Entender a alta utilização da CPU relatada pelo vManage para plataformas de nuvem vEdge 5000/2000/1000/100B e vEdge

Contents

Introduction

Este documento descreve por que você pode ver o alto uso da CPU relatado nas plataformas vManage para vEdge 5000/2000/1000/100B e vEdge Cloud, apesar do desempenho das plataformas estar normal sem que a CPU esteja alta e seja relatada como vista na **parte superior**.

Entender a alta utilização da CPU relatada nas plataformas de nuvem vEdge 5000/2000/1000/100B e vEdge

Com as versões 17.2.x e posteriores, é possível observar um maior consumo de CPU e memória para plataformas vEdge e vEdge Cloud. Isso é observado no painel do vManage para um determinado dispositivo. Em alguns casos, isso também leva a um número maior de alertas e avisos no vManage.

Explicação

O motivo para o alto uso da CPU relatado quando o dispositivo executa normalmente com carga normal, baixa ou sem carga é devido a uma alteração na fórmula usada para calcular o uso. Com as versões 17.2, a utilização da CPU é calculada com base na **Média de carga** do **show system status** no vEdge.

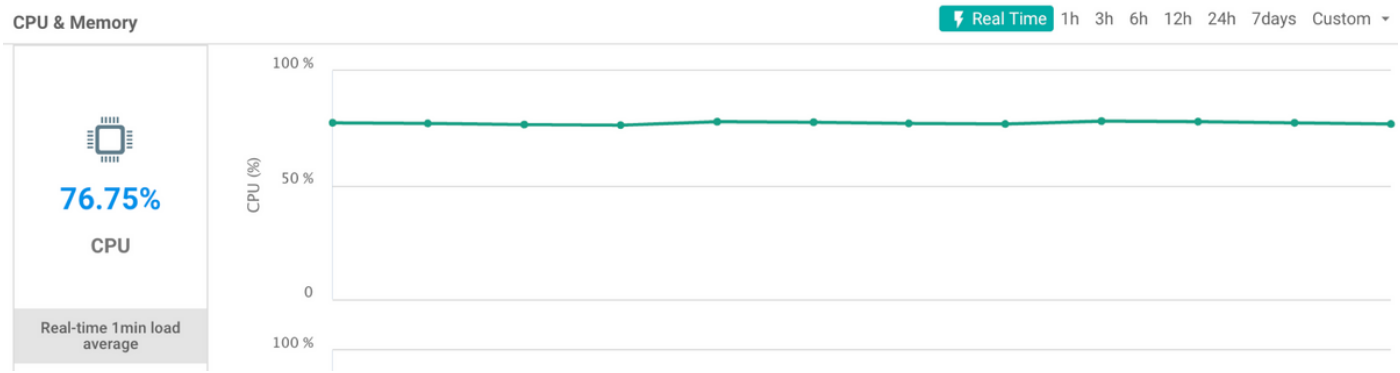
O vManage mostra a utilização da CPU em tempo real para um dispositivo. Ele puxa a **Média de 1 minuto [min1_avg]** e **Média de 5 minutos [min5_avg]** com base nos dados históricos. **Carga média**, por definição, inclui várias coisas e não apenas ciclos de CPU que contribuem para o cálculo de utilização. Por exemplo, o tempo de espera de E/S, o tempo pendente do processo e outros valores são considerados quando você apresenta esse valor para a plataforma. Nesse caso, você ignora os valores mostrados para os estados da CPU e os valores da CPU no comando **superior** do vShell.

Aqui está um exemplo de como a utilização da CPU, que é na verdade a **média de carga de 1 minuto**, é calculada e mostrada no painel do vManage:

Quando você verifica a carga de uma CLI vEdge, isso pode ser visto:

```
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
```

Nesse caso, a utilização da CPU é calculada com base na **carga média / número de núcleos (vCPUs)**. Para este exemplo, o nó tem 4 núcleos. A média de carga é então convertida por um fator de 100 antes que você divida pelo número de núcleos. Ao calcular a média de carga de todos os núcleos e multiplicar por 100, você obtém um valor de ~310. Pegue esse valor e divida por 4 rendimentos, uma leitura da CPU de 77,5% da CPU, que se alinha ao valor visto no gráfico em tempo real no vManage capturado durante o tempo em que a saída da CLI foi coletada e como mostrado na imagem.



Para ver as médias de carga e o número de núcleos da CPU no sistema, a saída da **parte superior** pode ser consultada do vShell no dispositivo.

No exemplo aqui, o vEdge contém 4 vCPUs. O primeiro núcleo (**Cpu0**) é usado para **Controle** (visto através da menor utilização pelo usuário) enquanto os 3 núcleos restantes são usados para **Dados**:

```
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 obter o número de CPUs da CLI do vEdge, esse comando pode ser usado:

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

Outro exemplo do cálculo do valor mostrado no vManage no vEdge 1000 é fornecido aqui. Depois de emitir o **top** do vShell, **eu** estou interessado em exibir a carga para todos os núcleos:

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

Como um vEdge 1000 tem apenas um núcleo de CPU disponível, a carga relatada aqui é 55% (0,55*100).

Alto uso da CPU com processo fp-um

Às vezes, você também pode notar de **cima** que o processo **fp-um** é executado em alta e mostra até 100% de CPU. Isso é esperado nos núcleos da CPU usados para o processamento do plano de dados.

Do comando **superior** mencionado anteriormente, 3 núcleos operam em 100% de CPU e 1 núcleo mostra a utilização 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
...
```

Este primeiro núcleo (Cpu0) é usado para **Controle** e os três núcleos restantes são usados para **Dados**. Como você pode ver na lista de processos, o processo **fp-um** usa esses recursos.

fp-um é um processo que usa um driver de modo de pesquisa, o que significa que ele senta e pesquisa a porta subjacente de pacotes constantemente para que ele possa processar qualquer quadro assim que for recebido. Esse processo trata do encaminhamento e é equivalente ao encaminhamento de caminho rápido no vEdge 1000, vEdge 2000 e vEdge 100. Esta arquitetura de modo poll é usada pela Intel para um processamento eficiente de pacotes baseado na estrutura DPDK (Data Plane Development Kit). Como o encaminhamento de pacotes é implementado em um loop apertado, a CPU permanece em ou perto de 100% o tempo todo. Embora isso seja feito, nenhuma latência é introduzida através dessas CPUs, pois esse é o comportamento esperado.

Informações de fundo sobre a pesquisa do DPDK podem ser encontradas aqui.

As plataformas vEdge Cloud e vEdge 5000 usam a mesma arquitetura de encaminhamento e exibem o mesmo comportamento nesse sentido. Aqui está um exemplo de um vEdge 5000 extraído da saída **superior**. Ele tem 28 núcleos, dos quais 2 (Cpu0 e Cpu1) são usados para o **Controle** (como o vEdge 2000) e 26 são usados para **Dados**.

```
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
```

Aqui, a Média de Carga é sempre alta porque 26 dos 28 processadores são executados a 100% devido ao processo **fp-um**.

Conclusão

O uso relatado da CPU no vManage para versões 17.2.x anteriores a 17.2.7 não é o uso real da CPU, mas sim calculado com base na média de carga. Isso pode levar a confusão na compreensão do valor relatado e levar a alarmes falsos relacionados à alta CPU enquanto a plataforma opera normalmente com carga normal, baixa ou sem carga real de tráfego/rede.

Esse comportamento é alterado/modificado com as versões 17.2.7 e 18.2, de modo que a leitura da CPU agora pode ser precisa com base na leitura de cpu_user de **cima**.

O problema também é mencionado nas notas de versão 17.2.