

Dépannage de l'utilisation élevée du processeur Nexus 7000

Table des matières

[Introduction](#)

[Utilisation du processeur sur les plates-formes Nexus 7000](#)

[Commandes et scripts pour surveiller les processus et les processeurs](#)

[Commandes](#)

[Commande show processes](#)

[Commande show system resources](#)

[commande show processes cpu](#)

[Commande show processes cpu history](#)

[show process cpu detail, commande](#)

Introduction

Ce document décrit les processus de surveillance de l'utilisation du CPU et de dépannage des problèmes d'utilisation élevée du CPU sur les plates-formes de la gamme Cisco Nexus 7000.

Utilisation du processeur sur les plates-formes Nexus 7000

La plate-forme Nexus 7000 est un système Linux doté d'un planificateur préemptif qui permet un accès équitable aux ressources du processeur pour tous les processus.

Contrairement à la gamme Cisco Catalyst 6500, il n'y a pas de processeur de routage (RP) et de processeur de commutation (SP) séparés.

- Le Supervisor Engine 1 est équipé d'un processeur double coeur.
- Le Supervisor Engine 2 est équipé d'un processeur à quatre coeurs.
- Le Supervisor Engine 2E est équipé de deux processeurs à quatre coeurs.

Le système d'exploitation Cisco NX-OS tire parti du multitâche préemptif du processeur, de sorte que les processus peuvent tirer parti d'un processeur inactif afin d'effectuer des tâches plus rapidement.

Par conséquent, l'option d'historique signale les pics de CPU possibles qui n'indiquent pas nécessairement un problème. Cependant, si l'utilisation moyenne du processeur reste élevée par rapport à l'utilisation normale du processeur de base pour un réseau particulier, recherchez une utilisation élevée du processeur.

Les limiteurs de débit matériel par défaut (HWRL) et la réglementation du plan de contrôle par défaut (CoPP) sont activés pour protéger l'interface intrabande du superviseur sur les plates-formes Nexus 7000.

Les commandes et l'exemple de script EEM sont basés sur Nexus 7000 version 6.1 et antérieure et peuvent être modifiés dans les versions futures.

Commandes et scripts pour surveiller les processus et les processeurs

Commandes

Certaines commandes d'affichage (« show ») sont offertes par l'outil « Cisco CLI Analyzer » réservé aux clients inscrits. Utilisez cet outil pour obtenir une analyse des rapports produits par ces commandes.

Commande show processes

Utilisez cette commande afin d'afficher des informations sur les processus actifs.

```
switch# show processes
```

```

PID      State  PC          Start_cnt  TTY  Type  Process
-----  -----  -----  -
1        S      41520eb8    1          -    0     init
2        S      0           1          -    0     kthreadd
3        S      0           1          -    0     migration/0
4        S      0           1          -    0     ksoftirqd/0
5        S      0           1          -    0     watchdog/0
6        S      0           1          -    0     migration/1
7        S      0           1          -    0     ksoftirqd/1
8        S      0           1          -    0     watchdog/1
9        S      0           1          -    0     events/0
10       S      0           1          -    0     events/1
11       S      0           1          -    0     khelper
12       S      0           1          -    0     kblockd/0

```

Champ	Description
PID	ID du processus
Province	État du processus
PC	Compteur de programme actuel au format hexadécimal
Cnt_départ	Nombre de fois qu'un processus a été démarré ou redémarré
TTY	Terminal qui contrôle le processus. Un tiret (—) signifie généralement qu'un démon ne s'exécute pas sur un terminal particulier.
Process	Nom du processus

État du processus	Description
D	Veille sans coupure (généralement E/S)
R	Exécutable (dans la file d'attente d'exécution)
S	Sommeil
T	Suivi ou arrêt

Z	Défaut (zombie) processus
NR	Ne fonctionne pas
ER	Exécution attendue, mais pas en cours d'exécution

Commande show system resources

Utilisez cette commande afin d'afficher les statistiques relatives au CPU et à la mémoire liées au système.

```
switch#show system resources
Load average: 1 minute: 0.36 5 minutes: 0.39 15 minutes: 0.44
Processes : 1068 total, 1 running
CPU states : 0.5% user, 5.5% kernel, 94.0% idle
Memory usage: 8245436K total, 3289920K used, 4955516K free
Current memory status: OK
```

Champ	Description
Charge	Nombre de processus en cours d'exécution. La moyenne reflète la charge du système au cours des 1, 5 et 15 dernières minutes.
Processus	Nombre de processus dans le système et nombre de processus réellement en cours d'exécution lorsque la commande est exécutée.
état du processeur	Pourcentage d'utilisation du processeur en mode utilisateur, en mode noyau et en temps d'inactivité au cours de la dernière seconde. Dans le cas d'un superviseur double coeur, le processeur est moyenné sur les deux coeurs.
Utilisation de la mémoire	Mémoire totale, mémoire utilisée, mémoire libre, mémoire utilisée pour les tampons et mémoire utilisée pour le cache en kilo-octets. Les mémoires tampons et le cache sont inclus dans les statistiques de la mémoire utilisée.

commande show processes cpu

Utilisez cette commande afin d'afficher l'utilisation du CPU au niveau du processus :

```
switch#show processes cpu | ex 0.0
```

```
PID Runtime(ms) Invoked uSecs 1Sec Process
-----
26 66399 269718 246 0.9% kide/1
2908 115550 11310 10216 2.9% platform
3223 7248 9208 787 0.9% R2D2_usd
```

```
CPU util : 1.0% user, 3.0% kernel, 96.0% idle
Please note that only processes from the requested vdc are shown above
```

Champ	Description
Durée d'exécution (ms)	Temps processeur utilisé par le processus en millisecondes

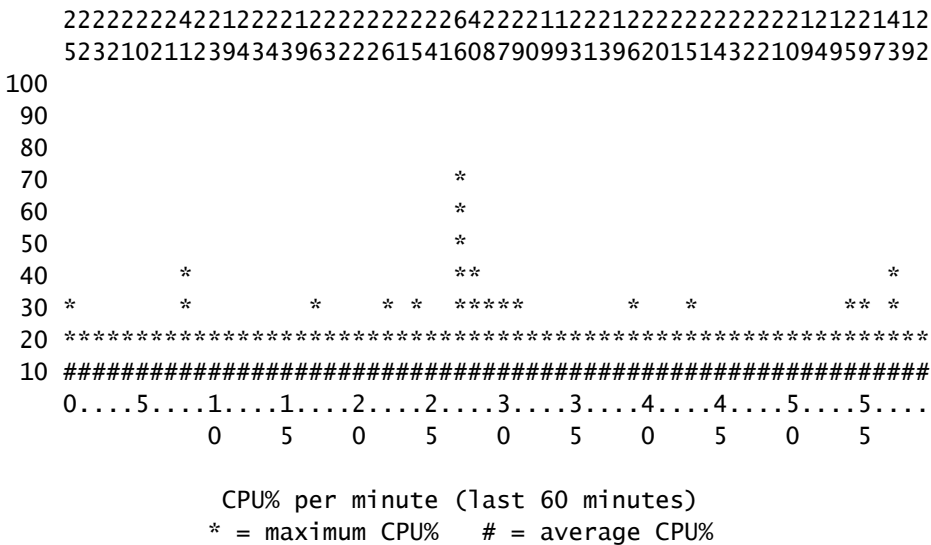
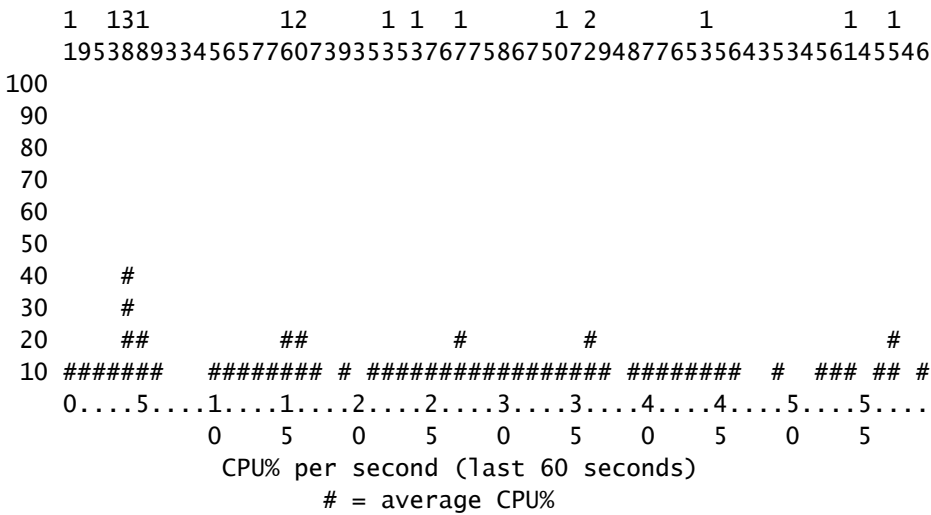
Invoked	Nombre de fois où le processus a été appelé
uSecs	Temps processeur moyen pour chaque appel de processus en microsecondes
1 Sec	Pourcentage d'utilisation du processeur pendant la dernière seconde

