Operate and Troubleshoot DHCP Snooping on Catalyst 9000 Switches

Contents

Introduction

This document describes how to operate and troubleshoot DHCP Snooping on Catalyst 9000 series switches.

Prerequisites

Requirements

Cisco recommends that you have knowledge of these topics:

- Catalyst 9000 Series Switches Architecture
- Cisco IOS® XE Software Architecture

Components Used

The information in this document is based on these software and hardware versions:

- C9200

- C9300
- C9400
- C9500
- C9600

Cisco IOS® XE 16.12.X

The information in this document was created from the devices in a specific lab environment. All of the devices used in this document started with a cleared (default) configuration. If your network is live, ensure that you understand the potential impact of any command.

Note: Consult the appropriate configuration guide for the commands that are used to enable these features on other Cisco platforms.

Background Information

DHCP Snooping

Dynamic Host Configuration Protocol (DHCP) Snooping is a security feature used to check DHCP traffic to block any malicious DHCP packet. It acts as a firewall between untrusted user ports and DHCP server ports on the network to prevent malicious DHCP servers in the network as this can cause a denial of service.

DHCP Snooping Operation

DHCP Snooping works with the concept of trusted and untrusted interfaces. Through the path of the DHCP traffic, the switch verifies the DHCP packets received on the interfaces and keep a track of the expected DHCP Server packets (OFFER & ACK) over trusted interfaces. In other words, untrusted interfaces block DHCP Server packets.

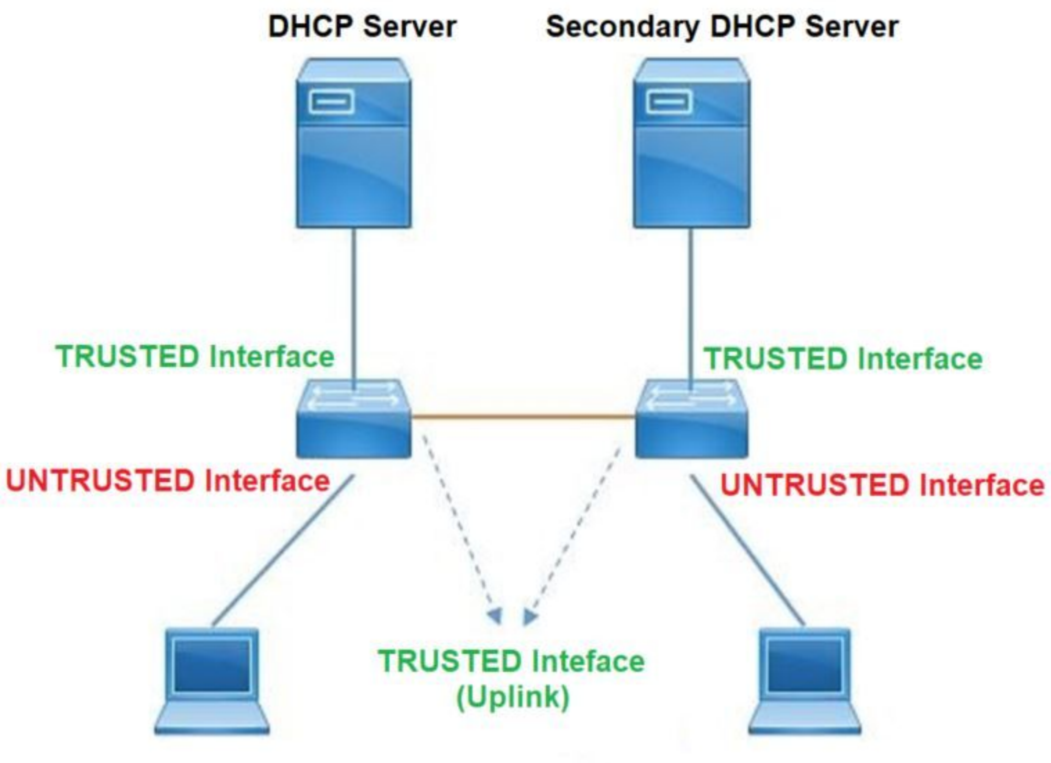
DHCP Packets are blocked on untrusted interfaces.

- A packet from a DHCP server, such as a DHCPOFFER, DHCPACK, DHCPNAK, or DHCPLEASEQUERY packet, is received from outside the network or firewall. This prevents a rogue DHCP server from an attack to the network on untrusted ports.
- A packet received on an untrusted interface, and the source MAC address and the DHCP client hardware address do not match. This prevents spoof of DHCP packets from a rogue client that could create a denial of service attack on a DHCP server.
- A DHCPRELEASE or DHCPDECLINE broadcast message that has a MAC address in the DHCP snooping binding database, but the interface information in the binding database does not match the interface on which the message was received. This prevents denial of service attacks on clients.
- A DHCP packet forwarded by a DHCP relay agent that includes a relay-agent IP address that is not 0.0.0.0, or the relay agent forwards a packet that includes option-82 information to an untrusted port. This prevents spoof of relay agent information on the network.

The switch where you configure DHCP Snooping builds a DHCP Snooping table or DHCP binding database. This table is used to keep a track of the IP addresses assigned from a legitimate DHCP server. The binding database is also used by other IOS security features such as Dynamic ARP Inspection and IP Source Guard.

Note: To allow DHCP Snooping to work correctly, ensure you trust all the uplink ports toward reach the DHCP server and untrust the end-user ports.

Topology



Configure

Global Configuration

<#root>

```
1. Enable DHCP snooping globally on the switch
    switch(config)#
```

ip dhcp snooping

```
2. Designate ports that forward traffic toward the DHCP server as trusted
    switch(config-if)#
```

ip dhcp snooping trust

(Additional verification)

```
    - List uplink ports according to the topology, ensure all the uplink ports toward the DHCP server a
```

trusted

```
    - List the port where the Legitimate DHCP Server is connected (include any Secondary DHCP Server)
```

- Ensure that no other port is configured as trusted

3. Configure DHCP rate limiting on each untrusted port (Optional)

```
    switch(config-if)#

ip dhcp snooping limit rate 10 << ----- 10 packets per second (pps)
```

4. Enable DHCP snooping in specific VLAN

```
    switch(config)#

ip dhcp snooping vlan 10


<< ----- Allow the switch to snoop the traffic for that specific VLAN
```

5. Enable the insertion and removal of option-82 information DHCP packets

```
    switch(config)#

ip dhcp snooping information option


<-- Enable insertion of option 82


    switch(config)#

no ip dhcp snooping information option


<-- Disable insertion of option 82
```

### Example ###

Legitimate DHCP Server Interface and Secondary DHCP Server, if available

Server Interface

```
interface FortyGigabitEthernet1/0/5
switchport mode access
switchport mode access vlan 11

ip dhcp snooping trust


end
```

Uplink interface

```
interface FortyGigabitEthernet1/0/10
switchport mode trunk

ip dhcp snooping trust
```

```
end


User Interface



 << ----- All interfaces are UNTRUSTED by default


interface FortyGigabitEthernet1/0/2
 switchport access vlan 10
 switchport mode access


ip dhcp snooping limit rate 10



<< ----- Optional


end
```

Note: To allow option-82 packets, you must enable **ip dhcp snooping information option allow-untrusted**.

