

# Catalyst 9000スイッチでのDHCPスヌーピングの操作とトラブルシューティング

## 内容

---

[はじめに](#)

[前提条件](#)

[要件](#)

[使用するコンポーネント](#)

[背景説明](#)

[DHCPスヌーピング](#)

[DHCPスヌーピングの動作](#)

[トポロジ](#)

[設定](#)

[確認](#)

[トラブルシュート](#)

[ソフトウェアのトラブルシューティング](#)

[パント/バストラフィック\(CPU\)のトラブルシューティング](#)

[ハードウェアトラブルシューティング](#)

[CPUバスパケットキャプチャ](#)

[便利なトレース](#)

[syslogと説明](#)

[DHCPスヌーピングの警告](#)

[SDAボーダーDHCPスヌーピング](#)

[関連情報](#)

---

## はじめに

このドキュメントでは、Catalyst 9000シリーズスイッチでのDHCPスヌーピングの動作とトラブルシューティングの方法について説明します

## 前提条件

### 要件

次の項目に関する知識があることが推奨されます。

- Catalyst 9000シリーズスイッチのアーキテクチャ
- Cisco IOS® XEソフトウェアアーキテクチャ

### 使用するコンポーネント


このドキュメントの情報は、次のソフトウェアとハードウェアのバージョンに基づいています。

- C9200
- C9300
- C9400
- C9500
- C9600

Cisco IOS® XE 16.12.X

このドキュメントの情報は、特定のラボ環境にあるデバイスに基づいて作成されました。このドキュメントで使用するすべてのデバイスは、クリアな（デフォルト）設定で作業を開始しています。本稼働中のネットワークでは、各コマンドによって起こる可能性がある影響を十分確認してください。

---

 注：他のシスコプラットフォームでこれらの機能を有効にするために使用するコマンドについては、該当するコンフィギュレーションガイドを参照してください。

---

## 背景説明

### DHCP スヌーピング

Dynamic Host Configuration Protocol(DHCP)スヌーピングは、DHCPトラフィックをチェックして悪意のあるDHCPパケットをブロックするために使用されるセキュリティ機能です。ネットワーク上の信頼できないユーザポートとDHCPサーバポートの間のファイアウォールとして機能し、ネットワーク内の悪意のあるDHCPサーバによるサービス拒否を防止します。

### DHCPスヌーピングの動作

DHCPスヌーピングは、信頼できるインターフェイスと信頼できないインターフェイスの概念と連携して動作します。スイッチは、DHCPトラフィックのパスを通じて、インターフェイスで受信したDHCPパケットを確認し、信頼できるインターフェイス上で予想されるDHCPサーバパケット（OFFERおよびACK）を追跡します。つまり、信頼できないインターフェイスはDHCPサーバパケットをブロックします。


DHCPパケットが信頼できないインターフェイスでブロックされている。

- DHCP OFFER、DHCP ACK、DHCP NAK、または DHCP PLEASE QUERY パケットなどのDHCPサーバからのパケットが、ネットワークまたはファイアウォールの外部から受信された。これにより、不正なDHCPサーバが信頼できないポートからネットワークに攻撃されるのを防ぐことができます。
- 信頼できないインターフェイスで受信されたパケットと、送信元MACアドレスおよびDHCPクライアントハードウェアアドレスが一致しない。これにより、DHCPサーバでサービス拒否攻撃を引き起こす可能性がある不正なクライアントからのDHCPパケットのスプーフィングを防止できます。
- DHCPスヌーピングバインディングデータベース内のMACアドレスが指定されているが、バ

