Implementing DCI VXLAN Layer 3 Gateway

This chapter module provides conceptual and configuration information for Data Center Interconnect (DCI) VXLAN Layer 3 Gateway on Cisco ASR 9000 Series Router.

Release	Modification
Release 5.3.2	This feature was introduced.
Release 6.1.x	• OpFlex

Prerequisites for Implementing Data Center Interconnect Layer 3 Gateway

- You must be in a user group associated with a task group that includes the proper task IDs. The command reference guides include the task IDs required for each command. If you suspect user group assignment is preventing you from using a command, contact your AAA administrator for assistance.
- You need to have understanding of the following features:
  - VxLAN: For detailed conceptual and configuration information, see the chapters *Implementing Layer 2 VxLAN Gateway* and *Implementing Layer 3 VxLAN Gateway* in *Cisco ASR 9000 Series Aggregation Services Router L2VPN and Ethernet Services Configuration Guide* and *Cisco ASR 9000 Series Aggregation Services Router MPLS Layer 3 VPN Configuration Guide*.

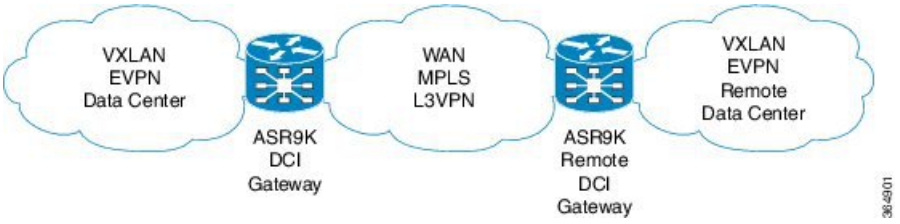
- MP-BGP: For detailed conceptual and configuration information, see the chapter *Implementing BGP* in the *Cisco ASR 9000 Series Aggregation Services Router Routing Configuration Guide*.
- MPLS L3VPN: For detailed conceptual and configuration information, see the *Cisco ASR 9000 Series Aggregation Services Router MPLS Layer 3 VPN Configuration Guide*.

Data Center Interconnect VXLAN Layer 3 Gateway

The Cisco ASR 9000 Series Router can serve as a Data Center Interconnect (DCI) L3 Gateway to provide IP connectivity between multi-tenant remote Data Center sites. Consider the following network topology, which has two Data Center sites connected through the intermediate Service Provider network. The multi-tenant Data Centers use VXLAN encapsulation to carry separate tenant IP traffic. The VXLAN-enabled Data Center sites use MP-BGP EVPN control plane for distributing both Layer-2 and Layer-3 forwarding information within the site. The RFC 5512 and *draft-ietf-bess-evpn-inter-subnet-forwarding-00* define how MP-BGP Network Layer Reachability Information (NLRI) carries VXLAN encapsulation as well as L2/L3 forwarding information details to provide an integrated routing and bridging solution within the Data Center site. The Cisco ASR 9000 Series Routeruses MPLS L3VPN application service over the Service Provider network to provide L3 connectivity between the two Data Center sites.

Note DCI Gateway does not provide layer 2 inter-connectivity across Data Centers.

Figure 1: Data Center Interconnect Layer 3 Gateway



The Cisco ASR 9000 Series Router functions as a Data Center Interconnect (DCI) gateway by intermediating between the two MP-BGP control planes, one on the Data Center site and the other on the MPLS L3VPN network. To enable this exchange of forwarding information between the two MP-BGP control planes, the DCI router has a VRF instance configured with two sets of import and export route-targets. One set of import/export route targets is associated with the Data Center BGP neighbor router that uses MP-BGP EVPN with route type 5 NLRI to exchange VXLAN encapsulation and L3 routing information with the DCI router. The other set of import/export route-targets is associated with the L3VPN BGP neighbor in the service provider network that uses VPNv4 or VPNv6 address-family to exchange L3 and MPLS information. The DCI router exchanges the IP prefixes in VRF instance as L3VPN NLRIs with the L3VPN BGP neighbor and as EVPN NLRIs with the EVPN BGP neighbor and thus, effectively stitches these two sets of route targets. This enables the DCI router to convert the received Data Center EVPN forwarding information into VPNv4 or VPNv6 routes that, in turn, is to be forwarded to the remote DCI router and vice versa. The remote DCI router connected to the remote Data Center performs same functions. This enables L3 connectivity between two hosts located across remote Data Center sites. The DCI Gateway enables tenant Layer 3 data traffic movement across remote Data Centers by stitching the per-tenant VXLAN encapsulation in the DCI Gateway router to the per-tenant MPLS encapsulation in L3VPN service provider network.

The DCI L3 Gateway can be configured on the Provider Edge (PE) router or a Data Center router that connects to the WAN. The WAN network can be any VRF-deployed network configured with any control plane protocol. For example, a VRF-lite WAN network configured with an IGP.

Note In a DCI deployment, for route reoriginate with stitching-rt for a particular VRF, using the same Route Distinguisher (RD) between ASR9K DCI and MPLS-VPN PE or same RD between ASR9K DCI and VxLAN Top of Rack (ToR) is not supported.

Route Targets

For each VRF on the DCI router, there are two sets of manually configured import and export route-targets. One set of import and export route-targets is associated with the Data Center BGP neighbor that uses EVPN address-family to exchange L3 information; the other set of import and export route-targets is associated with the L3VPN BGP neighbor that use VPNv4 or VPNv6 unicast address-family to exchange L3 information. This separation of route targets (RTs) enables the two sets of RTs to be independently configured. The DCI router effectively stitches the two set of RTs. The RTs associated with the EVPN BGP neighbor are labelled as stitching RTs. The RTs associated with the L3VPN BGP neighbor are normal RTs.

Route Re-origination

Consider the case of control plane information propagation by the DCI from the L3VPN side to the Data Center side. Here, instead of advertising the remote Data Center's original BGP EVPN routes, you can configure the DCI router to advertise to its BGP EVPN neighbor the routes that are re-originated after importing them from the L3VPN BGP neighbor. For this case of VPNv4 or VPNv6 routes being propagated to the BGP EVPN neighbors (Data Center neighbors), re-originating the routes refers to replacing the normal route-targets with the local route-target values associated with the BGP EVPN neighbors . The converse holds true for the routing information traffic propagation from the BGP EVPN control plane to BGP L3VPN control plane. You can configure this re-origination by using the re-originate keyword in the **import re-originate** command.. Configuring this command, by default, also enables advertisement of L2VPN EVPN prefixes to the EVPN BGP neighbors. You can suppress native L2VPN EVPN address-family NLRI advertisements towards the EVPN Neighbor using the **advertise l2vpn evpn disable** command under the EVPN BGP address-family configuration mode.

Route Address-Family and Encoded Address-Family

When an address-family is configured for a BGP neighbor, it means that the specified address-family routes encoded with the NLRI for that address-family is advertised to the neighbor. This does not hold for data center BGP neighbors because they use only EVPN address-family. Here, BGP neighbors advertise VPNv4 or VPNv6 unicast routes using the EVPN NLRI encoding. Thus, here the encoded address-family and route address family can be possibly different. You can advertise the VPNv4 or VPNv6 address-family using the **advertise vpnv4 unicast** or **advertise vpnv6 unicast** command. For example, a EVPN address-family BGP neighbor configured with the **advertise vpnv4 unicast** command sends VPNv4 unicast routes in an EVPN encoded NLRI.

Local VPNv4 or VPNv6 Routes Advertisement

On the DCI router, the locally sourced VPNv4 or VPNv6 routes can be advertised to the BGP EVPN neighbors with the normal route targets (RTs) configured for the VRF or the stitching RTs associated with the BGP EVPN neighbors. By default, these routes are advertised with the normal route targets. You can configure these local VPNv4 or VPNv6 route advertisements to be advertised with stitching RTs to the BGP EVPN neighbors by using the **advertise vpnv4 unicast local stitching-rt** or **advertise vpnv6 unicast local stitching-rt** command as required.

Data Center VXLAN with Support for MP-BGP

The Data Center VXLAN uses MP-BGP for control-plane learning of end-host Layer 2 and Layer 3 reachability information. The DCI router is configured with a VXLAN Tunnel EndPoint (VTEP). For VTEP configuration details, see the chapter *Implementing Layer 3 VXLAN Gateway*. You also need to run the **host-reachabilty protocol bgp** command to specify that control-plane learning within Data center site is through BGP routing protocol.

The DCI Gateway router and the EVPN BGP neighbor (Data Center BGP neighbor) exchange BGP EVPN NLRIs of route type 5 that carry L3 routing information and associated VXLAN encapsulation information. Some of the VXLAN information is carried in the EVPN NLRI and the rest is carried in RFC 5512 Tunnel Type Encapsulation EXTCOMM and Router MAC EXTCOMM defined in *draft-ietf-bess-evpn-inter-subnet-forwarding-00*. BGP downloads VXLAN encapsulation as *RIB remote next hop opaque* attribute to L3RIB.

Default-Originate Forwarding to BGP EVPN Neighbor

Instead of advertising the specific networks available in the remote Data Center, you can configure the DCI gateway to advertise a default route to the directly connected Data Center neighbor. To send the default route for a VRF instance to the Data Center BGP EVPN neighbor, the VPN default-originate information that is typically forwarded to the L3VPN BGP neighbor, is also configured to be forwarded to the BGP EVPN neighbor in the Data Center. To do so, you need to configure **allow vpn default-originate** command in the BGP VRF configuration mode and also configure **default-originate** command under EVPN BGP neighbor in L2VPN EVPN address-family configuration mode. This configures BGP to forward only one default route information for a VRF instance from the DCI Gateway to the BGP neighbor that has L2VPN EVPN address-family. This default route information is encoded in the EVPN "IP Prefix Route" NLRI.

With the advertisement of a default route to the connected Data Center, the DCI Gateway should not advertise specific prefixes of the remote Data Center to the BGP EVPN neighbor. To prevent forwarding of VRF prefixes, you need to configure the DCI gateway with a EVPN BGP neighbor policy that drops forwarding of all prefixes.

Configure Data Center Interconnect Router

This section describes tasks to configure the Data Center Interconnect (DCI) router. Perform the following tasks to complete the configuration:

Configure VRF and route targets import/export rules

Perform the following steps to configure VRF and define route targets to be used for import and export of forwarding information.

SUMMARY STEPS

1. **configure**
2. **vrf** *vrf-name*
3. **address-family** { **ipv4** | **ipv6** } **unicast**
4. **import route-target** *route-target-id*
5. **export route-target** *route-target-id*
6. **import route-target** *route-target-id* **stitching**
7. **export route-target** *route-target-id* **stitching**
8. **commit**

DETAILED STEPS

	Command or Action	Purpose
Step 1	**configure**	
Step 2	**vrf** *vrf-name* **Example:** `RP/0/RSP0/CPU0:router(config)# vrf data-center-10`	Defines a VPN routing and forwarding (VRF) instance and enters VRF configuration mode.
Step 3	**address-family** { **ipv4**	**ipv6** } **unicast** **Example:** `RP/0/RSP0/CPU0:router(config-vrf-af)# address-family ipv4 unicast`	Specifies either the IPv4 or IPv6 address family for the VRF instance and enters address family configuration submode.
Step 4	**import route-target** *route-target-id* **Example:** `RP/0/RSP0/CPU0:router(config-vrf-af)# import route-target 1:1`	Configures importing of routes to the VRF from the L3VPN BGP NLRIs that have the matching route-target value.
Step 5	**export route-target** *route-target-id* **Example:** `RP/0/RSP0/CPU0:router(config-vrf-af)# export route-target 1:2`	Configures exporting of routes from the VRF to the L3VPN BGP NLRIs and assigns the specified route-target identifiers to the L3VPN BGP NLRIs.
Step 6	**import route-target** *route-target-id* **stitching** **Example:** `RP/0/RSP0/CPU0:router(config-vrf-af)# import route-target 10:1 stitching`	Configures importing of routes from the EVPN BGP NLRI that have the matching route-target value.

	Command or Action	Purpose
Step 7	**export route-target** *route-target-id* **stitching** **Example:** `RP/0/RSP0/CPU0:router(config-vrf-af)# export route-target 10:2 stitching`	Configures exporting of routes from the VRF to the EVPN BGP NLRIs and assigns the specified route-target identifiers to the BGP EVPN NLRIs.
Step 8	**commit**	

Configure Bridge Domain for DCI Gateway

Perform the following steps to configure the bridge domain on the DCI Gateway.

