Layer 3 Access

This chapter explains configuring Layer 3 wireless controller deployments on Cisco Catalyst 9800 series, covering L3 access setup, OSPF and PIM multicast integration, DHCP/NAT with VRF, and verification commands.

Layer 3 access

A Layer 3 access is a wireless controller deployment mode that

- terminates wireless client subnets directly on the controller
- supports Layer 3 routing functions such as OSPFv2 and PIM-SM, and
- enables client subnet segmentation, overlapping IP support, and a scalable network design.

Starting from Cisco IOS XE 17.13.1, the Cisco Catalyst 9800 series wireless controller platforms can be deployed as a Layer 3 (L3) network to perform routing functions.

In Cisco IOS XE 17.12.x and earlier releases, these platforms are deployed as Layer 2 network elements. In these deployments, wireless client subnets are terminated at an upstream network element. Upstream refers to the direction in which data is transferred from clients to a server. The controller forwards traffic based on the MAC address of the clients.

The L3 access feature terminates wireless client subnets in the controller and supports Layer 3 forwarding for wireless client traffic. When L3 is enabled on a specific SSID, the client VLAN of that SSID is terminated at the controller. In this scenario, the wireless controller forwards traffic based on the network layer (IP) address.

This enables:

- Segmentation and client overlapping IP address support using VRF.
- Flexible network design and faster convergence.
- Consistency in network design.
- Scalability for upstream switches and routers.

The core focus is seamless integration of OSPF and multicast routing. This transition empowers wireless networks to respond dynamically to changing business requirements, ensuring optimal performance and agility in dynamic networking environments.

Restrictions for Layer 3 access

- By default, the L3 access is disabled on a WLAN.
- Only N+1 redundancy is supported with L3 access.
- You cannot configure multiple IP addresses in an SVI.
- High Availability SSO is not supported in L3 WLANs.
- In mixed mode (L2 and L3 WLANs), HA SSO with Loopback as WMI is not supported.
- The **ip radius source-interface vrf** global command is not supported.
- These NAT CLIs are not supported in Cisco IOS XE 17.13.1:

```
show ip nat aggregation
show ip nat bpa
show ip nat ha
show ip nat limits
show ip nat map
show ip nat platform
show ip nat pool
show ip nat portblock
show ip nat redundancy
show ip nat route-dia
show ip nat translations
clear ip nat translations
```

- Multicast stream is not supported with VRF.
- Client IPv6 is not supported. The controller or AP cannot have IPv6 addresses.

Use cases for Layer 3 access

Layer 3 access support

- Segmentation and client overlapping IP address support.
- Flexible and optimized network design using L3 access.

Network Address Translation (NAT) support

- Translating client traffic in the guest network to reach the corporate services (For instance, Cisco ISE).
- Hiding the private IP addresses of clients from outside networks.

Note Only NAT with IPv4 to IPv4 translation is supported in Cisco IOS XE 17.13.1.

Enable Layer 3 access on policy profile (GUI)

Enable Layer 3 access on a policy profile to allow client devices connected to the WLAN to utilize Layer 3 forwarding using the GUI.

Procedure

Step 1 Choose **Configuration** > **Tags & Profiles** > **Policy**.

Step 2 Select a policy profile and in the **Edit Policy Profile** window, go to the advanced policy profile properties.

Step 3 Under the **Advanced** tab, enable **L3 Access** on the policy profile so that client traffic on a WLAN that has this policy can benefit from Layer 3 forwarding.

Step 4 Click **Apply to Device**.

Enable Layer 3 access on policy profile (CLI)

Configure a wireless policy profile so that wireless clients can obtain IP addresses and network access through Layer 3 using commands.

Procedure

Step 1 Enter the global configuration mode.

Example:

```
Device# configure terminal
```

Step 2 Configure a wireless policy profile.

Example:

```
Device(config)# wireless profile policy profile-policy
```

Step 3 Disable the wireless policy profile.

Example:

```
Device(config-wireless-policy)# shutdown
```

Step 4 Enable L3 access in the wireless policy profile.

Example:

```
Device(config-wireless-policy)# l3-access
```

Step 5 Enable the wireless policy profile.

Example:

```
Device(config-wireless-policy)# no shutdown
```

Configure a client gateway (GUI)

Configure a client gateway by associating an SVI interface with a VRF and enabling Autostate Disable to keep the SVI up regardless of connected port status using the GUI.

Procedure

Step 1 Choose **Configuration** > **Layer2** > **VLAN** and select the **SVI** tab.

Step 2 Click an SVI interface. On the **General** tab of the **Edit SVI** window, select a VRF from the drop-down list to associate it with the SVI interface.

Step 3 Enable the **Autostate Disable** to keep the SVI up, even when all ports on the VLAN are down.

Step 4 Click **Save & Apply to Device**.

Configure a client gateway (CLI)

Establish a client gateway interface, assign it to a VRF, and configure its operational parameters using the commands.

Procedure

Step 1 Enter the global configuration mode.

Example:

```
Device# configure terminal
```

Step 2 Specify an interface and enter the interface configuration mode.

Example:

```
Device(config)# interface type number
```

Step 3 Activate multiprotocol VRF in an interface.

Example:

```
Device(config-if)# vrf forwarding vrf-name
```

Step 4 Define the IP address for the VRF.

Example:

```
Device(config-if)# ip address ip-address mask-address
```

Step 5 Configure SVI to ensure that SVI is up even if the VLAN is not switched out.

Example:

```
Device(config-if)# no autostate
```

Step 6 Exit the interface configuration mode and enter the global configuration mode.

Example:

```
Device(config-if)# end
```

Enable internal DHCP with VRF (CLI)

Enable the internal DHCP server on a WLAN with VRF support by configuring the policy profile and related settings using the device commands.

Procedure

Step 1 Enter the global configuration mode.

Example:

```
Device# configure terminal
```

Step 2 Configure WLAN policy profile and enter the wireless policy configuration mode.

Example:

```
Device(config)# wireless profile policy profile-policy
```

Step 3 Add a description for the policy profile and configure AAA policy override.

Example:

```
Device(config-wireless-policy)# description profile-policy-description
Device(config-wireless-policy)# aaa-override
```

Step 4 Enable DHCP Option 82 for the wireless clients and enable VRF on DHCP Option 82.

Example:

```
Device(config-wireless-policy)# ipv4 dhcp opt82
Device(config-wireless-policy)# ipv4 dhcp opt82 vrf
```

Step 5 Configure the WLAN's IPv4 DHCP server IP address and VRF name.

Example:

```
Device(config-wireless-policy)# ipv4 dhcp server ip-address vrf vrf-name
```

Step 6 Disable the wireless policy profile and enable L3 access in the wireless policy profile.

Example:

```
Device(config-wireless-policy)# shutdown
Device(config-wireless-policy)# l3-access
```

Step 7 Configure Network Access Control in the policy profile.

Example:

```
Device(config-wireless-policy)# nac
```

Step 8 Map the VLAN to a policy profile.

Example:

```
Device(config-wireless-policy)# vlan vlan-id
```

If vlan-id is not specified, the default native vlan 1 is applied. The valid range for *vlan-id* is 1 to 4096.

Step 9 Enable the wireless policy profile.

Example:

```
Device(config-wireless-policy)# no shutdown
```

Verify internal DHCP with VRF details

To verify the internal DHCP details, use this command:

```
Device# show run int Vlan55
Building configuration...
Current configuration : 290 bytes
!
interface Vlan55
vrf forwarding sample_guest
ip address 55.55.55.2 255.255.255.0
no ip proxy-arp
ip nat inside
ip cef accounting non-recursive external
ip ospf authentication message-digest
ip ospf message-digest-key 1 md5 cisco123
no autostate
no mop enabled
no mop sysid
end
```

To verify the NAT datapath statistics, use this command:

```
Device# show run int Loopback1
Building configuration...
Current configuration : 90 bytes
!
interface Loopback1
vrf forwarding sample_guest
ip address 7.7.7.1 255.255.255.0
end
ip dhcp pool l3_sample_guest
vrf sample_guest
network 55.55.55.0 255.255.255.0
default-router 55.55.55.2
```

To verify the IP entries from database, use this command:

```
Device# show wireless device-tracking database ip
IP                                          ZONE/VRF-TABLE-ID     STATE       DISCOVERY
  MAC                VRF-NAME
55.55.55.2                                  0x00000003          Reachable       Local
 001e.bd11.a0ff

               55.55.55.6                                    0x00000003          Reachable
     IPv4 DHCP   58a0.239b.d25f     sample_guest
```

Verify Layer 3 access details

To verify whether Layer 3 access is enabled for a specific policy profile, use this command:

```
Device# show wireless profile policy detailed default-policy-profile
Policy Profile Name                : default-policy-profile
Description                        : default policy profile
Status                             : ENABLED
VLAN                               : 20
.
.
.

                L3 Forwarding                                                   :ENABLED
```

To view whether the Layer 3 access is enabled under policy profile, use this command:

```
Device# show wireless profile policy all
Policy Profile Name                : default-policy-profile
Description                        : default policy profile
Status                             : ENABLED
VLAN                               : 20
.
.
.

                L3 Forwarding                                                   :ENABLED
```

To verify the client information, use this command:

```
Device# show wireless client mac-address <mac-address> detail
Client MAC Address : a886.ddb2.05e9
.
.
.

L3 Forwarding: Enabled
```

To verify the client gateway details, use this command:

```
Device# show wireless client mac-address 0024.d742.46e4 detail | inc Gateway
.
.
.

                Client Gateway IPv4 Address : 117.117.117.1
```

Note The client gateway is displayed only if the client performs DHCP.