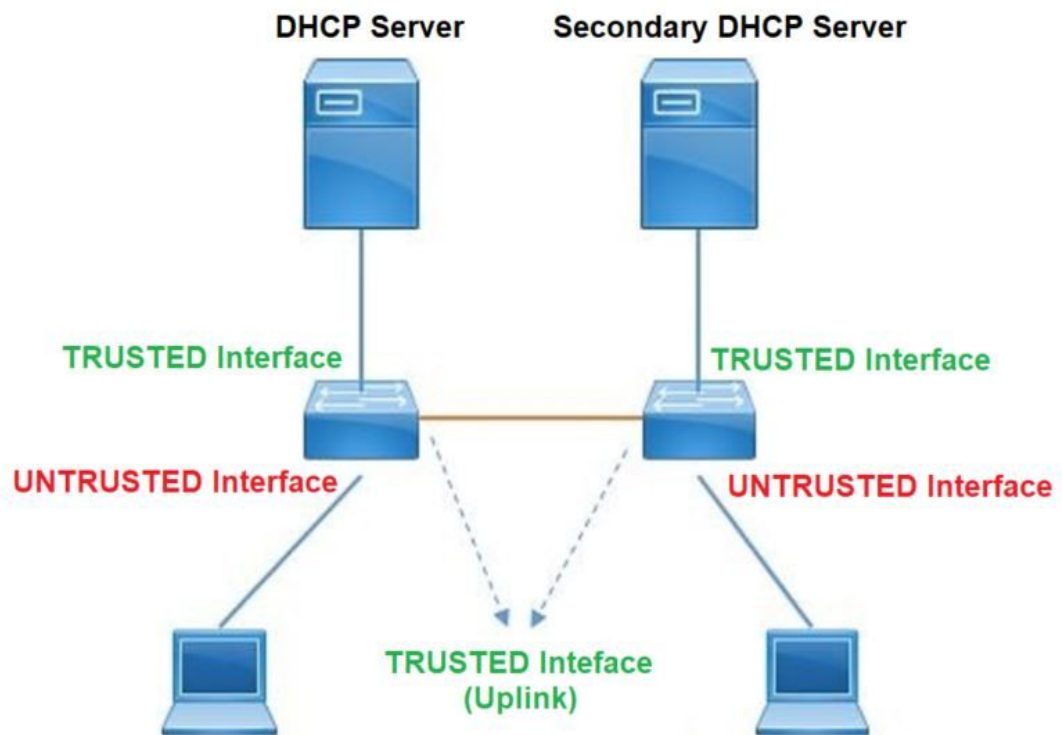
インデイングデータベース内のインターフェイス情報が、メッセージが受信されたインターフェイスと一致しないDHCPRELEASEまたはDHCPDECLINEブロードキャストメッセージ。これにより、クライアントに対するサービス拒否攻撃を防止できます。

- 0.0.0.0ではないリレーエージェントIPアドレスを含むDHCPリレーエージェントによって転送されるDHCPパケット、またはリレーエージェントがoption-82情報を含むパケットを信頼できないポートに転送する。これにより、ネットワーク上のリレーエージェント情報のスプーフィングが防止されます。

DHCPスヌーピングを設定するスイッチは、DHCPスヌーピングテーブルまたはDHCPバインディングデータベースを構築します。このテーブルは、正当なDHCPサーバから割り当てられたIPアドレスを追跡するために使用されます。バインディングデータベースは、ダイナミックARPインスペクションやIPソースガードなどの他のIOSセキュリティ機能でも使用されます。

 注:DHCPスヌーピングが正しく動作するには、DHCPサーバに到達するすべてのアップリンクポートを信頼し、エンドユーザポートを信頼しないようにします。

## トポロジ



## 設定

グローバル設定

<#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



注：option-82パケットを許可するには、ip dhcp snooping information option allow-untrustedを有効にする必要があります。

---

## 確認

目的のVLANでDHCPスヌーピングが有効になっているかどうかを確認し、信頼できるインターフェイスと信頼できないインターフェイスが適切にリストされていることを確認します。レートが設定されている場合は、そのレートもリストされていることを確認します。

<#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:

ユーザがDHCPによってIPを受信すると、次の出力にリストされます。

- DHCPスヌーピングは、IPアドレスのリースが期限切れになるか、スイッチがホストからDHCPRELEASEメッセージを受信すると、データベース内のエントリを削除します。
- エンドユーザのMACアドレスにリストされている情報が正しいことを確認します。


