dépannage des opérations du plan de contrôle sur les commutateurs Catalyst 9000

Table des matières

- Introduction
- Informations générales
  - Terminologie
- Catalyst 9000 CoPP
  - Mise en oeuvre de la CoPP
  - Stratégie par défaut
  - Ajuster CoPP
- Dépannage
  - Méthode
  - Commandes show utiles
    - Détermination de l'utilisation globale et historique
    - Vérifier la réglementation du plan de contrôle
    - Collecte d'informations sur le trafic pointé
    - Inspecter le trafic lié au processeur
- Scénarios courants
  - Perte ICMP intermittente (Ping) vers IP locale
  - Redirections ICMP élevées et fonctionnement DHCP lent
- Ressources supplémentaires

Introduction

Ce document décrit comment dépanner et valider l'état du plan de contrôle sur les commutateurs de la gamme Catalyst 9000 qui exécutent Cisco IOS® XE.

Informations générales

La tâche principale d’un commutateur est de transférer les paquets aussi rapidement que possible. La plupart des paquets sont transférés dans le matériel, mais certains types de trafic doivent être gérés par le processeur du système. Le trafic qui arrive au processeur est traité aussi rapidement que possible. Une certaine quantité de trafic est attendue au niveau du processeur, mais une surabondance entraîne des problèmes de fonctionnement. La gamme de commutateurs Catalyst 9000 intègre par défaut un mécanisme robuste CoPP (Control Plane Policing) pour empêcher les problèmes causés par la saturation du trafic du processeur.

Des problèmes inattendus apparaissent dans certains cas d'utilisation en fonction du fonctionnement normal. La corrélation entre la cause et l'effet n'est pas évidente parfois, ce qui

rend le problème difficile à aborder. Ce document fournit des outils pour valider l'intégrité du plan de contrôle et fournit un workflow sur la façon d'aborder les problèmes qui impliquent le point du plan de contrôle ou le chemin d'injection. Il fournit également plusieurs scénarios courants basés sur les problèmes constatés sur le terrain.

Gardez à l'esprit que le chemin ponctuel du processeur est une ressource limitée. Les commutateurs de transfert matériel modernes peuvent gérer un volume de trafic exponentiellement plus important. La gamme de commutateurs Catalyst 9000 prend en charge environ 19 000 paquets par seconde (pps) au total au niveau du processeur à tout moment. Dépassez ce seuil et le trafic pointé est réglementé sans pondération.

Terminologie

- Pilote FED (Forwarding Engine Driver) : il s'agit du coeur du commutateur Cisco Catalyst et il est responsable de la programmation/transmission de tout le matériel
- IOSd : démon Cisco IOS qui s'exécute sur le noyau Linux. Il est exécuté en tant que processus logiciel dans le noyau
- Packet Delivery System (PDS) : architecture et processus de livraison des paquets vers et depuis les différents sous-systèmes. Par exemple, il contrôle la manière dont les paquets sont livrés de la FED à l'IOSd et vice versa
- Control Plane (CP) : le plan de contrôle est un terme générique utilisé pour regrouper les fonctions et le trafic qui impliquent le CPU du commutateur Catalyst. Cela inclut le trafic comme le protocole Spanning Tree (STP), le protocole HSRP (Hot Standby Router Protocol) et les protocoles de routage qui sont destinés au commutateur ou envoyés à partir du commutateur. Cela inclut également les protocoles de couche application tels que Secure Shell (SSH) et SNMP (Simple Network Management Protocol) qui doivent être gérés par le processeur
- Plan de données (DP) : généralement, le plan de données englobe les circuits ASIC matériels et le trafic qui est transféré sans assistance du plan de contrôle
- Punt : paquet de contrôle de protocole entrant intercepté sur le protocole DP envoyé au protocole CP pour le traiter
- Injection : paquet de protocole généré par le protocole CP envoyé au protocole DP pour sortir sur les interfaces E/S
- LSMPI:interface de point de mémoire partagée Linux

Catalyst 9000 CoPP

La protection du processeur sur la gamme de commutateurs Catalyst 9000 repose sur CoPP. Avec CoPP, une politique de qualité de service (QoS) générée par le système est appliquée sur le chemin d'injection/punt du processeur. Le trafic lié au processeur est regroupé en de nombreuses classes différentes, puis mappé sur les contrôleurs matériels individuels associés au processeur. Les régulateurs empêchent la sursaturation du processeur par une classe particulière de trafic.