Verify

Confirm if DHCP Snooping is enabled on the desired VLAN and ensure trusted and untrusted interfaces are well listed. If there is a rate configured, ensure it is listed as well.

<#root>

```
switch#show ip dhcp snooping


Switch DHCP snooping is

enabled


Switch DHCP gleaning is disabled
DHCP snooping is configured on following VLANs:

10-11


DHCP

snooping is operational on following VLANs

:

<<---- Configured and operational on Vlan 10 & 11



10-11
```

```
DHCP snooping is configured on the following L3 Interfaces:


Insertion of option 82 is disabled



<<---- Option 82 can not be added to DHCP packet


   circuit-id default format: vlan-mod-port
   remote-id: 00a3.d144.1a80 (MAC)
Option 82 on untrusted port is not allowed
Verification of hwaddr field is enabled
Verification of giaddr field is enabled
DHCP snooping trust/rate is configured on the following Interfaces:

Interface

  Trusted

     Allow option    Rate limit (pps)
-----------------------        -------     ------------    ----------------
FortyGigabitEthernet1/0/2

no

          no             10

<<--- Trust is NOT set on this interface


Custom circuit-ids:
FortyGigabitEthernet1/0/10

 yes

         yes            unlimited

<<--- Trust is set on this interface


Custom circuit-ids:
```

Once users receive an IP by DHCP, they are listed in this output.

- DHCP Snooping removes the entry in the database when the IP address lease expires or the switch receives a DHCPRELEASE message from the host.
- Ensure information listed for the end-user MAC address is correct.

<#root>

```
c9500#show ip dhcp snooping binding


MacAddress          IpAddress        Lease(sec)  Type          VLAN  Interface
------------------ --------------- ---------- ------------- ---- --------------------
00:A3:D1:44:20:46  10.0.0.3

85556
```

```
 dhcp-snooping 10   FortyGigabitEthernet1/0/2
Total number of bindings: 1
```

This table lists the various commands that can be used to monitor DHCP Snooping information.

Command	Purpose
show ip dhcp snooping binding **show ip dhcp snooping binding** [IP-address] [MAC-address] [interface ethernet slot/port] [vlan-id]	Displays only the dynamically configured bindings in the DHCP snooping binding database, also referred to as a binding table. - Binding entry IP address - Binding entry Mac address - Binding entry input interface - Binding entry VLAN
show ip dhcp snooping database	Displays the DHCP snooping binding database status and statistics.
show ip dhcp snooping statistics	Displays the DHCP snooping statistics in summary or detail form.
show ip source binding	Displays the dynamically and statically configured bindings.
show interface vlan xyz **show buffer input-interface Vlan xyz dump**	DHCP packet is sent to relay agent configured in the client VLAN via client VLAN SVI. If input queue shows drop or reach maximum limit, it is likely the DHCP packet from client was dropped and was not able to reach relay agent configured. **Note**: Ensure drops are not seen in the input queue. switch#**show int vlan 670** Load for five secs: 13%/0%; one minute: 10%; five minutes: 10% Time source is NTP, 18:39:52.476 UTC Thu Sep 10 2020 VLAN670 is up, line protocol is up, Autostate Enabled Hardware is Ethernet SVI, address is 00fd.227a.5920 (bia 00fd.227a.5920) Description: ion_media_client Internet address is 10.27.49.254/23 MTU 1500 bytes, BW 1000000 Kbit/sec, DLY 10 usec, reliability 255/255, txload 1/255, rxload 1/255 Encapsulation ARPA, loopback not set Keepalive not supported ARP type: ARPA, ARP Timeout 04:00:00 Last input 03:01:29, output 00:00:02, output hang never Last clearing of "show interface" counters never Input queue: 375/375/4020251/0 (size/max/drops/flushes); Total

	output drops: 0 <-- 375 packets in input the queue / 4020251 have been dropped

Troubleshoot

Troubleshoot Software

Verify what the switch receives. These packets are processed at the CPU control-plane, so ensure you see the all packets in the inject and punt direction, and confirm if the information is correct.

⚠ **Caution**: Use the debug commands with caution. Please be aware many debug commands have impact on the live network and are only recommended to use in a lab environment when the issue is reproduced.

The Conditional Debug feature allows you to selectively enable debugs and logs for specific features based on a set of conditions you define. This is useful to contain debug information only for specific hosts or traffic.

A condition refers to a feature or identity, where identity could be an interface, IP Address, or a MAC address and so on..

How to enable the Conditional Debug feature for debugging packets and events when troubleshooting DHCP Snooping.

Command	Purpose
debug condition mac <mac-address> Example: switch#**debug condition mac bc16.6509.3314**	Configures conditional debugging for the MAC Address specified.
debug condition vlan <VLAN Id> Example: switch#**debug condition vlan 10**	Configures conditional debugging for the VLAN specified.
debug condition interface <interface> Example: switch#**debug condition interface twentyFiveGigE 1/0/8**	Configures conditional debugging for the interface specified.

To debug DHCP Snooping, use the commands shown in this table.

Command	Purpose

debug dhcp [detail | oper | redundancy]	**detail** DHCP packet content **oper** DHCP internal OPER **redundancy** DHCP client redundancy support
debug ip dhcp server packet detail	Decode message receptions and transmission in detail.
debug ip dhcp server events	Report address assignments, lease expiration, and so on.
debug ip dhcp snooping agent	Debug DHCP snooping database read and write.
debug ip dhcp snooping event	Debug event between each components.
debug ip dhcp snooping packet	Debug DHCP packet in dhcp snooping module.

This is a partial sample output of the**debug ip dhcp snooping** command.

