Comprendre l'utilisation élevée du CPU signalée par vManage pour les plates-formes cloud vEdge 5000/2000/1000/100B et vEdge

Table des matières

Introduction

Comprendre l'utilisation élevée du CPU signalée sur les plates-formes cloud vEdge 5000/2000/1000/100B et vEdge

- Explication

Utilisation CPU élevée avec le processus fp-um

Conclusion

Introduction

Ce document explique pourquoi l'utilisation élevée du CPU peut être signalée sur les plates-formes vManage pour vEdge 5000/2000/1000/100B et vEdge Cloud, bien que les performances des plates-formes soient normales et qu'aucune utilisation élevée du CPU ne soit signalée comme indiqué en haut.

Comprendre l'utilisation élevée du CPU signalée sur les plates-formes cloud vEdge 5000/2000/1000/100B et vEdge

Avec la version 17.2.x et les versions ultérieures, une consommation accrue du processeur et de la mémoire pour les plates-formes cloud vEdge et vEdge peut être observée. Ceci est remarqué sur le tableau de bord vManage pour un périphérique donné. Dans certains cas, cela entraîne également une augmentation du nombre d'alertes et d'avertissements dans vManage.

Explication

La raison de l'utilisation élevée de l'UC signalée lorsque le périphérique fonctionne normalement avec une charge normale, faible ou nulle est due à un changement dans la formule utilisée afin de calculer l'utilisation. Avec les versions 17.2, l'utilisation de l'UC est calculée sur la base de la moyenne de charge à partir de show system status sur le vEdge.

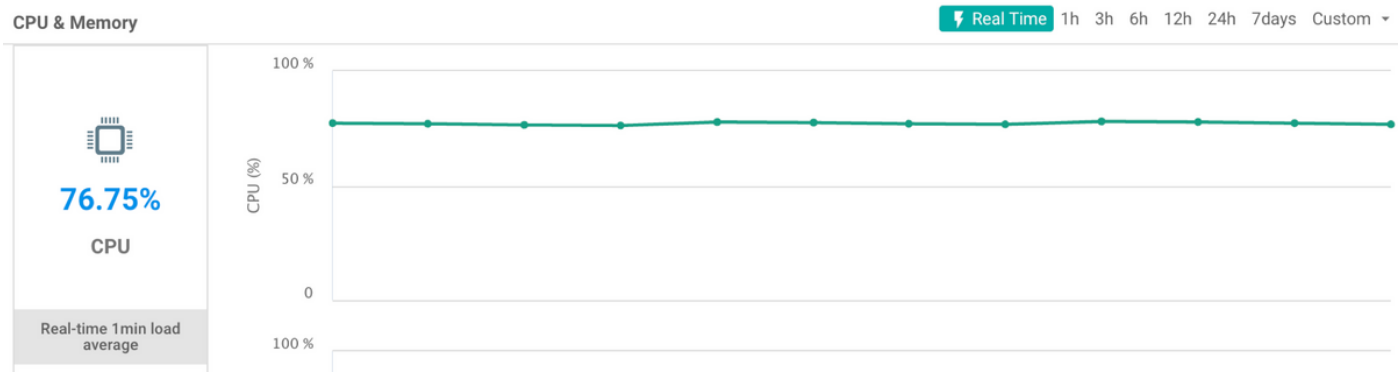
vManage indique l'utilisation du processeur en temps réel pour un périphérique. Il extrait la moyenne sur 1 minute [min1_avg] et la moyenne sur 5 minutes [min5_avg] sur la base des données historiques. La charge moyenne, par définition, inclut divers éléments et pas seulement les cycles de CPU qui contribuent au calcul de l'utilisation. Par exemple, le temps d'attente des E/S, le temps d'attente des processus et d'autres valeurs sont pris en compte lorsque vous présentez cette valeur pour la plate-forme. Dans ce cas, vous ignorez les valeurs affichées pour les états de CPU et les valeurs de CPU dans la commande top de vShell.