Mise en oeuvre de la CoPP

Le trafic lié au processeur est classé en files d'attente. Ces files d'attente/classes sont définies par

le système et ne sont pas configurables par l'utilisateur. Les contrôleurs sont configurés dans le matériel. La gamme Catalyst 9000 prend en charge 32 contrôleurs matériels pour 32 files d'attente.

Les valeurs spécifiques varient d'une plate-forme à l'autre. En général, il y a 32 files d'attente définies par le système. Ces files d'attente se rapportent à des cartes-classes, qui se rapportent à des indices de régulateur. Les index du régulateur ont un taux de régulateur par défaut. Ce débit peut être configuré par l'utilisateur, bien que les modifications apportées à la stratégie CoPP par défaut augmentent la probabilité d'un impact imprévu sur le service.

Valeurs définies par le système pour CoPP

Noms des mappages de classe	Index du contrôleur (n° du contrôleur)	Files d'attente CPU (Queue No.)
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_ CONTROL(1)	WK_CPU_Q_L2_CONTROL(1)
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)

Noms des mappages de classe	Index du contrôleur (n° du contrôleur)	Files d'attente CPU (Queue No.)
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(1 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)
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-	WK_CPP_POLICE_MULTICAST_SNOOPING(15)	WK_CPU_Q_MCAST_END_STATION_SER

Noms des mappages de classe	Index du contrôleur (n° du contrôleur)	Files d'attente CPU (Queue No.)
end-station		
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)

Chaque file d'attente se rapporte à un type de trafic ou à un ensemble particulier de caractéristiques. La liste n'est pas exhaustive:

Files d'attente du processeur et fonctionnalités associées

Files d'attente CPU (Queue No.)	Fonction(s)
WK_CPU_Q_DOT1X_AUTH(0)	Authentification par port IEEE 802.1x
WK_CPU_Q_L2_CONTROL(1)	Dynamic Trunking Protocol (DTP) Protocole VTP (VLAN Trunking Protocol) Protocole d'agrégation de ports (PAgP) Protocole CISP (Client Information Signaling Protocol) Protocole de relais de session de message

Files d'attente CPU (Queue No.)	Fonction(s)
	Protocole MVRP (Multiple VLAN Registration Protocol) Réseau mobile métropolitain (MMN) Protocole LLDP (Link Level Discovery Protocol) Unidirectional Link Detection (UDLD) Protocole de contrôle d'agrégation de lien (LACP) Cisco Discovery Protocol (CDP) Protocole Spanning Tree (STP)
WK_CPU_Q_FORUS_TRAFFIC(2)	Hôte tel que Telnet, Pingv4 et Pingv6, et SNMP Keepalive / détection de bouclage Protocole IKE (Initiate-Internet Key Exchange) (IPSec)
WK_CPU_Q_ICMP_GEN(3)	ICMP - destination inaccessible ICMP-TTL expiré
WK_CPU_Q_ROUTING_CONTROL(4)	Protocole RIPv1 (Routing Information Protocol version 1) RIPv2 Interior Gateway Routing Protocol (IGRP) Protocole BGP (Border Gateway Protocol) PIM-UDP Virtual Router Redundancy Protocol (VRRP) Protocole HSRPv1 (Hot Standby Router Protocol version 1)

