Catalyst 9000スイッチでのコントロールプレーン動作のトラブルシューティング

内容

はじめに

このドキュメントでは、Cisco IOS® XEを実行するCatalyst 9000ファミリスイッチのコントロールプレーンの健全性をトラブルシューティングし、検証する方法について説明します。

背景説明

スイッチの主な役割は、パケットをできるだけ迅速に転送することです。ほとんどのパケットはハードウェアで転送されますが、特定のタイプのトラフィックはシステムCPUで処理する必要があります。CPUに到達するトラフィックは、可能な限り迅速に処理されます。ある程度の量のトラフィックがCPUで発生すると予想されますが、過剰なトラフィックは運用上の問題につながります。Catalyst 9000ファミリのスイッチには、CPUのトラフィック過飽和による問題を防ぐために、堅牢なコントロールプレーンポリシング(CoPP)メカニズムがデフォルトで組み込まれています。

予期しない問題は、通常の動作の関数として特定のユースケースで発生します。原因と結果の相関関係が明確でないことがあり、その場合、問題の解決が困難になります。このドキュメントで

は、コントロールプレーンの健全性を検証するツールについて説明し、コントロールプレーンのパントまたは挿入パスに関連する問題に対処するワークフローについて説明します。また、現場で見られる問題に基づく一般的なシナリオもいくつか示します。

CPUパントパスは限られたリソースであることに注意してください。最近のハードウェア転送スイッチは、急増するトラフィックを処理できます。Catalyst 9000ファミリのスイッチは、任意の時点でCPU全体で約19,000パケット/秒(pps)をサポートします。このしきい値を超えると、パントされたトラフィックは重みなしでポリシングされます。

用語

- Forwarding Engine Driver(FED)：これはCisco Catalystスイッチの中心であり、すべてのハードウェアのプログラミング/転送を行います
- IOSd:Linuxカーネルで動作するCisco IOSデーモンです。カーネル内のソフトウェアプロセスとして実行される
- パケット配信システム(PDS)：これは、さまざまなサブシステムとの間でパケットを送受信する方法のアーキテクチャとプロセスです。たとえば、FEDからIOSdへのパケットの配信方法を制御したり、その逆を制御したりします
- コントロールプレーン(CP):コントロールプレーンは、CatalystスイッチのCPUに関連する機能とトラフィックをグループ化するために使用される一般用語です。これには、スパニングツリープロトコル(STP)、ホットスタンバイルータプロトコル(HSRP)、スイッチを宛先とする、またはスイッチから送信されるルーティングプロトコルなどのトラフィックが含まれます。これには、CPUで処理する必要があるセキュアシェル(SSH)や簡易ネットワーク管理プロトコル(SNMP)などのアプリケーション層プロトコルも含まれます
- データプレーン(DP):通常、データプレーンには、コントロールプレーンの支援なしで転送されるハードウェアASICとトラフィックが含まれます
- パント：CPに送信されたDPを代行受信して処理する、入力プロトコル制御パケット
- Inject:CPで生成されたプロトコルパケットがDPに送信され、IOインターフェイスから出力される
- LSMPI:Linux共有メモリパントインターフェイス

Catalyst 9000のCoPP

Catalyst 9000ファミリスイッチのCPU保護の基盤はCoPPです。CoPPを使用すると、システム生成のQuality of Service(QoS)ポリシーがCPUパント/インジェクトパスに適用されます。CPUに送られるトラフィックは多くの異なるクラスにグループ化され、その後CPUに関連付けられた個々のハードウェアポリサーにマップされます。ポリサーは、特定のトラフィッククラスによるCPUの過飽和を防ぎます。

CoPPの実装

CPUに送られるトラフィックはキューに分類されます。これらのキュー/クラスはシステムによって定義され、ユーザによる設定はできません。ポリサーはハードウェアで設定されます。Catalyst 9000ファミリでは、32のキューに対して32のハードウェアポリサーがサポートされています。

具体的な値はプラットフォームによって異なります。一般に、システム定義キューは32個あります。これらのキューは、ポリサーインデックスに関連するクラスマップに関連します。ポリサーインデックスには、デフォルトのポリサーレートがあります。このレートはユーザが設定できますが、デフォルトのCoPPポリシーを変更すると、予期しないサービスの影響を受けやすくなります。

CoPPのシステム定義値

クラスマップ名	ポリサーインデックス（ポリサー番号）	CPUキュー数
system-cpp-police-data（ポリシングデータ）	WK_CPP_POLICE_DATA(0)	WK_CPU_Q_ICMP_GEN(3) WK_CPU_Q_BROADCAST(12) WK_CPU_Q_ICMP_REDIRECT(6)
system-cpp-police-l2 – 制御	WK_CPP_POLICE_L2_制御(1)	WK_CPU_Q_L2_CONTROL(1)
system-cpp-police-routing-control（システム – cpp – ポリシング – ルーティング – 制御）	WK_CPP_POLICE_ROUTING_CONTROL(2)	WK_CPU_Q_ROUTING_CONTROL(4) WK_CPU_Q_LOW_LATENCY(27)
system-cpp-police-control-low-priority（優先順位の低いシステム）	WK_CPP_POLICE_CO CONTROL_LOW_PRI(3)	WK_CPU_Q_GENERAL_PUNT(25)
system-cpp-police-punt-webauth（オプション）	WK_CPP_POLICE_PU NT_WEBAUTH(7)	WK_CPU_Q_PUNT_WEBAUTH(22)
system-cpp-police-topology-control	WK_CPP_POLICE_TOPOLOGY_CONTROL(8)	WK_CPU_Q_TOPOLOGY_CONTROL(1
system-cpp-police-multicast	WK_CPP_POLICE_MULTICAST(9)	WK_CPU_Q_TRANSIT_トラフィック(18 WK_CPU_Q_MCAST_DATA(30)

クラスマップ名	ポリサーインデックス（ポリサー番号）	CPUキュー数
system-cpp-police-sys – データ	WK_CPP_POLICE_SYS _DATA(10)	WK_CPU_Q_LEARNING_CACHE_OVFL WK_CPU_Q_CRYPTO_CONTROL(23) WK_CPU_Q_EXCEPTION(24) WK_CPU_Q_EGR_EXCEPTION(28) WK_CPU_Q_NFL_SAMPLED_DATA(26) WK_CPU_Q_GOLD_PKT(31) WK_CPU_Q_RPF_FAILED(19)
system-cpp-police-dot1x-auth（オプション）	WK_CPP_POLICE_DOT1X(11)	WK_CPU_Q_DOT1X_AUTH(0)
system-cpp-police-protocol-snooping（プロトコルスヌーピング）	WK_CPP_POLICE_PR(12)	WK_CPU_Q_PROTO_SNOOPING(16)
system-cpp-police-sw-forward（オプション）	WK_CPP_POLICE_SW_FWD (13)	WK_CPU_Q_SW_FORWARDING_Q(14) WK_CPU_Q_LOGGING(21) WK_CPU_Q_L2_LVX_DATA_PACK(11)
system-cpp-police-forus（ポリシングフォーラム）	WK_CPP_POLICE_FORUS(14)	WK_CPU_Q_FORUS_ADDR_RESOLUT WK_CPU_Q_FORUS_TRAFFIC(2)
system-cpp-police-multicast-end-station（マルチキャスト端末）	WK_CPP_POLICE_MULTICAST_SNOOPING(15)	WK_CPU_Q_MCAST_END_STA TION_SERVICE(20)

クラスマップ名	ポリサーインデックス (ポリサー番号)	CPUキュー数
システムcppデフォルト	WK_CPP_POLICE_DEFAULT_POLICER(16)	WK_CPU_Q_DHCP_SNOOPING(17) WK_CPU_Q_UNUSED(7) WK_CPU_Q_EWLC_CONTROL(9) WK_CPU_Q_EWLC_DATA(10)
system-cpp-police-stackwise-virt-control	WK_CPP_STACKWISE_VIRTUAL_CONTROL(5)	WK_CPU_Q_STACKWISE_VIRTUAL_C
system-cpp-police-l2lvx-control	WK_CPP_ L2_LVX_CONT_PACK(4)	WK_CPU_Q_L2_LVX_CONT_PACK(8)