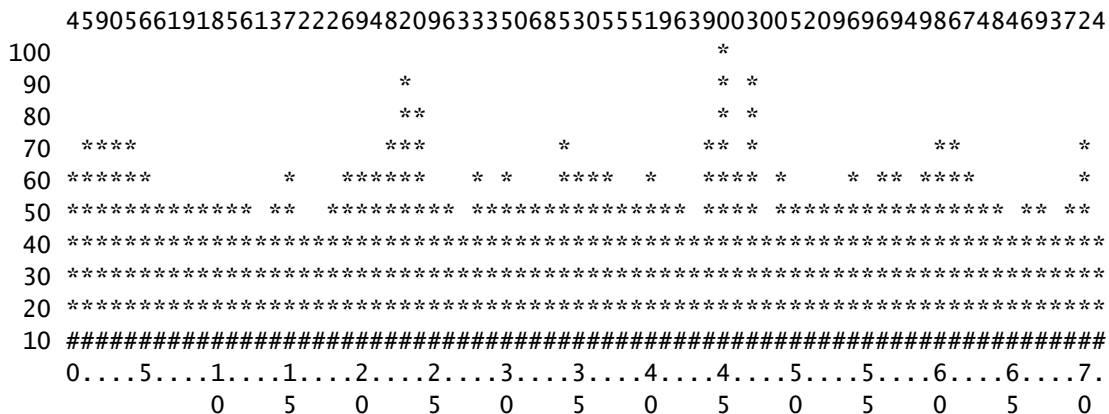
Pour connaître l'utilisation du CPU pour tous les threads qui appartiennent à un ID de processus (PID) spécifique, utilisez la commande `show process cpu detail <pid>` disponible dans NX-OS version 6.2x.

Commande `show processes cpu history`

Utilisez cette commande afin d'afficher l'utilisation du CPU pendant les 60 dernières secondes, 60 minutes et 72 heures. Vérifiez l'utilisation moyenne du processeur (#) et les pics (*).

```
switch# show processes cpu history
```





CPU% per hour (last 72 hours)
 * = maximum CPU% # = average CPU%

Commande show process cpu detail <pid>

Cette commande, qui a été ajoutée dans la version 6.2, affiche les informations d'utilisation du CPU pour tous les threads qui appartiennent à un PID spécifique.

```

switch# show processes cpu sorted | grep cli
3965      23734      17872      1328      0.0%      0.1%      0.7%      -      cli
4024      3047      1256      2426      0.0%      0.0%      0.0%      -      diagclient
4094      787      258      3052      0.0%      0.0%      0.0%      -      cardclient
4728      227      209      1088      0.0%      0.0%      0.0%      -      port_client
4729      1351      499      2708      0.0%      0.0%      0.0%      -      statsclient
4730      2765      550      5028      0.0%      0.0%      0.0%      -      xbar_client
  
```

```

switch# show processes cpu sorted | grep clis
3965      23734      17872      1328      0.0%      0.1%      0.7%      -      cli
switch# show process cpu detailed 3965
  
```

CPU utilization for five seconds: 3%/3%; one minute: 0%; five minutes: 1%

PID	Runtime(ms)	Invoked	uSecs	5Sec	1Min	5Min	TTY	Process
3965	23734	17873	1327	0.0%	0.1%	0.6%	-	cli
4227	45	334	135	0.0%	0.0%	0.0%	-	cli:cli-cli-t
4228	24	153	162	0.0%	0.0%	0.0%	-	cli:cli-nvdb-
4760	75	224	335	0.0%	0.0%	0.0%	-	cli:cli-seria


```

switch# show processes cpu sorted | grep netstack
4133      353      892      395      0.0%      0.0%      0.0%      -      netstack
switch# show process cpu detailed 4133
  
```

CPU utilization for five seconds: 5%/5%; one minute: 1%; five minutes: 1%

PID	Runtime(ms)	Invoked	uSecs	5Sec	1Min	5Min	TTY	Process
4133	353	892	395	0.0%	0.0%	0.0%	-	netstack
4145	322	6492	49	0.0%	0.0%	0.0%	-	netstack:active
4151	239	247	971	0.0%	0.0%	0.0%	-	netstack:ip-sys
4153	0	3	162	0.0%	0.0%	0.0%	-	netstack:mplsda
4155	2	3	717	0.0%	0.0%	0.0%	-	netstack:mplsct
4163	0	2	240	0.0%	0.0%	0.0%	-	netstack:ipv6-d
4164	97	957	101	0.0%	0.0%	0.0%	-	netstack:netsta

4166	15	628	25	0.0%	0.0%	0.0%	-	netstack:ip-sys
4167	0	3	224	0.0%	0.0%	0.0%	-	netstack:ip-pm-
4170	1	12	154	0.0%	0.0%	0.0%	-	netstack:ip-uri
4171	9	30	323	0.0%	0.0%	0.0%	-	netstack:ip-ipc
4173	0	5	167	0.0%	0.0%	0.0%	-	netstack:ip-ipc
4175	0	2	305	0.0%	0.0%	0.0%	-	netstack:ip-ret
4176	12	7	1838	0.0%	0.0%	0.0%	-	netstack:ip-ppf
4178	4	15	289	0.0%	0.0%	0.0%	-	netstack:ipv6-c
4179	41	445	93	0.0%	0.0%	0.0%	-	netstack:disp
4180	0	6	98	0.0%	0.0%	0.0%	-	netstack:worker
4181	33	501	66	0.0%	0.0%	0.0%	-	netstack:worker
4182	0	2	232	0.0%	0.0%	0.0%	-	netstack:worker
4183	0	2	227	0.0%	0.0%	0.0%	-	netstack:worker
4184	0	3	152	0.0%	0.0%	0.0%	-	netstack:worker
4185	0	2	278	0.0%	0.0%	0.0%	-	netstack:worker
4186	0	2	254	0.0%	0.0%	0.0%	-	netstack:worker
4187	0	3	168	0.0%	0.0%	0.0%	-	netstack:worker
4188	0	2	266	0.0%	0.0%	0.0%	-	netstack:worker
4189	0	2	248	0.0%	0.0%	0.0%	-	netstack:worker
4190	0	2	254	0.0%	0.0%	0.0%	-	netstack:worker
4191	0	3	201	0.0%	0.0%	0.0%	-	netstack:worker
4192	0	2	258	0.0%	0.0%	0.0%	-	netstack:worker
4193	0	7	111	0.0%	0.0%	0.0%	-	netstack:worker
4194	0	8	78	0.0%	0.0%	0.0%	-	netstack:worker
4195	0	2	313	0.0%	0.0%	0.0%	-	netstack:worker
4196	15	632	23	0.0%	0.0%	0.0%	-	netstack:ptacti
4197	0	5	120	0.0%	0.0%	0.0%	-	netstack:tcp_ip
4198	4	11	390	0.0%	0.0%	0.0%	-	netstack:ipv6-m
4199	0	3	240	0.0%	0.0%	0.0%	-	netstack:ipv6-c
4200	0	1	561	0.0%	0.0%	0.0%	-	netstack:ipv6-c
4201	0	3	246	0.0%	0.0%	0.0%	-	netstack:icmpv6
4513	0	5	112	0.0%	0.0%	0.0%	-	netstack:ipv6-m
4514	0	2	291	0.0%	0.0%	0.0%	-	netstack:ipv6-m