If the client learns IP using static or ARP, the client gateway will not be displayed.

OSPF protocols

An OSPF protocol is a link-state routing protocol that

- calculates the best path through a network using the shortest path first algorithm
- enables dynamic route selection for IP networks, and
- supports multiple authentication methods for secure device communication.

OSPF is a widely used Interior Gateway Protocol (IGP). Each device can verify the identity of the other devices it communicates with.

OSPF supports several types of authentication:

- Simple password authentication: Each device uses a clear-text password configured for authentication with other devices. However, this method is not secure as the password appears in configuration files and OSPF messages. This is not a secure way to configure devices.
- MD5 authentication: This method uses the MD5 algorithm to compute a hash value from the OSPF packet contents and a password.

Note Starting with Cisco IOS XE 17.13.1, OSPFv2 and ECMP are supported.

Configure OSPF

Set up Open Shortest Path First (OSPF) to enable dynamic routing across device interfaces.

Procedure

Step 1 Configure a clear-text password (or) message digest key in an OSPF-enabled interface.
Step 2 Create an OSPF routing process.
Step 3 Specify the range of IP addresses to associate with the routing process.
Step 4 Assign area IDs to be associated with that range.

What to do next

What to do next

Note To enable OSPF in SVI interfaces, you must enable Multicast over Multicast (MOM) using the **wireless multicast** *ip-address* command. This allows OSPF to establish neighbor adjacencies between SVIs.

These topics describe procedures to configure routing protocol:

Configure OSPF interfaces (GUI)

Enable and customize OSPF parameters on specific interfaces through the GUI.

Procedure

Step 1 Choose **Configuration** > **Interface** > **Ethernet** and select an interface to configure it with OSPF settings.

Step 2 In the **Configure Interface** window, ensure that you have configured an IP address, subnet mask and optionally a secondary IP address.

Step 3 In the **OSPF** section, enter the **Process ID** to enable OSPF on the interface.

Step 4 Enable the **BFD** to create a Bidirectional Forwarding Detection session between two systems. BFD provides a short-duration method of detecting failures in the forwarding path between two adjacent peers.

Step 5 Select the **Dead Interval Minimal** and enter the number of seconds in the **Hello Multiplier** field to set the interval at which at least one hello packet must be received, or else the neighbor is considered down.

Step 6 Select **Message Digest Authentication** to configure the authentication supported by OSPF.

Step 7 Under the **Message Digest Authentication- Key Map** association box, enter the Key, Type and Password.

Step 8 Click **Save & Apply to Device**.

Note
To configure OSPF in SVI interfaces, you must enable Multicast over Multicast (MOM). This allows OSPF to establish neighbor adjacencies between SVIs.

Configure OSPF protocol (GUI)

Configure OSPF protocol for dynamic routing between network devices using the GUI.

Procedure

Step 1 Choose **Configuration** > **Routing Protocol** > **OSPF** and click **Add**.

Step 2 In the **Add Route** page, select the router from the drop-down list.

Step 3 Enter the **Process ID**. It identifies the router's OSPF routing process to other routers.

Step 4 Enter a **Router ID**.

Step 5 Enable the **BFD** to create a Bidirectional Forwarding Detection session between two systems. BFD provides a short-duration method for detecting failures in the forwarding path between two adjacent switches. This includes interfaces, data links, and forwarding planes. OSPF registers as a protocol with BFD and receives forwarding path detection failure messages from BFD. You can configure BFD support for OSPF globally on all interfaces, or selectively on specific interfaces. BFD timers are negotiated. The BFD peers begin sending BFD control packets to each other at the negotiated interval.

Step 6 Enable **NSR** on a router with redundant Route Processors (RPs) so that it maintains its Open Shortest Path First (OSPF) state and adjacencies during both planned and unplanned RP switchovers. The active RP transfers OSPF state information to the standby RP by checkpointing the data. After a switchover to the standby RP, OSPF uses the checkpointed information to continue operation without interruption.

Optionally, you can check the corresponding check box to enable VRF and select the VRF Name. If you have not configured the VRF, use the link to configure it on the **Interface** > **VRF** page.

Step 7 For advanced options, check the **Advanced** radio button and populate these fields:

IP Address: Enter the address of the destination network for this route.

Wildcard: Enter the subnet mask used on that network.

Area: The OSPF area number for that network. Each router in a particular OSPF area maintains a topological database for that area.

Step 8 Click **Save & Apply to Device**.

Configure basic OSPF parameters (CLI)

Enable OSPF routing and define key parameters for efficient dynamic route advertisement and network segmentation using commands.

Procedure

Step 1 Enter the global configuration mode.

Example:

```
Device# configure terminal
```

Step 2 Enable OSPF routing.

Example:

```
Device(config)# router ospf process-id
```

The *process-id* is an internally used identification parameter that is locally assigned and can be any positive integer. Each OSPF routing process has a unique value.

Note
The OSPF for Routed Access supports a maximum of 1000 dynamically learned routes.

Step 3 Define a network on which the OSPF runs an area ID for that interface.

Example:

```
Device(config-router)# network address wildcard-mask area area-id
```

You can use the *wildcard-mask* to define one or more interfaces to be associated with a specific OSPF area. The *area-id* can be a decimal value or an IP address.

Step 4 Enable Bidirectional Forwarding Detection (BFD) in all interfaces.

Example:

```
Device(config-router)# bfd all-interfaces
```

Step 5 Return to privileged EXEC mode.

Example:

```
Device(config-router)# end
```

Configure OSPF interfaces (CLI)

Configure router interfaces to participate in OSPF routing, enabling dynamic route exchange within OSPF areas on the network.

Procedure

Step 1 Enter the global configuration mode.

Example:

```
Device# configure terminal
```

Step 2 Specify interface to configure OSPF interfaces.

Example:

```
Device(config)# interface GigabitEthernet interface-number
```

Step 3 Configure IP address for the OSPF interface.

Example:

```
Device(config-if)# ip address ip-address mask-address
```

Step 4 Enable message digest for a specific interface.

Example:

```
Device(config-if)# ip ospf authentication message-digest
```

Step 5 Enable message digest key for the OSPF.

Example:

```
Device(config-if)# ip ospf authentication message-digest-key key-number md5 password
```

Step 6 Assign interface and its network to OSPF process and area.

Example:

```
Device(config-if)# ip ospf value area area-id
```

Step 7 Enable BFD in an interface.

Example:

```
Device(config-if)# ip ospf bfd
```

Step 8 Return to privileged EXEC mode.

