Entienda CPU elevada la utilización señalada por el vManage para el vEdge 5000/1000/100B y las Plataformas de la nube del vEdge

Contenido

Introducción

Este documento describe porqué usted puede ser que vea CPU elevada el uso señalado en el vManage para el vEdge 5000/1000/100B y las Plataformas de la nube del vEdge a pesar del funcionamiento de las Plataformas que eran normal sin CPU elevada señalado según lo visto en el **top**.

Entienda CPU elevada la utilización que está señalada sobre el vEdge 5000/1000/100B y las Plataformas de la nube del vEdge

Con el 17.2.x y las versiones posteriores, una CPU y una consumición de la memoria más altas para las Plataformas del vEdge y de la nube del vEdge pueden ser observadas. Esto se nota en el panel del vManage para un dispositivo dado. En algunos casos, esto también lleva a un mayor número de alertas y de advertencias en el vManage.

Explicación

La razón CPU elevada del uso señalado cuando el dispositivo se realiza normalmente con normal, del punto bajo, o de ninguna carga es debido a un cambio en la fórmula usada para calcular el uso. Con las 17.2 versiones, la utilización CPU se computa sobre la base del **promedio de carga del estado del sistema de la demostración** en el vEdge.

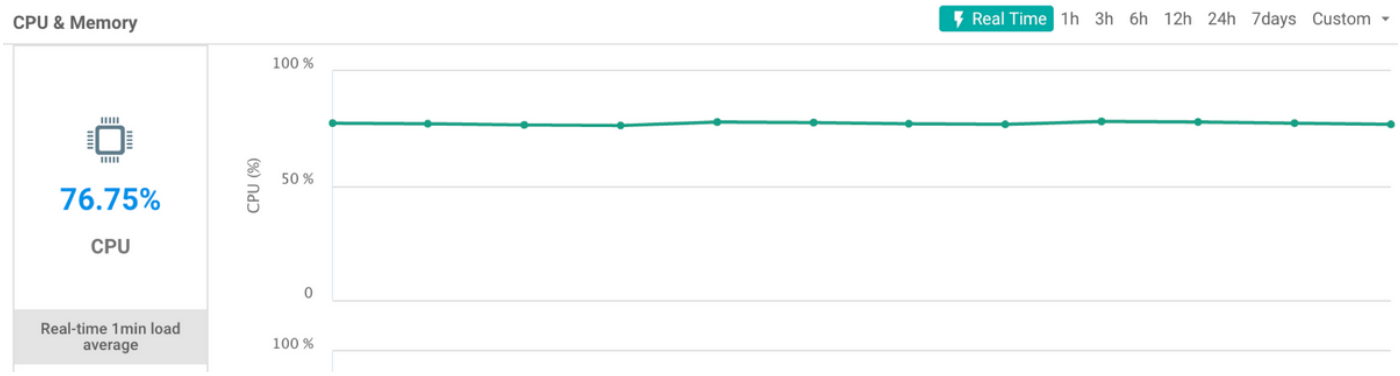
el vManage muestra la utilización en tiempo real CPU para un dispositivo. Tira de la **1 media minuciosa [min1_avg]** y de **5 minutos [min5_avg] medio** basados en los datos históricos. **El promedio de carga**, por definición, incluye las diversas cosas y no apenas los ciclos CPU que contribuyen al cálculo de la utilización. Por ejemplo, tiempo de espera IO, de proceso hasta que finalice el tiempo, y otros valores se consideran cuando usted presenta este valor para la plataforma. En este caso, usted ignora los valores mostrados para los estados CPU y los valores CPU en el **comando top del** vShell.