SUMMARY STEPS

1. **configure**
2. **interface bvi** *bvi-id*
3. **vrf** *vrf-id*
4. **ipv4 address** *ip-address subnet-mask*
5. **exit**
6. **l2vpn**
7. **bridge group** *bridge-group-identifier*
8. **bridge-domain** *bridge-group-identifier*
9. **routed interface** *interface-identifier*
10. **member vni** *vni-id*
11. **commit**

DETAILED STEPS

	Command or Action	Purpose
Step 1	**configure**	
Step 2	**interface bvi** *bvi-id* **Example:** `RP/0/RSP0/CPU0:router(config)# interface bvi 1`	Creates a BVI (Bridge Virtual Interface) with the user-defined identifier and enters the interface configuration mode.
Step 3	**vrf** *vrf-id* **Example:** `RP/0/RSP0/CPU0:router(config-if)# vrf cust1`	Associates a VRF (Virtual Routing and Forwarding) instance to the BVI.
Step 4	**ipv4 address** *ip-address subnet-mask* **Example:** `RP/0/RSP0/CPU0:router(config-if)# ipv4 address 40.1.1.1 255.255.255.255`	Creates an IPv4 address for the BVI.

	Command or Action	Purpose
Step 5	**exit** **Example:** `RP/0/RSP0/CPU0:router(config)# exit`	Exits the interface configuration mode and returns to Global Configuration mode.
Step 6	**l2vpn** **Example:** `RP/0/RSP0/CPU0:router(config)# l2vpn`	Enters the L2VPN configuration mode.
Step 7	**bridge group** *bridge-group-identifier* **Example:** `RP/0/RSP0/CPU0:router(config-l2vpn)# bridge group bg1`	Configures a bridge group and enters the Bridge Group configuration mode.
Step 8	**bridge-domain** *bridge-group-identifier* **Example:** `RP/0/RSP0/CPU0:router(config-l2vpn-bg)# bridge-domain bd1`	Configures a bridge domain and enters Bridge Domain configuration mode.
Step 9	**routed interface** *interface-identifier* **Example:** `RP/0/RSP0/CPU0:router(config-l2vpn-bg-bd)# routed interface BVI1`	Configures a BVI interface as a routing interface for the bridge domain.
Step 10	**member vni** *vni-id* **Example:** `RP/0/RSP0/CPU0:router(config-l2vpn-bg-bd)# member vni 5001`	Assigns the bridge domain to a VxLAN segment.
Step 11	**commit**	

Configure VTEP

Perform the following steps to configure VTEP (VxLAN Terminal EndPoint) on the DCI Gateway.

SUMMARY STEPS

1. **configure**
2. **interface loopback** *loopback-id*
3. **ipv4 address** *ip-address subnet-mask*
4. **exit**
5. **interface nve** *nve-id*

6. **source interface loopback** *loopback-interface-identifier*
7. **member vni** *vni-id*
8. **vrf** *vrf-id*
9. **host reachability protoocol bgp**
10. **commit**

DETAILED STEPS

	Command or Action	Purpose
Step 1	**configure**	
Step 2	**interface loopback** *loopback-id* **Example:** `RP/0/RSP0/CPU0:router(config)# interface loopback 0`	Creates a loopback interface with the user-defined loopback identifier and enters the interface configuration mode.
Step 3	**ipv4 address** *ip-address subnet-mask* **Example:** `RP/0/RSP0/CPU0:router(config-if)# ipv4 address 40.1.1.1 255.255.255.255`	Creates an IPv4 address for the loopback interface.
Step 4	**exit** **Example:** `RP/0/RSP0/CPU0:router(config)# exit`	Exits the interface configuration mode and returns to Global Configuration mode.
Step 5	**interface nve** *nve-id* **Example:** `RP/0/RSP0/CPU0:router(config)# interface nve 1`	Creates an NVE (Network Virtualization Endpoint) interface with the identifier and enters the interface configuration mode.
Step 6	**source interface loopback** *loopback-interface-identifier* **Example:** `RP/0/RSP0/CPU0:router(config-if)# source interface loopback 0`	Configures the loopback interface IP address as the source IP address for the NVE interface.
Step 7	**member vni** *vni-id* **Example:** `RP/0/RSP0/CPU0:router(config-if)# member vni 5001`	Configures the NVE interface under a VxLAN with VNI (VxLAN Network Identifier) 5001 and enters the NVE VNI configuration mode.
Step 8	**vrf** *vrf-id* **Example:** `RP/0/RSP0/CPU0:router(config-nve-vni)# vrf cust1`	Associates a VRF (Virtual Routing and Forwarding) instance to the VxLAN segment.

	Command or Action	Purpose
Step 9	**host reachability protoocol bgp** **Example:** `RP/0/RSP0/CPU0:router(config-nve-vni)# host reachability protocol bgp`	Specifies BGP protocol as the control protocol for VxLAN VTEP end-host reachability.
Step 10	**commit**	

Configure EVPN BGP neighbor and route advertisements

Perform this task on the DCI router to configure BGP neighbor relationship and route advertisements with the EVPN BGP neighbor.

SUMMARY STEPS

1. **configure**
2. **router bgp** *asn-id*
3. **address-family l2vpn evpn**
4. **exit**
5. **neighbor** *neighbor-ip-address*
6. **remote-as** *remote-as-id*
7. **address-family l2vpn evpn**
8. *(Optional)* **default-originate**
9. **import stitching-rt reoriginate**
10. **advertise** { **vpnv4** | **vpnv6** } **unicast re-originated**
11. **advertise** { **vpnv4** | **vpnv6** } **unicast local stitching-rt**
12. *(Optional)* **advertise l2vpn evpn disable**
13. **commit**

DETAILED STEPS

	Command or Action	Purpose
Step 1	**configure**	
Step 2	**router bgp** *asn-id* **Example:** `RP/0/RSP0/CPU0:router(config)# router bgp 100`	Specifies the BGP AS number and enters the BGP configuration mode, allowing you to configure the BGP routing process.
Step 3	**address-family l2vpn evpn** **Example:** `RP/0/RSP0/CPU0:router(config-bgp)# address-family l2vpn evpn`	Enables EVPN address family globally under BGP routing process and enters EVPN address family configuration submode.

	Command or Action	Purpose
Step 4	**exit** **Example:** `RP/0/RSP0/CPU0:router(config-bgp-af)# exit`	Exits the EVPN address family configuration and returns to the BGP router configuration mode.
Step 5	**neighbor** *neighbor-ip-address* **Example:** `RP/0/RSP0/CPU0:router(config-bgp)# neighbor 1.1.1.1`	Configures the specified neighbor IP address as a BGP peer and enters neighbor configuration mode for BGP routing.
Step 6	**remote-as** *remote-as-id* **Example:** `RP/0/RSP0/CPU0:router(config-bgp-nbr)# remote-as 100`	Specifies the autonomous system identifier of the BGP neighbor.
Step 7	**address-family l2vpn evpn** **Example:** `RP/0/RSP0/CPU0:router(config-bgp-nbr)# address-family l2vpn evpn`	Enables EVPN address family under BGP routing process and enters EVPN address family configuration submode.
Step 8	*(Optional)* **default-originate** **Example:** `RP/0/RSP0/CPU0:router(config-bgp-nbr)# default-originate`	*(Optional)* Configures advertisement of a default route instead of specific prefixes to the BGP EVPN neighbor.
Step 9	**import stitching-rt reoriginate** **Example:** `RP/0/RSP0/CPU0:router(config-bgp-nbr-af)# import stitching-rt reoriginate`	Enables import of routing information from BGP EVPN NLRIs that has route target identifier matching the stitching route target identifier and exports this routing information after re-origination to the L3VPN BGP neighbor.
Step 10	**advertise** { **vpnv4** | **vpnv6** } **unicast re-originated** **Example:** `RP/0/RSP0/CPU0:router(config-bgp-nbr-af)# advertise vpnv4 unicast re-originated`	Configures advertisement of VPNv4 or VPNv6 unicast routes that are redistributed from the L3VPN BGP neighbor, to the EVPN BGP neighbor. The route targets are changed to the stitching route targets before advertising onto the EVPN BGP neighbor.
Step 11	**advertise** { **vpnv4** | **vpnv6** } **unicast local stitching-rt** **Example:** `RP/0/RSP0/CPU0:router(config-bgp-nbr-af)# advertise vpnv4 unicast local stitching-rt`	Configures the local VPNv4 or VPNv6 unicast routes to be advertised with stitching route target identifiers to the EVPN BGP neighbor.

	Command or Action	Purpose
Step 12	*(Optional)* **advertise l2vpn evpn disable** **Example:** `RP/0/RSP0/CPU0:router(config-bgp-nbr-af)# advertise`	Suppresses the advertisement of the L2VPN EVPN routes to the EVPN BGP neighbor. The step 7 **address-family l2vpn evpn** command, by default, causes the L2VPN EVPN routes to be advertised along with the VPNv4 or VPNv6 unicast routes received from the L3VPN BGP neighbor.
Step 13	**commit**	

Configure L3VPN BGP neighbor relationship and route advertisements

Perform the following steps to configure BGP neighbor relationship and route advertisements with the L3VPN BGP neighbor.

SUMMARY STEPS

1. **configure**
2. **router bgp** *asn-id*
3. **address-family** { **vpnv4 | vpnv6** }
4. **exit**
5. **neighbor** *neighbor-ipv4/6-address*
6. **remote-as** *remote-as-id*
7. **address-family** { **vpnv4 | vpnv6** }
8. **import reoriginate stitching-rt**
9. **advertise** { **vpnv4 | vpnv6** } **unicast re-originated**
10. **commit**

DETAILED STEPS

	Command or Action	Purpose
Step 1	**configure**	
Step 2	**router bgp** *asn-id* **Example:** `RP/0/RSP0/CPU0:router(config)# router bgp 100`	Specifies the BGP AS number and enters the BGP configuration mode, allowing you to configure the BGP routing process.
Step 3	**address-family** { **vpnv4	vpnv6** } **Example:** `RP/0/RSP0/CPU0:router(config-bgp-nbr)# address-family vpnv4`	Configures VPNv4 or VPNv6 address family for the global BGP routing process and enters VPNv4 or VPNv6 address family configuration mode.
Step 4	**exit** **Example:** `RP/0/RSP0/CPU0:router(config-bgp-af)# exit`	Exits the VPNv4 or VPNv6 address family configuration and returns to the BGP router configuration mode.

	Command or Action	Purpose
Step 5	**neighbor** *neighbor-ipv4/6-address* **Example:** `RP/0/RSP0/CPU0:router(config-bgp)# neighbor 1.1.1.1`	Configures the specified neighbor IP address as a BGP peer and enters neighbor configuration mode for BGP routing.
Step 6	**remote-as** *remote-as-id* **Example:** `RP/0/RSP0/CPU0:router(config-bgp-nbr)# remote-as 100`	Specifies the autonomous system identifier of the BGP neighbor.
Step 7	**address-family** { **vpnv4 | vpnv6** } **Example:** `RP/0/RSP0/CPU0:router(config-bgp-nbr)# address-family vpnv4`	Configures VPNv4 or VPNv6 address family for the BGP neighbor and enters VPNv4 or VPNv6 address family configuration mode.
Step 8	**import reoriginate stitching-rt** **Example:** `RP/0/RSP0/CPU0:router(config-bgp-nbr-af)# import reoriginate stitching-rt`	Configures import of routing information from the L3VPN BGP NLRIs that has route target identifier matching the normal route target identifier and exports this routing information after re-origination that assigns it with stitching route target identifier, to the BGP EVPN neighbor.
Step 9	**advertise** { **vpnv4 | vpnv6** } **unicast re-originated** **Example:** `RP/0/RSP0/CPU0:router(config-bgp-nbr-af)# advertise vpnv4 unicast re-originated`	Configures advertisement of VPNv4 or VPNv6 unicast routes that are redistributed from the EVPN BGP neighbor, to the L3VPN BGP neighbor. The route targets are changed to the normal route targets before advertising onto the L3VPN BGP neighbor.
Step 10	**commit**	

Verification of Data Center Gateway Configuration

These show commands can be used to verify the DCI Gateway configurations:

SUMMARY STEPS

1. **show bgp l2vpn evpn**
2. **show bgp l2vpn evpn rd** *rd-id l2vpn-evpn-prefix* **detail**
3. **show bgp l2vpn evpn neighbors** *neighbor-ip-address* **detail**
4. **show bgp sessions**
5. **show bgp vpnv4 unicast**
6. **show bgp vpnv4 unicast** *rd-id vpnv4-prefix* **detail**
7. **show bgp vrf foo**
8. **show bgp vrf foo ipv4 unicast 100.1.1.1/32 detail**
9. **show bgp vpnv4 unicast update-group**