<#root>

```
Apr 14 16:16:46.835: DHCP_SNOOPING: process new DHCP packet,

message type: DHCPDISCOVER, input interface: Fo1/0/2

, MAC da: ffff.ffff.ffff, MAC

sa: 00a3.d144.2046,

 IP da: 255.255.255.255, IP sa: 0.0.0.0, DHCP ciaddr: 0.0.0.0, DHCP yiaddr: 0.0.0.0, DHCP siaddr: 0.0.0
Apr 14 16:16:46.835: DHCP_SNOOPING: bridge packet get invalid mat entry: FFFF.FFFF.FFFF, packet is floo

Apr 14 16:16:48.837: DHCP_SNOOPING:

received new DHCP packet from input interface (FortyGigabitEthernet1/0/10)


Apr 14 16:16:48.837: DHCP_SNOOPING:

process new DHCP packet, message type: DHCPOFFER, input interface: Fo1/0/10,

MAC da: ffff.ffff.ffff, MAC

sa: 701f.539a.fe46,

 IP da: 255.255.255.255, IP sa: 10.0.0.1, DHCP ciaddr: 0.0.0.0, DHCP yiaddr: 10.0.0.5, DHCP siaddr: 0.0
Apr 14 16:16:48.837: platform lookup dest vlan for input_if: FortyGigabitEthernet1/0/10, is NOT tunnel,
Apr 14 16:16:48.837: DHCP_SNOOPING: direct forward dhcp replyto output port: FortyGigabitEthernet1/0/2.
Apr 14 16:16:48.838: DHCP_SNOOPING: received new DHCP packet from input interface (FortyGigabitEthernet
Apr 14 16:16:48.838: Performing rate limit check

Apr 14 16:16:48.838: DHCP_SNOOPING: process new DHCP packet,

message type: DHCPREQUEST, input interface: Fo1/0/2,

 MAC da: ffff.ffff.ffff, MAC
```

```
sa: 00a3.d144.2046,

 IP da: 255.255.255.255, IP sa: 0.0.0.0, DHCP ciaddr: 0.0.0.0, DHCP yiaddr: 0.0.0.0, DHCP siaddr: 0.0.0
Apr 14 16:16:48.838: DHCP_SNOOPING: bridge packet get invalid mat entry: FFFF.FFFF.FFFF, packet is floo
Apr 14 16:16:48.839: DHCP_SNOOPING: received new DHCP packet from input interface (FortyGigabitEthernet

Apr 14 16:16:48.840: DHCP_SNOOPING: process new DHCP packet,

message type: DHCPACK, input interface: Fo1/0/10,

 MAC da: ffff.ffff.ffff, MAC

sa: 701f.539a.fe46,

 IP da: 255.255.255.255, IP

sa: 10.0.0.1,

 DHCP ciaddr: 0.0.0.0, DHCP yiaddr: 10.0.0.5, DHCP siaddr: 0.0.0.0, DHCP giaddr: 0.0.0.0, DHCP chaddr:
Apr 14 16:16:48.840: DHCP_SNOOPING: add binding on port FortyGigabitEthernet1/0/2 ckt_id 0 FortyGigabit
Apr 14 16:16:48.840: DHCP_SNOOPING: added entry to table (index 331)

Apr 14 16:16:48.840:

DHCP_SNOOPING: dump binding entry: Mac=00:A3:D1:44:20:46 Ip=10.0.0.5

 Lease=86400 Type=dhcp-snooping

Vlan=10 If=FortyGigabitEthernet1/0/2


Apr 14 16:16:48.840: No entry found for mac(00a3.d144.2046) vlan(10) FortyGigabitEthernet1/0/2
Apr 14 16:16:48.840: host tracking not found for update add dynamic (10.0.0.5, 0.0.0.0, 00a3.d144.2046)
Apr 14 16:16:48.840: platform lookup dest vlan for input_if: FortyGigabitEthernet1/0/10, is NOT tunnel,
Apr 14 16:16:48.840: DHCP_SNOOPING: direct forward dhcp replyto output port: FortyGigabitEthernet1/0/2.
```

To debug DHCP Snooping events, use these steps:

Caution: Use the debug commands with caution. Please be aware many debug commands have impact on live network, and are only recommended to use in a lab environment when the issue is reproduced.

Summary Steps

1. **enable**
2. **debug platform condition mac {mac-address }**
3. **debug platform condition start**
4. **show platform condition** OR **show debug**
5. **debug platform condition stop**
6. **show platform software trace message ios R0 reverse | include DHCP**
7. **clear platform condition all**

Detailed Steps

	Command or Action	Purpose
Step 1	**enable**	Enables privileged EXEC mode.

	Command or Action	Purpose
	Example: switch#**enable**	- Enter your password if prompted.
Step 2	**debug platform condition mac** {mac-address} Example: switch#**debug platform condition mac 0001.6509.3314**	Configures conditional debugging for the MAC Address specified.
Step 3	**debug platform condition start** Example: switch#**debug platform condition start**	Starts conditional debugging (this can start radioactive tracing if there is a match on one of the conditions).
Step 4	**show platform condition** OR **show debug** Example: switch#**show platform condition** switch#**show debug**	Displays the current conditions set.
Step 5	**debug platform condition stop** Example: switch#**debug platform condition stop**	Stops conditional debugging (this can stop radioactive tracing).
Step 6	**show platform software trace message ios R0 reverse | include DHCP** Example: switch#**show platform software trace message ios R0 reverse | include DHCP**	Displays HP logs merged from the latest trace file.
Step 7	**clear platform condition all** Example: switch# **clear platform condition all**	Clears all conditions.

This is a partial sample output example of the **debug platform dhcp-snoop all** command.

<#root>

```
debug platform dhcp-snoop all
```

```
DHCP Server UDP port
```

(67)

```
DHCP Client UDP port
```

(68)

RELEASE

```
Apr 14 16:44:18.629: pak->vlan_id = 10
Apr 14 16:44:18.629: dhcp packet src_ip(10.0.0.6) dest_ip(10.0.0.1) src_udp(68) dest_udp(67) src_mac(00
Apr 14 16:44:18.629: ngwc_dhcpsn_process_pak(305): Packet handedover to SISF on vlan 10
Apr 14 16:44:18.629: dhcp pkt processing routine is called for pak with SMAC = 00a3.d144.2046{mac} and
```

DISCOVER

```
Apr 14 16:44:24.637: dhcp packet src_ip(0.0.0.0) dest_ip(255.255.255.255) src_udp(68) dest_udp(67) src_
Apr 14 16:44:24.637: ngwc_dhcpsn_process_pak(305): Packet handedover to SISF on vlan 10
Apr 14 16:44:24.637: dhcp pkt processing routine is called for pak with SMAC = 00a3.d144.2046{mac} and
Apr 14 16:44:24.637: sending dhcp packet out after processing with SMAC = 00a3.d144.2046{mac} and SRC_A
Apr 14 16:44:24.638: pak->vlan_id = 10
```

OFFER

```
Apr 14 16:44:24.638: dhcp packet src_ip(10.0.0.1) dest_ip(255.255.255.255) src_udp(67) dest_udp(68) src_
Apr 14 16:44:24.638: ngwc_dhcpsn_process_pak(305): Packet handedover to SISF on vlan 10
Apr 14 16:44:24.638: dhcp pkt processing routine is called for pak with SMAC = 701f.539a.fe46{mac}  and
```

REQUEST

```
Apr 14 16:44:24.638: ngwc_dhcpsn_process_pak(284): Packet handedover to SISF on vlan 10
c9500#dhcp pkt processing routine is called for pak with SMAC = 0a3.d144.2046{mac} and SRC_ADDR = 0.0.0
```

ACK

```
Apr 14 16:44:24.640:  dhcp paket src_ip(10.10.10.1) dest_ip(255.255.255.255) src_udp(67) dest_udp(68) s
Apr 14 16:44:24.640: ngwc_dhcpsn_process_pak(284): Packet handedover to SISF on vlan 10dhcp pkt process
```

This table lists the various commands that can be used to debug DHCP Snooping in platform.

⚠ **Caution**: Use the debug commands with caution. Please be aware many debug commands have an impact on live network, and only are recommended to use in a lab environment when the issue is reproduced.

Command	Purpose
switch#**debug platform dhcp-snoop [all | packet | pd-shim]**	**all** NGWC DHCP Snooping **packet** NGWC DHCP Snooping Packet Debug Info **pd-shim** NGWC DHCP Snooping IOS Shim Debug Info
switch#**debug platform software infrastructure punt dhcp-snoop**	Packets that are received on the FP that are punted to the control plane).
switch#**debug platform software infrastructure inject**	Packets that are injected into the FP from the control plane.

Troubleshoot Punt/Path Traffic (CPU)

Verify from FED perspective what traffic is received in each CPU queue (DHCP Snooping is a type of traffic that is processed by the control-plane).