各キューは、トラフィックタイプまたは特定のフィーチャセットに関連します。これは完全なリストではありません。

CPUキューおよび関連機能

CPUキュー数	機能
WK_CPU_Q_DOT1X_AUTH(0)	IEEE 802.1xポートベース認証
WK_CPU_Q_L2_CONTROL(1)	Dynamic Trunking Protocol (DTP; ダイナミック トランキング プロトコル) VLAN Trunking Protocol (VTP; VLAN トランキング プロトコル) Port Aggregation Protocol (PAgP) クライアント情報シグナリングプロトコル(CISP) メッセージセッションリレープロトコル マルチVLAN登録プロトコル(MVRP) メトロポリタンモバイルネットワーク(MMN)

CPUキュー数	機能
	リンクレベル検出プロトコル(LLDP) 単方向リンク検出（UDLD） Link Aggregation Control Protocol（LACP） シスコ検出プロトコル（CDP） スパニング ツリー プロトコル（STP）
WK_CPU_Q_FORUS_TRAFFIC(2)	Telnet、Pingv4、Pingv6、SNMPなどのホスト キープアライブ/ループバック検出 初期インターネットキーエクスチェンジ(IKE)プロトコル(IPSec)
WK_CPU_Q_ICMP_GEN(3)	ICMP – 宛先到達不能 ICMP-TTL期限切れ
WK_CPU_Q_ROUTING_CONTROL(4)	Routing Information Protocolバージョン 1(RIPv1) RIPv2 Interior Gateway Routing Protocol（IGRP） ボーダー ゲートウェイ プロトコル（BGP） PIM-UDP（必須） Virtual Router Redundancy Protocol（VRRP; 仮想ルータ冗長プロトコル） ホットスタンバイルータプロトコルバージョン 1(HSRPv1) HSRPv2 ゲートウェイロードバランシングプロトコル(GLBP) ラベル配布プロトコル(LDP) Webキャッシュ通信プロトコル(WCCP)

CPUキュー数	機能
	次世代ルーティング情報プロトコル(RIPng) Open Shortest Path First（OSPF） Open Shortest Path Firstバージョン3(OSPFv3) Enhanced Interior Gateway Routing Protocol（EIGRP） Enhanced Interior Gateway Routing Protocolバージョン6(EIGRPv6) DHCPv6 Protocol Independent Multicast（PIM） Protocol Independent Multicast(PIMv)バージョン6(PIMv6) 次世代ホットスタンバイルータプロトコル(HSRPng) IPv6制御 総称ルーティングカプセル化(GRE)キープアライブ ネットワークアドレス変換(NAT)パント Intermediate System-to-Intermediate System（IS-IS）
WK_CPU_Q_FORUS_ADDR_RESOLUTION(5)	アドレス解決プロトコル(ARP) IPv6ネイバーアドバタイズメントとネイバー送信要求
WK_CPU_Q_ICMP_REDIRECT(6)	インターネット制御メッセージプロトコル(ICMP)リダイレクト
WK_CPU_Q_INTER_FED_TRAFFIC(7)	レイヤ2ブリッジドメインは内部通信用に挿入されます。
WK_CPU_Q_L2_LVX_CONT_PACK(8)	Exchange ID(XID)パケット

CPUキュー数	機能
WK_CPU_Q_EWLC_CONTROL(9)	組み込み型ワイヤレスコントローラ(eWLC)[Control and Provisioning of Wireless Access Points(CAPWAP)(UDP 5246)]
WK_CPU_Q_EWLC_DATA(10)	eWLCデータパケット（CAPWAPデータ、UDP 5247）
WK_CPU_Q_L2_LVX_DATA_PACK(11)	マップ要求のためにパントされた不明なユニキャストパケット。
WK_CPU_Q_BROADCAST(12)	あらゆる種類のブロードキャスト
WK_CPU_Q_OPENFLOW(13)	学習キャッシュのオーバーフロー（レイヤ2 +レイヤ3）
WK_CPU_Q_CONTROLLER_PUNT(14)	データ – アクセスコントロールリスト(ACL)がいっぱいです データ – IPv4オプション データ – IPv6ホップバイホップ データ – リソース不足/すべてを捕捉 データ – リバースパス転送(RPF)が不完全です 収集パケット
WK_CPU_Q_TOPOLOGY_CONTROL(15)	スパニング ツリー プロトコル（STP） Resilient Ethernet Protocol（REP；復元イーサネットプロトコル） 共有スパニングツリープロトコル(SSTP)
WK_CPU_Q_PROTO_SNOOPING(16)	ダイナミックARPインスペクション(DAI)のためのアドレス解決プロトコル(ARP)スヌーピング
WK_CPU_Q_DHCP_SNOOPING(17)	DHCP スヌーピング

CPUキュー数	機能
WK_CPU_Q_TRANSIT_トラフィック(18)	これは、ソフトウェアパスで処理する必要があるNATによってパントされるパケットに使用されます。
WK_CPU_Q_RPF_FAILED(19)	データ – mRPF（マルチキャストRPF）に失敗しました
WK_CPU_Q_MCAST_END_STATION _サービス(20)	Internet Group Management Protocol(IGMP)/マルチキャストリスナー検出(MLD)制御
WK_CPU_Q_LOGGING(21)	アクセスコントロールリスト(ACL)ロギング
WK_CPU_Q_PUNT_WEBAUTH(22)	Web 認証
WK_CPU_Q_HIGH_RATE_APP(23)	ブロードキャスト
WK_CPU_Q_EXCEPTION(24)	IKE表示 IPラーニング違反 IPポートセキュリティ違反 IPスタティックアドレス違反 IPv6スコープチェック リモートコピープロトコル(RCP)例外 ユニキャストRPFの失敗
WK_CPU_Q_SYSTEM_CRITICAL(25)	メディアシグナリング/ワイヤレスプロキシARP
WK_CPU_Q_NFL_SAMPLED_DATA(26)	Netflowサンプルデータおよびメディアサービスプロキシ(MSP)
WK_CPU_Q_LOW_LATENCY(27)	双方向フォワーディング検出(BFD)、高精度時間プロトコル(PTP)

CPUキュー数	機能
WK_CPU_Q_EGR_EXCEPTION(28)	出力解決の例外
WK_CPU_Q_STACKWISE_仮想_制御(29)	前面スタッキングプロトコル(SVL)
WK_CPU_Q_MCAST_DATA(30)	データ – (S,G)の作成 データ – ローカル結合 データ – PIM登録 データ – SPTスイッチオーバー データ – マルチキャスト
WK_CPU_Q_GOLD_PKT(31)	ゴールド

Default policy

デフォルトでは、システムで生成されたCoPPポリシーがパント/インジェクトパスに適用されます。デフォルトポリシーは、一般的なMQCベースのコマンドを使用して表示できます。また、スイッチの設定内でも確認できます。CPU/コントロールプレーンの入力または出力に適用できるポリシーは、システム定義ポリシーのみです。

「show policy-map control-plane」を使用して、コントロールプレーンに適用されているポリシーを表示します。

<#root>

```
Catalyst-9600#
show policy-map control-plane


Control Plane

 Service-policy input: system-cpp-policy

   Class-map: system-cpp-police-ios-routing (match-any)
      0 packets, 0 bytes
      5 minute offered rate 0000 bps, drop rate 0000 bps
      Match: none
      police:
          rate 17000 pps, burst 4150 packets
        conformed 95904305 bytes; actions:
          transmit
        exceeded 0 bytes; actions:
          drop
```

```
<snip>

   Class-map: class-default (match-any)
      0 packets, 0 bytes
      5 minute offered rate 0000 bps, drop rate 0000 bps
      Match: any
```

CoPPの調整

CoPPポリサーレートはユーザが設定できます。また、キューを無効にすることもできます。

この例では、個々のポリサー値を調整する方法を示します。この例では、調整後のクラスは「system-cpp-police-protocol-snooping」です。

```
<#root>

Device>

enable

Device#

configure terminal


Device(config)#

policy-map system-cpp-policy

Device(config-pmap)#

Device(config-pmap)#

class system-cpp-police-protocol-snooping

Device(config-pmap-c)#

Device(config-pmap-c)#

police rate 100 pps

Device(config-pmap-c-police)#

Device(config-pmap-c-police)#

exit

Device(config-pmap-c)#

exit
```

```
Device(config-pmap)#
exit

Device(config)#
Device(config)#
control-plane

Device(config-cp)#
Device(config)#
control-plane

Device(config-cp)#
service-policy input system-cpp-policy

Device(config-cp)#
Device(config-cp)#
end

Device#
show policy-map control-plane
```