10. **show bgp l2vpn evpn update-group**

DETAILED STEPS

	Command or Action	Purpose
Step 1	**show bgp l2vpn evpn** **Example:**	Displays a summary of the BGP information advertised from the EVPN BGP neighbor.

```
RP/0/RSP0/CPU0:router# show bgp l2vpn evpn

Fri Aug 21 00:24:10.773 PDT
BGP router identifier 30.30.30.30, local AS number
 100
BGP generic scan interval 60 secs
Non-stop routing is enabled
BGP table state: Active
Table ID: 0x0   RD version: 0
BGP main routing table version 16
BGP NSR Initial initsync version 1 (Reached)
BGP NSR/ISSU Sync-Group versions 16/0
BGP scan interval 60 secs

Status codes: s suppressed, d damped, h history,
 * valid, > best
              i - internal, r RIB-failure, S
stale, N Nexthop-discard
Origin codes: i - IGP, e - EGP, ? - incomplete
   Network            Next Hop            Metric
 LocPrf Weight Path
Route Distinguisher: 100:1
*>i[2][10000][48][0226.51bd.c81c][32][200::1001]/232

                       11.0.0.1
    100      0 i
*>i[2][10000][48][0226.51bd.c81c][32][200:1::1001]/232

                       11.0.0.1
    100      0 i
*>i[2][10000][48][0226.51bd.c81c][32][200.1.1.1]/136

                       11.0.0.1
    100      0 i
*>i[2][10000][48][0226.51bd.c81c][32][200.1.1.2]/136

                       11.0.0.1
    100      0 i
*>i[5][4231][32][100.1.1.1]/80
                       11.0.0.1
    100      0 i
*>i[5][4231][32][100.1.1.2]/80
                       11.0.0.1
    100      0 i
*>i[5][4231][112][fec0::1001]/176
                       11.0.0.1
    100      0 i
*>i[5][4232][112][fec0::1:1001]/176
                       11.0.0.1
    100      0 i

Processed 8 prefixes, 8 paths
```

	Command or Action	Purpose

Step 2	**show bgp l2vpn evpn rd** *rd-id l2vpn-evpn-prefix* **detail**
	Example:
	`RP/0/RSP0/CPU0:router# show bgp l2vpn evpn rd 100:1 [5][4231][112][fec0::1001]/176 detail`
	`Fri Aug 21 00:34:43.747 PDT`
	`BGP routing table entry for [5][4231][112][fec0::1001]/176, Route Distinguisher: 100:1`
	`Versions:`
	`  Process  bRIB/RIB  SendTblVer`
	`  Speaker  5  5`
	`  Flags: 0x04040001+0x00000000;`
	`Last Modified: Aug 21 00:16:58.000 for 00:17:46`
	`Paths: (1 available, best #1)`
	`  Not advertised to any peer`
	`  Path #1: Received by speaker 0`
	`  Flags: 0x4000600025060005, import: 0x3f`
	`  Not advertised to any peer`
	`  Local`
	`  11.0.0.1 (metric 2) from 20.0.0.1 (11.0.0.1)`
	`  Received Label 16001`
	`  Origin IGP, localpref 100, valid, internal, best, group-best, import-candidate, reoriginate, not-in-vrf`
	`  Received Path ID 0, Local Path ID 1, version 5`
	`  Extended community: Flags 0x2: Encapsulation Type:8 Router MAC:aabb.ccdd.eeff RT:65540:1 RT:40.40.40.40:1 RT:100:1`
	`  Originator: 11.0.0.1, Cluster list: 20.20.20.20`
	`  EVPN ESI: ffff.ffff.ffff.ffff.ff01, Gateway Address : fec0::254`
		Displays detailed information for a specific prefix advertised from an EVPN BGP neighbor.

Step 3	**show bgp l2vpn evpn neighbors** *neighbor-ip-address* **detail**
	Example:
	`RP/0/RSP0/CPU0:router# show bgp l2vpn evpn neighbors 20.0.0.1 detail`
	`Fri Aug 21 00:25:37.383 PDT`
	`BGP neighbor is 20.0.0.1`
	`  Remote AS 100, local AS 100, internal link`
	`  Remote router ID 20.20.20.20`
	`  BGP state = Established, up for 00:08:58`
	`  NSR State: NSR Ready`
	`  Last read 00:00:34, Last read before reset 00:00:00`
	`  Hold time is 180, keepalive interval is 60 seconds`
	`  Configured hold time: 180, keepalive: 60, min acceptable hold time: 3`
	`  Last write 00:00:36, attempted 19, written 19`
		Displays a detailed information of the specified L2VPN EVPN BGP neighbor.

	Command or Action	Purpose

```
  Second last write 00:01:36, attempted 143,
written 143
  Last write before reset 00:00:00, attempted 0,
 written 0
  Second last write before reset 00:00:00,
attempted 0, written 0
  Last write pulse rcvd  Aug 21 00:25:03.667 last
 full not set pulse count 33
  Last write pulse rcvd before reset 00:00:00
  Socket not armed for io, armed for read, armed
 for write
  Last write thread event before reset 00:00:00,
 second last 00:00:00
  Last KA expiry before reset 00:00:00, second
last 00:00:00
  Last KA error before reset 00:00:00, KA not sent
 00:00:00
  Last KA start before reset 00:00:00, second last
 00:00:00
  Precedence: internet
  Non-stop routing is enabled
  Entered Neighbor NSR TCP mode:
    TCP Initial Sync :              Aug 21
00:18:07.291
    TCP Initial Sync Phase Two :    Aug 21
00:18:07.319
    TCP Initial Sync Done :         Aug 21
00:18:08.334
  Multi-protocol capability received
  Neighbor capabilities:            Adv
Rcvd
    Route refresh:                  Yes
Yes
    4-byte AS:                      Yes
Yes
    Address family VPNv4 Unicast:   Yes
No
    Address family VPNv6 Unicast:   Yes
No
    Address family L2VPN EVPN:      Yes
Yes
  Message stats:
    InQ depth: 0, OutQ depth: 0
                    Last_Sent               Sent
  Last_Rcvd               Rcvd
    Open:           Aug 21 00:16:38.087        1
  Aug 21 00:16:40.123        1
    Notification:   ---                        0
  ---                       0
    Update:         Aug 21 00:24:01.421        9
  Aug 21 00:24:03.652       13
    Keepalive:      Aug 21 00:25:01.434        8
  Aug 21 00:25:03.667        9
    Route_Refresh:  Aug 21 00:24:01.377        3
  ---                       0
    Total:                                    21
                           23
  Minimum time between advertisement runs is 0
secs
  Inbound message logging enabled, 3 messages
buffered
  Outbound message logging enabled, 3 messages
```

	Command or Action	Purpose

```
buffered

 For Address Family: VPNv4 Unicast
  BGP neighbor version 35
  Update group: 0.3 Filter-group: 0.1  No Refresh
 request being processed
  Advertise Reorigination Enabled
  Advertise AFI EoR can be sent
  Route refresh request: received 0, sent 0
  0 accepted prefixes, 0 are bestpaths
  Cumulative no. of prefixes denied: 0.
  Prefix advertised 4, suppressed 0, withdrawn 0
  Maximum prefixes allowed 2097152
  Threshold for warning message 75%, restart
interval 0 min
  AIGP is enabled
  An EoR was not received during read-only mode
  Last ack version 35, Last synced ack version 35

  Outstanding version objects: current 0, max 1
  Additional-paths operation: None
  Send Multicast Attributes

 For Address Family: VPNv6 Unicast
  BGP neighbor version 29
  Update group: 0.3 Filter-group: 0.1  No Refresh
 request being processed
  Advertise Reorigination Enabled
  Advertise AFI EoR can be sent
  Route refresh request: received 0, sent 0
  0 accepted prefixes, 0 are bestpaths
  Cumulative no. of prefixes denied: 0.
  Prefix advertised 0, suppressed 0, withdrawn 0
  Maximum prefixes allowed 1048576
  Threshold for warning message 75%, restart
interval 0 min
  AIGP is enabled
  An EoR was not received during read-only mode
  Last ack version 29, Last synced ack version 29

  Outstanding version objects: current 0, max 0
  Additional-paths operation: None
  Send Multicast Attributes
  Advertise VPNv4 routes enabled with
Reoriginate,Local with stitching-RT option

 For Address Family: L2VPN EVPN
  BGP neighbor version 18
  Update group: 0.2 Filter-group: 0.1  No Refresh
 request being processed
  Route refresh request: received 0, sent 3
  8 accepted prefixes, 8 are bestpaths
  Cumulative no. of prefixes denied: 0.
  Prefix advertised 4, suppressed 0, withdrawn 6
  Maximum prefixes allowed 2097152
  Threshold for warning message 75%, restart
interval 0 min
  AIGP is enabled
  An EoR was received during read-only mode
  Last ack version 18, Last synced ack version 18

  Outstanding version objects: current 0, max 2
```

	Command or Action	Purpose
	Additional-paths operation: None Send Multicast Attributes Advertise VPNv4 routes enabled with Reoriginate, option Advertise VPNv6 routes is enabled with Reoriginate, option Import Stitching is enabled for this neighbor address-family Import Reoriginate is enabled for this neighbor address-family Connections established 1; dropped 0 Local host: 30.0.0.1, Local port: 59405, IF Handle: 0x00000000 Foreign host: 20.0.0.1, Foreign port: 179 Last reset 00:00:00	

Step 4	**show bgp sessions**
	Example:
	
	`RP/0/RSP0/CPU0:router# show bgp sessions`
	
	`Fri Aug 21 00:25:57.216 PDT`
	
	`Neighbor VRF Spk AS InQ OutQ NBRState NSRState`
	`20.0.0.1 default 0 100 0 0 Established NSR Ready[PP]`
	`32.0.0.2 default 0 200 0 0 Established NSR Ready`
		Displays current BGP sessions.

Step 5	**show bgp vpnv4 unicast**
	Example:
	
	`RP/0/RSP0/CPU0:router# show bgp vpnv4 unicast`
	
	`Fri Aug 21 00:28:41.253 PDT`
	`BGP router identifier 30.30.30.30, local AS number 100`
	`BGP generic scan interval 60 secs`
	`Non-stop routing is enabled`
	`BGP table state: Active`
	`Table ID: 0x0 RD version: 0`
	`BGP main routing table version 39`
	`BGP NSR Initial initsync version 4 (Reached)`
	`BGP NSR/ISSU Sync-Group versions 39/0`
	`BGP scan interval 60 secs`
	
	`Status codes: s suppressed, d damped, h history, * valid, > best`
	`i - internal, r RIB-failure, S stale, N Nexthop-discard`
	`Origin codes: i - IGP, e - EGP, ? - incomplete`
	`Network Next Hop Metric LocPrf Weight Path`
	`Route Distinguisher: 1:1`
	`*> 1.1.1.0/24 32.0.0.2`
		Displays summary information of unicast BGP prefixes. The **show bgp vpnv6 unicast** provides similar output information for IPv6 prefixes.

	Command or Action	Purpose
	0 200 300 i *> 1.1.2.0/24 32.0.0.2 0 200 300 i Route Distinguisher: 30.30.30.30:0 (default for vrf foo) *> 1.1.1.0/24 32.0.0.2 0 200 300 i *> 1.1.2.0/24 32.0.0.2 0 200 300 i *>i100.1.1.1/32 11.0.0.1 100 0 i *>i100.1.1.2/32 11.0.0.1 100 0 i *>i200.1.1.1/32 11.0.0.1 100 0 i *>i200.1.1.2/32 11.0.0.1 100 0 i	
Step 6	**show bgp vpnv4 unicast** *rd-id vpnv4-prefix* **detail** **Example:** RP/0/RSP0/CPU0:router# **show bgp vpnv4 unicast rd 30.30.30.30:0 1.1.1.0/24 detail** Fri Aug 21 00:28:57.824 PDT BGP routing table entry for 1.1.1.0/24, Route Distinguisher: 30.30.30.30:0 Versions: Process bRIB/RIB SendTblVer Speaker 26 26 Flags: 0x04103001+0x00000000; Last Modified: Aug 21 00:24:01.000 for 00:04:58 Paths: (1 available, best #1) Advertised to peers (in unique update groups): 20.0.0.1 Path #1: Received by speaker 0 Flags: 0x4000c00005060001, import: 0x80 Advertised to peers (in unique update groups): 20.0.0.1 200 300 32.0.0.2 from 32.0.0.2 (40.40.40.40) Received Label 24001 Origin IGP, localpref 100, valid, external, best, group-best, import-candidate, imported, reoriginated with stitching-rt Received Path ID 0, Local Path ID 1, version 26 Extended community: RT:100:2 Source AFI: VPNv4 Unicast, Source VRF: default, Source Route Distinguisher: 1:1	Displays detailed information of specified VPNv4 prefixes. The **show bgp vpnv6 unicast** provides similar output information for VPNv6 prefixes.
Step 7	**show bgp vrf foo** **Example:** RP/0/RSP0/CPU0:router# **show bgp vrf foo** Fri Aug 21 00:24:36.523 PDT	Displays summary information of prefixes in the specified VRF instance BGP table.