Example:

```
Device(config-if)# end
```

Verify routing protocol details

To verify the OSPF details, use this command:

```
Device# show ip ospf 1
Routing Process "ospf 1" with ID 31.31.31.1
 Start time: 00:01:46.103, Time elapsed: 03:12:34.745
 Supports only single TOS(TOS0) routes
 Supports opaque LSA
 Supports Link-local Signaling (LLS)
 Supports area transit capability
 Supports NSSA (compatible with RFC 3101)
 Supports Database Exchange Summary List Optimization (RFC 5243)
 Event-log enabled, Maximum number of events: 1000, Mode: cyclic
 Router is not originating router-LSAs with maximum metric
 Initial SPF schedule delay 50 msecs
 Minimum hold time between two consecutive SPFs 200 msecs
 Maximum wait time between two consecutive SPFs 5000 msecs
 Incremental-SPF disabled
 Initial LSA throttle delay 50 msecs
 Minimum hold time for LSA throttle 200 msecs
 Maximum wait time for LSA throttle 5000 msecs
 Minimum LSA arrival 100 msecs
 LSA group pacing timer 240 secs
 Interface flood pacing timer 33 msecs
 Retransmission pacing timer 66 msecs
 EXCHANGE/LOADING adjacency limit: initial 300, process maximum 300
 Number of external LSA 0. Checksum Sum 0x000000
 Number of opaque AS LSA 0. Checksum Sum 0x000000
 Number of DCbitless external and opaque AS LSA 0
 Number of DoNotAge external and opaque AS LSA 0
 Number of areas in this router is 1. 1 normal 0 stub 0 nssa
 Number of areas transit capable is 0
 External flood list length 0
 IETF NSF helper support enabled
 Cisco NSF helper support enabled
 Reference bandwidth unit is 100 mbps
    Area 1
        Number of interfaces in this area is 3
 Area has no authentication
 SPF algorithm last executed 03:11:47.277 ago
 SPF algorithm executed 9 times
 Area ranges are
 Number of LSA 5. Checksum Sum 0x0212EE
 Number of opaque link LSA 0. Checksum Sum 0x000000
 Number of DCbitless LSA 0
 Number of indication LSA 0
 Number of DoNotAge LSA 0
 Flood list length 0
```

To verify the OSPF database details, use this command:

```
Device# show ip ospf 1 database
OSPF Router with ID (31.31.31.1) (Process ID 1)
  Router Link States (Area 1)
Link ID         ADV Router      Age         Seq#       Checksum Link count
31.31.31.1      31.31.31.1      1470        0x8000000C 0x00289A 3
50.50.50.1      50.50.50.1      1745        0x8000000A 0x001018 3
51.51.51.1      51.51.51.1      1500        0x8000000A 0x008EFB 2
  Net Link States (Area 1)
Link ID         ADV Router      Age         Seq#       Checksum
30.30.30.2      50.50.50.1      1745        0x80000006 0x00B793
31.31.31.2      51.51.51.1      1500        0x80000006 0x0093AE
```

To verify the IP route details, use this command:

```
Device# show ip route
Codes: L - local, C - connected, S - static, R - RIP, M - mobile, B - BGP
       D - EIGRP, EX - EIGRP external, O - OSPF, IA - OSPF inter area
       N1 - OSPF NSSA external type 1, N2 - OSPF NSSA external type 2
       E1 - OSPF external type 1, E2 - OSPF external type 2, m - OMP
       n - NAT, Ni - NAT inside, No - NAT outside, Nd - NAT DIA
       i - IS-IS, su - IS-IS summary, L1 - IS-IS level-1, L2 - IS-IS level-2
       ia - IS-IS inter area, * - candidate default, U - per-user static route
       H - NHRP, G - NHRP registered, g - NHRP registration summary
       o - ODR, P - periodic downloaded static route, l - LISP
       a - application route
       + - replicated route, % - next hop override, p - overrides from PfR
       & - replicated local route overrides by connected
Gateway of last resort is not set
      5.0.0.0/8 is variably subnetted, 2 subnets, 2 masks
C        5.5.5.0/24 is directly connected, Vlan5
L        5.5.5.2/32 is directly connected, Vlan5
      6.0.0.0/8 is variably subnetted, 2 subnets, 2 masks
C        6.6.6.0/24 is directly connected, Vlan6
L        6.6.6.2/32 is directly connected, Vlan6
      30.0.0.0/8 is variably subnetted, 2 subnets, 2 masks
C        30.30.30.0/24 is directly connected, GigabitEthernet3
L        30.30.30.1/32 is directly connected, GigabitEthernet3
      31.0.0.0/8 is variably subnetted, 2 subnets, 2 masks
C        31.31.31.0/24 is directly connected, GigabitEthernet4
L        31.31.31.1/32 is directly connected, GigabitEthernet4
      32.0.0.0/24 is subnetted, 1 subnets
O        32.32.32.0 [110/2] via 30.30.30.2, 03:11:58, GigabitEthernet3
      50.0.0.0/32 is subnetted, 1 subnets
O        50.50.50.1 [110/2] via 30.30.30.2, 03:11:58, GigabitEthernet3
      51.0.0.0/32 is subnetted, 1 subnets
O        51.51.51.1 [110/2] via 31.31.31.2, 03:12:00, GigabitEthernet4
```

To verify the IP OSPF route list details, use this command:

```
Device# show ip ospf 1 route-list
OSPF Router with ID (31.31.31.1) (Process ID 1)
  Base Topology (MTID 0)
    Area 1
    Intra-area Route List
*   31.31.31.0/24, Intra, cost 1, area 1, Connected
      via 31.31.31.1, GigabitEthernet4
*   30.30.30.0/24, Intra, cost 1, area 1, Connected
      via 30.30.30.1, GigabitEthernet3
*   6.6.6.0/24, Intra, cost 1, area 1, Connected
      via 6.6.6.2, Vlan6
*>  32.32.32.0/24, Intra, cost 2, area 1
      via 30.30.30.2, GigabitEthernet3
*>  50.50.50.1/32, Intra, cost 2, area 1
      via 30.30.30.2, GigabitEthernet3
*>  51.51.51.1/32, Intra, cost 2, area 1
      via 31.31.31.2, GigabitEthernet4
    First Hop Forwarding Gateway Tree
 31.31.31.1 on GigabitEthernet4, count 1
 31.31.31.2 on GigabitEthernet4, count 1
 30.30.30.1 on GigabitEthernet3, count 1
 30.30.30.2 on GigabitEthernet3, count 2
 6.6.6.2 on Vlan6, count 1
```

To verify the OSPF traffic details, use this command:

```
Device# show ip ospf 1 traffic
OSPF Router with ID (31.31.31.1) (Process ID 1)
```

```
OSPF queue statistics for process ID 1:
                    InputQ     UpdateQ    OutputQ
  Limit             0          200        0
  Drops             0          0          0
  Max delay [msec]  1          1          1
  Max size          2          2          2
    Invalid         0          0          0
    Hello           0          0          0
    DB des          0          0          1
    LS req          1          1          1
    LS upd          1          1          0
    LS ack          0          0          0
  Current size      0          0          0
    Invalid         0          0          0
    Hello           0          0          0
    DB des          0          0          0
    LS req          0          0          0
    LS upd          0          0          0
    LS ack          0          0          0
Interface statistics:
.
.
.
    Interface GigabitEthernet4
Summary traffic statistics for process ID 1:
OSPF packets received/sent
  Type          Packets              Bytes
  RX Invalid    0                    0
  RX Hello      2435                 116880
  RX DB des     17                   584
  RX LS req     2                    96
  RX LS upd     24                   2360
  RX LS ack     24                   1436
  RX Total      2502                 121356
  TX Failed     0                    0
  TX Hello      3653                 506540
  TX DB des     6                    704
  TX LS req     2                    144
  TX LS upd     31                   4204
  TX LS ack     14                   1560
  TX Total      3706                 513152
OSPF header errors
  Length 0, Instance ID 0, Checksum 0, Auth Type 0,
  Version 0, Bad Source 0, No Virtual Link 0,
  Area Mismatch 0, No Sham Link 0, Self Originated 0,
  Duplicate ID 0, Hello 0, MTU Mismatch 0,
  Nbr Ignored 0, LLS 0, Unknown Neighbor 0,
  Authentication 0, TTL Check Fail 0, Adjacency Throttle 0,
  BFD 0, Test discard 0
OSPF LSA errors
  Type 0, Length 0, Data 0, Checksum 0
```

To verify the OSPF neighbor details, use this command:

```
Device# show ip ospf 1 neighbor
Neighbor ID     Pri   State           Dead Time   Address         Interface
51.51.51.1        1   FULL/DR         00:00:37    31.31.31.2      GigabitEthernet4
50.50.50.1        1   FULL/DR         00:00:39    30.30.30.2      GigabitEthernet3
```

To verify the OSPF neighbor summary, use this command:

```
Device#show ip ospf 1 neighbor summary
            OSPF Router with ID (31.31.31.1) (Process ID 1)
DOWN         0
ATTEMPT      0
```

```
INIT          0
2WAY          0
EXSTART       0
EXCHANGE      0
LOADING       0
FULL          2
Total count   2    (Undergoing NSF 0)
```

