

Résolution des problèmes de lenteur TCP dus au réglage MSS dans les commutateurs Catalyst 9K

Table des matières

[Introduction](#)

[Informations sur l'ajustement MSS TCP](#)

[Comportement](#)

[Topologie](#)

[Scénario](#)

[Configuration et comportement initiaux](#)

[Comportement après ajustement TCP MSS](#)

[Ajustement TCP MSS entraînant une lenteur lors d'un trafic TCP important](#)

[Points importants](#)

Introduction

Ce document décrit comment un commutateur Catalyst 9K effectue le réglage MSS TCP, et comment la lenteur TCP est liée à cette fonctionnalité.

Informations sur l'ajustement MSS TCP

La fonction d'ajustement de la taille maximale de segment (MSS) du protocole TCP (Transmission Control Protocol) permet de configurer la taille maximale de segment pour les paquets transitoires qui traversent un routeur, en particulier les segments TCP avec le bit SYN défini. La commande est utilisée en mode de configuration d'interface pour spécifier la valeur MSS sur le routeur intermédiaire des paquets SYN afin d'éviter la troncature `ip tcp adjust-mss`.

Lorsqu'un hôte (généralement un PC) lance une session TCP avec un serveur, il négocie la taille du segment IP à l'aide du champ d'option MSS dans le paquet TCP SYN. La configuration MTU sur l'hôte détermine la valeur du champ MSS. La valeur MTU par défaut d'une carte réseau du PC est de 1 500 octets avec une valeur MSS TCP de 1 460 (1 500 octets - en-tête IP de 20 octets - en-tête TCP de 20 octets).

La norme PPP over Ethernet (PPPoE) prend en charge une MTU de seulement 1 492 octets.

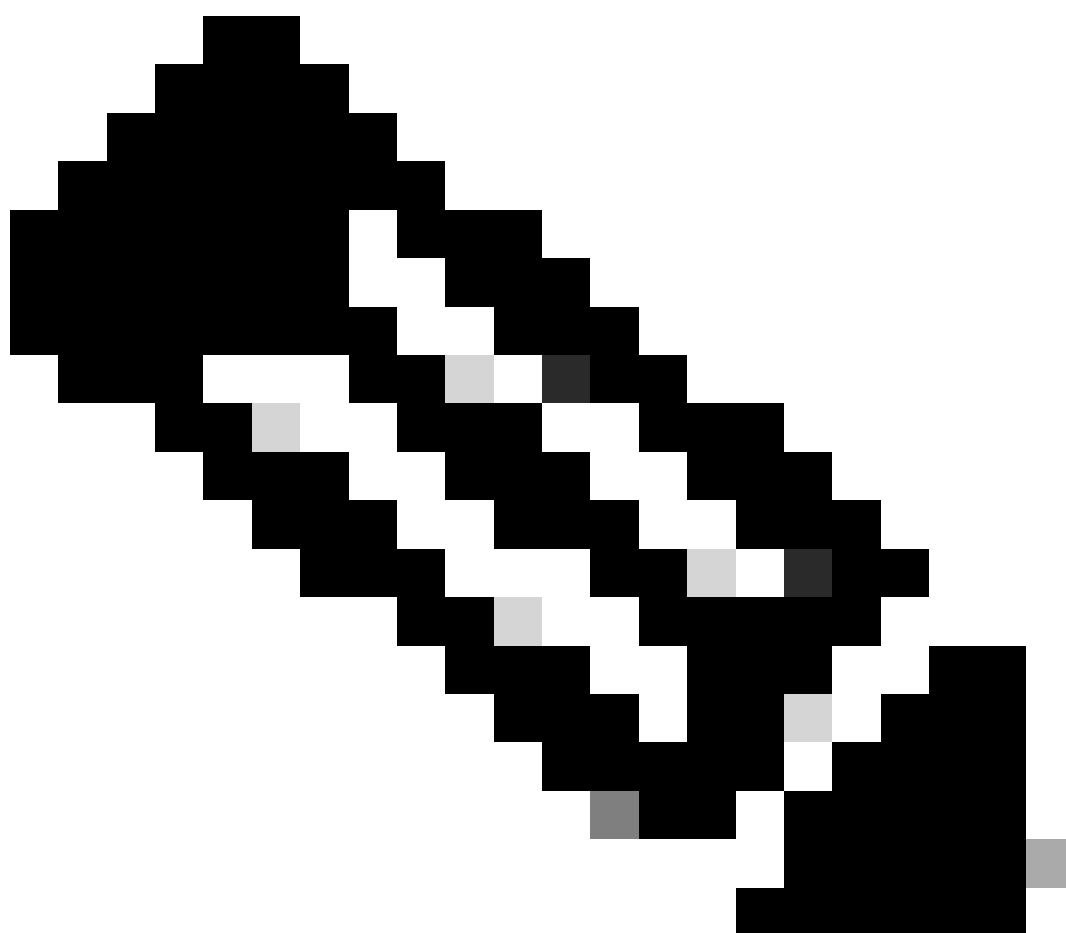
La disparité entre la taille de l'hôte et celle de la MTU PPPoE peut entraîner l'abandon de paquets de 1 500 octets par le routeur situé entre l'hôte et le serveur et l'interruption de sessions TCP sur le réseau PPPoE.

Même si le MTU du chemin (qui détecte le MTU correct sur le chemin) est activé sur l'hôte, les

sessions peuvent être abandonnées car les administrateurs système désactivent parfois les messages d'erreur ICMP (Internet Control Message Protocol) qui doivent être relayés depuis l'hôte pour que le MTU du chemin fonctionne.

La commande `ip tcp adjust-mss` aide à empêcher les sessions TCP d'être abandonnées en ajustant la valeur MSS des paquets TCP SYN. La commande `ip tcp adjust-mss` n'est efficace que pour les connexions TCP passant par le routeur. Dans la plupart des cas, la valeur optimale pour l'argument `max-segment-size` de la commande `ip tcp adjust-mss` est de 1 452 octets.

Cette valeur ajoutée à l'en-tête IP de 20 octets, à l'en-tête TCP de 20 octets et à l'en-tête PPPoE de 8 octets s'ajoute à un paquet de 1 500 octets qui correspond à la taille MTU de la liaison Ethernet.