 Remarque : toutes les informations de processus sont basées sur le processus dans NX-OS. Dans NX-OS, tous les threads partagent la mémoire allouée par un autre thread, il n'est donc pas possible d'afficher les informations par thread.

show system internal processes cpu, commande

Cette commande est équivalente à la commande top dans Linux, qui fournit un aperçu continu de l'activité du processeur en temps réel.

```
switch# show system internal processes cpu
```

```
top - 23:51:41 up 51 min, 3 users, load average: 0.56, 0.49, 0.46
Tasks: 433 total, 1 running, 431 sleeping, 0 stopped, 1 zombie
Cpu(s): 5.9%us, 7.8%sy, 0.0%ni, 81.9%id, 3.6%wa, 0.1%hi, 0.6%si, 0.0%st
Mem: 8245436k total, 3531776k used, 4713660k free, 5360k buffers
Swap: 0k total, 0k used, 0k free, 1458188k cached
```

```
PID USER PR NI VIRT RES SHR S %CPU %MEM TIME+ COMMAND
3589 svc-isan 25 5 112m 8864 4572 S 5.7 0.1 0:21.60 stats_client
10881 sjl'an 20 0 3732 1648 1140 R 3.8 0.0 0:00.04 top
```

```

26 root 20 0 0 0 0 S 1.9 0.0 1:07.07 kide/1
3280 root -2 0 101m 6104 3680 S 1.9 0.1 0:32.57 octopus
3570 root 20 0 123m 19m 6456 S 1.9 0.2 0:06.07 diag_port_lb
5151 root 20 0 205m 45m 9.8m S 1.9 0.6 0:02.61 netstack
1 root 20 0 1988 604 524 S 0.0 0.0 0:03.75 init
2 root 15 -5 0 0 0 S 0.0 0.0 0:00.00 kthreadd
3 root RT -5 0 0 0 S 0.0 0.0 0:00.00 migration/0
4 root 15 -5 0 0 0 S 0.0 0.0 0:00.61 ksoftirqd/0
5 root -2 -5 0 0 0 S 0.0 0.0 0:00.06 watchdog/0
6 root RT -5 0 0 0 S 0.0 0.0 0:00.00 migration/1
7 root 15 -5 0 0 0 S 0.0 0.0 0:04.80 ksoftirqd/1

```

Champ	Description
PID	ID du processus
UTILISATEUR	Nom de l'utilisateur propriétaire du processus
RP	Priorité attribuée au processus
NI	La valeur du processus
VIRTUALITÉ	Quantité de mémoire virtuelle utilisée par le processus
RÉS	Quantité de mémoire vive physique utilisée par le processus (sa taille résidente) en kilo-octets
RSH	Quantité de mémoire partagée utilisée par le processus
S	<p>État du processus. Les valeurs possibles sont les suivantes :</p> <ul style="list-style-type: none"> • D - Veille sans interruption • R - En cours • S - Veille • T - Suivi ou arrêt • Z - Zombie
%CPU	Pourcentage de temps processeur utilisé par le processus
%MEM	Pourcentage de mémoire vive physique disponible utilisé par le processus
TEMPS+	Temps processeur total consommé par le processus depuis son démarrage
COMMANDE	Nom de la commande qui a été entrée pour démarrer le processus

Le {#seconds} | no-more permet d'exécuter automatiquement la commande à chaque #seconds jusqu'à ce qu'un Ctrl-C soit entré. Voici un exemple de sortie:

<#root>

```
switch# show system internal processes cpu
```

```
5 | no-more
```

```

top - 17:31:12 up 4 days, 18:31, 3 users, load average: 0.52, 0.40, 0.32
Tasks: 449 total, 3 running, 446 sleeping, 0 stopped, 0 zombie
Cpu(s): 3.5%us, 4.5%sy, 0.0%ni, 91.2%id, 0.1%wa, 0.1%hi, 0.5%si, 0.0%st
Mem: 8245436k total, 4192740k used, 4052696k free, 27644k buffers
Swap: 0k total, 0k used, 0k free, 1919612k cached
  PID USER      PR  NI  VIRT  RES  SHR  S %CPU %MEM    TIME+  COMMAND

```

```

2908 root      20    0  112m 8516 5516 S  7.5  0.1 264:44.25 pfm
31487 sjlan      20    0  3732 1652 1140 R  5.6  0.0   0:00.05 top
3059 svc-isan  20    0 80288 7536 4440 S  3.8  0.1  65:44.59 diagmgr
3192 root      20    0  334m 47m  11m S  1.9  0.6  25:36.52 netstack
3578 svc-isan  20    0  118m 13m 6952 S  1.9  0.2  24:57.36 stp
5119 svc-isan  20    0  139m 14m 7028 S  1.9  0.2   3:48.60 urib
5151 root      20    0  209m 46m  11m S  1.9  0.6  38:53.39 netstack
5402 svc-isan  20    0  117m 15m 9140 S  1.9  0.2  36:07.13 stp
6175 svc-isan  20    0  118m 16m 9580 S  1.9  0.2  47:09.41 stp
  1 root      20    0  1988  604  524 S  0.0  0.0   0:06.51 init
  2 root      15   -5    0    0    0 S  0.0  0.0   0:00.00 kthreadd
  3 root      RT   -5    0    0    0 S  0.0  0.0   0:00.08 migration/0
  4 root      15   -5    0    0    0 S  0.0  0.0   1:07.77 ksoftirqd/0

```

```

top - 17:31:18 up 4 days, 18:31,  3 users,  load average: 0.48, 0.39, 0.32
Tasks: 449 total,  1 running, 448 sleeping,  0 stopped,  0 zombie
Cpu(s):  3.5%us,  4.5%sy,  0.0%ni, 91.2%id,  0.1%wa,  0.1%hi,  0.5%si,  0.0%st
Mem:   8245436k total, 4192592k used, 4052844k free,  27644k buffers
Swap:      0k total,      0k used,      0k free, 1919612k cached

```

```

  PID USER      PR  NI  VIRT  RES  SHR S %CPU %MEM    TIME+  COMMAND
 2908 root      20   0  112m 8516 5516 S  7.5  0.1 264:44.47 pfm
31490 sjlan      20   0  3732 1656 1140 R  3.8  0.0   0:00.04 top
   1 root      20   0  1988  604  524 S  0.0  0.0   0:06.51 init
   2 root      15  -5    0    0    0 S  0.0  0.0   0:00.00 kthreadd
   3 root      RT  -5    0    0    0 S  0.0  0.0   0:00.08 migration/0
   4 root      15  -5    0    0    0 S  0.0  0.0   1:07.77 ksoftirqd/0
   5 root      -2  -5    0    0    0 S  0.0  0.0   0:13.74 watchdog/0
   6 root      RT  -5    0    0    0 S  0.0  0.0   0:00.10 migration/1
   7 root      15  -5    0    0    0 S  0.0  0.0   0:54.47 ksoftirqd/1
   8 root      -2  -5    0    0    0 S  0.0  0.0   0:00.20 watchdog/1
   9 root      15  -5    0    0    0 S  0.0  0.0   0:02.94 events/0
  10 root      15  -5    0    0    0 S  0.0  0.0   0:02.58 events/1
  11 root      15  -5    0    0    0 S  0.0  0.0   0:00.00 khelper

```

```

top - 17:31:23 up 4 days, 18:31,  3 users,  load average: 0.44, 0.39, 0.32
Tasks: 449 total,  1 running, 448 sleeping,  0 stopped,  0 zombie
Cpu(s):  3.5%us,  4.5%sy,  0.0%ni, 91.2%id,  0.1%wa,  0.1%hi,  0.5%si,  0.0%st
Mem:   8245436k total, 4192584k used, 4052852k free,  27644k buffers
Swap:      0k total,      0k used,      0k free, 1919612k cached