次の例は、キューを完全に無効にする方法を示しています。キューを無効にする場合は、CPUが過飽和する可能性があるため、注意が必要です。

```
<#root>
Device>
enable

Device#
configure terminal

Device(config)#
policy-map system-cpp-policy

Device(config-pmap)#
Device(config-pmap)#
class system-cpp-police-protocol-snooping
```

```
Device(config-pmap-c)#


Device(config-pmap-c)#
no police rate 100 pps
Device(config-pmap-c)#
end
```

トラブルシュート

方法

CPU使用率は、プロセスと中断という2つの基本的なアクティビティの影響を受けます。プロセスは構造化されたアクティビティを実行し、割り込みはデータプレーンでインターセプトされ、アクションのためにCPUに送信されるパケットを指します。これらのアクティビティは、CPUの総使用率を構成します。CoPPはデフォルトで有効になっているため、サービスへの影響と高いCPU使用率は必ずしも相関しません。CoPPがジョブを実行する場合、CPU使用率に大きな影響はありません。CPUの全体的な使用率を考慮することは重要ですが、全体的な使用率はストーリー全体を示しているわけではありません。このセクションのshowコマンドとユーティリティは、CPUの状態を迅速に評価し、CPUに送られるトラフィックに関する詳細情報を特定するために使用されます。

ガイドライン：

- 問題がコントロールプレーンに関連しているかどうかを確認します。ほとんどの中継トラフィックはハードウェアで転送されます。CPUとコントロールプレーンが関係するのは特定のトラフィックタイプと特定のシナリオだけなので、調査全体を通じてこの点に留意してください。
- 使用率のベースラインを把握します。ノルムからの逸脱を特定できるように、通常の使用率を理解することが重要です。
- プロセスと割り込みの両方の全体的な使用率を検証します。予期しない量のCPUサイクルを占有するプロセスを特定します。使用率が期待値の範囲外になると、問題が生じる可能性があります。ノルム外の偏差が認識されるように、システムの平均使用率を理解することが重要です。使用率のみではコントロールプレーンの健全性を完全に把握できないことに注意してください。
- CoPPでドロップが積極的に増加しているかどうかを判断します。CoPPドロップは必ずしも問題を示すものではありませんが、アクティブにポリシングされているトラフィッククラスに関連する問題をトラブルシューティングする場合、これは関連性の強いインジケータになります。

便利な show コマンド

このスイッチでは、CPUの健全性とCoPPの統計情報をすばやく監視できます。また、CPUに送られるトラフィックの入り口を迅速に特定する便利なCLIもあります。

全体的な使用率と使用率の履歴を確認する

- 「Show processes cpu sorted」は、全体的なCPU使用率を表示するために使用します。「sorted」引数は、使用率に基づいてプロセス出力を並べ替えます。より多くのCPUリソースを使用するプロセスが出力の先頭に表示されます。割り込みによる使用率もパーセンテージで表示されます。

<#root>

```
Catalyst-9600#

show processes cpu sorted


CPU utilization for five seconds: 92%/13%; one minute: 76%; five minutes: 73%

<<<--- Utilization is displayed for 5 second (both process and interrupt), 1 minute and 5 minute interva


                                                                                92% refers to the c


                                                                                The 13% value refer


PID Runtime(ms)    Invoked      uSecs   5Sec   1Min   5Min  TTY  Process

<<<--- Runtime statistics, as well as utilization averages are displayed here. The process is also ident


344   547030523  607054509        901  38.13% 30.61% 29.32%   0  SISF Switcher Th
345   394700227  615024099        641  31.18% 22.68% 21.66%   0  SISF Main Thread
 98   112308516  119818535        937   4.12%  4.76%  5.09%   0  Crimson flush tr
247    47096761   92250875        510   2.42%  2.21%  2.18%   0  Spanning Tree
123    35303496  679878082         51   1.85%  1.88%  1.84%   0  IOSXE-RP Punt Se
234         955       1758        543   1.61%  0.71%  0.23%   3  SSH Process
547     5360168    5484910        977   1.04%  0.46%  0.44%   0  DHCPD Receive
229    27381066  963726156         28   1.04%  1.34%  1.23%   0 IP Input
 79    13183805  108951712        121   0.48%  0.55%  0.55%   0 IOSD ipc task
  9     1073134     315186       3404   0.40%  0.06%  0.03%   0 Check heaps
 37    11099063  147506419         75   0.40%  0.54%  0.52%   0 ARP Input
312     2986160  240782059         12   0.24%  0.12%  0.14%   0 DAI Packet Proce
<snip>
565           0          1          0   0.00%  0.00%  0.00%   0 LICENSE AGENT
566          14       1210         11   0.00%  0.00%  0.00%   0 DHCPD Timer
567          40         45        888   0.00%  0.00%  0.00%   0 OVLD SPA Backgro
568          12       2342          5   0.00%  0.00%  0.00%   0 DHCPD Database
569           0         12          0   0.00%  0.00%  0.00%   0 SpanTree Flush
571           0          1          0   0.00%  0.00%  0.00%   0 EM Action CNS
572         681     140276          4   0.00%  0.00%  0.00%   0 Inline power inc
```

- 「Show processes cpu history」では、過去60秒、5分、72時間のCPU使用率の履歴グラフが提供されます。

<#root>

```
Catalyst-9600#
```

show processes cpu history

```
    9997777766666888886666677777777778888877777666669999988888866
```

<<<--- The numbers at the top of each column represent the highest value seen throughout the time period

```
    2225555599999444444444000008888888881111177777333335555500
```

It is read top-down. "9" over "2" in this example means "92%" for example.

```
100
 90  *** ***** **********
 80  ******** ***** ********** **********
 70  ****************** ***********************************
 60  ************************************************************
 50  ************************************************************
 40  ************************************************************
 30  ************************************************************
 20  ************************************************************
 10  ************************************************************
```

<<<--- The "*" represents the highest value during the given time period. This relates to a momentary sp

```
0....5....1....1....2....2....3....3....4....4....5....5....6
```

In this example, utilization spiked to 92% in the last 5 seconds.

```
          0    5    0    5    0    5    0    5    0    5    0
          CPU% per second (last 60 seconds)
         * = maximum CPU% # = average CPU%




    9998989899998989989989989889999989889898899999999899999999
    4318230911026353162351292837713365748928096040142309011335 11
100 ** *
 90  ***** **************************************************
 80  ************#***#*#**#***####*##******#**#***#***#**********
 70  ############################################################
```

<<<--- The "#" represents the average utilization. This indicates sustained utilization.

```
 60  ############################################################
```

In this example, within the last 5 minutes the average utilization was sustained around 70% while

```
 50  ############################################################
```

the maximum utilization spiked to 94%.

```
 40  ############################################################
 30  ############################################################
 20  ############################################################
 10  ############################################################
```

```
0....5....1....1....2....2....3....3....4....4....5....5....6
          0    5    0    5    0    5    0    5    0    5    0
          CPU% per minute (last 60 minutes)
         * = maximum CPU% # = average CPU%




    9999999999999999999999999999999999999999999999999999999999999999999999999999
    6656565666465556666556565756545565677375555675745455457759575546485767 57
100 ********** ****************** ******* ********* * ** ********* * *****
90  ************************************************************************
80  ************************************************************************
70  ########################################################################
60  ########################################################################
50  ########################################################################
40  ########################################################################
30  ########################################################################
20  ########################################################################
10  ########################################################################
0....5....1....1....2....2....3....3....4....4....5....5....6....6....7..
          0    5    0    5    0    5    0    5    0    5    0    5    0
               CPU% per hour (last 72 hours)
              * = maximum CPU% # = average CPU%
```

コントロールプレーンポリシングのチェック

- 「show platform hardware fed <switch> active qos queue stats internal cpu policer」を使用して、集約CoPP統計情報と、キュー/ポリサー構造に関する追加情報を表示します。この出力は、コントロールプレーンが最後にリセットされた後のポリサー統計情報の履歴を示します。これらのカウンタも手動でクリアできます。 一般に、ポリサーによるコントロールプレーンのドロップの証拠は、関連するキュー/クラスに問題があることを示していますが、問題が発生している間はドロップがアクティブに増加することを確認してください。コマンドを何度か実行して、キューのドロップ値の増加を観察します。

<#root>

Catalyst9500#

show platform hardware fed active qos queue stats internal cpu policer

```
                          CPU Queue Statistics
============================================================================================
                                                 (default) (set)     Queue         Queue
QId PlcIdx  Queue Name               Enabled   Rate     Rate      Drop(Bytes)  Drop(Frames)

<-- The top section of this output gives a historical view of CoPP drops. Run the command several times
```

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

CPU queues correlate with a Policer Index (PlcIdx) and Queue (QId).

