** *ip-address* **remote-as** *as-number*
	
	Example:
	
	`switch(config-router)# neighbor 209.165.201.1`
	
	`remote-as 1.2`
	`switch(config-router-neighbor)#`
		Adds an entry to the BGP or multiprotocol BGP neighbor
		table. The ip-address argument specifies the IP address of
		the neighbor in dotted decimal notation. The as-number
		argument specifies the autonomous system to which the
		neighbor belongs.

Step 6	**update-source loopback [0 | 1]**
	Example:
	`switch(config-router-neighbor)# update-source`
	`loopback 0#`
		Specifies the source address of the BGP session.

Step 7	**address-family { ipv4 | ipv6 } unicast**
	
	Example:
	
	`switch(config-router-neighbor)# address-family`
	`vpnv4`
	
	`switch(config-router-neighbor-af)#`
		Enters address family configuration mode for configuring
		routing sessions, such as BGP, that use standard VPNv4
		or VPNv6 address prefixes. The optional unicast keyword
		specifies VPNv4 or VPNv6 unicast address prefixes.

Step 8	**send-community extended**
	
	Example:
	`switch(config-router-neighbor-af)# send-community`
	`extended`
		Specifies that a communities attribute should be sent to a
		BGP neighbor.

Step 9	**vrf** *vrf-name*
	Example:
	
	`switch(config-router-neighbor-af)# vrf vpn1`
	
	`switch(config-router-vrf)#`
		Enters router VRF configuration mode.
		The vrf-name argument is any case-sensitive, alphanumeric
		string up to 32 characters.

Step 10	**address-family { ipv4 | ipv6 } unicast**
	
	Example:
	
	`switch(config-router-vrf)# address-family ipv4`
	`unicast`
		Enters address family configuration mode for configuring
		routing sessions that use standard IPv4 or IPv6 address
		prefixes.

	Command or Action	Purpose
	`switch(config-router-vrf-af)#`	

Step 11	**redistribute** { **direct** | { **egrip** | **ospfv3** | **ospfv3** |**rip** } *instance-tag* | static } **route-map** *map-name*
	
	Example:
	
	`switch(config-router-af-vrf)# redistribute eigrp`
	` Test2 route-map EigrpMap`
		Redistributes routes from one routing domain into another
		routing domain. The as number can be a 16-bit integer or
		a 32-bit integer in the form of a higher 16-bit decimal
		number and a lower 16-bit decimal number in xx.xx
		format. The instance-tag can be any case-sensitive,
		alphanumeric string up to 20 characters. The map-name
		can be any case-sensitive alphanumeric string up to 63
		characters.

Step 12	**show bgp { ipv4 | ipv6 } unicast vrf** *vrf-name*
	
	Example:
	
	`switch(config-router--vrf-af)# show bgp ipv4`
	`unicast vrf vpn1vpn1`
		(Optional) Displays information about BGP. The vrf-name
		argument is any case-sensitive, alphanumeric string up to
		32 characters.

Step 13	**copy running-config startup-config**
	
	Example:
	
	`switch(config-router-vrf)# copy running-config`
	`startup-config`
		(Optional) Copies the running configuration to the startup
		configuration.

Configuring a Hub-and-Spoke Topology

Configuring VRFs on the Hub PE Router

You can configure hub and spoke VRFs on the hub PE router.

SUMMARY STEPS

1. **configure terminal**
2. **feature-set mpls**
3. **feature-set mpls l3vpn**
4. **vrf context** *vrf-hub*
5. **rd** *route-distinguisher*
6. **address-family { ipv4 | ipv6 } unicast**
7. **route-target** { **import** | **export** } route-target-ext-community **}**
8. **vrf context** *vrf-spoke*
9. **address-family { ipv4 | ipv6 } unicast**
10. **route-target** { **import** | **export** } *route-target-ext-community* **}**
11. **show running-config vrf** *vrf-name*

12. **copy running-config startup-config**

DETAILED STEPS

	Command or Action	Purpose

Step 1	**configure terminal**
	Example:
	`switch# configure terminal`
	`switch(config)#`
		Enters global configuration mode.

Step 2	**feature-set mpls**
	Example:
	`switch(config)# feature-set mpls`
	`switch(config)#`
		Enables the MPLS feature-set.

Step 3	**feature-set mpls l3vpn**
	Example:
	`switch(config)# feature-set mpls l3vpn`
	`switch(config)#`
		Enables the MPLS Layer 3 VPN feature.

Step 4	**vrf context** *vrf-hub*
	Example:
	`switch(config)# vrf context 2hub`
	`switch(config-vrf)#`
		Defines the VPN routing instance for the PE hub by
		assigning a VRF name and enters VRF configuration mode.
		The vrf-hub argument is any case-sensitive alphanumeric
		string up to 32 characters.

Step 5	**rd** *route-distinguisher*
	Example:
	`switch(config-vrf)# rd 1.2:1`
	`switch(config-vrf)#`
		Configures the route distinguisher. The route-distinguisher
		argument adds an 8-byte value to an IPv4 prefix to create
		a VPN IPv4 prefix. You can enter an RD in either of these
		formats:
		• 16-bit or 32-bit AS number: your 32-bit number, for example, 1.2:3
		• 32-bit IP address: your 16-bit number, for example, 192.0.2.1:1

Step 6	**address-family { ipv4 | ipv6 } unicast**
	Example:
	`switch(config-vrf)# address-family ipv4 unicast`
	`switch(config-vrf-af-ipv4)#`
		Specifies the IPv4 address family type and enters address
		family configuration mode.

	Command or Action	Purpose
Step 7	**route-target** { **import** | **export** } route-target-ext-community **}** **Example:** `switch(config-vrf-af-ipv4)# route-target import 1.0:1`	Specifies a route-target extended community for a VRF as follows: • The **import** keyword imports routing information from the target VPN extended community. • The **export** keyword exports routing information to the target VPN extended community. • The route-target-ext-community argument adds the route-target extended community attributes to the VRF's list of import or export route-target extended communities. You can enter the route-target-ext-community argument in either of these formats: • 16-bit or 32-bit AS number: your 32-bit number, for example, 1.2:3 • 32-bit IP address: your 16-bit number, for example, 192.0.2.1:1

Step 8	**vrf context** *vrf-spoke*
	
	Example:
	`switch(config-vrf-af-ipv4)# vrf context 2spokes`
	`switch(config-vrf)#`
		Defines the VPN routing instance for the PE spoke by assigning a VRF name and enters VRF configuration mode.
		The vrf-spoke argument is any case-sensitive, alphanumeric string up to 32 characters.

Step 9	**address-family { ipv4 | ipv6 } unicast**
	
	Example:
	`switch(config-vrf)# address-family ipv4 unicast`
	`switch(config-vrf-af-ipv4)#`
		Specifies the IPv4 address family type and enters address family configuration mode.

Step 10	**route-target** { **import** | **export** } *route-target-ext-community* **}** **Example:** `switch(config-vrf-af-ipv4)# route-target export`	Specifies a route-target extended community for a VRF as follows: • Creates a route-target extended community for a VRF. The **import** keyword imports routing

	Command or Action	Purpose
	`1:100`	information from the target VPN extended community. The **export** keyword exports routing information to the target VPN extended community. The route-target-ext-community argument adds the route-target extended community attributes to the VRF's list of import or export route-target extended communities. You can enter the route-target-ext-community argument in either of these formats: • 16-bit or 32-bit AS number: your 32-bit number, for example, 1.2:3 • 32-bit IP address: your 16-bit number, for example, 192.0.2.1:1
Step 11	**show running-config vrf** *vrf-name* **Example:** `switch(config-vrf-af-ipv4)# show running-config vrf 2spokes`	(Optional) Displays the running configuration for the VRF. The vrf-name argument is any case-sensitive, alphanumeric string up to 32 characters. .
Step 12	**copy running-config startup-config** **Example:** `switch(config-router-vrf)# copy running-config startup-config`	(Optional) Copies the running configuration to the startup configuration.

Configuring eBGP on the Hub PE Router

You can use eBGP to configure PE-to-CE hub routing sessions.

SUMMARY STEPS

1. **configure terminal**
2. **feature-set mpls**
3. **feature mpls l3vpn**
4. **feature bgp**
5. **router bgp** *as - number*
6. **neighbor** *ip-address***remote-as** *as-number*
7. **address-family { ipv4 | ipv6 } unicast**

8. **send-community extended**
9. **vrf** *vrf-hub*
10. **neighbor** *ip-address***remote-as** *as-number*
11. **address-family { ipv4 | ipv6 } unicast**
12. **as-override**
13. **vrf** *vrf-spoke*
14. **neighbor** *ip-address***remote-as** *as-number*
15. **address-family { ipv4 | ipv6 } unicast**
16. **allowas-in** [*number*]
17. **show running-config bgp** *vrf-name*
18. **copy running-config startup-config**

DETAILED STEPS

	Command or Action	Purpose

Step 1	**configure terminal**
	Example:
	`switch# configure terminal`
	`switch(config)#`
		Enters global configuration mode.

Step 2	**feature-set mpls** **Example:** `switch(config)# feature-set mpls`	Enables the MPLS feature-set.
Step 3	**feature mpls l3vpn** **Example:** `switch(config)# feature mpls l3vpn`	Enables the MPLS Layer 3 VPN feature.

Step 4	**feature bgp**
	Example:
	`switch(config)# feature bgp`
	`switch(config)#`
		Enables the BGP feature.

Step 5	**router bgp** *as - number*
	Example:
	`switch(config)# router bgp 1.1`
	`switch(config-router)#`
		Configures a BGP routing process and enters router configuration mode.
		The *as-number* argument indicates the number of an
		autonomous system that identifies the router to other BGP
		routers and tags the routing information passed along. The
		AS number can be a 16-bit integer or a 32-bit integer in
		the form of a higher 16-bit decimal number and a lower
		16-bit decimal number in xx.xx format.

Step 6	**neighbor** *ip-address***remote-as** *as-number*	Adds an entry to the iBGP neighbor table.

	Command or Action	Purpose

	Example:
	`switch(config-router)# neighbor 209.165.201.1 remote-as 1.2`
	`switch(config-router-neighbor)#`
		• The ip-address argument specifies the IP address of the neighbor in dotted decimal notation.
		• The as-number argument specifies the autonomous system to which the neighbor belongs.

Step 7	**address-family { ipv4 | ipv6 } unicast**
	Example:
	`switch(config-router-vrf-neighbor)# address-family ipv4 unicast`
	`switch(config-router-neighbor-af)#`
		Specifies the IP address family type and enters address family configuration mode.

Step 8	**send-community extended** **Example:** `switch(config-router-neighbor-af)# send-community extended`	(Optional) Configures BGP to advertise extended community lists.

Step 9	**vrf** *vrf-hub*
	Example:
	`switch(config-router-neighbor-af)# vrf 2hub`
	`switch(config-router-vrf)#`
		Enters VRF configuration mode. The *vrf-hub* argument is any case-sensitive, alphanumeric string up to 32 characters.

Step 10	**neighbor** *ip-address***remote-as** *as-number*
	Example:
	`switch(config-router-vrf)# neighbor 33.0.0.33 1 remote-as 150`
	`switch(config-router-vrf-neighbor)#`
		Adds an entry to the BGP or multiprotocol BGP neighbor table for this VRF.
		• The ip-address argument specifies the IP address of the neighbor in dotted decimal notation.
		• The as-number argument specifies the autonomous system to which the neighbor belongs.

Step 11	**address-family { ipv4 | ipv6 } unicast**
	Example:
	`switch(config-router-vrf-neighbor)# address-family ipv4 unicast`
	`switch(config-router--vrf-neighbor-af)#`
		Specifies the IP address family type and enters address family configuration mode.

Step 12	**as-override**	(Optional) Overrides the AS-number when sending

	Command or Action	Purpose
	Example: `switch(config-router-vrf-neighbor-af)# as-override`	an update. If all BGP sites are using the same AS number, coof the cfollowing commands: • Configure the BGP as-override command at the PE (hub) or • Configure the allowas-in command at the receiving CE router.

Step 13	**vrf** *vrf-spoke*
	Example:
	`switch(config-router-vrf-neighbor-af)# vrf 2spokes`
	`switch(config-router-vrf)#`
		Enters VRF configuration mode. The vrf-spoke argument
		is any case-sensitive, alphanumeric string up to 32 characters.

Step 14	**neighbor** *ip-address***remote-as** *as-number*
	Example:
	`switch(config-router-vrf)# neighbor 33.0.0.33 1 remote-as 150`
	`switch(config-router-vrf-neighbor)#`
		Adds an entry to the BGP or multiprotocol BGP neighbor
		table for this VRF.
		• The ip-address argument specifies the IP address of the neighbor in dotted decimal notation.
		• The as-number argument specifies the autonomous system to which the neighbor belongs.

Step 15	**address-family { ipv4 | ipv6 } unicast**
	Example:
	`switch(config-router-vrf-neighbor)# address-family ipv4 unicast`
	`switch(config-router--vrf-neighbor-af)#`
		Specifies the IP address family type and enters address
		family configuration mode.

Step 16	**allowas-in** [*number*] **Example:** `switch(config-router-vrf-neighbor-af)# allowas-in 3`	(Optional) Allows duplicate AS numbers in the AS path. Configure this parameter in the VPN address family configuration mode at the PE spokes and at the neighbor mode at the PE hub.
Step 17	**show running-config bgp** *vrf-name*	(Optional) Displays the running configuration for BGP.

	Command or Action	Purpose
	Example: `switch(config-router-vrf-neighbor-af)# show running-config bgp`	
Step 18	**copy running-config startup-config** **Example:** `switch(config-router-vrf)# copy running-config startup-config`	(Optional) Copies the running configuration to the startup configuration.

Configuring eBGP on the Hub CE Router

You can use eBGP to configure PE-to-CE hub routing sessions.

- Configure either the as-override command at the PE (hub) or the **allowas-in** command at the receiving CE router.
- Configure the disable-peer-as-check command at the CE router.
- To advertise BGP routes learned from one ASN back to the same ASN, configure the disable-peer-as-check command at the PE router to prevent loopback.

SUMMARY STEPS

1. **configure terminal**
2. **feature-set mpls**
3. **feature mpls l3vpn**
4. **feature bgp**
5. **router bgp** *as - number*
6. **neighbor** *ip-address***remote-as** *as-number*
7. **address-family { ipv4 | ipv6 } unicast**
8. **send-community extended**
9. **vrf** *vrf-hub*
10. **neighbor** *ip-address***remote-as** *as-number*
11. **address-family { ipv4 | ipv6 } unicast**
12. **as-override**
13. **vrf** *vrf-spoke*
14. **neighbor** *ip-address***remote-as** *as-number*
15. **address-family { ipv4 | ipv6 } unicast**
16. **allowas-in** [*number*]
17. **show running-config bgp** *vrf-name*
18. **copy running-config startup-config**

DETAILED STEPS

	Command or Action	Purpose

Step 1	**configure terminal**
	Example:
	`switch# configure terminal`
	`switch(config)#`
		Enters global configuration mode.

Step 2	**feature-set mpls**	Enables the MPLS feature-set.

	Command or Action	Purpose
	Example: `switch(config)# feature-set mpls`	
Step 3	**feature mpls l3vpn** **Example:** `switch(config)# feature mpls l3vpn`	Enables the MPLS Layer 3 VPN feature.

Step 4	**feature bgp**
	Example:
	`switch(config)# feature bgp`
	`switch(config)#`
		Enables the BGP feature.

Step 5	**router bgp** *as - number*
	Example:
	`switch(config)# router bgp 1.1`
	`switch(config-router)#`
		Configures a BGP routing process and enters router configuration mode.
		The *as-number* argument indicates the number of an autonomous system that identifies the router to other BGP routers and tags the routing information passed along. The AS number can be a 16-bit integer or a 32-bit integer in the form of a higher 16-bit decimal number and a lower 16-bit decimal number in xx.xx format.

Step 6	**neighbor** *ip-address* **remote-as** *as-number*
	Example:
	`switch(config-router)# neighbor 209.165.201.1 remote-as 1.2`
	`switch(config-router-neighbor)#`
		Adds an entry to the iBGP neighbor table.
		• The *ip-address* argument specifies the IP address of the neighbor in dotted decimal notation.
		• The as-number argument specifies the autonomous system to which the neighbor belongs.

Step 7	**address-family { ipv4 | ipv6 } unicast**
	Example:
	`switch(config-router-vrf-neighbor)# address-family ipv4 unicast`
	`switch(config-router-neighbor-af)#`
		Specifies the IP address family type and enters address family configuration mode.

Step 8	**send-community extended** **Example:** `switch(config-router-neighbor-af)# send-community extended`	(Optional) Configures BGP to advertise extended community lists.

	Command or Action	Purpose

Step 9	**vrf** *vrf-hub*
	
	Example:
	
	`switch(config-router-neighbor-af)# vrf 2hub`
	`switch(config-router-vrf)#`
		Enters VRF configuration mode. The *vrf-hub* argument is
		any case-sensitive, alphanumeric string up to 32 characters.

Step 10	**neighbor** *ip-address***remote-as** *as-number*
	
	Example:
	
	`switch(config-router-vrf)# neighbor 33.0.0.33 1`
	`remote-as 150`
	`switch(config-router-vrf-neighbor)#`
		Adds an entry to the BGP or multiprotocol BGP neighbor
		table for this VRF.
		• The ip-address argument specifies the IP address of
		the neighbor in dotted decimal notation.
		• The as-number argument specifies the autonomous
		system to which the neighbor belongs.

Step 11	**address-family { ipv4 | ipv6 } unicast**
	
	Example:
	
	`switch(config-router-vrf-neighbor)# address-family`
	` ipv4 unicast`
	`switch(config-router--vrf-neighbor-af)#`
		Specifies the IP address family type and enters address
		family configuration mode.

Step 12	**as-override** **Example:** `switch(config-router-vrf-neighbor-af)# as-override`	(Optional) Overrides the AS-number when sending an update. If all BGP sites are using the same AS number, coof the cfollowing commands: • Configure the BGP as-override command at the PE (hub) or • Configure the allowas-in command at the receiving CE router.

Step 13	**vrf** *vrf-spoke*
	
	Example:
	
	`switch(config-router-vrf-neighbor-af)# vrf 2spokes`
	`switch(config-router-vrf)#`
		Enters VRF configuration mode. The vrf-spoke argument
		is any case-sensitive, alphanumeric string up to 32 characters.

Step 14	**neighbor** *ip-address***remote-as** *as-number*	Adds an entry to the BGP or multiprotocol BGP neighbor

	Command or Action	Purpose

	Example:
	`switch(config-router-vrf)# neighbor 33.0.0.33 1`
	`remote-as 150`
	`switch(config-router-vrf-neighbor)#`
		table for this VRF.
		• The *ip-address* argument specifies the IP address of the neighbor in dotted decimal notation.
		• The as-number argument specifies the autonomous system to which the neighbor belongs.

Step 15	**address-family { ipv4 | ipv6 } unicast**
	Example:
	`switch(config-router-vrf-neighbor)# address-family`
	` ipv4 unicast`
	`switch(config-router--vrf-neighbor-af)#`
		Specifies the IP address family type and enters address family configuration mode.

Step 16	**allowas-in** [*number*]
	Example:
	`switch(config-router-vrf-neighbor-af)# allowas-in`
	` 3`
		(Optional) Allows duplicate AS numbers in the AS path.
		Configure this parameter in the VPN address family configuration mode at the PE spokes and at the neighbor mode at the PE hub.

Step 17	**show running-config bgp** *vrf-name*
	Example:
	`switch(config-router-vrf-neighbor-af)# show`
	`running-config bgp`
		(Optional) Displays the running configuration for BGP.

Step 18	**copy running-config startup-config**
	Example:
	`switch(config-router-vrf)# copy running-config`
	`startup-config`
		(Optional) Copies the running configuration to the startup configuration.

Configuring VRFs on the Spoke PE Router

You can configure hub and spoke VRFs on the spoke PE router.

SUMMARY STEPS

1. **configure terminal**
2. **feature-set mpls**
3. **feature-set mpls l3vpn**
4. **vrf context** *vrf-spoke*
5. **rd** *route-distinguisher*
6. **address-family { ipv4 | ipv6 } unicast**

7. **route-target** { **import** | **export** } route-target-ext-community **}**
8. **show running-config vrf** *vrf-name*
9. **copy running-config startup-config**

DETAILED STEPS

	Command or Action	Purpose

Step 1	**configure terminal**
	Example:
	`switch# configure terminal`
	`switch(config)#`
		Enters global configuration mode.

