排除Catalyst 9000交换机上的控制平面操作故障

目录

简介

本文档介绍如何对运行Cisco IOS® XE的Catalyst 9000系列交换机上的控制平面运行状况进行故障排除和验证。

背景信息

交换机的主要任务是尽快转发数据包。大多数数据包在硬件中转发，但某些类型的流量必须由系统CPU处理。尽快处理到达CPU的流量。预计在CPU中会看到一定数量的流量，但是流量过大会导致操作问题。Catalyst 9000系列交换机默认采用强大的控制平面策略(CoPP)机制，以防止CPU流量过度饱和引起的问题。

在某些使用案例中，作为正常操作的功能，会出现意外问题。因果关系有时并不明显，使问题难以解决。本文档提供了用于验证控制平面运行状况的工具，并提供了有关如何处理涉及控制平面传送或插入路径问题的工作流。它还根据现场发现的问题提供了几种常见方案。

请记住，CPU传送路径是有限的资源。现代硬件转发交换机可以处理呈指数级增长的流量。在任何给定时间，Catalyst 9000系列交换机在CPU上总共支持约19,000个数据包/秒(pps)。超过此阈值，将监管传送的流量，而不考虑权重。

术语

- 转发引擎驱动程序(FED)：这是Cisco Catalyst交换机的核心，负责所有硬件编程/转发
- IOSd：这是在Linux内核上运行的Cisco IOS守护进程。它在内核中作为软件进程运行
- 数据包传输系统(PDS)：这是将数据包传输到各个子系统和从各个子系统传输数据包的体系结构和过程。例如，它控制数据包如何从FED发送到IOSd，反之亦然
- 控制平面(CP)：控制平面是一个通用术语，用于将涉及Catalyst交换机CPU的功能和流量组合在一起。这包括发往交换机或从交换机发送的流量，例如生成树协议(STP)、热备份路由器协议(HSRP)和路由协议。这也包括必须由CPU处理的应用层协议，如安全外壳(SSH)和简单网络管理协议(SNMP)
- 数据平面(DP)：通常，数据平面包含硬件ASIC和转发而无控制平面帮助的流量
- 传送：在DP上拦截的入口协议控制数据包，该数据包被发送到CP进行处理
- 注入：CP生成的协议数据包发送到DP以从IO接口输出
- LSMPI：Linux共享内存传送接口

Catalyst 9000 CoPP

CoPP是Catalyst 9000系列交换机上CPU保护的基础。使用CoPP时，系统生成的服务质量(QoS)策略应用于CPU传送/插入路径。与CPU绑定的流量分为许多不同的类别，随后映射到与CPU关联的各个硬件策略器。监察器可防止特定流量类别导致CPU过饱和。

CoPP实施

CPU绑定的流量被分类为队列。这些队列/类是系统定义的，不可由用户配置。策略器在硬件中配置。Catalyst 9000系列支持32个队列的32个硬件策略器。

具体值因平台而异。一般来说，有32个系统定义的队列。这些队列与类映射相关，类映射与监察器索引相关。监察器索引具有默认监察器速率。此速率可由用户配置，但更改默认CoPP策略会增加意外服务影响的敏感度。

CoPP的系统定义值

类映射名称	管制器索引（管制器编号）	CPU队列（队列编号）
system-cpp-police-data	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-control	WK_CPP_POLICE_L2_控制(1)	WK_CPU_Q_L2_CONTROL(1)

类映射名称	管制器索引（管制器编号）	CPU队列（队列编号）
system-cpp-police-routing-control	WK_CPP_POLICE_ROUTING_CONTROL(2)	WK_CPU_Q_ROUTING_CONTROL(4) WK_CPU_Q_LOW_LATENCY (27)
system-cpp-police-control-low-priority	WK_CPP_POLICE_CO NTROL_LOW_PRI(3)	WK_CPU_Q_GENERAL_PUNT(25)
system-cpp-police-punt-webauth	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(15)
system-cpp-police-multicast	WK_CPP_POLICE_MULTICAST(9)	WK_CPU_Q_TRANSIT_TRAFFIC(18) WK_CPU_Q_MCAST_DATA(30)
system-cpp-police-sys-data	WK_CPP_POLICE_SYS_DATA(10)	WK_CPU_Q_LEARNING_CACHE_OVFL(13 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	WK_CPP_POLICE_DOT1X(11)	WK_CPU_Q_DOT1X_AUTH(0)