To verify the OSPF event details, use this command:

```
                Device# show ip ospf 1 events
             OSPF Router with ID (31.31.31.1) (Process ID 1)
1    Sep 21 21:49:12.406: Generate Changed Type-1 LSA, LSID 31.31.31.1, Seq# 8000000C, Age
 0, Area 1
2    Sep 21 21:48:44.064: Rcv Unchanged Type-2 LSA, LSID 31.31.31.2, Adv-Rtr 51.51.51.1,
Seq# 80000006, Age 1, Area 1
3    Sep 21 21:48:44.064: Rcv Unchanged Type-1 LSA, LSID 51.51.51.1, Adv-Rtr 51.51.51.1,
Seq# 8000000A, Age 1, Area 1
4    Sep 21 21:44:38.726: Rcv Unchanged Type-2 LSA, LSID 30.30.30.2, Adv-Rtr 50.50.50.1,
Seq# 80000006, Age 1, Area 1
5    Sep 21 21:44:38.726: Rcv Unchanged Type-1 LSA, LSID 50.50.50.1, Adv-Rtr 50.50.50.1,
Seq# 8000000A, Age 1, Area 1
.
.
.
30   Sep 21 19:01:45.594: End of SPF, Topo Base, SPF time 1ms, next wait-interval 800ms
.
.
.
74

                Sep 21 19:01:44.676: Generic:  ospf_external_route_sync  0x1

                75   Sep 21 19:01:44.676: Generic:  ospf_external_route_sync  0x1
76   Sep 21 19:01:44.676: Generic:  ospf_external_route_sync  0x0
77   Sep 21 19:01:44.676: Generic:  ospf_external_route_sync  0x0
78   Sep 21 19:01:44.676: Starting External processing, Topo Base in area 1
79   Sep 21 19:01:44.676: Starting External processing, Topo Base
80

                Sep 21 19:01:44.676: Generic:  ospf_inter_route_sync  0x1

81   Sep 21 19:01:44.676: Generic:  ospf_inter_route_sync  0x1
82   Sep 21 19:01:44.676: Starting summary processing, Topo Base, Area 1
83

                Sep 21 19:01:44.676: Generic:  post_spf_intra  0x0
84   Sep 21 19:01:44.676: Generic:  ospf_intra_route_sync  0x1

.
.
.
```

To verify the OSPF details in the database summary, use this command:

```
Device# show ip ospf 1 database database-summary
OSPF Router with ID (31.31.31.1) (Process ID 1)
Area 1 database summary
  LSA Type      Count    Delete   Maxage
  Router        3        0        0
  Network       2        0        0
  Summary Net   0        0        0
  Summary ASBR  0        0        0
```

```
  Type-7 Ext    0        0        0
    Prefixes redistributed in Type-7  0
  Opaque Link   0        0        0
  Opaque Area   0        0        0
  Subtotal      5        0        0
Process 1 database summary
  LSA Type      Count    Delete   Maxage
  Router        3        0        0
  Network       2        0        0
  Summary Net   0        0        0
  Summary ASBR  0        0        0
  Type-7 Ext    0        0        0
  Opaque Link   0        0        0
  Opaque Area   0        0        0
  Type-5 Ext    0        0        0
      Prefixes redistributed in Type-5  0
  Opaque AS     0        0        0
  Total         5        0        0
  Non-self      4
```

To verify the OSPF details in the internal database, use this command:

```
Device# show ip ospf 1 database internal
OSPF Router with ID (31.31.31.1) (Process ID 1)
  Stub Link States (Area 1)
Link ID         ADV Router      Age         Seq#       Checksum Mask
6.6.6.255       31.31.31.1      11545       0x0        0x006611 /24
30.30.30.255    31.31.31.1      11546       0x0        0x00032C /24
31.31.31.255    31.31.31.1      11548       0x0        0x00DE4D /24
32.32.32.255    50.50.50.1      11545       0x0        0x00F0FE /24
50.50.50.1      50.50.50.1      11545       0x0        0x005C5C /32
51.51.51.1      51.51.51.1      11547       0x0        0x002092 /32
  Router Link States (Area 1)
Link ID         ADV Router      Age         Seq#       Checksum Link count
31.31.31.1      31.31.31.1      1498        0x8000000C 0x00289A 3
50.50.50.1      50.50.50.1      1772        0x8000000A 0x001018 3
51.51.51.1      51.51.51.1      1527        0x8000000A 0x008EFB 2
  Net Link States (Area 1)
Link ID         ADV Router      Age         Seq#       Checksum
30.30.30.2      50.50.50.1      1772        0x80000006 0x00B793
31.31.31.2      51.51.51.1      1527        0x80000006 0x0093AE
```

To verify the OSPF details in the database network, use this command:

```
Device# show ip ospf 1 database network
OSPF Router with ID (31.31.31.1) (Process ID 1)
  Net Link States (Area 1)
  LS age: 1772
  Options: (No TOS-capability, DC)
  LS Type: Network Links
  Link State ID: 30.30.30.2 (address of Designated Router)
  Advertising Router: 50.50.50.1
  LS Seq Number: 80000006
  Checksum: 0xB793
  Length: 32
  Network Mask: /24
 Attached Router: 50.50.50.1
 Attached Router: 31.31.31.1
  LS age: 1527
  Options: (No TOS-capability, DC)
  LS Type: Network Links
  Link State ID: 31.31.31.2 (address of Designated Router)
  Advertising Router: 51.51.51.1
  LS Seq Number: 80000006
```

```
  Checksum: 0x93AE
  Length: 32
  Network Mask: /24
 Attached Router: 51.51.51.1
 Attached Router: 31.31.31.1
```

To verify the OSPF details in the database router, use this command:

```
Device# show ip ospf 1 database router
OSPF Router with ID (31.31.31.1) (Process ID 1)
  Router Link States (Area 1)
  LS age: 1498
  Options: (No TOS-capability, DC)
  LS Type: Router Links
  Link State ID: 31.31.31.1
  Advertising Router: 31.31.31.1
  LS Seq Number: 8000000C
  Checksum: 0x289A
  Length: 60
  Number of Links: 3
    Link connected to: a Transit Network
     (Link ID) Designated Router address: 31.31.31.2
     (Link Data) Router Interface address: 31.31.31.1
      Number of MTID metrics: 0
       TOS 0 Metrics: 1
    Link connected to: a Transit Network
     (Link ID) Designated Router address: 30.30.30.2
     (Link Data) Router Interface address: 30.30.30.1
      Number of MTID metrics: 0
       TOS 0 Metrics: 1
    Link connected to: a Stub Network
     (Link ID) Network/subnet number: 6.6.6.0
     (Link Data) Network Mask: 255.255.255.0
      Number of MTID metrics: 0
       TOS 0 Metrics: 1
.
.
.
```

To verify the OSPF details in the database topology, use this command:

```
Device# show ip ospf 1 database topology
OSPF Router with ID (31.31.31.1) (Process ID 1)
  Base Topology (MTID 0)
  Router Link States (Area 1)
Link ID         ADV Router      Age         Seq#       Checksum Link count
31.31.31.1      31.31.31.1      1498        0x8000000C 0x00289A 3
50.50.50.1      50.50.50.1      1772        0x8000000A 0x001018 3
51.51.51.1      51.51.51.1      1527        0x8000000A 0x008EFB 2
  Net Link States (Area 1)
Link ID         ADV Router      Age         Seq#       Checksum
30.30.30.2      50.50.50.1      1772        0x80000006 0x00B793
31.31.31.2      51.51.51.1      1527        0x80000006 0x0093AE
vWLC_TB1#
vWLC_TB1#show ip ospf 1 request-list
            OSPF Router with ID (31.31.31.1) (Process ID 1)
 Neighbor 51.51.51.1, interface GigabitEthernet4 address 31.31.31.2
 Request list size 0, maximum list size 1
 Neighbor 50.50.50.1, interface GigabitEthernet3 address 30.30.30.2
 Request list size 0, maximum list size 1
vWLC_TB1#
vWLC_TB1#show ip ospf flood-list
            OSPF Router with ID (31.31.31.1) (Process ID 1)
 Interface GigabitEthernet4, Queue length 0
```

```
 Interface GigabitEthernet3, Queue length 0
 Interface Vlan6, Queue length 0
```

To verify the OSPF request details, use this command:

```
Device# show ip ospf request-list Gi3 50.50.50.1
OSPF Router with ID (31.31.31.1) (Process ID 1)
 Neighbor 50.50.50.1, interface GigabitEthernet3 address 30.30.30.2
 Request list size 0, maximum list size 1
```