Remarque : le trafic basé sur l'ajustement MSS TCP est commuté par logiciel dans les commutateurs Catalyst 9K. Ce document explique des scénarios qui supposent que le trafic basé sur l'ajustement de MSS TCP est commuté par logiciel. Reportez-vous au Guide de configuration afin de confirmer si un logiciel matériel/logiciel spécifique commute le trafic TCP MSS basé sur l'ajustement.

Comportement

Comme mentionné précédemment, le trafic basé sur les ajustements MSS TCP est toujours commuté par logiciel.

Cela signifie que si vous essayez d'effectuer un ajustement TCP, le commutateur envoie le trafic TCP au processeur pour la modification MSS.

Par exemple, si vous modifiez la valeur MSS TCP sur une interface, tout le trafic TCP reçu sur cette interface est envoyé au processeur.

Le processeur modifie ensuite la valeur MSS et envoie le trafic à l'interface requise vers laquelle le paquet TCP était dirigé.

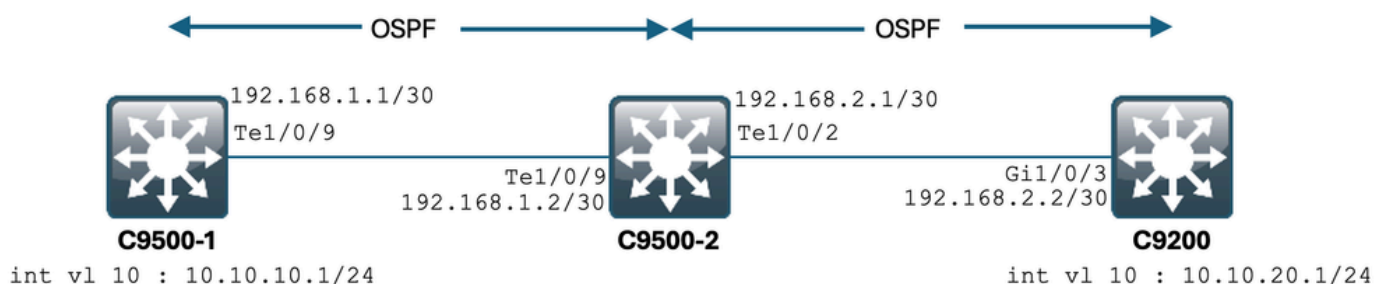
Pour cette raison, s'il y a une énorme quantité de trafic TCP avec l'ajustement MSS, alors cela surcharge la file d'attente CPU.

Lorsqu'une file d'attente de CPU est surchargée, le contrôleur de plan de contrôle (COPP) gère ce trafic et abandonne les paquets pour maintenir le débit du contrôleur de file d'attente. Cela entraîne l'abandon des paquets TCP.

Par conséquent, des problèmes tels que la lenteur du transfert de fichiers, les créations de session SSH et la lenteur de l'application Citrix (si vous utilisez TCP) sont observés.

Un exemple réel de la façon dont cela se produit est montré ici.

Topologie



Scénario

Vous allez SSH dans le C9200 à partir du C9500-1.

SSH utilisant le VLAN 10 (10.10.10.1) de C9500-1 comme source.

La destination du SSH est le VLAN 20 de C9200 (10.10.20.1/24).

SSH étant basé sur TCP, toute lenteur dans TCP affecte également la création de cette session SSH.

Il existe un commutateur C3 de transit (C9500-2) entre C9500-1 et C9200.

Il existe deux liaisons C3 de transit /30, l'une entre C9500-1 et C9500-2 et l'autre entre C9500-2 et

C9200.

Le protocole OSPF est utilisé pour l'accessibilité sur les trois commutateurs, et tous les sous-réseaux et SVI 30/30 sont annoncés dans le protocole OSPF.

Toutes les adresses IP indiquées précédemment sont accessibles entre elles.

Dans C9500-2 Te1/0/9, la modification de la valeur MSS TCP est effectuée.

Lorsque le SSH de C9500-1 est initié, une connexion TCP en trois étapes se produit.

Le paquet SYN atteint le C9500-2 Te1/0/9 (en entrée), où le réglage MSS TCP est effectué.

Configuration et comportement initiaux

Une capture EPC sur C9500-2 Te1/0/9 (dans les deux directions) a été effectuée et SSH de C9500-1 à C9200 a été lancé.

Voici la configuration EPC :

```
C9500-2#show monitor capture mycap
Status Information for Capture mycap
Target Type:
Interface: TenGigabitEthernet1/0/9, Direction: BOTH
Status : Inactive
Filter Details:
Capture all packets
Buffer Details:
Buffer Type: LINEAR (default)
Buffer Size (in MB): 80
File Details:
File not associated
Limit Details:
Number of Packets to capture: 0 (no limit)
Packet Capture duration: 0 (no limit)
Packet Size to capture: 0 (no limit)
Maximum number of packets to capture per second: 1000
Packet sampling rate: 0 (no sampling)
C9500-2#
```

Démarrage de l'EPC :

```
C9500-2#monitor capture mycap start
Started capture point : mycap
C9500-2#
```

Démarrage du SSH de C9500-1 à C9200 :

```
C9500-1#ssh -l admin 10.10.20.1
Password:
```

Arrêt de l'EPC :

```
C9500-2#monitor capture mycap stop
Capture statistics collected at software:
Capture duration - 6 seconds
Packets received - 47
Packets dropped - 0
Packets oversized - 0
Bytes dropped in asic - 0
Capture buffer will exist till exported or cleared
Stopped capture point : mycap
C9500-2#
```