```

```

  PID USER      PR  NI  VIRT  RES  SHR S %CPU %MEM    TIME+  COMMAND
31493 sjlan      20   0  3732 1656 1140 R  3.8  0.0   0:00.04 top
 5004 svc-isan  20   0  118m 13m 6852 S  1.9  0.2  41:35.81 stp
10337 svc-isan  20   0  133m 11m 7948 S  1.9  0.1   1:42.81 mcecm
   1 root      20   0  1988  604  524 S  0.0  0.0   0:06.51 init
   2 root      15  -5    0    0    0 S  0.0  0.0   0:00.00 kthreadd
   3 root      RT  -5    0    0    0 S  0.0  0.0   0:00.08 migration/0
   4 root      15  -5    0    0    0 S  0.0  0.0   1:07.77 ksoftirqd/0
   5 root      -2  -5    0    0    0 S  0.0  0.0   0:13.74 watchdog/0
   6 root      RT  -5    0    0    0 S  0.0  0.0   0:00.10 migration/1
   7 root      15  -5    0    0    0 S  0.0  0.0   0:54.47 ksoftirqd/1
   8 root      -2  -5    0    0    0 S  0.0  0.0   0:00.20 watchdog/1
   9 root      15  -5    0    0    0 S  0.0  0.0   0:02.94 events/0
  10 root      15  -5    0    0    0 S  0.0  0.0   0:02.58 events/1

```

```

top - 17:31:29 up 4 days, 18:31,  3 users,  load average: 0.41, 0.38, 0.32
Tasks: 449 total,  1 running, 448 sleeping,  0 stopped,  0 zombie
Cpu(s):  3.5%us,  4.5%sy,  0.0%ni, 91.2%id,  0.1%wa,  0.1%hi,  0.5%si,  0.0%st
Mem:   8245436k total, 4192708k used, 4052728k free,  27644k buffers
Swap:      0k total,      0k used,      0k free, 1919616k cached

```


show system internal sysmgr service pid <pid>, commande

Utilisez cette commande afin d'afficher des détails supplémentaires, tels que l'heure de redémarrage, l'état de panne et l'état actuel, sur le processus/service par PID.


```
switch# show system internal processes cpu
top - 17:37:26 up 4 days, 18:37, 3 users, load average: 0.16, 0.35, 0.33
Tasks: 450 total, 2 running, 448 sleeping, 0 stopped, 0 zombie
Cpu(s): 3.5%us, 4.5%sy, 0.0%ni, 91.2%id, 0.1%wa, 0.1%hi, 0.5%si, 0.0%st
Mem: 8245436k total, 4193248k used, 4052188k free, 27668k buffers
Swap: 0k total, 0k used, 0k free, 1919664k cached
  PID USER      PR  NI  VIRT  RES  SHR  S  %CPU  %MEM   TIME+  COMMAND
 2908 root        20   0  112m 8516 5516 S   7.5  0.1 264:58.67 pfm
31710 sjlan       20   0  3732 1656 1140 R   3.8  0.0  0:00.04 top
 3192 root        20   0  334m 47m  11m S   1.9  0.6 25:38.39 netstack
 3578 svc-isan   20   0  118m 13m 6952 S   1.9  0.2 24:59.08 stp
 5151 root        20   0  209m 46m  11m S   1.9  0.6 38:55.52 netstack
 5402 svc-isan   20   0  117m 15m 9140 S   1.9  0.2 36:09.08 stp
 5751 root        20   0  209m 46m  10m S   1.9  0.6 41:20.58 netstack
 6098 svc-isan   20   0  151m 15m 6188 S   1.9  0.2  3:58.40 mrrib
 6175 svc-isan   20   0  118m 16m 9580 S   1.9  0.2 47:12.00 stp
   1 root        20   0  1988 604  524 S   0.0  0.0  0:06.52 init
   2 root        15  -5     0   0   0 S   0.0  0.0  0:00.00 kthreadd
   3 root        RT  -5     0   0   0 S   0.0  0.0  0:00.08 migration/0
   4 root        15  -5     0   0   0 S   0.0  0.0  1:07.83 ksoftirqd/0
```

```
switch# show system internal sysmgr service pid 2908
Service "Platform Manager" ("platform", 5):
  UUID = 0x18, PID = 2908, SAP = 39
  State: SRV_STATE_HANDSHAKED (entered at time Mon Oct 15 23:03:45 2012).
  Restart count: 1
  Time of last restart: Mon Oct 15 23:03:44 2012.
  The service never crashed since the last reboot.
  Tag = N/A
  Plugin ID: 0
```

Exemple de script EEM

Voici un exemple de script qui capture l'utilisation élevée intermittente du CPU. Les valeurs utilisées ainsi que les commandes émises peuvent être modifiées en fonction des besoins :

```
event manager applet HIGH-CPU
 event snmp oid 1.3.6.1.4.1.9.9.109.1.1.1.6.1 get-type exact entry-op ge
   entry-val 80 exit-val 30 poll-interval 5
 action 1.0 syslog msg High CPU hit $_event_pub_time
 action 2.0 cli enable
 action 3.0 cli show clock >> bootflash:high-cpu.txt
 action 4.0 cli show processes cpu sort >> bootflash:high-cpu.txt
```

 Remarque : il est nécessaire de définir exit-val. À mesure que le script collecte des données, il augmente l'utilisation du processeur. Une valeur pour exit-val garantit que le script ne s'exécute pas dans une boucle sans fin.

Utilisation CPU élevée due au processus ou au trafic

Il n'y a pas de processus ou d'interruption de l'utilisation du processeur (comme sur les plateformes logicielles Cisco IOS®) lorsque l'utilisation du processeur est surveillée. Une façon rapide de déterminer la cause de l'utilisation élevée du CPU est d'utiliser la commande `show system internal processes cpu`. En général, une utilisation CPU élevée déclenchée par le trafic entraînerait l'exécution de Netstack, ainsi que d'autres fonctionnalités et processus tels que le protocole ARP (Address Resolution Protocol) et le protocole IGMP (Internet Group Management Protocol), à un niveau élevé.

Le processus entraîne une utilisation CPU élevée

En fonction des processus et des problèmes qui entraînent une utilisation élevée du CPU, il est possible que vous deviez capturer des commandes spécifiques. Ces sections décrivent des méthodes utiles.


`show system internal <feature> mem-stats/memstats` | sous le commandement suprême

Utilisez cette commande afin d'afficher l'allocation de mémoire pour un processus ; utilisez l'option 'in Grand' pour surveiller la mémoire totale. Une fuite de mémoire peut entraîner un mauvais comportement du processus, ce qui peut entraîner une utilisation élevée du processeur.

Ethanalysateur

Utilisez Ethalyzer pour surveiller le trafic vers le processeur.

Commandes de débogage

 Remarque : Consulter les renseignements importants sur les commandes de débogage avant d'utiliser les commandes de débogage. Utilisez les commandes debug à bon escient sur un commutateur de production pour éviter toute interruption de service.

