

Catalyst 9000 스위치에서 DHCP 스누핑 운영 및 문제 해결

목차

[소개](#)

[사전 요구 사항](#)

[요구 사항](#)

[사용되는 구성 요소](#)

[배경 정보](#)

[DHCP 스누핑](#)

[DHCP 스누핑 작업](#)

[토폴로지](#)

[구성](#)

[다음](#)을 확인합니다.

[문제 해결](#)

[소프트웨어 트러블슈팅](#)

[CPU\(Punt/Path Traffic\) 문제 해결](#)

[하드웨어 문제 해결](#)

[CPU 경로 패킷 캡처](#)

[유용한 추적](#)

[Syslog 및 설명](#)

[DHCP 스누핑 주의 사항](#)

[SDA 보더 DHCP 스누핑](#)

[관련 정보](#)

소개

이 문서에서는 Catalyst 9000 Series 스위치에서 DHCP 스누핑을 운영하고 문제를 해결하는 방법에 대해 설명합니다.

사전 요구 사항

요구 사항

다음 주제에 대한 지식을 보유하고 있으면 유용합니다.

- Catalyst 9000 Series 스위치 아키텍처
- Cisco IOS® XE 소프트웨어 아키텍처


사용되는 구성 요소

이 문서의 정보는 다음 소프트웨어 및 하드웨어 버전을 기반으로 합니다.

- C9200
- C9300
- C9400
- C9500
- C9600

Cisco IOS® XE 16.12.X

이 문서의 정보는 특정 랩 환경의 디바이스를 토대로 작성되었습니다. 이 문서에 사용된 모든 디바이스는 초기화된(기본) 컨피그레이션으로 시작되었습니다. 현재 네트워크가 작동 중인 경우 모든 명령의 잠재적인 영향을 미리 숙지하시기 바랍니다.

 참고: 다른 Cisco 플랫폼에서 이러한 기능을 활성화하는 데 사용되는 명령에 대해서는 해당 컨피그레이션 가이드를 참조하십시오.

배경 정보

DHCP 스누핑

DHCP(Dynamic Host Configuration Protocol) 스누핑은 DHCP 트래픽을 검사하여 악성 DHCP 패킷을 차단하는 데 사용되는 보안 기능입니다. 신뢰할 수 없는 사용자 포트와 네트워크의 DHCP 서버 포트 간의 방화벽 역할을 하여 네트워크에서 악성 DHCP 서버가 서비스 거부를 일으킬 수 있으므로 이를 방지합니다.

DHCP 스누핑 작업


DHCP Snooping은 신뢰할 수 있는 신뢰할 수 없는 인터페이스의 개념과 작동 합니다. DHCP 트래픽의 경로를 통해 스위치는 인터페이스에서 수신된 DHCP 패킷을 확인하고, 신뢰할 수 있는 인터페이스를 통해 예상되는 DHCP 서버 패킷(OFFER 및 ACK)을 추적합니다. 즉, 신뢰할 수 없는 인터페이스는 DHCP 서버 패킷을 차단합니다.

DHCP 패킷은 신뢰할 수 없는 인터페이스에서 차단됩니다.