To verify the OSPF interface details, use this command:

```
Device# show ip ospf interface
GigabitEthernet4 is up, line protocol is up
  Internet Address 31.31.31.1/24, Interface ID 10, Area 1
  Attached via Network Statement
  Process ID 1, Router ID 31.31.31.1, Network Type BROADCAST, Cost: 1
  Topology-MTID    Cost    Disabled    Shutdown      Topology Name
        0           1         no          no            Base
  Transmit Delay is 1 sec, State BDR, Priority 1
  Designated Router (ID) 51.51.51.1, Interface address 31.31.31.2
  Backup Designated router (ID) 31.31.31.1, Interface address 31.31.31.1
  Timer intervals configured, Hello 10, Dead 40, Wait 40, Retransmit 5
    oob-resync timeout 40
    Hello due in 00:00:03
  Supports Link-local Signaling (LLS)
  Cisco NSF helper support enabled
  IETF NSF helper support enabled
  Can be protected by per-prefix Loop-Free FastReroute
  Can be used for per-prefix Loop-Free FastReroute repair paths
  Not Protected by per-prefix TI-LFA
  Index 1/3/3, flood queue length 0
  Next 0x0(0)/0x0(0)/0x0(0)
  Last flood scan length is 1, maximum is 2
  Last flood scan time is 0 msec, maximum is 0 msec
  Neighbor Count is 1, Adjacent neighbor count is 1
    Adjacent with neighbor 51.51.51.1  (Designated Router)
  Suppress hello for 0 neighbor(s)
  Cryptographic authentication enabled
    Youngest key id is 1
GigabitEthernet3 is up, line protocol is up
  Internet Address 30.30.30.1/24, Interface ID 9, Area 1
  Attached via Network Statement
  Process ID 1, Router ID 31.31.31.1, Network Type BROADCAST, Cost: 1
  Topology-MTID    Cost    Disabled    Shutdown      Topology Name
        0           1         no          no            Base
  Transmit Delay is 1 sec, State BDR, Priority 1
  Designated Router (ID) 50.50.50.1, Interface address 30.30.30.2
  Backup Designated router (ID) 31.31.31.1, Interface address 30.30.30.1
  Timer intervals configured, Hello 10, Dead 40, Wait 40, Retransmit 5
    oob-resync timeout 40
    Hello due in 00:00:06
  Supports Link-local Signaling (LLS)
  Cisco NSF helper support enabled
  IETF NSF helper support enabled
  Can be protected by per-prefix Loop-Free FastReroute
  Can be used for per-prefix Loop-Free FastReroute repair paths
  Not Protected by per-prefix TI-LFA
  Index 1/2/2, flood queue length 0
  Next 0x0(0)/0x0(0)/0x0(0)
  Last flood scan length is 1, maximum is 2
  Last flood scan time is 0 msec, maximum is 0 msec
  Neighbor Count is 1, Adjacent neighbor count is 1
    Adjacent with neighbor 50.50.50.1  (Designated Router)
  Suppress hello for 0 neighbor(s)
```

```
  Cryptographic authentication enabled
    Youngest key id is 1
Vlan6 is up, line protocol is up
  Internet Address 6.6.6.2/24, Interface ID 16, Area 1
  Attached via Interface Enable
  Process ID 1, Router ID 31.31.31.1, Network Type BROADCAST, Cost: 1
  Topology-MTID    Cost    Disabled    Shutdown      Topology Name
        0           1         no          no            Base
  Enabled by interface config, including secondary ip addresses
  Transmit Delay is 1 sec, State DR, Priority 1
  Designated Router (ID) 31.31.31.1, Interface address 6.6.6.2
  No backup designated router on this network
  Timer intervals configured, Hello 10, Dead 40, Wait 40, Retransmit 5
    oob-resync timeout 40
    Hello due in 00:00:01
  Supports Link-local Signaling (LLS)
  Cisco NSF helper support enabled
  IETF NSF helper support enabled
  Can be protected by per-prefix Loop-Free FastReroute
  Can be used for per-prefix Loop-Free FastReroute repair paths
  Not Protected by per-prefix TI-LFA
  Index 1/1/1, flood queue length 0
  Next 0x0(0)/0x0(0)/0x0(0)
  Last flood scan length is 0, maximum is 0
  Last flood scan time is 0 msec, maximum is 0 msec
  Neighbor Count is 0, Adjacent neighbor count is 0
  Suppress hello for 0 neighbor(s)
  Cryptographic authentication enabled
    Youngest key id is 1
```

PIM sparse mode

A PIM sparse mode is a multicast routing protocol that

- assumes recipients for multicast groups are sparsely distributed throughout the network
- builds multicast distribution trees rooted at a selected Rendezvous Point (RP) node by default, and
- requires routers to explicitly signal interest in specific multicast groups or sources.

Protocol Independent Multicast (PIM) includes multicast routing protocols for various environments.

For information about PIM-SM, see https://www.cisco.com/c/en/us/td/docs/ios-xml/ios/ipmulti_pim/configuration/15-sy/imc-pim-15-sy-book/ip6-mcast-pim-sm.html.

PIM-SM

Most network subnets do not require every multicast packet. Routers explicitly inform upstream neighbors of their interest in specific multicast groups and sources to receive multicast data.

A key requirement of PIM-SM is discovering the RP address for a multicast group through a shared tree.

Enable multicast traffic without VRF (GUI)

Enable multicast routing and required protocols to support multicast traffic delivery in VLANs without VRF using the GUI.

Procedure

Step 1 Choose **Configuration** > **Services** > **Multicast**.

Step 2 In the **PIM and Multicast Routing** section, configure multicast routing globally by enabling **Distributed Multicast-Routing**.

Step 3 Configure PIM RP-Address in the **PIM Configuration** sub-section. This configuration is required so that receivers can find the multicast source in the network. Choose the configuration options from these options:

- Enter the address to statically configure the RP Address.
- Enable Auto RP Listener to dynamically discover RP in a PIM-SM network.

Step 4 Click **Save & Apply to Device**.

Step 5 Designate the interface on which multicast traffic should be sent. To do so, go to **Configuration** > **Layer 2** > **VLAN** and select the **SVI** interface.

Step 6 Enable the **PIM Sparse Mode** protocol so that the SVI interface can participate in sparse mode multicast traffic transmission and join the multicast shared tree. This ensures that clients in that VLAN can receive multicast traffic from multiple multicast groups or sources.

Step 7 Select the desired IGMP version from the drop-down list to improve multicast packet direction. When this feature is enabled, the controller gathers IGMP reports from the clients and processes them. It creates unique multicast group IDs (MGIDs) from the IGMP reports by selecting the Layer 3 multicast address and the VLAN number, then sends the IGMP reports to the infrastructure switch.

Step 8 Select **IPv4** checkbox and enter the details.

Step 9 Click **Save & Apply to Device**.

Enable multicast traffic without VRF (CLI)

Enable multicast routing capabilities on a device in a flat (non-VRF) environment and ensure proper configuration for multicast traffic propagation using commands.

Procedure

Step 1 Enter the global configuration mode and enable IP multicast routing.

Example:

```
Device# configure terminal
Device(config)# ip multicast-routing distributed
```

The **distributed** keyword enables multicast globally.

Step 2 Enable multicast traffic and configure the address of a PIM Rendezvous Point (RP).

Example:

```
Device(config)# wireless multicast ip-address
Device(config)# ip pim rp-address ip-address
```

Step 3 Select an interface connected to hosts on which PIM can be enabled.

Example:

```
Device(config)# interface interface-type-number
```

Step 4 Add a description for the VLAN.

Example:

```
Device(config-if)# description description
```

Step 5 Enable IP address on an interface and disable proxy ARP.

Example:

```
Device(config-if)# ip address ip-address mask-address
Device(config-if)# no ip proxy-arp
```

Step 6 Enable PIM-SM mode and enable OSPF authentication for a specific interface.

Example:

```
Device(config-if)# ip pim sparse-mode
Device(config-if)# ip ospf authentication message-digest
```

Step 7 Enable message digest key for the OSPF.

Example:

```
Device(config-if)# ip ospf authentication message-digest-key key-number md5 password
```

Step 8 Disable the maintenance operation protocol (MOP) for an interface.

Example:

```
Device(config-if)# no mop enabled
```

Step 9 Disable the task of sending MOP periodic system ID messages and return to the privileged EXEC mode.

Example:

```
Device(config-if)# no mop sysid
Device(config-if)# end
```

Enable multicast traffic with PIM-SSM (CLI)

Enable efficient multicast traffic routing for applications that require directed streaming to specific receivers using commands.

Procedure

Step 1 Enter the global configuration mode and enable IP multicast routing.

Example:

```
Device# configure terminal
Device(config)# ip multicast-routing distributed
```

The **distributed** keyword enables MDS globally.

Step 2 Enable multicast traffic.

Example:

```
Device(config)# wireless multicast ip address
```

For information about the multicast traffic, see Wireless Multicast.

Step 3 Configure PIM-SSM on all network devices.

Example:

```
Device(config)# ip pim ssm default
```

Note
The default SSM range is **232.0.0.0/8**. So, if you do not configure different range, the default SSM range is used.