Utilisez la commande `debug logfile` chaque fois que possible pour diriger la sortie vers un fichier spécifié et pour éviter de verrouiller la session pour remplir le syslog. Voici un exemple de débogage du protocole SNMP (Simple Network Management Protocol) :

```
switch# debug logfile snmpdebug
switch# debug snmp all
switch# show debug logfile snmpdebug
2012 Oct 17 23:53:25.905914 snmpd: SDWRAP message Successfully processed
```

```
2012 Oct 17 23:53:25.906162 snmpd: Src: 0x00000501/23852 Dst: 0x00000501/28 ID
: 0x006E3C9B Size: 276 [REQ] Opc: 182 (MTS_OPC_DEBUG_WRAP_MSG) RR: 0x006E3C9B
HA_SEQNO: 0x00000000 TS: 0x10ADFFA1666FC REJ:0 SYNC:0 OPTIONS:0x0
2012 Oct 17 23:53:25.906208 snmpd: 01 00 00 00 E7 03 00 00 00 00 00 00 00 00 00
2012 Oct 17 23:53:25.906225 snmpd: 00 00 00 00 00 00 00 00 00 00 00 00 00 00 00
2012 Oct 17 23:53:25.906239 snmpd: 00 00 00 00 00 00 00 00 00 00 00 00 00 00 00
2012 Oct 17 23:53:25.906255 snmpd: FF FF FF FF 2F 64 65 76 2F 70 74 73 2F 30 00 00
2012 Oct 17 23:53:25.906271 snmpd: 00 00 00 00 00 00 00 00 00 00 00 00 00 00 00
```

```
switch# show log last 10
```

```
2012 Oct 17 17:51:06 SITE1-AGG1 %ETHPORT-5-IF_TX_FLOW_CONTROL: Interface
Ethernet10/10, operational Transmit Flow Control state changed to off
2012 Oct 17 17:51:09 SITE1-AGG1 %ETH_PORT_CHANNEL-5-PORT_SUSPENDED:
Ethernet10/10: Ethernet10/10 is suspended
2012 Oct 17 17:51:51 SITE1-AGG1 last message repeated 1 time
2012 Oct 17 17:51:51 SITE1-AGG1 %ETHPORT-5-IF_DOWN_LINK_FAILURE:
Interface Ethernet10/10 is down (Link failure)
2012 Oct 17 17:51:52 SITE1-AGG1 %ETHPORT-5-SPEED: Interface Ethernet10/10,
operational speed changed to 10 Gbps
2012 Oct 17 17:51:52 SITE1-AGG1 %ETHPORT-5-IF_DUPLEX: Interface
Ethernet10/10, operational duplex mode changed to Full
2012 Oct 17 17:51:52 SITE1-AGG1 %ETHPORT-5-IF_RX_FLOW_CONTROL: Interface
Ethernet10/10, operational Receive Flow Control state changed to off
2012 Oct 17 17:51:52 SITE1-AGG1 %ETHPORT-5-IF_TX_FLOW_CONTROL: Interface
Ethernet10/10, operational Transmit Flow Control state changed to off
2012 Oct 17 17:51:55 SITE1-AGG1 %ETH_PORT_CHANNEL-5-PORT_UP: port-channel11:
Ethernet10/10 is up
2012 Oct 17 17:51:56 SITE1-AGG1 %ETHPORT-5-IF_UP: Interface Ethernet10/10
is up in mode trunk
```

Utilisez la commande debug-filter si possible afin de minimiser la sortie sur un système de production. Par exemple, une perte de paquets provoque un écho vide UDLD (Unidirectional Link Detection) :

```
switch# debug logfile test size 1000000
switch# debug-filter pktmgr direction inbound
switch# debug-filter pktmgr dest-mac 0100.0ccc.cccc
switch# debug pktmgr client uuid 376
switch# debug pktmgr frame
switch# debug pktmgr pkt-errors
```

```
switch# debug-filter ?
fabricpath Debug fabricpath events
ip IP events
ipv6 IPv6 events
l2pt L2 Protocol Tunneling events
mpls MPLS events
pktmgr Pm debug-filter
routing Routing events
```

Le trafic entraîne une utilisation CPU élevée

Utilisez ces outils lorsque le trafic entraîne une utilisation CPU élevée :

- Ethalyzer : surveille le type de trafic en provenance ou à destination du processeur.
- Configuration - Vérifiez la configuration du commutateur/de l'interface/de la fonction
- CoPP/Limiteur de débit matériel - Assurez-vous que CoPP et HWRL sont configurés correctement. Parfois, le processeur ne fonctionne pas à un niveau élevé parce qu'il est protégé par le protocole CoPP et des limiteurs de débit. Vérifiez CoPP et HWRL pour voir s'il y a des abandons pour certains trafics/paquets.



Remarque : CoPP et HWRL sont disponibles uniquement à partir du contexte de périphérique virtuel par défaut (VDC). Ils sont appliqués par chaque module d'E/S individuel. Le trafic agrégé provenant de plusieurs modules peut encore surcharger lourdement le processeur.

Analyse de la cause première de l'utilisation élevée du CPU

Une panne de réseau peut être résolue par l'intervention de l'utilisateur, ou elle peut être restaurée par elle-même. Si vous pensez qu'une utilisation CPU élevée a provoqué une panne réseau, suivez ces instructions afin d'en rechercher les causes.

Symptômes

Les symptômes d'une utilisation élevée du CPU incluent l'instabilité du plan de contrôle, les problèmes de connectivité du plan de données causés par une défaillance du plan de contrôle, le battement du protocole tel que le battement HSRP/RP (Hot Standby Router Protocol), la désactivation des erreurs UDLD, la défaillance du protocole STP (Spanning Tree Protocol) et d'autres problèmes de connectivité.

Historique du processeur

Commande `show processes cpu history`

Si le commutateur n'a pas été rechargé ou commuté, exécutez la commande `show processes cpu history` dans les 72 heures suivant la panne afin de voir si une utilisation élevée du CPU s'est produite au moment de l'événement.

CoPP et HWRL

Si l'utilisation élevée du CPU était la cause principale d'une panne passée, et si vous suspectez que la panne a été déclenchée par le trafic réseau, vous pouvez utiliser CoPP et HWRL (limiteur de débit matériel) afin d'aider à identifier le type de trafic.

Commande `show policy-map interface control-plane`

Voici un exemple de sortie de la commande `show policy-map interface control-plane` :

```
switch# show policy-map interface control-plane
Control Plane
```

```
service-policy input: copp-system-p-policy-strict
```

```
class-map copp-system-p-class-critical (match-any)
  match access-group name copp-system-p-acl-bgp
  match access-group name copp-system-p-acl-bgp6
  match access-group name copp-system-p-acl-igmp
  match access-group name copp-system-p-acl-msdp
  match access-group name copp-system-p-acl-ospf

  match access-group name copp-system-p-acl-pim
  match access-group name copp-system-p-acl-pim6
  match access-group name copp-system-p-acl-rip
  match access-group name copp-system-p-acl-rip6
  match access-group name copp-system-p-acl-vpc
  match access-group name copp-system-p-acl-eigrp
  match access-group name copp-system-p-acl-eigrp6
  match access-group name copp-system-p-acl-mac-l2pt
  match access-group name copp-system-p-acl-mps-ldp
  match access-group name copp-system-p-acl-mps-oam
  match access-group name copp-system-p-acl-ospf6
  match access-group name copp-system-p-acl-otv-as
  match access-group name copp-system-p-acl-mac-otv-isis
  match access-group name copp-system-p-acl-mps-rsvp
  match access-group name copp-system-p-acl-mac-fabricpath-isis
  match protocol mpls router-alert
  match protocol mpls exp 6
  set cos 7
  police cir 39600 kbps , bc 250 ms
  module 1 :
    conformed 1108497274 bytes; action: transmit
    violated 0 bytes; action: drop

  module 3 :
    conformed 0 bytes; action: transmit
    violated 0 bytes; action: drop

  module 10 :
    conformed 0 bytes; action: transmit
  .
  .
  .
```

Commande show hardware rate-limiter mod <x>

Voici un exemple de sortie de la commande show hardware rate-limiter mod 1 antérieure à NX-OS version 6.1 :

```
switch# show hardware rate-limiter mod 1
```

```
Units for Config: packets per second
Allowed, Dropped & Total: aggregated since last clear counters
```

Rate Limiter Class	Parameters
layer-3 mtu	Config : 500 Allowed : 0 Dropped : 0 Total : 0
layer-3 ttl	Config : 500 Allowed : 0 Dropped : 0 Total : 0
layer-3 control	Config : 10000 Allowed : 0 Dropped : 0
.	
.	
.	