Step 2	**feature-set mpls**
	Example:
	`switch(config)# feature-set mpls`
	`switch(config)#`
		Enables the MPLS feature-set.

Step 3	**feature-set mpls l3vpn**
	Example:
	`switch(config)# feature-set mpls l3vpn`
	`switch(config)#`
		Enables the MPLS Layer 3 VPN feature.

Step 4	**vrf context** *vrf-spoke*
	Example:
	`switch(config)# vrf context spoke`
	`switch(config-vrf)#`
		Defines the VPN routing instance for the PE spoke by
		assigning a VRF name and enters VRF configuration mode.
		The vrf-spoke argument is any case-sensitive, alphanumeric
		string up to 32 characters.

Step 5	**rd** *route-distinguisher*
	Example:
	`switch(config-vrf)# rd 1.101`
	`switch(config-vrf)#`
		Configures the route distinguisher. The route-distinguisher
		argument adds an 8-byte value to an IPv4 prefix to create
		a VPN IPv4 prefix. You can enter an RD in either of these
		formats:
		• 16-bit or 32-bit AS number: your 32-bit number, for example, 1.2:3
		• 32-bit IP address: your 16-bit number, for example, 192.0.2.1:1

Step 6	**address-family { ipv4 | ipv6 } unicast** **Example:** `switch(config-vrf)# address-family ipv4 unicast`	Specifies the IPv4 address family type and enters address family configuration mode.

	Command or Action	Purpose
	switch(config-vrf-af-ipv4)#	
Step 7	**route-target** { **import** | **export** } route-target-ext-community **}** **Example:** `switch(config-vrf-af-ipv4)# route-target import 1.0:1`	Specifies a route-target extended community for a VRF as follows: • The **import** keyword imports routing information from the target VPN extended community. • The **export** keyword exports routing information to the target VPN extended community. • The route-target-ext-community argument adds the route-target extended community attributes to the VRF's list of import or export route-target extended communities. You can enter the route-target-ext-community argument in either of these formats: • 16-bit or 32-bit AS number: your 32-bit number, for example, 1.2:3 • 32-bit IP address: your 16-bit number, for example, 192.0.2.1:1
Step 8	**show running-config vrf** *vrf-name* **Example:** `switch(config-vrf-af-ipv4)# show running-config vrf 2spokes`	(Optional) Displays the running configuration for the VRF. The vrf-name argument is any case-sensitive, alphanumeric string up to 32 characters. .
Step 9	**copy running-config startup-config** **Example:** `switch(config-router-vrf)# copy running-config startup-config`	(Optional) Copies the running configuration to the startup configuration.

Configuring eBGP on the Spoke PE Router

You can use eBGP to configure PE spoke routing sessions.

SUMMARY STEPS

1. **configure terminal**
2. **feature-set mpls**
3. **feature mpls l3vpn**
4. **feature bgp**
5. **router bgp** *as - number*
6. **neighbor** *ip-address* **remote-as** *as-number*
7. **address-family { ipv4 | ipv6 } unicast**
8. **allowas-in** *number*
9. **send-community extended**
10. **show running-config bgp**
11. **copy running-config startup-config**

DETAILED STEPS

	Command or Action	Purpose

Step 1	**configure terminal**
	Example:
	`switch# configure terminal`
	`switch(config)#`
		Enters global configuration mode.

Step 2	**feature-set mpls** **Example:** `switch(config)# feature-set mpls`	Enables the MPLS feature-set.
Step 3	**feature mpls l3vpn** **Example:** `switch(config)# feature mpls l3vpn`	Enables the MPLS Layer 3 VPN feature.

Step 4	**feature bgp**
	Example:
	`switch(config)# feature bgp`
	`switch(config)#`
		Enables the BGP feature.

Step 5	**router bgp** *as - number*
	Example:
	`switch(config)# router bgp 100`
	`switch(config-router)#`
		Configures a BGP routing process and enters router configuration mode.
		The *as-number* argument indicates the number of an
		autonomous system that identifies the router to other BGP
		routers and tags the routing information passed along. The
		AS number can be a 16-bit integer or a 32-bit integer in
		the form of a higher 16-bit decimal number and a lower
		16-bit decimal number in xx.xx format.

Step 6	**neighbor** *ip-address***remote-as** *as-number*	Adds an entry to the iBGP neighbor table.

	Command or Action	Purpose

	Example:
	`switch(config-router)# neighbor 63.63.0.63 remote-as 100`
	`switch(config-router-neighbor)#`
		• The ip-address argument specifies the IP address of the neighbor in dotted decimal notation.
		• The as-number argument specifies the autonomous system to which the neighbor belongs.

Step 7	**address-family { ipv4 | ipv6 } unicast**
	Example:
	`switch(config-router-vrf-neighbor)# address-family ipv4 unicast`
	`switch(config-router-neighbor-af)#`
		Specifies the IPv4 or IPv6 address family type and enters address family configuration mode.

Step 8	**allowas-in** *number* **Example:** `switch(config-router-vrf-neighbor-af)# allowas-in 3`	(Optional) Allows an AS path with the PE ASN for a specified number of times. • The range is from 1 to 10 • If all BGP sites are using the same AS number, coof the cfollowing commands: **Note** Configure the BGP as-override command at the PE (hub) or Configure the allowas-in command at the receiving CE router. The *as-number* argument indicates the number of an autonomous system that identifies the router to other BGP routers and tags the routing information passed along. The AS number can be a 16-bit integer or a 32-bit integer in the form of a higher 16-bit decimal number and a lower 16-bit decimal number in xx.xx format.
Step 9	**send-community extended** **Example:** `switch(config-router-neighbor)# send-community extended`	(Optional) Configures BGP to advertise extended community lists.
Step 10	**show running-config bgp**	(Optional) Displays the running configuration for BGP.

	Command or Action	Purpose
	Example: `switch(config-router-vrf-neighbor-af)# show running-config bgp`	
Step 11	**copy running-config startup-config** **Example:** `switch(config-router-vrf)# copy running-config startup-config`	(Optional) Copies the running configuration to the startup configuration.

4.5.3 Configuring MPLS using Hardware Profile Command

Inspur CN12904 and CN12908 switches with CN129-X636C-R, CN129-X636Q-R, or CN129-X6136YC-R line cards supports multiple hardware profiles. You can configure MPLS and/or VXLAN using hardware profile configuration command in a switch. The hardware profile configuration command invokes appropriate configuration files that are available on the switch. VXLAN is enabled by default

Before you begin

SUMMARY STEPS

1. **configure terminal**
2. **feature bgp**
3. **hardware profile** [**vxlan** | **mpls**] **module all**
4. **show hardware profile module** [**all** | *number*]
5. **show module internal sw info** | [**i** | **mpls**]
6. **show running configuration** | [**i** | **mpls**]

DETAILED STEPS

	Command or Action	Purpose

Step 1	**configure terminal**
	Example:
	`switch# configure terminal`
	`switch(config)#`
		Enters global configuration mode.

Step 2	**feature bgp**
	Example:
	`switch(config)# feature bgp`
	`switch(config)#`
		Enables the BGP feature.

Step 3	**hardware profile** [**vxlan** | **mpls**] **module all** **Example:** `switch(config)# hardware profile mpls  module all`	Enables MPLS on all the switch modules. .
Step 4	**show hardware profile module** [**all** | *number*] **Example:**	Displays the hardware profile of all the modules or specific module.

	Command or Action	Purpose
	`switch(config)# show hardware profile module all` `switch(config)#`	
Step 5	**show module internal sw info** | [**i** | **mpls**] **Example:** `switch(config)# show module internal sw info`	Displays the switch software information.
Step 6	**show running configuration** | [**i** | **mpls**] **Example:** `switch(config)# show module internal sw info`	Displays the running configuration.

CHAPTER 5 Configuring MPLS L3VPN Label Allocation

This chapter describes how to configure label allocation for Multiprotocol Label Switching (MPLS) Layer 3 virtual private networks (L3VPNs) on Inspur CN12904 and CN12908 switches.

5.1 Information About MPLS L3VPN Label Allocation

The MPLS provider edge (PE) router stores both local and remote routes and includes a label entry for each route. By default, Inspur INOS-CN uses per-prefix label allocation which means that each prefix is assigned a label. For distributed platforms, the per-prefix labels consume memory. When there are many VPN routing and forwarding instances (VRFs) and routes, the amount of memory that the per-prefix labels consume can become an issue.

You can enable per-VRF label allocation to advertise a single VPN label for local routes throughout the entire VRF. The router uses a new VPN label for the VRF decoding and IP-based lookup to learn where to forward packets for the PE or customer edge (CE) interfaces.

You can enable different label allocation modes for Border Gateway Protocol (BGP) Layer 3 VPN routes to meet different requirements and to achieve trade-offs between scalability and performance. All labels are allocated within the global label space. Inspur INOS-CN supports the following label allocation modes:

- Per-prefix—A label is allocated for each VPN prefix. VPN packets received from remote PEs can be directly forwarded to the connected CE that advertised the prefix, based on the label forwarding table. However, this mode also uses many labels. This mode is the only mode available when VPN packets sent from PE to CE are label switched. This is the default label allocation mode.
- Per-VRF—A single label is assigned to all local VPN routes in a VRF. This mode requires an IPv4 or IPv6 lookup in the VRF forwarding table once the VPN label is removed at the egress PE. This mode is the most efficient in terms of label space as well as BGP advertisements, and the lookup does not result in any performance degradation. Inspur INOS-CN uses the same per-VRF label for both IPv4 and IPv6 prefixes.
- Aggregate Labels—BGP can allocate and advertise a local label for an aggregate prefix. Forwarding requires an IPv4 or IPv6 lookup that is similar to the per-VRF scenario. A single per-VRF label is allocated and used for all prefixes that need a lookup.
- VRF connected routes—When directly connected routes are redistributed and exported, an aggregate label is allocated for each route. The packets that come in from the core are decapsulated and a lookup is done in the VRF IPv4 or IPv6 table to determine whether the packet is for the local router or for another router or host that is directly connected. A single per-VRF label is allocated for all such routes.
- Label hold down—When a local label is no longer associated with a prefix, to allow time for updates to be sent to other PEs, the local label is not released immediately. A ten-minute hold down timer is started per label. Within this hold down period, the label can be reclaimed for the prefix. When the timer expires, BGP releases the label.

5.1.1 IPv6 Label Allocation

IPv6 prefixes are advertised with the allocated label to iBGP peers that have the labeled-unicast address-family enabled. The received eBGP next hop is not propagated to such peers; instead, the local IPv4 session address is sent as an IPv4-mapped IPv6 next hop. The remote peer resolves this next hop through one or more IPv4 MPLS LSPs in the core network.

You can use a route reflector to advertise the labeled 6PE prefixes between PEs. You must enable the labeled-unicast address-family between the route reflector and all such peers. The route reflector does not need to be in the forwarding path and propagates the received next hop as is to iBGP peers and route reflector clients.

5.1.2 Per-VRF Label Allocation Mode

The following conditions apply when you configure per-VRF label allocation:

- The VRF uses one label for all local routes.
- When you enable per-VRF label allocation, any existing per-VRF aggregate label is used. If no per-VRF aggregate label is present, the software creates a new per-VRF label.

The CE does not lose data when you disable per-VRF label allocation because the configuration reverts to the default per-prefix labeling configuration.

- A per-VRF label forwarding entry is deleted only if the VRF, BGP, or address family configuration is removed.

5.2 Licensing Requirements for MPLS L3VPN Label Allocation

Table 3：Licensing Requirements for MPLS L3VPN Label Allocation

Product	License Requirement
Inspur INOS-CN	L3VPN label allocation requires an MPLS license. • CN129-AD-M4 for CN12904 • CN129-AD-M8 for CN12908

5.3 Prerequisites for MPLS L3VPN Label Allocation

L3VPN label allocation has the following prerequisites:

- Ensure that you have configured MPLS, and LDP or RSVP TE in your network. All routers in the core, including the PE routers, must be able to support MPLS forwarding.
- Ensure that you have installed the correct license for MPLS and any other features you will be using with MPLS.
- Ensure that you disable the external/internal Border Gateway Protocol (BGP) multipath feature if it is enabled before you configure per-VRF label allocation mode.
- Before configuring a 6VPE per VRF label, ensure that the IPv6 address family is configured on that VRF.

5.4 Guidelines and Limitations for MPLS L3VPN Label Allocation

L3VPN label allocation has the following configuration guidelines and limitations:

- Enabling per-VRF label allocation causes BGP reconvergence, which can result in data loss for traffic coming from the MPLS VPN core.
- Aggregate labels and per-VRF labels are global across all virtual device contexts (VDCs) and are in a separate, dedicated label range.
- Aggregate prefixes for per-prefix label allocation share the same label in a given VRF.

5.5 Default Settings for MPLS L3VPN Label Allocation

Table 4：Default L3VPN Label Allocation Parameters

Parameters	Default
L3VPN feature	Disabled

Parameters	**Default**
Label allocation mode	Per prefix

5.6 Configuring MPLS L3VPN Label Allocation

5.6.1 Configuring Per-VRF L3VPN Label Allocation Mode

You can configure per-VRF L3VPN label allocation mode for Layer 3 VPNs.

SUMMARY STEPS

1. **configure terminal**
2. **feature bgp**
3. **feature-set mpls**
4. **feature-set mpls l3vpn**
5. **router bgp** *as - number*
6. **vrf** *vrf-name*
7. **address-family { ipv4 | ipv6 } unicast | multicast }**
8. **label-allocation-mode per-vrf**
9. **show bgp l3vpn detail vrf** *vrf-name*
10. **copy running-config startup-config**

DETAILED STEPS

	Command or Action	**Purpose**

Step 1	**configure terminal**
	Example:
	`switch# configure terminal`
	`switch(config)#`
		Enters global configuration mode.

Step 2	**feature bgp**
	Example:
	`switch(config)# feature bgp`
	`switch(config)#`
		Enables the BGP feature.

Step 3	**feature-set mpls**
	Example:
	`switch(config)# feature-set mpls`
	`switch(config)#`
		Enables the MPLS feature-set.

Step 4	**feature-set mpls l3vpn**
	Example:
	`switch(config)# feature-set mpls l3vpn`
	`switch(config)#`
		Enables the MPLS Layer 3 VPN feature.

Step 5	**router bgp** *as - number* **Example:**	Configures a BGP routing process and enters router configuration mode. The as-number argument indicates the number of an autonomous system that identifies

	Command or Action	Purpose
	`switch(config)# router bgp 1.1`	the router to other BGP routers and tags the routing information. The AS number can be a 16-bit integer or a 32-bit integer in the form of a higher 16-bit decimal number and a lower 16-bit decimal number in xx.xx format.
Step 6	**vrf** *vrf-name* **Example:** `switch(config-router)# vrf vpn1`	Enters router VRF configuration mode. The vrf-name can be any case-sensitive, alphanumeric string up to 32 characters.
Step 7	**address-family { ipv4 | ipv6 } unicast | multicast }** **Example:** `switch(config-router-vrf)# address-family ipv6 unicast`	Specifies the IP address family type and enters address family configuration mode.

Step 8	**label-allocation-mode per-vrf**
	Example:
	`switch(config-router-vrf-af)#`
	`label-allocation-mode per-vrf`
		Allocates labels on a per-VRF basis.

Step 9	**show bgp l3vpn detail vrf** *vrf-name*
	Example:
	`switch(config-router-vrf-af)# show bgp l3vpn`
	`detail vrf vpn1`
		(Optional) Displays information about Layer 3 VPN
		configuration on BGP for this VRF. The vrf-name can be
		any case-sensitive, alphanumeric string up to 32 characters.

Step 10	**copy running-config startup-config** **Example:** `switch(config-router-vrf)# copy running-config startup-config`	(Optional) Copies the running configuration to the startup configuration.

5.6.2 Allocating Labels for IPv6 Prefixes in the Default VRF

If you are running IPv6 over an IPv4 MPLS core network (6PE), you can allocate labels for the IPv6 prefixes in the default VRF.

SUMMARY STEPS

1. **configure terminal**
2. **feature bgp**
3. **feature-set mpls**
4. **feature-set mpls l3vpn**
5. **router bgp** *as - number*
6. **address-family { ipv4 | ipv6 } unicast | multicast }**
7. **allocate-label { all | route-map** *route-map* **}**
8. **show running-config bgp**
9. **copy running-config startup-config**

DETAILED STEPS

	Command or Action	Purpose

Step 1	**configure terminal**
	Example:
	`switch# configure terminal`
	`switch(config)#`
		Enters global configuration mode.

Step 2	**feature bgp**
	Example:
	`switch(config)# feature bgp`
	`switch(config)#`
		Enables the BGP feature.

Step 3	**feature-set mpls**
	Example:
	`switch(config)# feature-set mpls`
	`switch(config)#`
		Enables the MPLS feature-set.

Step 4	**feature-set mpls l3vpn**
	Example:
	`switch(config)# feature-set mpls l3vpn`
	`switch(config)#`
		Enables the MPLS Layer 3 VPN feature.

Step 5	**router bgp** *as - number* **Example:** `switch(config)# router bgp 1.1`	Configures a BGP routing process and enters router configuration mode. The as-number argument indicates the number of an autonomous system that identifies the router to other BGP routers and tags the routing information. The AS number can be a 16-bit integer or a 32-bit integer in the form of a higher 16-bit decimal number and a lower 16-bit decimal number in xx.xx format.
Step 6	**address-family { ipv4 | ipv6 } unicast | multicast }** **Example:**	Specifies the IP address family type and enters address family configuration mode.

	Command or Action	Purpose
	`switch(config-router-vrf)# address-family ipv6 unicast`	
Step 7	**allocate-label { all | route-map** *route-map* **}** **Example:** `switch(config-router-af)# allocate-label all`	Allocates labels for IPv6 prefixes in the default VRF. • The **all** keyword allocates labels for all IPv6 prefixes. • The **route-map** keyword allocates labels for IPv6 prefixes matched in the specified route map. The route-map can be any case-sensitive alphanumeric string up to 63 characters.
Step 8	**show running-config bgp** **Example:** `switch(config-router-af)# show running-config bgp`	(Optional) Displays information about the BGP configuration.
Step 9	**copy running-config startup-config** **Example:** `switch(config-router-vrf)# copy running-config startup-config`	(Optional) Copies the running configuration to the startup configuration.

5.6.3 Enabling Sending MPLS Labels in IPv6 over an IPv4 MPLS Core Network (6PE) for iBGP Neighbors

You can enable sending MPLS labels to iBGP neighbors.

SUMMARY STEPS

1. **configure terminal**
2. **feature bgp**
3. **feature-set mpls**
4. **feature-set mpls l3vpn**
5. **router bgp** *as - number*
6. **neighbor** *ip-address*
7. **address-family ipv6 labeled-unicast**
8. **show running-config bgp**
9. **copy running-config startup-config**

DETAILED STEPS

	Command or Action	Purpose
Step 1	**configure terminal** **Example:**	Enters global configuration mode.

	Command or Action	Purpose
	`switch# configure terminal` `switch(config)#`	

Step 2	**feature bgp**
	Example:
	`switch(config)# feature bgp`
	`switch(config)#`
		Enables the BGP feature.

Step 3	**feature-set mpls**
	Example:
	`switch(config)# feature-set mpls`
	`switch(config)#`
		Enables the MPLS feature-set.

Step 4	**feature-set mpls l3vpn**
	Example:
	`switch(config)# feature-set mpls l3vpn`
	`switch(config)#`
		Enables the MPLS Layer 3 VPN feature.

Step 5	**router bgp** *as - number* **Example:** `switch(config)# router bgp 1.1`	Configures a BGP routing process and enters router configuration mode. The as-number argument indicates the number of an autonomous system that identifies the router to other BGP routers and tags the routing information. The AS number can be a 16-bit integer or a 32-bit integer in the form of a higher 16-bit decimal number and a lower 16-bit decimal number in xx.xx format.

Step 6	**neighbor** *ip-address*
	Example:
	`switch(config-router)# neighbor 209.165.201.1`
	`switch(config-router-neighbor)#`
		Adds an entry to the BGP or multiprotocol BGP neighbor
		table. The *ip-address* argument specifies the IP address of
		the neighbor in dotted decimal notation.