0    11     DOT1X Auth                  Yes     1000      1000     0           0

Note that multiple policer indices map to the same queue for some classes.

1    1      L2 Control                  Yes     2000      2000     0           0
2    14     Forus traffic               Yes     4000      4000     0           0
3    0      ICMP GEN                    Yes     750       750      0           0
4    2      Routing Control             Yes     5500      5500     0           0
5    14     Forus Address resolution    Yes     4000      4000     83027876    1297199
6    0      ICMP Redirect               Yes     750       750      0           0
7    16     Inter FED Traffic           Yes     2000      2000     0           0
8    4      L2 LVX Cont Pack            Yes     1000      1000     0           0
9    19     EWLC Control                Yes     13000     13000    0           0
10   16     EWLC Data                   Yes     2000      2000     0           0
11   13     L2 LVX Data Pack            Yes     1000      1000     0           0
12   0      BROADCAST                   Yes     750       750      0           0
13   10     Openflow                    Yes     250       250      0           0
14   13     Sw forwarding               Yes     1000      1000     0           0
15   8      Topology Control            Yes     13000     16000    0           0
16   12     Proto Snooping              Yes     2000      2000     0           0
17   6      DHCP Snooping               Yes     500       500      0           0
18   13     Transit Traffic             Yes     1000      1000     0           0
19   10     RPF Failed                  Yes     250       250      0           0
20   15     MCAST END STATION           Yes     2000      2000     0           0
21   13     LOGGING                     Yes     1000      1000     769024      12016
22   7      Punt Webauth                Yes     1000      1000     0           0
23   18     High Rate App               Yes     13000     13000    0           0
24   10     Exception                   Yes     250       250      0           0
25   3      System Critical             Yes     1000      1000     0           0
26   10     NFL SAMPLED DATA            Yes     250       250      0           0
27   2      Low Latency                 Yes     5500      5500     0           0
28   10     EGR Exception               Yes     250       250      0           0
29   5      Stackwise Virtual OOB       Yes     8000      8000     0           0
30   9      MCAST Data                  Yes     500       500      0           0
31   3      Gold Pkt                    Yes     1000      1000     0           0

* NOTE: CPU queue policer rates are configured to the closest hardware supported value

                      CPU Queue Policer Statistics
=============================================================================
Policer    Policer Accept   Policer Accept  Policer Drop  Policer Drop
  Index         Bytes           Frames         Bytes           Frames
-----------------------------------------------------------------------------
0          59894            613             0             0
1          15701689         57082           0             0
2          5562892          63482           0             0
3          3536             52              0             0
4          0                0               0             0
5          0                0               0             0
6          0                0               0             0
7          0                0               0             0
8          2347194476       32649666        0             0
9          0                0               0             0
10         0                0               0             0
11         0                0               0             0
12         0                0               0             0
13         577043           8232            769024        12016
14         719225176        11182355        83027876      1297199
15         132766           1891            0             0
```

```
16          0                0                0              0
17          0                0                0              0
18          0                0                0              0
19          0                0                0              0

                    Second Level Policer Statistics

<-- Second level policer information begins here. Catalyst CoPP is organized with two policers to allow

=====================================================================
20          2368459057       32770230         0              0
21          719994879        11193091         0              0

Policer Index Mapping and Settings
---------------------------------------------------------------------
level-2   :   level-1                      (default)   (set)
PlcIndex  :   PlcIndex                       rate       rate
---------------------------------------------------------------------
20        :   1  2  8                        13000      17000
21        :   0 4 7 9 10 11 12 13 14 15      6000       6000
=====================================================================

                Second Level Policer Config
=====================================================================
    level-1 level-2                              level-2
QId PlcIdx  PlcIdx  Queue Name                   Enabled
---------------------------------------------------------------------
0    11      21      DOT1X Auth                   Yes
1    1       20      L2 Control                   Yes
2    14      21      Forus traffic                Yes
3    0       21      ICMP GEN                     Yes
4    2       20      Routing Control              Yes
5    14      21      Forus Address resolution     Yes
6    0       21      ICMP Redirect                Yes
7    16      -       Inter FED Traffic            No
8    4       21      L2 LVX Cont Pack             Yes
9    19      -       EWLC Control                 No
10   16      -       EWLC Data                    No
11   13      21      L2 LVX Data Pack             Yes
12   0       21      BROADCAST                    Yes
13   10      21      Openflow                     Yes
14   13      21      Sw forwarding                Yes
15   8       20      Topology Control             Yes
16   12      21      Proto Snooping               Yes
17   6       -       DHCP Snooping                No
18   13      21      Transit Traffic              Yes
19   10      21      RPF Failed                   Yes
20   15      21      MCAST END STATION            Yes
21   13      21      LOGGING                      Yes
22   7       21      Punt Webauth                 Yes
23   18      -       High Rate App                No
24   10      21      Exception                    Yes
25   3       -       System Critical              No
26   10      21      NFL SAMPLED DATA             Yes
27   2       20      Low Latency                  Yes
28   10      21      EGR Exception                Yes
29   5       -       Stackwise Virtual OOB        No
30   9       21      MCAST Data                   Yes
31   3       -       Gold Pkt                     No

                        CPP Classes to queue map

<-- Information on how different traffic types map to different queues are found here.
```

```
================================================================================

PlcIdx CPP Class                                 :  Queues
--------------------------------------------------------------------------------
0      system-cpp-police-data                    :  ICMP GEN/ BROADCAST/ ICMP Redirect/
10     system-cpp-police-sys-data                :  Openflow/ Exception/ EGR Exception/ NFL SAMPLED DATA/
13     system-cpp-police-sw-forward              :  Sw forwarding/ LOGGING/ L2 LVX Data Pack/ Transit Tra
9      system-cpp-police-multicast               :  MCAST Data/
15     system-cpp-police-multicast-end-station   :  MCAST END STATION /
7      system-cpp-police-punt-webauth            :  Punt Webauth/
1      system-cpp-police-l2-control              :  L2 Control/
2      system-cpp-police-routing-control         :  Routing Control/ Low Latency/
3      system-cpp-police-system-critical         :  System Critical/ Gold Pkt/
4      system-cpp-police-l2lvx-control           :  L2 LVX Cont Pack/
8      system-cpp-police-topology-control        :  Topology Control/
11     system-cpp-police-dot1x-auth              :  DOT1X Auth/
12     system-cpp-police-protocol-snooping       :  Proto Snooping/
6      system-cpp-police-dhcp-snooping           :  DHCP Snooping/
14     system-cpp-police-forus                   :  Forus Address resolution/ Forus traffic/
5      system-cpp-police-stackwise-virt-control  :  Stackwise Virtual OOB/
16     system-cpp-default                        :  Inter FED Traffic/ EWLC Data/
18     system-cpp-police-high-rate-app           :  High Rate App/
19     system-cpp-police-ewlc-control            :  EWLC Control/
20     system-cpp-police-ios-routing             :  L2 Control/ Topology Control/ Routing Control/ Low La
21     system-cpp-police-ios-feature             :  ICMP GEN/ BROADCAST/ ICMP Redirect/ L2 LVX Cont Pack/
```

パントされたトラフィックに関する情報の収集

これらのコマンドは、トラフィックのタイプや入力の物理的なポイントなど、CPUにパントされたトラフィックに関する情報を収集するために使用されます。

- 「Show platform software fed <switch> active punt cpuq all」または「Show platform software fed <switch> active punt cpuq <0-31 Queue ID>」を使用して、すべてまたは特定のCPUキューに関連する統計情報を表示できます。

<#root>

```
C9300#

show platform software fed switch active punt cpuq all

Punt CPU Q Statistics
============================================

CPU Q Id                     : 0
CPU Q Name                   : CPU_Q_DOT1X_AUTH
Packets received from ASIC   : 964
Send to IOSd total attempts  : 964
Send to IOSd failed count    : 0
RX suspend count             : 0
RX unsuspend count           : 0
RX unsuspend send count      : 0
RX unsuspend send failed count : 0
RX consumed count            : 0
RX dropped count             : 0
RX non-active dropped count  : 0
```

