Configuring Additional BGP EVPN Features

This module discusses some features of BGP EVPN such as Dynamic BGP Peering, Route Map support, and so on.

- Dynamic BGP Peering for EVPN, on page 1
- EVPN Route Map Support, on page 2
- BGP EVPN VRF Auto Route Distinguisher, on page 5
- BGP EVPN VRF Auto Route Target, on page 6

Dynamic BGP Peering for EVPN

Dynamic BGP Peering feature allows BGP peering to a group of remote neighbors that are defined by a range of IP addresses. Each range can be configured as a subnet IP address. BGP dynamic neighbors are configured using a range of IP addresses and BGP peer groups.

For information about configuring Dynamic BGP Peering, see *BGP Dynamic Neighbors*.

In Cisco IOS XE Release 17.11.1, support for Dynamic BGP Peering is extended to the L2VPN EVPN address family. To understand how to configure, verify, and troubleshoot BGP Dynamic Peering for the EVPN address family, refer to *Configure BGP DN for Multiple AF on Catalyst 9000 Series Switches*.

Configuration Example for Dynamic BGP Peering

Here is a sample configuration for dynamic BGP peering for L2VPN EVPN Family:

```
router bgp 10
 bgp log-neighbor-changes
 bgp listen range 10.10.10.0/24 peer-group DN-GRP
 bgp listen limit 2
 neighbor DN-GRP peer-group
 neighbor DN-GRP remote-as 12 alternate-as 13
 !
 address-family l2vpn evpn
  neighbor DN-GRP activate
  neighbor DN-GRP send-community extended
 exit-address-family
 !
```

EVPN Route Map Support

The Leaf, Spine and Border nodes of a BGP EVPN fabric support route map for L2VPN EVPN address-family. A route map allows you to check for certain match conditions and set a value. With the route map support, the BGP attributes and their values can be modified to customize the routing policy based on the requirement. The routing policy can be applied for both inbound and outbound EVPN routes.

EVPN routes contain many fields and carry attributes specific to EVPN. Using route maps, you can filter routes on the basis of these attributes. The following route-filtering and attribute-handling options are available for the routes belonging to the EVPN address family:

- Matching based on IP prefix:

  You can use IPv4 or IPv6 prefix or host route that is embedded in EVPN prefix field to filter EVPN Type 2 and Type 5 routes.

- Matching based on the EVPN route type:

  Seven types of network layer reachability information (NLRI) are available in EVPN and called the Route Types. You can filter the routes based on the EVPN route type. For Type 2, you can further filter the routes based on the MAC address or the MAC and IP addresses.

- Setting a value:

  A set clause is used in a route map to set the BGP attributes, community, and so on. In a route map operation, after all the match statements are successful, the set clauses are executed.

Configuring Route Maps

You can configure route maps to filter the EVPN address-family routes. A route map evaluates match clauses until a successful match occurs. After the match occurs, the route map stops evaluating match clauses and starts executing set clauses, in the order in which they were configured. If a successful match does not occur, the route map "falls through" and evaluates the next sequence number of the route map until all configured route map entries have been evaluated or a successful match occurs.

Matching based on IP prefix

Use this procedure to filter the BGP EVPN routes based on the value of the EVPN prefix field. You can use the IPv4 or IPv6 prefix or host route that is embedded in the EVPN prefix field to filter the EVPN Type 2, Type 3, and Type 5 routes.

Procedure

	Command or Action	Purpose
Step 1	**configure terminal** **Example:** `Device# configure terminal`	Enters global configuration mode.
Step 2	**route-map** *route-map-name* **Example:**	Creates a route map and enters route-map configuration mode.

	Command or Action	Purpose
	`Device(config)# route-map rmap1`	
Step 3	**match ip address prefix-list** *prefix-list-name* **Example:** `Device(config-route-map)# match ip address prefix-list plist1`	Matches the BGP EVPN routes with the specified prefix-list.
Step 4	**end** **Example:** `Device(config-router-af)# end`	Exits route-map configuration mode and enters privileged EXEC mode.

Example

The following snippet is a sample configuration that matches an EVPN route based on IP prefix.

```
ip prefix-list plist1 permit 172.16.1.0/24
route-map rmap1
  match ip address prefix-list plist1
  set local-preference 222
```

Matching Based on EVPN Route Type

Procedure

	Command or Action	Purpose
Step 1	**configure terminal** **Example:** `Device# configure terminal`	Enters global configuration mode.
Step 2	**route-map** *route-map-name* **Example:** `Device(config)# route-map rmap1`	Creates a route map and enters route-map configuration mode.
Step 3	**match evpn route-type** { **1** | **2** | **2-mac-only** | **2-mac-ip** | **3** | **4** | **5** | **6** } **Example:** `Device(config-route-map)# match evpn route-type 6`	Matches the BGP EVPN routes with the specified route type.
Step 4	**end** **Example:** `Device(config-router-af)# end`	Exits route-map configuration mode and enters privileged EXEC mode.