Step 7	**address-family ipv6 labeled-unicast**
	Example:
	`switch(config-router-neighbor)# address-family ipv6`
	` labeled-unicast`
	`switch(config-router-neighbor-af)#`
		Specifies IPv6 labeled unicast address prefixes. This command is accepted only for iBGP neighbors.

Step 8	**show running-config bgp** **Example:**	(Optional) Displays information about the BGP configuration.

	Command or Action	Purpose
	`switch(config-router-af)# show running-config bgp`	
Step 9	**copy running-config startup-config** **Example:** `switch(config-router-vrf)# copy running-config startup-config`	(Optional) Copies the running configuration to the startup configuration.

5.7 Verifying MPLS L3VPN Label Allocation Configuration

To display the L3VPN label allocation configuration, perform one of the following tasks:

Table 5： Verifying MPLS L3VPN Label Allocation Configuration

Command	Purpose
show bgp l3vpn [**detail**] [**vrf** *v rf-name*]	Displays Layer 3 VPN information for BGP in a VRF.
show bgp vpnv4 unicast labels [**vrf** *v rf-name*]	Displays label information for BGP.
show ip route [**vrf** *v rf-name*]	Displays label information for routes.

5.8 Configuration Examples for MPLS L3VPN Label Allocation

The following example shows how to configure per-VRF label allocation for an IPv4 MPLS network.

```
    PE1
    -----
    vrf
    context
    vpn1 rd
    100:1
    address-family ipv4
    unicast route-target
    export 200:1 router
    bgp 100
    neighbor 10.1.1.2 remote-
    as 100 address-family
    vpnv4 unicast send-
    community extended
    update-source loopback10
    vrf vpn1
    address-family ipv4
    unicast label-allocation-
    mode per-vrf neighbor
    36.0.0.2 remote-as 300
address-family ipv4 unicast
```

CHAPTER 6 Configuring MPLS Layer 3 VPN Load Balancing

This chapter describes how to configure load balancing for Multiprotocol Label Switching (MPLS) Layer 3 virtual private networks (VPNs) on Inspur CN12904 and CN12908 switches.

6.1 Information About MPLS Layer 3 VPN Load Balancing

Load balancing distributes traffic so that no individual router is overburdened. In an MPLS Layer 3 network, you can achieve load balancing by using the Border Gateway Protocol (BGP). When multiple iBGP paths are installed in a routing table, a route reflector advertises only one path (next hop). If a router is behind a route reflector, all routes that are connected to multihomed sites are not advertised unless a different route distinguisher is configured for each virtual routing and forwarding instance (VRF). (A route reflector passes learned routes to neighbors so that all iBGP peers do not need to be fully meshed.)

6.1.1 iBGP Load Balancing

When a BGP-speaking router configured with no local policy receives multiple network layer reachability information (NLRI) from the internal BGP (iBGP) for the same destination, the router chooses one iBGP path as the best path and installs the best path in its IP routing table. iBGP load balancing enables the BGP-speaking router to select multiple iBGP paths as the best paths to a destination and to install multiple best paths in its IP routing table.

6.1.2 eBGP Load Balancing

When a router learns two identical eBGP paths for a prefix from a neighboring autonomous system, it chooses the path with the lower route ID as the best path. The router installs this best path in the IP routing table. You can enable eBGP load balancing to install multiple paths in the IP routing table when the eBGP paths are learned from a neighboring autonomous system instead of picking one best path.

During packet switching, depending on the switching mode, the router performs either per-packet or per-destination load balancing among the multiple paths.

6.1.3 Layer 3 VPN Load Balancing

Layer 3 VPN load balancing for both eBGP and iBGP allows you to configure multihomed autonomous systems and provider edge (PE) routers to distribute traffic across both external BGP (eBGP) and iBGP multipaths.

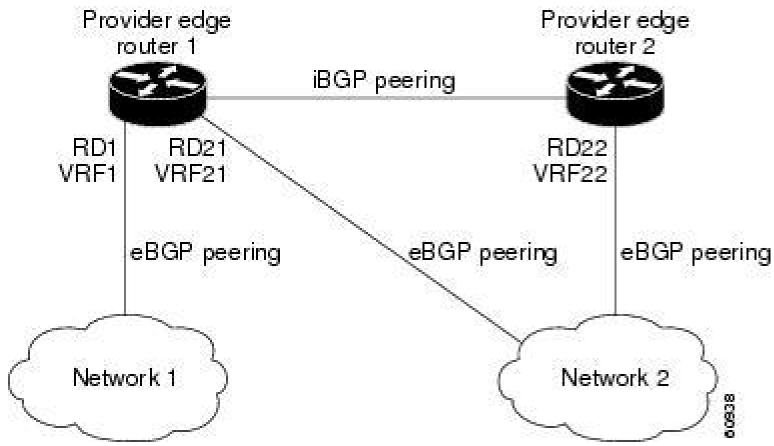
Layer 3 VPN load balancing supports IPv4 and IPv6 for the PE routers and VPNs.

BGP installs up to the maximum number of multipaths allowed. BGP uses the best path algorithm to select one path as the best path, inserts the best path into the routing information base (RIB) and advertises the best path to BGP peers. The router can insert other paths into the RIB but selects only one path as the best path.

Layer 3 VPNs load balance on a per-packet or per-source or destination pair basis. To enable load balancing, configure the router with Layer 3 VPNs that contain VPN routing and forwarding instances (VRFs) that import both eBGP and iBGP paths. You can configure the number of paths separately for each VRF.

The following figure shows an MPLS provider network that uses BGP. In the figure, two remote networks are connected to PE1 and PE2, which are both configured for VPN unicast iBGP peering. Network 2 is a multihomed network that is connected to PE1 and PE2. Network 2 also has extranet VPN services configured with Network 1. Both Network 1 and Network 2 are configured for eBGP peering with the PE routers.

Figure 4 : Provider MPLS Network Using BGP



You can configure PE1 so that it can select both iBGP and eBGP paths as multipaths and import these paths into the VPN routing and forwarding instance (VRF) of Network 1 to perform load balancing.

Traffic is distributed as follows:

- IP traffic that is sent from Network 2 to PE1 and PE2 is sent across the eBGP paths as IP traffic.
- IP traffic that is sent from PE1 to PE2 is sent across the iBGP path as MPLS traffic.
- Traffic that is sent across an eBGP path is sent as IP traffic.

Any prefix that is advertised from Network 2 will be received by PE1 through route distinguisher (RD) 21 and RD22.

- The advertisement through RD21 is carried in IP packets.
- The advertisement through RD22 is carried in MPLS packets.

The router can select both paths as multipaths for VRF1 and insert these paths into the VRF1 RIB.

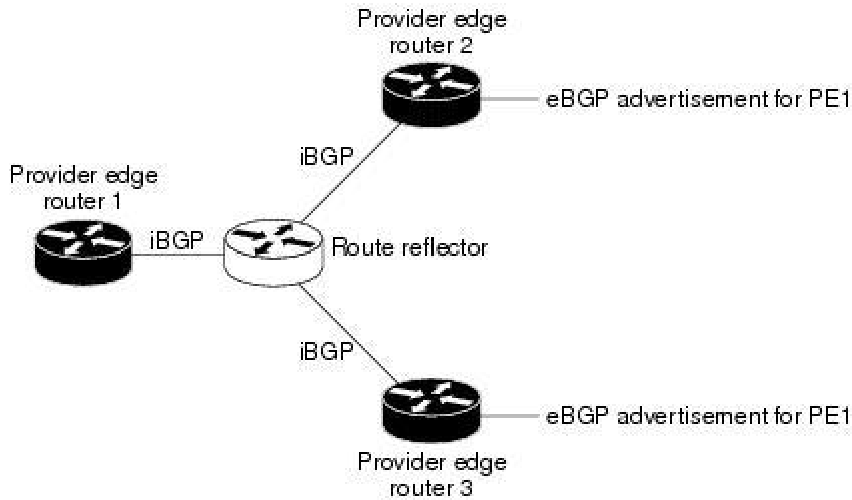
Layer 3 VPN Load Balancing with Route Reflectors

Route reflectors reduce the number of sessions on PE routers and increase the scalability of Layer 3 VPN networks. Route reflectors hold on to all received VPN routes to peer with PE routers. Different PEs can require different route target-tagged VPNv4 and VPNv6 routes. The route reflector may also need to send a refresh for a specific route target to a PE when the VRF configuration has changed. Storing all routes increases the scalability requirements on a route reflector. You can configure a route reflector to only hold routes that have a defined set of route target communities.

You can configure route reflectors to service a different set of VPNs and configure a PE to peer with all route reflectors that service the VRFs configured on the PE. When you configure a new VRF with a route target that the PE does not already hold routes for, the PE issues route refreshes to the route reflectors and retrieves the relevant VPN routes.

The following figure shows a topology that contains three PE routers and a route reflector, all configured for iBGP peering. PE2 and PE3 each advertise an equal preference eBGP path to PE1. By default, the route reflector chooses only one path and advertises PE1.

Figure 5： Topology with a Route Reflector



For all equal preference paths to PE1 to be advertised through the route reflector, you must configure each VRF with a different RD. The prefixes received by the route reflector are recognized differently and advertised to PE1.

Layer 2 Load Balancing Coexistence

The load balance method that is required in the Layer 2 VPN is different from the method that is used for Layer 3 VPN. Layer 3 VPN and Layer 2 VPN forwarding is performed independently using two different

types of adjacencies. The forwarding is not impacted by using a different method of load balancing for the Layer 2 VPN.

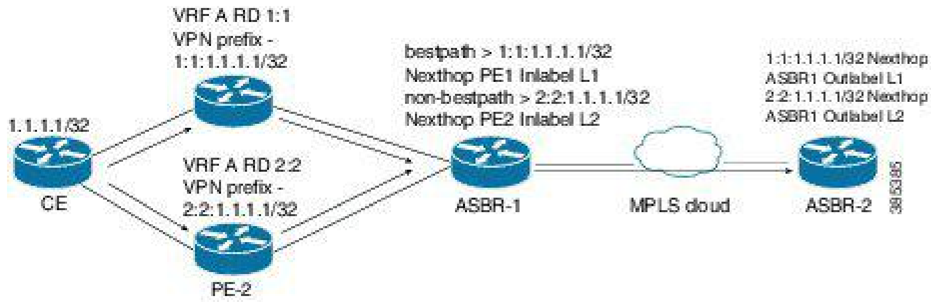
6.1.4 BGP VPNv4 Multipath

BGP VPNv4 Multipath feature helps to achieve Equal Cost Multi-Path (ECMP) for traffic flowing from an Autonomous System Border Router (ASBR) towards the Provider Edge (PE) device in an Multi-Protocol Label Switching (MPLS) cloud network by using a lower number of prefixes and MPLS labels. This feature configures the maximum number of multipaths for both eBGP and iBGP paths. This feature can be configured on PE devices and Route Reflectors in an MPLS topology.

Consider a scenario in which a dual homed Customer Edge (CE) device is connected to 2 PE devices and you have to utilize both the PE devices for traffic flow from ASBR-2 to the CE device.

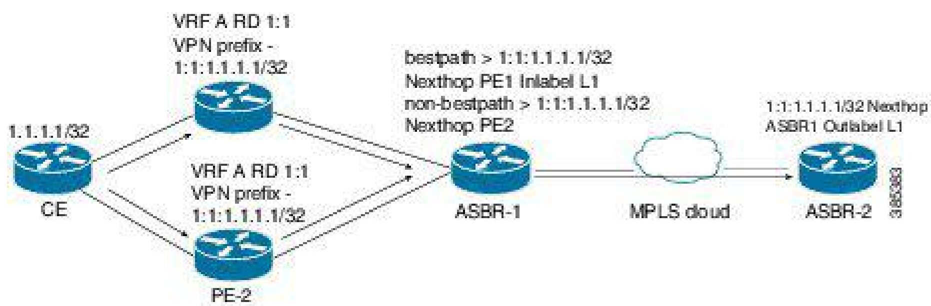
Currently, as shown in following figure, Virtual Routing and Forwarding (VRF) on each PE is configured using separate Route Distinguishers (RD). The CE device generates a BGP IPv4 prefix. The PE devices are configured with 2 separate RDs and generate two different VPN-IPv4 prefixes for the BGP IPv4 prefix sent by the CE device. ASBR-1 receives both the VPN-IPv4 prefixes and adds them to the routing table. ASBR-1 allocates Inter-AS option-B labels, Inlabel L1 and Inlabel L2, to both the VPN routes and then advertises both VPN routes to ASBR-2. To use both PE devices to maintain traffic flow, ASBR-1 has to utilize two Inter-AS option-B labels and two prefixes which limits the scale that can be supported.

Figure 6： Virtual Routing and Forwarding (VRF) on each PE configured using separate Route Distinguishers



Using the BGP VPN Multipath feature, as shown in Figure 22-4, you can enable the VRF on both PE devices to use the same RD. In such a scenario, ASBR-1 receives the same prefix from both the PE devices. ASBR-1 allocates only one Inter-AS option-B label, Inlabel L1, to the received prefix and advertises the VPN route to ASBR-2. In this case, the scale is enhanced as traffic flow using both PE devices is established with only one prefix and label on ASBR-1.

Figure 7： Enabling the VRF on both PE devices to use the same RD



6.1.5 BGP Cost Community

The BGP cost community is a nontransitive extended community attribute that is passed to iBGP and confederation peers but not to eBGP peers. (A confederation is a group of iBGP peers that use the same autonomous system number to communicate to external networks.) The BGP cost community attributes includes a cost community ID and a cost value. You can customize the BGP best path selection process for a local autonomous system or confederation by configuring the BGP cost community attribute. You configure the cost community attribute in a route map with a community ID and cost value. BGP prefers the path with the lowest community ID, or for identical community IDs, BGP prefers the path with the lowest cost value in the BGP cost community attribute.

BGP uses the best path selection process to determine which path is the best where multiple paths to the same destination are available. You can assign a preference to a specific path when multiple equal cost paths are available.

Since the administrative distance of iBGP is worse than the distance of most Interior Gateway Protocols (IGPs), the unicast Routing Information Base (RIB) may apply the same BGP cost community compare algorithm before using the normal distance or metric comparisons of the protocol or route. VPN routes that are learned through iBGP can be preferred over locally learned IGP routes.

The cost extended community attribute is propagated to iBGP peers when an extended community exchange is enabled.

How the BGP Cost Community Influences the Best Path Selection Process

The cost community attribute influences the BGP best path selection process at the point of insertion (POI). The POI follows the IGP metric comparison. When BGP receives multiple paths to the same destination, it uses the best path selection process to determine which path is the best path. BGP automatically makes the decision and installs the best path into the routing table. The POI allows you to assign a preference to a specific path when multiple equal cost paths are available. If the POI is not valid for local best path selection, the cost community attribute is silently ignored.

You can configure multiple paths with the cost community attribute for the same POI. The path with the lowest cost community ID is considered first. All of the cost community paths for a specific POI are considered, starting with the one with the lowest cost community ID. Paths that do not contain the cost community (for the POI and community ID being evaluated) are assigned with the default community cost value.

Applying the cost community attribute at the POI allows you to assign a value to a path originated or learned by a peer in any part of the local autonomous system or confederation. The router can use the cost community as a tie breaker during the best path selection process. You can configure multiple instances of the cost community for separate equal cost paths within the same autonomous system or confederation. For example, you can apply a lower cost community value to a specific exit path in a network with multiple equal cost exits points, and the BGP best path selection process prefers that specific exit path.

Cost Community and EIGRP PE-CE with Back-Door Links

BGP prefers back-door links in an Enhanced Interior Gateway Protocol (EIGRP) Layer 3 VPN topology if the back-door link is learned first. A back-door link, or a route, is a connection that is configured outside of the Layer 3 VPN between a remote and main site.

The pre-best path point of insertion (POI) in the BGP cost community supports mixed EIGRP Layer 3 VPN network topologies that contain VPN and back-door links. This POI is applied automatically to EIGRP routes that are redistributed into BGP. The pre-best path POI carries the EIGRP route type and metric. This POI influences the best-path calculation process by influencing BGP to consider this POI before any other comparison step.

6.2 Licensing Requirements for MPLS Layer 3 VPN Load Balancing

Table 6 :

Product	License Requirement
Inspur INOS-CN	MPLS Layer 3 VPN load balancing requires an Advantage license package: • CN129-AD-M4 for CN12904 • CN129-AD-M8 for CN12908

6.3 Prerequisites for MPLS Layer 3 VPN Load Balancing

MPLS Layer 3 VPN load balancing has the following prerequisites:

- You must enable the MPLS and L3VPN features.
- You must install the correct license for MPLS.

6.4 Guidelines and Limitations for MPLS Layer 3 VPN Load Balancing

MPLS Layer 3 VPN load balancing has the following configuration guidelines and limitations:

- You can configure MPLS Layer 3 VPN load balancing for Inspur CN12904 and CN12908 platform switches with the CN129-X636C-R, CN129-X636Q-R, or CN129-X6136YC-R line cards.
- If you place a router behind a route reflector and it is connected to multihomed sites, the router will not be advertised unless separate VRFs with different RDs are configured for each VRF.
- Each IP routing table entry for a BGP prefix that has multiple iBGP paths uses additional memory. We recommend that you do not use this feature on a router with a low amount of available memory or when it is carrying a full Internet routing table.
- You should not ignore the BGP cost community when a back-door link is present and EIGRP is the PE-CE routing protocol.
- A maximum of 16K VPN prefixes is supported on Inspur CN12904 and CN12908 platform switches with CN129-X636C-R, CN129-X636Q-R, or CN129-X6136YC-R line cards, and a maximum of 470K VPN prefixes is supported on Inspur CN12904 and CN12908 platform switches with CN129-X9636C-R line cards.
- 4K VRFs are supported.

6.5 Default Settings for MPLS Layer 3 VPN Load Balancing

The following table lists the default settings for MPLS Layer 3 VPN load balancing parameters.

Table 7： Default MPLS Layer 3 VPN Load Balancing Parameters

Parameters	Default
Layer 3 VPN feature	Disabled
BGP cost community ID	128
BGP cost community cost	2147483647
maximum multipaths	1
BGP VPNv4 Multipath	Disabled

6.6 Configuring MPLS Layer 3 VPN Load Balancing

6.6.1 Configuring BGP Load Balancing for eBGP and iBGP

You can configure a Layer 3 VPN load balancing for an eBGP or iBGP network.

SUMMARY STEPS

1. **configure terminal**
2. **feature-set mpls**
3. **feature mpls l3vpn**
4. **feature bgp**
5. **router bgp** *as - number*
6. **bestpath cost-community ignore remote-as** *as-number*

7. **address-family { ipv4 | ipv6 } unicast**
8. **maximum-paths** [**bgp**] *number-of-paths*
9. **show running-config bgp**
10. **copy running-config startup-config**

DETAILED STEPS

	Command or Action	Purpose

Step 1	**configure terminal**
	Example:
	`switch# configure terminal`
	`switch(config)#`
		Enters global configuration mode.

Step 2	**feature-set mpls** **Example:** `switch(config)# feature-set mpls`	Enables the MPLS feature-set.
Step 3	**feature mpls l3vpn** **Example:** `switch(config)# feature mpls l3vpn`	Enables the MPLS Layer 3 VPN feature.

Step 4	**feature bgp**
	Example:
	`switch(config)# feature bgp`
	`switch(config)#`
		Enables the BGP feature.

Step 5	**router bgp** *as - number*
	Example:
	`switch(config)# router bgp 1.1`
	`switch(config-router)#`
		Configures a BGP routing process and enters router configuration mode.
		The *as-number* argument indicates the number of an
		autonomous system that identifies the router to other BGP
		routers and tags the routing information passed along. The
		AS number can be a 16-bit integer or a 32-bit integer in
		the form of a higher 16-bit decimal number and a lower
		16-bit decimal number in xx.xx format.

Step 6	**bestpath cost-community ignore remote-as** *as-number*
	Example:
	`switch(config-router)# bestpath cost-community`
	`ignore#`
		(Optional) Ignores the cost community for BGP bestpath
		calculations.