Et voici les paquets capturés par l'EPC :

```
C9500-2#show monitor capture mycap buffer brief
Starting the packet display ..... Press Ctrl + Shift + 6 to exit
1 0.000000 10.10.10.1 -> 10.10.20.1 TCP 60 44274 -> 22 [SYN] Seq=0 Win=4128 Len=0 MSS=536
2 0.001307 10.10.20.1 -> 10.10.10.1 TCP 60 22 -> 44274 [SYN, ACK] Seq=0 Ack=1 Win=4128 Len=0 MSS=536
3 0.001564 10.10.10.1 -> 10.10.20.1 TCP 60 44274 -> 22 [ACK] Seq=1 Ack=1 Win=4128 Len=0
4 0.003099 10.10.20.1 -> 10.10.10.1 SSH 73 Server: Protocol (SSH-2.0-Cisco-1.25)
5 0.003341 10.10.10.1 -> 10.10.20.1 SSH 73 Client: Protocol (SSH-2.0-Cisco-1.25)
6 0.003419 10.10.10.1 -> 10.10.20.1 TCP 118 [TCP segment of a reassembled PDU]
7 0.003465 10.10.10.1 -> 10.10.20.1 TCP 118 44274 -> 22 [ACK] Seq=84 Ack=20 Win=4109 Len=64 [TCP segment of a reassembled PDU]
8 0.003482 10.10.10.1 -> 10.10.20.1 TCP 118 44274 -> 22 [ACK] Seq=148 Ack=20 Win=4109 Len=64 [TCP segment of a reassembled PDU]
9 0.003496 10.10.10.1 -> 10.10.20.1 TCP 118 44274 -> 22 [ACK] Seq=212 Ack=20 Win=4109 Len=64 [TCP segment of a reassembled PDU]
10 0.003510 10.10.10.1 -> 10.10.20.1 TCP 118 44274 -> 22 [ACK] Seq=276 Ack=20 Win=4109 Len=64 [TCP segment of a reassembled PDU]
11 0.003525 10.10.10.1 -> 10.10.20.1 TCP 118 44274 -> 22 [ACK] Seq=340 Ack=20 Win=4109 Len=64 [TCP segment of a reassembled PDU]
12 0.004719 10.10.20.1 -> 10.10.10.1 TCP 60 22 -> 44274 [ACK] Seq=20 Ack=84 Win=4045 Len=0
~ Output Cut ~
```

Vous pouvez voir la connexion TCP se produire dans le paquet numéro 1,2,3.

Le paquet n° 1 est le paquet SYN.

Vous pouvez voir qu'il est fourni avec une valeur MSS de 536.

Le paquet SYN, ACK (paquet n° 2) est également vu en provenance du C9200 avec une valeur MSS de 536.

Ici, la valeur MSS reste intacte et n'est pas modifiée par le commutateur.

Comportement après ajustement TCP MSS

Voici la configuration d'ajustement TCP MSS sur C9500-2 Te1/0/9 :

```
C9500-2#sh run int te1/0/9
Building configuration...
Current configuration : 119 bytes
!
```

```
interface TenGigabitEthernet1/0/9
no switchport
ip address 192.168.1.2 255.255.255.252
ip tcp adjust-mss 512 -----> Here we are changing the MSS value to 512.
```

Maintenant, prenez une capture EPC sur C9500-2 Te1/0/9 (dans les deux directions), et démarrez SSH de C9500-1 à C9200.

Voici la configuration EPC :

```
C9500-2#show monitor capture mycap
Status Information for Capture mycap
Target Type:
Interface: TenGigabitEthernet1/0/9, Direction: BOTH
Status : Inactive
Filter Details:
Capture all packets
Buffer Details:
Buffer Type: LINEAR (default)
Buffer Size (in MB): 80
File Details:
File not associated
Limit Details:
Number of Packets to capture: 0 (no limit)
Packet Capture duration: 0 (no limit)
Packet Size to capture: 0 (no limit)
Maximum number of packets to capture per second: 1000
Packet sampling rate: 0 (no sampling)
C9500-2#
```

Démarrez la capture, établissez une connexion SSH entre C9500-1 et C9200, puis arrêtez la capture.

Voici les paquets capturés par le processeur :

```
C9500-2#show monitor capture mycap buffer brief
Starting the packet display ..... Press Ctrl + Shift + 6 to exit
1 0.000000 b8:a3:77:ec:ba:f7 -> 01:00:0c:cc:cc:cc CDP 398 Device ID: C9500-1.cisco.com Port ID: TenGiga
2 0.636138 10.10.10.1 -> 10.10.20.1 TCP 60 53865 -> 22 [SYN] Seq=0 Win=4128 Len=0 MSS=536
3 0.637980 10.10.20.1 -> 10.10.10.1 TCP 60 22 -> 53865 [SYN, ACK] Seq=0 Ack=1 Win=4128 Len=0 MSS=512
4 0.638214 10.10.10.1 -> 10.10.20.1 TCP 60 53865 -> 22 [ACK] Seq=1 Ack=1 Win=4128 Len=0
5 0.639997 10.10.20.1 -> 10.10.10.1 SSH 73 Server: Protocol (SSH-2.0-Cisco-1.25)
6 0.640208 10.10.10.1 -> 10.10.20.1 SSH 73 Client: Protocol (SSH-2.0-Cisco-1.25)
7 0.640286 10.10.10.1 -> 10.10.20.1 TCP 118 [TCP segment of a reassembled PDU]
8 0.640341 10.10.10.1 -> 10.10.20.1 TCP 118 53865 -> 22 [ACK] Seq=84 Ack=20 Win=4109 Len=64 [TCP segmen
9 0.640360 10.10.10.1 -> 10.10.20.1 TCP 118 53865 -> 22 [ACK] Seq=148 Ack=20 Win=4109 Len=64 [TCP segmen
10 0.640375 10.10.10.1 -> 10.10.20.1 TCP 118 53865 -> 22 [ACK] Seq=212 Ack=20 Win=4109 Len=64 [TCP segmen
11 0.640390 10.10.10.1 -> 10.10.20.1 TCP 118 53865 -> 22 [ACK] Seq=276 Ack=20 Win=4109 Len=64 [TCP segmen
12 0.640410 10.10.10.1 -> 10.10.20.1 TCP 118 53865 -> 22 [ACK] Seq=340 Ack=20 Win=4109 Len=64 [TCP segmen
~ Output Cut ~
```

Vous pouvez voir la connexion TCP se produire dans les paquets numéro 2,3,4.

Le paquet n° 2 est le paquet SYN.

Vous pouvez voir qu'il est fourni avec une valeur MSS de 536.

Mais le paquet SYN, ACK (paquet n° 3) est vu en provenance du C9200 avec une valeur MSS de 512.

En effet, lorsque le paquet SYN atteint le Te1/0/9 C9500-2, il est envoyé au processeur du C9500-2 pour une modification MSS TCP de 536 à 512.

Le processeur du C9500-2 change le MSS en 512 et envoie le paquet SYN de Te1/0/2 vers C9200.

Toutes les transactions TCP suivantes utilisent la même valeur MSS modifiée.

Maintenant, nous allons approfondir la façon dont le paquet SYN traverse le commutateur et le changement MSS se produit.

Une fois que ce paquet SYN atteint l'interface du C9500-2, il est envoyé au processeur pour modification MSS.

Il passe d'abord par le FED (où vous pouvez le capturer), puis va au CPU (où vous pouvez le capturer également).

Prenons d'abord une capture de point FED sur C9500-2.