<#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
```

次の表に、DHCPスヌーピング情報の監視に使用できるさまざまなコマンドを示します。


コマンド	目的
<pre>show ip dhcp snooping binding  show ip dhcp snooping binding [IPアドレス] [MACアドレス] [インターフェイス/サネット/ポート] [vlan-id]</pre>	DHCPスヌーピングバインディングデータベース (バインディングテーブルとも呼ばれる) 内で動的に設定されたバインディングだけを表示します。  - エントリIPアドレスのバインド - エントリのMACアドレスのバインディング - エントリ入カインターフェイスのバインド - エントリVLANのバインディング
<pre>show ip dhcp snooping database</pre>	DHCPスヌーピングバインディングデータベースのステータスと統計情報を表示します。
<pre>show ip dhcp snooping statistics (DHCPスヌーピング統計情報の表示)</pre>	DHCPスヌーピングの統計情報を概要または詳細の形式で表示します。

show ip source binding	動的および静的に設定されたバインディングを表示します。
<p>show interface vlan xyz</p> <p>show buffer input-interface Vlan xyz dump</p>	<p>DHCPパケットは、クライアントVLAN SVI経由で、クライアントVLANに設定されたリレーエージェントに送信されます。入力キューに「drop」または「reach maximum limit」が表示されている場合、クライアントからのDHCPパケットがドロップされ、設定されたリレーエージェントに到達できなかった可能性があります。</p> <p> 注：入力キューにドロップが表示されないことを確認します。</p> <pre>switch#show int vlan 670 5秒間の負荷：13 %/0 %、1分間：10 %、5分間：10 % 時刻源：NTP, 18:39:52.476 UTC Thu Sep 10 2020  Vlan670 is up, line protocol is up , Autostate Enabled ( VLAN670はアップ、ラインプロトコルはアップ、自動ステートはイネーブル ) ハードウェアはイーサネットSVI、アドレスは 00fd.227a.5920(bia 00fd.227a.5920) 説明：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タイプ：ARPA、ARPタイムアウト04:00:00 Last input 03:01:29, output 00:00:02, output hang never Last clearing of "show interface" counters never 入力キュー：375/375/4020251/0(size/max/drops/flushes)、 合計出カドロップ数：0 ← キューの入力内の375パケット /4020251がドロップされた</pre>


## トラブルシューティング

### ソフトウェアのトラブルシューティング

スイッチが受信する内容を確認します。これらのパケットはCPUコントロールプレーンで処理されるため、インジェクト方向とパント方向のすべてのパケットを確認し、情報が正しいことを確認してください。

 注意:debugコマンドは注意して使用してください。多くのdebugコマンドは実稼働中のネッ



 ネットワークに影響を与えるため、問題が再現されるラボ環境でのみ使用することを推奨します。

条件付きデバッグ機能を使用すると、定義した一連の条件に基づいて、特定の機能のデバッグとログを選択的に有効にできます。これは、特定のホストまたはトラフィックだけにデバッグ情報を含める場合に便利です。

条件とは、インターフェイス、IPアドレス、またはMACアドレスなどのアイデンティティを含む機能またはアイデンティティを指します。

DHCPスヌーピングをトラブルシューティングするために、パケットデバッグとイベントデバッグの条件付きデバッグを有効にする方法。

コマンド	目的
デバッグ条件mac <macアドレス> 以下に例を挙げます。 switch#debug condition mac bc16.6509.3314	指定したMACアドレスの条件付きデバッグを設定します。
debug condition vlan <VLAN ID> 以下に例を挙げます。 switch#debug condition vlan 10	指定したVLANの条件付きデバッグを設定します。
debug condition interface <インターフェイス> 以下に例を挙げます。 switch#debug condition interface twentyFiveGigE 1/0/8	指定したインターフェイスの条件付きデバッグを設定します。

DHCPスヌーピングをデバッグするには、次の表に示すコマンドを使用します。

コマンド	目的
debug dhcp [detail   開く   redundancy]	dhcpパケットの内容の詳細 oper DHCP内部OPER

	冗長性DHCPクライアントの冗長性サポート
debug ip dhcp server packet detail	メッセージの受信と送信を詳細にデコードする
debug ip dhcp server events	住所の割り当て、リースの有効期限などをレポートします。 。
debug ip dhcp snooping agent	Debug dhcp snooping database read and write ( DHCPスヌーピングデータベースの読み取りと書き込みのデバッグ )
debug ip dhcp snooping event	各コンポーネント間のデバッグイベント
debug ip dhcp snooping packet	DHCPスヌーピングモジュールでのDHCPパケットのデバッグ

次に、debug ip dhcp snoopingコマンドの出力例の一部を示します。

<#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 reply to 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 flood
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: 0.0.0.0
Apr 14 16:16:48.840: DHCP_SNOOPING: add binding on port FortyGigabitEthernet1/0/2 ckt_id 0 FortyGigabitEthernet1/0/2
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 reply to output port: FortyGigabitEthernet1/0/2.
```

DHCPスヌーピングイベントをデバッグするには、次の手順を使用します。

---

 注意: debug コマンドは注意して使用してください。debug コマンドの多くは稼働中のネットワークに影響を与えるため、問題が再現されるラボ環境でのみ使用することを推奨します。