Step 7	**address-family { ipv4 | ipv6 } unicast**	Enters address family configuration mode for configuring

	Command or Action	Purpose

	Example:
	`switch(config-router)# address-family ipv4 unicast`
	`switch(config-router-af)#`
		IP routing sessions.

Step 8	**maximum-paths** [**bgp**] *number-of-paths* **Example:** `switch(config-router-af)# maximum-paths 4`	Configures the maximum number of multipaths allowed. Use the ibgp keyword to configure **iBGP** load balancing. The range is from 1 to 16.
Step 9	**show running-config bgp** **Example:** `switch(config-router-vrf-neighbor-af)# show running-config bgp`	(Optional) Displays the running configuration for BGP.
Step 10	**copy running-config startup-config** **Example:** `switch(config-router-vrf)# copy running-config startup-config`	(Optional) Copies the running configuration to the startup configuration.

6.6.2 Configuring BGPv4 Multipath

SUMMARY STEPS

1. **configure terminal**
2. **feature bgp**
3. **router bgp** *as - number*
4. **address-family vpnv4 unicast**
5. **maximum-paths eibgp** *parallel-paths*

DETAILED STEPS

	Command or Action	Purpose

Step 1	**configure terminal**
	Example:
	`switch# configure terminal`
	`switch(config)#`
		Enters global configuration mode.

Step 2	**feature bgp** **Example:** `switch(config)# feature bgp`	Enables the BGP feature.
Step 3	**router bgp** *as - number*	Assigns an autonomous system (AS) number to a router

	Command or Action	Purpose

	Example:
	`switch(config)# router bgp 2`
	`switch(config-router)#`
		and enter the router BGP configuration mode.

Step 4	**address-family vpnv4 unicast**
	Example:
	`switch(config-router)# address-family vpnv4 unicast`
	`switch(config-router-af)#`
		Enters address family configuration mode for configuring
		routing sessions, such as BGP, that use standard VPNv4
		address prefixes.

Step 5	**maximum-paths eibgp** *parallel-paths* **Example:** `switch(config-router-af)# maximum-paths eibgp 3`	Specifies the maximum number of BGP VPNv4 multipaths for both eBGP and iBGP paths. The range is from 1 to 32.

6.7 Configuration Examples for MPLS Layer 3 VPN Load Balancing

6.7.1 Example: MPLS Layer 3 VPN Load Balancing

The following example shows how to configure iBGP load balancing:

```
 configure  terminal
feature-set    mpls
feature  mpls  l3vpn
feature bgp
router bgp 1.1
bestpath       cost-community
ignore   address-family   ipv6
unicast maximum-paths ibgp 4
```

6.7.2 Example: BGP VPNv4 Multipath

The following example shows how to configure a maximum of 3 BGP VPNv4 multipaths:

```
 configure  terminal
router bgp 100
address-family  vpnv4  unicast
maximum-paths eibgp 3
```

6.7.3 Example: MPLS Layer 3 VPN Cost Community

The following example shows how to configure the BGP cost community:

```
 configure terminal
feature-set mpls
feature mpls l3vpn
feature bgp
route-map CostMap permit
set extcommunity cost 1
100 router bgp 1.1
router-id 192.0.2.255
neighbor 192.0.2.1 remote-as
1.1 address-family vpnv4
unicast send-community extended
route-map CostMap in
```

CHAPTER 7 Configuring Segment Routing

This chapter contains information on how to configure segment routing.

7.1 About Segment Routing

Segment routing is a technique by which the path followed by a packet is encoded in the packet itself, similar to source routing. A node steers a packet through a controlled set of instructions, called segments, by prepending the packet with a segment routing header. Each segment is identified by a segment ID (SID) consisting of a flat unsigned 32-bit integer.

Border Gateway Protocol (BGP) segments, a subclass of segments, identify a BGP forwarding instruction. There are two groups of BGP segments: prefix segments and adjacency segments. Prefix segments steer packets along the shortest path to the destination, using all available equal-cost multi-path (ECMP) paths.

Adjacency segments steer packets onto a specific link to a neighbor.

The segment routing architecture is applied directly to the MPLS data plane.

7.1.1 BGP Prefix SID

In order to support segment routing, BGP requires the ability to advertise a segment identifier (SID) for a BGP prefix. A BGP prefix SID is always global within the segment routing BGP domain and identifies an instruction to forward the packet over the ECMP-aware best path computed by BGP to the related prefix. The BGP prefix SID identifies the BGP prefix segment.

7.1.2 Segment Routing Global Block

The segment routing global block (SRGB) is the range of local labels reserved for MPLS segment routing. The default label range is from 16000 to 23999.

SRGB is the local property of a segment routing node. Each node can be configured with a different SRGB value, and hence the absolute SID value associated to a BGP prefix segment can change from node to node.

The SRGB must be a proper subset of the dynamic label range and must not overlap the optional MPLS static label range. If dynamic labels in the configured or defaulted SRGB range already have been allocated, the configuration is accepted, and the existing dynamic labels that fall in the SRGB range will remain allocated to the original client. If the BGP router attempts to allocate one of these labels, the SRGB mapping fails, and the BGP router reverts to dynamic label allocation. A change to the SRGB range results in the clients deallocating their labels independent of whether the new range can be allocated.

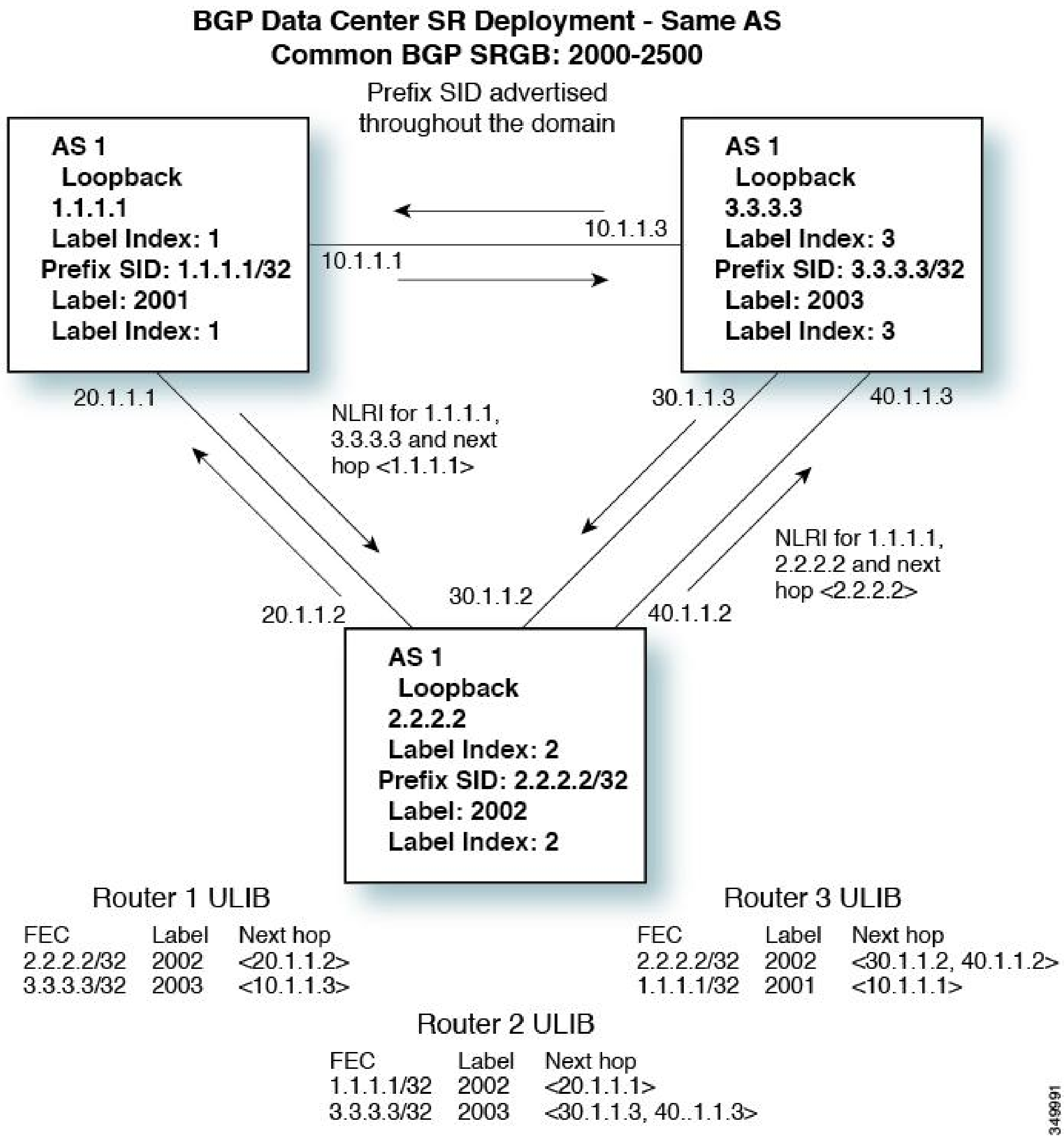
7.1.3 High Availability for Segment Routing

In-service software upgrades (ISSUs) are minimally supported with BGP graceful restart. All states (including the segment routing state) must be relearned from the BGP router's peers. During the graceful restart period, the previously learned route and label state are retained.

7.1.4 BGP Prefix SID Deployment Example

In the simple example below, all three routers are running iBGP and advertising Network Layer Reachability Information (NRLI) to one another. The routers are also advertising their loopback interface as the next hop, which provides the ECMP between routers 2.2.2.2 and 3.3.3.3.

Figure 8 : BGP Prefix SID Simple Example



7.2 Licensing Requirements for Segment Routing

The following table shows the licensing requirements for this feature:

Product	**License Requirement**
Inspur INOS-CN	Layer 3 EVPN over segment routing MPLS requires an Advantage license package: • CN129-AD-M4 for CN12904 • CN129-AD-M8 for CN12908 Any feature not included in a license package is bundled

Product	License Requirement
	with the INOS-CN image and is provided at no extra charge to you. Border Gateway Protocol (BGP) requires an Essentials license package: • CN129-ES-M4 for CN12904 • CN129-ES-M8 for CN12908

7.3 Guidelines and Limitations for Segment Routing

Segment routing has the following guidelines and limitations:

- Segment Routing Application (SR-APP) module is used to configure the segment routing functionality. Segment Routing Application (SR-APP) is a separate internal process that handles all the CLIs related to segment routing. It is responsible for reserving the SRGB range and for notifying the clients about it. It is also responsible for maintaining the prefix to SID mappings. See Configuring Segment Routing Using Segment Routing Application Module for more information.
- BGP allocates a SRGB label for iBGP route-reflector clients only when next-hop-self is in effect (for example, the prefix is advertised with the next hop being one of the local IP/IPv6 addresses on RR). When you have configured next-hop-self on a RR, the next hop is changed for the routes that are being affected (subject to route-map filtering).
- Static MPLS, MPLS segment routing, and MPLS stripping cannot be enabled at the same time.
- Because static MPLS, MPLS segment routing, and MPLS stripping are mutually exclusive, the only segment routing underlay for multi-hop BGP is single-hop BGP. iBGP multi-hop topologies with eBGP running as an overlay are not supported.
- MPLS pop followed by a forward to a specific interface is not supported. The penultimate hop pop (PHP) is avoided by installing the Explicit NULL label as the out-label in the label FIB (LFIB) even when the control plane installs an IPv4 Implicit NULL label.
- BGP labeled unicast and BGP segment routing are not supported for IPv6 prefixes.
- BGP labeled unicast and BGP segment routing are not supported over tunnel interfaces (including GRE and VXLAN) or with vPC access interfaces.
- MTU path discovery (RFC 2923) is not supported over MPLS label switched paths (LSPs) or segment routed paths.
- For the Inspur CN12900 Series switches, MPLS LSPs and segment routed paths are not supported on subinterfaces (either port channels or normal Layer 3 ports).
- For the Inspur CN12900 Series switches, segment routing is supported only in the default hierarchical routing mode.
- The BGP configuration commands **neighbor-down fib-accelerate** and **suppress-fib-pending** are not supported for MPLS prefixes.
- The uniform model as defined in RFC 2973 and RFC 3270 is not supported. Consequently, the IP DSCP bits are not copied into the imposed MPLS header.
- Reconfiguration of the segment routing global block (SRGB) results in an automatic restart of the BGP process to update the existing URIB and ULIB entries. Traffic loss will occur for a few seconds, so you should not reconfigure the SRGB in production.
- If the segment routing global block (SRGB) is set to a range but the route-map label-index delta value is outside of the configured range, the allocated label is dynamically generated. For example, if the SRGB is set to a range of 16000-23999 but a route-map label-index is set to 9000, the label is dynamically allocated.
- For network scalability, Inspur recommends using a hierarchical routing design with multi-hop BGP for advertising the attached prefixes from a top-of-rack (TOR) or border leaf switch.

- BGP sessions are not supported over MPLS LSPs or segment routed paths.
- The Layer 3 forwarding consistency checker is not supported for MPLS routes.
- You can configure segment routing traffic engineering on Inspur CN12900 Series switches.
- You can configure OSPFv2 as an IGP control plane for Segment Routing on Inspur CN12900 switches.
- Layer3 VPN and Layer3 EVPN Stitching for Segment Routing is supported on Inspur CN12900 Series switches

7.4 Overview of BGP Egress Peer Engineering With Segment Routing

Inspur CN12900 Series switches are often deployed in massive scale data centers (MSDCs). In such environments, there is a requirement to support BGP Egress Peer Engineering (EPE) with Segment Routing (SR).

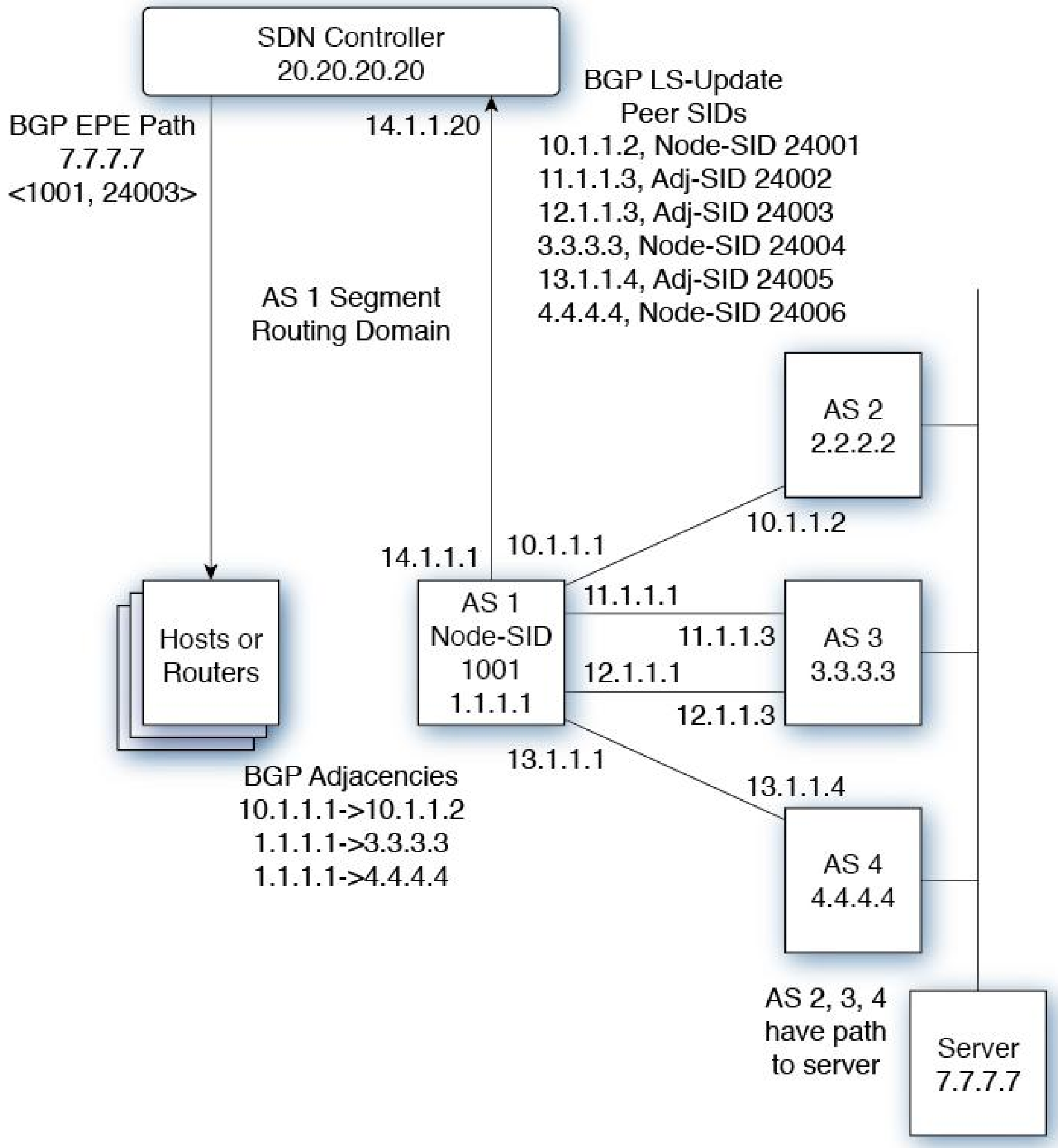
Segment Routing (SR) leverages source routing. A node steers a packet through a controlled set of instructions, known as segments, by prepending the packet with an SR header. A segment can represent any topological or service-based instruction. SR allows steering a flow through any topological path or any service chain while maintaining per-flow state only at the ingress node of the SR domain. For this feature, the Segment Routing architecture is applied directly to the MPLS data plane.

In order to support Segment Routing, BGP requires the ability to advertise a Segment Identifier (SID) for a BGP prefix. A BGP prefix is always global within the SR or BGP domain and it identifies an instruction to forward the packet over the ECMP-aware best-path that is computed by BGP to the related prefix. The BGP prefix is the identifier of the BGP prefix segment.

The SR-based Egress Peer Engineering (EPE) solution allows a centralized (SDN) controller to program any egress peer policy at ingress border routers or at hosts within the domain.

In the following example, all three routers run iBGP and they advertise NRLI to one another. The routers also advertise their loopback as the next-hop and it is recursively resolved. This provides an ECMP between the routers as displayed in the illustration.

Figure 9： Example of Egress Peer Engineering



The SDN controller receives the Segment IDs from the egress router 1.1.1.1 for each of its peers and adjacencies. It can then intelligently advertise the exit points to the other routers and the hosts within the controller's routing domain. As displayed in the illustration, the BGP Network Layer Reachability Information (NLRI) contains both the Node-SID to Router 1.1.1.1 and the Peer-Adjacency-SID 24003 indicating that the traffic to 7.7.7.7 should egress over the link 12.1.1.1->12.1.1.3.

7.4.1 Guidelines and Limitations for BGP Egress Peer Engineering

See the following guidelines and limitations for BGP Egress Peer Engineering:

- BGP Egress Peer Engineering is only supported for IPv4 BGP peers. IPv6 BGP peers are not supported.

- BGP Egress Peer Engineering is only supported in the default VPN Routing and Forwarding (VRF) instance.
- Any number of Egress Peer Engineering (EPE) peers may be added to an EPE peer set. However, the installed resilient per-CE FEC is limited to 32 peers.
- A given BGP neighbor can only be a member of a single peer-set. Peer-sets are configured. Multiple peer-sets are not supported. An optional **peer-set** name may be specified to add neighbor to a peer-set. The corresponding RPC FEC load-balances the traffic across all the peers in the peer-set. The peer-set name is a string that is a maximum length of 63 characters (64 NULL terminated). This length is consistent with the INOS-CN policy name lengths. A peer can only be a member of a single peer-set.
- Adjacencies for a given peer are not separately assignable to different peer-sets.

7.5 Configuring Segment Routing

7.5.1 Configuring Segment Routing Using Segment Routing Application Module

Segment Routing Application (SR-APP) module is used to configure the segment routing functionality. Segment Routing Application (SR-APP) is a separate internal process that handles all the CLIs related to segment routing. It is responsible for reserving the SRGB range and for notifying the clients about it. It is also responsible for maintaining the prefix to SID mappings. The SR-APP support is also available c for the BGP and IS-IS protocols.

Complete the following steps to configure segment routing:

Before you begin

Confirm that the following conditions are met before configuring Segment Routing using the Segment Routing Application (SR-APP) module.