类映射名称	管制器索引（管制器编号）	CPU队列（队列编号）
system-cpp-police-protocol-snooping	WK_CPP_POLICE_PR(12)	WK_CPU_Q_PROTO_SNOOPING(16)
system-cpp-police-sw-forward	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	WK_CPP_POLICE_FORUS(14)	WK_CPU_Q_FORUS_ADDR_RESOLUTIO WK_CPU_Q_FORUS_TRAFFIC(2)
system-cpp-police-multicast-end-station	WK_CPP_POLICE_MULTICAST_SNOOPING(15)	WK_CPU_Q_MCAST_END_STA TION_SERVICE(20)
system-cpp-default	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_CON (29)
system-cpp-police-l2lvx-control	WK_CPP_ L2_LVX_CONT_PACK(4)	WK_CPU_Q_L2_LVX_CONT_PACK(8)

每个队列与流量类型或特定功能集相关。此列表并不详尽：

CPU队列和相关功能

CPU队列（队列编号）	功能
WK_CPU_Q_DOT1X_AUTH(0)	基于IEEE 802.1x端口的身份验证
WK_CPU_Q_L2_CONTROL(1)	动态中继协议 (DTP) VLAN 中继 协议 (VTP) 端口聚合协议 (PAgP) 客户端信息信令协议(CISP) 消息会话中继协议 多VLAN注册协议(MVRP) 城域移动网络(MMN) 链路级发现协议(LLDP) 单向链路检测 (UDLD) 链路聚合控制协议 (LACP) Cisco 发现协议 (CDP) 生成树协议 (STP)
WK_CPU_Q_FORUS_TRAFFIC(2)	主机，例如Telnet、Pingv4和Pingv6，以及SNMP 保持连接/环回检测 Initiate-Internet Key Exchange (IKE)协议 (IPSec)
WK_CPU_Q_ICMP_GEN(3)	ICMP -目标不可达 ICMP-TTL已过期
WK_CPU_Q_ROUTING_CONTROL(4)	路由信息协议第1版(RIPv1) RIPv2 内部网关路由协议(IGRP)

CPU队列（队列编号）	功能
	边界网关协议 (BGP) PIM-UDP 虚拟路由器冗余协议 (VRRP) 热备份路由器协议第1版(HSRPv1) HSRPv2 网关负载均衡协议 (GLBP) 标签分发协议(LDP) 网络高速缓存通信协议(WCCP) 下一代路由信息协议(RIPng) 开放最短路径优先(OSPF) 开放最短路径优先版本3 (OSPFv3) 增强型内部网关路由协议 (EIGRP) 增强型内部网关路由协议第6版(EIGRPv6) DHCPv6 独立于协议的多播 (PIM) 协议无关组播版本6 (PIMv6) 下一代热备份路由器协议(HSRPng) IPv6控制 通用路由封装(GRE) Keepalive 网络地址转换(NAT)传送 IS-IS（中间系统到中间系统）
WK_CPU_Q_FORUS_ADDR_RESOLUTION(5)	地址解析协议 (ARP) IPv6邻居通告和邻居请求
WK_CPU_Q_ICMP_REDIRECT(6)	互联网控制消息协议(ICMP)重定向

CPU队列（队列编号）	功能
WK_CPU_Q_INTER_FED_TRAFFIC(7)	第2层网桥域注入用于内部通信。
WK_CPU_Q_L2_LVX_CONT_PACK(8)	交换ID (XID)数据包
WK_CPU_Q_EWLC_CONTROL(9)	嵌入式无线控制器(eWLC) [无线接入点的控制和调配(CAPWAP) (UDP 5246)]
WK_CPU_Q_EWLC_DATA(10)	eWLC数据包（CAPWAP数据，UDP 5247）
WK_CPU_Q_L2_LVX_DATA_PACK(11)	为映射请求传送未知单播数据包。
WK_CPU_Q_BROADCAST(12)	所有类型的广播
WK_CPU_Q_OPENFLOW(13)	学习缓存溢出（第2层+第3层）
WK_CPU_Q_CONTROLLER_PUNT(14)	数据-访问控制列表(ACL)已满 数据- IPv4选项 数据- IPv6逐跳 数据-资源不足/全部捕获 数据-反向路径转发(RPF)未完成 收集数据包
WK_CPU_Q_TOPOLOGY_CONTROL(15)	生成树协议 (STP) 弹性以太网协议(REP) 共享生成树协议(SSTP)
WK_CPU_Q_PROTO_SNOOPING(16)	动态ARP检测(DAI)的地址解析协议(ARP)监听
WK_CPU_Q_DHCP_SNOOPING(17)	DHCP 监听
WK_CPU_Q_TRANSIT_TRAFFIC(18)	这用于由NAT传送的数据包，需要在软件路径