Voici la configuration de la capture ponctuelle FED :

```
C9500-2#debug platform software fed switch 1 punt packet-capture buffer limit 16384
Punt PCAP buffer configure: one-time with buffer size 16384...done
```

Démarrage de la capture ponctuelle FED :

```
C9500-2#debug platform software fed switch 1 punt packet-capture start
Punt packet capturing started.
```

Démarrage du SSH de C9500-1 à C9200 :

```
C9500-1#ssh -l admin 10.10.20.1
Password:
```

Arrêt de la capture de point FED :

```
C9500-2#debug platform software fed switch 1 punt packet-capture stop
Punt packet capturing stopped. Captured 3 packet(s)
```

Et voici les paquets capturés par la sonde de la FED :

```
C9500-2#show platform software fed switch active punt packet-capture brief
Punt packet capturing: disabled. Buffer wrapping: disabled
Total captured so far: 3 packets. Capture capacity : 16384 packets
```

```
----- Punt Packet Number: 1, Timestamp: 2024/07/31 01:29:46.373 -----
interface : physical: TenGigabitEthernet1/0/9[if-id: 0x00000040], pal: TenGigabitEthernet1/0/9 [if-id: 0x00000040]
metadata : cause: 55 [For-us control], sub-cause: 0, q-no: 4, linktype: MCP_LINK_TYPE_IP [1]
ether hdr : dest mac: 0100.5e00.0005, src mac: b8a3.77ec.baf7
ether hdr : ethertype: 0x0800 (IPv4)
ipv4 hdr : dest ip: 224.0.0.5, src ip: 192.168.1.1
ipv4 hdr : packet len: 100, ttl: 1, protocol: 89
```

```
----- Punt Packet Number: 2, Timestamp: 2024/07/31 01:29:47.432 -----
interface : physical: TenGigabitEthernet1/0/9[if-id: 0x00000040], pal: TenGigabitEthernet1/0/9 [if-id: 0x00000040]
metadata : cause: 11 [For-us data], sub-cause: 1, q-no: 14, linktype: MCP_LINK_TYPE_IP [1]
ether hdr : dest mac: 00a3.d144.4bf7, src mac: b8a3.77ec.baf7
ether hdr : ethertype: 0x0800 (IPv4)
ipv4 hdr : dest ip: 10.10.20.1, src ip: 10.10.10.1
ipv4 hdr : packet len: 44, ttl: 254, protocol: 6 (TCP)
tcp hdr : dest port: 22, src port: 35916
```

```
----- Punt Packet Number: 3, Timestamp: 2024/07/31 01:29:48.143 -----
interface : physical: TenGigabitEthernet1/0/1[if-id: 0x00000009], pal: TenGigabitEthernet1/0/1 [if-id: 0x00000009]
metadata : cause: 96 [Layer2 control protocols], sub-cause: 0, q-no: 1, linktype: MCP_LINK_TYPE_LAYER2
ether hdr : dest mac: 0100.0ccc.cccc, src mac: 78bc.1a27.c203
ether hdr : length: 443
```

Vous pouvez voir que le paquet n° 2 est le paquet TCP SYN de 10.10.10.1 à 10.10.20.1, provenant de Te1/0/9.

Le « q-no » est important à noter ici. Vous pouvez voir qu'il choisit la file d'attente n° 14 pour passer du FED au CPU.

Vous pouvez voir ici toutes les 32 files d'attente présentes pour le trafic à déplacer du FED vers le CPU :

```
C9500-2#show platform hardware fed switch 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)
```

```
-----
0 11 DOT1X Auth Yes 1000 1000 0 0
1 1 L2 Control Yes 2000 2000 0 0
2 14 Forus traffic Yes 4000 4000 0 0
3 0 ICMP GEN Yes 600 600 0 0
4 2 Routing Control Yes 5400 5400 0 0
5 14 Forus Address resolution Yes 4000 4000 0 0
6 0 ICMP Redirect Yes 600 600 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 600 600 0 0
13 10 Openflow Yes 200 200 0 0
14 13 Sw forwarding Yes 1000 1000 0 0
15 8 Topology Control Yes 13000 13000 0 0
16 12 Proto Snooping Yes 2000 2000 0 0
17 6 DHCP Snooping Yes 400 400 0 0
18 13 Transit Traffic Yes 1000 1000 0 0
19 10 RPF Failed Yes 200 200 0 0
20 15 MCAST END STATION Yes 2000 2000 0 0
21 13 LOGGING Yes 1000 1000 0 0
22 7 Punt Webauth Yes 1000 1000 0 0
23 18 High Rate App Yes 13000 13000 0 0
24 10 Exception Yes 200 200 0 0
25 3 System Critical Yes 1000 1000 0 0
26 10 NFL SAMPLED DATA Yes 200 200 0 0
27 2 Low Latency Yes 5400 5400 0 0
28 10 EGR Exception Yes 200 200 0 0
29 5 Stackwise Virtual OOB Yes 8000 8000 0 0
30 9 MCAST Data Yes 400 400 0 0
31 3 Gold Pkt Yes 1000 1000 0 0

```

Comme vous pouvez le voir, la file d'attente n° 14 est la file d'attente « Sw forwarding ». Dans ce cas, cette file d'attente est utilisée par le trafic TCP afin d'être envoyée au CPU.

Maintenant, prenons une capture de CPU (Control-Plane) sur C9500-2.

Voici la configuration de capture du processeur :

```

C9500-2#sh mon cap test
Status Information for Capture test
Target Type:
Interface: Control Plane, Direction: BOTH
Status : Inactive
Filter Details:
Capture all packets
Buffer Details:
Buffer Type: LINEAR (default)
Buffer Size (in MB): 80
File Details:
File not associated
Limit Details:
Number of Packets to capture: 0 (no limit)
Packet Capture duration: 0 (no limit)
Packet Size to capture: 0 (no limit)
Packet sampling rate: 0 (no sampling)
C9500-2#

```

Démarrez la capture, passez de C9500-1 à C9200 par SSH, puis arrêtez la capture.