- The **feature-set mpls** and **feature mpls segment-routing** commands should be present for configuring the **segment-routing mpls** command.
- The **feature mpls segment-routing** command starts the SR-APP process.
- If the global block is configured, the specified range is used. Otherwise, the default 16000 – 23999 range is used.
- With the introduction of SR-APP, all configuration is done under **segment-routing mpls** and the prefix SID configuration is handled by SR-APP.
- BGP now uses both **set label-index** <*value*> configuration and the new **connected-prefix-sid-map** CLI. In case of a conflict, the configuration in SR-APP is preferred.

SUMMARY STEPS

1. **configure terminal**
2. **segment-routing mpls**
3. **global-block** <*min*> <*max*>
4. **connected-prefix-sid-map**
5. **address-family ipv4**
6. <*prefix*>/<*masklen*>**[index|absolute]** <*label*>

DETAILED STEPS

	Command or Action	Purpose
Step 1	**configure terminal**	Enters global configuration mode.
Step 2	**segment-routing mpls**	Activates the Segment Routing functionality
Step 3	**global-block** <*min*> <*max*>	Reserves the non-default SRGB range.

	Command or Action	Purpose
	Example: global-block 201000 280000	
Step 4	**connected-prefix-sid-map**	Provides the SID label for the interface IP covered by the prefix-SID map.
Step 5	**address-family ipv4**	Enters global address family configuration mode for the IPv4 address family.
Step 6	*<prefix>*/*<masklen>***[index|absolute]** *<label>* **Example:** 2.1.1.5/32 absolute 201101 2.10.1.5/32 index 10001	The optional keywords **index** or **absolute** indicate whether the label value entered should be interpreted as an index into the SRGB or as an absolute value.

Example

See the following configuration examples of the show commands:

The SRGB allocation needs to be confirmed by an internal process that requires the clients to confirm their cleanup. The amount of time SR-APP waits for the clients to clean their labels, is determined by the cleanup interval. The default value for the cleanup interval is 60 seconds. It can be modified using the **timers srgb cleanup** <*interval*> CLI command.

Retry interval is amount of time for which SR-APP retries the allocation of the SRGB from the internal process if it fails. The default value for the retry interval is 180 and it can be modified using the **timers srgb retry** <*interval*> CLI command. The SR-APP module retries the SRGB allocation 10 times within the configured retry timer value, at equal intervals. See the **show segment-routing** CLI output as displayed in the following example:

```
switch# show segment-routing
Segment-Routing Global info
Service Name: segment-routing
State: Enabled
Process Id: 29123
Configured SRGB: 17000 - 24999
SRGB Allocation status: Alloc-Successful
Current SRGB: 17000 - 24999
Cleanup Interval: 60
Retry Interval: 180
```

The following CLI displays the clients that are registered with SR-APP. It lists the VRFs, for which the clients have registered interest.

```
switch# show segment-routing clients
```

```
               Segment-Routing Client Info
   Client: isis-1      UUID: 0x41000118    PID: 29463    MTS SAP: 412

       PIB index: 1
       TIBs registered:
           VRF: default Table: base
   Client: bgp-1       UUID: 0x11b    PID: 18546    MTS SAP: 62252

       PIB index: 2
       TIBs registered:
           VRF: default Table: base

Total Clients: 2
```

In the **show segment-routing ipv4 connected-prefix-sid-map** CLI command example, SRGB indicates whether the prefix SID is within the configured SRGB. The **Indx** field indicates that the configured label is an index into the global block. The **Abs** field indicates that the configured label is an absolute value.

If the SRGB field displays N, it means that the configured prefix SID is not within the SRGB range and it is not provided to the SR-APP clients. Only the prefix SIDs that fall into the SRGB range are given to the SR-APP clients.

```
switch# show segment-routing ipv4 connected-prefix-sid-map
            Segment-Routing Prefix-SID Mappings
Prefix-SID mappings for VRF default   Table base
Prefix             SID   Type Range SRGB
13.11.2.0/24       713   Indx 1     Y
30.7.7.7/32        730   Indx 1     Y
59.3.24.0/30       759   Indx 1     Y
150.101.1.0/24     801   Indx 1     Y
150.101.1.1/32     802   Indx 1     Y
150.101.2.0/24     803   Indx 1     Y
1.1.1.1/32         16013 Abs  1     Y
```

The following CLI displays the **show running-config segment-routing** output.

```
switch# show running-config segment-routing

!Command: show running-config segment-routing
!Time: Mon Oct 1 10:13:53 2018

    version 9.2(1i)
    segment-routing mpls
      global-block 22000 35000
      connected-prefix-sid-map
      address-family ipv4
          42.11.11.0/24 index 251
          42.11.12.0/24 index 252
          42.11.13.0/24 index 253
          42.11.14.0/24 index 254
          42.11.15.0/24 index 255
          42.11.16.0/24 index 256
          42.11.17.0/24 index 257
          42.11.18.0/24 index 258
          42.11.19.0/24 index 259
          42.11.20.0/24 index 260
          132.10.54.0/24 absolute
```

```
22101 2.2.2.9/32 index 202
2.2.2.10/32 index 203
2.2.2.11/32 index 204
```

7.5.2 Enabling MPLS Segment Routing

You can enable MPLS segment routing as long as mutually-exclusive MPLS features such as static MPLS are not enabled.

Before you begin

You must install and enable the MPLS feature set using the **install feature-set mpls** and **feature-set mpls** commands.

SUMMARY STEPS

1. **configure terminal**
2. [**no**] **feature mpls segment-routing**
3. (Optional) **show running-config | inc 'feature mpls segment-routing'**
4. (Optional) **copy running-config startup-config**

DETAILED STEPS

	Command or Action	Purpose

Step 1	**configure terminal**
	Example:
	`switch# configure terminal`
	`switch(config)#`
		Enters global configuration mode.

Step 2	[**no**] **feature mpls segment-routing** **Example:** `switch(config)# feature mpls segment-routing`	Enables the MPLS segment routing feature. The **no** form of this command disables the MPLS segment routing feature.
Step 3	(Optional) **show running-config | inc 'feature mpls** **segment-routing'** **Example:** `switch(config)# show running-config | inc 'feature mpls segment-routing'`	Displays the status of the MPLS segment routing feature.
Step 4	(Optional) **copy running-config startup-config** **Example:** `switch(config)# copy running-config startup-config`	Copies the running configuration to the startup configuration.

7.5.3 Enabling MPLS on an Interface

You can enable MPLS on an interface for use with segment routing.

Before you begin

You must install and enable the MPLS feature set using the **install feature-set mpls** and **feature-set mpls** commands.

SUMMARY STEPS

1. **configure terminal**
2. **interface** *type slot/port*
3. [**no**] **mpls ip forwarding**
4. (Optional) **copy running-config startup-config**

DETAILED STEPS

	Command or Action	Purpose

Step 1	**configure terminal**
	Example:
	`switch# configure terminal`
	`switch(config)#`
		Enters global configuration mode.

Step 2	**interface** *type slot/port*
	Example:
	`switch(config)# interface ethernet 2/2`
	`switch(config-if)#`
		Enters the interface configuration mode for the specified interface.

Step 3	[**no**] **mpls ip forwarding** **Example:** `switch(config-if)# mpls ip forwarding`	Enables MPLS on the specified interface. The **no** form of this command disables MPLS on the specified interface.
Step 4	(Optional) **copy running-config startup-config** **Example:** `switch(config-if)# copy running-config startup-config`	Copies the running configuration to the startup configuration.

7.5.4 Configuring the Segment Routing Global Block

You can configure the beginning and ending MPLS labels in the segment routing global block (SRGB).

Before you begin

You must install and enable the MPLS feature set using the **install feature-set mpls** and **feature-set mpls** commands.

You must enable the MPLS segment routing feature.

SUMMARY STEPS

1. **configure terminal**
2. [**no**] **segment-routing mpls**
3. [**no**] **global-block** *beginning-label ending-label*

4. (Optional) **show mpls label range**
5. **show segment-routing**
6. (Optional) **copy running-config startup-config**

DETAILED STEPS

	Command or Action	Purpose

Step 1	**configure terminal**
	Example:
	`switch# configure terminal`
	`switch(config)#`
		Enters global configuration mode.

Step 2	[**no**] **segment-routing mpls**
	Example:
	`switch(config)# segment-routing mpls`
	`switch(config-segment-routing-mpls)#`
		Enters the segment routing configuration mode and enables the default SRGB of 16000 to 23999. The **no** form of this command unallocates that block of labels.
		If the configured dynamic range cannot hold the default SRGB, an error message appears, and the default SRGB will not be allocated. If desired, you can configure a different SRGB in the next step.

Step 3	[**no**] **global-block** *beginning-label ending-label*
	Example:
	`switch(config-segment-routing-mpls)# global-block`
	`16000471804`
		Specifies the MPLS label range for the SRGB. Use this command if you want to change the default SRGB label range that is configured with the **segment-routing mpls** command.
		The permissive values for the beginning MPLS label and the ending MPLS label are from 16000 to 471804. The **mpls label range** command permits 16 as the minimum label, but the SRGB can start only from 16000.
		Note The minimum value for the **global-block** command starts from 16000. If you upgrading from previous releases, you should modify the SRGB so that it falls within the supported range before triggering an upgrade.

Step 4	(Optional) **show mpls label range** **Example:**	Displays the SRGB, only if the SRGB allocation is successful.

	Command or Action	Purpose
	`switch(config-segment-routing-mpls)# show mpls label range`	
Step 5	**show segment-routing**	Displays the configured SRGB.
Step 6	(Optional) **copy running-config startup-config** **Example:** `switch(config-segment-routing-mpls)# copy running-config startup-config`	Copies the running configuration to the startup configuration.

7.5.5 Configuring the Label Index

You can set the label index for routes that match the **network** command. Doing so causes the BGP prefix SID to be advertised for local prefixes that are configured with a route map that includes the **set label-index** command, provided the route map is specified in the **network** command that specifies the local prefix. (For more information on the **network** command, see the "Configuring Basic BGP" chapter in the Inspur CN12900 Series INOS-CN Unicast Routing Configuration Guide.)

SUMMARY STEPS

1. **configure terminal**
2. **route-map** *map-name*
3. [**no**] **set label-index** *index*
4. **exit**
5. **router bgp** *autonomous-system-number*
6. **address-family ipv4 unicast**
7. **network** *ip-prefix* [**route-map** *map-name*]
8. (Optional) **show route-map** [*map-name*]
9. (Optional) **copy running-config startup-config**

DETAILED STEPS

	Command or Action	Purpose

Step 1	**configure terminal**
	Example:
	`switch# configure terminal`
	`switch(config)#`
		Enters global configuration mode.

Step 2	**route-map** *map-name*
	Example:
	`switch(config)# route-map SRmap`
	`switch(config-route-map)#`
		Creates a route map or enters route-map configuration mode
		for an existing route map.

Step 3	[**no**] **set label-index** *index* **Example:** `switch(config-route-map)# set label-index 10`	Sets the label index for routes that match the **network** command. The range is from 0 to 471788. By default, a label index is not added to the route.

	Command or Action	Purpose

Step 4	**exit**
	Example:
	`switch(config-route-map)# exit`
	`switch(config)#`
		Exits route-map configuration mode.

Step 5	**router bgp** *autonomous-system-number*
	Example:
	`switch(config)# router bgp 64496`
	`switch(config-router)#`
		Enables BGP and assigns the AS number to the local BGP speaker. The AS number can be a 16-bit integer or a 32-bit integer in the form of a higher 16-bit decimal number and a lower 16-bit decimal number in xx.xx format.

Step 6	Required: **address-family ipv4 unicast**
	Example:
	`switch(config-router)# address-family ipv4 unicast`
	`switch(config-router-af)#`
		Enters global address family configuration mode for the IPv4 address family.

Step 7	**network** *ip-prefix* [**route-map** *map-name*] **Example:** `switch(config-router-af)# network 10.10.10.10/32 route-map SRmap`	Specifies a network as local to this autonomous system and adds it to the BGP routing table.
Step 8	(Optional) **show route-map** [*map-name*] **Example:** `switch(config-router-af)# show route-map`	Displays information about route maps, including the label index.
Step 9	(Optional) **copy running-config startup-config** **Example:** `switch(config-router-af)# copy running-config startup-config`	Copies the running configuration to the startup configuration.

7.5.6 Configuring Neighbor Egress Peer Engineering Using BGP

With the introduction of RFC 7752 and draft-ietf-idr-bgpls-segment-routing-epe, you can configure Egress Engineering. The feature is valid only for external BGP neighbors and it is not configured by default. Egress Engineering uses RFC 7752 encoding.

Before you begin

- You must enable BGP.

SUMMARY STEPS

1. **configure terminal**

2. **router bgp** <*bgp autonomous number*>
3. **neighbor** <*IP address*>
4. [**no**|**default**] **egress-engineering** [**peer-set** *peer-set-name*]

DETAILED STEPS

	Command or Action	Purpose

Step 1	**configure terminal**
	Example:
	`switch# configure terminal`
	`switch(config)#`
		Enters global configuration mode.

Step 2	**router bgp** <*bgp autonomous number*>	Specifies the autonomous router BGP number.
Step 3	**neighbor** <*IP address*>	Configures the IP address for the neighbor.

Step 4	[**no**|**default**] **egress-engineering** [**peer-set** *peer-set-name*]
	Example:
	`switch(config)# router bgp 1`
	`switch(config-router)# neighbor 4.4.4.4`
	`switch(config-router)# egress-engineering peer-set NewPeer`
		Specifies whether a Peer-Node-SID is allocated for the neighbor and it is advertised in an instance of a BGP Link-State (BGP-LS) address family Link NLRI. If the neighbor is a multi-hop neighbor, a BGP-LS Link NLRI instance is also advertised for each Equal-Cost-MultiPath (ECMP) path to the neighbor and it includes a unique Peer-Adj-SID.
		Optionally, you can add the neighbor to a peer-set. The Peer-Set-SID is also advertised in the BGP-LS Link NLRI in the same instance as the Peer-Node-SID. BGP Link-State NLRI is advertised to all neighbors with the link-state address family configured.
		See RFC 7752 and
		draft-ietf-idr-bgpls-segment-routing-epe-05 for more information on EPE.

7.5.7 Configuration Example for Egress Peer Engineering

See the Egress Peer Engineering sample configuration for the BGP speaker 1.1.1.1. Note that the neighbor 20.20.20.20 is the SDN controller.

```
 hostname epe-as-1
install feature-set
mpls feature-set mpls
feature telnet
```

```
feature bash-shell
feature scp-server
feature bgp
feature mpls segment-routing
segment-routing
mpls vlan 1
vrf context management
  ip route 0.0.0.0/0 10.30.97.1
  ip route 0.0.0.0/0 10.30.108.1
interface Ethernet1/1
  no switchport
  ip address 10.1.1.1/24
  no shutdown
interface Ethernet1/2
  no switchport
  ip address 11.1.1.1/24
  no shutdown
interface Ethernet1/3
  no switchport
  ip address 12.1.1.1/24
  no shutdown
interface Ethernet1/4
  no switchport
  ip address 13.1.1.1/24
  no shutdown
interface Ethernet1/5
  no switchport
  ip address 14.1.1.1/24
  no shutdown
interface mgmt0

  ip address dhcp
  vrf member management

interface loopback1
  ip address 1.1.1.1/32 line
console
line vty
ip route 2.2.2.2/32 10.1.1.2 ip
route 3.3.3.3/32 11.1.1.3 ip route
3.3.3.3/32 12.1.1.3 ip route
4.4.4.4/32 13.1.1.4
ip route 20.20.20.20/32 14.1.1.20
router bgp 1
  address-family ipv4 unicast
  address-family link-state
 neighbor 10.1.1.2 remote-as 2
    address-family ipv4
    egress-engineering
 neighbor 3.3.3.3 remote-as 3
   address-family ipv4 update-
   source loopback1 ebgp-
```

```
    multihop 2 egress-
    engineering
  neighbor 4.4.4.4 remote-as 4
    address-family ipv4 update-
    source loopback1 ebgp-
    multihop 2 egress-
    engineering
 neighbor 20.20.20.20 remote-as 1
    address-family link-state
    update-source loopback1 ebgp-
    multihop 2
 neighbor 124.11.50.5 bfs
     remote-as 6
     update-source port-channel50.11 egress-
     engineering peer-set pset2 <<<<<<<
     address-family ipv4 unicast
 neighbor 124.11.101.2
     bfd remote-as 6
     update-source Vlan2401 egress-
     engineering address-family ipv4
     unicast
```

This example shows sample output for the **show bgp internal epe** command.

```
switch# show bgp internal epe
BGP Egress Peer Engineering (EPE) Information:
Link-State Server: Inactive
Link-State Client: Active
Configured EPE Peers: 26
Active EPE Peers: 3 EPE
SID State:
RPC SID Peer or Set Assigned
ID Type Set Name ID Label Adj-Info, iod 1
Node 124.1.50.5 1 1600
```

```
    2 Set pset1 2 1601
    3 Node 6.6.6.6 3 1602
    4 Node 124.11.50.5 4 1603
    5 Set pset2 5 1604
    6 Adj 6.6.6.6 6 1605 124.11.50.4->124.11.50.5/0x1600b031, 80
    7 Adj 6.6.6.6 7 1606 124.1.50.4->124.1.50.5/0x16000031,
    78 EPE Peer-Sets:
    IPv4 Peer-Set: pset1, RPC-Set 2, Count 7, SID 1601
    Peers: 124.11.116.2 124.11.111.2 124.11.106.2
    124.11.101.2 124.11.49.5 124.1.50.5 124.1.49.5
    IPv4 Peer-Set: pset2, RPC-Set 5, Count 5, SID 1604
    Peers: 124.11.117.2 124.11.112.2 124.11.107.2
    124.11.102.2 124.11.50.5
    IPv4 Peer-Set: pset3, RPC-Set 0, Count 4, SID unspecified
    Peers: 124.11.118.2 124.11.113.2 124.11.108.2 124.11.103.2
    IPv4 Peer-Set: pset4, RPC-Set 0, Count 4, SID unspecified
    Peers: 124.11.119.2 124.11.114.2 124.11.109.2 124.11.104.2
    IPv4 Peer-Set: pset5, RPC-Set 0, Count 4, SID unspecified
    Peers: 124.11.120.2 124.11.115.2 124.11.110.2
    124.11.105.2 switch#
```

7.5.8 Configuring the BGP Link State Address Family

You can configure the BGP link state address family for a neighbor session with a controller to advertise the corresponding SIDs. You can configure this feature in global configuration mode and neighbor address family configuration mode.

Before you begin

You must enable BGP.

SUMMARY STEPS

1. **configure terminal**
2. **router bgp** *<bgp autonomous number>*
3. [**no**] **address-family link-state**
4. **neighbor** *<IP address>*
5. [**no**] **address-family link-state**

DETAILED STEPS

	Command or Action	Purpose

Step 1	**configure terminal**
	Example:
	`switch# configure terminal`
	`switch(config)#`
		Enters global configuration mode.

Step 2	**router bgp** *<bgp autonomous number>*	Specifies the autonomous router BGP number.

Step 3	[**no**] **address-family link-state**
	Example:
	`switch(config)# router bgp 64497`
	
	`switch (config-router af)# address-family link-state`
		Enters address-family interface configuration mode.
		Note This command can also be configured in neighbor address-family configuration mode.

Step 4	**neighbor** *<IP address>*	Configures the IP address for the neighbor.

Step 5	[**no**] **address-family link-state**
	Example:
	`switch(config)#router bgp 1`
	
	`switch(config-router)#address-family link-state`
	`switch(config-router)#neighbor 20.20.20.20`
	`switch(config-router)#address-family link-state`
		Enters address-family interface configuration mode.
		Note This command can also be configured in neighbor address-family configuration mode.

7.6 Configuring Layer 3 EVPN over Segment Routing MPLS

You can configure EVPN over segment routing MPLS.

Before you begin

Install the appropriate Advantage license package:

- CN129-AD-M4 for CN12904
- CN129-AD-M8 for CN12908

Make sure that the **feature interface-vlan** command is enabled.