- When the traffic comes into the switch, it is sent to CPU in the PUNT direction and is sent to the DHCP snoop queue.
- Once the traffic is processed by the switch, the traffic leaves via the INJECT direction. DHCP OFFER and ACK packets fall into theL2 control/legacy queue.

<#root>

```
c9500#show platform software fed switch active punt cause summary


Statistics for all causes

Cause   Cause Info                   Rcvd         Dropped
------------------------------------------------------------------------------
21      RP<->QFP keepalive           8533         0

79      dhcp snoop                   71           0      <<---- If drop counter increases, there can be a


96      Layer2 control protocols     45662        0
109     snoop packets                100          0
------------------------------------------------------------------------------


c9500#show platform software fed sw active inject cause summary


Statistics for all causes

Cause Cause Info                   Rcvd           Dropped
------------------------------------------------------------------------------

1     L2 control/legacy
```

```
        128354         0     <<---- dropped counter must NOT increase


2     QFP destination lookup      18             0
5     QFP <->RP keepalive         8585           0
12    ARP request or response     68             0
25    Layer2 frame to BD          81             0
------------------------------------------------------------------------------
```

You can use this command to confirm the traffic that is punted to the CPU, and verify if DHCP Snooping drops traffic.

<#root>

```
c9500#

show platform software fed switch active punt cpuq rates

Punt Rate CPU Q Statistics

Packets per second averaged over 10 seconds, 1 min and 5 mins
=====================================================================================
Q  |               Queue              |  Rx   |  Rx    |  Rx   |  Drop   |  Drop  |  Drop
no |               Name               | 10s   |  1min  |  5min |  10s    |  1min  |  5min
=====================================================================================
0 CPU_Q_DOT1X_AUTH                       0        0       0        0        0        0
1 CPU_Q_L2_CONTROL                       0        0       0        0        0        0
2 CPU_Q_FORUS_TRAFFIC                    0        0       0        0        0        0
3 CPU_Q_ICMP_GEN                         0        0       0        0        0        0
4 CPU_Q_ROUTING_CONTROL                  0        0       0        0        0        0
5 CPU_Q_FORUS_ADDR_RESOLUTION            0        0       0        0        0        0
6 CPU_Q_ICMP_REDIRECT                    0        0       0        0        0        0
7 CPU_Q_INTER_FED_TRAFFIC                0        0       0        0        0        0
8 CPU_Q_L2LVX_CONTROL_PKT                0        0       0        0        0        0
9 CPU_Q_EWLC_CONTROL                     0        0       0        0        0        0
10 CPU_Q_EWLC_DATA                       0        0       0        0        0        0
11 CPU_Q_L2LVX_DATA_PKT                  0        0       0        0        0        0
12 CPU_Q_BROADCAST                       0        0       0        0        0        0
13 CPU_Q_LEARNING_CACHE_OVFL             0        0       0        0        0        0
14 CPU_Q_SW_FORWARDING                   0        0       0        0        0        0
15 CPU_Q_TOPOLOGY_CONTROL                2        2       2        0        0        0
16 CPU_Q_PROTO_SNOOPING                  0        0       0        0        0        0

17 CPU_Q_DHCP_SNOOPING

0        0      0        0        0

        0    <<---- drop counter must NOT increase

18 CPU_Q_TRANSIT_TRAFFIC                 0        0       0        0        0        0
19 CPU_Q_RPF_FAILED                      0        0       0        0        0        0
20 CPU_Q_MCAST_END_STATION_SERVICE       0        0       0        0        0        0
21 CPU_Q_LOGGING                         0        0       0        0        0        0
22 CPU_Q_PUNT_WEBAUTH                    0        0       0        0        0        0
23 CPU_Q_HIGH_RATE_APP                   0        0       0        0        0        0
24 CPU_Q_EXCEPTION                       0        0       0        0        0        0
25 CPU_Q_SYSTEM_CRITICAL                 8        8       8        0        0        0
26 CPU_Q_NFL_SAMPLED_DATA                0        0       0        0        0        0
27 CPU_Q_LOW_LATENCY                     0        0       0        0        0        0
```

```
28 CPU_Q_EGR_EXCEPTION                  0        0      0        0        0        0
29 CPU_Q_FSS                            0        0      0        0        0        0
30 CPU_Q_MCAST_DATA                     0        0      0        0        0        0
31 CPU_Q_GOLD_PKT                       0        0      0        0        0        0

-------------------------------------------------------------------------------------
```

Troubleshoot Hardware

Forwarding Engine Driver (FED)

FED is the driver that programs the ASIC. FED commands are used to verify that hardware and software states match.

Obtain the DI_Handle value.

- The DI handle refers to the destination index for a specific port.

<#root>

```
c9500#show platform software fed switch active security-fed dhcp-snoop vlan vlan-id 10


Platform Security DHCP Snooping Vlan Information


Value of Snooping DI handle

 is::

0x7F7FAC23E438   <<---- If DHCP Snooping is not enabled the hardware handle can not be present




                                                       Port             Trust Mode
----------------------------------------------------------------------------------
                                              FortyGigabitEthernet1/0/10

 trust <<---- Ensure TRUSTED ports are listed
```

Check the IFM mapping to determine the ASIC and Core of the ports.

- IFM is an internal interface index mapped to a specific port/core/asic.

<#root>

```
c9500#show platform software fed switch active ifm mappings


Interface                  IF_ID  Inst Asic Core Port SubPort Mac Cntx LPN GPN Type Active
FortyGigabitEthernet1/0/10
```

```
0xa

    3

1    1

    1    0       4    4   2   2   NIF  Y
```

Use the DI_Handle to get the hardware index.

<#root>

```
c9500#show platform hardware fed switch active fwd-asic abstraction print-resource-handle 0x7F7FAC23E438

 0
Handle:0x7f7fac23e438 Res-Type:ASIC_RSC_DI Res-Switch-Num:255 Asic-Num:255 Feature-ID:AL_FID_DHCPSNOOPIN
priv_ri/priv_si Handle: (nil)Hardware Indices/Handles:

index0:0x5f03

 mtu_index/l3u_ri_index0:0x0 index1:0x5f03 mtu_index/l3u_ri_index1:0x0 index2:0x5f03 mtu_index/l3u_ri_in
<SNIP>

<-- Index is 0x5f03
```

Convert from hexadecimal the index value 0x5f03 to decimal.

0x5f03 = 24323

Use this index value in decimal, and the ASIC and Core values in this command to see what flags are set for the port.

<#root>

```
c9500#show platform hardware fed switch 1 fwd-asic regi read register-name SifDestinationIndexTable-2432

 asic

1

 core

1


For asic 1 core 1

Module 0 - SifDestinationIndexTable[0][

24323

]

<-- the decimal hardware index matches 0x5f03 = 24323



copySegment0  :

0x1 <<----  If you find this as 0x0, means that the traffic is not forwarded out of this port. (refer to
 CSCvi39202)copySegment1  : 0x1
dpuSegment0   : 0x0
dpuSegment1   : 0x0
ecUnicast     : 0x0
etherChannel0 : 0x0
etherChannel1 : 0x0
hashPtr1      : 0x0
stripSegment  : 0x0
```

Ensure DHCP Snooping is enabled for the specific VLAN.