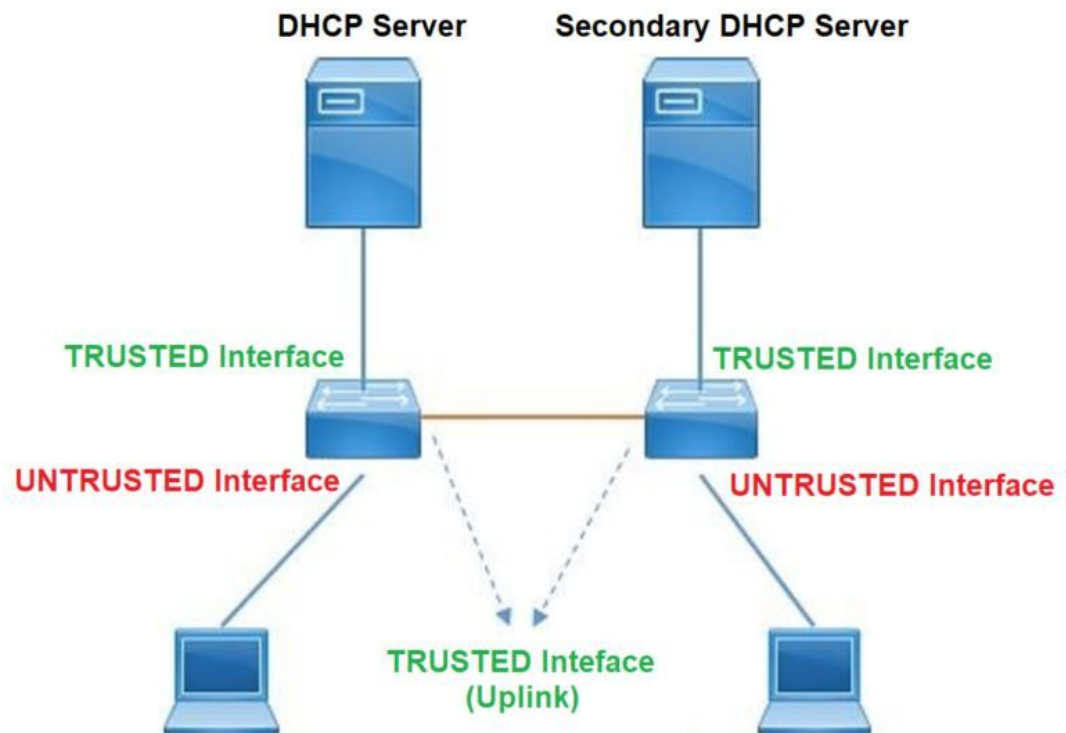
- DHCP OFFER, DHCP ACK, DHCP NAK 또는 DHCP REQUEST 패킷과 같은 DHCP 서버의 패킷은 네트워크 또는 방화벽 외부에서 수신됩니다. 그러면 비인가 DHCP 서버가 신뢰할 수 없는 포트에서 네트워크에 대한 공격을 방지합니다.
- 신뢰할 수 없는 인터페이스에서 수신된 패킷과 소스 MAC 주소 및 DHCP 클라이언트 하드웨어 주소가 일치하지 않습니다. 이렇게 하면 DHCP 서버에서 서비스 거부 공격을 생성할 수 있는 비인가 클라이언트에서 DHCP 패킷이 스누핑되는 것을 방지할 수 있습니다.
- DHCP 스누핑 바인딩 데이터베이스에 MAC 주소가 있지만 바인딩 데이터베이스의 인터페이스 정보가 메시지를 받은 인터페이스와 일치하지 않는 DHCP RELEASE 또는 DHCP DENY 브로드캐스트 메시지입니다. 이렇게 하면 클라이언트에 대한 서비스 거부 공격을 방지할 수 있습니다.
- 0.0.0.0이 아닌 릴레이 에이전트 IP 주소를 포함하는 DHCP 릴레이 에이전트가 전달한 DHCP

패킷 또는 옵션 82 정보를 포함하는 패킷을 신뢰할 수 없는 포트로부터 전달합니다. 이렇게 하면 네트워크에서 릴레이 에이전트 정보가 스푸핑되지 않습니다.

DHCP Snooping을 구성하는 스위치는 DHCP Snooping 테이블 또는 DHCP 바인딩 데이터베이스를 구축합니다. 이 테이블은 합법적인 DHCP 서버에서 할당된 IP 주소를 추적하는 데 사용됩니다. 바인딩 데이터베이스는 동적 ARP 검사 및 IP 소스 보호와 같은 다른 IOS 보안 기능에서도 사용됩니다.