---

## 手順の概要

1. enable
2. debug platform condition mac {mac-address }
3. debug platform condition start
4. show platform condition または show debug
5. debug platform condition stop
6. show platform software trace message ios R0 reverse | DHCP を含める
7. clear platform condition all (プラットフォーム条件をすべてクリア)

## 手順の詳細

	コマンドまたはアクション	目的
手順 1	enable 以下に例を挙げます。 switch#enable	特権EXECモードを有効にします。 。 ・ パスワードを入力します ( 要求された場合 )。
手順 2	debug platform condition mac {mac-address} ( プラットフォームの状態をデバッグするmacアドレス ) 以下に例を挙げます。 switch#debug platform condition mac 0001.6509.3314	指定したMACアドレスの条件付きデバッグを設定します。
手順 3	debug platform condition start 以下に例を挙げます。 switch#debug platform condition start	条件付きデバッグを開始します ( いずれかの条件に一致する場合は、放射性トレースを開始できます )。
手順 4	show platform conditionまたはshow debug 以下に例を挙げます。 switch#show platform condition switch#show debug	現在の条件セットが表示されます。
手順 5	debug platform condition stop 以下に例を挙げます。 switch#debug platform condition stop	条件付きデバッグを停止します ( これにより、放射性トレースを停止できます )。
手順 6	show platform software trace message ios R0 reverse   DHCPを含める 以下に例を挙げます。 switch#show platform software trace message ios R0 reverse   DHCPを含める	最新のトレースファイルからマージされたHPログを表示します。 。

	コマンドまたはアクション	目的
ステップ7	clear platform condition all (プラットフォーム条件をすべてクリア) 以下に例を挙げます。  switch# clear platform condition all	すべての条件をクリアします。

dの出力例の一部を次に示しますBugプラットフォーム dhcp-snoop all コマンドを使用します。

<#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{mac})
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{mac})
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{mac})
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

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

次の表に、プラットフォームでDHCPスヌーピングのデバッグに使用できるさまざまなコマンドを示します。

 注意:debugコマンドは注意して使用してください。多くのdebugコマンドは実稼働中のネットワークに影響を与えるため、ラボ環境で問題が再現された場合にのみ使用することを推奨します。

コマンド	目的
switch#debug platform dhcp-snoop [all   パケット   pd-shim]	すべてのNGWC DHCPスヌーピング パケットNGWC DHCPスヌーピングパケット デバッグ情報  pd-shim NGWC DHCPスヌーピングIOS Shimデバッグ情報
switch#debug platform software infrastructure punt dhcp-snoop	FPで受信され、コントロールプレーンにパ ントされるパケット)
switch#debug platform software infrastructure inject	コントロールプレーンからFPに注入されるパ ケット

## パント/パストラフィック(CPU)のトラブルシューティング

FEDの観点からどのトラフィックが各CPUキューで受信されるかを確認します ( DHCPスヌーピングはコントロールプレーンで処理されるトラフィックのタイプです )。

- トラフィックはスイッチに着信すると、PUNT方向でCPUに送信され、dhcp snoopキューに送信されます。
- トラフィックがスイッチで処理されると、トラフィックはINJECT方向を経由して送信されます。 DHCP OFFERおよびACKパケットはL2制御/レガシーキューに分類されます。

<#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

このコマンドを使用して、CPUにパントされたトラフィックを確認し、DHCPスヌープがトラフィックをドロップしているかどうかを確認できます。

```
<#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	<<---- 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

## ハードウェアトラブルシューティング

### フォワーディングエンジンドライバ(FED)

FEDは、ASICをプログラムするドライバです。FEDコマンドは、ハードウェアとソフトウェアの状態が一致していることを確認するために使用されます。

DI\_Handle値を取得します

- DIハンドルは、特定のポートの宛先インデックスを参照します。

<#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マッピングをチェックして、ポートのAsicとコアを判別します。

- IFMは、特定のポート/コア/ASICにマッピングされた内部インターフェイスインデックスです。

<#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
```

DI\_Handleを使用して、ハードウェアインデックスを取得します。

<#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_DHCP Snooping  
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_index2:0x0  
<SNIP>  
<-- Index is 0x5f03
```

インデックス値0x5f03を16進数から10進数に変換します。

0x5f03 = 24323

このインデックス値を10進数で使用し、このコマンドのASIC値とコア値を使用して、ポートに設定されているフラグを確認します。

```
<#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
```

DHCPスヌーピングが特定のVLANに対して有効になっていることを確認します。