Step 4 Define SSM range of IP multicast addresses.

Example:

```
Device(config)# ip pim ssm range access-list
```

Step 5 Select an interface connected to hosts on which PIM can be enabled.

Example:

```
Device(config)# interface interface-type-number
```

Step 6 Add a description for the VLAN.

Example:

```
Device(config-if)# description description
```

Step 7 Enable IP address on an interface.

Example:

```
Device(config-if)# ip address ip address mask-address
```

Step 8 Disable proxy ARP.

Example:

```
Device(config-if)# no ip proxy-arp
```

Step 9 Enable PIM-SM on an interface and return to privileged EXEC mode.

Example:

```
Device(config-if)# ip pim sparse-mode
Device(config-if)# end
```

Verify multicast traffic details

To verify if a multicast group supports SSM or not, use this command:

```
Device# show ip mroute
```

```
IP Multicast Routing Table
Flags: D - Dense, S - Sparse, B - Bidir Group, s - SSM Group, C - Connected,
       L - Local, P - Pruned, R - RP-bit set, F - Register flag,
       T - SPT-bit set, J - Join SPT, M - MSDP created entry, E - Extranet,
       X - Proxy Join Timer Running, A - Candidate for MSDP Advertisement,
       U - URD, I - Received Source Specific Host Report,
       Z - Multicast Tunnel, z - MDT-data group sender,
       Y - Joined MDT-data group, y - Sending to MDT-data group,
       G - Received BGP C-Mroute, g - Sent BGP C-Mroute,
       N - Received BGP Shared-Tree Prune, n - BGP C-Mroute suppressed,
       Q - Received BGP S-A Route, q - Sent BGP S-A Route,
       V - RD & Vector, v - Vector, p - PIM Joins on route,
       x - VxLAN group, c - PFP-SA cache created entry,
       * - determined by Assert, # - iif-starg configured on rpf intf,
       e - encap-helper tunnel flag, l - LISP decap ref count contributor
Outgoing interface flags: H - Hardware switched, A - Assert winner, p - PIM Join
                          t - LISP transit group
 Timers: Uptime/Expires
 Interface state: Interface, Next-Hop or VCD, State/Mode
(*, 239.0.0.158), 00:00:07/stopped, RP 15.1.1.2, flags: SJC
  Incoming interface: GigabitEthernet3, RPF nbr 13.1.1.2
  Outgoing interface list:
    Vlan12, Forward/Sparse, 00:00:07/00:02:52, flags:
(17.1.1.1, 239.0.0.158), 00:00:06/00:02:53, flags: JT
  Incoming interface: GigabitEthernet3, RPF nbr 13.1.1.2
  Outgoing interface list:
    Vlan12, Forward/Sparse, 00:00:06/00:02:53, flags:
(*, 231.1.1.1), 02:32:08/stopped, RP 15.1.1.2, flags: SJCF
  Incoming interface: GigabitEthernet3, RPF nbr 13.1.1.2
  Outgoing interface list:
    Vlan12, Forward/Sparse, 00:01:31/00:01:28, flags:
(12.1.0.198, 231.1.1.1), 02:32:08/00:02:53, flags: PFT
  Incoming interface: Vlan12, RPF nbr 0.0.0.0
  Outgoing interface list: Null
(*, 224.0.1.40), 02:32:14/00:02:47, RP 15.1.1.2, flags: SJPL
  Incoming interface: GigabitEthernet3, RPF nbr 13.1.1.2
  Outgoing interface list: Null
```

To verify the IGMP membership details, use this command:

```
Device# show ip igmp membership
Flags: A  - aggregate, T - tracked
       L  - Local, S - static, V - virtual, R - Reported through v3
       I - v3lite, U - Urd, M - SSM (S,G) channel
       1,2,3 - The version of IGMP, the group is in
Channel/Group-Flags:
       / - Filtering entry (Exclude mode (S,G), Include mode (G))
Reporter:
       <mac-or-ip-address> - last reporter if group is not explicitly tracked
       <n>/<m>      - <n> reporter in include mode, <m> reporter in exclude
 Channel/Group                 Reporter        Uptime   Exp.  Flags  Interface
 *,239.255.255.250             11.1.1.4        00:01:38 02:57 2A     Vl12
 *,239.0.0.158                 11.1.1.3        00:00:05 02:54 2A     Vl12
 *,231.1.1.1                   12.1.0.8        00:00:07 02:52 2A     Vl12
 *,224.0.1.40                  13.1.1.1        02:34:15 02:45 2LA    Gi3
```

To verify the IGMP snooping details, use this command:

```
Device# show ip igmp snooping igmpv2-tracking
Client to SGV mappings
----------------------
Client: 11.1.1.3 Port: Ca2
    Group: 239.0.0.158 Vlan: 12 Source: 0.0.0.0 blacklisted: no
```

```
Client: 11.1.1.4 Port: Ca2
    Group: 239.255.255.250 Vlan: 12 Source: 0.0.0.0 blacklisted: no
SGV to Client mappings
----------------------
Group: 239.0.0.158 Source: 0.0.0.0 Vlan: 12
    Client: 11.1.1.3 Port: Ca2 Blacklisted: no
Group: 239.255.255.250 Source: 0.0.0.0 Vlan: 12
    Client: 11.1.1.4 Port: Ca2 Blacklisted: no
```

To verify the multicast group summary details, use this command:

```
Device# show wireless multicast group summary
IPv4 groups
-------------
MGID        Group              Vlan
---------------------------------------
4160        239.255.255.250    12
4161        239.255.255.250    12
 IPv6 groups
-------------
MGID        Group                          Vlan
-------------------------------------------------
```

To verify the IGMP snooping groups, use this command:

```
Device# show ip igmp snooping groups
Vlan      Group                   Type        Version     Port List
------------------------------------------------------------------------------
12        239.0.0.158             igmp        v2          Ca2
12        239.255.255.250         igmp        v2          Ca2
```

To verify the IGMP snooping, use this command:

```
Device# show ip igmp snooping
Global IGMP Snooping configuration:
-------------------------------------------
IGMP snooping                : Enabled
Global PIM Snooping          : Disabled
IGMPv3 snooping (minimal)    : Enabled
Report suppression           : Enabled
TCN solicit query            : Disabled
TCN flood query count        : 2
Robustness variable          : 2
Last member query count      : 2
Last member query interval   : 1000
.
.
.
Vlan 11:
--------
IGMP snooping                        : Enabled
Pim Snooping                         : Disabled
IGMPv2 immediate leave               : Disabled
Multicast router learning mode       : pim-dvmrp
CGMP interoperability mode           : IGMP_ONLY
Robustness variable                  : 2
Last member query count              : 2
Last member query interval           : 1000
Vlan 12:
--------
IGMP snooping                        : Enabled
Pim Snooping                         : Disabled
IGMPv2 immediate leave               : Disabled
```

```
Multicast router learning mode      : pim-dvmrp
CGMP interoperability mode          : IGMP_ONLY
Robustness variable                 : 2
Last member query count             : 2
Last member query interval          : 1000
```

To verify the active streams from any sources, use this command:

```
Device# show ip mroute active
Active IP Multicast Sources - sending >= 4 kbps
Group: 239.255.0.1, (?)
   Source: 192.168.33.32 (?)
   Rate: 10 pps/115 kbps(1sec), 235 kbps(last 23 secs), 87 kbps(life avg)
```

To verify the TTL related issues in the path for the given stream, use this command:

```
Device# show ip traffic | include bad hop count
0 format errors, 0 checksum errors, 1529 bad hop count
```

To verify the RPF failures, use this command:

```
               Device# show ip mroute count | inc RPF failed|Other
Other counts: Total/RPF failed/Other drops(OIF-null, rate-limit etc)
  RP-tree: Forwarding: 0/0/0/0, Other: 2/2/0
  RP-tree: Forwarding: 3/0/74/0, Other: 3/0/0
  Source: 32.32.32.32/32, Forwarding: 218747/2/74/1, Other: 218747/0/0
  RP-tree: Forwarding: 0/0/0/0, Other: 0/0/0
  Source: 9.4.168.10/32, Forwarding: 31/0/146/0, Other: 3841861/0/3841830
```

Network address translation (NAT)

A network address translation is a networking technique that

- allows multiple local IP addresses within a priavte network to be mapped to public IP addresses
- enables devices on private networks to access external (Internet or Cloud) resources, and
- enhances network security by concealing internal addressing schemes from outside networks.

Port address translation (PAT) enables a single IP address to be shared by multiple hosts through the use of IP and port translations.