<#root>

```
c9500#show platform software fed switch 1 vlan 10


VLAN Fed Information


Vlan Id IF Id               LE Handle           STP Handle          L3 IF Handle        SVI IF
-------------------------------------------------------------------------------------------------------
10      0x0000000000420011

0x00007f7fac235fa8

 0x00007f7fac236798  0x0000000000000000  0x0000000000000000  15



c9500#

show platform hardware fed switch active fwd-asic abstraction print-resource-handle



0x00007f7fac235fa8 1  <<---- Last number might be 1 or 0, 1 means detailed, 0 means brief output


Handle:0x7f7fac235fa8 Res-Type:ASIC_RSC_VLAN_LE Res-Switch-Num:255 Asic-Num:255 Feature-ID:AL_FID_L2 Lk
priv_ri/priv_si Handle: (nil)Hardware Indices/Handles: index0:0xf mtu_index/l3u_ri_index0:0x0 sm handle
Cookie length: 56
00 00 00 00 00 00 00 00 0a 00 00 00 00 00 00 00 00 00 00 00 00 00 00 00 00 00 00 00 00 00 00 00 00 0


Detailed Resource Information (ASIC_INSTANCE# 0)
----------------------------------------

LEAD_VLAN_IGMP_MLD_SNOOPING_ENABLED_IPV4 value 1 Pass   <<---- Verify the highlighted values, if any are


LEAD_VLAN_IGMP_MLD_SNOOPING_ENABLED_IPV6 value 0 Pass

LEAD_VLAN_ARP_OR_ND_SNOOPING_ENABLED_IPV4 value 1 Pass


LEAD_VLAN_ARP_OR_ND_SNOOPING_ENABLED_IPV6 value 1 Pass
LEAD_VLAN_BLOCK_L2_LEARN value 0 Pass
LEAD_VLAN_CONTENT_MATCHING_ENABLED value 0 Pass
LEAD_VLAN_DEST_MOD_INDEX_TVLAN_LE value 0 Pass

LEAD_VLAN_DHCP_SNOOPING_ENABLED_IPV4 value 1 Pass
```

```
LEAD_VLAN_DHCP_SNOOPING_ENABLED_IPV6 value 1 Pass
LEAD_VLAN_ENABLE_SECURE_VLAN_LEARNING_IPV4 value 0 Pass
LEAD_VLAN_ENABLE_SECURE_VLAN_LEARNING_IPV6 value 0 Pass
LEAD_VLAN_EPOCH value 0 Pass
LEAD_VLAN_L2_PROCESSING_STP_TCN value 0 Pass
LEAD_VLAN_L2FORWARD_IPV4_MULTICAST_PKT value 0 Pass
LEAD_VLAN_L2FORWARD_IPV6_MULTICAST_PKT value 0 Pass
LEAD_VLAN_L3_IF_LE_INDEX_PRIO value 0 Pass
LEAD_VLAN_L3IF_LE_INDEX value 0 Pass
LEAD_VLAN_LOOKUP_VLAN value 15 Pass
LEAD_VLAN_MCAST_LOOKUP_VLAN value 15 Pass
LEAD_VLAN_RIET_OFFSET value 4095 Pass
LEAD_VLAN_SNOOPING_FLOODING_ENABLED_IGMP_OR_MLD_IPV4 value 1 Pass
LEAD_VLAN_SNOOPING_FLOODING_ENABLED_IGMP_OR_MLD_IPV6 value 1 Pass
LEAD_VLAN_SNOOPING_PROCESSING_STP_TCN_IGMP_OR_MLD_IPV4 value 0 Pass
LEAD_VLAN_SNOOPING_PROCESSING_STP_TCN_IGMP_OR_MLD_IPV6 value 0 Pass
LEAD_VLAN_VLAN_CLIENT_LABEL value 0 Pass
LEAD_VLAN_VLAN_CONFIG value 0 Pass
LEAD_VLAN_VLAN_FLOOD_ENABLED value 0 Pass
LEAD_VLAN_VLAN_ID_VALID value 1 Pass
LEAD_VLAN_VLAN_LOAD_BALANCE_GROUP value 15 Pass
LEAD_VLAN_VLAN_ROLE value 2 Pass
LEAD_VLAN_VLAN_FLOOD_MODE_BITS value 3 Pass
LEAD_VLAN_LVX_VLAN value 0 Pass
LEAD_VLAN_EGRESS_DEJAVU_CANON value 0 Pass
LEAD_VLAN_EGRESS_INGRESS_VLAN_MODE value 0 Pass
LEAD_VLAN_EGRESS_LOOKUP_VLAN value 0 Pass
LEAD_VLAN_EGRESS_LVX_VLAN value 0 Pass
LEAD_VLAN_EGRESS_SGACL_DISABLED value 3 Pass
LEAD_VLAN_EGRESS_VLAN_CLIENT_LABEL value 0 Pass
LEAD_VLAN_EGRESS_VLAN_ID_VALID value 1 Pass
LEAD_VLAN_EGRESS_VLAN_LOAD_BALANCE_GROUP value 15 Pass
LEAD_VLAN_EGRESS_INTRA_POD_BCAST value 0 Pass

LEAD_VLAN_EGRESS_DHCP_SNOOPING_ENABLED_IPV4 value 1 Pass


LEAD_VLAN_EGRESS_DHCP_SNOOPING_ENABLED_IPV6 value 1 Pass
LEAD_VLAN_EGRESS_VXLAN_FLOOD_MODE value 0 Pass
LEAD_VLAN_MAX value 0 Pass
<SNIP>
```

This table lists the various common Punject show/debug commands that can be used to trace the path of DHCP packet on a live network.

Common Punt / Inject show & debug commands
debug plat soft fed swit acti inject add-filter cause 255 sub_cause 0 src_mac 0 0 0 dst_mac 0 0 0 src_ipv4 192.168.12.1 dst_ipv4 0.0.0.0 if_id 0xf **set platform software trace fed [switch<num|active|standby>] inject verbose --- > use filter command shown to scope the traces to this specific host** **set platform software trace fed [switch<num|active|standby>] inject debug boot --- > for reload** **set platform software trace fed [switch<num|active|standby>] punt noise**

```
show platform software fed [switch<num|active|standby>] inject cause summary

show platform software fed [switch<num|active|standby>] punt cause summary

show platform software fed [switch<num|active|standby>] inject cpuq 0

show platform software fed [switch<num|active|standby>] punt cpuq 17 (dhcp queue)

show platform software fed [switch<num|active|standby>] active inject packet-capture det

show platform software infrastructure inject

show platform software infrastructure punt

show platform software infrastructure lsmpi driver

debug platform software infra punt dhcp

debug platform software infra inject
```

These commands are useful to check if any DHCP packet is received for a particular client.

- This feature allows you to capture all DHCP snooping communication associated with a given client mac address that is processed by the CPU via the IOS-DHCP software.
- This functionality is supported for both IPv4 and IPv6 traffic.
- This feature is enabled automatically.