 참고: DHCP Snooping이 올바르게 작동하도록 하려면 DHCP 서버에 도달할 때까지 모든 업링크 포트를 신뢰하고 최종 사용자 포트의 신뢰를 취소해야 합니다.

토폴로지



구성

전역 컨피그레이션

<#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 are 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

 참고: option-82 패킷을 허용하려면 ip dhcp snooping information 옵션 allow-untrusted를 활성화해야 합니다.

다음을 확인합니다.

원하는 VLAN에서 DHCP Snooping이 활성화되어 있는지 확인하고 신뢰할 수 있는 인터페이스와 신뢰할 수 없는 인터페이스가 잘 나열되어 있는지 확인합니다. 구성된 속도가 있는 경우 해당 속도도 나열되어 있는지 확인합니다.

<#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:

사용자가 DHCP를 통해 IP를 받으면 이 출력에 나열됩니다.

- DHCP Snooping은 IP 주소 임대가 만료되거나 스위치에서 호스트로부터 DHCPRELEASE 메시지를 받으면 데이터베이스의 항목을 제거합니다.
- 최종 사용자 MAC 주소에 대해 나열된 정보가 올바른지 확인합니다.

<#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

```

이 표에는 DHCP Snooping 정보를 모니터링하는 데 사용할 수 있는 다양한 명령이 나열되어 있습니다.


명령을 사용합니다	목적
<pre>show ip dhcp snooping binding show ip dhcp snooping binding [IP-address] [MAC-address] [interface ethernet slot/port] [vlan-id]</pre>	<p>바인딩 테이블이라고도 하는 DHCP 스누핑 바인딩 데이터베이스에서 동적으로 구성된 바인딩만 표시합니다.</p> <ul style="list-style-type: none"> - 바인딩 항목 IP 주소 - 바인딩 항목 Mac 주소 - 바인딩 항목 입력 인터페이스 - 바인딩 항목 VLAN
<pre>show ip dhcp snooping 데이터베이스</pre>	<p>DHCP 스누핑 바인딩 데이터베이스 상태 및 통계를 표시합니다.</p>
<pre>show ip dhcp snooping 통계</pre>	<p>DHCP 스누핑 통계를 요약 또는 세부사항 형식으로 표시합니다.</p>
<pre>show ip source binding</pre>	<p>동적으로 고정으로 구성된 바인딩을 표시합니다.</p>
<pre>show interface vlan xyz</pre>	<p>DHCP 패킷은 클라이언트 VLAN SVI를 통해 클라이언트</p>

<pre>show buffer input-interface Vlan xyz dump</pre>	<p>VLAN에 구성된 릴레이 에이전트로 전송됩니다. 입력 대기열에 삭제 또는 최대 제한에 도달하는 것이 표시되면 클라이언트의 DHCP 패킷이 삭제되어 구성된 릴레이 에이전트에 연결할 수 없는 것일 수 있습니다.</p> <hr/> <p> 참고: 입력 대기열에서 삭제는 표시되지 않습니다.</p> <hr/> <pre>switch#show int vlan 670 5초 동안 로드: 13%/0%, 1분: 10%, 5분: 10% 시간 소스는 NTP, 18:39:52.476 UTC Thu Sep 10 2020입니다.</pre> <p>Vlan670이 작동, 라인 프로토콜이 작동, 자동 상태가 활성화됨 하드웨어는 이더넷 SVI이며 주소는 00fd.227a.5920(bia 00fd.227a.5920)입니다. 설명: ion_media_client 인터넷 주소는 10.27.49.254/23입니다 MTU 1500바이트, BW 1000000Kbit/sec, DLY 10usec, 안정성 255/255, txload 1/255, rxload 1/255 캡슐화 ARPA, 루프백이 설정되지 않음 Keepalive가 지원되지 않음 ARP 유형: ARPA, ARP 시간 초과 04:00:00 마지막 입력 03:01:29, 출력 00:00:02, 출력 중단 없음 "show interface" 카운터를 마지막으로 지운 후 입력 큐: 375/375/4020251/0(size/max/drops/flushes), 총 출력 삭제: 0 ← 큐 입력 패킷 375개/4020251이 삭제되었습니다.</p>