L3 access on the controller supports only these NAT use cases:

- translating client traffic in the guest network to reach corporate services, such as Cisco ISE, and
- hiding the private IP addresses of clients from outside networks.

These types of NAT are supported:

- Static address translation (static NAT) allows a one-to-one mapping between local and global addresses. The static translation is useful when a host from the inside is accessible from a fixed address from the outside.
- Dynamic address translation (dynamic NAT or PAT) maps between the client subnet and a public global IP address or source port pool.

This can be achieved using these:

- Dynamic NAT without VRF.
- Dynamic NAT with VRF.

Selective NAT support

A selective NAT support is a NAT configuration feature that

- enables only a specified subset of NAT options within a software release, and
- ensures compliance with recent Cisco IOS XE 17.13.1 platform capabilities.

Enable static NAT without VRF (CLI)

Configure static NAT to enable a device on the internal network to communicate with external networks using a fixed public IP address using commands.

Procedure

Step 1 Enter the global configuration mode.

Example:

```
Device# configure terminal
```

Step 2 Specify an interface and enter the interface configuration mode.

Example:

```
Device(config)# interface interface-type number
```

Step 3 Set the IP address for an interface.

Example:

```
Device(config-if)# ip address ip-address mask-address
```

Step 4 Connect the interface to the outside network.

Example:

```
Device(config-if)# ip nat outside
```

Step 5 Exit the interface configuration mode and enter the global configuration mode.

Example:

```
Device(config-if)# end
```

Step 6 Specify a different interface and enter the interface configuration mode.

Example:

```
Device(config)# interface interface-type number
```

Step 7 Set the IP address for an interface.

Example:

```
Device(config-if)# ip address ip-address mask-address
```

Step 8 Mark the interface as connected to the inside.

Example:

```
Device(config-if)# ip nat inside
```

Step 9 Exit the interface configuration mode and enter the global configuration mode. Translate between an inside local address and inside global address.

Example:

```
Device(config-if)# end
Device(config)# ip nat inside source static 10.10.10.100 209.165.200.226
```

Enable static NAT with VRF (CLI)

Establish a static Network Address Translation (NAT) mapping within a specific Virtual Routing and Forwarding (VRF) environment using commands.

Procedure

Step 1 Enter the global configuration mode. Specify an interface and enter the interface configuration mode.

Example:

```
Device# configure terminal
Device(config)# interface interface-type-number
```

Step 2 Activate multiprotocol VRF on an interface.

Example:

```
Device(config-if)# vrf forwarding vrf-name
```

Step 3 Enable IP address on an interface.

Example:

```
Device(config-if)# ip address ip-address mask-address
```

Step 4 Mark the interface as connected to the outside.

Example:

```
Device(config-if)# ip nat outside
```

Step 5 Return to privileged EXEC mode.

Example:

```
Device(config-if)# end
```

Step 6 Specify an interface and enter the interface configuration mode.

Example:

```
Device(config)# interface interface-type-number
```

Step 7 Activate multiprotocol VRF on an interface.

Example:

```
Device(config-if)# vrf forwarding vrf-name
```

Step 8 Enable IP address on an interface. Mark the interface as connected to the inside.

Example:

```
Device(config-if)# ip address ip-address mask-address
Device(config-if)# ip nat inside
```

Step 9 Return to the privileged EXEC mode. Translate between an inside local address and inside global address.

Example:

```
Device(config-if)# end
Device(config)# ip nat inside source static local-ip global-ip vrf vrf_name match-in-vrf
```

Note
The **match-in-vrf** keyword is optional and required when the same VRF is configured in the inside and outside NAT interface. For more information about match-in-vrf, see

https://www.cisco.com/c/en/us/td/docs/ios-xml/ios/ipaddr_nat/configuration/xe-16/nat-xe-16-book/iadnat-match-vrf.html

Enable dynamic NAT without VRF (CLI)

Configure a Cisco device to provide dynamic network address translation (NAT) for internal devices without using VRF using commands.

Procedure

Step 1 Enter the global configuration mode. Specify an interface and enter the interface configuration mode.

Example:

```
Device# configure terminal
Device(config)# interface interface-type number
```

Step 2 Set the IP address for an interface.

Example:

```
Device(config-if)# ip address ip address mask-address
```

Step 3 Mark the interface as connected to the outside.

Example:

```
Device(config-if)# ip nat outside
```

Step 4 Specify a different interface and enter the interface configuration mode.

Example:

```
Device(config)# interface interface-type number
```

Step 5 Set the IP address for an interface.

Example:

```
Device(config-if)# ip address ip address mask-address
```

Step 6 Mark the interface as connected to the inside.

Example:

```
Device(config-if)# ip nat inside
```

Step 7 Define a pool of network addresses for NAT.

Example:

```
Device(config)# ip nat pool name start-ip end-ip {netmask netmask | prefix-length
prefix-length}
```

Step 8 Define a standard access list for the addresses to be translated.

Example:

```
Device(config)# access-list access-list-number permit ip source-address source-wildcard-bits
 host destination-address
```

Note
The **host** keyword is optional for access-list configuration. It depends on the type of ACL you want to configure.

Step 9 Establish dynamic source translation with overloading using the defined access list. Return to the privileged EXEC mode.

Example:

```
Device(config)# ip nat inside source list access-list-number pool name overload

Device(config)# exit
```

Enable dynamic NAT with VRF (CLI)

Enable dynamic Network Address Translation (NAT) on interfaces associated with a specific Virtual Routing and Forwarding (VRF) instance using commands.

Procedure

Step 1 Enter the global configuration mode. Specify an interface and enter the interface configuration mode.

Example:

```
Device# configure terminal

Device(config)# interface interface-type-number
```

Step 2 Activate multiprotocol VRF on an interface. Enable IP address on an interface.

Example:

```
Device(config-if)# vrf forwarding vrf-name

Device(config-if)# ip address ip address mask-address
```

Step 3 Mark the interface as connected to the outside. Return to the privileged EXEC mode.

Example:

```
Device(config-if)# ip nat outside
Device(config-if)# end
```

Step 4 Specify an interface and enter the interface configuration mode. Activate multiprotocol VRF on an interface.

Example:

```
Device(config)# interface interface-type-number
Device(config-if)# vrf forwarding vrf-name
```

Step 5 Enable IP address on an interface. Mark the interface as connected to the inside.

Example:

```
Device(config-if)# ip address ip address mask-address
Device(config-if)# ip nat inside
```

Step 6 Return to privileged EXEC mode. Define a standard IPv4 access list using a name.

Example:

```
Device(config-if)# end
Device(config)# ip access-list standard name
```

The *name* can be a number from one to 99.

Step 7 Specify the forwarded packet. Exit interface configuration mode and return to the global configuration mode.

Example:

```
Device(config-if)# sequence-number permit host-network wildcard-address
Device(config-if)# exit
```

Note
sequence-number refers to the number where the rule should be in the list. Here, lower the sequence number higher the priority for the rule.

Step 8 Define a pool of network addresses for NAT. Establish dynamic source translation with overloading using the defined access list.

Example:

```
Device(config)# ip nat pool name start-ip end-ip {netmask netmask |  prefix-length
prefix-length}
Device(config)# ip nat inside source list access-list-number pool name vrf vrf-name
match-in-vrf overload
```

Note
The **match-in-vrf** keyword is optional and required when the same VRF is configured in the inside and outside NAT interface. For more information about match-in-vrf, see

https://www.cisco.com/c/en/us/td/docs/ios-xml/ios/ipaddr_nat/configuration/xe-16/nat-xe-16-book/iadnat-match-vrf.html

Step 9 Return to the privileged EXEC mode.

Example:

```
Device(config)# end
```

Enable timeout for NAT (CLI)

Adjust the timeout values for NAT translations to optimize device performance using commands.

Procedure

Step 1 Enter the global configuration mode.

Example:

```
Device# configure terminal
```

Step 2 Specify timeouts for NAT translations.

Example:

```
Device(config)# ip nat translation [icmp-timeout | tcp-timeout | timeout | udp-timeout]
number-of-seconds
```

These timeout options are supported:

- **icmp-timeout**: ICMP packets timeout.
- **tcp-timeout**: TCP packets timeout.
- **timeout**: Global timeout for all protocol types.
- **udp-timeout**: UDP packets timeout.

Step 3 Return to the privileged EXEC mode.