Aquí está un ejemplo en cómo la utilización CPU, que es realmente el **1 promedio de carga minucioso**, consigue calculada y mostrada en el panel del vManage:

Cuando usted controla la carga de un vEdge CLI, esto puede ser vista:

```
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
```

En este caso, la utilización CPU se computa sobre la base del **promedio de carga/# de las memorias (vCPUs)**. Por este ejemplo, el nodo tiene 4 memorias. El promedio de carga entonces es convertido por un factor de 100 antes de que usted divida por el número de memorias. Cuando usted hace un promedio del promedio de carga de todas las memorias y se multiplica por 100, usted llega en un valor de ~310. Tome este valor y la divisoria por 4 producciones, una lectura CPU de la CPU 77.5%, que alinea con el valor visto en el gráfico en tiempo real en el vManage capturado alrededor del tiempo la salida CLI fue recogida y tal y como se muestra en de la imagen.



Para considerar los promedios de carga y el número de memorias CPU en el sistema, la salida del **top** se puede consultar del vShell en el dispositivo.

En el ejemplo abajo, el vEdge contiene 4 vCPUs. La primera base (**Cpu0**) se utiliza para el **control** (véase a través la utilización más baja del usuario) mientras que siguen habiendo las 3 memorias se utilizan para los **datos**:

```
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
```

Para conseguir el número de CPU del vEdge CLI, este comando puede ser utilizado:

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

Otro ejemplo del cálculo para el valor mostrado en el vManage en el vEdge 1000 se proporciona

aquí. Después de que usted publique el **top del** vShell, **l** está interesado para visualizar la carga para todas las memorias:

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

Puesto que un vEdge 1000 tiene solamente una base CPU disponible, la carga señalada aquí es el 55% (0.55*100).

CPU elevada uso con punto de congelación-um el proceso

Usted puede también notar a veces del **top** que punto de congelación-**um el** proceso funciona con arriba y muestra la CPU del hasta 100%. Esto se espera en las memorias CPU que se utilizan para el proceso plano de los datos.

Del comando top referido anterior, 3 memorias actúan en la CPU del 100% y 1 base muestra el uso normal:

```
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
...
```

Esta primera base (Cpu0) se utiliza para el **control** y las tres memorias restantes usadas para los **datos**. Como usted puede ver en la lista de procesos, punto de congelación-**um el** proceso utiliza esos recursos.

punto de congelación-um es un proceso que utiliza un driver encuesta-MODE, así que significa que sienta y sondea el puerto subyacente para los paquetes constantemente de modo que pueda procesar cualquier marco tan pronto como se reciba. Este proceso maneja la expedición y es equivalente a la expedición del trayecto rápido en el vEdge 1000, el vEdge 2000, y el vEdge 100. Esta arquitectura encuesta-MODE es utilizada por Intel para el proceso del paquete eficiente basado en el marco DPDK. Porque el reenvío de paquete se ejecuta en un loop apretado, la CPU permanece en o cerca del 100% siempre. Aunque se haga esto, no se introduce ningún tiempo de espera a través de estas CPU pues ésta es conducta esperada.

La información previa en la interrogación DPDK puede él encontró aquí.

La nube del vEdge y las Plataformas del vEdge 5000 utilizan la misma arquitectura de la expedición y exhiben el mismo comportamiento a este respecto. Aquí está un ejemplo de un vEdge 5000 tirado de la salida **superior**. Tiene 28 memorias, cuyo 2 (Cpu0 y Cpu1) se utilizan para el **control** (como el vEdge 2000) y 26 se utilizan para los **datos**.

```
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
```

Aquí, el promedio de carga es siempre alto porque 26 fuera de los 28 procesadores se ejecutan en el 100% debido punto de congelación-**um al** proceso.

Conclusión

El uso señalado CPU en el vManage para las versiones 17.2.x antes de 17.2.7 no es el uso real CPU sino en lugar de otro se calcula sobre la base del promedio de carga. Esto puede llevar a la confusión en la comprensión del valor señalado y llevar a las alarmas falsas relacionadas CPU elevada mientras que la plataforma actúa normalmente con normal, al punto bajo, o a ningún tráfico/carga de la red reales.

Este comportamiento se cambia/se modifica con los 17.2.7 y los 18.2 release/versión tales que la lectura CPU puede ahora ser exacta basada en la lectura del cpu_user del **top**.

El problema también se menciona en los 17.2 Release Note.