Voici un exemple sur la façon dont l'utilisation du CPU, qui est en fait la moyenne de charge d'1 minute, est calculée et affichée dans le tableau de bord vManage :

Lorsque vous contrôlez la charge à partir d'une interface de ligne de commande vEdge, vous pouvez observer :

```
vEdge# show system status | include Load
Load average:            1 minute: 3.10, 5 minutes: 3.06, 15 minutes: 3.05
Load average:            1 minute: 3.12, 5 minutes: 3.07, 15 minutes: 3.06
Load average:            1 minute: 3.13, 5 minutes: 3.08, 15 minutes: 3.07
Load average:            1 minute: 3.10, 5 minutes: 3.07, 15 minutes: 3.05
```

Dans ce cas, l'utilisation du CPU est calculée sur la base de Load-Average / # of Cores (vCPUs). Dans cet exemple, le noeud comporte 4 coeurs. La charge moyenne est ensuite convertie par un facteur de 100 avant que vous ne divisiez par le nombre de coeurs. Lorsque vous calculez la charge moyenne de tous les coeurs et multipliez par 100, vous obtenez une valeur d'environ 310. Prenez cette valeur et divisez par 4 rendements, une valeur UC de 77,5 % UC, qui correspond à la valeur affichée dans le graphique en temps réel dans vManage capturée au moment où la sortie CLI a été collectée et comme illustré dans l'image.



Afin de voir les moyennes de charge et le nombre de coeurs de CPU dans le système, la sortie de top peut être consultée à partir de vShell sur le périphérique.

Dans l'exemple ici, le serveur vEdge contient 4 processeurs virtuels. Le premier coeur (Cpu0) est utilisé pour le contrôle (vu à travers la plus faible utilisation de l'utilisateur) tandis que les 3 coeurs restants sont utilisés pour les données :

```
top - 01:14:57 up 1 day,  3:15,  1 user,  load average: 3.06, 3.06, 3.08
Tasks: 219 total,   5 running, 214 sleeping,   0 stopped,   0 zombie
Cpu0  :  1.7%us,  4.0%sy,  0.0%ni, 94.3%id,  0.0%wa,  0.0%hi,  0.0%si,  0.0%st
Cpu1  : 56.0%us, 44.0%sy,  0.0%ni,  0.0%id,  0.0%wa,  0.0%hi,  0.0%si,  0.0%st
Cpu2  : 54.2%us, 45.8%sy,  0.0%ni,  0.0%id,  0.0%wa,  0.0%hi,  0.0%si,  0.0%st
Cpu3  : 59.3%us, 40.7%sy,  0.0%ni,  0.0%id,  0.0%wa,  0.0%hi,  0.0%si,  0.0%st
Mem:   7382664k total,  2835232k used,  4547432k free,   130520k buffers
Swap:        0k total,        0k used,        0k free,   587880k cached

  PID USER      PR  NI  VIRT  RES  SHR S %CPU %MEM    TIME+  COMMAND
```

```
  978 root      20   0 3392m 664m 127m R  100  9.2   1635:21 fp-um-2
  692 root      20   0 3392m 664m 127m R  100  9.2   1635:18 fp-um-1
  979 root      20   0 3392m 664m 127m R  100  9.2   1634:51 fp-um-3
  694 root      20   0 1908m 204m 131m S    1  2.8  15:29.95 ftmd
  496 root      20   0  759m  72m 3764 S    0  1.0   1:31.50 confd
```

Afin d'obtenir le nombre de CPU à partir de l'interface de ligne de commande vEdge, cette commande peut être utilisée :

```
vEdge# show system status | display xml | include total_cpu
    <total_cpu_count>4</total_cpu_count>
```

Un autre exemple de calcul de la valeur affichée dans vManage sur vEdge 1000 est fourni ici. Après avoir émis top à partir de vShell, l est intéressé afin d'afficher la charge pour tous les coeurs :

```
top - 18:19:49 up 19 days,  1:37,  1 user,  load average: 0.55, 0.71, 0.73
```

Comme un serveur vEdge 1000 ne dispose que d'un coeur de processeur, la charge indiquée ici est de 55 % (0,55*100).