Voici un exemple de sortie de la commande show hardware rate-limiteur mod 1 dans NX-OS version 6.1 ou ultérieure :

```
switch# show hardware rate-limiter mod 1
switch# show hardware rate-limiter module 1
```

Units for Config: packets per second
Allowed, Dropped & Total: aggregated since last clear counters

Module: 1

R-L Class	Config	Allowed	Dropped	Total
L3 mtu	500	0	0	0
L3 ttl	500	0	0	0
L3 control	10000	0	0	0
L3 glean	100	0	0	0
L3 mcast dirconn	3000	0	0	0
L3 mcast loc-grp	3000	0	0	0
L3 mcast rpf-leak	500	0	0	0
L2 storm-ctrl	Disable			
access-list-log	100	0	0	0
copy	30000	0	0	0
receive	30000	40583	0	40583
L2 port-sec	500	20435006	0	20435006
L2 mcast-snoop	10000	0	0	0
L2 vpc-low	4000	0	0	0
L2 l2pt	500	0	0	0
f1 r1-1	4500		0	
f1 r1-2	1000		0	
f1 r1-3	1000		0	
f1 r1-4	100		0	
f1 r1-5	1500		0	
L2 vpc-peer-gw	5000	0	0	0
L2 lisp-map-cache	5000	0	0	0

Recherchez n'importe quelle classe dont le nombre d'abandons est incrémenté. Déterminez s'il

est normal qu'une classe dépasse le seuil configuré.

Pilote Inband

show hardware internal cpu-mac inband [counters | statistiques | events] Commande

Utilisez cette commande afin de vérifier les abandons dans le chemin du processeur, le contrôle de flux XOFF, les débits maximum de réception et de transmission du processeur, etc.

```
switch# show hardware internal cpu-mac inband stats
i82571 registers
```

```
=====
RMON counters
-----+-----+-----
total packets          70563313          139905960
good packets          70563313          139905960
64 bytes packets              0              0
65-127 bytes packets    66052368          135828505
128-255 bytes packets   1424632           1327796
256-511 bytes packets    280422            325220
512-1023 bytes packets   17060             14480
1024-max bytes packets   2788831          2409959

broadcast packets              0              0
multicast packets              0              0
good octets (hi)                0              0
good octets (low)          18573099828      25929913975
total octets (hi)              0              0
total octets (low)          18573090123      25929922452
XON packets                     0              0
XOFF packets                    0              0
-----> Pause Frame back to R2D2 when the traffic exceeds SUP limit
management packets              0              0

Interrupt counters
-----+-----
Mine          57079706
Other         0
Assertions   57079706
Rx packet timer  9638
Rx absolute timer  0
Rx overrun     0
Rx descr min thresh 0
Tx packet timer  4189
Tx absolute timer  6476
Tx queue empty  0
Tx descr thresh low 0
txdw ..... 44983549
txqe ..... 2
lsc ..... 0
rxseq .... 0
rxdmt .... 213229
rxo ..... 0
rxt ..... 32433891
mdac ..... 0
```

rxcfg 0
gpi 0

Error counters

```
-----+-----  
CRC errors ..... 0  
Alignment errors ..... 0  
Symbol errors ..... 0  
Sequence errors ..... 0  
RX errors ..... 0  
Missed packets (FIFO overflow) 0  
Single collisions ..... 0  
Excessive collisions ..... 0  
Multiple collisions ..... 0  
Late collisions ..... 0  
Collisions ..... 0  
Defers ..... 0  
Tx no CRS ..... 0  
Carrier extension errors ..... 0  
  
Rx length errors ..... 0  
FC Rx unsupported ..... 0  
Rx no buffers ..... 0 ----- no buffer  
Rx undersize ..... 0  
Rx fragments ..... 0  
Rx oversize ..... 0  
Rx jabbers ..... 0  
Rx management packets dropped .. 0  
Tx TCP segmentation context .... 0  
Tx TCP segmentation context fail 0
```

Throttle statistics

```
-----+-----  
Throttle interval ..... 2 * 100ms  
Packet rate limit ..... 32000 pps  
Rate limit reached counter .. 0  
Tick counter ..... 2132276  
Active ..... 0  
Rx packet rate (current/max) 169 / 610 pps ----- Rx rate (current/max)  
Tx packet rate (current/max) 429 / 926 pps
```

NAPI statistics

```
-----+-----  
Weight ..... 64  
Poll scheduled . 57079706  
Poll rescheduled 0  
Poll invoked ... 117135124  
Weight reached . 9  
Tx packets ..... 139905960  
Rx packets ..... 70563313  
Rx congested ... 0  
Rx redelivered . 0
```

qdisc stats:

```
-----+-----  
Tx queue depth . 1000  
qlen ..... 0  
packets ..... 139905960  
bytes ..... 23411617016  
drops ..... 0
```


Bahrain registers (cleared by chip reset only)

```

=====
revision          0x00000108
scratchpad        0xaaaaaaaa
MAC status        0x00000001
MAC SerDes synced 0x00000001
MAC status 2      0x000100f8
Auto-XOFF config  1
Auto-XOFF status  0
  
```

MAC counters	MAC0 (R2D2)		MAC1 (CPU)	
	Rx	Tx	Rx	Tx
64 bytes packets	0	0	0	0
65-127 bytes packets	66907289	136682635	135828505	66052368
128-255 bytes packets	570131	473705	1327796	1424632
256-511 bytes packets	280003	325182	325220	280422
512-1023 bytes packets	17061	14482	14480	17060
1024-1518 bytes packets	623614	242009	241831	623569
1519-max bytes packets	2165215	2167947	2168128	2165262
total packets	70563313	139905960	139905960	70563313
total bytes	405350248	2496404376	160120520	1393236630
undersized packets	0		0	
fragmented packets	0		0	
FCS errors	0		0	
auto-XOFF state entered	0 times			
auto-XOFF reset	0 times			
XOFF packets auto-generated		0		
XOFF packets		0	0	
XON packets	0		0	
parity error	0	0	0	0
fifo errors	0		0	
overflow errors		0		0

Après la version 5.X de NX-OS, « events » est une option de commande qui indique le moment où le débit maximal de paquets par seconde (PPS) en réception (RX) ou en transmission (TX) est atteint. Cet exemple montre comment déterminer l'heure à laquelle le dernier pic de trafic CPU a été rencontré :

```
switch# show hardware internal cpu-mac inband events
```

- 1) Event:TX_PPS_MAX, length:4, at 648617 usecs after Fri Oct 19 13:23:06 2012
new maximum = 926

- 2) Event:TX_PPS_MAX, length:4, at 648622 usecs after Fri Oct 19 13:15:06 2012
new maximum = 916

- 3) Event:TX_PPS_MAX, length:4, at 648612 usecs after Fri Oct 19 13:14:06 2012
new maximum = 915

- 4) Event:TX_PPS_MAX, length:4, at 648625 usecs after Fri Oct 19 13:12:06 2012
new maximum = 914
- 5) Event:TX_PPS_MAX, length:4, at 648626 usecs after Fri Oct 19 13:11:06 2012
new maximum = 911
- 6) Event:TX_PPS_MAX, length:4, at 648620 usecs after Fri Oct 19 13:08:06 2012
new maximum = 910

show system internal pktmgr internal vdc inband <int>, commande

Utilisez cette commande pour identifier la source du trafic acheminé vers le processeur.

```
switch# show system internal pktmgr internal vdc inband e1/5
Interface          Src Index      VDC ID      Packet rcvd
-----
Ethernet1/5        0xa1d         1           14640
```

Netstack/Pktmgr

Netstack est une pile IP complète implémentée dans l'espace utilisateur de Nexus 7000. Les composants incluent un gestionnaire de paquets de couche 2, ARP, un gestionnaire de contiguïté, IPv4, ICMPv4 (Internet Control Message Protocol v4), IPv6, ICMPv6, TCP/UDP et une bibliothèque de sockets. Lorsque le trafic vers le processeur déclenche une utilisation élevée du processeur, vous constatez souvent que Netstack et son processus respectif s'exécutent à un niveau élevé.

Commande show system inband queuing status

Cet exemple montre comment afficher l'algorithme de mise en file d'attente Netstack utilisé :

```
switch# show system inband queuing status
Weighted Round Robin Algorithm
Weights BPDU - 32, Q0 - 8, Q1 - 4, Q2 - 2 Q3 - 64
```

Commande show system inband queuing statistics

Cet exemple montre les compteurs dans le module chargeable par le noyau (KLM) et le processus d'espace utilisateur.