```
RX conversion failure dropped  : 0
RX INTACK count                : 964
RX packets dq'd after intack   : 0
Active RxQ event               : 964
RX spurious interrupt          : 0
RX phy_idb fetch failed: 0
RX table_id fetch failed: 0
RX invalid punt cause: 0

CPU Q Id                       : 1
CPU Q Name                     : CPU_Q_L2_CONTROL
Packets received from ASIC     : 80487
Send to IOSd total attempts    : 80487
Send to IOSd failed count      : 0
RX suspend count               : 0
RX unsuspend count             : 0
RX unsuspend send count        : 0
RX unsuspend send failed count : 0
RX consumed count              : 0
RX dropped count               : 0
RX non-active dropped count    : 0
RX conversion failure dropped  : 0
RX INTACK count                : 80474
RX packets dq'd after intack   : 16
Active RxQ event               : 80474
RX spurious interrupt          : 9
RX phy_idb fetch failed: 0
RX table_id fetch failed: 0
RX invalid punt cause: 0

CPU Q Id                       : 2
CPU Q Name                     : CPU_Q_FORUS_TRAFFIC
Packets received from ASIC     : 176669
Send to IOSd total attempts    : 176669
Send to IOSd failed count      : 0
RX suspend count               : 0
RX unsuspend count             : 0
RX unsuspend send count        : 0
RX unsuspend send failed count : 0
RX consumed count              : 0
RX dropped count               : 0
RX non-active dropped count    : 0
RX conversion failure dropped  : 0
RX INTACK count                : 165584
RX packets dq'd after intack   : 12601
Active RxQ event               : 165596
RX spurious interrupt          : 11851
RX phy_idb fetch failed: 0
RX table_id fetch failed: 0
RX invalid punt cause: 0
<snip>

C9300#
```

show platform software fed switch active punt cpuq 16 <-- Queue ID 16 correlates with Protocol Snooping.

```
Punt CPU Q Statistics
=============================================

CPU Q Id                       : 16
CPU Q Name                     : CPU_Q_PROTO_SNOOPING
Packets received from ASIC     : 55661
Send to IOSd total attempts    : 55661
```

```
Send to IOSd failed count      : 0
RX suspend count               : 0
RX unsuspend count             : 0
RX unsuspend send count        : 0
RX unsuspend send failed count : 0
RX consumed count              : 0
RX dropped count               : 0
RX non-active dropped count    : 0
RX conversion failure dropped  : 0
RX INTACK count                : 55659
RX packets dq'd after intack   : 9
Active RxQ event               : 55659
RX spurious interrupt          : 23
RX phy_idb fetch failed: 0
RX table_id fetch failed: 0
RX invalid punt cause: 0

Replenish Stats for all rxq:
----------------------------------------
Number of replenish            : 4926842
Number of replenish suspend    : 0
Number of replenish un-suspend : 0
----------------------------------------
```

- 「show platform software fed <switch> active punt cause summary」を使用し、CPUで発生している異なるトラフィックタイプをすべて確認します。ゼロ以外の原因だけが表示されることに注意してください。

<#root>

```
C9300#

show platform software fed switch active punt cause summary

Statistics for all causes

Cause  Cause Info                     Rcvd                 Dropped
--------------------------------------------------------------------------------
7      ARP request or response        142962               0
11     For-us data                    490817               0
21     RP<->QFP keepalive             448742               0
24     Glean adjacency                2                    0
55     For-us control                 415222               0
58     Layer2 bridge domain data packe 3654659             0
60     IP subnet or broadcast packet  37167                0
75     EPC                            17942                0
96     Layer2 control protocols       358614               0
97     Packets to LFTS                964                  0
109    snoop packets                  48867                0
--------------------------------------------------------------------------------
```

- コマンド「show platform software fed <switch> active punt rates interfaces」を使用して、システムに入力されるCPUバウンドトラフィックのインターフェイスを迅速に表示します。このコマンドでは、0以外の入力キューを持つインターフェイスだけが表示されます。

<#root>

```
C9300#

show platform software fed switch active punt rates interfaces

Punt Rate on Interfaces Statistics

Packets per second averaged over 10 seconds, 1 min and 5 mins
================================================================================================
                              |           |  Recv |  Recv |  Recv |  Drop |  Drop  |  Drop
 Interface Name               |  IF_ID    |  10s  |  1min |  5min |  10s  |  1min  |  5min
================================================================================================
 TenGigabitEthernet1/0/2       0x0000000a       5       5       5       0       0       0
 TenGigabitEthernet1/0/23      0x0000001f       1       1       1       0       0       0

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

- 「show platform software fed <switch> active punt rates interfaces <IF-ID>」を使用してドリルダウンし、インターフェイスの個々のキューを表示します。このコマンドは、集約の統計情報を表示します。また、入力キューの履歴動作や、トラフィックがポリシングされているかどうかを表示するために使用できます。

<#root>

```
C9300#

show platform software fed switch active punt rates interfaces 0x1f <-- "0x1f" is the IF_ID of Te1/0/23,

Punt Rate on Single Interfaces Statistics

Interface : TenGigabitEthernet1/0/23 [if_id: 0x1F]

  Received                          Dropped
  --------                          -------
   Total          : 1010652          Total           : 0
   10 sec average : 1                10 sec average  : 0
    1 min average : 1                 1 min average  : 0
    5 min average : 1                 5 min average  : 0

Per CPUQ punt stats on the interface (rate averaged over 10s interval)
=============================================================================
 Q  |         Queue                 |  Recv   |  Recv  |  Drop  |  Drop  |
 no |         Name                  |  Total  |  Rate  |  Total |  Rate  |
=============================================================================
 0   CPU_Q_DOT1X_AUTH                      0        0        0        0
 1   CPU_Q_L2_CONTROL                   9109        0        0        0
 2   CPU_Q_FORUS_TRAFFIC              176659        0        0        0
 3   CPU_Q_ICMP_GEN                        0        0        0        0
 4   CPU_Q_ROUTING_CONTROL            447374        0        0        0
 5   CPU_Q_FORUS_ADDR_RESOLUTION       80693        0        0        0
 6   CPU_Q_ICMP_REDIRECT                   0        0        0        0
 7   CPU_Q_INTER_FED_TRAFFIC               0        0        0        0
 8   CPU_Q_L2LVX_CONTROL_PKT               0        0        0        0
 9   CPU_Q_EWLC_CONTROL                    0        0        0        0
 10  CPU_Q_EWLC_DATA                       0        0        0        0
 11  CPU_Q_L2LVX_DATA_PKT                  0        0        0        0
 12  CPU_Q_BROADCAST                   22680        0        0        0
```

```
 13  CPU_Q_CONTROLLER_PUNT                  0        0        0        0
 14  CPU_Q_SW_FORWARDING                    0        0        0        0
 15  CPU_Q_TOPOLOGY_CONTROL            271014        0        0        0
 16  CPU_Q_PROTO_SNOOPING                   0        0        0        0
 17  CPU_Q_DHCP_SNOOPING                    0        0        0        0
 18  CPU_Q_TRANSIT_TRAFFIC                  0        0        0        0
 19  CPU_Q_RPF_FAILED                       0        0        0        0
 20  CPU_Q_MCAST_END_STATION_SERVICE     2679        0        0        0
 21  CPU_Q_LOGGING                        444        0        0        0
 22  CPU_Q_PUNT_WEBAUTH                     0        0        0        0
 23  CPU_Q_HIGH_RATE_APP                    0        0        0        0
 24  CPU_Q_EXCEPTION                        0        0        0        0
 25  CPU_Q_SYSTEM_CRITICAL                  0        0        0        0
 26  CPU_Q_NFL_SAMPLED_DATA                 0        0        0        0
 27  CPU_Q_LOW_LATENCY                      0        0        0        0
 28  CPU_Q_EGR_EXCEPTION                    0        0        0        0
 29  CPU_Q_FSS                              0        0        0        0
 30  CPU_Q_MCAST_DATA                       0        0        0        0
 31  CPU_Q_GOLD_PKT                         0        0        0        0

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

CPUに送られるトラフィックの検査

Catalyst 9000スイッチファミリは、CPUに送られるトラフィックを監視および表示するユーティリティを提供します。これらのツールを使用して、どのトラフィックがCPUにアクティブにパントされるかを理解します。

組み込みパケット キャプチャ（EPC）

コントロールプレーン上のEPCは、いずれかの方向（または両方）で実行できます。パントされたトラフィックの場合は、着信をキャプチャします。コントロールプレーンのEPCは、バッファまたはファイルに保存できます。

<#root>

```
C9300#

monitor capture CONTROL control-plane in match any buffer circular size 10


C9300#

show monitor capture CONTROL parameter <-- Check to ensure parameters are as expected.

   monitor capture CONTROL control-plane IN
   monitor capture CONTROL match any
   monitor capture CONTROL buffer size 10 circular
C9300#

monitor capture CONTROL start <-- Starts the capture.