--	---

문제 해결

소프트웨어 트러블슈팅

스위치에서 수신하는 것을 확인합니다. 이러한 패킷은 CPU 컨트를 플레인에서 처리되므로 inject 및 punt 방향의 모든 패킷을 확인하고 정보가 올바른지 확인합니다.

 주의: debug 명령은 주의해서 사용하십시오. 많은 debug 명령은 라이브 네트워크에 영향을 미치며 문제가 재현될 때만 랩 환경에서 사용하는 것이 좋습니다.

조건부 디버그 기능을 사용하면 정의한 조건 세트에 따라 특정 기능에 대한 디버그 및 로그를 선택적으로 활성화할 수 있습니다. 이는 특정 호스트 또는 트래픽에 대한 디버그 정보만 포함할 때 유용합니다.

조건은 기능 또는 ID를 나타냅니다. 여기서 ID는 인터페이스, IP 주소 또는 MAC 주소 등이 될 수 있

습니다.

DHCP Snooping 문제를 해결하기 위해 패킷 및 이벤트 디버그에 대한 조건부 디버그를 활성화하는 방법.

명령을 사용합니다	목적
디버그 조건 mac <mac-address> 예: switch#debug 조건 mac bc16.6509.3314	지정된 MAC 주소에 대한 조건부 디버깅을 구성합니다.
디버그 조건 vlan <VLAN ID> 예: switch#debug 조건 vlan 10	지정된 VLAN에 대한 조건부 디버깅을 구성합니다.
디버그 조건 인터페이스 <interface> 예: switch#debug condition interface twentyFiveGigE 1/0/8	지정된 인터페이스에 대한 조건부 디버깅을 구성합니다.

DHCP 스누핑을 디버깅하려면 표에 나와 있는 명령을 사용합니다.

명령을 사용합니다	목적
debug dhcp [detail 작 이중화]	상세 DHCP 패킷 내용 oper DHCP 내부 OPER 이중화 DHCP 클라이언트 이중화 지원
debug ip dhcp server packet detail	메시지 수신 및 전송을 세부적으로 디코딩
ip dhcp 서버 이벤트 디버그	주소 할당, 리스 만료 등을 보고합니다.
debug ip dhcp snooping agent	DHCP 스누핑 데이터베이스 읽기 및 쓰기 디버그

debug ip dhcp snooping event	각 구성 요소 간의 이벤트 디버그
debug ip dhcp snooping 패킷	dhcp snooping 모듈에서 DHCP 패킷 디버그

debug ip dhcp snooping 명령의 부분 샘플 출력입니다.

<#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: 0.0.0.0

Apr 14 16:16:46.835: DHCP_SNOOPING: bridge packet get invalid mat entry: FFFF.FFFF.FFFF, packet is flooded

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: 0.0.0.0

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 (FortyGigabitEthernet1/0/10)

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: 0.0.0.0

Apr 14 16:16:48.838: DHCP_SNOOPING: bridge packet get invalid mat entry: FFFF.FFFF.FFFF, packet is flooded

Apr 14 16:16:48.839: DHCP_SNOOPING: received new DHCP packet from input interface (FortyGigabitEthernet1/0/10)

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.

```

DHCP 스누핑 이벤트를 디버깅하려면 다음 단계를 수행합니다.

 주의: debug 명령은 주의해서 사용하십시오. 많은 debug 명령은 라이브 네트워크에 영향을 주며 문제가 재현될 때만 랩 환경에서 사용하는 것이 좋습니다.