Files d'attente CPU (Queue No.)	Fonction(s)
	HSRPv2 Protocole GLBP (Gateway Load Balancing Protocol) Protocole LDP (Label Distribution Protocol) Protocole WCCP (Web Cache Communication Protocol) Protocole RIPng (Routing Information Protocol Next Generation) Protocole OSPF (Open Shortest Path First) Open Shortest Path First version 3 (OSPFv3) Enhanced Interior Gateway Routing Protocol (EIGRP) Protocole EIGRP version 6 (Enhanced Interior Gateway Routing Protocol) DHCPv6 Protocol Independent Multicast (PIM) Protocole PIMv6 (Protocol Independent Multicast version 6) Protocole HSRPng (Hot Standby Router Protocol nouvelle génération) Contrôle IPv6 keepalive GRE (Generic Routing Encapsulation) Point de traduction d'adresses de réseau (NAT) IS-IS (Intermediate System-to-Intermediate System)
WK_CPU_Q_FORUS_ADDR_RESOLUTION(5)	Protocole ARP (Address Resolution Protocol)

Files d'attente CPU (Queue No.)	Fonction(s)
	Annonce de voisin IPv6 et sollicitation de voisin
WK_CPU_Q_ICMP_REDIRECT(6)	Redirection ICMP (Internet Control Message Protocol)
WK_CPU_Q_INTER_FED_TRAFFIC(7)	Injection de domaine de pont de couche 2 pour la communication interne.
WK_CPU_Q_L2_LVX_CONT_PACK(8)	Paquet XID (Exchange ID)
WK_CPU_Q_EWLC_CONTROL(9)	Contrôleur sans fil intégré (eWLC) [Control and Provisioning of Wireless Access Points (CAPWAP) (UDP 5246)]
WK_CPU_Q_EWLC_DATA(10)	Paquet de données eWLC (CAPWAP DATA, UDP 5247)
WK_CPU_Q_L2_LVX_DATA_PACK(11)	Paquet de monodiffusion inconnu pointé pour la requête de mappage.
WK_CPU_Q_BROADCAST(12)	Tous les types de diffusion
WK_CPU_Q_OPENFLOW(13)	Débordement du cache d'apprentissage (couche 2 + couche 3)
WK_CPU_Q_CONTROLLER_PUNT(14)	Données - liste de contrôle d'accès (ACL) Complète Données - Options IPv4 Données - IPv6 saut par saut Données - hors ressources / attraper tout Données - RPF (Reverse Path Forwarding) incomplet paquet Glean

Files d'attente CPU (Queue No.)	Fonction(s)
WK_CPU_Q_TOPOLOGY_CONTROL(15)	Protocole Spanning Tree (STP) Protocole REP (Resilient Ethernet Protocol) Protocole STP (Shared Spanning Tree Protocol)
WK_CPU_Q_PROTO_SNOOPING(16)	Surveillance ARP (Address Resolution Protocol) pour l'inspection ARP dynamique (DAI)
WK_CPU_Q_DHCP_SNOOPING(17)	Surveillance DHCP
WK_CPU_Q_TRANSIT_TRAFFIC(18)	Il est utilisé pour les paquets pontés par NAT, qui doivent être traités dans le chemin logiciel.
WK_CPU_Q_RPF_FAILED(19)	Données - échec de mRPF (multicast RPF)
WK_CPU_Q_MCAST_END_STATION _SERVICE(20)	Contrôle IGMP (Internet Group Management Protocol) / MLD (Multicast Listener Discovery)
WK_CPU_Q_LOGGING(21)	Journalisation de la liste de contrôle d’accès
WK_CPU_Q_PUNT_WEBAUTH(22)	Authentification Web
WK_CPU_Q_HIGH_RATE_APP(23)	Diffusion
WK_CPU_Q_EXCEPTION(24)	indication IKE Violation d’apprentissage IP Violation de la sécurité des ports IP Violation d'adresse IP statique Vérification de l'étendue IPv6