	Command or Action	Purpose
	BGP VRF foo, state: Active BGP Route Distinguisher: 30.30.30.30:0 VRF ID: 0x60000002 BGP router identifier 30.30.30.30, local AS number 100 Non-stop routing is enabled BGP table state: Active Table ID: 0xe0000011 RD version: 35 BGP main routing table version 35 BGP NSR Initial initsync version 4 (Reached) BGP NSR/ISSU Sync-Group versions 31/0 Status codes: s suppressed, d damped, h history, * valid, > best i - internal, r RIB-failure, S stale, N Nexthop-discard Origin codes: i - IGP, e - EGP, ? - incomplete Network Next Hop Metric LocPrf Weight Path Route Distinguisher: 30.30.30.30:0 (default for vrf foo) *> 1.1.1.0/24 32.0.0.2 0 200 300 i *> 1.1.2.0/24 32.0.0.2 0 200 300 i *>i100.1.1.1/32 11.0.0.1 100 0 i *>i100.1.1.2/32 11.0.0.1 100 0 i *>i200.1.1.1/32 11.0.0.1 100 0 i *>i200.1.1.2/32 11.0.0.1 100 0 i Processed 6 prefixes, 6 paths	
Step 8	**show bgp vrf foo ipv4 unicast 100.1.1.1/32 detail** **Example:** RP/0/RSP0/CPU0:router# **show bgp vrf foo ipv4 unicast 100.1.1.1/32 detail** Mon Dec 8 23:24:50.243 PST BGP routing table entry for 100.1.1.1/32, Route Distinguisher: 30.30.30.30:0 Versions: Process bRIB/RIB SendTblVer Speaker 43 43 Local Label: 24001 (with rewrite); Flags: 0x05081001+0x00000200; Last Modified: Dec 8 18:04:21.000 for 05:20:30 Paths: (1 available, best #1) Advertised to PE peers (in unique update groups): 32.0.0.2 Path #1: Received by speaker 0 Flags: 0x400061000d060005, import: 0x80 Advertised to PE peers (in unique update groups):	Displays detailed information for a specified prefix in the specified VRF instance BGP table.

	Command or Action	Purpose
	32.0.0.2 Local 11.0.0.1 (metric 2) from 20.0.0.1 (11.0.0.1) Received Label 1234 Origin IGP, localpref 100, valid, internal, best, group-best, import-candidate, imported, reoriginated Received Path ID 0, Local Path ID 1, version 43 Extended community: Encapsulation Type:8 Router MAC:aabb.ccdd.eeff RT:1:2 Originator: 11.0.0.1, Cluster list: 20.20.20.20 RIB RNH: table_id 0xe0000011, Encap 8, VNI 1234, MAC Address: aabb.ccdd.eeff, IP Address: 11.0.0.1, IP table_id 0xe0000000 Source AFI: L2VPN EVPN, Source VRF: default, Source Route Distinguisher: 100:1	
Step 9	**show bgp vpnv4 unicast update-group** **Example:** RP/0/RSP0/CPU0:router# **show bgp vpnv4 unicast update-group** Fri Aug 21 00:27:57.910 PDT Update group for VPNv4 Unicast, index 0.1: Attributes: Outbound policy: pass First neighbor AS: 200 Send communities Send GSHUT community if originated Send extended communities 4-byte AS capable Send Re-originated VPN routes Send multicast attributes Minimum advertisement interval: 30 secs Update group desynchronized: 0 Sub-groups merged: 0 Number of refresh subgroups: 0 Messages formatted: 8, replicated: 8 All neighbors are assigned to sub-group(s) Neighbors in sub-group: 0.2, Filter-Groups num:1 Neighbors in filter-group: 0.2(RT num: 0) 32.0.0.2 Update group for VPNv4 Unicast, index 0.3: Attributes: Neighbor sessions are IPv4 Internal Common admin First neighbor AS: 100 Send communities Send GSHUT community if originated Send extended communities 4-byte AS capable Send AIGP	Displays update-group details for BGP VPNv4 unicast address-family.

	Command or Action	Purpose
	Send Re-originated VPN routes Send multicast attributes Minimum advertisement interval: 0 secs Update group desynchronized: 0 Sub-groups merged: 0 Number of refresh subgroups: 0 Messages formatted: 2, replicated: 2 All neighbors are assigned to sub-group(s) Neighbors in sub-group: 0.1, Filter-Groups num:1 Neighbors in filter-group: 0.1(RT num: 0) 20.0.0.1	

Step 10	**show bgp l2vpn evpn update-group**
	Example:
	`RP/0/RSP0/CPU0:router# show bgp l2vpn evpn update-group`
	`Fri Aug 21 00:27:42.786 PDT`
	`Update group for L2VPN EVPN, index 0.2:`
	`Attributes:`
	`Neighbor sessions are IPv4`
	`Internal`
	`Common admin`
	`First neighbor AS: 100`
	`Send communities`
	`Send GSHUT community if originated`
	`Send extended communities`
	`4-byte AS capable`
	`Send AIGP`
	`Send multicast attributes`
	`Minimum advertisement interval: 0 secs`
	`Update group desynchronized: 0`
	`Sub-groups merged: 0`
	`Number of refresh subgroups: 0`
	`Messages formatted: 4, replicated: 4`
	`All neighbors are assigned to sub-group(s)`
	`Neighbors in sub-group: 0.1, Filter-Groups num:1`
	`Neighbors in filter-group: 0.1(RT num: 0)`
	`20.0.0.1`
		Displays update-group details for BGP L2VPN EVPN address-family.

Example: Data Center Interconnection Layer 3 Gateway Configuration

The following configurations provide an example Data Center Interconnection (DCI) Layer 3 Gateway configuration.

VTEP-related configuration

```
interface Loopback1
 ipv4 address 40.1.1.1 255.255.255.255
```

```
!

interface nve1
 source-interface Loopback1
 member vni 1
  vrf cust1
  host-reachabilty protocol bgp
 !
interface BVI1
 vrf cust1
 ipv4 address 10.99.1.30 255.255.255.0
 ipv6 address 10:99:1::30/64
!

l2vpn
 bridge group bg1
  bridge-domain bd1
   routed interface BVI1
   member vni 1
  !
 !
```

VRF-related configuration

```
vrf data-center-10
  import route-target 1:1
  export route-target 1:2
  import route-target 10:10 stitching
  export route-target 10:20 stitching
```

Data Center EVPN BGP neighbor-related configuration

```
router bgp 1
 neighbor 1.1.1.1
   address-family l2vpn evpn
     import stitching-rt reoriginate
     advertise vpnv4 unicast reoriginated
     advertise vpnv6 unicast reoriginated
     advertise vpnv4 unicast local stitching-rt
     advertise vpnv6 unicast local stitching-rt
     advertise l2vpn evpn disable
```

L3VPN BGP neighbor-related configuration

```
router bgp 2
neighbor 10.10.10.10
  address-family vpnv4
    import reoriginate stitching-rt
    advertise vpnv4 unicast reoriginated
```

The following example configuration shows how to configure the DCI router to forward default route to its Data Center neighbor.

```
router bgp 1
 address-family vpnv4 unicast
 address-family vpnv6 unicast
 address-family l2vpn evpn
 exit
 neighbor 1.1.1.1
  address-family l2vpn evpn
  default-originate
  exit
```

```
 vrf foo
  rd 2:1
  address-family ipv4 unicast
  allow vpn default-originate
  exit
 address-family ipv6 unicast
 allow vpn default-originate
 exit
exit
!
```

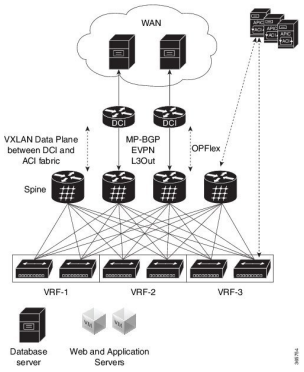
OpFlex

OpFlex is an open and extensible policy protocol used for transferring the policy information between a network policy controller such as the Cisco Application Policy Infrastructure Controller (APIC) and network elements such as routers that are configured as Data Center Interconnect (DCI) gateway. The policies are distributed using the Cisco® Application Centric Infrastructure (ACI) infrastructure within the fabric to the spine nodes. The spine nodes send policies to the DCI gateway through the OpFlex framework. An OpFlex framework resides between the spines and the DCIs. It enables the distribution of the DCI policy model from the fabric to the DCI gateways. DCI gateway acts as an OpFlex agent and the spine acts a policy repository. Fabric tenant interconnect (FTI) is the OpFlex agent application that runs on the DCI to generate and apply the tenant device configuration on the DCI. Policies configure the DCI service for a given tenant on the DCI gateway.

OpFlex Topology

Consider the topology where OpFlex framework is used between the DCI gateway and the Cisco ACI spine switches to automate fabric-facing tenant provisioning on the DCI gateway. When you configure a new external Layer 3 outside (L3Out) policy for a tenant on the Cisco Application Policy Infrastructure Controller (APIC), the controller programs all related information associated with that tenant, such as VRF instance name and BGP extended community route-target attributes for the Cisco ACI spine switches. The OpFlex framework running on the spine switches reads the L3Out managed object and converts it to the OpFlex model. This information is then pushed to the DCI gateway, which acts as a policy element for the OpFlex framework. On the DCI, the fabric facing configuration for the tenant VFR is auto-generated.

Figure 2: OpFlex Topology



Restrictions

The OpFlex feature is supported with the following restrictions:

- OpFlex feature is not supported on ASR9K with power PC based route-processor.
- FTI cannot generate configuration for multiple RTs of one address family in a tenant VRF provisioned in one fabric.
- On exhaustion of FTI configuration pools, the OpFlex notifications to add tenants are ignored. If existing tenants are deleted, the new tenants must be added again to enable OpFlex notifications to be re-sent to the DCI.
- FTI supports only Type 0 RT format: 2 byte ASN + 4 byte value. Type 1 and Type 2 RT formats are not supported.
- XML configuration and oper schema are not supported for FTI configuration and show commands.

Configure OpFlex

Perform the following tasks to configure the OpFlex session to automate fabric-facing tenant provisioning on the DCI gateway. This includes the one-time configuration that must be done on the DCI to enable DCI hand-off from an ACI fabric.

Configure BGP

Perform this task to enable address-family under BGP routing process for fabric and WAN peering.

SUMMARY STEPS

1. **configure**
2. **router bgp** *as-number*
3. **bgp router-id** *ip-address*
4. **address-family {vpnv4 | vpnv6} unicast**
5. **address-family l2vpn evpn**
6. Use the **commit** or **end** command.

DETAILED STEPS

	Command or Action	Purpose
Step 1	**configure** **Example:** `RP/0/RSP0/CPU0:router# configure`	Enters global configuration mode.
Step 2	**router bgp** *as-number* **Example:** `RP/0/RSP0/CPU0:router(config)# router bgp 1234`	Specifies the autonomous system number and enters the BGP configuration mode, allowing you to configure the BGP routing process.

	Command or Action	Purpose
Step 3	**bgp router-id** *ip-address* **Example:** `RP/0/RSP0/CPU0:router(config-bgp)# bgp router-id 198.51.100.1`	Configures the router with a specified router ID.
Step 4	**address-family {vpnv4 | vpnv6} unicast** **Example:** `RP/0/RSP0/CPU0:router(config-bgp)# address-family vpnv4 unicast`	Specifies either the vpnv4 or vpnv6 address family.
Step 5	**address-family l2vpn evpn** **Example:** `RP/0/RSP0/CPU0:router(config-bgp-af)# address-fmaily l2vpn evpn`	Configures EVPN address family.
Step 6	Use the **commit** or **end** command.	**commit** - Saves the configuration changes and remains within the configuration session. **end** - Prompts user to take one of these actions: • **Yes** - Saves configuration changes and exits the configuration session. • **No** - Exits the configuration session without committing the configuration changes. • **Cancel** - Remains in the configuration mode, without committing the configuration changes.

Configure BGP Session on the Fabric Side

Perform this task to configure BGP session on the fabric side.

SUMMARY STEPS

1. **configure**
2. **router bgp** *asn_id*
3. **neighbor** *ip-address*
4. **remote-as** *autonomous-system-number*
5. **update-source** *loopback*
6. **address-family l2vpn evpn**
7. **import stitching-rt reoriginate**
8. **advertise {vpnv4 | vpnv6} unicast re-originated**
9. Use the **commit** or **end** command.

DETAILED STEPS

	Command or Action	Purpose
Step 1	**configure** **Example:** `RP/0/RSP0/CPU0:router# configure`	Enters global configuration mode.
Step 2	**router bgp** *asn_id* **Example:** `RP/0/RSP0/CPU0:router(config)# router bgp 200`	Specifies the BGP AS number and enters the BGP configuration mode, allowing you to configure the BGP routing process.
Step 3	**neighbor** *ip-address* **Example:** `RP/0/RSP0/CPU0:router(config-bgp)# neighbor 209.165.201.1`	Places the router in neighbor configuration mode for BGP routing and configures the neighbor IP address 209.165.201.1 as a BGP peer.
Step 4	**remote-as** *autonomous-system-number* **Example:** `RP/0/RSP0/CPU0:router(config-bgp-nbr)# remote-as 100`	Creates a neighbor and assigns it a remote autonomous system number.
Step 5	**update-source** *loopback* **Example:** `RP/0/RSP0/CPU0:router(config-bgp-nbr)# update-source loopback2`	Allows BGP sessions to use the primary IP address from a particular interface as the local address.
Step 6	**address-family l2vpn evpn** **Example:** `RP/0/RSP0/CPU0:router(config-bgp-nbr)# address-fmaily l2vpn evpn`	Configures EVPN address family.
Step 7	**import stitching-rt reoriginate** **Example:** `RP/0/RSP0/CPU0:router(config-bgp-nbr-af)# import stitching-rt reoriginate`	Enables import of routing information from BGP EVPN NLRIs that has route target identifier matching the stitching route target identifier and exports this routing information after re-origination to the L2VPN BGP neighbor.
Step 8	**advertise {vpnv4 | vpnv6} unicast re-originated** **Example:** `RP/0/RSP0/CPU0:router(config-bgp-nbr-af)# advertise vpnv4 unicast re-originated`	Configures advertisement of VPNv4 or VPNv6 unicast routes that are redistributed from the L2VPN BGP neighbor, to the EVPN BGP neighbor. The route targets are changed to the stitching route targets before advertising onto the EVPN BGP neighbor.
Step 9	Use the **commit** or **end** command.	**commit** - Saves the configuration changes and remains within the configuration session. **end** - Prompts user to take one of these actions: • **Yes** - Saves configuration changes and exits the configuration session.