요약 단계

1. 사용
2. 디버그 플랫폼 조건 mac {mac-address}
3. 디버그 플랫폼 조건 시작
4. 플랫폼 상태 표시 또는 디버그 표시
5. 디버그 플랫폼 조건 중지
6. show platform software trace message ios R0 reverse | DHCP 포함
7. 플랫폼 조건 모두 지우기

세부 단계

	명령 또는 작업	목적
1단계	<p>사용</p> <p>예:</p> <pre>switch#enable</pre>	<p>특별 권한 EXEC 모드를 활성화합니다.</p> <ul style="list-style-type: none"> • 프롬프트가 표시되면 비밀 번호를 입력합니다.
2단계	<p>디버그 플랫폼 조건 mac {mac-address}</p> <p>예:</p> <pre>switch#debug 플랫폼 조건 mac 0001.6509.3314</pre>	<p>지정된 MAC 주소에 대한 조건부 디버깅을 구성합니다.</p>

	명령 또는 작업	목적
3단계	디버그 플랫폼 조건 시작 예: switch#debug 플랫폼 조건 시작	조건부 디버깅을 시작합니다(조건 중 하나에 일치하는 경우 방사성 추적을 시작할 수 있음).
4단계	show platform condition 또는 show debug 예: switch#show 플랫폼 조건 switch#show 디버그	현재 설정된 조건을 표시합니다.
5단계	디버그 플랫폼 조건 중지 예: switch#debug 플랫폼 조건 중지	조건부 디버깅을 중지합니다(방사성 추적 중지 가능).
6단계	show platform software trace message ios R0 reverse DHCP 포함 예: switch#show platform software trace message ios R0 reverse DHCP 포함	최신 추적 파일에서 병합된 HP 로그를 표시합니다.
7단계	플랫폼 조건 모두 지우기 예: switch# 플랫폼 조건 모두 지우기	모든 조건을 지웁니다.

d의 부분 샘플 출력 예입니다.ebug 플랫폼 dhcp-snoop all 명령.

<#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(00a3.d144.2046)
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 SRC_ADDR = 10.0.0.6
```

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_mac(00a3.d144.2046)
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 SRC_ADDR = 0.0.0.0
Apr 14 16:44:24.637: sending dhcp packet out after processing with SMAC = 00a3.d144.2046{mac} and SRC_ADDR = 0.0.0.0
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_mac(00a3.d144.2046)
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 SRC_ADDR = 10.0.0.1
```


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 = 0.0.0.0
```

ACK

```
Apr 14 16:44:24.640: dhcp packet src_ip(10.10.10.1) dest_ip(255.255.255.255) src_udp(67) dest_udp(68) src_mac(00a3.d144.2046)
Apr 14 16:44:24.640: ngwc_dhcpsn_process_pak(284): Packet handedover to SISF on vlan 10 dhcp pkt processing routine is called for pak with SMAC = 00a3.d144.2046{mac} and SRC_ADDR = 10.10.10.1
```

이 표에는 플랫폼에서 DHCP 스누핑을 디버깅하는 데 사용할 수 있는 다양한 명령이 나열되어 있습니다.

 주의: debug 명령은 주의해서 사용하십시오. 많은 debug 명령은 라이브 네트워크에 영향을 미치며 문제가 재현될 때만 랩 환경에서 사용하는 것이 좋습니다.

명령을 사용합니다	목적
switch#debug platform dhcp-snoop [모두 패킷	모든 NGWC DHCP 스누핑

[피디심]	패킷 NGWC DHCP 스누핑 패킷 디버그 정보 pd-shim NGWC DHCP Snooping IOS Shim 디버그 정보
switch#debug 플랫폼 소프트웨어 인프라 punt dhcp-snoop	FP에서 수신되며 컨트롤 플레인으로 전송되는 패킷)
switch#debug 플랫폼 소프트웨어 인프라 삽입	컨트롤 플레인에서 FP로 주입되는 패킷

CPU(Punt/Path Traffic) 문제 해결

FED 관점에서 각 CPU 대기열에서 어떤 트래픽이 수신되는지 확인합니다(DHCP 스누핑은 컨트롤 플레인에서 처리되는 트래픽 유형).

- 트래픽이 스위치로 들어오면 PUNT 방향으로 CPU에 전송되고 dhcp snoop 큐로 전송됩니다.
- 스위치에서 트래픽을 처리하면 INJECT 방향을 통해 트래픽이 이동합니다. DHCP OFFER 및 ACK 패킷은 L2 제어/레거시 큐에 포함됩니다.

<#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