Started capture point : CONTROL
C9300#

monitor capture CONTROL stop <-- Stops the capture.
```

```
Capture statistics collected at software:
    Capture duration - 5 seconds
    Packets received - 39
    Packets dropped - 0
    Packets oversized - 0

Bytes dropped in asic - 0

Capture buffer will exists till exported or cleared

Stopped capture point : CONTROL
```

キャプチャ結果は、簡単な出力または詳細な出力で表示できます。

<#root>

```
C9300#

show monitor capture CONTROL buffer brief

Starting the packet display ........ Press Ctrl + Shift + 6 to exit

    1   0.000000 5c:5a:c7:61:4c:5f -> 00:00:04:00:0e:00 ARP 64 192.168.10.1 is at 5c:5a:c7:61:4c:5f
    2   0.030643 00:00:00:00:00:00 -> 00:06:df:f7:20:01 0x0000 30 Ethernet II
    3   0.200016 5c:5a:c7:61:4c:5f -> 00:00:04:00:0e:00 ARP 64 192.168.10.1 is at 5c:5a:c7:61:4c:5f
    4   0.400081 5c:5a:c7:61:4c:5f -> 00:00:04:00:0e:00 ARP 64 192.168.10.1 is at 5c:5a:c7:61:4c:5f
    5   0.599962 5c:5a:c7:61:4c:5f -> 00:00:04:00:0e:00 ARP 64 192.168.10.1 is at 5c:5a:c7:61:4c:5f
    6   0.800067 5c:5a:c7:61:4c:5f -> 00:00:04:00:0e:00 ARP 64 192.168.10.1 is at 5c:5a:c7:61:4c:5f
    7   0.812456 00:1b:0d:a5:e2:a5 -> 01:80:c2:00:00:00 STP 60 RST. Root = 0/10/00:1b:53:bb:91:00  Cost
    8   0.829809 10.122.163.3 -> 224.0.0.2    HSRP 92 Hello (state Active)
    9   0.981313 10.122.163.2 -> 224.0.0.13   PIMv2 72 Hello
   10   1.004747 5c:5a:c7:61:4c:5f -> 00:00:04:00:0e:00 ARP 64 192.168.10.1 is at 5c:5a:c7:61:4c:5f
   11   1.200082 5c:5a:c7:61:4c:5f -> 00:00:04:00:0e:00 ARP 64 192.168.10.1 is at 5c:5a:c7:61:4c:5f
   12   1.399987 5c:5a:c7:61:4c:5f -> 00:00:04:00:0e:00 ARP 64 192.168.10.1 is at 5c:5a:c7:61:4c:5f
   13   1.599944 5c:5a:c7:61:4c:5f -> 00:00:04:00:0e:00 ARP 64 192.168.10.1 is at 5c:5a:c7:61:4c:5f
<snip>

C9300#

show monitor capture CONTROL buffer detail | begin Frame 7

Frame 7: 60 bytes on wire (480 bits), 60 bytes captured (480 bits) on interface /tmp/epc_ws/wif_to_ts_p
    Interface id: 0 (/tmp/epc_ws/wif_to_ts_pipe)
        Interface name: /tmp/epc_ws/wif_to_ts_pipe
    Encapsulation type: Ethernet (1)
    Arrival Time: May  3, 2023 23:58:11.727432000 UTC
    [Time shift for this packet: 0.000000000 seconds]
    Epoch Time: 1683158291.727432000 seconds
    [Time delta from previous captured frame: 0.012389000 seconds]
    [Time delta from previous displayed frame: 0.012389000 seconds]
    [Time since reference or first frame: 0.812456000 seconds]
    Frame Number: 7
    Frame Length: 60 bytes (480 bits)
    Capture Length: 60 bytes (480 bits)
    [Frame is marked: False]
    [Frame is ignored: False]
    [Protocols in frame: eth:llc:stp]
IEEE 802.3 Ethernet
    Destination: 01:80:c2:00:00:00 (01:80:c2:00:00:00)
```

```
        Address: 01:80:c2:00:00:00 (01:80:c2:00:00:00)
        .... ..0. .... .... .... .... = LG bit: Globally unique address (factory default)
        .... ...1 .... .... .... .... = IG bit: Group address (multicast/broadcast)
    Source: 00:1b:0d:a5:e2:a5 (00:1b:0d:a5:e2:a5)
        Address: 00:1b:0d:a5:e2:a5 (00:1b:0d:a5:e2:a5)
        .... ..0. .... .... .... .... = LG bit: Globally unique address (factory default)
        .... ...0 .... .... .... .... = IG bit: Individual address (unicast)
    Length: 39
    Padding: 00000000000000
Logical-Link Control
    DSAP: Spanning Tree BPDU (0x42)
        0100 001. = SAP: Spanning Tree BPDU
        .... ...0 = IG Bit: Individual
    SSAP: Spanning Tree BPDU (0x42)
        0100 001. = SAP: Spanning Tree BPDU
        .... ...0 = CR Bit: Command
    Control field: U, func=UI (0x03)
        000. 00.. = Command: Unnumbered Information (0x00)
        .... ..11 = Frame type: Unnumbered frame (0x3)
Spanning Tree Protocol
    Protocol Identifier: Spanning Tree Protocol (0x0000)
    Protocol Version Identifier: Rapid Spanning Tree (2)
    BPDU Type: Rapid/Multiple Spanning Tree (0x02)
    BPDU flags: 0x3c, Forwarding, Learning, Port Role: Designated
        0... .... = Topology Change Acknowledgment: No
        .0.. .... = Agreement: No
        ..1. .... = Forwarding: Yes
        ...1 .... = Learning: Yes
        .... 11.. = Port Role: Designated (3)
        .... ..0. = Proposal: No
        .... ...0 = Topology Change: No
    Root Identifier: 0 / 10 / 00:1b:53:bb:91:00
        Root Bridge Priority: 0
        Root Bridge System ID Extension: 10
        Root Bridge System ID: 00:1b:53:bb:91:00 (00:1b:53:bb:91:00)
    Root Path Cost: 19
    Bridge Identifier: 32768 / 10 / 00:1b:0d:a5:e2:80
        Bridge Priority: 32768
        Bridge System ID Extension: 10
        Bridge System ID: 00:1b:0d:a5:e2:80 (00:1b:0d:a5:e2:80)
    Port identifier: 0x8025
    Message Age: 1
    Max Age: 20
    Hello Time: 2
    Forward Delay: 15
    Version 1 Length: 0

C9300#

monitor capture CONTROL buffer display-filter "frame.number==9" detailed <-- Most Wireshark display filt

Starting the packet display ........ Press Ctrl + Shift + 6 to exit

Frame 9: 64 bytes on wire (512 bits), 64 bytes captured (512 bits) on interface /tmp/epc_ws/wif_to_ts_p
    Interface id: 0 (/tmp/epc_ws/wif_to_ts_pipe)
        Interface name: /tmp/epc_ws/wif_to_ts_pipe
    Encapsulation type: Ethernet (1)
    Arrival Time: May  4, 2023 00:07:44.912567000 UTC
    [Time shift for this packet: 0.000000000 seconds]
    Epoch Time: 1683158864.912567000 seconds
    [Time delta from previous captured frame: 0.123942000 seconds]
    [Time delta from previous displayed frame: 0.000000000 seconds]
    [Time since reference or first frame: 1.399996000 seconds]
```