Voici les paquets capturés par le processeur :

```

C9500-2#show monitor capture test buffer brief
Starting the packet display ..... Press Ctrl + Shift + 6 to exit
 1 0.000000 00:a3:d1:44:4b:81 -> 01:80:c2:00:00:00 STP 60 RST. Root = 32768/1/00:a3:d1:44:4b:80 Cost = 0
 2 0.000010 00:a3:d1:44:4b:a3 -> 01:80:c2:00:00:00 STP 60 RST. Root = 32768/1/00:a3:d1:44:4b:80 Cost = 0
 3 0.000013 00:a3:d1:44:4b:a4 -> 01:80:c2:00:00:00 STP 60 RST. Root = 32768/1/00:a3:d1:44:4b:80 Cost = 0
 4 0.000016 00:a3:d1:44:4b:a6 -> 01:80:c2:00:00:00 STP 60 RST. Root = 32768/1/00:a3:d1:44:4b:80 Cost = 0
 5 0.000019 00:a3:d1:44:4b:a7 -> 01:80:c2:00:00:00 STP 60 RST. Root = 32768/1/00:a3:d1:44:4b:80 Cost = 0
 6 0.000022 00:a3:d1:44:4b:a8 -> 01:80:c2:00:00:00 STP 60 RST. Root = 32768/1/00:a3:d1:44:4b:80 Cost = 0
 7 0.055470 c0:8b:2a:04:f0:6c -> 01:80:c2:00:00:0e LLDP 117 TTL = 120 SysName = bg118-cx-amx-b02-2.cisco
 9 0.220331 28:63:29:20:31:39 -> 00:01:22:53:74:20 0x3836 30 Ethernet II
10 0.327316 192.168.1.1 -> 224.0.0.5 OSPF 114 Hello Packet
11 0.442986 c0:8b:2a:04:f0:68 -> 01:80:c2:00:00:0e LLDP 117 TTL = 120 SysName = bg118-cx-amx-b02-2.cisco
12 1.714121 10.10.10.1 -> 10.10.20.1 TCP 60 23098 -> 22 [SYN] Seq=0 Win=4128 Len=0 MSS=536
13 1.714375 10.10.10.1 -> 10.10.20.1 TCP 60 [TCP Out-Of-Order] 23098 -> 22 [SYN] Seq=0 Win=4128 Len=0 MSS=512
14 2.000302 00:a3:d1:44:4b:81 -> 01:80:c2:00:00:00 STP 60 RST. Root = 32768/1/00:a3:d1:44:4b:80 Cost = 0
15 2.000310 00:a3:d1:44:4b:a3 -> 01:80:c2:00:00:00 STP 60 RST. Root = 32768/1/00:a3:d1:44:4b:80 Cost = 0
~ Output Cut ~

```

Le paquet n° 12 est le paquet TCP SYN entrant dans le processeur (punt), avec la valeur MSS par défaut de 536.

Le paquet n° 13 est le paquet TCP SYN envoyé par le CPU (injection), après avoir modifié la valeur MSS à 512.

Vous pouvez également effectuer un débogage rapide du processeur afin de voir ce changement se produire.

Voici la configuration de débogage du processeur :

```

C9500-2#debug ip tcp adjust-mss
TCP Adjust Mss debugging is on

```

Démarrage du SSH de C9500-1 à C9200 :

```

C9500-1#ssh -l admin 10.10.20.1
Password:

```

Arrêt du débogage du processeur :

```

C9500-2#undebug all
All possible debugging has been turned off

```

Recherche des débogages dans les journaux :

```

C9500-2#show logging
Syslog logging: enabled (0 messages dropped, 2 messages rate-limited, 0 flushes, 0 overruns, xml disabled)
No Active Message Discriminator.
No Inactive Message Discriminator.
Console logging: disabled
Monitor logging: level debugging, 0 messages logged, xml disabled,
filtering disabled
Buffer logging: level debugging, 230 messages logged, xml disabled,
filtering disabled
Exception Logging: size (4096 bytes)
Count and timestamp logging messages: disabled
File logging: disabled
Persistent logging: disabled
No active filter modules.
Trap logging: level informational, 210 message lines logged
Logging Source-Interface: VRF Name:
TLS Profiles:
Log Buffer (102400 bytes):
*Jul 31 01:46:32.052: TCPADJMSS: process_enqueue_feature
*Jul 31 01:46:32.893: TCPADJMSS: process_enqueue_feature
*Jul 31 01:46:36.136: TCPADJMSS: process_enqueue_feature
*Jul 31 01:46:41.318: TCPADJMSS: process_enqueue_feature
*Jul 31 01:46:42.412: TCPADJMSS: process_enqueue_feature
*Jul 31 01:46:43.254: TCPADJMSS: process_enqueue_feature
*Jul 31 01:46:43.638: TCPADJMSS: process_enqueue_feature
*Jul 31 01:46:45.783: TCPADJMSS: Input (process)
*Jul 31 01:46:45.783: TCPADJMSS: orig_mss = 536 adj_mss = 512 src_ip = 10.10.10.1 dest_ip = 10.10.20.1
*Jul 31 01:46:45.783: TCPADJMSS: paktype = 0x7F83C7BCBF78
*Jul 31 01:46:50.456: TCPADJMSS: process_enqueue_feature
*Jul 31 01:46:51.985: TCPADJMSS: process_enqueue_feature
C9500-2#

```

Vous pouvez voir que la valeur MSS d'origine de 536 est ajustée à 512.

Enfin, vous pouvez effectuer une capture EPC sur C9200 Gi1/0/3 afin de confirmer que le paquet TCP SYN arrive bien avec une MSS de 512.

Voici la configuration EPC :

```

C9200#sh mon cap mycap
Status Information for Capture mycap
Target Type:
Interface: GigabitEthernet1/0/3, Direction: BOTH
Status : Inactive
Filter Details:
Capture all packets
Buffer Details:
Buffer Type: LINEAR (default)
Buffer Size (in MB): 80
Limit Details:
Number of Packets to capture: 0 (no limit)
Packet Capture duration: 0 (no limit)
Packet Size to capture: 0 (no limit)
Packet sampling rate: 0 (no sampling)

```

C9200#

Démarrez la capture, passez de C9500-1 à C9200 par SSH, puis arrêtez la capture.