CPU队列（队列编号）	功能
	中处理。
WK_CPU_Q_RPF_FAILED(19)	数据- mRPF （组播RPF）失败
WK_CPU_Q_MCAST_END_STATION_SERVICE(20)	互联网组管理协议(IGMP)/组播侦听程序发现(MLD)控制
WK_CPU_Q_LOGGING(21)	访问控制列表(ACL)日志记录
WK_CPU_Q_PUNT_WEBAUTH(22)	Web 身份验证
WK_CPU_Q_HIGH_RATE_APP(23)	广播
WK_CPU_Q_EXCEPTION(24)	IKE指示 IP学习违规 IP端口安全违规 IP静态地址违规 IPv6范围检查 远程复制协议(RCP)异常 单播RPF失败
WK_CPU_Q_SYSTEM_CRITICAL(25)	媒体信令/无线代理ARP
WK_CPU_Q_NFL_SAMPLED_DATA(26)	Netflow采样数据和媒体服务代理(MSP)
WK_CPU_Q_LOW_LATENCY(27)	双向转发检测(BFD)、精确时间协议(PTP)
WK_CPU_Q_EGR_EXCEPTION(28)	出口解析异常
WK_CPU_Q_STACKWISE_VIRTUAL_CONTROL(29)	前端堆叠协议，即SVL

CPU队列（队列编号）	功能
WK_CPU_Q_MCAST_DATA(30)	数据- (S，G)创建 数据-本地联接 数据- PIM注册 数据- SPT切换 数据-组播
WK_CPU_Q_GOLD_PKT(31)	金牌

默认策略

默认情况下，系统生成的CoPP策略应用于传送/注入路径。通过使用常见的基于MQC的命令可以查看默认策略。它也可以在交换机配置中查看。允许应用于CPU/控制平面的入口或出口的唯一策略是系统定义的策略。

使用“show policy-map control-plane”查看应用于控制层面的策略：

<#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
```

调整CoPP

CoPP策略器速率可由用户配置。用户还可以禁用队列。

此示例说明如何调整单个监察器值。在本示例中，调整的类是“system-cpp-police-protocol-snooping”。

<#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
```

此示例说明如何完全禁用队列。禁用队列时要小心，因为这可能会导致CPU过饱和。

```
<#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
```

故障排除

方法

CPU利用率受两个基本活动的影响-进程和中断。进程是CPU执行的结构化活动，而中断是指数据包在数据平面被拦截并发送到CPU以便执行操作。这些活动共同构成了CPU的总利用率。由于默认情况下启用CoPP，服务影响并不一定与高CPU利用率相关。如果CoPP执行其工作，CPU利用率不会受到严重影响。考虑CPU的整体利用率很重要，但总体利用率并不能反映全部情况。本部分中的show命令和实用程序用于快速评估CPU的运行状况和识别有关计算密集型流量的相关详细信息。

指南：

- 确定问题是否与控制平面有关。大多数中转流量在硬件中转发。只有某些流量类型和某些场景涉及CPU和控制平面，因此在整个调查过程中请记住这一点。
- 了解您的利用率基线。了解正常使用状况很重要，这样才能确定与标准值的偏差。
- 验证流程和中断的整体利用率。确定占用意外量CPU周期的所有进程。如果利用率在预期范围之外，这可能会导致问题。了解系统的平均利用率很重要，这样可以识别标准之外的偏差。请记住，仅凭使用率并不能全面反映控制平面的运行状况。
- 确定CoPP中是否存在主动递增的丢包。CoPP丢包并不总是表示存在问题，但是，如果您对与主动管制的数据流类相关的问题进行故障排除，则这是相关性很强的指标。

有用的 show 命令

该交换机可快速监控CPU运行状况和CoPP统计信息。此外，还可以使用有用的CLI快速确定计算密集型数据流的入口点。

确定总体利用率和历史利用率

- “Show processes cpu sorted”用于查看整体CPU使用率。“sorted”参数根据使用率百分比对进程输出进行排序。使用更多CPU资源的进程位于输出的顶部。此外，还会以百分比形式提供由于中断而导致的使用率。

<#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”提供过去60秒、5分钟和72小时的CPU使用率历史图表。

```
<#root>

Catalyst-9600#

show processes cpu history


    999777776666688888666667777777778888877777666669999988888866

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


    222555559999944444444440000088888888811111777773333335555500

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%
```

检查控制平面策略

- 使用“show platform hardware fed <switch> active qos queue stats internal cpu policer”查看聚合CoPP统计信息和有关队列/监察器结构的其他信息。此输出提供自上次重置控制平面以来的监察器统计信息的历史视图。这些计数器也可以手动清除。 一般来说，由监察器执行的控制平面丢弃的证据指向相关队列/类的问题，但请确保在问题发生时丢弃会主动增加。多次运行该命令，以观察队列丢弃值的增加情况。

<#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 Index	Policer Accept Bytes	Policer Accept Frames	Policer Drop Bytes	Policer Drop 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 PlcIndex	:	level-1 PlcIndex	(default) rate	(set) rate
20	:	1 2 8	13000	17000
21	:	0 4 7 9 10 11 12 13 14 15	6000	6000