```
    Frame Number: 9
    Frame Length: 64 bytes (512 bits)
    Capture Length: 64 bytes (512 bits)
    [Frame is marked: False]
    [Frame is ignored: False]
    [Protocols in frame: eth:ethertype:vlan:ethertype:arp]
Ethernet II, Src: 5c:5a:c7:61:4c:5f (5c:5a:c7:61:4c:5f), Dst: 00:00:04:00:0e:00 (00:00:04:00:0e:00)
    Destination: 00:00:04:00:0e:00 (00:00:04:00:0e:00)
        Address: 00:00:04:00:0e:00 (00:00:04:00:0e:00)
        .... ..0. .... .... .... .... = LG bit: Globally unique address (factory default)
        .... ...0 .... .... .... .... = IG bit: Individual address (unicast)
    Source: 5c:5a:c7:61:4c:5f (5c:5a:c7:61:4c:5f)
        Address: 5c:5a:c7:61:4c:5f (5c:5a:c7:61:4c:5f)
        .... ..0. .... .... .... .... = LG bit: Globally unique address (factory default)
        .... ...0 .... .... .... .... = IG bit: Individual address (unicast)
    Type: 802.1Q Virtual LAN (0x8100)
802.1Q Virtual LAN, PRI: 0, DEI: 0, ID: 10
    000. .... .... .... = Priority: Best Effort (default) (0)
    ...0 .... .... .... = DEI: Ineligible
    .... 0000 0000 1010 = ID: 10
    Type: ARP (0x0806)
    Padding: 00000000000000000000000000000000
    Trailer: 00000000
Address Resolution Protocol (reply)
    Hardware type: Ethernet (1)
    Protocol type: IPv4 (0x0800)
    Hardware size: 6
    Protocol size: 4
    Opcode: reply (2)
    Sender MAC address: 5c:5a:c7:61:4c:5f (5c:5a:c7:61:4c:5f)
    Sender IP address: 192.168.10.1
    Target MAC address: 00:00:04:00:0e:00 (00:00:04:00:0e:00)
    Target IP address: 192.168.10.25
```

キャプチャ結果は、ファイルに直接書き込むか、バッファからエクスポートできます。

<#root>

```
C9300#

monitor capture CONTROL export location flash:control.pcap <-- Exports the current buffer to file. Exter

Export Started Successfully

Export completed for capture point CONTROL
C9300#
C9300#

dir flash: | in control.pcap

475231  -rw-             3972   May 4 2023 00:00:38 +00:00  control.pcap
C9300#
```

FED CPUパケットキャプチャ

Catalyst 9000ファミリのスイッチは、CPUとの間で送受信されるパケットの可視性を向上させるデバッグユーティリティをサポートしています。

```
C9300#debug platform software fed switch active punt packet-capture ?
  buffer        Configure packet capture buffer
  clear-filter  Clear punt PCAP filter
  set-filter    Specify wireshark like filter (Punt PCAP)
  start         Start punt packet capturing
  stop          Stop punt packet capturing

C9300#$re fed switch active punt packet-capture buffer limit 16384
Punt PCAP buffer configure: one-time with buffer size 16384...done

C9300#show platform software fed switch active punt packet-capture status
Punt packet capturing: disabled. Buffer wrapping: disabled
Total captured so far: 0 packets. Capture capacity : 16384 packets

C9300#debug platform software fed switch active punt packet-capture start
Punt packet capturing started.

C9300#debug platform software fed switch active punt packet-capture stop
Punt packet capturing stopped. Captured 55 packet(s)
```

バッファの内容には、出力の簡単なオプションと詳細なオプションがあります。

<#root>

C9300#

show platform software fed switch active punt packet-capture brief

```
Punt packet capturing: disabled. Buffer wrapping: disabled
Total captured so far: 55 packets. Capture capacity : 16384 packets

------ Punt Packet Number: 1, Timestamp: 2023/05/04 00:17:41.709 ------
 interface : physical: TenGigabitEthernet1/0/2[if-id: 0x0000000a], pal: TenGigabitEthernet1/0/2 [if-id:
 metadata  : cause: 109 [snoop packets], sub-cause: 1, q-no: 16, linktype: MCP_LINK_TYPE_IP [1]
 ether hdr : dest mac: 0000.0400.0e00, src mac: 5c5a.c761.4c5f
 ether hdr : vlan: 10, ethertype: 0x8100

------ Punt Packet Number: 2, Timestamp: 2023/05/04 00:17:41.909 ------
 interface : physical: TenGigabitEthernet1/0/2[if-id: 0x0000000a], pal: TenGigabitEthernet1/0/2 [if-id:
 metadata  : cause: 109 [snoop packets], sub-cause: 1, q-no: 16, linktype: MCP_LINK_TYPE_IP [1]
 ether hdr : dest mac: 0000.0400.0e00, src mac: 5c5a.c761.4c5f
 ether hdr : vlan: 10, ethertype: 0x8100

------ Punt Packet Number: 3, Timestamp: 2023/05/04 00:17:42.109 ------
 interface : physical: TenGigabitEthernet1/0/2[if-id: 0x0000000a], pal: TenGigabitEthernet1/0/2 [if-id:
 metadata  : cause: 109 [snoop packets], sub-cause: 1, q-no: 16, linktype: MCP_LINK_TYPE_IP [1]
 ether hdr : dest mac: 0000.0400.0e00, src mac: 5c5a.c761.4c5f
 ether hdr : vlan: 10, ethertype: 0x8100

------ Punt Packet Number: 4, Timestamp: 2023/05/04 00:17:42.309 ------
 interface : physical: TenGigabitEthernet1/0/2[if-id: 0x0000000a], pal: TenGigabitEthernet1/0/2 [if-id:
 metadata  : cause: 109 [snoop packets], sub-cause: 1, q-no: 16, linktype: MCP_LINK_TYPE_IP [1]
 ether hdr : dest mac: 0000.0400.0e00, src mac: 5c5a.c761.4c5f
```

```
 ether hdr : vlan: 10, ethertype: 0x8100

------ Punt Packet Number: 5, Timestamp: 2023/05/04 00:17:42.509 ------
 interface : physical: TenGigabitEthernet1/0/2[if-id: 0x0000000a], pal: TenGigabitEthernet1/0/2 [if-id:
 metadata  : cause: 109 [snoop packets], sub-cause: 1, q-no: 16, linktype: MCP_LINK_TYPE_IP [1]
 ether hdr : dest mac: 0000.0400.0e00, src mac: 5c5a.c761.4c5f
 ether hdr : vlan: 10, ethertype: 0x8100

C9300#
```

show platform software fed switch active punt packet-capture detailed <-- Detailed provides the same inf

```
Punt packet capturing: disabled. Buffer wrapping: disabled
Total captured so far: 55 packets. Capture capacity : 16384 packets

------ Punt Packet Number: 1, Timestamp: 2023/05/04 00:17:41.709 ------
 interface : physical: TenGigabitEthernet1/0/2[if-id: 0x0000000a], pal: TenGigabitEthernet1/0/2 [if-id:
 metadata  : cause: 109 [snoop packets], sub-cause: 1, q-no: 16, linktype: MCP_LINK_TYPE_IP [1]
 ether hdr : dest mac: 0000.0400.0e00, src mac: 5c5a.c761.4c5f
 ether hdr : vlan: 10, ethertype: 0x8100

 Packet Data Hex-Dump (length: 68 bytes) :
   000004000E005C5A  C7614C5F8100000A  0806000108000604  00025C5AC7614C5F
   C0A80A0100000400  0E00C0A80A190000  0000000000000000  0000000000000000
   E9F1C9F3

 Doppler Frame Descriptor :
   fdFormat                  = 0x4           systemTtl                 = 0xe
   loadBalHash1              = 0x20          loadBalHash2              = 0xc
   spanSessionMap            = 0             forwardingMode            = 0
   destModIndex              = 0             skipIdIndex               = 0
   srcGpn                    = 0x2           qosLabel                  = 0x83
   srcCos                    = 0             ingressTranslatedVlan     = 0x7
   bpdu                      = 0             spanHistory               = 0
   sgt                       = 0             fpeFirstHeaderType        = 0
   srcVlan                   = 0xa           rcpServiceId              = 0x1
   wccpSkip                  = 0             srcPortLeIndex            = 0x1
   cryptoProtocol            = 0             debugTagId                = 0
   vrfId                     = 0             saIndex                   = 0
   pendingAfdLabel           = 0             destClient                = 0x1
   appId                     = 0             finalStationIndex         = 0x74
   decryptSuccess            = 0             encryptSuccess            = 0
   rcpMiscResults            = 0             stackedFdPresent          = 0
   spanDirection             = 0             egressRedirect            = 0
   redirectIndex             = 0             exceptionLabel            = 0
   destGpn                   = 0             inlineFd                  = 0x1
   suppressRefPtrUpdate      = 0             suppressRewriteSideEfects = 0
   cmi2                      = 0             currentRi                 = 0x1
   currentDi                 = 0x527b        dropIpUnreachable         = 0
   srcZoneId                 = 0             srcAsicId                 = 0
   originalDi                = 0             originalRi                = 0
   srcL3IfIndex              = 0x27          dstL3IfIndex              = 0
   dstVlan                   = 0             frameLength               = 0x44
   fdCrc                     = 0x97          tunnelSpokeId             = 0
   isPtp                     = 0             ieee1588TimeStampValid    = 0
   ieee1588TimeStamp55_48    = 0             lvxSourceRlocIpAddress    = 0
   sgtCachingNeeded          = 0