Voici les paquets capturés par le processeur :

```
C9200#sh mon cap mycap buff br
```

```
-----  
# size timestamp source destination dscp protocol  
-----  
0 118 0.000000 192.168.2.1 -> 224.0.0.5 48 CS6 OSPF  
1 64 0.721023 10.10.10.1 -> 10.10.20.1 48 CS6 TCP  
2 64 0.722015 10.10.10.1 -> 10.10.20.1 48 CS6 TCP  
3 77 0.728026 10.10.10.1 -> 10.10.20.1 48 CS6 TCP  
4 122 0.728026 10.10.10.1 -> 10.10.20.1 48 CS6 TCP  
5 122 0.728026 10.10.10.1 -> 10.10.20.1 48 CS6 TCP  
6 122 0.728026 10.10.10.1 -> 10.10.20.1 48 CS6 TCP  
7 122 0.728026 10.10.10.1 -> 10.10.20.1 48 CS6 TCP  
8 122 0.728026 10.10.10.1 -> 10.10.20.1 48 CS6 TCP  
9 122 0.728026 10.10.10.1 -> 10.10.20.1 48 CS6 TCP  
10 122 0.730025 10.10.10.1 -> 10.10.20.1 48 CS6 TCP  
~ Output Cut ~
```

Dans C9200, vous ne pouvez pas voir les détails du paquet comme dans Wireshark, seuls les détails brefs et hexadécimaux sont disponibles.

Vous pouvez donc exporter les paquets précédents vers un fichier pcap dans la mémoire flash.

```
C9200#mon cap mycap export flash:Gi1-0-3-Both.capng
```

Exportation réussie

Ensuite, vous pouvez copier ce fichier via TFTP sur votre PC local, et ouvrir le fichier dans Wireshark.

Voici la capture Wireshark.

The screenshot shows a network traffic analysis tool interface. At the top, there's a toolbar and a search bar. Below that, a table lists network packets with columns for No., Time, Source, Destination, Protocol, Length, ID, Message type, and Info. The selected packet (No. 2) is a SYN packet from 192.168.2.1 to 10.10.20.1. Below the table, a detailed view of the packet is shown, including Ethernet II, Internet Protocol Version 4, and Transmission Control Protocol details. The TCP Flags section shows 'SYN' and the MSS Value is highlighted as 512.

No.	Time	Source	Destination	Protocol	Length	ID	Message type	Info
1	2024-07-31 05:21:46.915937	192.168.2.1	224.0.0.5	OSPF	118			Hello Packet
2	2024-07-31 05:21:47.636960	10.10.10.1	10.10.20.1	TCP	64			37885 → 22 [SYN] Seq=0 Win=4128 Len=0 MSS=512 [Packet size limited during capture]
3	2024-07-31 05:21:47.637952	10.10.10.1	10.10.20.1	TCP	64			37885 → 22 [ACK] Seq=1 Ack=1 Win=4128 Len=0 [Packet size limited during capture]
4	2024-07-31 05:21:47.643963	10.10.10.1	10.10.20.1	SSHv2	77			Client: Protocol (SSH-2.0-Cisco-1.25)
5	2024-07-31 05:21:47.643963	10.10.10.1	10.10.20.1	TCP	122			37885 → 22 [ACK] Seq=20 Ack=20 Win=4109 Len=64 [TCP segment of a reassembled PDU]
6	2024-07-31 05:21:47.643963	10.10.10.1	10.10.20.1	TCP	122			37885 → 22 [ACK] Seq=84 Ack=20 Win=4109 Len=64 [TCP segment of a reassembled PDU]
7	2024-07-31 05:21:47.643963	10.10.10.1	10.10.20.1	TCP	122			37885 → 22 [ACK] Seq=148 Ack=20 Win=4109 Len=64 [TCP segment of a reassembled PDU]
8	2024-07-31 05:21:47.643963	10.10.10.1	10.10.20.1	TCP	122			37885 → 22 [ACK] Seq=212 Ack=20 Win=4109 Len=64 [TCP segment of a reassembled PDU]
9	2024-07-31 05:21:47.643963	10.10.10.1	10.10.20.1	TCP	122			37885 → 22 [ACK] Seq=276 Ack=20 Win=4109 Len=64 [TCP segment of a reassembled PDU]
10	2024-07-31 05:21:47.643963	10.10.10.1	10.10.20.1	TCP	122			37885 → 22 [ACK] Seq=340 Ack=20 Win=4109 Len=64 [TCP segment of a reassembled PDU]
11	2024-07-31 05:21:47.645962	10.10.10.1	10.10.20.1	TCP	122			37885 → 22 [ACK] Seq=404 Ack=20 Win=4109 Len=64 [TCP segment of a reassembled PDU]
12	2024-07-31 05:21:47.645962	10.10.10.1	10.10.20.1	TCP	122			37885 → 22 [ACK] Seq=468 Ack=20 Win=4109 Len=64 [TCP segment of a reassembled PDU]
13	2024-07-31 05:21:47.645962	10.10.10.1	10.10.20.1	SSHv2	114			Client: Key Exchange Init
14	2024-07-31 05:21:47.648953	10.10.10.1	10.10.20.1	TCP	64			37885 → 22 [ACK] Seq=588 Ack=524 Win=4128 Len=0 [Packet size limited during capture]

Frame 2: 64 bytes on wire (512 bits), 60 bytes captured (480 bits)
 Ethernet II, Src: Cisco_44:4b:d6 (00:a3:d1:44:4b:d6), Dst: Cisco_fd:72:d8 (5c:5a:c7:fd:72:d8)
 Internet Protocol Version 4, Src: 10.10.10.1, Dst: 10.10.20.1
 Transmission Control Protocol, Src Port: 37885, Dst Port: 22, Seq: 0, Len: 0
 Source Port: 37885
 Destination Port: 22
 [Stream Index: 0]
 [TCP Segment Len: 0]
 Sequence Number: 0 (relative sequence number)
 Sequence Number (raw): 3239154205
 [Next Sequence Number: 1 (relative sequence number)]
 Acknowledgment Number: 0
 Acknowledgment number (raw): 0
 0110 = Header Length: 24 bytes (6)
 Flags: 0x002 (SYN)
 Window: 4128
 [Calculated window size: 4128]
 Checksum: 0x7582 [unverified]
 [Checksum Status: Unverified]
 Urgent Pointer: 0
 Options: (4 bytes), Maximum segment size
 TCP Option - Maximum segment size: 512 bytes
 Kind: Maximum Segment Size (2)
 Length: 4
 MSS Value: 512
 [Timestamps]
 [Packet size limited during capture: Ethertype truncated]

Vous pouvez voir que la valeur MSS TCP du paquet SYN est 512.

Ajustement TCP MSS entraînant une lenteur lors d'un trafic TCP important

Supposons maintenant qu'un réseau comporte plusieurs périphériques utilisant le trafic TCP. Par exemple, ils peuvent transférer des fichiers ou accéder à une application basée sur TCP (telle qu'un serveur Citrix).