Note: These commands are available from Cisco IOS XE Gibraltar 16.12.X.

switch#**show platform dhcpsnooping client stats** {mac-address}
switch#**show platform dhcpv6snooping ipv6 client stats** {mac-address}

<#root>

```
C9300#

show platform dhcpsnooping client stats 0000.1AC2.C148

DHCPSN: DHCP snooping server
DHCPD:  DHCP protocol daemen
L2FWD:  Transmit Packet to driver in L2 format
FWD:    Transmit Packet to driver
Packet Trace for client MAC 0000.1AC2.C148:
Timestamp            Destination MAC  Destination Ip  VLAN  Message      Handler:Action
-------------------- ---------------- --------------- ----  ------------ --------------
06-27-2019 20:48:28  FFFF.FFFF.FFFF   255.255.255.255 88   DHCPDISCOVER PUNT:RECEIVED
06-27-2019 20:48:28  FFFF.FFFF.FFFF   255.255.255.255 88   DHCPDISCOVER PUNT:TO_DHCPSN
06-27-2019 20:48:28  FFFF.FFFF.FFFF   255.255.255.255 88   DHCPDISCOVER BRIDGE:RECEIVED
06-27-2019 20:48:28  FFFF.FFFF.FFFF   255.255.255.255 88   DHCPDISCOVER BRIDGE:TO_DHCPD
06-27-2019 20:48:28  FFFF.FFFF.FFFF   255.255.255.255 88   DHCPDISCOVER BRIDGE:TO_INJECT
06-27-2019 20:48:28  FFFF.FFFF.FFFF   255.255.255.255 88   DHCPDISCOVER L2INJECT:TO_FWD
06-27-2019 20:48:28  0000.0000.0000   192.168.1.1     0    DHCPDISCOVER INJECT:RECEIVED
06-27-2019 20:48:28  0000.0000.0000   192.168.1.1     0    DHCPDISCOVER INJECT:TO_L2FWD
```

```
06-27-2019 20:48:30  0000.0000.0000   10.1.1.3        0    DHCPOFFER    INJECT:RECEIVED
06-27-2019 20:48:30  0000.1AC2.C148   10.1.1.3        0    DHCPOFFER    INTERCEPT:RECEIVED
06-27-2019 20:48:30  0000.1AC2.C148   10.1.1.3        88   DHCPOFFER    INTERCEPT:TO_DHCPSN
06-27-2019 20:48:30  0000.1AC2.C148   10.1.1.3        88   DHCPOFFER    INJECT:CONSUMED
06-27-2019 20:48:30  FFFF.FFFF.FFFF   255.255.255.255 88   DHCPREQUEST  PUNT:RECEIVED
06-27-2019 20:48:30  FFFF.FFFF.FFFF   255.255.255.255 88   DHCPREQUEST  PUNT:TO_DHCPSN
06-27-2019 20:48:30  FFFF.FFFF.FFFF   255.255.255.255 88   DHCPREQUEST  BRIDGE:RECEIVED
06-27-2019 20:48:30  FFFF.FFFF.FFFF   255.255.255.255 88   DHCPREQUEST  BRIDGE:TO_DHCPD
06-27-2019 20:48:30  FFFF.FFFF.FFFF   255.255.255.255 88   DHCPREQUEST  BRIDGE:TO_INJECT
06-27-2019 20:48:30  FFFF.FFFF.FFFF   255.255.255.255 88   DHCPREQUEST  L2INJECT:TO_FWD
06-27-2019 20:48:30  0000.0000.0000   192.168.1.1     0    DHCPREQUEST  INJECT:RECEIVED
06-27-2019 20:48:30  0000.0000.0000   192.168.1.1     0    DHCPREQUEST  INJECT:TO_L2FWD
06-27-2019 20:48:30  0000.0000.0000   10.1.1.3        0    DHCPACK      INJECT:RECEIVED
06-27-2019 20:48:30  0000.1AC2.C148   10.1.1.3        0    DHCPACK      INTERCEPT:RECEIVED
06-27-2019 20:48:30  0000.1AC2.C148   10.1.1.3        88   DHCPACK      INTERCEPT:TO_DHCPSN
```

Use these commands to clear the trace.

switch#**clear platform dhcpsnooping pkt-trace ipv4**
switch#**clear platform dhcpsnooping pkt-trace ipv6**

CPU Path Packet Capture

Confirm if DHCP Snooping packets arrive and leave the control-plane properly.

Note: For additional references about how to use Forwarding Engine Driver CPU capture tool, refer to Further Reading section.

<#root>

```
debug platform software fed

 [switch<num|active|standby>]

punt/inject

 packet-capture start

debug platform software fed

 [switch<num|active|standby>]

punt/inject

 packet-capture stop

show platform software fed

  [switch<num|active|standby>]

punt/inject

 packet-capture brief
```

### PUNT ###

DISCOVER

```
------ Punt Packet Number: 16, Timestamp: 2021/04/14 19:10:09.924 ------
interface :
```

physical: FortyGigabitEthernet1/0/2

```
[if-id: 0x0000000a], pal: FortyGigabitEthernet1/0/2 [if-id: 0x0000000a]
metadata : cause: 79
```

[dhcp snoop],

```
 sub-cause: 11, q-no: 17, linktype: MCP_LINK_TYPE_IP [1]
ether hdr : dest mac: ffff.ffff.ffff,
```

src mac: 00a3.d144.2046

```
ether hdr : ethertype: 0x0800 (IPv4)
ipv4 hdr : dest ip: 255.255.255.255, src ip: 0.0.0.0
ipv4 hdr : packet len: 347, ttl: 255, protocol: 17 (UDP)
udp hdr : dest port:
```

67

```
, src port:
```

68

OFFER

```
------ Punt Packet Number: 23, Timestamp: 2021/04/14 19:10:11.926 ------
interface :
```

physical: FortyGigabitEthernet1/0/10

```
[if-id: 0x00000012], pal: FortyGigabitEthernet1/0/10 [if-id: 0x00000012]
metadata : cause: 79
```

[dhcp snoop]

```
, sub-cause: 11, q-no: 17, linktype: MCP_LINK_TYPE_IP [1]
ether hdr : dest mac: ffff.ffff.ffff,
```

src mac: 701f.539a.fe46

```
ether hdr : vlan: 10, ethertype: 0x8100
ipv4 hdr : dest ip: 255.255.255.255,
```

src ip: 10.0.0.1

```
ipv4 hdr : packet len: 330, ttl: 255, protocol: 17 (UDP)
udp hdr : dest port:
```

68

```
, src port:

67




REQUEST


------ Punt Packet Number: 24, Timestamp: 2021/04/14 19:10:11.927 ------
interface :

physical: FortyGigabitEthernet1/0/2

[if-id: 0x0000000a], pal: FortyGigabitEthernet1/0/2 [if-id: 0x0000000a]
metadata : cause: 79

[dhcp snoop]

, sub-cause: 11, q-no: 17, linktype: MCP_LINK_TYPE_IP [1]
ether hdr : dest mac: ffff.ffff.ffff,

src mac: 00a3.d144.2046


ether hdr : ethertype: 0x0800 (IPv4)
ipv4 hdr : dest ip: 255.255.255.255, src ip: 0.0.0.0
ipv4 hdr : packet len: 365, ttl: 255, protocol: 17 (UDP)
udp hdr : dest port:

67

, src port:

 68




ACK


------ Punt Packet Number: 25, Timestamp: 2021/04/14 19:10:11.929 ------
interface :

physical: FortyGigabitEthernet1/0/10

[if-id: 0x00000012], pal: FortyGigabitEthernet1/0/10 [if-id: 0x00000012]
metadata : cause: 79

[dhcp snoop]

, sub-cause: 11, q-no: 17, linktype: MCP_LINK_TYPE_IP [1]
ether hdr : dest mac: ffff.ffff.ffff,

src mac: 701f.539a.fe46


ether hdr : vlan: 10, ethertype: 0x8100
ipv4 hdr : dest ip: 255.255.255.255,

src ip: 10.0.0.1


ipv4 hdr : packet len: 330, ttl: 255, protocol: 17 (UDP)
```

```
udp hdr : dest port:

 68

, src port:

67
```