Utilisation CPU élevée avec le processus fp-um

Vous pouvez aussi parfois remarquer d'en haut que le processus fp-um s'exécute haut et montre jusqu'à 100% CPU. Cela est prévu sur les coeurs de processeur utilisés pour le traitement du plan de données.

À partir de la commande top mentionnée précédemment, 3 coeurs fonctionnent à 100% CPU et 1 coeur montre une utilisation normale :

```
top - 01:14:57 up 1 day,  3:15,  1 user,  load average: 3.06, 3.06, 3.08
Tasks: 219 total,   5 running, 214 sleeping,   0 stopped,   0 zombie
Cpu0  :  1.7%us,  4.0%sy,  0.0%ni, 94.3%id,  0.0%wa,  0.0%hi,  0.0%si,  0.0%st
Cpu1  : 56.0%us, 44.0%sy,  0.0%ni,  0.0%id,  0.0%wa,  0.0%hi,  0.0%si,  0.0%st
Cpu2  : 54.2%us, 45.8%sy,  0.0%ni,  0.0%id,  0.0%wa,  0.0%hi,  0.0%si,  0.0%st
Cpu3  : 59.3%us, 40.7%sy,  0.0%ni,  0.0%id,  0.0%wa,  0.0%hi,  0.0%si,  0.0%st
Mem:   7382664k total,  2835232k used,  4547432k free,   130520k buffers
Swap:        0k total,        0k used,        0k free,   587880k cached

  PID USER      PR  NI  VIRT  RES  SHR S %CPU %MEM    TIME+  COMMAND
  978 root      20   0 3392m 664m 127m R  100  9.2   1635:21 fp-um-2
  692 root      20   0 3392m 664m 127m R  100  9.2   1635:18 fp-um-1
  979 root      20   0 3392m 664m 127m R  100  9.2   1634:51 fp-um-3
...
```

Ce premier coeur (Cpu0) est utilisé pour le contrôle et les trois autres coeurs pour les données. Comme vous pouvez le voir dans la liste des processus, le processus fp-um utilise ces ressources.

fp-um est un processus qui utilise un pilote de mode d'interrogation, ce qui signifie qu'il se place et interroge le port sous-jacent pour les paquets en permanence afin qu'il puisse traiter n'importe quelle trame dès qu'elle est reçue. Ce processus gère le transfert et est équivalent au transfert par chemin rapide dans les systèmes vEdge 1000, vEdge 2000 et vEdge 100. Cette architecture de mode d'interrogation est utilisée par Intel pour un traitement efficace des paquets basé sur l'infrastructure du Kit de développement du plan de données (DPDK). Étant donné que le transfert de paquets est mis en oeuvre dans une boucle étroite, le processeur reste à 100 % ou à un niveau proche de 100 % à tout moment. Bien que cela soit fait, aucune latence n'est introduite par ces CPU car c'est le comportement attendu.

Des informations générales sur l'interrogation DPDK sont disponibles ici.