```

이 명령을 사용하여 CPU에 대한 트래픽을 확인하고 DHCP Snooping이 트래픽을 삭제하는지 확인할 수 있습니다.

```
<#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 no	Queue Name	Rx 10s	Rx 1min	Rx 5min	Drop 10s	Drop 1min	Drop 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	
		0	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	

하드웨어 문제 해결

포워딩 엔진 드라이버(FED)

FED가 ASIC를 프로그래밍하는 원동력입니다. FED 명령은 하드웨어 및 소프트웨어 상태가 일치하는지 확인하는 데 사용됩니다.

DI_Handle 값을 가져옵니다.

- DI 핸들은 특정 포트의 대상 인덱스를 참조합니다.

<#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
```

ifm 매핑을 확인하여 포트의 Asic 및 Core를 확인합니다.

- IFM은 특정 포트/코어/asic에 매핑된 내부 인터페이스 인덱스입니다.

<#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


```
1 1
1 0 4 4 2 2 NIF Y
```

하드웨어 인덱스를 가져오려면 DI_Handle를 사용합니다.

<#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_DHCPN00PI
priv_ri/priv_si Handle: (nil)Hardware Indices/Handles:
index0:0x5f03
mtu_index/13u_ri_index0:0x0 index1:0x5f03 mtu_index/13u_ri_index1:0x0 index2:0x5f03 mtu_index/13u_ri_i
<SNIP>
<-- Index is 0x5f03
```

인덱스 값 0x5f03을 16진수에서 10진수로 변환합니다.

0x5f03 = 24323

이 인덱스 값은 10진수로, ASIC 및 Core 값은 이 명령에서 포트에 대해 어떤 플래그가 설정되었는 지 확인합니다.

<#root>

```
c9500#show platform hardware fed switch 1 fwd-asic regi read register-name SifDestinationIndexTable-24323
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
```

특정 VLAN에 대해 DHCP Snooping이 활성화되었는지 확인합니다.

<#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/13u_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 00 00
```

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>
```

이 표에는 라이브 네트워크에서 DHCP 패킷의 경로를 추적하는 데 사용할 수 있는 다양한 일반적인 Punct show/debug 명령이 나열되어 있습니다.

일반 Punt/Inject show & debug 명령

```
debug plat soft fed switch acti inject add-filter cause 255 sub_cause 0 src_mac 0 0 dst_mac 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 — > filter
cpmmand를 사용하여 이 특정 호스트에 대한 추적 범위를 지정합니다.
```

```
set platform software trace fed [switch<num|active|standby>] inject debug boot — > for reload
```

```

set platform software trace fed [switch<num|active|standby>] punt 노이즈
show platform software fed [switch<num|active|standby>] inject summary 삽입
show platform software fed [switch<num|active|standby>] punt 원인 요약
show platform software fed [switch<num|active|standby>] inject cpuq 0
show platform software fed [switch<num|active|standby>] punt cpu 17(dhcp 큐)
show platform software fed [switch<num|active|standby>] active inject packet-capture det
플랫폼 소프트웨어 인프라 삽입 표시
플랫폼 소프트웨어 인프라 펀트 표시
show platform software infrastructure lsmipi driver
디버그 플랫폼 소프트웨어 infra punt dhcp
디버그 플랫폼 소프트웨어 infra inject

```

이 명령은 특정 클라이언트에 대해 DHCP 패킷이 수신되는지 확인하는 데 유용합니다.

- 이 기능을 사용하면 IOS-DHCP 소프트웨어를 통해 CPU에서 처리되는 지정된 클라이언트 mac 주소와 연결된 모든 DHCP 스누핑 통신을 캡처할 수 있습니다.
- 이 기능은 IPv4 및 IPv6 트래픽에서 모두 지원됩니다.
- 이 기능은 자동으로 활성화됩니다.

 **중요:** 이 명령은 Cisco IOS XE Gibraltar 16.12.X에서 사용할 수 있습니다.