```
<#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
```



```
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>
```

次の表に、ライブネットワーク上のDHCPパケットのパスをトレースするために使用できるさまざまな一般的なPunject show/debugコマンドを示します。

#### 一般的なパント/インジェクトshowおよびdebugコマンド

```
debug plat soft fed swit acti inject add-filter cause 255 sub_cause 0 src_mac 0 0 dst_mac 0 0 0
src_ipv4 192.168.12.1 dst_ipv4 0.0.0 if_id 0xf

set platform software trace fed [switch<num|active|standby>] inject verbose — >表示されるフィルタcpmmandを使用して、トレースをこの特定のホストに送ります

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 ( 登録ユーザ専用 )

show platform software infrastructureパント

show platform software infrastructure lsmipiドライバ

debug platform software infra punt dhcp

debug platform software infra inject
```

これらのコマンドは、特定のクライアントに対してDHCPパケットが受信されているかどうかを確認するのに役立ちます。

- この機能を使用すると、IOS-DHCPソフトウェアを介してCPUによって処理される、特定のクライアントMACアドレスに関連付けられたすべてのDHCPスヌーピング通信をキャプチャできます。
- この機能は、IPv4とIPv6の両方のトラフィックでサポートされます。
- この機能は自動的に有効になります。

 **重要**：これらのコマンドは、Cisco IOS XE Gibraltar 16.12.Xから使用できます。

```
switch#show platform dhcp snooping client stats {mac-address}
```

```
switch#show platform dhcpv6 snooping ipv6 client stats {mac-address}
```

<#root>

C9300#

```
show platform dhcp snooping client stats 0000.1AC2.C148
```

DHCP SN: DHCP snooping server

DHCPD: DHCP protocol daemon

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	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_DHCP SN
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_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

トレースをクリアするには、次のコマンドを使用します。


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

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

## CPUパスパケットキャプチャ

DHCPスヌーピングパケットが到着したかどうかを確認し、コントロールプレーンを正しく離します。

---

 注:フォワーディングエンジンドライバCPUキャプチャツールの使用方法に関する追加の参考資料については、「その他の資料」のセクションを参照してください。

---

```
<#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 : 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

## 便利なトレース

これらは、プロセスまたはコンポーネントごとのイベントを表示するバイナリトレースです。この例では、トレースはdhcpcsnコンポーネントに関する情報を示します。

- トレースは手動で回転できます。つまり、トラブルシューティングを開始する前に新しいファイルを作成して、よりクリーンな情報を含めることができます。

<#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}: [dhcpcn] [17035]: (debug): Program trust ports for this vlan

2021/04/14 19:24:19.159978 {fed\_F0-0}{1}: [dhcpcn] [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}: [dhcpcn] [17035]: (debug): trusted\_if\_q size=1 for vlan=10

2021/04/14 19:24:19.160041 {fed\_F0-0}{1}: [dhcpcn] [17035]: (ERR): update ri has failed vlanid[10]

2021/04/14 19:24:19.160042 {fed\_F0-0}{1}: [dhcpcn] [17035]: (debug): vlan mode changed to enable

2021/04/14 19:24:27.507358 {fed\_F0-0}{1}: [dhcpcn] [23451]: (debug): get di for vlan\_id 10

2021/04/14 19:24:27.507365 {fed\_F0-0}{1}: [dhcpcn] [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 dhcpcn fai

2021/04/14 19:24:29.511774 {fed\_F0-0}{1}: [dhcpcn] [23451]: (debug): get di for vlan\_id 10

2021/04/14 19:24:29.511780 {fed\_F0-0}{1}: [dhcpcn] [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 dhcpcn 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と説明

DHCPレート制限の違反。

説明：DHCPスヌーピングにより、指定されたインターフェイスでDHCPパケットのレート制限違反が検出されました。

```
%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
```

信頼できないポートでのDHCPサーバスプーフィング

説明：DHCPスヌーピング機能によって、信頼できないインターフェイスで特定のタイプのDHCPメッセージが許可されていないことが検出されました。これは、一部のホストがDHCPサーバとして動作しようとしていることを示しています。

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

レイヤ2 MACアドレスがDHCP要求の内部のMACアドレスと一致しない。

説明：DHCPスヌーピング機能によってMACアドレスの検証が試行され、チェックが失敗しました。イーサネットヘッダーの送信元MACアドレスが、DHCP要求メッセージのchaddrフィールドのアドレスと一致しない。DHCPサーバでサービス拒否攻撃を実行しようとする悪意のあるホストが存在する可能性があります。