 Doppler Frame Descriptor Hex-Dump :
   0000000044004E04  000B40977B520000  0000000000000100  000000070A000000
   0000000001000010  0000000074000100  0000000027830200  0000000000000000
```

多くの表示フィルタを使用できます。最も一般的なWireshark表示フィルタがサポートされています。

<#root>

C9300#

show platform software fed switch active punt packet-capture display-filter-help

```
FED Punject specific filters :
   1. fed.cause              FED punt or inject cause
   2. fed.linktype           FED linktype
   3. fed.pal_if_id          FED platform interface ID
   4. fed.phy_if_id          FED physical interface ID
   5. fed.queue              FED Doppler hardware queue
   6. fed.subcause           FED punt or inject sub cause
Generic filters supported :
   7. arp                    Is this an ARP packet
   8. bootp                  DHCP packets [Macro]
   9. cdp                    Is this a CDP packet
  10. eth                    Does the packet have an Ethernet header
  11. eth.addr               Ethernet source or destination MAC address
  12. eth.dst                Ethernet destination MAC address
  13. eth.ig                 IG bit of ethernet destination address (broadcast/multicast)
  14. eth.src                Ethernet source MAC address
  15. eth.type               Ethernet type
  16. gre                    Is this a GRE packet
  17. icmp                   Is this a ICMP packet
  18. icmp.code              ICMP code
  19. icmp.type              ICMP type
  20. icmpv6                 Is this a ICMPv6 packet
  21. icmpv6.code            ICMPv6 code
  22. icmpv6.type            ICMPv6 type
  23. ip                     Does the packet have an IPv4 header
  24. ip.addr                IPv4 source or destination IP address
  25. ip.dst                 IPv4 destination IP address
  26. ip.flags.df            IPv4 dont fragment flag
  27. ip.flags.mf            IPv4 more fragments flag
  28. ip.frag_offset         IPv4 fragment offset
  29. ip.proto               Protocol used in datagram
  30. ip.src                 IPv4 source IP address
  31. ip.ttl                 IPv4 time to live
  32. ipv6                   Does the packet have an IPv4 header
  33. ipv6.addr              IPv6 source or destination IP address
  34. ipv6.dst               IPv6 destination IP address
  35. ipv6.hlim              IPv6 hop limit
  36. ipv6.nxt               IPv6 next header
  37. ipv6.plen              IPv6 payload length
  38. ipv6.src               IPv6 source IP address
  39. stp                    Is this a STP packet
  40. tcp                    Does the packet have a TCP header
  41. tcp.dstport            TCP destination port
  42. tcp.port               TCP source OR destination port
  43. tcp.srcport            TCP source port
  44. udp                    Does the packet have a UDP header
  45. udp.dstport            UDP destination port
  46. udp.port               UDP source OR destination port
  47. udp.srcport            UDP source port
  48. vlan.id                Vlan ID (dot1q or qinq only)
  49. vxlan                  Is this a VXLAN packet
```

```
C9300#

show platform software fed switch active punt packet-capture display-filter arp brief

Punt packet capturing: disabled. Buffer wrapping: disabled
Total captured so far: 55 packets. Capture capacity : 16384 packets

------ Punt Packet Number: 1, Timestamp: 2023/05/04 00:17:41.709 ------
 interface : physical: TenGigabitEthernet1/0/2[if-id: 0x0000000a], pal: TenGigabitEthernet1/0/2 [if-id:
 metadata  : cause: 109 [snoop packets], sub-cause: 1, q-no: 16, linktype: MCP_LINK_TYPE_IP [1]
 ether hdr : dest mac: 0000.0400.0e00, src mac: 5c5a.c761.4c5f
 ether hdr : vlan: 10, ethertype: 0x8100

------ Punt Packet Number: 2, Timestamp: 2023/05/04 00:17:41.909 ------
 interface : physical: TenGigabitEthernet1/0/2[if-id: 0x0000000a], pal: TenGigabitEthernet1/0/2 [if-id:
 metadata  : cause: 109 [snoop packets], sub-cause: 1, q-no: 16, linktype: MCP_LINK_TYPE_IP [1]
 ether hdr : dest mac: 0000.0400.0e00, src mac: 5c5a.c761.4c5f
 ether hdr : vlan: 10, ethertype: 0x8100
<snip>
```

フィルタは、キャプチャフィルタとしても適用できます。

```
<#root>

C9300#

show platform software fed switch active punt packet-capture set-filter arp <-- Most common Wireshark fi

Filter setup successful. Captured packets will be cleared

C9300#$e fed switch active punt packet-capture status
Punt packet capturing: disabled. Buffer wrapping: disabled
Total captured so far: 0 packets. Capture capacity : 16384 packets
Capture filter : "arp"
```

一般的なシナリオ

ローカルIPへの断続的なICMP(ping)損失

スイッチ上のローカルIPに転送されるトラフィックは、Forus (文字通り「私たちのために」) キューにパントされます。Forus CoPPキューの増加は、ローカルスイッチ宛てのパケットのドロップに関連しています。これは比較的単純で、概念化が容易です。

ただし、一部の状況では、Forusドロップときれいに関連付けられていないローカルで宛先が指定されたトラフィックが失われる可能性があります。

十分なCPUバウンドトラフィックフローがあると、パントパスは、ポリシングするトラフィックに優先順位を付けるCoPPの機能を超えて飽和状態になります。トラフィックは、ファーストインファーストアウト(FIFO)ベースで「サイレントポリシング」されます。

このシナリオでは、大量のコントロールプレーンポリシング(CoPP)の証拠が見られますが、対象のトラフィックタイプ（この例ではForus）はアクティブに増加するとは限りません。

要約すると、アクティブなCoPPポリシングの両方によって明らかになり、パケットキャプチャまたはFEDパントデバッグで示された、CPUに送られるトラフィックが非常に大量の場合は、トラブルシューティングするキューに合わない損失が発生する可能性があります。このシナリオでは、過剰な量のCPUに送られるトラフィックが存在する理由を判別し、コントロールプレーンの負荷を軽減する対策を講じます。

高いICMPリダイレクトと遅いDHCP動作

Catalyst 9000シリーズスイッチのCoPPは、32のハードウェアキューに分かれています。これらの32個のハードウェアキューは、20個の個別ポリサーインデックスに対応します。 各ポリサーインデックスは、1つ以上のハードウェアキューに関連付けられます。

機能的には、これは複数のトラフィッククラスがポリサーインデックスを共有し、共通の集約ポリサー値の対象となることを意味します。

DHCPリレーエージェントが有効になっているスイッチで見られる一般的な問題には、DHCP応答の遅延が含まれます。クライアントは散発的にIPを取得できますが、完了するまでに数回試行が必要で、一部のクライアントはタイムアウトします。

ICMPリダイレクトキューとブロードキャストキューはポリサーインデックスを共有するため、同じスイッチ仮想インターフェイス(SVI)で受信され、同じスイッチからルーティングされる大量のトラフィックは、ブロードキャストトラフィックに依存するアプリケーションに影響を与えます。 これは、スイッチがリレーエージェントとして動作している場合に特に顕著です。

このドキュメントでは、この概念の詳細と、「Catalyst 9000 DHCPリレーエージェントのDHCP問題のトラブルシューティング」の緩和方法について説明します。

関連情報

Catalyst 9000 DHCPリレーエージェントの低速または断続的なDHCPのトラブルシューティング

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

Catalyst 9300スイッチ：コントロールプレーンポリシングの設定

パケットキャプチャの設定：ネットワーク管理コンフィギュレーションガイド、Cisco IOS XE Bengaluru 17.6.x（Catalyst 9300スイッチ）

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

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。