```
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
```

DHCP SN: 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_DHCP SN
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	DHCP OFFER	INJECT:RECEIVED
06-27-2019	20:48:30	0000.1AC2.C148	10.1.1.3	0	DHCP OFFER	INTERCEPT:RECEIVED
06-27-2019	20:48:30	0000.1AC2.C148	10.1.1.3	88	DHCP OFFER	INTERCEPT:TO_DHCP SN
06-27-2019	20:48:30	0000.1AC2.C148	10.1.1.3	88	DHCP OFFER	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_DHCP SN
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_DHCP SN


추적을 지우려면 다음 명령을 사용합니다.

```
switch#clear platform dhcpsnooping pkt-trace ipv4
```

```
switch#clear platform dhcpsnooping pkt-trace ipv6
```

CPU 경로 패킷 캡처

DHCP Snooping 패킷이 도착하고 제어 평면을 올바르게 유지하는지 확인합니다.

 참고: Forwarding Engine Driver CPU 캡처 툴 사용 방법에 대한 추가 참조는 추가 읽기 섹션을 참조하십시오.

```
<#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], pa1: 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], pa1: 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], pa1: 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], pa1: 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 : pa1:

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 : pa1:

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
```

유용한 추적

프로세스 또는 구성 요소당 이벤트를 표시하는 이진 추적입니다. 이 예에서 추적은 dhcpcn 구성 요소에 대한 정보를 표시합니다.

- 추적은 수동으로 회전할 수 있습니다. 즉, 문제를 해결하기 전에 새 파일을 만들어 더 정확한 정보를 포함할 수 있습니다.

```
<#root>
```

```
9500#
```

```
request platform software trace rotate all
```

```
9500#
```

```
set platform software trace fed [switch
```

```
] dhcpcn 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}: [dhcpcn] [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

] 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
```

Syslog 및 설명

DHCP 속도 제한 위반.

설명: DHCP 스누핑이 지정된 인터페이스에서 DHCP 패킷 속도 제한 위반을 감지했습니다.