Example

The following snippet is a sample configuration that matches an EVPN route based on the route type 2.

```
route-map rmap1 permit 10
 match evpn route-type 2
 set local-preference 222
```

Applying Route Maps

You can apply the route maps to filter the EVPN routes that are advertised to or received from a BGP peer and to set BGP attributes to specific routes.

After you've configured the route map with match and set clauses based on your requirements, use the following procedure to apply the route map at the inbound or outbound level.

Procedure

	Command or Action	Purpose
Step 1	**configure terminal** **Example:** `Device# configure terminal`	Enters global configuration mode.
Step 2	**router bgp** *autonomous-system* **Example:** `Device(config)# router bgp 100`	Enables a BGP routing process, assigns it an AS number, and enters router configuration mode. The AS number can be from 1 to 65535, with 64512 to 65535 designated as private autonomous numbers.
Step 3	**neighbor** {*ip-address* | *peer-group-name* } **remote-as** *number* **Example:** `Device(config-router)# neighbor 10.1.1.2 remote-as 200`	Adds an entry to the BGP neighbor table specifying that the neighbor that is identified by the IP address belongs to the specified AS.
Step 4	**address-family l2vpn evpn** **Example:** `Device(config-router)# address-family l2vpn evpn`	Specifies the L2VPN address family and enters address family configuration mode.
Step 5	**neighbor** *ip-address* **activate** **Example:** `Device(config-router-af)# neighbor 10.1.1.2 activate`	Enables the exchange of information with a BGP neighbor.

	Command or Action	Purpose
Step 6	**neighbor** *ip-address* **route-map** *route-map-name* {**in** | **out**} **Example:** `Device(config-router-af)# neighbor 10.1.1.2 route-map rmap1 in`	Applies a route map to incoming or outgoing routes.
Step 7	**end** **Example:** `Device(config-router-af)# end`	Exits address-family configuration mode and enters privileged EXEC mode.

Example

The following example shows how to configure a route map as an inbound neighbor policy.

```
ip prefix-list plist1 permit 10.1.1.0/24
 route-map rmap1 permit 10
 match ip address prefix-list plist1
 set community comm1

 router bgp 100
 neighbor 10.1.1.2 remote-as 200
 address-family l2vpn evpn
  neighbor 10.1.1.2 activate
  neighbor 10.1.1.2 route-map rmap1 in
!
```

BGP EVPN VRF Auto Route Distinguisher

You can now configure devices in a BGP EVPN VXLAN fabric to autogenerate the route distinguisher (RD) value. Autogenerated RD value is based on the Type 1 encoding format as described in IETF RFC 4364.

Prerequisites for Configuring Auto Route Distinguisher in a BGP EVPN VXLAN Fabric

You must explicitly configure the BGP router ID at the global level, using the **bgp router-id** command.

RD is autogenerated using four bytes of BGP router ID and two bytes of VRF ID.

Configuring Auto Route Distinguisher in a BGP EVPN VXLAN Fabric

To enable autogeneration of RD for a specific VRF, use the **rd-auto** command in the VRF configuration mode.

Example

```
VTEP(config)# vrf definition green
VTEP(config-vrf)# rd-auto
```

To enable autogeneration of RDs at a global level for all the VRFs in the fabric, use the **vrf rd-auto** command in the global configuration mode.

Example

```
VTEP(config)# vrf rd-auto
```

To override the global auto RD configuration and configure a static RD for a specific VRF, use the **rd-auto disable** command in the VRF configuration mode. This command disables the autogeneration of RD for this specific VRF and lets you configure a static RD value.

Example

```
VTEP(config)# vrf definition green
VTEP(config-vrf)# rd-auto disable
VTEP(config-vrf)# rd 103:2
```

You can disable the manual configuration of RD using the **no** form of the command.

The following example disables the static configuration of RD and enables autogeneration of RD.

Example

```
VTEP(config)# vrf definition green
VTEP(config-vrf)# no rd 103:2
VTEP(config-vrf)# no rd-auto disable
```

Note You cannot enable both auto RD and static RD for a given VRF.

In Cisco IOS XE Dublin 17.12.1, the **show vrf** command is enhanced to indicate if an RD has been assigned automatically.