SUMMARY STEPS

1. **feature bgp**
2. **install feature-set mpls**
3. **feature-set mpls**
4. **feature mpls segment-routing**
5. **feature mpls evpn**
6. **feature mpls l3vpn**

DETAILED STEPS

	Command or Action	Purpose
Step 1	**feature bgp**	Enables BGP feature and configurations.
Step 2	**install feature-set mpls**	Enables MPLS configuration commands.
Step 3	**feature-set mpls**	Enables MPLS configuration commands.
Step 4	**feature mpls segment-routing**	Enables segment routing configuration commands.
Step 5	**feature mpls evpn**	Enables EVPN over MPLS configuration commands. This command is mutually exclusive with the **feature-nv** CLI command.
Step 6	**feature mpls l3vpn**	Enables EVPN over MPLS configuration commands. This command is mutually exclusive with the **feature-nv** CLI command.

Example

See the following example for VRF configuration:

```
vrf context customer1 rd auto
    address-family ipv4 unicast route-target import
        route-target export route-target import evpn
        route-target export evpn
```

See the following example for BGP segment routing configuration:

```
mpls label range 1000 25000 segment-routing mpls
  global-block 11000 20000
!
int lo1
  ip address 200.0.0.1/32
!
interface e1/13
  description "MPLS interface towards Core" ip address
```

```
  192.168.5.1/24
  mpls ip forwarding no shut
route-map label_index_pol_100 permit 10 set label-index 100
route-map label_index_pol_101 permit 10 set label-index 101
route-map label_index_pol_102 permit 10 set label-index 102
route-map label_index_pol_103 permit 10 set label-index 103
router bgp 100
 neighbor 10.0.0.1 remote-as 200 update-source
    loopback0 address-family vpnv4 unicast
      import l2vpn evpn reoriginate address-family vpnv6
    unicast
      import l2vpn evpn reoriginate neighbor 20.0.0.1 remote-
  as 100
    address-family l2vpn evpn import vpn unicast reoriginate
      encapsulation mpls
  vrf LTE
    address-family ipv4 unicast advertise l2vpn evpn
    address-family ipv6 unicast advertise l2vpn evpn

router bgp 65000 address-family ipv4 unicast
   network 200.0.0.1/32 route-map label_index_pol_100 network 192.168.5.1/32
    route-map label_index_pol_101 network 101.0.0.0/24 route-map
    label_index_pol_103 allocate-label all
  neighbor 192.168.5.6 remote-as 65000 address-family ipv4 labeled-
      unicast
         send-community extended
```

7.7 Configuring BGP EVPN and Label Allocation Mode

You can use MPLS tunnel encapsulation using the **encapsulation mpls** command. You can configure the label allocation mode for the EVPN address family. The default tunnel encapsulation in EVPN for IP Route type in INOS-CN is VXLAN.

Advertisement of (IP or Label) bindings from an Inspur CN12900 Series switch via BGP EVPN enables a remote switch to send the routed traffic to that IP using the label for that IP to the switch that advertised the IP over MPLS.

The IP prefix route (Type-5) is:

• Type-5 route with MPLS encapsulation

```
RT-5 Route – IP Prefix

RD: L3 RD
IP Length:   prefix length
IP address:  IP (4 bytes)
Label1:  BGP MPLS Label
Route Target
RT for IP-VRF
```

The default label allocation mode is per-VRF for Layer 3 EVPN over MPLS.

Complete the following steps to configure BGP EVPN and label allocation mode:

Before you begin

You must install and enable the MPLS feature set using the **install feature-set mpls** and **feature-set mpls** commands.

You must enable the MPLS segment routing feature.

SUMMARY STEPS

1. **configure terminal**
2. [**no**] **router bgp** *autonomous-system-number*
3. **address-family l2vpn evpn**
4. **exit**
5. **neighbor** *ipv4-address* **remote-as** *autonomous-system-number*
6. **address-family l2vpn evpn**
7. **encapsulation mpls**
8. **vrf** *<customer_name>*
9. **address-family ipv4 unicast**
10. **advertise l2vpn evpn**
11. **redistribute direct route-map DIRECT_TO_BGP**
12. **label-allocation-mode per-vrf**

DETAILED STEPS

	Command or Action	Purpose
Step 1	**configure terminal**	Enters global configuration mode.

Step 2	[**no**] **router bgp** *autonomous-system-number*
	
	Example:
	`switch(config)# router bgp 64496`
	`switch(config-router)#`
		Enables BGP and assigns the AS number to the local BGP speaker. The AS number can be a 16-bit integer or a 32-bit integer in the form of a higher 16-bit decimal number and a lower 16-bit decimal number in xx.xx format. Use the **no** option with this command to remove the BGP process and the associated configuration.

Step 3	Required: **address-family l2vpn evpn**
	
	Example:
	`switch(config-router)# address-family l2vpn evpn`
	`switch(config-router-af)#`
		Enters global address family configuration mode for the Layer 2 VPN EVPN.

Step 4	Required: **exit**
	Example:
	`switch(config-router-af)# exit`
	`switch(config-router)#`
		Exits global address family configuration mode.

Step 5	**neighbor** *ipv4-address* **remote-as**
	
	autonomous-system-number
	Example:
	`switch(config-router)# neighbor 10.1.1.1 remote-as 64497`
	`switch(config-router-neighbor)#`
		Configures the IPv4 address and AS number for a remote BGP peer.

Step 6	**address-family l2vpn evpn**	Advertises the labeled Layer 2 VPN EVPN.

	Command or Action	Purpose

	Example:
	`switch(config-router-neighbor)# address-family`
	`l2vpn evpn`
	`switch(config-router-neighbor-af)#`	

Step 7	**encapsulation mpls**
	Example:
	`router bgp 100`
	`  address-family l2vpn evpn`
	`neighbor NVE2 remote-as 100`
	`    address-family l2vpn evpn`
	`      send-community extended`
	`      encapsulation mpls`
	`  vrf foo`
	`    address-family ipv4 unicast`
	`      advertise l2vpn evpn`
	BGP segment routing configuration:
	`router bgp 100`
	` address-family ipv4 unicast`
	`   network 200.0.0.1/32 route-map`
	`label_index_pol_100`
	`    network 192.168.5.1/32 route-map`
	`label_index_pol_101`
	`    network 101.0.0.0/24 route-map`
	`label_index_pol_103`
	`    allocate-label all`
	`  neighbor 192.168.5.6 remote-as 20`
	`      address-family ipv4 labeled-unicast`
	`         send-community extended`
		Enables BGP EVPN address family and sends EVPN type-5 route update to the neighbors.
		Note The default tunnel encapsulation in EVPN for the IP route type in INOS-CN is VXLAN. To override that, a CLI is introduced to indicate MPLS tunnel encapsulation.

Step 8	**vrf** *<customer_name>*	Configures the VRF.
Step 9	**address-family ipv4 unicast**	Enters global address family configuration mode for the IPv4 address family.
Step 10	**advertise l2vpn evpn**	Advertises Layer 2 VPN EVPN.
Step 11	**redistribute direct route-map DIRECT_TO_BGP**	Redistributes the directly connected routes into BGP-EVPN.
Step 12	**label-allocation-mode per-vrf**	Sets the label allocation mode to per-VRF. If you want to configure the per-prefix label mode, use the **no label-allocation-mode per-vrf** CLI command. For the EVPN address family, the default label

	Command or Action	Purpose
		allocation is per-vrf, compared to per-prefix mode for the other address-families where the label allocation CLI is supported. No form of CLI is displayed in the running configuration.

Example

See the following example for configuring per-prefix label allocation:

```
router bgp 65000 [address-
    family l2vpn evpn]
    neighbor 10.1.1.1
        remote-as 100 address-
        family l2vpn evpn send-
        community extended
    neighbor 20.1.1.1
        remote-as 65000
        address-family l2vpn
        evpn encapsulation mpls
        send-community extended
    vrf customer1
        address-family ipv4 unicast
            advertise l2vpn evpn
            redistribute direct route-map DIRECT_TO_BGP
            no label-allocation-mode per-vrf
```

7.8 Configuring Segment Routing with IS-IS Protocol

You can configure segment routing with IS-IS protocol.

Before you begin

IS-IS segment routing is fully enabled when the following conditions are met:
- The **mpls segment-routing** feature is enabled.
- The IS-IS feature is enabled.
- Segment routing is enabled for at least one address family under IS-IS.

SUMMARY STEPS

1. **configure terminal**
2. **router isis** *instance-tag*
3. **net** *network-entity-title*
4. (Optional) **is-type** {*level-1* | *level-2* | *level-1-2*}
5. **log-adjacency-changes**
6. **address-family** *ipv4* **unicast**
7. **segment-routing mpls**
8. (Optional) **show running-config segment-routing**

DETAILED STEPS

	Command or Action	Purpose
Step 1	**configure terminal**	Enters global configuration mode.
Step 2	**router isis** *instance-tag*	Creates a new IS-IS instance with the configured instance tag.
Step 3	**net** *network-entity-title*	Configures the NET for this IS-IS instance.
Step 4	(Optional) **is-type** {*level-1* | *level-2* | *level-1-2*}	Configures the area level for this IS-IS instance. The default is level-1-2.
Step 5	**log-adjacency-changes**	Sends a system message whenever an IS-IS neighbor changes the state.
Step 6	**address-family** *ipv4* **unicast**	Enters address family configuration mode.
Step 7	**segment-routing mpls**	Configures segment routing with IS-IS protocol. **Note** The IS-IS command is supported only on the IPv4 address family. It is not supported on the IPv6 address family. Redistribution is not supported from any other protocol to ISIS for the SR prefixes. You need to enable **ip router isis** command on all the prefix SID interfaces.
Step 8	(Optional) **show running-config segment-routing**	Displays the status of the segment routing.

See the following configuration example for configuring segment routing with IS-IS protocol.

Example

```
switch# config t
router isis SR-ISIS-1
  bfd
  net 31.0000.0000.0000.000e.00
  is-type level-1-2
  log-adjacency-changes
  address-family ipv4 unicast    >>> # New command added for ISIS.
    segment-routing mpls
    address-family ipv6 unicast
    bfd

switch# show running-config segment-routing

!Command: show running-config segment-routing
!Time: Mon Oct 1 12:51:59 2018
```

```
version 9.2(1i)
segment-routing mpls
  global-block 201000 280000
  connected-prefix-sid-map
    address-family ipv4
      2.1.1.5/32 absolute 201101
      2.10.1.5/32 index 10001

switch# show running-config isis

!Command: show running-config isis
!Time: Mon Oct 1 10:18:19 2018

version 9.2(1i)

feature isis

router isis 10
  bfd
  net 56.0000.0000.0003.00
  is-type level-1-2 maximum-
  paths 64 log-adjacency-
  changes address-family ipv4
  unicast
    segment-routing mpls

interface Vlan12

  ip router isis 10

interface Vlan13
  ip router isis 10
```

7.9 Configuring Segment Routing with OSPFv2

Before you begin

OSPFv2 segment routing is fully enabled when the following conditions are met:

- The mpls segment-routing feature is enabled.
- The OSPFv2 feature is enabled.
- Segment routing is enabled under OSPF.

SUMMARY STEPS

1. **configure terminal**
2. [**no**]**router ospf**
3. **segment-routing mpls**

DETAILED STEPS

	Command or Action	Purpose
Step 1	**configure terminal**	Enters global configuration mode.
Step 2	[**no**]**router ospf**	Enables the OSPF mode.

	Command or Action	Purpose
Step 3	**segment-routing mpls**	Configures the Segment Routing functionality

See the following configuration example for configuring segment routing with OSPFv2.

Example

```
switch# show running-config
ospf !Command: show running-config
ospf
!Running configuration last done at: Mon Oct 1 15:09:07 2018
!Time: Mon Oct 1 15:09:09 2018

version 9.2(1i) Bios:version

07.60 feature ospf

router ospf SR_OSPF
  segment-routing mpls
 router-id 2.2.2.1

interface loopback1

  ip router ospf SR_OSPF area 0.0.0.0

switch# show running-config interface loopback
1 !Command: show running-config interface loopback1
!Running configuration last done at: Mon Oct 1 15:11:16 2018
!Time: Mon Oct 1 15:13:05 2018

version 9.2(1i) Bios:version 07.60

interface loopback1
  ip address 2.2.2.1/32
  ip router ospf SR_OSPF area 0.0.0.0

switch# show running-config segment-
routing !Command: show running-config segment-
routing
!Running configuration last done at: Mon Oct 1 15:11:16 2018
!Time: Mon Oct 1 15:11:54 2018

version 9.2(1i) Bios:version
07.60 segment-routing mpls
  global-block 201000 400000
  connected-prefix-sid-map
    address-family ipv4
      2.2.2.1/32 absolute 201101
```

7.10 About Segment Routing for Traffic Engineering

Segment routing for traffic engineering (SR-TE) takes place through a tunnel between a source and destination pair. Segment routing for traffic engineering uses the concept of source routing, where the source calculates the path and encodes it in the packet header as a segment. A Traffic Engineered (TE) tunnel is a container of TE LSPs instantiated between the tunnel ingress and the tunnel destination. A TE tunnel can instantiate one or more SR-TE LSPs that are associated with the same tunnel.

With segment routing for traffic engineering (SR-TE), the network no longer needs to maintain a per-application and per-flow state. Instead, it simply obeys the forwarding instructions provided in the packet.

SR-TE utilizes network bandwidth more effectively than traditional MPLS-TE networks by using ECMP at every segment level. It uses a single intelligent source and relieves remaining routers from the task of calculating the required path through the network.

7.10.1 About SR-TE Policies

Segment routing for traffic engineering (SR-TE) uses a “policy” to steer traffic through the network. A segment routing policy is a container that includes sets of segments or labels. This list of segments can be provisioned by an operator, a stateful PCE or the SR-TE infra can dynamically calculate the path by applying Constrained Shortest Path First (CSPF) algorithm on its local IGP database. The head-end imposes the corresponding MPLS label stack on a traffic flow to be carried over the SR-TE policy. Each transit node along the SR-TE policy path uses the incoming top label to select the next-hop, pop or swap the label, and forward the packet to the next node with the remainder of the label stack, until the packet reaches the ultimate destination.

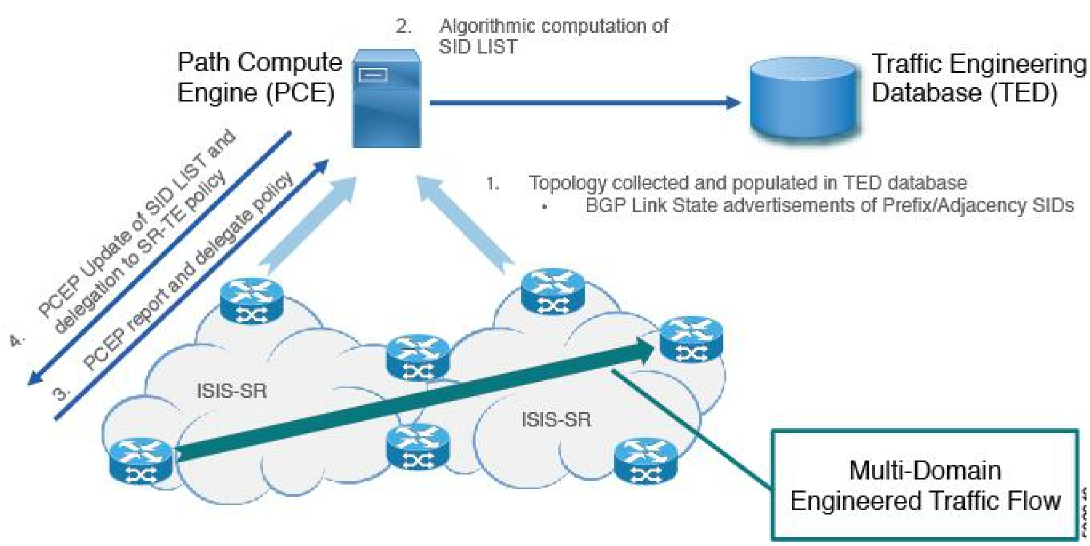
At a given head-end, a SR Policy is uniquely identified by a tuple (color, end-point). A color is represented as a 32-bit number and an end-point is either an IPv4 and IPv6 address. Multiple SR-TE policies can be created between the same two endpoints by choosing different colors for the policies.

Local dynamic SR-TE policy is supported. When you configure local dynamic SR-TE, the head-end locally calculates the path to the destination address. Dynamic path calculation results in a list of interface IP addresses that traffic engineering (TE) maps to adj-SID labels. Routes are learned by way of forwarding adjacencies over the TE tunnel.

7.10.2 About Segment Routing On Demand Next Hop

On-Demand Next hop (ODN) leverages upon BGP Dynamic SR-TE capabilities and adds the path computation (PCE) ability to find and download the end to end path based on the requirements. ODN triggers an SR-TE auto-tunnel based on the defined BGP policy. As shown in the following figure, an end-to-end path between ToR1 and AC1 can be established from both ends based on IGP Metric. The work-flow for ODN is summarized as follows:

Figure 10： ODN Operation



7.10.3 Guidelines and Limitations for SR-TE On-Demand Next Hop

SR-TE ODN has the following guidelines and limitations:

- ODN for IPv6 is not supported.

- SR-TE ODN is supported only with ISIS Underlay.

7.11 Configuring SR-TE

You can configure segment routing for traffic engineering.

Before you begin

You must ensure that the MPLS segment routing feature is enabled.

SUMMARY STEPS

1. **configure terminal**
2. **feature mpls segment-routing traffic-engineering**
3. **segment-routing**
4. **traffic-engineering**
5. **encapsulation mpls source ipv4** *tunnel_ip_address*
6. **pcc**
7. **source-adress ipv4** *pcc_source_address*
8. **pce-address ipv4 pce_source_address** *precedence* **num**
9. **on-demand color** *color_num*
10. **metric-type** *igp*

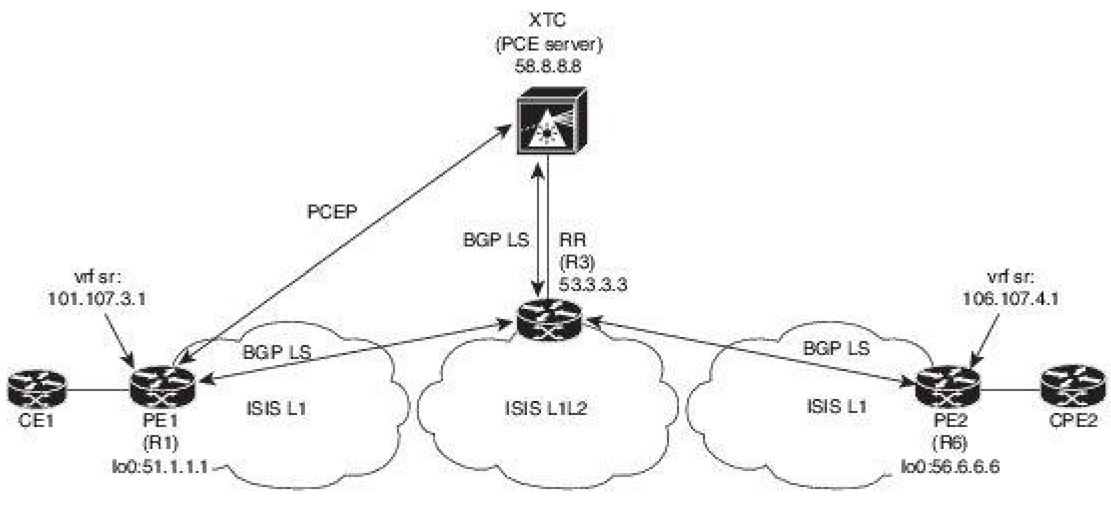
DETAILED STEPS

	Command or Action	Purpose
Step 1	**configure terminal**	Enters global configuration mode.
Step 2	**feature mpls segment-routing traffic-engineering**	Enables mpls SR-TE.
Step 3	**segment-routing**	Enters the segment-routing mode
Step 4	**traffic-engineering**	Enters the traffic engineering mode.
Step 5	**encapsulation mpls source ipv4** *tunnel_ip_address*	Configures the source address for the SR TE Tunnel.
Step 6	**pcc**	Enters the PCC mode.
Step 7	**source-adress ipv4** *pcc_source_address*	Configure source address for the PCC
Step 8	**pce-address ipv4 pce_source_address** *precedence* **num**	Configure IP address of the PCE. The lowest numbered PCE will take precedence, and the other(s) be used as a backip.
Step 9	**on-demand color** *color_num*	Enters the on-demand mode to configure the color.
Step 10	**metric-type** *igp*	Configures the metric type.