Le KLM est une instance unique qui s'exécute sur le VDC par défaut et fonctionne à la fois sur l'interface intrabande et l'interface de gestion. Le KLM n'entre dans l'image que pendant le traitement des paquets en entrée pour l'envoi des trames en entrée à la pile réseau VDC appropriée pour le traitement.

```
switch# show system inband queuing statistics
Inband packets unmapped to a queue: 0
Inband packets mapped to bpdu queue: 7732593
Inband packets mapped to q0: 686667
Inband packets mapped to q1: 0
Inband packets mapped to q2: 0
Inband packets mapped to q3: 20128
In KLM packets mapped to bpdu: 7732593
In KLM packets mapped to arp : 912
In KLM packets mapped to q0 : 686667
In KLM packets mapped to q1 : 0
In KLM packets mapped to q2 : 0
In KLM packets mapped to q3 : 20128
In KLM packets mapped to veobc : 0
Inband Queues:
bpdu: rcv 1554390, drop 0, congested 0 rcvbuf 2097152, sndbuf 262142 no drop 1
(q0): rcv 686667, drop 0, congested 0 rcvbuf 2097152, sndbuf 262142 no drop 0
(q1): rcv 0, drop 0, congested 0 rcvbuf 2097152, sndbuf 262142 no drop 0
(q2): rcv 0, drop 0, congested 0 rcvbuf 2097152, sndbuf 262142 no drop 0
(q3): rcv 20128, drop 0, congested 0 rcvbuf 2097152, sndbuf 262142 no drop 0
```

Commande show system internal pktmgr internal vdc global-stats

Cette commande est similaire à la commande précédente show system inband queuing statistics et fournit de nombreux détails :

```
switch# show system internal pktmgr internal vdc global-stats
```

```
VDC KLM global statistics:
Inband packets not mapped to a VDC: 0
Inband diag packets received: 998222
Weighted Round Robin Algorithm
Weights BPDU - 32, Q0 - 8, Q1 - 4, Q2 - 2 Q3 - 64
Inband packets unmapped to a queue: 0
Inband packets mapped to bpdu queue: 7734430 (7734430)
Inband packets mapped to q0: 686779 (686779)
Inband packets mapped to q1: 0 (0)
Inband packets mapped to q2: 0 (0)
Inband packets mapped to q3: 20128 (20128)
Pkt Size History : 2811395 for index 1
Pkt Size History : 274508 for index 2
Pkt Size History : 74284 for index 3
Pkt Size History : 43401 for index 4
Pkt Size History : 70915 for index 5
Pkt Size History : 35602 for index 6
Pkt Size History : 30085 for index 7
Pkt Size History : 29408 for index 8
Pkt Size History : 21221 for index 9
```

Pkt Size History : 15683 for index 10
Pkt Size History : 13212 for index 11
Pkt Size History : 10646 for index 12
Pkt Size History : 9290 for index 13
Pkt Size History : 50298 for index 14
Pkt Size History : 5473 for index 15
Pkt Size History : 4871 for index 16
Pkt Size History : 4687 for index 17
Pkt Size History : 5507 for index 18
Pkt Size History : 15416 for index 19
Pkt Size History : 11333 for index 20
Pkt Size History : 5478 for index 21
Pkt Size History : 4281 for index 22
Pkt Size History : 3543 for index 23
Pkt Size History : 3059 for index 24
Pkt Size History : 2228 for index 25
Pkt Size History : 4390 for index 26
Pkt Size History : 19892 for index 27
Pkt Size History : 524 for index 28
Pkt Size History : 478 for index 29
Pkt Size History : 348 for index 30
Pkt Size History : 447 for index 31
Pkt Size History : 1545 for index 32
Pkt Size History : 152 for index 33
Pkt Size History : 105 for index 34
Pkt Size History : 1424 for index 35
Pkt Size History : 43 for index 36
Pkt Size History : 60 for index 37
Pkt Size History : 60 for index 38
Pkt Size History : 46 for index 39
Pkt Size History : 58 for index 40
Pkt Size History : 829 for index 41
Pkt Size History : 32 for index 42
Pkt Size History : 26 for index 43
Pkt Size History : 1965 for index 44
Pkt Size History : 21 for index 45
Pkt Size History : 1 for index 46
Pkt Size History : 1 for index 48
Pkt Size History : 1 for index 51
Pkt Size History : 1 for index 52
Pkt Size History : 1 for index 53
Pkt Size History : 3 for index 55
In KLM packets mapped to bpdu: 7734430
In KLM packets mapped to arp : 912
In KLM packets mapped to q0 : 686779
In KLM packets mapped to q1 : 0
In KLM packets mapped to q2 : 0
In KLM packets mapped to q3 : 20128
In KLM packets mapped to veobc : 0
In KLM Queue Mapping (0 1 2 3 4)
Data Available in FDs (0 0 0 0 0)
Inband Queues:
bpdu: rcv 1556227, drop 0, congested 0 rcvbuf 2097152, sndbuf 262142 no drop 1
(q0): rcv 686779, drop 0, congested 0 rcvbuf 2097152, sndbuf 262142 no drop 0
(q1): rcv 0, drop 0, congested 0 rcvbuf 2097152, sndbuf 262142 no drop 0
(q2): rcv 0, drop 0, congested 0 rcvbuf 2097152, sndbuf 262142 no drop 0
(q3): rcv 20128, drop 0, congested 0 rcvbuf 2097152, sndbuf 262142 no drop 0
Mgmt packets not mapped to a VDC: 227551
Mgmt multicast packets dropped: 92365
Mgmt multicast packets delivered: 0
Mgmt packets broadcast to each VDC: 23119
Mgmt debugging packets copied: 0

```
Mgmt IPv6 multicast packets delivered: 0
Mgmt IPv6 link-local packets delivered: 0
Mgmt LLDP packets received: 0
```

show system internal pktmgr interface ethernet <int>, commande

Utilisez cette commande afin d'examiner le débit de paquets ainsi que le type de trafic (monodiffusion ou multidiffusion) pour le trafic envoyé par le processeur à partir d'une interface.

```
switch# show system internal pktmgr interface e1/5
Ethernet1/5, ordinal: 73
  SUP-traffic statistics: (sent/received)
    Packets: 63503 / 61491
    Bytes: 6571717 / 5840641
    Instant packet rate: 0 pps / 0 pps
    Packet rate limiter (Out/In): 0 pps / 0 pps
    Average packet rates(1min/5min/15min/EWMA):
    Packet statistics:
      Tx: Unicast 3198, Multicast 60302
         Broadcast 3
      Rx: Unicast 3195, Multicast 58294
         Broadcast 2
```

show system internal pktmgr client <uuid>, commande

Cette commande affiche les applications telles que STP ou CDP (Cisco Discovery Protocol) qui sont enregistrées auprès de Packet Manager, ainsi que le nombre de paquets envoyés et reçus par ces applications.

```
switch# show system internal pktmgr client
Client uuid: 268, 4 filters, pid 3127
  Filter 1: EthType 0x0806,
  Rx: 2650, Drop: 0
  Filter 2: EthType 0xffff0, Exc 8,
  Rx: 0, Drop: 0
  Filter 3: EthType 0x8841, Snap 34881,
  Rx: 0, Drop: 0
  Filter 4: EthType 0x0800, DstIf 0x150b0000, Excl. Any
  Rx: 0, Drop: 0
Options: TO 0, Flags 0x18040, AppId 0, Epid 0
Ctrl SAP: 278, Data SAP 337 (1)
Total Rx: 2650, Drop: 0, Tx: 1669, Drop: 0
Recirc Rx: 0, Drop: 0
Rx pps Inst/Max: 0/20
Tx pps Inst/Max: 0/5
COS=0 Rx: 0, Tx: 0    COS=1 Rx: 912, Tx: 0
COS=2 Rx: 0, Tx: 0    COS=3 Rx: 0, Tx: 0
COS=4 Rx: 0, Tx: 0    COS=5 Rx: 0, Tx: 1669
COS=6 Rx: 0, Tx: 0    COS=7 Rx: 1738, Tx: 0
```

```
Client uuid: 270, 1 filters, pid 3128
Filter 1: EthType 0x86dd, DstIf 0x150b0000, Excl. Any
Rx: 0, Drop: 0
Options: TO 0, Flags 0x18040, AppId 0, Epid 0
Ctrl SAP: 281, Data SAP 283 (1)
Total Rx: 0, Drop: 0, Tx: 0, Drop: 0
Recirc Rx: 0, Drop: 0
Rx pps Inst/Max: 0/0
Tx pps Inst/Max: 0/0
COS=0 Rx: 0, Tx: 0   COS=1 Rx: 0, Tx: 0
COS=2 Rx: 0, Tx: 0   COS=3 Rx: 0, Tx: 0
COS=4 Rx: 0, Tx: 0   COS=5 Rx: 0, Tx: 0
COS=6 Rx: 0, Tx: 0   COS=7 Rx: 0, Tx: 0
```