Files d'attente CPU (Queue No.)	Fonction(s)
	Exception RCP (Remote Copy Protocol) Échec RPF monodiffusion
WK_CPU_Q_SYSTEM_CRITICAL(25)	Signalisation multimédia/Proxy sans fil ARP
WK_CPU_Q_NFL_SAMPLED_DATA(26)	Données échantillonnées Netflow et proxy de services multimédias (MSP)
WK_CPU_Q_LOW_LATENCY(27)	Détection de transfert bidirectionnel (BFD), protocole PTP (Precision Time Protocol)
WK_CPU_Q_EGR_EXCEPTION(28)	Exception de résolution de sortie
WK_CPU_Q_STACKWISE_VIRTUAL_CONTROL(29)	Protocoles d'empilage frontaux, à savoir SVL
WK_CPU_Q_MCAST_DATA(30)	Données - (S, G) création Données - Jointures locales Données - Enregistrement PIM Données - Basculement SPT Données - Multidiffusion
WK_CPU_Q_GOLD_PKT(31)	Or

Stratégie par défaut

Par défaut, la stratégie CoPP générée par le système est appliquée au chemin point/injection. La stratégie par défaut peut être affichée à l'aide de commandes MQC courantes. Elle est également visible dans la configuration du commutateur. La seule stratégie qui peut être appliquée en entrée ou en sortie du CPU/plan de contrôle est la stratégie définie par le système.

Utilisez "show policy-map control-plane" pour afficher la stratégie appliquée au plan de contrôle :

<#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
```

Ajuster CoPP

Les débits du régulateur CoPP peuvent être configurés par l'utilisateur. Les utilisateurs peuvent également désactiver les files d'attente.

Cet exemple montre comment ajuster une valeur de régulateur individuelle. Dans cet exemple, la classe ajustée est « 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
```

Cet exemple montre comment désactiver entièrement une file d'attente. Soyez prudent lorsque vous désactivez les files d'attente, car cela pourrait entraîner une saturation potentielle du processeur.

<#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
```

Dépannage

Méthode

L'utilisation du CPU est affectée par deux activités de base : les processus et l'interruption. Les processus sont des activités structurées que le processeur effectue tandis que l'interruption fait référence aux paquets interceptés sur le plan de données et envoyés au processeur pour action. Ensemble, ces activités comprennent l'utilisation totale du processeur. Puisque CoPP est activé par défaut, un impact de service n'est pas nécessairement corrélé avec une utilisation CPU élevée. Si CoPP fait son travail, l'utilisation du CPU n'est pas grandement affectée. Il est important de tenir compte de l'utilisation globale du processeur, mais cette utilisation ne suffit pas à expliquer tout. Les commandes et utilitaires show de cette section sont utilisés pour évaluer rapidement l'état du processeur et identifier les détails pertinents sur le trafic lié au processeur.

Directives :

- Déterminez si le problème est lié au plan de contrôle. La plupart du trafic de transit est transféré dans le matériel. Seuls certains types de trafic et certains scénarios impliquent le processeur et le plan de contrôle. Gardez cela à l'esprit tout au long de l'enquête.
- Comprendre votre base d'utilisation. Il est important de comprendre à quoi ressemble une utilisation normale afin de pouvoir identifier les écarts par rapport à la norme.

- Validez l'utilisation globale des processus et des interruptions. Identifiez les processus qui prennent en charge des volumes inattendus de cycles CPU. Si l'utilisation se situe en dehors de la plage prévue, cela peut être une source de préoccupation. Il est important de comprendre l'utilisation moyenne d'un système, afin que les écarts en dehors de la norme soient reconnus. Gardez à l'esprit que l'utilisation seule n'est pas une image complète de l'état du plan de contrôle.
- Déterminez s'il y a une augmentation active des abandons dans CoPP. Les abandons CoPP ne sont pas toujours indicatifs d'un problème, mais si vous dépannez un problème lié à une classe de trafic qui est régulée activement, c'est un indicateur fort de pertinence.

Commandes show utiles

Le commutateur permet une surveillance rapide de l'état du processeur et des statistiques CoPP. Il existe également une interface de ligne de commande utile pour déterminer rapidement le point d'entrée du trafic lié au processeur.

Détermination de l'utilisation globale et historique

- La commande « Show processes cpu sorted » permet de visualiser l'utilisation globale du processeur. L'argument "tri" trie la sortie du processus en fonction du pourcentage d'utilisation. Les processus utilisant davantage de ressources CPU sont en haut de la sortie. L'utilisation due aux interruptions est également fournie sous forme de pourcentage.

<#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
```