Second Level Policer Config

QId	level-1 PlcIdx	level-2 PlcIdx	Queue Name	level-2 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/
```

收集有关传送流量的信息

这些命令用于收集有关传送到CPU的数据流的信息，包括数据流类型和物理入口点。

- 可以使用Show platform software fed <switch> active punt cpuq all或Show platform software fed <switch> active punt cpuq <0-31 Queue ID>查看与所有或特定CPU队列有关的统计信息。

<#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
-----------------------------------------
```

- 使用show platform software fed <switch> active punt cause summary快速查看CPU中发现的所有不同流量类型。请注意，仅显示非零原因。

<#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
-------------------------------------------------------------------------------
```

- 使用命令show platform software fed <switch> active punt rates interfaces”快速查看传入系统的接口CPU数据流。此命令仅显示具有非零输入队列的接口。

<#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

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

- 使用show platform software fed <switch> active punt rates interfaces <IF-ID>"深入查看接口的单个队列。此命令显示聚合统计信息，并且可用于查看历史输入队列活动以及流量是否受到管制。

<#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

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

检查CPU绑定的流量

Catalyst 9000系列交换机提供监控和查看计算密集型流量的实用程序。使用这些工具了解哪些流量被主动传送到CPU。

嵌入式数据包捕获(EPC)

控制平面上的EPC可在任一方向（或两者）上完成。对于传送的流量，捕获入站流量。控制平面上的EPC可以保存到缓冲区或文件。

```
<#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
```

捕获结果可以在简要或详细输出中查看。

```
<#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
```

捕获结果可以直接写入文件，也可以从缓冲区导出。

<#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数据包捕获

Catalyst 9000系列交换机支持调试实用程序，该实用程序可增强与CPU之间的数据包可视性。

```
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)
```

缓冲区内容具有用于输出的简要和详细选项。

<#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
```

许多显示过滤器可供使用。支持最常见的Wireshark显示过滤器。

<#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>
```

过滤器也可以作为捕获过滤器应用。

<#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"
```

常见情况

到本地IP的间歇ICMP (Ping)丢失

转发到交换机上的本地IP的流量在Forus（字面意思是“for us”）队列中传送。Forus CoPP队列中的增加与发往本地交换机的已丢弃数据包相关。这相对直截了当，而且易于概念化。

但在某些情况下，本地流量可能会丢失，而本地流量与Forus丢弃没有明确关联。

如果数据流足够多的CPU限制，则传送路径会变得过饱和，超出了CoPP确定受管制流量的优先级的能力。流量以先进先出的方式进行“静默式”监管。

在此场景中，可以看到大量控制平面策略的证据，但是感兴趣的流量类型（本示例中为Foru）不一定主动增加。

总之，如果存在异常高的CPU流量（通过主动CoPP管制和数据包捕获或FED传送调试进行证明），则可能会出现与您正在排除的队列不一致的丢失。在这种情况下，请确定存在过多的CPU密集型流量的原因，并采取措施减轻控制平面的负担。

高ICMP重定向和DHCP运行缓慢

Catalyst 9000系列交换机上的CoPP划分为32个硬件队列。这32个硬件队列对应于20个单独的监察器索引。 每个监察器索引与一个或多个硬件队列相关。

从功能上讲，这意味着多个流量类共享一个监察器索引，并受同一个聚合监察器值的约束。

在启用了DHCP中继代理的交换机上发现的一个常见问题是DHCP响应缓慢。客户端可以偶尔获得IP，但需要多次尝试才能完成，并且某些客户端会超时。

ICMP重定向队列和广播队列共享监察器索引，因此在同一交换机虚拟接口(SVI)上接收和路由的大量流量会影响依赖广播流量的应用。 当交换机充当中继代理时，这一点尤其明显。

本文档深入解释了这一概念以及如何缓解：解决Catalyst 9000 DHCP中继代理上的DHCP问题

其它资源

排除Catalyst 9000 DHCP中继代理上的慢速或间歇性DHCP故障

在Catalyst 9000交换机上配置FED CPU数据包捕获

Catalyst 9300交换机：配置控制平面策略

配置数据包捕获-网络管理配置指南，Cisco IOS XE Bengaluru 17.6.x（Catalyst 9300交换机）

运行Catalyst 9000交换机上的DHCP监听并对其进行故障排除

关于此翻译
思科采用人工翻译与机器翻译相结合的方式将此文档翻译成不同语言，希望全球的用户都能通过各自的语言得到支持性的内容。

请注意：即使是最好的机器翻译，其准确度也不及专业翻译人员的水平。

Cisco Systems, Inc. 对于翻译的准确性不承担任何责任，并建议您总是参考英文原始文档（已提供链接）。