Example

```
VTEP# show vrf
Name              Default RD            Protocols  Interfaces
green             1.1.1.1:60000(auto)   ipv4,ipv6   Gi1/0
blue              2:2                   ipv4,ipv6   Lo12
```

BGP EVPN VRF Auto Route Target

A VRF can be associated with one or more route targets (RT). You can enable the devices in a BGP EVPN VXLAN fabric to autogenerate the route target value for a specified VRF. The autogenerated route target value is based on the Type 0 encoding format as described in IETF RFC 4364.

Note Both statically assigned RT and autogenerated RT can exist in a VRF.

Prerequisites for Configuring Auto Route Target in a BGP EVPN Fabric

You must explicitly configure the BGP router-id at the global level, using the **bgp router-id** command in the router configuration mode.

A route target value is constructed using Autonomous System Number (ASN) as the 2-byte administrative field and the Virtual Network Identifier (VNID) for the 4-byte numbering field. If the ASN is 4 bytes, it is aliased with a special purpose 2-byte ASN, 23456. This 2-byte ASN 23456 is registered by the IANA as AS_TRANS.

Configuring Auto Route Target in a BGP EVPN VXLAN Fabric

To enable autogeneration of route target for a VRF, use the following command in the VRF configuration mode.

vnid *vnid* [**evpn-instance** [**vni** {*vni-id* | **auto**} **core-vlan** *vlan-id*]]

The following table describes the different fields of the command syntax.

vnid	Specifies the Virtual Network Identifier (VNID). The VNID range is from 1 to 2147483647. This number is primarily used to auto generate the route targets.
evpn-instance	(Optional) Specify the **evpn-instance** keyword to autogenerate the stitching of Route Targets.
vni *vni-id*	(Optional) Specifies the NVE Layer 3 VNI number. The range is from 4096 to 16777215.
vni auto	(Optional) Auto generates the NVE Layer 3 VNI number using the *vnid* value.
core-vlan *vlan-id*	(Optional) Auto generates the core VLAN configuration using *vnid* and *vlan-id*. *vlan-id* ranges from 1 to 4094. It also auto generates the core SVI which uses the IP address of the source-interface of the NVE interface. **Note** • If you use the **vni** option, it is mandatory to provide the **core-vlan** option too. • When the core SVI is autogenerated, if the NVE source interface is not configured with an IP address,

	the corresponding Layer 3 traffic does not flow because of a lack of address on the core SVI.

Example

```
VTEP(config)# router bgp 150
VTEP(config-bgp)# bgp router-id interface Loopback0
VTEP(config-bgp)# exit

VTEP(config)# vrf definition blue
VTEP(config-vrf)# vnid 123 evpn-instance vni 35000 core-vlan 345
VTEP(config-vrf)# exit

VTEP(config)# address-family ipv4
VTEP(config-af-ipv4)# exit-address-family
VTEP(config)# address-family ipv6
VTEP(config-af-ipv6)# exit-address-family
```

The **show vrf detail** command is enhanced to indicate if an RT has been assigned automatically.

Example

```
VTEP# show vrf detail blue
VRF blue (VRF Id = 2); default RD 192.168.1.1:1(auto); default VPNID <not set>
  New CLI format, supports multiple address-families
  vnid: 123 evpn-instance vni 35000 core-vlan 345
  Flags: 0x180C
  Interfaces:
    Vl345                   Lo101                   Et1/1
    Vl4                     Vl15
Address family ipv4 unicast (Table ID = 0x2):
  Flags: 0x0
  Export VPN route-target communities
    RT:100:123              RT:65000:123 (auto)
  Import VPN route-target communities
    RT:100:123              RT:65000:123 (auto)
  Export VPN route-target stitching communities
    RT:65000:123 (auto)
  Import VPN route-target stitching communities
    RT:65000:123 (auto)
  No import route-map
  No global export route-map
  No export route-map
  VRF label distribution protocol: not configured
  VRF label allocation mode: per-prefix
Address family ipv6 unicast not active
Address family ipv4 multicast not active
Address family ipv6 multicast not active

VTEP1#
```

Use the **show derived-config** command to check the acquired configurations for core VLAN, core SVI and the NVE interface.

```
VTEP1# show derived-config interface Vlan345

Building configuration...
```

```
Derived configuration : 66 bytes
!
interface Vlan345
 vrf forwarding blue
 ip unnumbered Loopback1
 ipv6 unnumbered Loopback1
 no autostate
end

VTEP1# show derived-config interface nve1

Building configuration...
Derived configuration : 296 bytes
!
interface nve1
 source-interface Loopback1
 host-reachability protocol bgp
 member vni 35555 vrf black
 member vni 35000 vrf blue
 member vni 20011 ingress-replication
 member vni 20012 ingress-replication
 member vni 30000 vrf red
 member vni 30002 vrf green
 member vni 30004 vrf yellow
end
```