### INJECT ###

DISCOVER

```
------ Inject Packet Number: 33, Timestamp: 2021/04/14 19:53:01.273 ------
interface : pal:

FortyGigabitEthernet1/0/2

 [if-id: 0x0000000a]
metadata : cause: 25 [Layer2 frame to BD], sub-cause: 1, q-no: 0, linktype: MCP_LINK_TYPE_IP [1]
ether hdr : dest mac: ffff.ffff.ffff,

src mac: 00a3.d144.2046


ether hdr : ethertype: 0x0800 (IPv4)
ipv4 hdr : dest ip: 255.255.255.255, src ip: 0.0.0.0
ipv4 hdr : packet len: 347, ttl: 255, protocol: 17 (UDP)
udp hdr : dest port:

67

, src port:

68
```

OFFER

```
------ Inject Packet Number: 51, Timestamp: 2021/04/14 19:53:03.275 ------
interface : pal:

FortyGigabitEthernet1/0/2

[if-id: 0x0000000a]
metadata : cause: 1 [L2 control/legacy], sub-cause: 0, q-no: 0, linktype: MCP_LINK_TYPE_LAYER2 [10]
ether hdr : dest mac: ffff.ffff.ffff,

src mac: 701f.539a.fe46


ether hdr : ethertype: 0x0800 (IPv4)
ipv4 hdr : dest ip: 255.255.255.255,

src ip: 10.0.0.1


ipv4 hdr : packet len: 330, ttl: 255, protocol: 17 (UDP)
```

```
udp hdr : dest port:

68,

 src port:

67



REQUEST


------ Inject Packet Number: 52, Timestamp: 2021/04/14 19:53:03.276 ------
interface : pal:

FortyGigabitEthernet1/0/2

[if-id: 0x0000000a]
metadata : cause: 25 [Layer2 frame to BD], sub-cause: 1, q-no: 0, linktype: MCP_LINK_TYPE_IP [1]
ether hdr : dest mac: ffff.ffff.ffff,

src mac: 00a3.d144.2046


ether hdr : ethertype: 0x0800 (IPv4)
ipv4 hdr : dest ip: 255.255.255.255, src ip: 0.0.0.0
ipv4 hdr : packet len: 365, ttl: 255, protocol: 17 (UDP)
udp hdr : dest port:

 67

, src port:

68



ACK


------ Inject Packet Number: 53, Timestamp: 2021/04/14 19:53:03.278 ------
interface : pal:

FortyGigabitEthernet1/0/2

 [if-id: 0x0000000a]
metadata : cause: 1 [L2 control/legacy], sub-cause: 0, q-no: 0, linktype: MCP_LINK_TYPE_LAYER2 [10]
ether hdr : dest mac: ffff.ffff.ffff,

src mac: 701f.539a.fe46


ether hdr : ethertype: 0x0800 (IPv4)
ipv4 hdr : dest ip: 255.255.255.255,

src ip: 10.0.0.1


ipv4 hdr : packet len: 330, ttl: 255, protocol: 17 (UDP)
udp hdr : dest port:

68

, src port:

67
```

Useful Traces

These are binary traces which display events per process or component. In this example, the traces show information about the DHCPSN component.

- The traces can be manually rotated, which means that you can create a new file before you start to troubleshoot so that it contains cleaner information.

<#root>

```
9500#

request platform software trace rotate all


9500#

set platform software trace fed [switch<num|active|standby>] dhcpsn verbose




c9500#show logging proc fed internal | inc dhcp



<<---- DI_Handle must match with the output which retrieves the DI handle


2021/04/14 19:24:19.159536 {fed_F0-0}{1}: [dhcpsn] [17035]: (info):

VLAN event on vlan 10, enabled 1


2021/04/14 19:24:19.159975 {fed_F0-0}{1}: [dhcpsn] [17035]: (debug): Program trust ports for this vlan
2021/04/14 19:24:19.159978 {fed_F0-0}{1}: [dhcpsn] [17035]: (debug):

GPN (10) if_id (0x0000000000000012) <<---- if_id must match with the TRUSTED port


2021/04/14 19:24:19.160029 {fed_F0-0}{1}: [dhcpsn] [17035]: (debug): trusted_if_q size=1 for vlan=10
2021/04/14 19:24:19.160041 {fed_F0-0}{1}: [dhcpsn] [17035]: (ERR): update ri has failed vlanid[10]
2021/04/14 19:24:19.160042 {fed_F0-0}{1}: [dhcpsn] [17035]: (debug): vlan mode changed to enable
2021/04/14 19:24:27.507358 {fed_F0-0}{1}: [dhcpsn] [23451]: (debug): get di for vlan_id 10
2021/04/14 19:24:27.507365 {fed_F0-0}{1}: [dhcpsn] [23451]: (debug): Allocated rep_ri for vlan_id 10
2021/04/14 19:24:27.507366 {fed_F0-0}{1}: [inject] [23451]: (verbose): Changing di_handle from 0x7f7fac

0x7f7fac23e438

 by dhcp snooping
2021/04/14 19:24:27.507394 {fed_F0-0}{1}: [inject] [23451]: (debug): TX: getting REP RI from dhcpsn fai
2021/04/14 19:24:29.511774 {fed_F0-0}{1}: [dhcpsn] [23451]: (debug): get di for vlan_id 10
2021/04/14 19:24:29.511780 {fed_F0-0}{1}: [dhcpsn] [23451]: (debug): Allocated rep_ri for vlan_id 10
2021/04/14 19:24:29.511780 {fed_F0-0}{1}: [inject] [23451]: (verbose): Changing di_handle from 0x7f7fac

0x7f7fac23e438

 by dhcp snooping
2021/04/14 19:24:29.511802 {fed_F0-0}{1}: [inject] [23451]: (debug): TX: getting REP RI from dhcpsn fai
```

```
c9500#set platform software trace fed [switch<num|active|standby>] asic_app verbose
```

```
c9500#show logging proc fed internal | inc dhcp


2021/04/14 20:13:56.742637 {fed_F0-0}{1}: [dhcpsn] [17035]: (info):

VLAN event on vlan 10

, enabled 0
2021/04/14 20:13:56.742783 {fed_F0-0}{1}: [dhcpsn] [17035]: (debug): vlan mode changed to disable
2021/04/14 20:14:13.948214 {fed_F0-0}{1}: [dhcpsn] [17035]: (info): VLAN event on vlan 10, enabled 1
2021/04/14 20:14:13.948686 {fed_F0-0}{1}: [dhcpsn] [17035]: (debug):

Program trust ports for this vlan


2021/04/14 20:14:13.948688 {fed_F0-0}{1}: [dhcpsn] [17035]: (debug):

GPN (10) if_id (0x0000000000000012) <<---- if_id must match with the TRUSTED port


2021/04/14 20:14:13.948740 {fed_F0-0}{1}: [dhcpsn] [17035]: (debug): trusted_if_q size=1 for vlan=10
2021/04/14 20:14:13.948753 {fed_F0-0}{1}: [dhcpsn] [17035]: (ERR): update ri has failed vlanid[10]
2021/04/14 20:14:13.948754 {fed_F0-0}{1}: [dhcpsn] [17035]: (debug): vlan mode changed to enable
```