- La commande « Show processes cpu history » fournit un graphique historique de l'utilisation du processeur au cours des 60 dernières secondes, 5 minutes et 72 heures.

```
<#root>

Catalyst-9600#

show processes cpu history


    999777776666688888666667777777778888877777666669999998888866

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


    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%




    999898989999898989989989898899999898898989999999899999999
```

```
    43182309110263531623512928377133657489280960401423090113351 1
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%
```

```
    99999999999999999999999999999999999999999999999999999999999999999999999999999999
    66565656664655566665565657565455656773755556757454554577595755464857675 7
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%
```

Vérifier la réglementation du plan de contrôle

- Utilisez la commande "show platform hardware fed <switch> active qos queue stats internal cpu policer" pour afficher les statistiques CoPP agrégées et des informations

supplémentaires sur la structure de file d'attente/policer. Cette sortie fournit une vue historique des statistiques du régulateur depuis la dernière réinitialisation du plan de contrôle. Ces compteurs peuvent également être effacés manuellement. Généralement, la preuve des abandons du plan de contrôle par le régulateur indique un problème avec la file d'attente/classe associée, mais assurez-vous que les abandons s'incrémentent activement pendant que le problème se produit. Exécutez la commande plusieurs fois pour observer l'augmentation des valeurs de suppression de la file d'attente.

<#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/
```

Collecte d'informations sur le trafic pointé

Ces commandes sont utilisées pour collecter des informations sur le trafic envoyé au processeur, y compris le type de trafic et les points physiques d'entrée.

- "Show platform software fed <switch> active punt cpuq all" ou "Show platform software fed

<switch> active punt cpuq <0-31 Queue ID>" peuvent être utilisés pour afficher des statistiques relatives à la totalité ou à une file d'attente CPU spécifique.

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

- Utilisez « show platform software fed <switch> active punt cause summary » pour un aperçu rapide de tous les différents types de trafic qui ont été vus au niveau du processeur. Notez que seules les causes non nulles sont affichées.

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

- Utilisez la commande "show platform software fed <switch> active punt rates interfaces" pour afficher rapidement les interfaces où le trafic lié au processeur pénètre dans le système. Cette commande affiche uniquement les interfaces avec une file d'attente d'entrée non nulle.

<#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

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

- Utilisez la commande "show platform software fed <switch> active punt rates interfaces <IF-ID>" pour effectuer une hiérarchisation vers le bas et afficher les files d'attente individuelles de l'interface. Cette commande affiche des statistiques agrégées et peut être utilisée pour afficher l'historique de l'activité de la file d'attente d'entrée et si le trafic a été réglementé.

<#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

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

Inspecter le trafic lié au processeur

La gamme de commutateurs Catalyst 9000 propose des utilitaires permettant de surveiller et de visualiser le trafic lié au processeur. Utilisez ces outils pour comprendre quel trafic est envoyé activement au processeur.

Capture de paquets intégrée (EPC)

L'EPC sur le plan de contrôle peut être effectué dans l'une ou l'autre direction (ou les deux). Pour le trafic pointé, capturez le trafic entrant. EPC sur le plan de contrôle peut être enregistré dans un tampon ou un fichier.

```
<#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
```

Les résultats de la capture peuvent être affichés de manière brève ou détaillée.

```
<#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: 000000000000000000000000000000
    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
```

Les résultats de capture peuvent être écrits directement dans un fichier ou exportés à partir d'une mémoire tampon.

<#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#
```

Capture de paquets CPU FED

La gamme de commutateurs Catalyst 9000 prend en charge un utilitaire de débogage qui permet une meilleure visibilité des paquets en provenance et à destination du processeur.

```
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)
```

Le contenu de la mémoire tampon comporte des options brèves et détaillées pour la sortie.