```
%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 three
```

신뢰할 수 없는 포트에서 DHCP 서버가 스누핑합니다.

설명:DHCP 스누핑 기능이 신뢰할 수 없는 인터페이스에서 허용되지 않는 특정 유형의 DHCP 메시지를 검색했습니다. 이는 일부 호스트가 DHCP 서버로 작동하려고 시도하고 있음을 나타냅니다.

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

레이어 2 MAC 주소가 DHCP 요청 내의 MAC 주소와 일치하지 않습니다.

설명: DHCP 스누핑 기능이 MAC 주소 유효성 검사를 시도했으나 실패했습니다. 이더넷 헤더의 소

스 MAC 주소가 DHCP 요청 메시지의 chaddr 필드의 주소와 일치하지 않습니다. DHCP 서버에서 서비스 거부 공격을 수행하려는 악의적인 호스트가 있을 수 있습니다.

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

옵션 82 삽입 문제.

설명: DHCP 스누핑 기능이 신뢰할 수 없는 포트에서 허용되지 않는 옵션 값이 있는 DHCP 패킷을 검색했습니다. 이는 일부 호스트가 DHCP 릴레이 또는 서버로 작동하려고 시도하고 있음을 나타냅니다.

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

잘못된 포트에서 수신된 레이어 2 MAC 주소.

설명: DHCP 스누핑 기능이 네트워크의 다른 호스트에서 서비스 거부 공격을 수행하려는 호스트를 감지했습니다.

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

신뢰할 수 없는 인터페이스에서 수신된 DHCP 메시지입니다.

설명: DHCP 스누핑 기능이 신뢰할 수 없는 인터페이스에서 허용되지 않는 특정 유형의 DHCP 메시지를 검색했습니다. 이는 일부 호스트가 DHCP 서버로 작동하려고 시도하고 있음을 나타냅니다.

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

DHCP Snooping 전송에 실패했습니다. URL에 액세스할 수 없습니다.

설명: DHCP 스누핑 바인딩 전송에 실패했습니다.

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


DHCP 스누핑 주의 사항

Cisco 버그 ID 번호	설명
CSCvi39202	업링크 etherchannel에서 DHCP 스누핑 트러스트가 활성화된 경우 DHCP가 실패합니다.
CSCvp49518	DHCP Snooping 데이터베이스는 다시 로드 후 새로 고쳐지지 않습니다.
CSCv16813	DHCP 클라이언트 트래픽이 DHCP 스누핑 및 포트 채널 또는 교차 스택 업링크에서 삭제되었습니다.
CSCvd51480	IP dhcp 스누핑 및 디바이스 추적 바인딩 해제.
CSCvm55401	DHCP 스누핑은 dhcp option 82 패킷을 삭제할 수 있으며 ip dhcp snooping information option allow-untrusted가 있습니다.
CSCvx25841	REP 세그먼트가 변경되면 DHCP 스누핑 신뢰 상태가 중단됩니다.
CSCvs15759	DHCP 갱신 프로세스 중에 DHCP 서버가 NAK 패킷을 보냅니다.
CSCv34927	다시 로드할 때 DHCP 스누핑 DB 파일에서 DHCP 스누핑 테이블이 업데이트되지 않았습니다.

SDA 보더 DHCP 스누핑

DHCP Snooping Statistics CLI.

SDA에서 DHCP 스누핑 통계를 확인할 수 있는 새 CLI가 제공됩니다.

 참고: Cisco SD-Access Fabric Edge DHCP Process/Packet Flow and Decoding에 대한 추가 참조는 관련 정보 섹션의 가이드를 참조하십시오.

```
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 Instance
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

관련 정보

[IP 주소 지정 서비스 컨피그레이션 가이드, Cisco IOS XE Amsterdam 17.3.x\(Catalyst 9200 스위치\)](#)

[IP 주소 지정 서비스 컨피그레이션 가이드, Cisco IOS XE Amsterdam 17.3.x\(Catalyst 9300 스위치\)](#)

[IP 주소 지정 서비스 컨피그레이션 가이드, Cisco IOS XE Amsterdam 17.3.x\(Catalyst 9400 스위치\)](#)

[IP 주소 지정 서비스 컨피그레이션 가이드, Cisco IOS XE Amsterdam 17.3.x\(Catalyst 9500 스위치\)](#)

[IP 주소 지정 서비스 컨피그레이션 가이드, Cisco IOS XE Amsterdam 17.3.x\(Catalyst 9600 스위치\)](#)

[Cisco SD-Access Fabric Edge DHCP 프로세스/패킷 흐름 및 디코딩](#)

[Catalyst 9000 스위치에서 FED CPU 패킷 캡처 구성](#)

[기술 지원 및 문서 - Cisco Systems](#)

이 번역에 관하여

Cisco는 전 세계 사용자에게 다양한 언어로 지원 콘텐츠를 제공하기 위해 기계 번역 기술과 수작업 번역을 병행하여 이 문서를 번역했습니다. 아무리 품질이 높은 기계 번역이라도 전문 번역가의 번역 결과물만큼 정확하지는 않습니다. Cisco Systems, Inc.는 이 같은 번역에 대해 어떠한 책임도 지지 않으며 항상 원본 영문 문서(링크 제공됨)를 참조할 것을 권장합니다.