Les plates-formes vEdge Cloud et vEdge 5000 utilisent la même architecture de transfert et présentent le même comportement à cet égard. Voici un exemple d'un serveur vEdge 5000 tiré de la sortie supérieure. Il comporte 28 coeurs, dont 2 (Cpu0 et Cpu1) sont utilisés pour le contrôle (comme le vEdge 2000) et 26 pour les données.

```
top - 02:18:30 up 1 day,  7:33,  1 user,  load average: 26.24, 26.28, 26.31
Tasks: 382 total,  27 running, 355 sleeping,   0 stopped,   0 zombie
Cpu0  :  0.7%us,  1.3%sy,  0.0%ni, 98.0%id,  0.0%wa,  0.0%hi,  0.0%si,  0.0%st
Cpu1  :  0.7%us,  1.3%sy,  0.0%ni, 98.0%id,  0.0%wa,  0.0%hi,  0.0%si,  0.0%st
Cpu2  : 79.4%us, 20.6%sy,  0.0%ni,  0.0%id,  0.0%wa,  0.0%hi,  0.0%si,  0.0%st
Cpu3  : 73.4%us, 26.6%sy,  0.0%ni,  0.0%id,  0.0%wa,  0.0%hi,  0.0%si,  0.0%st
Cpu4  : 73.4%us, 26.6%sy,  0.0%ni,  0.0%id,  0.0%wa,  0.0%hi,  0.0%si,  0.0%st
Cpu5  :100.0%us,  0.0%sy,  0.0%ni,  0.0%id,  0.0%wa,  0.0%hi,  0.0%si,  0.0%st
Cpu6  :100.0%us,  0.0%sy,  0.0%ni,  0.0%id,  0.0%wa,  0.0%hi,  0.0%si,  0.0%st
Cpu7  :100.0%us,  0.0%sy,  0.0%ni,  0.0%id,  0.0%wa,  0.0%hi,  0.0%si,  0.0%st
Cpu8  :100.0%us,  0.0%sy,  0.0%ni,  0.0%id,  0.0%wa,  0.0%hi,  0.0%si,  0.0%st
Cpu9  :100.0%us,  0.0%sy,  0.0%ni,  0.0%id,  0.0%wa,  0.0%hi,  0.0%si,  0.0%st
Cpu10 :100.0%us,  0.0%sy,  0.0%ni,  0.0%id,  0.0%wa,  0.0%hi,  0.0%si,  0.0%st
Cpu11 :100.0%us,  0.0%sy,  0.0%ni,  0.0%id,  0.0%wa,  0.0%hi,  0.0%si,  0.0%st
Cpu12 :100.0%us,  0.0%sy,  0.0%ni,  0.0%id,  0.0%wa,  0.0%hi,  0.0%si,  0.0%st
Cpu13 :100.0%us,  0.0%sy,  0.0%ni,  0.0%id,  0.0%wa,  0.0%hi,  0.0%si,  0.0%st
Cpu14 :100.0%us,  0.0%sy,  0.0%ni,  0.0%id,  0.0%wa,  0.0%hi,  0.0%si,  0.0%st
Cpu15 :100.0%us,  0.0%sy,  0.0%ni,  0.0%id,  0.0%wa,  0.0%hi,  0.0%si,  0.0%st
Cpu16 :100.0%us,  0.0%sy,  0.0%ni,  0.0%id,  0.0%wa,  0.0%hi,  0.0%si,  0.0%st
Cpu17 :100.0%us,  0.0%sy,  0.0%ni,  0.0%id,  0.0%wa,  0.0%hi,  0.0%si,  0.0%st
Cpu18 :100.0%us,  0.0%sy,  0.0%ni,  0.0%id,  0.0%wa,  0.0%hi,  0.0%si,  0.0%st
Cpu19 :100.0%us,  0.0%sy,  0.0%ni,  0.0%id,  0.0%wa,  0.0%hi,  0.0%si,  0.0%st
Cpu20 :100.0%us,  0.0%sy,  0.0%ni,  0.0%id,  0.0%wa,  0.0%hi,  0.0%si,  0.0%st
Cpu21 :100.0%us,  0.0%sy,  0.0%ni,  0.0%id,  0.0%wa,  0.0%hi,  0.0%si,  0.0%st
Cpu22 :100.0%us,  0.0%sy,  0.0%ni,  0.0%id,  0.0%wa,  0.0%hi,  0.0%si,  0.0%st
Cpu23 :100.0%us,  0.0%sy,  0.0%ni,  0.0%id,  0.0%wa,  0.0%hi,  0.0%si,  0.0%st
Cpu24 :100.0%us,  0.0%sy,  0.0%ni,  0.0%id,  0.0%wa,  0.0%hi,  0.0%si,  0.0%st
Cpu25 :100.0%us,  0.0%sy,  0.0%ni,  0.0%id,  0.0%wa,  0.0%hi,  0.0%si,  0.0%st
Cpu26 :100.0%us,  0.0%sy,  0.0%ni,  0.0%id,  0.0%wa,  0.0%hi,  0.0%si,  0.0%st
Cpu27 :100.0%us,  0.0%sy,  0.0%ni,  0.0%id,  0.0%wa,  0.0%hi,  0.0%si,  0.0%st
Mem:  32659508k total, 10877980k used, 21781528k free,   214788k buffers
Swap:        0k total,        0k used,        0k free,  1039104k cached
```

```
  PID USER      PR  NI    VIRT    RES    SHR S  %CPU  %MEM     TIME+  COMMAND