	Command or Action	Purpose
		• **No**- Exits the configuration session without committing the configuration changes. • **Cancel** - Remains in the configuration mode, without committing the configuration changes.

Configure BGP Session on the WAN Side

Perform this task to configure BGP session on the WAN side.

SUMMARY STEPS

1. **configure**
2. **router bgp** *asn_id*
3. **neighbor** *ip-address*
4. **remote-as** *autonomous-system-number*
5. **update-source** *loopback*
6. **address-family vpnv4 unicast**
7. **import re-originate stitching-rt**
8. **advertise {vpnv4 | vpnv6} unicast re-originated**
9. Use the **commit** or **end** command.

DETAILED STEPS

	Command or Action	Purpose
Step 1	**configure** **Example:** `RP/0/RSP0/CPU0:router# configure`	Enters global configuration mode.
Step 2	**router bgp** *asn_id* **Example:** `RP/0/RSP0/CPU0:router(config)# router bgp 200`	Specifies the BGP AS number and enters the BGP configuration mode, allowing you to configure the BGP routing process.
Step 3	**neighbor** *ip-address* **Example:** `RP/0/RSP0/CPU0:router(config-bgp)# neighbor 209.165.200.226`	Places the router in neighbor configuration mode for BGP routing and configures the neighbor IP address 209.165.200.226 as a BGP peer.
Step 4	**remote-as** *autonomous-system-number* **Example:** `RP/0/RSP0/CPU0:router(config-bgp-nbr)# remote-as 100`	Creates a neighbor and assigns it a remote autonomous system number.
Step 5	**update-source** *loopback* **Example:**	Allows BGP sessions to use the primary IP address from a particular interface as the local address.

	Command or Action	Purpose
	`RP/0/RSP0/CPU0:router(config-bgp-nbr)# update-source loopback2`	
Step 6	**address-family vpnv4 unicast** **Example:** `RP/0/RSP0/CPU0:router(config-bgp-nbr)# address-family vpnv4 unicast`	Enters VPNv4 address family configuration mode for the VPNv4 address family.
Step 7	**import re-originate stitching-rt** **Example:** `RP/0/RSP0/CPU0:router(config-bgp-nbr-af)# import re-originate stitching-rt`	Enables import of routing information from BGP EVPN NLRIs that has route target identifier matching the stitching route target identifier and exports this routing information after re-origination to the L2VPN BGP neighbor.
Step 8	**advertise {vpnv4 | vpnv6} unicast re-originated** **Example:** `RP/0/RSP0/CPU0:router(config-bgp-nbr-af)# advertise vpnv4 unicast re-originated`	Configures advertisement of VPNv4 or VPNv6 unicast routes that are redistributed from the L2VPN BGP neighbor, to the EVPN BGP neighbor. The route targets are changed to the stitching route targets before advertising onto the EVPN BGP neighbor.
Step 9	Use the **commit** or **end** command.	**commit** - Saves the configuration changes and remains within the configuration session. **end** - Prompts user to take one of these actions: • **Yes** - Saves configuration changes and exits the configuration session. • **No**- Exits the configuration session without committing the configuration changes. • **Cancel** - Remains in the configuration mode, without committing the configuration changes.

Configure DCI Underlay for Fabric and WAN Interfaces

Perform this task to configure DCI underlay for fabric facing interface and WAN facing interface. Perform this task on both the interfaces.

SUMMARY STEPS

1. **configure**
2. **interface** *type interface-path-id*
3. **ipv4 address** *ipv4-address mask*
4. Use the **commit** or **end** command.

DETAILED STEPS

	Command or Action	Purpose
Step 1	**configure** **Example:** `RP/0/RSP0/CPU0:router# configure`	Enters global configuration mode.
Step 2	**interface** *type interface-path-id* **Example:** `RP/0/RSP0/CPU0:router(config)# interface GigabitEthernet 0/0/0/0`	Configures Gigabit Ethernet interface.
Step 3	**ipv4 address** *ipv4-address mask* **Example:** `RP/0/RSP0/CPU0:router(config-if)# ipv4 address 209.165.200.226 255.255.255.224`	Specifies the IPv4 address and subnet mask for the interface.
Step 4	Use the **commit** or **end** command.	**commit** - Saves the configuration changes and remains within the configuration session. **end** - Prompts user to take one of these actions: • **Yes** - Saves configuration changes and exits the configuration session. • **No**- Exits the configuration session without committing the configuration changes. • **Cancel** - Remains in the configuration mode, without committing the configuration changes.

Configure IGP for ACI and WAN Reachability

Perform this task to configure IGP for ACI and WAN reachability.

SUMMARY STEPS

1. **configure**
2. **router ospf** *process-name*
3. **area** *area-id*
4. **interface** *type interface-path-id*
5. **exit**
6. **exit**
7. **area** *area-id*
8. **nssa**
9. **interface loopback** *loopback-id*
10. **exit**
11. **interface** *type interface-path-id*
12. Use the **commit** or **end** command.

DETAILED STEPS

	Command or Action	Purpose
Step 1	**configure** **Example:** `RP/0/RSP0/CPU0:router# configure`	Enters global configuration mode.
Step 2	**router ospf** *process-name* **Example:** `RP/0/RSP0/CPU0:router(config)# router ospf 100`	Enables OSPF routing for the specified routing process and places the router in router configuration mode.
Step 3	**area** *area-id* **Example:** `RP/0/RSP0/CPU0:router(config-ospf)# area 0`	Enters area configuration mode and configures an area for the OSPF process.
Step 4	**interface** *type interface-path-id* **Example:** `RP/0/RSP0/CPU0:router(config-ospf-ar)# interface GigabitEthernet 0/0/0/1`	Configures Gigabit Ethernet interface. Enables reachability to WAN.
Step 5	**exit** **Example:** `RP/0/RSP0/CPU0:router(config-ospf-ar-if)# exit`	Exits the interface submode and returns to area submode.
Step 6	**exit** **Example:** `RP/0/RSP0/CPU0:router(config-ospf-ar)# exit`	Exits the area submode and returns to router configuration mode.
Step 7	**area** *area-id* **Example:** `RP/0/RSP0/CPU0:router(config-ospf)# area 100`	Enters area configuration mode and configures an area for the OSPF process.
Step 8	**nssa** **Example:** `RP/0/RSP0/CPU0:router(config-ospf-ar)# nssa`	Specifies area as a NSSA area
Step 9	**interface loopback** *loopback-id* **Example:** `RP/0/RSP0/CPU0:router(config-ospf-ar)# interface loopback0`	Creates a loopback interface with the user-defined loopback identifier and enters the interface configuration mode.
Step 10	**exit** **Example:** `RP/0/RSP0/CPU0:router(config-ospf-ar-if)# exit`	Exits the interface submode and returns to area submmode.
Step 11	**interface** *type interface-path-id* **Example:**	Configures Gigabit Ethernet interface. Enables reachability to ACI.

	Command or Action	Purpose
	`RP/0/RSP0/CPU0:router(config-ospf-ar)# interface GigabitEthernet 0/0/0/0`	
Step 12	Use the **commit** or **end** command.	**commit** - Saves the configuration changes and remains within the configuration session. **end** - Prompts user to take one of these actions: • **Yes** - Saves configuration changes and exits the configuration session. • **No**- Exits the configuration session without committing the configuration changes. • **Cancel** - Remains in the configuration mode, without committing the configuration changes.

Configure MPLS towards WAN

Perform this task to configure MPLS on the DCI.

SUMMARY STEPS

1. **configure**
2. **mpls ldp**
3. **interface** *type interface-path-id*
4. **exit**
5. **exit**
6. **interface loopback** *instance*
7. **ipv4 address** *ipv4-address mask*
8. **exit**
9. **interface nve** *nve-identifier*
10. **source-interface loopback** *loopback-interface-identifier*
11. Use the **commit** or **end** command.

DETAILED STEPS

	Command or Action	Purpose
Step 1	**configure** **Example:** `RP/0/RSP0/CPU0:router# configure`	Enters global configuration mode.
Step 2	**mpls ldp** **Example:** `RP/0/RSP0/CPU0:router# mpls ldp`	Enables MPLS LDP configuration mode.

	Command or Action	Purpose
Step 3	**interface** *type interface-path-id* **Example:** `RP/0/RSP0/CPU0:router(config-ldp)# interface GigabitEthernet 0/0/0/1`	Configures Gigabit Ethernet interface.
Step 4	**exit** **Example:** `RP/0/RSP0/CPU0:router(config-ldp-if)# exit`	Exits the interface submode and returns to MPLS LDP submode.
Step 5	**exit** **Example:** `RP/0/RSP0/CPU0:router(config-ldp)# exit`	Exits the MPLS LDP submode and returns to global configuration mode.
Step 6	**interface loopback** *instance* **Example:** `RP/0/RSP0/CPU0:router(config)# interface Loopback0`	Enters interface configuration mode and names the new loopback interface.
Step 7	**ipv4 address** *ipv4-address mask* **Example:** `RP/0/RSP0/CPU0:router(config-if)# ipv4 address 209.165.200.227 255.255.255.224`	Specifies the IPv4 address and subnet mask for the interface.
Step 8	**exit** **Example:** `RP/0/RSP0/CPU0:router(config-if)# exit`	Exits the interface submode and returns to global configuration mode.
Step 9	**interface nve** *nve-identifier* **Example:** `RP/0/RSP0/CPU0:router(config)# interface nve 1`	Creates the NVE interface and enters the NVE interface configuration sub-mode.
Step 10	**source-interface loopback** *loopback-interface-identifier* **Example:** `RP/0/RSP0/CPU0:router(config-if)# source-interface loopback 0`	Sets a loopback interface as the source interface for the VTEP.
Step 11	Use the **commit** or **end** command.	**commit** - Saves the configuration changes and remains within the configuration session. **end** - Prompts user to take one of these actions: • **Yes** - Saves configuration changes and exits the configuration session. • **No**- Exits the configuration session without committing the configuration changes. • **Cancel** - Remains in the configuration mode, without committing the configuration changes.

Configure FTI Auto-Configuration Parameters

Perform this task to configure FTI auto-configuration parameters.

SUMMARY STEPS

1. **configure**
2. **dci-fabric-interconnect**
3. **auto-configuration-pool**
4. **bgp-as** *AS number*
5. **bridge group** *bridge-group-name*
6. **vrf** *vrf name* **ipv4-address** *ipv4 address*
7. **bd-pool** *bd range minimum bd range maximum*
8. **bvi-pool** *bvi range minimum bvi range maximum*
9. **vni-pool** *vni minimum range vni maximum range*
10. **local-vtep nve** *index*
11. Use the **commit** or **end** command.

DETAILED STEPS

	Command or Action	Purpose
Step 1	**configure** **Example:** `RP/0/RSP0/CPU0:router# configure`	Enters global configuration mode.
Step 2	**dci-fabric-interconnect** **Example:** `RP/0/RSP0/CPU0:router(config)# dci-fabric-interconnect`	Enters the fabric tenant interconnect submode.
Step 3	**auto-configuration-pool** **Example:** `RP/0/RSP0/CPU0:router(config-fti)# auto-configuration-pool`	Enters the auto configuration pool submode and enables to set the auto configuration pool parameters.
Step 4	**bgp-as** *AS number* **Example:** `RP/0/RSP0/CPU0:router(config-fti-acp)# bgp-as 1234`	Specifies the BGP AS number that is used when the configuration is generated. The BGP AS must be configured separately.
Step 5	**bridge group** *bridge-group-name* **Example:** `RP/0/RSP0/CPU0:router(config-fti-acp)# bridge group bg1`	Specifies the L2VPN bridge group to be used for generation of configuration.
Step 6	**vrf** *vrf name* **ipv4-address** *ipv4 address* **Example:** `vrf vrf1 ipv4-address 198.51.100.1`	Configures per-VRF BVI interface IP address. If the default IPv4 address from link-local range is not acceptable for tenant addressing, this IP address must be configured.