7.12 Configuration Example for a SR-TE ODN - Usecase

Perform the following steps to configure ODN for SR-TE. The following figure is used as a reference to explain the configuration steps.

Figure 11： Reference Topology



1. Configure all links with ISIS point-to-point session from PE1 to PE2. Also, configure the domains as per the above topology.

2. Enable “distribute link-state” for ISIS session on R1, R3, and R6.

```
router isis 1
  net   31.0000.0000.0000.712a.00
  log-adjacency-changes
  distribute link-state address-
  family ipv4 unicast
    bfd
    segment-routing  mpls
    maximum-paths 32
    advertise interface loopback0
```

3. Configure the router R1 (headend) and R6 (tailend) with VRF interface.

VRF configuration on R1:

```
interface Ethernet1/49.101
encapsulation dot1q 201
  vrf member sr
  ip address 101.10.1.1/24
  no shutdown
vrf context
  sr rd auto
  address-family ipv4 unicast route-
    target import 101:101 route-
    target import 101:101 evpn
    route-target export 101:101
    route-target export 101:101 evpn
```

```
router bgp
  6500 vrf sr
    bestpath as-path multipath-relax
    address-family ipv4 unicast
      advertise l2vpn evpn
```

4. Tags VRF prefix with BGP community on R6 (tailend).

```
route-map color1001 permit 10
  set extcommunity color 1001
```

5. Enable BGP on R6 (tailend) and R1 (headend) to advertise and receive VRF SR prefix and match on community set on R6 (tailend).

R6 < EVPN > R3 < EVPN > R1

BGP Configuration R6:

```
router bgp 6500 address-
  family ipv4 unicast
     allocate-label all
  neighbor 53.3.3.3
    remote-as 6500 log-
    neighbor-changes update-
    source loopback0 address-
    family l2vpn evpn
      send-community extended
     route-map Color1001 out
     encapsulation mpls
```

BGP Configuration R1:

```
router bgp 6500 address-
  family ipv4 unicast
     allocate-label all
  neighbor 53.3.3.3
    remote-as 6500 log-
    neighbor-changes update-
    source loopback0 address-
    family l2vpn evpn
      send-community extended
       encapsulation mpls
```

6. Enable BGP configuration on R3 and BGP LS with XTC on R1, R3.abd

BGP Configuration R3:

```
router bgp 6500
  router-id 2.20.1.2
address-family ipv4
unicast allocate-label all
address-family l2vpn evpn
retain route-target all
  neighbor 56.6.6.6 remote-
```

```
    as 6500 log-neighbor-
    changes update-source
    loopback0 address-family
    l2vpn evpn
      send-community extended
       route-reflector-client
       route-map NH_UNCHANGED out
      encapsulation mpls
  neighbor 51.1.1.1
    remote-as 6500 log-
    neighbor-changes update-
    source loopback0 address-
    family l2vpn evpn
      send-community extended
      route-reflector-client
      route-map NH_UNCHANGED out
      encapsulation mpls
neighbor 58.8.8.8
    remote-as 6500
        log-neighbor-changes
        update-source loopback0
        address-family link-state
    route-map NH_UNCHANGED permit
      10 set ip next-hop unchanged
```

BGP Configuration R1:

```
router bgp 6500
neighbor 58.8.8.8
              remote-as 6500 log-
               neighbor-changes update-
               source loopback0 address-
               family link-state
```

BGP Configuration R6:

```
outer bgp 6500
   neighbor 58.8.8.8
    remote-as 6500 log-neighbor-
       changes update-source
       loopback0 address-family
       link-state
```

7. Enable PCE and SR-TE tunnel configurations on R1.

```
segment-routing  traffic-
  engineering
    pcc
      source-address   ipv4   51.1.1.1
      pce-address ipv4 58.8.8.8
    on-demand   color   1001
      metric-type igp
```

7.13 Verifying SR-TE for Layer3 EVPN

The ODN verifications are based on L3VPN VRF prefixes.

1. Verify that the PCEP session between R1 (headend and PCE server) is established.

```
R1# show srte pce ipv4 peer

PCC's peer database:
--------------------
Remote PCEP conn IPv4 addr: 58.8.8.8
Local PCEP conn IPv4 addr: 51.1.1.1
Precedence: 0
State: up
```

2. Verify BGP LS and BGP EVPN session on R1, R3 and R6 using the following commands:

- Show bgp l2vpn evpn summary
- Show bgp link-state summary

3. Verify that the R1 (headend) has no visibility to the R6 loopback address.

```
R1# show ip route 56.6.6.6
IP Route Table for VRF "default"
'*' denotes best ucast next-hop
'**' denotes best mcast next-hop
'[x/y]' denotes [preference/metric]
'%<string>' in via output denotes VRF
<string>
56.6.6.6/32, ubest/mbest: 1/0
    *via Null0, [1/0], 1d02h, static
```

4. Verify that the VRF prefix is injected via MP-BGP in R1 VRF SR routing table.

```
R1#  show  ip  route  vrf  sr
106.107.4.1/32,  ubest/mbest:
1/0
    *via binding label 100534%default, [20/0], 1d01h, bgp-6503, external, tag
6500 (mpls-vpn)
```

5. Verify the SR-TE Tunnel.

```
R1# show srte policy
Policy name: 51.1.1.1|1001
    Source: 51.1.1.1
    End-point: 56.6.6.6
    Created by: bgp
    State: UP
    Color: 1001
    Insert: FALSE
    Re-opt timer: 0    100534
    Binding-sid Label:
    Policy-Id: 2
    Flags:                      Path options count: 1
    Path type = MPLS
     Path-option Preference:100 ECMP path count: 1
      1.      PCE         Weighted: No
        Delegated PCE: 58.8.8.8          Label: 101104
                Index: 1
```

```
Index: 2                Label: 201102
Index: 3                Label: 201103
```

7.14 Verifying the Segment Routing Configuration

To display the segment routing configuration, perform one of the following tasks:

Command	Purpose
show bgp ipv4 labeled-unicast *prefix*	Displays the advertised label index and the selected local label for the specified IPv4 prefix.
show bgp paths	Displays the BGP path information, including the advertised label index.
show mpls label range	Displays the configured SRGB range of labels.
show route-map [*map-name*]	Displays information about a route map, including the label index.
show running-config | inc 'feature mpls segment-routing'	Displays the status of the MPLS segment routing feature.
show running-config segment-routing	Displays the status of the segment routing feature.

This example shows how the **show bgp ipv4 labeled-unicast** command can be used with a prefix specification to display the advertised label index and the selected local label:

```
switch# show bgp ipv4 labeled-unicast 19.19.19.19/32
BGP routing table information for VRF default, address family IPv4 Label
Unicast BGP routing table entry for 19.19.19.19/32, version 2
Paths: (1 available, best #1)
Flags: (0x20c0012) on xmit-list, is in urib, is backup urib route, has
  label label af: version 2, (0x100002) on xmit-list
  local label: 16010

  Advertised path-id 1, Label AF advertised path-
  id 1 Path type: external, path is valid, is
  best path AS-Path: 19 , path sourced external
  to AS
60.1.1.19     (metric     0)     from     60.1.1.19
      (100.100.100.100) Origin IGP, MED not set,
      localpref 100, weight 0 Received label 3
      Prefix-SID Attribute: Length: 10
        Label Index TLV: Length 7, Flags 0x0 Label Index 10

  Path-id 1 not advertised to any peer

  Label AF advertisement
  Path-id 1 not advertised to any peer
```

7.15 Configuration Examples for Segment Routing

The examples in this section show a common BGP prefix SID configuration between two routers.

This example shows how to advertise a BGP speaker configuration of 10.10.10.10/32 and 20.20.20.20/32 with a label index of 10 and 20, respectively. It uses the default segment routing global block (SRGB) range of 16000 to 23999.

```
hostname s1
install feature-set
mpls feature-set
mpls
feature telnet
feature bash-
shell feature
scp-server
feature bgp
feature mpls segment-routing
segment-routing
mpls vlan 1
route-map label-index-10
  permit 10 set label-index 10
route-map label-index-20
  permit 10 set label-index 20
vrf context management
  ip route 0.0.0.0/0 10.30.108.1
interface
  Ethernet1/1 no
  switchport
  ip address
  10.1.1.1/24 no
  shutdown
interface mgmt0
  ip address dhcp
  vrf member management
interface loopback1
  ip address 10.10.10.10/32
interface loopback2
  ip address 20.20.20.20/32
line console
line vty
router bgp 1
  address-family ipv4 unicast
    network 10.10.10.10/32 route-map label-index-10
    network 20.20.20.20/32 route-map label-index-20
    allocate-label all
  neighbor 10.1.1.2 remote-as 2 address-
    family ipv4 labeled-unicast
```

This example shows how to receive the configuration from a BGP speaker.

```
hostname s2
install feature-set mpls
```

```
feature-set mpls
feature telnet
feature bash-shell
feature scp-server
feature bgp
feature mpls segment-routing
segment-routing mpls
vlan 1
vrf context management
  ip route 0.0.0.0/0 10.30.97.1 ip
  route 0.0.0.0/0 10.30.108.1
interface Ethernet1/1 no
  switchport
  ip address 10.1.1.2/24 ipv6
  address 10:1:1::2/64 no
  shutdown
interface mgmt0 ip
  address dhcp
  vrf member management
interface loopback1
  ip address 2.2.2.2/32
line console
line vty
router bgp 2
  address-family ipv4 unicast
    allocate-label all
  neighbor 10.1.1.1 remote-as 1 address-
    family ipv4 labeled-unicast
```

This example shows how to display the configuration from a BGP speaker. The **show** command in this example displays the prefix 10.10.10.10 with label index 10 mapping to label 16010 in the SRGB range of 16000 to 23999.

```
switch# show bgp ipv4 labeled-unicast 10.10.10.10/32

BGP routing table information for VRF default, address family IPv4 Label
Unicast BGP routing table entry for 10.10.10.10/32, version 7
Paths: (1 available, best #1)
Flags: (0x20c001a) on xmit-list, is in urib, is best urib route, is in HW, , has
  label label af: version 8, (0x100002) on xmit-list
  local label: 16010

  Advertised path-id 1, Label AF advertised path-id 1
  Path type: external, path is valid, is best path, no labeled nexthop, in rib
  AS-Path: 1 , path sourced external to AS
    10.1.1.1 (metric 0) from 10.1.1.1
    (10.10.10.10)
      Origin IGP, MED not set, localpref 100, weight
      0 Received label 0
      Prefix-SID Attribute: Length: 10
        Label Index TLV: Length 7, Flags 0x0 Label Index 10

  Path-id 1 not advertised to any
```

```
    peer Label AF advertisement
    Path-id 1 not advertised to any peer
```

This example shows how to configure egress peer engineering on a BGP speaker.

```
hostname epe-as-1
install feature-set
mpls feature-set mpls
feature telnet
feature bash-
shell feature
scp-server
feature bgp
feature mpls segment-routing
segment-routing
mpls vlan 1
vrf context management
  ip route 0.0.0.0/0 10.30.97.1
  ip route 0.0.0.0/0 10.30.108.1
interface
  Ethernet1/1 no
  switchport
  ip address
  10.1.1.1/24 no
  shutdown
interface
  Ethernet1/2 no
  switchport
  ip address
  11.1.1.1/24 no
  shutdown
interface
  Ethernet1/3 no
  switchport
  ip address
  12.1.1.1/24 no
  shutdown
interface
  Ethernet1/4 no
  switchport
  ip address 13.1.1.1/24
 no shutdown

interface Ethernet1/5
  no switchport
  ip address 14.1.1.1/24
  no shutdown
```

The following is an example of show ip route vrf 2 command.

```
show ip route vrf 2
IP Route Table for VRF "2"
```

```
'*' denotes best ucast next-hop '**'
denotes best mcast next-hop '[x/y]'
denotes [preference/metric]
'%<string>' in via output denotes VRF <string>

41.11.2.0/24, ubest/mbest: 1/0
    *via 1.1.1.9%default, [20/0], 13:26:48, bgp-2, external, tag 11 (mpls-
vpn) 42.11.2.0/24, ubest/mbest: 1/0, attached
    *via  42.11.2.1,  Vlan2,  [0/0],  13:40:52,
direct 42.11.2.1/32, ubest/mbest: 1/0, attached
    *via 42.11.2.1, Vlan2, [0/0], 13:40:52, local
```

The following is an example of **show forwarding route vrf 2** command.

```
slot   1
=======

IPv4 routes for table 2/base

Prefix                                           +----------------------    +---------------- | Labels
                  | Next-hop                                          | Interface
          | Partial___________________________Install +----------------------    +----------------
------------------      +      Drop
0.0.0.0/32                                                           Null0
127.0.0.0/8          Drop                                            Null0
255.255.255.255/32   Receive                                         sup-eth1             PUSH
*41.11.2.0/24        27.1.31.4                                       Ethernet1/3
  30002  492529      27.1.32.4                                       Ethernet1/21         PUSH
  30002  492529
                     27.1.33.4                                       port-channel23       PUSH
  30002  492529
                     27.11.31.4                                      Ethernet1/3.11       PUSH
  30002  492529
                     27.11.33.4                                      port-channel23.11    PUSH
  30002  492529
                     37.1.53.4                                       Ethernet1/53/1       PUSH
  29002  492529
                     37.1.54.4                                       Ethernet1/54/1       PUSH
  29002  492529
                     37.2.53.4                                       Ethernet1/53/2       PUSH
  29002  492529
                     37.2.54.4                                       Ethernet1/54/2       PUSH
  29002  492529
                     80.211.11.1                                     Vlan801              PUSH
  30002  492529
```

The following is an example of **show bgp l2vpn evpn summary** command.

```
show bgp l2vpn evpn summary
BGP summary information for VRF  default, address family L2VPN EVPN
BGP router identifier 2.2.2.3, local AS number  2
BGP table version is 17370542, L2VPN EVPN config peers 4, capable peers 1
1428 network entries and 1428 paths using 268464 bytes of memory
BGP attribute entries [476/76160], BGP AS path entries [1/6]
BGP community entries [0/0], BGP clusterlist entries [0/0]
476 received paths for inbound soft reconfiguration
476 identical, 0 modified, 0 filtered received  paths using 0 bytes

Neighbor        V    AS MsgRcvd  MsgSent   TblVer  InQ OutQ Up/Down  State/PfxRcd
1.1.1.1         4    11       0        0        0    0    0 23:01:53 Shut (Admin)
```

```
1.1.1.9         4    11     4637    1836 17370542    0    0 23:01:40 476
1.1.1.10        4    11        0       0        0    0    0 23:01:53 Shut (Admin)
1.1.1.11        4    11        0       0        0    0    0 23:01:52 Shut (Admin)
```

The following is an example of **show bgp l2vpn evpn** command.

```
show bgp l2vpn evpn 41.11.2.0
BGP routing table information for VRF default, address family L2VPN EVPN Route Distinguisher:
14.1.4.1:115
BGP routing table entry for [5]:[0]:[0]:[24]:[41.11.2.0]:[0.0.0.0]/224, version 17369591 Paths: (1 available,
best #1)
Flags: (0x000002) on xmit-list, is not in l2rib/evpn, is not in HW

  Advertised path-id 1
  Path type: external, path is valid, received and used, is best path Imported to 2
             destination(s)
  AS-Path: 11 , path sourced external to AS 1.1.1.9 (metric 0) from
    1.1.1.9 (14.1.4.1)
      Origin incomplete, MED 0, localpref 100, weight 0 Received label
      492529
      Extcommunity: RT:2:20

  Path-id 1 not advertised to any peer

Route Distinguisher: 2.2.2.3:113
BGP routing table entry for [5]:[0]:[0]:[24]:[41.11.2.0]:[0.0.0.0]/224, version 17369595 Paths: (1 available,
best #1)
Flags: (0x000002) on xmit-list, is not in l2rib/evpn, is not in HW

  Advertised path-id 1
  Path type: external, path is valid, is best path
             Imported from 14.1.4.1:115:[5]:[0]:[0]:[24]:[41.11.2.0]:[0.0.0.0]/224 AS-Path: 11 , path sourced
  external to AS
    1.1.1.9 (metric 0) from 1.1.1.9 (14.1.4.1)
```

7.16 Additional References

7.16.1 Related Documents

Related Topic	Document Title
BGP	Inspur CN12900 Series INOS-CN Unicast Routing Configuration Guide

CHAPTER 8 Configuring MPLS Segment Routing OAM

This chapter describes the Multiprotocol Label Switching (MPLS) segment routing OAM functionality.

8.1 Overview of MPLS Segment Routing OAM

BGP MPLS segment routing (SR) has been deployed on the Inspur CN12900 Series switches. As MPLS segment routing (SR) is deployed, a few diagnostic tools are required to help resolve the misconfigurations or failures in the segment routing network. Only Nil FEC is supported and none of the other FEC types are supported. The Nil FEC is the basic OAM FEC that is described in RFC-4379.

MPLS OAM provides two main functions for diagnostics purposes:

1. MPLS ping
2. MPLS traceroute

OAM draws the information from the FEC type to help diagnose the issues. The Nil FEC is not associated with a protocol like the other FEC types, and it is also not associated with a real FEC. For example, it is not associated with LDP etc. Logically, it only validates the data plane programming; it does not query the BGP or other routing protocols in the control plane unlike other FEC types.

To enable MPLS OAM on Inspur CN12900 Series switches, use the **feature mpls oam** CLI command. Use the **no feature mpls oam** CLI command to disable MPLS OAM on Inspur CN12900 Series switches.

8.2 Segment Routing OAM Support for LSP Ping and Traceroute

The Nil-FEC LSP ping and traceroute operations are extensions of regular MPLS ping and traceroute. Nil-FEC LSP Ping/Traceroute functionality supports segment routing and MPLS Static. It also acts as an additional diagnostic tool for all other LSP types. This feature allows operators to provide the ability to freely test any label stack by allowing them to specify the following:

- Label stack
- Outgoing interface
- Nexthop address

In case of segment routing, each segment nodal label and adjacent label along the routing path is put into the label stack of an echo request message from the initiator Label Switch Router (LSR); MPLS data plane forwards this packet to the label stack target, and the label stack target sends the echo message back.

Use the **ping mpls nil-fec labels** *comma-separated-labels* [**output** {**interface** *tx-interface*} [**nexthop** *nexthop-ip-addr*]] CLI command to execute a ping. Use the **traceroute mpls nil-fec labels** *comma-separated-labels* [**output** {**interface** *tx-interface*} [**nexthop** *nexthop-ip-addr*]] CLI command to execute a traceroute.

8.3 Guidelines and Limitations

See the following guidelines and limitations for configuring MPLS OAM Nil FEC:

- A maximum of 4 labels can be specified in the **ping mpls nil-fec** and **traceroute mpls nil-fec** commands. This value is enforced by querying the platform and currently Inspur CN12900 Series switches limit the label stack to 5. It means that for a Nil FEC echo request, you can specify a maximum of 4 labels because internally an extra explicit-null is added.
- The nexthop specified in the ping and traceroute commands must be a connected nexthop on the originator and it should not be a recursive nexthop.
- There is no support for treetrace.

- Nil FEC does not carry any information to identify the intended target. The packet may mis-forward at an incorrect node but the validation may return success if the packet ends up at a node after popping the non-null labels.
- Nil FEC operates on forwarding the information alone. It cannot detect the inconsistencies between the control plane and the forwarding plane by definition.
- Nil FEC ping and traceroute is not supported for deaggregator (per-VRF) labels. This includes the BGP EVPN-Layer 3 deaggregator labels.
- On Inspur CN12900 Series switches that use Broadcom chipsets, there is no support to allow the software to send a query to determine which ECMP a packet takes. It means that for MPLS traceroutes that traverse one of these switches may display an error at the next hop if there is more than one ECMP as displayed in the following example:

```
D 2 6.0.0.2 MRU 1496 [Labels: 2003/explicit-null Exp: 0/0] 4 ms
```

- When you use OAM to test a BGP EPE LSP (for example, the last label in the ping/traceroute label stack is an EPE label), OAM only returns success if the final router has OAM enabled and MPLS is enabled on the incoming interface.

For example, if you have a setup as A---B---C, A and B are in the SR network, and B acts like a PE and C acts like a CE, B is configured with C as a BGP EPE peer (using egress-engineering on B), then C must have OAM and MPLS forwarding enabled on the incoming interface.