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

オプション82の挿入の問題

説明：DHCPスヌーピング機能によって、信頼できないポートで許可されていないオプション値を持つDHCPパケットが検出されました。これは、一部のホストがDHCPリレーまたはサーバとして動作しようとしていることを示しています。

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

レイヤ2 MACアドレスが誤ったポートで受信された。

説明：DHCPスヌーピング機能によって、ネットワーク内の別のホストに対してサービス拒否攻撃を実行しようとしているホストが検出されました。

%DHCP\_SNOOPING-5-DHCP\_SNOOPING\_FAKE\_INTERFACE: DHCP\_SNOOPING drop message with mismatched source interface

信頼できないインターフェイスで受信されたDHCPメッセージ。

説明：DHCPスヌーピング機能によって、信頼できないインターフェイスで特定のタイプのDHCPメッセージが許可されていないことが検出されました。これは、一部のホストがDHCPサーバとして動作しようとしていることを示しています。

%DHCP\_SNOOPING-5-DHCP\_SNOOPING\_UNTRUSTED\_PORT: DHCP\_SNOOPING drop message on untrusted port: GigabitEthernet

DHCPスヌーピングの転送に失敗しました。URLにアクセスできません。

説明：DHCPスヌーピングバインディングの転送に失敗しました。

%DHCP\_SNOOPING-4-AGENT\_OPERATION\_FAILED: DHCP snooping binding transfer failed. Unable to access URL

## DHCPスヌーピングの警告


Cisco Bug ID番号	説明
<a href="#">CSCvi39202</a>	アップリンクetherchannelでDHCPスヌーピング信頼が有効になっていると、DHCPは失敗します。
<a href="#">CSCvp49518</a>	リロード後、DHCPスヌーピングデータベースは更新されません。
<a href="#">CSCvk16813</a>	DHCPスヌーピングとポートチャネルまたはクロススタックアップリンクを使用してDHCPクライアントトラフィックをドロップする。
<a href="#">CSCvd51480</a>	ip dhcpスヌーピングとデバイストラッキングのバインド解除

<a href="#">CSCvm55401</a>	DHCPスヌーピングは、ip dhcp snooping information option allow-untrustedを使用して、dhcpオプション82のパケットをドロップできる
<a href="#">CSCvx25841</a>	REPセグメントに変更があると、DHCPスヌーピングの信頼状態が切断されま す。
<a href="#">CSCvs15759</a>	DHCPサーバは、DHCP更新プロセス中にNAKパケットを送信します。
<a href="#">CSCvk34927</a>	リロード時にDHCPスヌーピングDBファイルからDHCPスヌーピングテーブル が更新されない

## SDAボーダーDHCPスヌーピング

DHCPスヌーピング統計情報CLI。

DHCPスヌーピングの統計情報を確認するSDA用の新しいCLI。

 注: Cisco SDアクセスファブリックエッジのDHCPプロセス/パケットフローおよびデコードに関するその他の参考資料については、「関連情報」セクションのガイドを参照してください。

```
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

## 関連情報

[IPアドレッシングサービス設定ガイド、Cisco IOS XE Amsterdam 17.3.x \( Catalyst 9200スイッチ \)](#)

[IPアドレッシングサービス設定ガイド、Cisco IOS XE Amsterdam 17.3.x \( Catalyst 9300スイッチ \)](#)

[『IP Addressing Services Configuration Guide, Cisco IOS XE Amsterdam 17.3.x』 \( Catalyst 9400スイッチ \)](#)

[IPアドレッシングサービス設定ガイド、Cisco IOS XE Amsterdam 17.3.x \( Catalyst 9500スイッチ \)](#)

[IPアドレッシングサービス設定ガイド、Cisco IOS XE Amsterdam 17.3.x \( Catalyst 9600スイッチ \)](#)

[Cisco SDアクセスファブリックエッジのDHCPプロセス/パケットフローとデコード](#)

[Catalyst 9000スイッチでのFED CPUパケットキャプチャの設定](#)

[テクニカル サポートとドキュメント - Cisco Systems](#)



## 翻訳について

シスコは世界中のユーザにそれぞれの言語でサポート コンテンツを提供するために、機械と人による翻訳を組み合わせて、本ドキュメントを翻訳しています。ただし、最高度の機械翻訳であっても、専門家による翻訳のような正確性は確保されません。シスコは、これら翻訳の正確性について法的責任を負いません。原典である英語版（リンクからアクセス可能）もあわせて参照することを推奨します。