 2028 root      20   0  12.1g  668m  124m R   100   2.1   1897:21 fp-um-3
 2029 root      20   0  12.1g  668m  124m R   100   2.1   1897:22 fp-um-4
 2030 root      20   0  12.1g  668m  124m R   100   2.1   1897:12 fp-um-5
 2031 root      20   0  12.1g  668m  124m R   100   2.1   1897:22 fp-um-6
 2032 root      20   0  12.1g  668m  124m R   100   2.1   1897:22 fp-um-7
 2034 root      20   0  12.1g  668m  124m R   100   2.1   1897:22 fp-um-9
 2035 root      20   0  12.1g  668m  124m R   100   2.1   1897:21 fp-um-10
 2038 root      20   0  12.1g  668m  124m R   100   2.1   1897:21 fp-um-13
 2040 root      20   0  12.1g  668m  124m R   100   2.1   1897:23 fp-um-15
 2041 root      20   0  12.1g  668m  124m R   100   2.1   1897:23 fp-um-16
 2043 root      20   0  12.1g  668m  124m R   100   2.1   1897:22 fp-um-18
 2045 root      20   0  12.1g  668m  124m R   100   2.1   1897:23 fp-um-20
 2052 root      20   0  12.1g  668m  124m R   100   2.1   1897:18 fp-um-27
 2033 root      20   0  12.1g  668m  124m R   100   2.1   1897:21 fp-um-8
 2036 root      20   0  12.1g  668m  124m R   100   2.1   1897:21 fp-um-11
 2037 root      20   0  12.1g  668m  124m R   100   2.1   1897:21 fp-um-12
 2039 root      20   0  12.1g  668m  124m R   100   2.1   1897:09 fp-um-14
 2042 root      20   0  12.1g  668m  124m R   100   2.1   1897:23 fp-um-17
 2044 root      20   0  12.1g  668m  124m R   100   2.1   1897:23 fp-um-19
 2046 root      20   0  12.1g  668m  124m R   100   2.1   1897:23 fp-um-21
 2047 root      20   0  12.1g  668m  124m R   100   2.1   1897:23 fp-um-22
 2048 root      20   0  12.1g  668m  124m R   100   2.1   1897:23 fp-um-23
 2049 root      20   0  12.1g  668m  124m R   100   2.1   1897:23 fp-um-24
 2050 root      20   0  12.1g  668m  124m R   100   2.1   1897:22 fp-um-25
 2051 root      20   0  12.1g  668m  124m R   100   2.1   1897:23 fp-um-26
 1419 root      20   0   116m 5732 2280 S     0   0.0   0:02.00 chmgrd
 1323 root      20   0   753m   70m 3764 S     0   0.2   1:51.20 confd
 1432 root      20   0  1683m  172m  134m S     0   0.5   0:58.91 fpmd
```

Ici, la charge moyenne est toujours élevée car 26 des 28 processeurs fonctionnent à 100 % grâce au processus fp-um.

Conclusion

L'utilisation CPU signalée dans vManage pour les versions 17.2.x antérieures à 17.2.7 n'est pas l'utilisation CPU réelle, mais est plutôt calculée sur la base de la moyenne de charge. Cela peut conduire à une confusion dans la compréhension de la valeur signalée et à de fausses alarmes liées à une CPU élevée alors que la plate-forme fonctionne normalement avec un trafic/une charge réseau normal, faible ou inexistant.

Ce comportement est modifié avec les versions 17.2.7 et 18.2, de sorte que la lecture du CPU peut maintenant être précise en fonction de la lecture de cpu_user à partir du haut.

Le problème est également mentionné dans les notes de version 17.2.

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.