Vous l'avez simulé en connectant un IXIA (générateur de trafic) à C9500-2 Te1/0/37, en envoyant des paquets TCP SYN à un débit élevé.

Ce périphérique IXIA agit comme un segment de réseau, où plusieurs utilisateurs utilisent des applications basées sur TCP.

Vous avez configuré l'interface de ligne de commande `ip tcp adjust-mss` sur Te1/0/37.

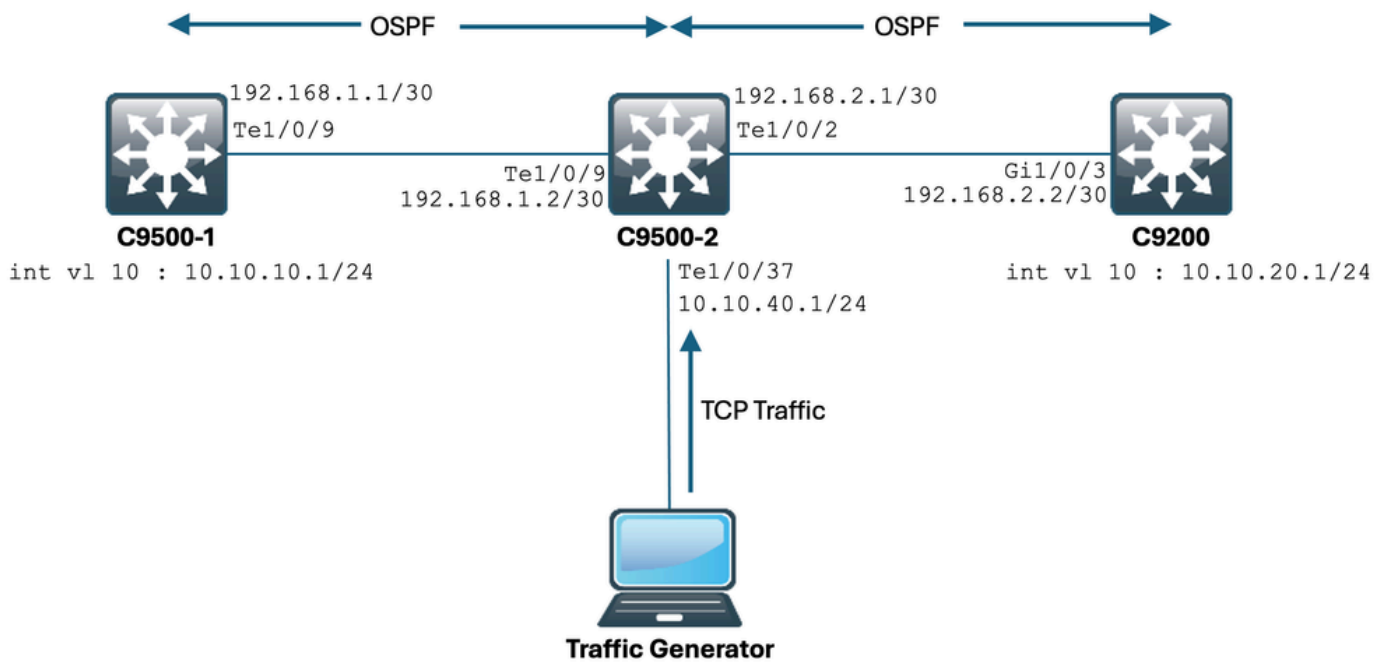
Cela entraîne la réception de tout le trafic TCP sur Te1/0/37 vers le CPU de C9500-2.

Ceci à son tour étouffe la file d'attente « Sw forwarding » du contrôleur COPP C9500-2, comme mentionné précédemment dans le document.

Par conséquent, l'établissement de la session SSH de C9500-1 à C9200 est affecté.

Soit la session SSH ne se forme pas, soit elle expire, soit elle est établie après un délai.

Voici à quoi ressemble la topologie :



Voyons voir ça en action.

Voici la configuration de C9500-2 Te1/0/37 :

```
C9500-2#sh run int te1/0/37
Building configuration...
Current configuration : 135 bytes
interface TenGigabitEthernet1/0/37
no switchport
ip address 10.10.40.1 255.255.255.0
ip tcp adjust-mss 500
load-interval 30
end
```

Vous commencez maintenant à envoyer un trafic énorme depuis IXIA vers l'interface Te1/0/37. Jetons un coup d'oeil au débit du trafic entrant :

```
C9500-2#sh int te1/0/37 | in rate
Queueing strategy: fifo
30 second input rate 6425812000 bits/sec, 12550415 packets/sec → We can see the enormous Input rate.
30 second output rate 0 bits/sec, 0 packets/sec
```

Essayons de passer de SSH de C9500-1 à C9200 maintenant :

```
C9500-1#ssh -l admin 10.10.20.1
% Connection timed out; remote host not responding
C9500-1#
```

Vous pouvez clairement voir que le C9500-1 n'a pas pu établir de connexion SSH avec le C9200. En effet, le paquet SYN TCP envoyé par le C9500-1 était abandonné par la file d'attente « Sw forwarding », qui est bombardée de trafic en provenance de Te1/0/37.

Jetons un coup d'oeil à la file d'attente :

```
C9500-2#sh platform hardware fed switch 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)
-----
0 11 DOT1X Auth Yes 1000 1000 0 0
1 1 L2 Control Yes 2000 2000 0 0
2 14 Forus traffic Yes 4000 4000 0 0
3 0 ICMP GEN Yes 600 600 0 0
4 2 Routing Control Yes 5400 5400 0 0
5 14 Forus Address resolution Yes 4000 4000 0 0
6 0 ICMP Redirect Yes 600 600 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 600 600 0 0
13 10 Openflow Yes 200 200 0 0
14 13 Sw forwarding Yes 1000 1000 39683368064 620052629 → We can see the huge number of dropped packets in t
15 8 Topology Control Yes 13000 13000 0 0
16 12 Proto Snooping Yes 2000 2000 0 0
17 6 DHCP Snooping Yes 400 400 0 0
18 13 Transit Traffic Yes 1000 1000 0 0
19 10 RPF Failed Yes 200 200 0 0
20 15 MCAST END STATION Yes 2000 2000 0 0
21 13 LOGGING Yes 1000 1000 0 0
22 7 Punt Webauth Yes 1000 1000 0 0
23 18 High Rate App Yes 13000 13000 0 0
24 10 Exception Yes 200 200 0 0
25 3 System Critical Yes 1000 1000 0 0
26 10 NFL SAMPLED DATA Yes 200 200 0 0
27 2 Low Latency Yes 5400 5400 0 0
28 10 EGR Exception Yes 200 200 0 0
29 5 Stackwise Virtual OOB Yes 8000 8000 0 0
30 9 MCAST Data Yes 400 400 0 0
31 3 Gold Pkt Yes 1000 1000 0 0
```