	Command or Action	Purpose
		If configured, this must match the WAN-side tenant VRF configuration.
Step 7	**bd-pool** *bd range minimum bd range maximum* **Example:** `RP/0/RSP0/CPU0:router(config-fti-acp)# bd-pool 1 1000`	Specifies the bridge domain range. The range is from 1 through 4000.
Step 8	**bvi-pool** *bvi range minimum bvi range maximum* **Example:** `RP/0/RSP0/CPU0:router(config-fti-acp)# bvi-pool 1 1000`	Specifies the bridge-group virtual interface (BVI) range. The range is from 1 through 4000.
Step 9	**vni-pool** *vni minimum range vni maximum range* **Example:** `RP/0/RSP0/CPU0:router(config-fti-acp)# vni-pool 1 1000`	Specifies the VNI range. The range is from 1 through 4000.
Step 10	**local-vtep nve** *index* **Example:** `RP/0/RSP0/CPU0:router(config-fti-acp)# local-vtep nve 1`	Specifies an NVE interface and configures it as VXLAN Tunnel EndPoint (VTEP) for the VXLAN.
Step 11	Use the **commit** or **end** command.	**commit** - Saves the configuration changes and remains within the configuration session. **end** - Prompts user to take one of these actions: • **Yes** - Saves configuration changes and exits the configuration session. • **No** - Exits the configuration session without committing the configuration changes. • **Cancel** - Remains in the configuration mode, without committing the configuration changes.

Configure OpFlex Session

This task enables the fabric tenant interconnect to setup an OpFlex session with the spine.

SUMMARY STEPS

1. **configure**
2. **dci-fabric-interconnect**
3. **fabric** *fabric identifier*
4. **opflex-peer** *spine IP address*
5. **exit**
6. **identity** *loopback IP address*

7. Use the **commit** or **end** command.

DETAILED STEPS

	Command or Action	Purpose
Step 1	**configure** **Example:** `RP/0/RSP0/CPU0:router# configure`	Enters global configuration mode.
Step 2	**dci-fabric-interconnect** **Example:** `RP/0/RSP0/CPU0:router(config)# dci-fabric-interconnect`	Enters the fabric tenant interconnect submode.
Step 3	**fabric** *fabric identifier* **Example:** `RP/0/RSP0/CPU0:router(config-fti)# fabric 1001`	Enters the fabric submode and you can configure the fabric parameters. The fabric identifier range is from 1000 through 9999.
Step 4	**opflex-peer** *spine IP address* **Example:** `RP/0/RSP0/CPU0:router(config-fti-fabric)# opflex-peer 192.0.2.1`	FTI sets up an OpFlex session with the spine.
Step 5	**exit** **Example:** `RP/0/RSP0/CPU0:router(config-fti-fabric)# exit`	Exits the current configuration mode and returns to fti submode.
Step 6	**identity** *loopback IP address* **Example:** `RP/0/RSP0/CPU0:router(config-fti)# identity 203.0.113.1`	Specifies the DCI's BGP loopback IP address.
Step 7	Use the **commit** or **end** command.	**commit** - Saves the configuration changes and remains within the configuration session. **end** - Prompts user to take one of these actions: • **Yes** - Saves configuration changes and exits the configuration session. • **No** - Exits the configuration session without committing the configuration changes. • **Cancel** - Remains in the configuration mode, without committing the configuration changes.

EVPN Default VRF Route Leaking

The EVPN Default VRF Route Leaking feature leak routes between EVPN address-family and IPv4/IPv6 unicast address-family (Default-VRF), enabling the data center hosts to access the Internet. This feature is an extension of Border Gateway Protocol (BGP) VRF Dynamic route leaking feature that provides connectivity between non-default VRF hosts and Default VRF hosts by exchanging routes between the non-default VRF and Default VRF. EVPN Default VRF Route Leaking feature extends the BGP VRF Dynamic leaking feature, by allowing EVPN/L3VPN hosts to communicate with Default VRF hosts.

The import process installs the Internet route in a VRF table or a VRF route in the Internet table, providing connectivity.

The BGP VRF Dynamic route leaking feature is enabled by:

- Importing from default-VRF to non-default-VRF using the following command in VRF address-family configuration mode.

  import from default-vrf route-policy *route-policy-name* **[advertise-as-vpn]**

  If the **advertise-as-vpn** keyword is used, the paths imported from the default-VRF to the non-default-VRF are advertised to the (EVPN/L3VPN) PEs as well as to the CEs. If the **advertise-as-vpn** keyword is not used, the paths imported from the default-VRF to the non-default-VRF are not advertised to the PEs. However, the paths are still advertised to the CEs.

  The EVPN Default VRF Route Leaking feature with **advertise-as-vpn** keyword, enables to advertise the paths imported from default-VRF to non-default VRFs to EVPN PE peers as well.

  A new command **advertise vpnv4/vpnv6 unicast imported-from-default-vrf disable** is added under neighbor address-family configuration mode for EVPN and VPNv4/VPNv6 unicast to disable advertisement of Default-VRF leaked routes to that neighbor.

- Importing from non-default-VRF to default-VRF using the following command in VRF address-family configuration mode.

  export to default-vrf route-policy *route-policy-name* **[advertise-as-vpn]**

  The Dynamic Route Leaking feature enables leaking of local and CE routes to Default-VRF.

  A new optional keyword **allow-imported-vpn** is added to the above command, when configured, enables the leaking of EVPN and L3VPN imported/re-originated routes to the Default-VRF.

A route-policy is mandatory to filter the imported routes. This reduces the risk of unintended import of routes between the Internet table and the VRF tables and the corresponding security issues. There is no hard limit on the number of prefixes that can be imported. The import creates a new prefix in the destination VRF, which increases the total number of prefixes and paths.

Note Each VRF importing global routes adds workload equivalent to a neighbor receiving the global table. This is true even if the user filters out all but a few prefixes.

Scale Limitation of Default Route Leaking

Default VRF route leaking uses Dynamic Route Leaking feature to leak prefixes between the default VRF and the DC VRF. Do not use Dynamic Route Leaking feature to leak default VRF prefixes to large number of DC VRFs, even if you filter out all prefixes except a few that are to be leaked.

The following are the key factors that affect the performance:

- The default VRF prefix scale, which is approximately 0.7 million internet prefixes.
- The number of DC VRFs the default VRF prefixes that are to be imported.

To improve the scale, either the prefix scale or the number of VRFs whose prefixes that are to be imported must be reduced.

To manage the scale limitation, Cisco recommends you to do the following:

- Host the Internet prefixes on an adjacent PE with IPv4 unicast peering with DCI, and advertise a default route towards the DCI. On the DCI, import the default route from default VRF to DC VRFs.
- Host the Internet prefixes on an adjacent PE with IPv4 unicast peering with DCI. On the DCI, configure a static default route in the DC VRF with the next hop of the default VRF pointing to the adjacent PE address.
- Configure the static default route 0.0.0.0/0 on DC VRF with nexthop as “vrf default”.

Note If the static routes are re-distributed to BGP, make sure it is not unintentionally advertised out.

EVPN Default VRF Route Leaking on the DCI for Internet Connectivity

The EVPN Default VRF Route Leaking feature leak routes between the Default-VRF and Data Center-VRF on the DCI to provide Internet access to data center hosts.

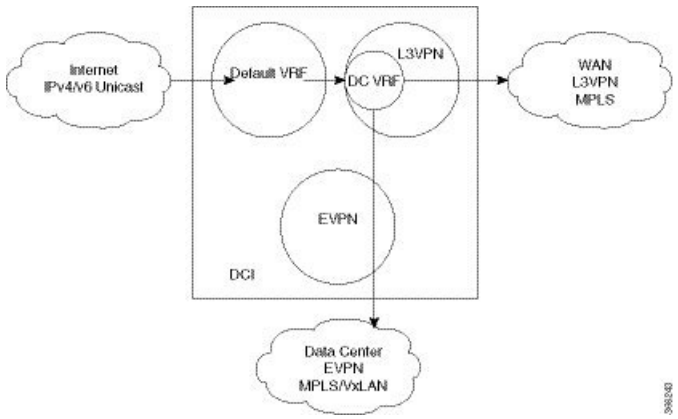
This feature is enabled by:

- Leaking routes from Default-VRF to Data Center-VRF
- Leaking routes to Default-VRF from Data Center-VRF

Leaking Routes from Default-VRF to Data Center-VRF

This section explains the process of leaking Default-VRF routes to Data Center-VRF.

Figure 3: Leaking Routes from Default-VRF to Data Center-VRF



Step 1 The Internet routes are present in the Default-VRF on the DCI.

Note A static default-route (0/0) can be configured under Default-VRF router static address-family configuration and redistributed to BGP.

Step 2 A route-policy is configured to select the routes to be leaked from Default-VRF to Data Center-VRF.

Example:

```
route-policy import-from-default-policy
  if destination in (100.10.0.0/16, 100.20.0.0/16) then
    pass
  endif
end-policy
!

route-policy import-from-default-policy-v6
  if destination in (100:10::0/64, 100:20::0/64) then
    pass
  endif
end-policy
!
```

Note Instead of leaking the internet routes, you can leak the default-route 0/0 from Default-VRF to Data Center-VRF using the following policy.

```
route-policy import-from-default-policy
  if destination in (0.0.0.0/0) then
    pass
  endif
end-policy
!

route-policy import-from-default-policy-v6
  if destination in (0::0/0) then
    pass
  endif
end-policy
!
```

Step 3 Leak Default-VRF routes specified in the route-policy to Data Center-VRF by configuring **import from default-vrf route-policy import-from-default-policy(-v6)** under Data Center VRF address-family configuration mode.

Example:

```
vrf data-center-vrf
 address-family ipv4 unicast
  import from default-vrf route-policy import-from-default-policy
!
address-family ipv6 unicast
  import from default-vrf route-policy import-from-default-policy-v6
!
```

Step 4 Advertise the leaked (Default-VRF) routes in the Data Center-VRF as EVPN routes towards Data Center routers by configuring **advertise-as-vpn** option.

Example:

```
vrf data-center-vrf
 address-family ipv4 unicast
  import from default-vrf route-policy import-from-default-policy advertise-as-vpn
!
address-family ipv6 unicast
  import from default-vrf route-policy import-from-default-policy-v6 advertise-as-vpn
!
```

Note To advertise any routes from L3VPN address-family to EVPN peers, use **advertise vpnv4/vpnv6 unicast re-originated [stitching-rt]** command under neighbor address-family L2VPN EVPN.

EVPN Default-originate

Instead of advertising the Default-VRF routes towards Data Center routers, default-originate can be configured under the EVPN neighbor address-family to advertise the default route. When default-originate is configured under the neighbor address-family for EVPN/L3VPN, there is no need to advertise the Default-VRF leaked routes to the data center and **advertise-as-vpn** need not be configured.

Example:

```
router bgp 100
  neighbor 40.0.0.1
    address-family l2vpn evpn
      default-originate

 vrf data-center-vrf
  rd auto
  address-family ipv4 unicast
    allow vpn default-originate
 !
  address-family ipv6 unicast
    allow vpn default-originate
```

Step 5 To block advertisement of the Default-VRF leaked routes towards a particular EVPN/L3VPN peer, use **advertise vpnv4/vpnv6 unicast imported-from-default-vrf disable** command under respective neighbor address-family.

Example:

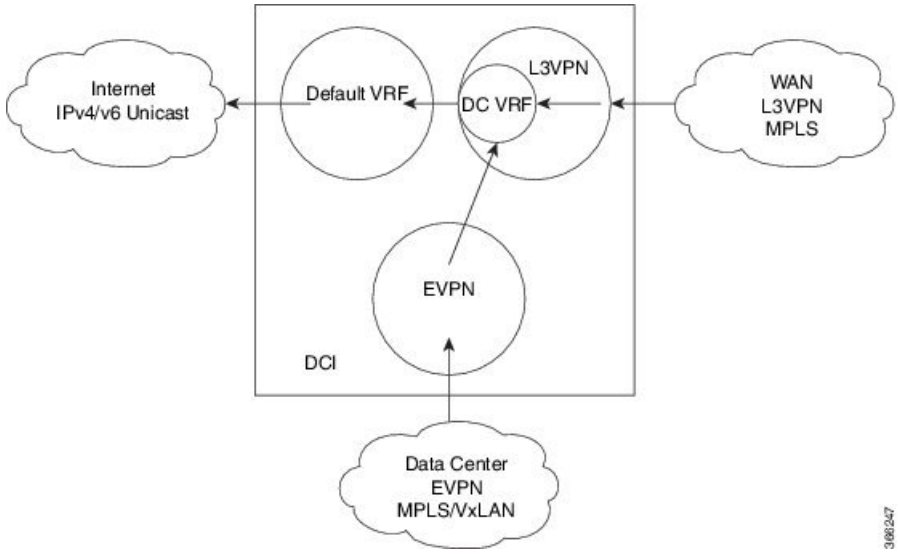
```
router bgp 100
  neighbor 40.0.0.1
    address-family l2vpn evpn
```

```
      advertise vpnv4 unicast imported-from-default-vrf disable
      advertise vpnv6 unicast imported-from-default-vrf disable
!
router bgp 100
  neighbor 60.0.0.1
    address-family vpnv4 unicast
      advertise vpnv4 unicast imported-from-default-vrf disable
    address-family vpnv6 unicast
      advertise vpnv6 unicast imported-from-default-vrf disable
```

Leaking Routes to Default-VRF from Data Center-VRF

This section explains the process of leaking Data Center-VRF routes to Default-VRF.