<#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
```

De nombreux filtres d'affichage sont disponibles. Les filtres d’affichage Wireshark les plus courants sont pris en charge.

<#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>
```

Les filtres peuvent également être appliqués en tant que filtres de capture.

<#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"
```

Scénarios courants

Perte ICMP intermittente (Ping) vers IP locale

Le trafic qui est transféré à une adresse IP locale sur un commutateur est placé dans la file d'attente Forus (littéralement « pour nous »). L'incrémentation de la file d'attente CoPP Forus concerne les paquets abandonnés destinés au commutateur local. C'est relativement simple et facile à conceptualiser.

Dans certaines conditions, cependant, il pourrait y avoir une perte de trafic destiné localement qui n'est pas clairement corrélée avec les abandons Forus.

Avec un flux de trafic lié au CPU suffisant, le chemin de point devient sursaturé au-delà de la capacité de CoPP à hiérarchiser le trafic qui est réglementé. Le trafic est régulé « en silence » sur la base du « premier entré, premier sorti ».

Dans ce scénario, des preuves de la réglementation du plan de contrôle dans un volume élevé sont observées, mais le type de trafic d'intérêt (Forus dans cet exemple) n'augmente pas activement nécessairement.

En résumé, s'il y a un volume exceptionnellement élevé de trafic lié au CPU, mis en évidence par la réglementation CoPP active et démontré avec une capture de paquets ou un débogage ponctuel FED, il peut y avoir une perte qui ne correspond pas à la file d'attente que vous dépannez. Dans ce scénario, déterminez pourquoi il y a une quantité excessive de trafic lié au CPU et prenez des mesures pour alléger la charge sur le plan de contrôle.

Redirections ICMP élevées et fonctionnement DHCP lent

La CoPP sur le commutateur de la gamme Catalyst 9000 est organisée en 32 files d'attente matérielles. Ces 32 files d'attente matérielles correspondent à 20 index de régulateur individuels. Chaque index de régulateur est mis en corrélation avec une ou plusieurs files d'attente matérielles.

Fonctionnellement, cela signifie que plusieurs classes de trafic partagent un index de régulateur et sont soumises à une valeur de régulateur agrégée commune.

Un problème courant rencontré sur les commutateurs avec des agents de relais DHCP activés implique une réponse DHCP lente. Les clients peuvent obtenir des adresses IP de manière sporadique, mais plusieurs tentatives sont nécessaires et certains clients expirent.

La file d'attente de redirection ICMP et la file d'attente de diffusion partagent un index de

régulateur, de sorte qu'un volume élevé de trafic qui est reçu et routé à partir de la même interface virtuelle de commutateur (SVI) affecte les applications qui dépendent du trafic de diffusion. Ceci est particulièrement visible lorsque le commutateur agit en tant qu'agent de relais.

Ce document offre une explication détaillée du concept et de la façon d'atténuer : Dépannage des problèmes DHCP sur les agents de relais DHCP de Catalyst 9000

Ressources supplémentaires

Dépannage du protocole DHCP lent ou intermittent sur les agents de relais DHCP du Catalyst 9000

Configuration de la capture de paquets CPU FED sur les commutateurs Catalyst 9000

Commutateurs Catalyst 9300 : configuration de la réglementation du plan de contrôle

Configuration de la capture de paquets - Guide de configuration de la gestion du réseau, Cisco IOS XE Bengaluru 17.6.x (commutateurs Catalyst 9300)

Fonctionnement et dépannage de la surveillance DHCP sur les commutateurs Catalyst 9000

À propos de cette traduction
Cisco a traduit ce document en traduction automatisée vérifiée par une personne dans le cadre d'un service mondial permettant à nos utilisateurs d'obtenir le contenu d'assistance dans leur propre langue.

Il convient cependant de noter que même la meilleure traduction automatisée ne sera pas aussi précise que celle fournie par un traducteur professionnel.