Example:

```
Device(config)# end
```

Verify static NAT details

Verify static NAT details without VRF

To verify the static IP NAT statistics without VRF, use this command:

```
Device# show ip nat statistics
Total active translations: 1 (1 static, 0 dynamic; 0 extended)
Outside interfaces:
Vlan62
Inside interfaces:
Vlan55
Hits: 1474 Misses: 0
Reserved port setting disabled provisioned no
Expired translations: 1
Dynamic mappings:
nat-limit statistics:
max entry: max allowed 0, used 0, missed 0
In-to-out drops: 0 Out-to-in drops: 0
Pool stats drop: 0 Mapping stats drop: 0
Port block alloc fail: 0
IP alias add fail: 0
Limit entry add fail: 0
```

To verify the static NAT without VRF on active chassis, use this command:

```
Device# show platform software nat chassis active F0 translation
Pro Inside global Inside local Outside local Outside global
--- 62.1.1.15 155.1.100.1 --- ---
--- 62.1.1.16 155.1.0.4 --- ---
udp 62.1.1.16:33334 155.1.0.4:33334 62.1.1.11:33333 62.1.1.11:33333
udp 62.1.1.16:30000 155.1.0.4:30000 62.1.1.11:30000 62.1.1.11:30000
Total number of translations: 4
```

Verify static NAT details with VRF

To verify the static IP NAT statistics with VRF, use this command:

```
Device# show ip nat statistics
Total active translations: 1 (1 static, 0 dynamic; 0 extended)
Outside interfaces:
Vlan62
Inside interfaces:
Vlan55
Hits: 1474 Misses: 0
Reserved port setting disabled provisioned no
Expired translations: 1
Dynamic mappings:
nat-limit statistics:
max entry: max allowed 0, used 0, missed 0
In-to-out drops: 0 Out-to-in drops: 0
Pool stats drop: 0 Mapping stats drop: 0
Port block alloc fail: 0
IP alias add fail: 0
Limit entry add fail: 0
```

To verify the static NAT with VRF on active chassis, use this command:

```
Device# show platform software nat chassis active F0 translation
Pro Inside global Inside local Outside local Outside global
--- 62.1.1.15 155.1.100.1 --- ---
--- 62.1.1.16 155.1.0.4 --- ---
udp 62.1.1.16:33334 155.1.0.4:33334 62.1.1.11:33333 62.1.1.11:33333
udp 62.1.1.16:30000 155.1.0.4:30000 62.1.1.11:30000 62.1.1.11:30000
Total number of translations: 4
```

Verify dynamic NAT details

Verify dynamic NAT details without VRF

To verify the dynamic IP NAT statistics without VRF, use this command:

```
Device# show ip nat statistics
Total active translations: 1 (0 static, 1 dynamic; 1 extended)
Outside interfaces:
  Vlan62
Inside interfaces:
  Vlan155
Hits: 3  Misses: 1
 Reserved port setting disabled provisioned no
Expired translations: 0
Dynamic mappings:
-- Inside Source
[Id: 2] access-list dest_nat_acl pool test_nat_pool refcount 1
 pool test_nat_pool: id 1, netmask 255.255.255.252
    start 62.1.1.101 end 62.1.1.101
```

```
    type generic, total addresses 1, allocated 1 (100%), misses 0
longest chain in pool: test_nat_pool's addr-hash: 0, average len 0,chains 0/256
nat-limit statistics:
 max entry: max allowed 0, used 0, missed 0
In-to-out drops: 0  Out-to-in drops: 0
Pool stats drop: 0  Mapping stats drop: 0
Port block alloc fail: 0
IP alias add fail: 0
Limit entry add fail: 0
```

To verify the dynamic NAT without VRF on active chassis, use this command:

```
Device# show platform software nat chassis active F0 translation
Pro  Inside global         Inside local          Outside local         Outside global
udp  62.1.1.101:30000      155.1.100.1:30000     62.1.1.11:30000       62.1.1.11:30000
Total number of translations: 1
```

Verify dynamic NAT details with VRF

To verify the dynamic IP NAT statistics with VRF, use this command:

```
Device# show ip nat statistics
Total active translations: 1 (0 static, 1 dynamic; 1 extended)
Outside interfaces:
  Vlan62
Inside interfaces:
  Vlan155
Hits: 3  Misses: 1
 Reserved port setting disabled provisioned no
Expired translations: 0
Dynamic mappings:
-- Inside Source
[Id: 2] access-list dest_nat_acl pool test_nat_pool refcount 1
 pool test_nat_pool: id 1, netmask 255.255.255.252
    start 62.1.1.101 end 62.1.1.101
    type generic, total addresses 1, allocated 1 (100%), misses 0
longest chain in pool: test_nat_pool's addr-hash: 0, average len 0,chains 0/256
nat-limit statistics:
 max entry: max allowed 0, used 0, missed 0
In-to-out drops: 0  Out-to-in drops: 0
Pool stats drop: 0  Mapping stats drop: 0
Port block alloc fail: 0
IP alias add fail: 0
Limit entry add fail: 0
```

To verify the dynamic NAT with VRF on active chassis, use this command:

```
Device# show platform software nat chassis active F0 translation
Pro  Inside global         Inside local          Outside local         Outside global
udp  62.1.1.101:30000      155.1.100.1:30000     62.1.1.11:30000       62.1.1.11:30000
Total number of translations: 1
```

Verify NAT details

To verify the NAT datapath pool details, use this command:

```
Device# show platform hardware chassis active qfp feature nat datapath pool
pool_id 1 type 1 addroute 0 mask 0xfffffffc allocated 0 misses 0 rotary idx 0x0 ahash sz 4
 size 1 max_pat_hash_size 1 next 0x0 hash_index 0x32, hilo ports 0x0 pool mem 0xde480010
flags 0x1 pool_name: test_nat_pool pat_wl 0 no_ports_wl 0 num_maps 1 num_overload_maps 1
vrf 0x0 port_used tcp 0 udp 0
Conf block info
start 62.1.1.102 end 62.1.1.102 flags 0x0 next 0x0 prev 0x0
```

```
TCP PAT block info
UDP PAT block info
ICMP PAT block info
GRE PAT block info
Alloced addr info
```

To verify the NAT datapath statistics, use this command:

```
Device# show platform hardware chassis active qfp feature nat datapath stats
Counter Value
-------------------------------------------------------------------------
number_of_session 0
udp 0
tcp 0
icmp 0
non_extended 0
statics 0
static_net 0
entry_timeouts 0
hits 0
misses 0
cgn_dest_log_timeouts 0
ipv4_nat_alg_bind_pkts 0
ipv4_nat_alg_sd_not_found 0
ipv4_nat_alg_sd_tail_not_found 0
ipv4_nat_rx_pkt 2043
ipv4_nat_tx_pkt 122169
ipv4_nat_flowdb_hits 0
ipv4_nat_stick_rx_pkts 0
ipv4_nat_stick_i2o_pkts 0
ipv4_nat_stick_o2i_pkts 0
ipv4_nat_stick_forus_hits_pkts 0
ipv4_nat_stick_hit_sb 0
ipv4_nat_stick_ha_divert_pkts 0
ipv4_nat_stick_ha_ar_pkts 0
ipv4_nat_stick_ha_tcp_fin 0
ipv4_nat_stick_ha_failed_pkts 0
ipv4_nat_non_natted_in2out_pkts 122165
ipv4_nat_non_nated_out2in_pkts 0
ipv4_nat_bypass_pkts 0
ipv4_nat_unmarked_pkts 0
ipv4_nat_res_port_in2out_pkts 0
ipv4_nat_res_port_out2in_pkts 0
ipv4_nat_ipc_retry_fail 0
ipv4_nat_cfg_rcvd 2
ipv4_nat_cfg_rsp 2
```

To clear the NAT details, use this commands:

```
clear platform software nat chassis active F0 translation forced
clear ip nat statistics
```

Verify NAT timeout details

To verify the NAT timeout details, use this command:

```
Device# show platform software nat chassis active r0 timeout
Dump NAT timeout config
  Type: generic, Timeout (sec): 86400, Enabled: Yes
  Type: tcp, Timeout (sec): 86400, Enabled: Yes
  Type: tcp-pptp, Timeout (sec): 86400, Enabled: Yes
  Type: udp, Timeout (sec): 60, Enabled: Yes
```

```
Type: tcp-fin-reset, Timeout (sec): 60, Enabled: Yes
Type: tcp-syn, Timeout (sec): 60, Enabled: Yes
Type: dns, Timeout (sec): 60, Enabled: Yes
Type: icmp, Timeout (sec): 60, Enabled: Yes
Type: skinny, Timeout (sec): 60, Enabled: Yes
Type: icmp-error, Timeout (sec): 60, Enabled: Yes
Type: esp, Timeout (sec): 300, Enabled: Yes
Type: rtmap, Timeout (sec): 3600, Enabled: Yes
```