Figure 4: Leaking Routes to Default-VRF from Data Center-VRF



Step 1 Data Center routes are received on the DCI as EVPN Route-type 2 and Route-type 5 NLRI and imported to the Data Center VRFs.

Step 2 A route-policy is configured to select the routes to be leaked from Data Center-VRF to Default-VRF.

Example:

```
route-policy export-to-default-policy
  if destination in (200.47.0.0/16, 200.168.0.0/16) then
    pass
  endif
end-policy
!

route-policy export-to-default-policy-v6
  if destination in (200:47::0/64, 200:168::0/64) then
    pass
  endif
end-policy
```

```
!
```

Step 3 Leak Data Center-VRF routes specified in the above policy to Default-VRF by configuring **export to default-vrf route-policy export-to-default-policy(-v6) [allow-imported-vpn]** under Data Center-VRF address-family configuration mode.

Normally only local and CE VRF routes are allowed to be leaked to the Default-VRF, but **allow-imported-vpn** configuration enables leaking of EVPN/L3VPN imported routes to the Default-VRF.

Example:

```
vrf data-center-vrf
  address-family ipv4 unicast
    export to default-vrf route-policy export-to-default-policy [allow-imported-vpn]
  !
  address-family ipv6 unicast
    export to default-vrf route-policy export-to-default-policy-v6 [allow-imported-vpn]
  !
```

Step 4 The Leaked routes in the Default VRF are advertised to the Internet.

Note Instead of advertising the leaked routes to the Internet, an aggregate can be configured and advertised to the Internet.

Sample Router Configuration

The following sample configuration specifies how EVPN Default VRF Route Leaking feature is configured on a DCI router to provide Internet access to the data center hosts.

```
vrf data-center-vrf
  address-family ipv4 unicast
    import from default-vrf route-policy import-from-default-policy advertise-as-vpn
    export to default-vrf route-policy export-to-default-policy allow-imported-vpn
  !
  address-family ipv6 unicast
    import from default-vrf route-policy import-from-default-policy-v6 advertise-as-vpn
    export to default-vrf route-policy export-to-default-policy-v6 allow-imported-vpn
  !

route-policy import-from-default-policy
  if destination in (100.10.0.0/16, 100.20.0.0/16) then
    pass
  endif
end-policy
!

route-policy import-from-default-policy-v6
  if destination in (100:10::0/64, 100:20::0/64) then
    pass
  endif
end-policy
!

route-policy export-to-default-policy
  if destination in (200.47.0.0/16, 200.168.0.0/16) then
```

```
    pass
  endif
end-policy
!

route-policy export-to-default-policy-v6
  if destination in (200:47::0/64, 200:168::0/64) then
    pass
  endif
end-policy
!

router bgp 100
  neighbor 40.0.0.1
    address-family l2vpn evpn
      import stitching-rt re-originate
      advertise vpnv4 unicast re-originated stitching-rt
      advertise vpnv6 unicast re-originated stitching-rt

  neighbor 60.0.0.1
    address-family vpnv4 unicast
      import re-originate stitching-rt
      advertise vpnv4 unicast re-originated
      advertise vpnv4 unicast imported-from-default-vrf disable

    address-family vpnv6 unicast
      import re-originate stitching-rt
      advertise vpnv6 unicast re-originated
      advertise vpnv6 unicast imported-from-default-vrf disable
```

Sample Router Configuration: with default-originate

The following sample configuration specifies how EVPN Default VRF Route Leaking feature is configured along with default-originate on a DCI router to provide Internet access to data center hosts.

```
vrf data-center-vrf
  address-family ipv4 unicast
    import from default-vrf route-policy import-from-default-policy <= Remove
advertise-as-vpn=>
    export to default-vrf route-policy export-to-default-policy allow-imported-vpn
  !
  address-family ipv6 unicast
    import from default-vrf route-policy import-from-default-policy-v6 <= Remove
advertise-as-vpn=>
    export to default-vrf route-policy export-to-default-policy-v6 allow-imported-vpn
  !
route-policy import-from-default-policy
  if destination in (100.10.0.0/16, 100.20.0.0/16) then
    pass
  endif
end-policy
!
route-policy import-from-default-policy-v6
  if destination in (100:10::0/64, 100:20::0/64) then
    pass
  endif
end-policy
!
route-policy export-to-default-policy
  if destination in (200.47.0.0/16, 200.168.0.0/16) then
    pass
  endif
```

```
end-policy
!
route-policy export-to-default-policy-v6
  if destination in (200:47::0/64, 200:168::0/64) then
    pass
  endif
end-policy
!
router bgp 100
  neighbor 40.0.0.1
    address-family l2vpn evpn
      import stitching-rt re-originate
      advertise vpnv4 unicast re-originated stitching-rt
      advertise vpnv6 unicast re-originated stitching-rt
      default-originate <= Added=>

  neighbor 60.0.0.1
    address-family vpnv4 unicast
      import re-originate stitching-rt
      advertise vpnv4 unicast re-originated
      advertise vpnv4 unicast imported-from-default-vrf disable

    address-family vpnv6 unicast
      import re-originate stitching-rt
      advertise vpnv6 unicast re-originated
      advertise vpnv6 unicast imported-from-default-vrf disable

 vrf data-center-vrf
  rd auto
  address-family ipv4 unicast
    allow vpn default-originate <= Added=>
  !
  address-family ipv6 unicast
    allow vpn default-originate <= Added=>
```

EVPN Service VRF Route Leaking

The EVPN Service VRF Route Leaking feature enables connectivity to the services in the Service VRF to customers in EVPN Data Center VRF. The Service VRF and Data Center VRF routes can be IPv4 and/or IPv6 addresses. The Services VRF is any L3 VRF providing services reachable through connected, static, re-distributed IGP or BGP routes.

This feature leaks routes between Data Center VRF and Service VRF, enabling the EVPN/L3VPN hosts to access the Services in the Service VRF. This feature rely on Border Gateway Protocol (BGP) VRF extranet feature that imports routes between two VRFs.

The import process installs the Data Center VRF routes in a Service VRF table or a Service VRF routes in the Data Center VRF table, providing connectivity.

The BGP Service VRF route leaking feature is enabled by:

- Importing routes from Service VRF to Data Center VRF and advertising it as EVPN/L3VPN route from Data Center VRF.
    - Importing Service VRF routes to Data Center VRF by attaching Data Center VRF import RTs to Service VRF routes.

      This can be achieved by configuring one or more Data Center VRF import RTs as export RT of Service VRF, or configuring a Service VRF export route-policy to attach import RT EXTCOMM

to Service VRF routes matching the import RTs of Data Center VRF using the following command in Service VRF address-family configuration mode.

export route-policy service-vrf-export-route-policy-name

Where the route-policy "service-vrf-export-route-policy-name" attaches the RT EXTCOMM matching the one or more import RTs of Data Center VRF to Service VRF routes.

- Advertising Data Center VRF imported routes that are exported from Service VRFs as EVPN/L3VPN NLRI from Data Center VRF using the following command in Data Center VRF address-family configuration mode.

  import from vrf advertise-as-vpn

  If the **advertise-as-vpn** keyword is used, the paths imported from the Service VRF to the Data Center VRF are advertised to the (EVPN/L3VPN) PEs as well as to the CEs. If the **advertise-as-vpn** keyword is not used, the paths imported from the Service VRF to the Data Center VRF are not advertised to the PEs. However, the paths are still advertised to the CEs.

- Block advertising Data Center VRF leaked routes from being advertised to a neighbor using the following command in neighbor address-family configuration mode.

  advertise vpnv4/vpnv6 unicast imported-from-vrf disable

  A new command **advertise vpnv4/vpnv6 unicast imported-from-vrf disable** is added under neighbor address-family configuration mode for EVPN and VPNv4/VPNv6 unicast to disable advertisement of VRF to VRF leaked routes to that neighbor.

- Importing EVPN/L3VPN routes from Data Center VRF to Service VRF
  - Importing EVPN/L3VPN routes from Data Center VRF to Service VRF by attaching Service VRF import RTs.

    This can be achieved by configuring one or more Service VRF import RTs as export RT of Data Center VRF, or configuring a Data Center VRF export route-policy to attach import RT EXTCOMM to Data Center VRF routes matching the import RTs of Service VRF using the following command in Data Center VRF address-family configuration mode.

    export route-policy data-center-vrf-export-route-policy-name

    The route-policy "data-center-vrf-export-route-policy-name" attaches the RT EXTCOMM matching one or more import RTs of Service VRF.

  - Allow leaking of Data Center VRF routes to Service VRF by using the following command in Data Center VRF address-family configuration mode.

    export to vrf allow-imported-vpn

Note In order to prevent un-intended import of routes to VRFs, select unique RT's to import routes between Service VRF and Data Center VRF, which are not used for normal import of VPN/EVPN routes to Data Center VRFs.

The Extranet Route Leaking feature enables leaking of local and CE routes from one VRF to another VRF. A new command **export to vrf allow-imported-vpn** is added to enable the leaking of EVPN and L3VPN imported/re-originated Data Center VRF routes to the Service VRF.

Note A route-policy is preferred to filter the imported routes. This reduces the risk of unintended import of routes between the Data Center VRF and the Service VRF, and the corresponding security issues. There is no hard limit on the number of prefixes that can be imported. The import creates a new prefix in the destination VRF, which increases the total number of prefixes and paths.

Note This feature does not advertise EVPN/L3VPN PE routes imported to Data Center VRF and leaked to Service VRF as EVPN/L3VPN PE route.

EVPN Service VRF Route Leaking on the DCI for Service Connectivity

The EVPN Service VRF Route Leaking feature leaks routes between the Service VRF and Data Center VRF on the DCI to provide access to Services to data center hosts.

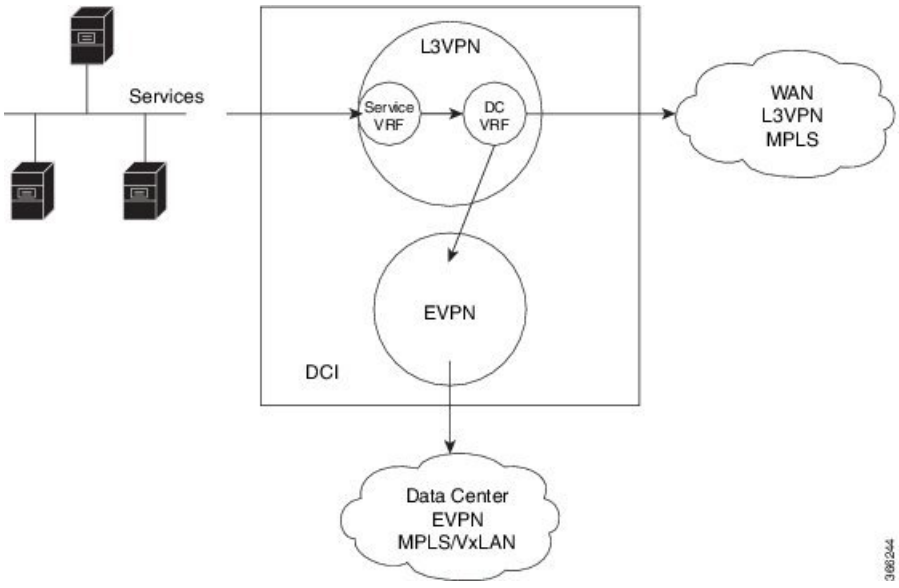
This feature is enabled by:

- Leaking routes from Service VRF to Data Center VRF
- Leaking routes to Service VRF from Data Center VRF

Leaking Routes from Service VRF to Data Center VRF

This section explains the process of leaking Service VRF routes to Data Center VRF.

Figure 5: Leaking Routes from Service VRF to Data Center VRF



Step 1 The Service routes are present in the Service VRF on the DCI.

Step 2 A route-policy is configured to select the routes to be leaked from Service VRF to Data Center VRF.

Example:

```
route-policy service-vrf-export-policy
  if destination in (100.10.0.0/16, 100.20.0.0/16) then
    set extcommunity rt (1:1) additive <--- matches import RT of Data Center-VRF
  endif
end-policy
!
route-policy service-vrf-export-policy-v6
  if destination in (100:10::0/64, 100:20::0/64) then
    set extcommunity rt (1:1) additive <--- matches import RT of Data Center-VRF
  endif
end-policy
!
```

Step 3 Leak Service VRF routes specified in the route-policy to Data Center VRF by configuring **export route-policy service-vrf-export-policy(-v6)** under Service VRF address-family configuration mode.