8.4 Examples for Using Ping and Traceroute CLI Commands

Using CLI to Execute a Ping

Use the **ping mpls nil-fec labels** *comma-separated-labels* [**output** {**interface** *tx-interface*} [**nexthop** *nexthop-ip-addr*]] CLI command to execute a ping.

For example, the following command sends an MPLS packet with the outermost two labels in the label stack being 2001 and 2000 out the interface Ethernet 1/1 with a nexthop IP address of 4.0.0.2:

```
switch# ping mpls nil-fec labels 2001,2000 output interface e1/1 nexthop 4.0.0.2
```

It is mandatory that the nexthop is a connected nexthop; it is not recursively resolved.

The above CLI format is a simplified version. The [**output** {**interface** *tx-interface*} [**nexthop** *nexthop-ip-addr*]] is mandatory to be present in the VSH server. For example:

```
switch# ping mpls nil-fec labels
1,2 ? output Output options
switch# ping mpls nil-fec labels 1,2
^
% Invalid command at '^' marker.
```

Using CLI to Execute a Traceroute

Use the following CLI command to execute a traceroute:

```
traceroute mpls nil-fec labels <comma-separated-labels> output interface <tx-interface>
nexthop <nexthop-ip-addr>
```

Displaying Show Statistics

Use the following command to display the statistics about the echo requests sent by the local MPLS OAM service:

```
show mpls oam echo statistics
```

CHAPTER 9 InterAS Option B

This chapter explains the different InterAS option B configuration options. The available options are InterAS option B, InterAS option B (with RFC 3107), and InterAS option B lite. The InterAS option B (with RFC 3107) implementation ensures complete IGP isolation between the data centers and WAN. When BGP advertises a particular route to ASBR, it also distributes the label which is mapped to that route.

9.1 Information About InterAS

An autonomous system (AS) is a single network or group of networks that is controlled by a common system administration group and using a single, clearly defined protocol. In many cases, virtual private networks (VPNs) extend to different ASes in different geographical areas. Some VPNs must extend across multiple service providers; these VPNs are called overlapping VPNs. The connection between ASes must be seamless to the customer, regardless of the complexity or location of the VPNs.

9.1.1 InterAS and ASBR

Separate ASes from different service providers can communicate by exchanging information in the form of VPN IP addresses. The ASBRs use EBGP to exchange that information. The IBGP distributes the network layer information for IP prefixes throughout each VPN and each AS. The following protocols are used for sharing routing information:

- Within an AS, routing information is shared using IBGP.
- Between ASes, routing information is shared using EBGP. EBGP allows service providers to set up an interdomain routing system that guarantees loop-free exchange of routing information between separate ASes.

The primary function of EBGP is to exchange network reachability information between ASes, including information about the list of AS routes. The ASes use EBGP border edge routers to distribute the routes, which includes label-switching information. Each border edge router rewrites the next-hop and MPLS labels.

InterAS configuration supported in this MPLS VPN can include an interprovider VPN, which is MPLS VPNs that include two or more ASes, connected by separate border edge routers. The ASes exchange routes use EBGP, and no IBGP or routing information is exchanged between the ASes.

9.1.2 Exchanging VPN Routing Information

ASes exchange VPN routing information (routes and labels) to establish connections. To control connections between ASes, the PE routers and EBGP border edge routers maintain a label forwarding information base (LFIB). The LFIB manages the labels and routes that the PE routers and EBGP border edge routers receive during the exchange of VPN information.

The ASes use the following guidelines to exchange VPN routing information:

- Routing information includes:
- The destination network.
- The next-hop field associated with the distributing router.
- A local MPLS label
- A route distinguisher (RD1) is part of a destination network address. It makes the VPN IP route globally unique in the VPN service provider environment.

The ASBRs are configured to change the next-hop when sending VPN NLRIs to the IBGP neighbors. Therefore, the ASBRs must allocate a new label when they forward the NLRI to the IBGP neighbors.

9.2 InterAS Options

Inspur CN12908 platform switches support the following InterAS options:

- **InterAS option A -** In an interAS option A network, autonomous system border router (ASBR) peers are connected by multiple subinterfaces with at least one interface VPN that spans the two ASes. These ASBRs associate each subinterface with a VPN routing and forwarding (VRF) instance and a BGP session to signal unlabeled IP prefixes. As a result, traffic between the back-to-back VRFs is IP. In this scenario, the VPNs are isolated from each other and, because the traffic is IP Quality of Service (QoS) mechanisms that operate on the IP traffic can be maintained. The downside of this configuration is that one BGP session is required for each subinterface (and at least one subinterface is required for each VPN), which causes scalability concerns as the network grows.
- **InterAS option B -** In an interAS option B network, ASBR ports are connected by one or more subinterfaces that are enabled to receive MPLS traffic. A Multiprotocol Border Gateway Router (MP-BGP) session distributes labeled VPN prefixes between the ASBRs. As a result, the traffic that flows between the ASBRs is labeled. The downside of this configuration is that, because the traffic is MPLS, QoS mechanisms that are applied only to IP traffic cannot be carried and the VRFs cannot be isolated. InterAS option B provides better scalability than option A because it requires only one BGP session to exchange all VPN prefixes between the ASBRs. Also, this feature provides nonstop forwarding (NSF) and Graceful Restart. The ASBRs must be directly connected in this option.

Some functions of option B are noted below:

- You can have an IBGP VPNv4/v6 session between Inspur CN12908 switches within an AS and you can have an EBGP VPNv4/v6 session between data center edge routers and WAN routers.
- There is no requirement for a per VRF IBGP session between data center edge routers, like in the lite version.
- – LDP distributes IGP labels between ASBRs.
- **InterAS option B (with BGP-3107 or RFC 3107 implementation)**
- You can have an IBGP VPNv4/v6 implementation between Inspur CN12908 switches within an AS and you can have an EBGP VPNv4/v6 session between data center edge routers and WAN routers.
- BGP-3107 enables BGP packets to carry label information without using LDP between ASBRs.
- The label mapping information for a particular route is piggybacked in the same BGP update message that is used to distribute the route itself.
- When BGP is used to distribute a particular route, it also distributes an MPLS label which is mapped to that route. Many ISPs prefer this method of configuration since it ensures complete IGP isolation between the data centers.
- **InterAS option B lite** –Details are noted in the Configuring InterAS Option B (lite version) section.

9.3 Licensing Requirements for InterAS Option B

The following table shows the licensing requirements for this feature:

Table 8： Licensing Requirements

Product	License Requirement
Inspur INOS-CN	MPLS Layer 3 requires an Advantage license package: • CN129-AD-M4 for CN12904 • CN129-AD-M8 for CN12908

•

9.4 Guidelines and Limitations for Configuring InterAS Option B

The InterAS option B feature is not supported with BGP confederation AS. The Option B implementation is supported on Inspur CN12908 platform switches.

9.5 Configuring the Switch for InterAS Option B

You enable certain features on the switch to run InterAS option B.

Before you begin

Ensure that you are in the correct VDC (or use the switchto vdc command). The install feature-set mpls command is available only in the default VDC, and you must enable it in default VDC.

Configure VRFs on the DC edge switches with following steps:

SUMMARY STEPS

1. **configure terminal**
2. **install feature-set mpls**
3. **feature mpls ldp**
4. **feature mpls l3vpn**
5. **feature bgp**
6. **vrf-context** *vrf-name*
7. **rd** *route-target-ext-community*
8. **address-family {ipv4 | ipv6} unicast**
9. **route-target {import | export}** *route-target-ext-community*
10. **copy running-config startup-config**

DETAILED STEPS

	Command or Action	Purpose

Step 1	**configure terminal**
	Example:
	`switch# configure terminal`
	`switch(config)#`
		Enters global configuration mode.

Step 2	**install feature-set mpls** **Example:** `switch(config)# install feature-set mpls`	Installs the MPLS feature set in the default VDC. **Note** You can only install and enable MPLS in the default VDC. Use the no form of this command to uninstall the MPLS feature set
Step 3	**feature mpls ldp** **Example:** `switch(config)# feature mpls ldp`	Enables the MPLS LDP feature on the device.. **Note** When the MPLS LDP feature is disabled on the device, no LDP commands are available.
Step 4	**feature mpls l3vpn** **Example:**	Enables the MPLS Layer 3 VPN feature.

	Command or Action	Purpose
	`switch(config)# feature mpls l3vpn`	
Step 5	**feature bgp** **Example:** `switch(config)# feature bgp`	Enables the BGP feature.
Step 6	**vrf-context** *vrf-name* **Example:** `switch(config)# vrf context VPN1`	Defines the VPN routing instance by assigning a VRF name and enters VRF configuration mode. The vrf-name argument is any case-sensitive, alphanumeric string up to 32 characters.
Step 7	**rd** *route-target-ext-community* **Example:** `switch(config-vrf)# rd100:1`	Configures the route distinguisher. The route-distinguisher argument adds an 8-byte value to an IPv4 prefix to create a VPN IPv4 prefix.
Step 8	**address-family {ipv4 | ipv6} unicast** **Example:** `switch(config-vrf)# address-family ipv4 unicast`	Specifies the IPv4 or IPv6 address family type and enters address family configuration mode.
Step 9	**route-target {import | export}** *route-target-ext-community* **Example:** `switch(config-vrf-af-ip4)# route-target import 1:1`	Specifies a route-target extended community for a VRF as follows: • The import keyword imports routing information from the target VPN extended community. • The export keyword exports routing information to the target VPN extended community. • The route-target-ext-community argument adds the route-target extended community attributes to the VRF's list of import or export route-target extended communities.
Step 10	**copy running-config startup-config** **Example:** `switch(config-vrf-af-ip4)# copy running-config startup-config`	(Optional) Copies the running configuration to the startup configuration.

9.6 Configuring BGP for InterAS Option B

Configure DC Edge switches with IBGP & EBGP VPNv4/v6 with the following steps:

Before you begin

To configure BGP for InterAS option B, you need to enable this configuration on both the IBGP and EBGP sides. Refer to Figure 1 for reference.

SUMMARY STEPS

1. **configure terminal**
2. **router bgp** *as-number*
3. **neighbor** *ip-address*
4. **remote-as** *as-number*
5. **address-family {vpnv4 | vpnv6} unicast**
6. **send-community {both | extended}**
1
7. **retain route-target all**
8. **vrf** *vrf-name*
9. **address-family {ipv4 | ipv6} unicast**
10. **exit**
11. **copy running-config startup-config**

DETAILED STEPS

	Command or Action	Purpose

Step 1	**configure terminal**
	Example:
	`switch# configure terminal`
	`switch(config)#`
		Enters global configuration mode.

Step 2	**router bgp** *as-number* **Example:** `switch(config)# router bgp 100`	Enters the router BGP configuration mode and assigns an autonomous system (AS) number to the local BGP speaker device.
Step 3	**neighbor** *ip-address* **Example:** `switch(config-router)# neighbor 10.0.0.2`	Adds an entry to the BGP or multiprotocol BGP neighbor table, and enters router BGP neighbor configuration mode.
Step 4	**remote-as** *as-number* **Example:** `switch(config-router-neighbor)# remote-as 200`	The as-number argument specifies the autonomous system to which the neighbor belongs.
Step 5	**address-family {vpnv4 | vpnv6} unicast**	Enters address family configuration mode for configuring

	Command or Action	Purpose

	Example:
	`switch(config-router-neighbor)# address-family`
	`vpnv4 unicast`
		IP VPN sessions.

Step 6	**send-community {both | extended}**
	Example:
	`switch(config-router-neighbor-af)# send-community`
	` both`
		Specifies that a communities attribute should be sent to both BGP neighbors.

Step 7	**retain route-target all**
	Example:
	`switch(config-router-neighbor-af)# retain`
	`route-target all`
		(Optional). Retains VPNv4/v6 address configuration on the ASBR without VRF configuration.
		Note If you have a VRF configuration on the ASBR, this command is not required.

Step 8	**vrf** *vrf-name* **Example:** `switch(config-router-neighbor-af)# vrf VPN1`	Associates the BGP process with a VRF.

Step 9	**address-family {ipv4 | ipv6} unicast**
	Example:
	`switch(config-router-vrf)# address-family ipv4`
	`unicast`
		Specifies the IPv4 or IPv6 address family and enters address family configuration mode.

Step 10	**exit** **Example:** `switch(config-vrf-af)# exit`	Exits IPv4 address family.

Step 11	**copy running-config startup-config**
	Example:
	`switch(config-router-vrf)# copy running-config`
	`startup-config`
		(Optional) Copies the running configuration to the startup configuration.

9.7 Configuring the Switch for InterAS Option B (with RFC 3107 implementation)

You enable certain features on the switch to run InterAS option B.

Before you begin

Configure VRFs on the DC edge switches with following steps:

SUMMARY STEPS

1. **configure terminal**
2. **install feature-set mpls**
3. **feature mpls ldp**
4. **feature mpls l3vpn**
5. **feature bgp**
6. **vrf-context** *vrf-name*
7. **rd** *route-distinguisher*
8. **address-family {ipv4 | ipv6} unicast**
9. **route-target {import | export}** *route-target-ext-community*
10. **copy running-config startup-config**

DETAILED STEPS

	Command or Action	Purpose

Step 1	**configure terminal**
	Example:
	`switch# configure terminal`
	`switch(config)#`
		Enters global configuration mode.

Step 2	**install feature-set mpls** **Example:** `switch(config)# install feature-set mpls`	Installs the MPLS feature set in the default VDC. **Note** You can only install and enable MPLS in the default VDC. Use the no form of this command to uninstall the MPLS feature set
Step 3	**feature mpls ldp** **Example:** `switch(config)# feature mpls ldp`	Enables the MPLS LDP feature on the device.. **Note** When the MPLS LDP feature is disabled on the device, no LDP commands are available.
Step 4	**feature mpls l3vpn** **Example:** `switch(config)# feature mpls l3vpn`	Enables the MPLS Layer 3 VPN feature.
Step 5	**feature bgp** **Example:** `switch(config)# feature bgp`	Enables the BGP feature.
Step 6	**vrf-context** *vrf-name* **Example:** `switch(config)# vrf context VPN1`	Defines the VPN routing instance by assigning a VRF name and enters VRF configuration mode. The vrf-name argument is any case-sensitive, alphanumeric string up to 32 characters.

	Command or Action	Purpose
Step 7	**rd** *route-distinguisher* **Example:** `switch(config-vrf)# rd100:1`	Configures the route distinguisher. The route-distinguisher argument adds an 8-byte value to an IPv4 prefix to create a VPN IPv4 prefix.
Step 8	**address-family {ipv4 | ipv6} unicast** **Example:** `switch(config-vrf)# address-family ipv4 unicast`	Specifies the IPv4 or IPv6 address family type and enters address family configuration mode.
Step 9	**route-target {import | export}** *route-target-ext-community* **Example:** `switch(config-vrf-af-ip4)# route-target import 1:1`	Specifies a route-target extended community for a VRF as follows: • The import keyword imports routing information from the target VPN extended community. • The export keyword exports routing information to the target VPN extended community. • The route-target-ext-community argument adds the route-target extended community attributes to the VRF's list of import or export route-target extended communities.
Step 10	**copy running-config startup-config** **Example:** **Command or Action** `switch(config-vrf-af-ip4)# copy running-config startup-config`	(Optional) Copies the running configuration to the startup configuration. **Purpose**

9.8 Configuring BGP for InterAS Option B (with RFC 3107 implementation)

Configure DC Edge switches with IBGP & EBGP VPNv4/v6 along with BGP labeled unicast family with following steps:

SUMMARY STEPS

1. **configure terminal**
2. **router bgp** *as-number*
3. **address-family {vpnv4 | vpnv6} unicast**
4. **redistribute direct route-map** *tag*

5. **allocate-label all**
6. **exit**
7. **neighbor** *ip-address*
8. **remote-as** *as-number*
9. **address-family {ipv4|ipv6} labeled-unicast**
10. **retain route-target all**
11. **exit**
12. **neighbor** *ip-address*
13. **remote-as** *as-number*
14. **address-family {vpnv4|vpnv6} unicast**
15. **exit**
16. **address-family {vpnv4|vpnv6} unicast**
17. **Repeat the process with ASBR2**
18. **copy running-config startup-config**

DETAILED STEPS

	Command or Action	Purpose

Step 1	**configure terminal**
	Example:
	`switch# configure terminal`
	`switch(config)#`
		Enters global configuration mode.

Step 2	**router bgp** *as-number* **Example:** `switch(config)# router bgp 100`	Enters the router BGP configuration mode and assigns an autonomous system (AS) number to the local BGP speaker device.

Step 3	**address-family {vpnv4 | vpnv6} unicast**
	Example:
	`switch(config-router-neighbor)# address-family`
	`vpnv4 unicast`
		Enters address family configuration mode for configuring
		IP VPN sessions.

Step 4	**redistribute direct route-map** *tag*
	Example:
	`switch(config-router-af)# redistribute direct`
	`route-map loopback`
		Redistributes directly connected routes using the Border
		Gateway Protocol.

Step 5	**allocate-label all** **Example:** `switch(config-router-af)# allocate-label all`	Configures ASBRs with the BGP labeled unicast address family to advertise labels for the connected interface.
Step 6	**exit**	Exits address family router configuration mode and

	Command or Action	Purpose
	Example: `switch(config-router-af)# exit`	enters router BGP configuration mode.
Step 7	**neighbor** *ip-address* **Example:** `switch(config-router)# neighbor 10.1.1.1`	Configures the BGP neighbour's IP address, and enters router BGP neighbour configuration mode.
Step 8	**remote-as** *as-number* **Example:** `switch(config-router-neighbor)# remote-as 100`	Specifies the BGP neighbour's AS number.

Step 9	**address-family {ipv4|ipv6} labeled-unicast**
	Example:
	`switch(config-router-neighbor)# address-family`
	`ipv4 labeled-unicast`
		Configures the ASBR with the BGP labeled unicast address
		family to advertise labels for the connected interface.
		Note This is the command that implements RFC
		3107.

Step 10	**retain route-target all**
	Example:
	`switch(config-router-neighbor-af)# retain`
	`route-target all`
		(Optional). Retains VPNv4/v6 address configuration on
		the ASBR without VRF configuration.
		Note If you have a VRF configuration on the ASBR,
		this command is not required.

Step 11	**exit** **Example:** `Switch(config-router-neighbor-af)# exit`	Exits router BGP neighbor address family configuration mode and returns to router BGP configuration mode.
Step 12	**neighbor** *ip-address* **Example:** `switch(config-router)# neighbor 10.1.1.1`	Configures a loopback IP address, and enters router BGP neighbor configuration mode..
Step 13	**remote-as** *as-number* **Example:** `switch(config-router-neighbor)# remote-as 100`	Specifies the BGP neighbor's AS number.
Step 14	**address-family {vpnv4|vpnv6} unicast** **Example:**	Configures the ASBR with the BGP VPNv4 unicast address family.

	Command or Action	Purpose
	`switch(config-router-vrf)# address-family ipv4 unicast`	
Step 15	**exit** **Example:** `switch(config-vrf-af)# exit`	Exits IPv4 address family.
Step 16	**address-family {vpnv4|vpnv6} unicast** **Example:** `switch(config-router-vrf)# address-family ipv4 unicast`	Configures the ASBR with the BGP VPNv4 unicast address family.
Step 17	**Repeat the process with ASBR2**	Configures ASBR2 with option B (RFC 3107) settings and implements complete IGP isolation between the two data centers DC1 and DC2.
Step 18	**copy running-config startup-config** **Example:** `switch(config-router-vrf)# copy running-config startup-config`	(Optional) Copies the running configuration to the startup configuration.

CHAPTER 10 IETF RFCs Supported for Label Switching

This appendix lists the IETF RFCs supported for label switching on the device.

10.1 IETF RFCs Supported for Label Switching

This table lists the IETF RFCs supported for label switching on the device.

RFCs	Title
RFC 3107	*Carrying Label Information in BGP-4*
RFC 7752	*North-Bound Distribution of Link-State and Traffic Engineering (TE) Information Using BGP*
Draft-ietf-idr-bgpls-segment-routing-epe-05	*Segment Routing BGP Egress Peer Engineering BGP-LS Extensions draft-ietf-idr-bgpls-segment-routing-epe-05*