Suggested Traces

```
set platform software trace fed [switch<num|active|standby>] pm_tdl verbose
set platform software trace fed [switch<num|active|standby>] pm_vec verbose
set platform software trace fed [switch<num|active|standby>] pm_vlan verbose
```

INJECT

```
set platform software trace fed [switch<num|active|standby>] dhcpsn verbose
set platform software trace fed [switch<num|active|standby>] asic_app verbose
set platform software trace fed [switch<num|active|standby>] inject verbose
```

PUNT

```
set platform software trace fed [switch<num|active|standby>] dhcpsn verbose
set platform software trace fed [switch<num|active|standby>] asic_app verbse
set platform software trace fed [switch<num|active|standby>] punt ver
```

Syslogs and Explanations

Violations of DHCP rate limits.

Explanation: DHCP snooping detected a DHCP packet rate-limit violation on the specified interface.

```
%DHCP_SNOOPING-4-DHCP_SNOOPING_ERRDISABLE_WARNING: DHCP Snooping received 300 DHCP packets on interface
%DHCP_SNOOPING-4-DHCP_SNOOPING_RATE_LIMIT_EXCEEDED: The interface Fa0/2 is receiving more than the thre
```

DHCP Server spoofing on an untrusted port.

Explanation:The DHCP snooping feature discovered certain types of DHCP messages not allowed on the untrusted interface, which indicates some host trying to act as a DHCP server.

```
%DHCP_SNOOPING-5-DHCP_SNOOPING_UNTRUSTED_PORT: DHCP_SNOOPING drop message on untrusted port, message ty
```

Layer 2 MAC address does not match the MAC address inside DHCP request.

Explanation: The DHCP snooping feature attempted MAC address validation and the check failed. The source MAC address in the Ethernet header does not match the address in the chaddr field of the DHCP request message. There can be a malicious host that tries to carry out a denial of service attack on the DHCP server.

```
%DHCP_SNOOPING-5-DHCP_SNOOPING_MATCH_MAC_FAIL: DHCP_SNOOPING drop message because the chaddr doesn't ma
```

Option 82 insertion issue.

Explanation: The DHCP snooping feature discovered a DHCP packet with option values not allowed on the untrusted port, which indicates some host trying to act as a DHCP relay or server.

```
%DHCP_SNOOPING-5-DHCP_SNOOPING_NONZERO_GIADDR: DHCP_SNOOPING drop message with non-zero giaddr or optio
```

Layer 2 MAC address received on wrong port.

Explanation: The DHCP snooping feature has detected a host trying to carry out a denial of service attack on another host in the network.

```
%DHCP_SNOOPING-5-DHCP_SNOOPING_FAKE_INTERFACE: DHCP_SNNOPING drop message with mismatched source interf
```

DHCP messages received on the untrusted interface.

Explanation: The DHCP snooping feature discovered certain types of DHCP messages not allowed on the untrusted interface, which indicates some host trying to act as a DHCP server.

```
%DHCP_SNOOPING-5-DHCP_SNOOPING_UNTRUSTED_PORT: DHCP_SNOOPING drop message on untrusted port: GigabitEth
```

DHCP Snooping transfer failed. Unable to access URL.

Explanation: The DHCP snooping binding transfer failed.

```
%DHCP_SNOOPING-4-AGENT_OPERATION_FAILED: DHCP snooping binding transfer failed. Unable to access URL
```

DHCP Snooping Caveats

Cisco Bug ID Number	Description
CSCvi39202	DHCP fails when DHCP snooping trust is enabled on uplink etherchannel.
CSCvp49518	DHCP Snooping database is not refreshed after reload.
CSCvk16813	DHCP client traffic dropped with DHCP snooping and port-channel or cross stack uplinks.
CSCvd51480	Unbinding ip DHCP snooping and device-tracking.
CSCvm55401	DHCP snooping can drop DHCP option 82 packets w/ ip DHCP snooping information option allow-untrusted.
CSCvx25841	DHCP snooping trust state breaks when there is a change in REP segment.
CSCvs15759	DHCP server sends out a NAK packet during DHCP renewal process.
CSCvk34927	DHCP snooping table not updated from DHCP snooping DB file upon reload.

SDA Border DHCP Snooping

DHCP Snooping Statistics CLI.

A new CLI available for SDA to verify DHCP snooping statistics.

Note: For additional references about Cisco SD-Access Fabric Edge DHCP Process/Packet Flow and Decoding, refer to the guide in the Related Information section.

switch#**show platform fabric border dhcp snooping ipv4 statistics**
switch#**show platform fabric border dhcp snooping ipv6 statistics**

<#root>

```
SDA-9300-BORDER#

show platform fabric border dhcp snooping ipv4 statistics

Timestamp           Source IP    Destination IP  Source Remote Locator  Lisp Instance ID  VLAN  PROCESS
------------------- ----------   --------------  ---------------------  ----------------  ----- -------
08-05-2019 00:24:16 10.30.30.1   10.40.40.1      192.168.0.1             8189              88    10
08-05-2019 00:24:16 10.30.30.1   10.40.40.1      192.168.0.1             8189              88    11

SDA-9300-BORDER#

show platform fabric border dhcp snooping ipv6 statistics

Timestamp            Source IP              Destination IP          Source Remote Locator  Lisp Instanc
-------------------  ---------------------- ----------------------  ---------------------  ------------
08-05-2019 00:41:46  11:11:11:11:11:11:11:1 22:22:22:22:22:22:22:1  192.168.0.3             8089
08-05-2019 00:41:47  11:11:11:11:11:11:11:1 22:22:22:22:22:22:22:1  192.168.0.3             8089
```

Related Information

IP Addressing Services Configuration Guide, Cisco IOS XE Amsterdam 17.3.x (Catalyst 9200 Switches)

IP Addressing Services Configuration Guide, Cisco IOS XE Amsterdam 17.3.x (Catalyst 9300 Switches)

IP Addressing Services Configuration Guide, Cisco IOS XE Amsterdam 17.3.x (Catalyst 9400 Switches)

IP Addressing Services Configuration Guide, Cisco IOS XE Amsterdam 17.3.x (Catalyst 9500 Switches)

IP Addressing Services Configuration Guide, Cisco IOS XE Amsterdam 17.3.x (Catalyst 9600 Switches)

Cisco SD-Access Fabric Edge DHCP Process/Packet Flow and Decoding

Configure FED CPU Packet Capture on Catalyst 9000 Switches

Technical Support & Documentation - Cisco Systems