Example:

```
vrf service-vrf
  address-family ipv4 unicast
    import route-target
      3:1
      4:1 stitching
    export route-policy service-vrf-export-policy
    export route-target
      3:1
      4:1 stitching
   !
  address-family ipv6 unicast
    import route-target
      3:1
      4:1 stitching
    export route-policy service-vrf-export-policy-v6
    export route-target
      3:1
      4:1 stitching
   !
```

Step 4 Advertise the leaked (Service VRF) routes in the Data Center VRF as EVPN/L3VPN routes towards Data Center routers by configuring **import from vrf advertise-as-vpn** under Data Center VRF address-family configuration mode..

Example:

```
vrf data-center-vrf
  address-family ipv4 unicast
    import from vrf advertise-as-vpn
    import route-target
      1:1
      100:1
      200:1 stitching
    export route-target
      100:1
      200:1 stitching
  !
  address-family ipv6 unicast
    import from vrf advertise-as-vpn
    import route-target
```

```
      1:1
      100:1
      200:1 stitching
    export route-target
      100:1
      200:1 stitching
  !
```

Note To advertise any routes from L3VPN address-family to EVPN peers, use **advertise vpnv4/vpnv6 unicast re-originated [stitching-rt]** command under neighbor address-family L2VPN EVPN.

EVPN Default-originate

Instead of advertising the Service VRF routes towards Data Center routers, default-originate can be configured under the EVPN neighbor address-family to advertise the default route. When **allow vpn default-originate** is configured under the Data Center VRF, there is no need to advertise the Service VRF leaked routes to the data center and **advertise-as-vpn** need not be configured.

Example:

```
router bgp 100
  neighbor 40.0.0.1
    address-family l2vpn evpn
      default-originate

 vrf data-center-vrf
  rd auto
  address-family ipv4 unicast
    allow vpn default-originate
 !
  address-family ipv6 unicast
    allow vpn default-originate
```

Step 5 To block advertisement of the Service VRF leaked routes towards a particular EVPN/L3VPN peer, use **advertise vpnv4/vpnv6 unicast imported-from-vrf disable** command under respective neighbor address-family.

Example:

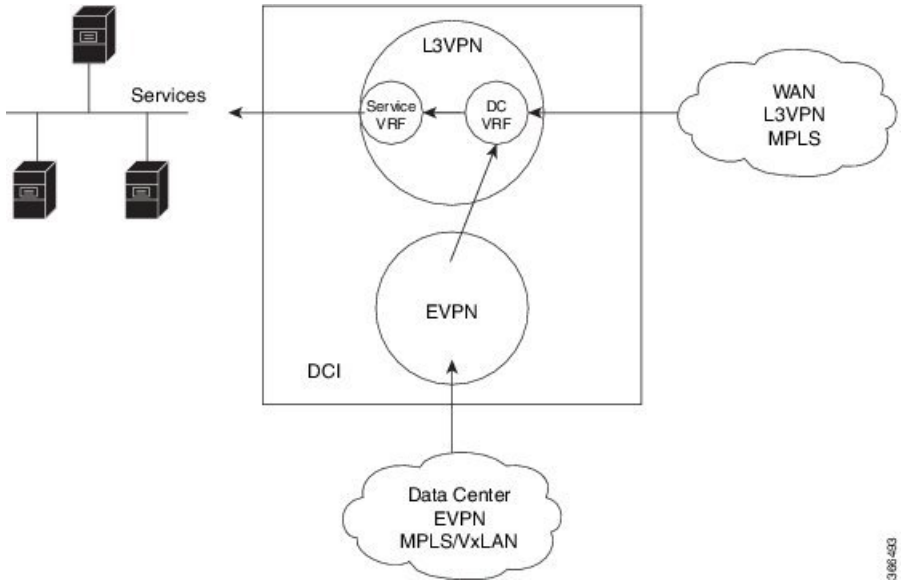
```
router bgp 100
  neighbor 40.0.0.1
    address-family l2vpn evpn
      import stitching-rt re-originate
      advertise vpnv4 unicast re-originated stitching-rt
      advertise vpnv4 unicast imported-from-vrf disable
      advertise vpnv6 unicast re-originated stitching-rt
      advertise vpnv6 unicast imported-from-vrf disable
    !
router bgp 100
  neighbor 60.0.0.1
    address-family vpnv4 unicast
      import re-originate stitching-rt
      advertise vpnv4 unicast re-originated
      advertise vpnv4 unicast imported-from-vrf disable
    address-family vpnv6 unicast
      import re-originate stitching-rt
      advertise vpnv6 unicast re-originated
```

```
advertise vpnv6 unicast imported-from-vrf disable
```

Leaking Routes to Service VRF from Data Center VRF

This section explains the process of leaking Data Center VRF routes to Service VRF.

Figure 6: Leaking Routes to Service VRF from Data Center VRF



Step 1 Data Center routes are received on the DCI as EVPN Route-type 2 and Route-type 5 NLRI and imported to the Data Center VRFs.

Step 2 A route-policy is configured to select the routes to be leaked from Data Center VRF to Service VRF.

The policy attaches RT EXTCOMM to Data Center VRF routes matching one or more import RT of the Service VRF.

Example:

```
route-policy data-center-vrf-export-policy
  if destination in (200.47.0.0/16) then <--- EVPN PE route
    set extcommunity rt (4:1) additive <--- matches import stitching-RT of service-VRF
  if destination in (200.168.0.0/16) then <--- VPNv4 PE route
    set extcommunity rt (3:1) additive <--- matches import RT of service-VRF
  endif
end-policy
!
route-policy data-center-vrf-export-policy-v6
  if destination in (200:47::0/64) then <--- EVPN PE route
    set extcommunity rt (4:1) additive <--- matches import stitching-RT of service-VRF
  elseif destination in (200:168::0/64) then <--- VPNv6 PE route
    set extcommunity rt (3:1) additive <--- matches import RT of service-VRF
  endif
end-policy
!
```

Note An EVPN/L3VPN route received from a neighbor configured locally with "import stitching-rt re-originate" is imported to Data Center VRF if the route's RT EXTCOMM matches with one or more Data Center VRF import stitching RTs, and is leaked to Service VRF if the Data Center VRF route's RT EXTCOMM matches with one or more Service VRF import stitching RTs.

Step 3 Leak Data Center VRF routes specified in the above policy to Service VRF by configuring **export route-policy data-center-vrf-export-policy(-v6)** under Data Center VRF address-family configuration mode.

Normally only local and CE VRF routes are allowed to be leaked to the Service VRF, but **allow-imported-vpn** configuration enables leaking of EVPN/L3VPN imported routes to the Service VRF.

Example:

```
vrf data-center-vrf
  address-family ipv4 unicast
    import from vrf advertise-as-vpn
    import route-target
      1:1
      100:1
      200:1 stitching
    export route-policy data-center-vrf-export-policy
    export to vrf allow-imported-vpn
    export route-target
      100:1
      200:1 stitching
  !
  address-family ipv6 unicast
    import from vrf advertise-as-vpn
    import route-target
      1:1
      100:1
      200:1 stitching
    export route-policy data-center-vrf-export-policy-v6
    export to vrf allow-imported-vpn
    export route-target
      100:1
      200:1 stitching
  !
```

Step 4 The Data Center VRF leaked routes in the Service VRF are advertised to Service VRF CE peers.

Sample Router Configuration

The following sample configuration specifies how EVPN Service VRF Route Leaking feature is configured on a DCI router providing access to data center hosts to Services in the Service VRF.

```
vrf data-center-vrf
  address-family ipv4 unicast
    import from vrf advertise-as-vpn
    import route-target
      1:1
      100:1
      200:1 stitching
    export route-policy data-center-vrf-export-policy
    export to vrf allow-imported-vpn
    export route-target
      100:1
      200:1 stitching
```

```
  !
  address-family ipv6 unicast
    import from vrf advertise-as-vpn
    import route-target
      1:1
      100:1
      200:1 stitching
    export route-policy data-center-vrf-export-policy-v6
    export to vrf allow-imported-vpn
    export route-target
      100:1
      200:1 stitching
  !

vrf service-vrf
  address-family ipv4 unicast
    import route-target
      3:1
      4:1 stitching
    export route-policy service-vrf-export-policy
    export route-target
      3:1
      4:1 stitching
   !
  address-family ipv6 unicast
    import route-target
      3:1
      4:1 stitching
    export route-policy service-vrf-export-policy-v6
    export route-target
      3:1
      4:1 stitching
   !

route-policy data-center-vrf-export-policy
  if destination in (200.47.0.0/16) then
    set extcommunity rt (4:1) additive
  if destination in (200.168.0.0/16)
    set extcommunity rt (3:1) additive
  endif
end-policy
!

route-policy data-center-vrf-export-policy-v6
  if destination in (200:47::0/64) then
    set extcommunity rt (4:1) additive
  elseif destination in (200:168::0/64)
    set extcommunity rt (3:1) additive
  endif
end-policy
!

route-policy service-vrf-export-policy
  if destination in (100.10.0.0/16, 100.20.0.0/16) then
    set extcommunity rt (1:1) additive
  endif
end-policy
!

route-policy service-vrf-export-policy-v6
  if destination in (100:10::0/64, 100:20::0/64) then
    set extcommunity rt (1:1) additive
  endif
end-policy
```

```
!

route-policy pass-all
  pass
end-policy
!

router bgp 100
  neighbor 40.0.0.1
    remote-as 100
    address-family l2vpn evpn
      import stitching-rt re-originate
      advertise vpnv4 unicast re-originated stitching-rt
      advertise vpnv6 unicast re-originated stitching-rt
    !
  neighbor 60.0.0.1
    remote-as 200
    address-family vpnv4 unicast
      import re-originate stitching-rt
      route-policy pass-all in
      route-policy pass-all out
      advertise vpnv4 unicast re-originated
      advertise vpnv4 unicast imported-from-vrf disable
    address-family vpnv6 unicast
      import re-originate stitching-rt
      route-policy pass-all in
      route-policy pass-all out
      advertise vpnv6 unicast re-originated
      advertise vpnv6 unicast imported-from-vrf disable
```

Sample Router Configuration: with default-originate

The following sample configuration specifies how EVPN Service VRF Route Leaking feature is configured along with default-originate on a DCI router to provide data center hosts access to Services in the Service VRF..

```
vrf data-center-vrf
  address-family ipv4 unicast
    import from vrf advertise-as-vpn
    import route-target
      1:1
      100:1
      200:1 stitching
    export route-policy data-center-vrf-export-policy
    export to vrf allow-imported-vpn
    export route-target
      100:1
      200:1 stitching
  !
  address-family ipv6 unicast
    import from vrf advertise-as-vpn
    import route-target
      1:1
      100:1
      200:1 stitching
    export route-policy data-center-vrf-export-policy-v6
    export to vrf allow-imported-vpn
    export route-target
      100:1
      200:1 stitching
  !
```

```
vrf service-vrf
  address-family ipv4 unicast
    import route-target
      3:1
      4:1 stitching
    export route-policy service-vrf-export-policy
    export route-target
      3:1
      4:1 stitching
   !
  address-family ipv6 unicast
    import route-target
      3:1
      4:1 stitching
    export route-policy service-vrf-export-policy-v6
    export route-target
      3:1
      4:1 stitching
   !

route-policy data-center-vrf-export-policy
  if destination in (200.47.0.0/16) then
    set extcommunity rt (4:1) additive
  if destination in (200.168.0.0/16) then
    set extcommunity rt (3:1) additive
  endif
end-policy
!

route-policy data-center-vrf-export-policy-v6
  if destination in (200:47::0/64) then
    set extcommunity rt (4:1) additive
  elseif destination in (200:168::0/64) then
    set extcommunity rt (3:1) additive
  endif
end-policy
!

route-policy service-vrf-export-policy
  if destination in (100.10.0.0/16, 100.20.0.0/16) then
    set extcommunity rt (1:1) additive
  endif
end-policy
!

route-policy service-vrf-export-policy-v6
  if destination in (100:10::0/64, 100:20::0/64) then
    set extcommunity rt (1:1) additive
  endif
end-policy
!

route-policy pass-all
  pass
end-policy
!

router bgp 100
  neighbor 40.0.0.1
    remote-as 100
    address-family l2vpn evpn
      import stitching-rt re-originate
      advertise vpnv4 unicast re-originated stitching-rt
      advertise vpnv4 unicast imported-from-vrf disable
```

```
    advertise vpnv6 unicast re-originated stitching-rt
    advertise vpnv6 unicast imported-from-vrf disable
    default-originate <= Added=>
  !
neighbor 60.0.0.1
  remote-as 200
  address-family vpnv4 unicast
    import re-originate stitching-rt
    route-policy pass-all in
    route-policy pass-all out
    advertise vpnv4 unicast re-originated
    advertise vpnv4 unicast imported-from-vrf disable
    default-originate <= Added=>
  address-family vpnv6 unicast
    import re-originate stitching-rt
    route-policy pass-all in
    route-policy pass-all out
    advertise vpnv6 unicast re-originated
    advertise vpnv6 unicast imported-from-vrf disable
    default-originate <= Added=>

vrf data-center-vrf
  rd auto
  address-family ipv4 unicast
    allow vpn default-originate <= Added=>
  !
  address-family ipv6 unicast
    allow vpn default-originate <= Added=>
```