Recueillons les résultats plusieurs fois afin de s'assurer que le nombre de pertes augmente pendant le problème :

```
C9500-2#sh platform hardware fed switch active qos queue stats internal cpu policer | in Sw forwarding
14 13 Sw forwarding Yes 1000 1000 47046906560 735107915
14 13 21 Sw forwarding Yes
13 system-cpp-police-sw-forward : Sw forwarding/ LOGGING/ L2 LVX Data Pack/ Transit Traffic/
```

```

21 system-cpp-police-ios-feature : ICMP GEN/ BROADCAST/ ICMP Redirect/ L2 LVX Cont Pack/ Proto Snooping
C9500-2#
!
C9500-2#sh platform hardware fed switch active qos queue stats internal cpu policer | in Sw forwarding
14 13 Sw forwarding Yes 1000 1000 47335535936 739617752
14 13 21 Sw forwarding Yes
13 system-cpp-police-sw-forward : Sw forwarding/ LOGGING/ L2 LVX Data Pack/ Transit Traffic/
21 system-cpp-police-ios-feature : ICMP GEN/ BROADCAST/ ICMP Redirect/ L2 LVX Cont Pack/ Proto Snooping
C9500-2#
!
C9500-2#sh platform hardware fed switch active qos queue stats internal cpu policer | in Sw forwarding
14 13 Sw forwarding Yes 1000 1000 47666441088 744788145
14 13 21 Sw forwarding Yes
13 system-cpp-police-sw-forward : Sw forwarding/ LOGGING/ L2 LVX Data Pack/ Transit Traffic/
21 system-cpp-police-ios-feature : ICMP GEN/ BROADCAST/ ICMP Redirect/ L2 LVX Cont Pack/ Proto Snooping
C9500-2#

```

Comme vous pouvez le voir, le nombre de paquets abandonnés augmente et le trafic SSH (paquet TCP SYN) est abandonné ici.

Maintenant, si vous ne savez pas par quelle interface/interface SVI vous obtenez cet afflux de trafic, vous avez une commande spécifique pour aider.

```

C9500-2#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
=====
TenGigabitEthernet1/0/37 0x00000042 1000 1000 1000 0 0 0
-----
C9500-2#

```

La commande show platform software fed switch active punt rates interfaces nous donne la liste des interfaces qui sont responsables de la réception d'une grande quantité de trafic qui est envoyé au CPU.

Vous pouvez clairement voir Te1/0/37 ici, qui est l'interface par laquelle vous obtenez le trafic TCP.

Maintenant, si vous voulez voir la quantité de trafic qui atteint toutes les files d'attente du contrôleur COPP (qui est en cours de réception sur l'interface précédente), vous pouvez utiliser : show platform software fed switch active punt rates interfaces <IF_ID du résultat ci-dessus>

Jetons un coup d'oeil :

```

C9500-2#show platform software fed switch active punt rates interfaces 0x42
Punt Rate on Single Interfaces Statistics
Interface : TenGigabitEthernet1/0/37 [if_id: 0x42]

```


Received Dropped

Total : 2048742 Total : 0
10 sec average : 1000 10 sec average : 0
1 min average : 1000 1 min average : 0
5 min average : 1000 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 7392 0 0 0
2 CPU_Q_FORUS_TRAFFIC 0 0 0 0
3 CPU_Q_ICMP_GEN 0 0 0 0
4 CPU_Q_ROUTING_CONTROL 0 0 0 0
5 CPU_Q_FORUS_ADDR_RESOLUTION 0 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 0 0 0 0
13 CPU_Q_CONTROLLER_PUNT 0 0 0 0
14 CPU_Q_SW_FORWARDING 2006390 1000 0 0 -----> We can see high amount of traffic hitting the Sw forward
15 CPU_Q_TOPOLOGY_CONTROL 0 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 0 0 0 0
21 CPU_Q_LOGGING 34960 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
=====
```

Collecte du résultat plusieurs fois à intervalles très courts :

```
C9500-2#show platform software fed switch active punt rates interfaces 0x42 | in SW_FORWARDING
14 CPU_Q_SW_FORWARDING 2126315 1000 0 0
C9500-2#
C9500-2#show platform software fed switch active punt rates interfaces 0x42 | in SW_FORWARDING
14 CPU_Q_SW_FORWARDING 2128390 1000 0 0
C9500-2#
C9500-2#show platform software fed switch active punt rates interfaces 0x42 | in SW_FORWARDING
14 CPU_Q_SW_FORWARDING 2132295 1000 0 0
C9500-2#
```

Cela montre clairement que la file d'attente de transfert du logiciel est saturée.

Une fois que vous supprimez la `ip tcp adjust-mss` commande du Te1/0/37, ou si vous arrêtez ce trafic TCP, l'accès SSH de C9500-1 à C9200 est immédiatement rétabli.

Examinons la session SSH après l'arrêt de C9500-2 Te1/0/37 :

```
C9500-1#ssh -l admin 10.10.20.1  
Password:
```

Vous pouvez voir que l'accès SSH est à nouveau restauré.

Par conséquent, vous pouvez corréliser la lenteur TCP ici (accès SSH bloqué) en raison de la quantité élevée de trafic TCP dans le réseau, avec l'ajustement TCP MSS.

Points importants

1. Chaque fois que vous avez une lenteur TCP dans votre réseau, telle que la lenteur du transfert de fichiers, l'accessibilité aux applications TCP, etc., et que vous avez un ajustement MSS TCP configuré sur un commutateur Catalyst, assurez-vous de vérifier les abandons du contrôleur COPP afin de vérifier s'il y a une quantité élevée de trafic TCP dans le réseau ou non.
2. Si vous avez configuré l'ajustement TCP MSS sur un commutateur Catalyst, assurez-vous que le trafic TCP de votre réseau ne dépasse pas le taux de COPP Policer, sinon des problèmes liés à TCP (lenteur, pertes de paquets) sont observés dans votre réseau.

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.