Commande show system internal pktmgr stats

Utilisez cette commande afin de vérifier si les paquets atteignent le gestionnaire de paquets dans le chemin d'entrée et si les paquets sont envoyés par le gestionnaire de paquets. Cette commande peut également vous aider à déterminer s'il existe des problèmes avec les tampons dans le chemin de réception ou de transmission.

```
switch# show system internal pktmgr stats
Route Processor Layer-2 frame statistics

Inband driver: valid 1, state 0, rd-thr 1, wr-thr 0, Q-count 0
Inband sent: 56441521, copy_drop: 0, ioctl_drop: 0,
  unavailable_buffer_hdr_drop: 0
Inband standby_sent: 0
Inband encap_drop: 0, linecard_down_drop: 0
Inband sent by priority [0=11345585,5=164281,6=43280117,7=1651538]
Inband max output queue depth 0
Inband recv: 89226232, copy_drop: 0, ioctl_drop: 0,
  unavailable_buffer_hdr_drop: 0
Inband decap_drop: 0, crc_drop: 0, recv by priority: [0=89226232]
Inband bad_si 0, bad_if 0, if_down 0
Inband last_bad_si 0, last_bad_if 0, bad_di 0
Inband kernel recv 44438488, drop 0, rcvbuf 2097152, sndbuf 4194304

Mgmt driver: valid 1, state 0, rd-thr 1, wr-thr 0, Q-count 0
Mgmt sent: 971834, copy_drop: 0, ioctl_drop: 0,
  unavailable_buffer_hdr_drop: 0
Mgmt standby_sent: 0
Mgmt encap_drop: 0, linecard_down_drop: 0
Mgmt sent by priority [0=925871,5=45963]
Mgmt max output queue depth 0
Mgmt recv: 1300932, copy_drop: 0, ioctl_drop: 0,
  unavailable_buffer_hdr_drop: 0
Mgmt decap_drop: 0, crc_drop: 0, recv by priority: [0=1300932]
Mgmt bad_si 0, bad_if 0, if_down 0
Mgmt last_bad_si 0, last_bad_if 0, bad_di 0
Mgmt kernel recv 1300932, drop 0, rcvbuf 2097152, sndbuf 2097152

Inband2 driver: valid 0, state 1, rd-thr 0, wr-thr 0, Q-count 0

No of packets passed by   PM Policy database      876452
No of packets dropped by  PM Policy database      0
```

No of packets bypassed by PM Policy database 424480
No of packets dropped by PM originating from kernel 0

MBUFSK Tx: 57413355 pkts (requested 57413355 denied 0), 62236110 mbufs
function invoked 57413355 denied 0/0 c/realloc 0/0
MBUFSK Rx: 90527161 pkts, 90527421 mbufs (requested 2388154951 denied 0)
function invoked 35132836

Global input drops: bad-interface 0, bad-encap 0, failed-decap 0,
no prot 42371
recv_encaptype_err 0, recv_decap_err 0, recv_mac_mismatch 0, recv_no_client 0
recv_no_svi 0, recv_no_vlan 0, recv_client_notreg 0, recv_enqueue_fail 0

Global output drops:
send_ifdown_fail 13, send_invalid_iod 0
send_invalid_vlan 0, send_security_drop 0 send_loopback_drop 0,
send_small_pkt_fail 0
send_vsl_err 0, send_dce_err 0, send_enqueue_fail 0, send_alloc_fail 0

DCE errors:
misc_err 0, lookup_err 0, encap_err 0, decap_err 0

Platform errors:
generic_encap_err 0, encap_err 0, decap_err 0
vlan_encap_err 0, vlan_decap_err 0

DC3HDR errors:
pkt_err 0, vlan_err 0, ifidx_err 0, portidx_err 0

RECIRC errors:
misc_err 0, lookup_err 0

Lcache errors:
init_err 0, timer_err 0

Stats errors:
misc_err 0, init_err 0, timer_err 0

Client errors:
alloc_err 0, pid_err 0, register_err 0, unregister_err 0
add_err 0, delete_err 0, update_err 0

VDC errors:
alloc_err 0, set_err 0, update_err 0

Misc. errors:
mts_err 0, mbuf_err 0, drop_exception 0
invalid_drv_type 0, interface_err 0
eth_output_err 0, gre_err 0 otv_err 0
tunnel_6to4_err 0, mcec_err 0, invalid_gpc 0 invalid_ftag 0 invalid_l2_type :0
register_err 0, unregister_err 0, invalid_args 0, file_open_err 0
inband_err 0, vlan_err 0, pm_alloc_err 0, pm_ha_err 0, pm_init_err 0
arp_init_err 0, rtm_init_err 0, am_init_err 0, ui_init_err 0, mpls_init_err 0,
evc_init_err 0
sdb_err 95670, sdb_init_err 0
sysmgr_err 0, eth_span_err 0, buf_pool_err 0, feature_err 0
uuid2client_err 16, dot1q_drop 0, nfcache_init_err 0

Crossbar down drops : 0
Exception packets: mtu-fail 0, icmp-redirect 0, icmp-unreach 0, ttl 0
options 0, rpf 0, two-mcast-rpf 0, l3-bridge-drop 0
mcast-next-hop 0, municast 0

drop 0, acl-redirect 0, acl-redir-arp 0, acl-redir-dhcp 0
sup-shim-pkt 229385 Pkts recvd with peergway SUP DI 0

VPC Frame Statistics

VPC Mgr reg state 1, im-ext-sdb-state 1
Ingress BPDUs qualified for redirection 0
Ingress BPDUs redirected to peer 0
Egress BPDUs qualified for redirection 0
Egress BPDUs dropped due to remote down 0
Egress BPDUs redirected to peer 0
Ingress pkts qualified for peergateway tunneling 0
Ingress pkts tunneled to peer with peergateway conf 0
Peer-gw pkts tunneled tx :
 From VPC+ leg 0, From VPC leg 0, From l2mp network 0
 From orphan port in VPC+ 0, from orphan port in VPC 0
 For ARP 0, IP 0, IPv6 0, unknown 0
Total Tunneled packets received from peer 0
Local delivery 0, Transmit down 0, peer-gw tunneled 0
Tunnel rx packets drop due to local vpc leg down 0
Peer-gw pkts tunneled rx :
 From VPC+ leg 0, VPC leg 0, From l2mp network 0
 From orphan port in VPC+ 0, from orphan port in VPC 0
 For ARP 0, IP 0, IPv6 0, unknown 0

Error Statistics

VPC manager: uninit 0, library 0
Tunnel (ingress): non-mct rx 0, bad hdr 0, badpkts 0, non gpc peer 0
Tunnel (ingress): redirlooperror 0
Tunnel (egress): in-bpdu 0, e-bpdu 0, peer-gw 0
Mbuf: alloc: 0, prepend: 0, pullup: 0
Invalid filter: 0
Peergw tunneling tx: invalid ftag 0, invalid swid 0
 invalid iftype 0, invalid GPC of peer 0
Peergw tunneling rx: invalid msg subtype 0, invalid GPC of core 0
 invalid GPC of peer 0, invalid svi 0
Unicast pkts which passed egress redirection check 0

statistics last reset 2w0d

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.