Begrijp het gebruik van hoge CPU's dat door vManager wordt gemeld voor vEdge 500/2000/1000/100B- en vEdge-cloudplatforms

Inhoud

Inleiding

Dit document beschrijft waarom u een hoog CPU-gebruik dat in vManager wordt gerapporteerd, kunt zien voor vEdge 500/2000/1000/100B- en vEdge-cloudplatforms, ondanks dat de prestaties van de platforms normaal zijn zonder dat er een hoge CPU wordt gerapporteerd, zoals **bovenaan** wordt weergegeven.

Begrijp het gebruik van hoge CPU's dat wordt gemeld op vEdge 500/2000/1000/100B- en vEdge-cloudplatforms

Met de releases van 17.2.x en later, kan een hoger CPU- en geheugenverbruik voor vEdge en vEdge-cloudplatforms worden waargenomen. Dit wordt op het vManager-dashboard voor een bepaald apparaat opgemerkt. In sommige gevallen leidt dit ook tot een groter aantal waarschuwingen en waarschuwingen in vManager.

verklaring

De reden voor het gerapporteerde gebruik van hoge CPU's wanneer het apparaat normaal, laag of geen lading presteert, is het gevolg van een verandering in de formule die wordt gebruikt om het gebruik te berekenen. Bij de releases van 17.2 wordt het gebruik van CPU's berekend op basis van het **taakgemiddelde** van de **systeemstatus** op de vEdge.

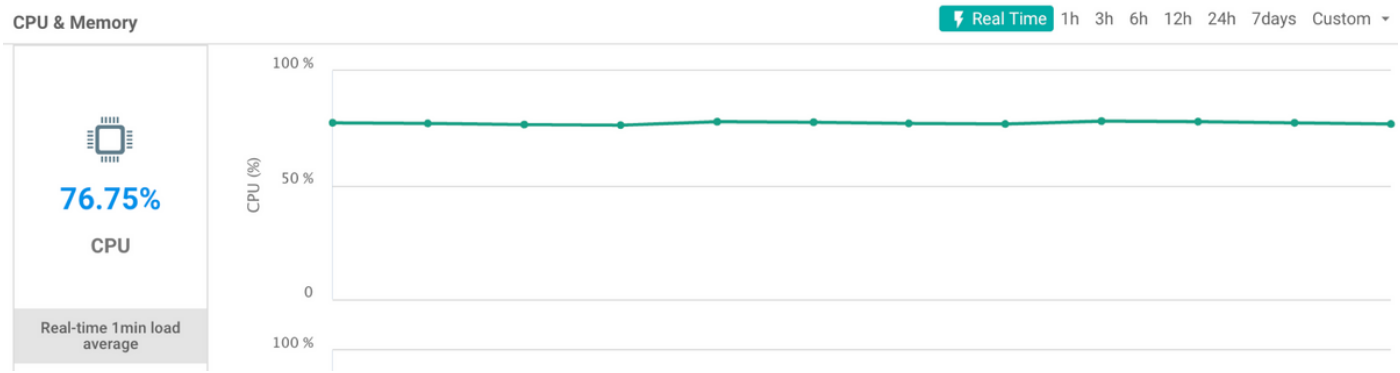
vManager toont realtime CPU-gebruik voor een apparaat. Het **gemiddelde van 1 minuten [min1_avg]** en het **gemiddelde van 5 minuten [min5_avg]** wordt op basis van historische gegevens **vernietigd**. **Laadgemiddelde** omvat per definitie verschillende dingen en niet alleen CPU-cycli die bijdragen aan de gebruiksberekening. Bijvoorbeeld, IO wachttijd, proces hangende tijd, en andere waarden worden overwogen wanneer u deze waarde voor het platform voorstelt. In dit geval, negeert u de waarden die voor de CPU-staten en CPU-waarden in het **bovenste** commando van vShell worden getoond.

Hier is een voorbeeld van hoe het gebruik van CPU's, dat in feite het **laadgemiddelde van 1 minuut** is, wordt berekend en weergegeven in het vManager-dashboard:

Wanneer u een vEdge CLI-functie controleert, is dit te zien:

```
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
```

In dit geval wordt het CPU-gebruik berekend op basis **van het taakgemiddelde / # cores (vCPU's)**. Dit voorbeeld heeft het knooppunt vier kernen. Het load-gemiddelde wordt dan geconverteerd met een factor 100 voordat u zich verdeelt door het aantal cores. Wanneer u het gemiddelde van de lading uit alle kernen gemiddelde en vermenigvuldigt met 100, arriveert u met een waarde van ~310. Neem deze waarde en verdeel door 4 rendementen, een CPU-aflezing van 77,5% CPU, die zich uitstrekt met de waarde in het real-time grafiek in vManager de tijd dat de CLI-uitvoer werd verzameld en zoals in de afbeelding wordt weergegeven.



Om de laadgemiddelden en het aantal CPU-kernen in het systeem te zien, kan de **top**-uitgang worden geraadpleegd door vShell op het apparaat.

In het voorbeeld hier bevat de vEdge 4 vCPU's. De eerste kern (**Cpu0**) wordt gebruikt voor **controle** (gezien door de lagere gebruikersbenutting) terwijl de overige 3 kernen gebruikt worden voor **gegevens**:

```
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
```

Om het aantal CPU's in de vEdge CLI te krijgen, kan deze opdracht worden gebruikt:

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

Hier wordt een ander voorbeeld gegeven van de berekening van de waarde in vManager op vEdge 1000. Nadat u **boven** vShell hebt uitgereikt, is **I** geïnteresseerd in het weergeven van de

lading voor alle kernen:

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

Aangezien een vEdge 1000 slechts één CPU-kern beschikbaar heeft, is de hier gerapporteerde lading 55% (0,55*100).

Gebruik van hoge CPU's met fp-um-proces

U kunt ook van **bovenaf** soms opmerken dat het **fp-um**-proces hoog is en tot 100% CPU's bevat. Dit wordt verwacht op de CPU-cores die worden gebruikt voor de verwerking van datacenters.

Van de hierboven genoemde **top** opdracht werken 3 kernen bij 100% CPU en 1 kern toont normaal gebruik:

```
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
...
```

Deze eerste kern (CPU0) wordt gebruikt voor **Controle** en de drie resterende kernen die voor **Gegevens** worden gebruikt. Zoals u in de proceslijst kunt zien, gebruikt het **kp-um**-proces deze middelen.

fp-um is een proces dat gebruik maakt van een poll-mode chauffeur, wat betekent dat het de onderliggende poort voor pakketten constant plaatst en peilt zodat het elk frame kan verwerken zodra het ontvangen is. Dit proces verwerkt het doorsturen en is gelijk aan snel-pad doorsturen in de vEdge 1000, vEdge 2000, en vEdge 100. Deze poll Mode architectuur wordt gebruikt door Intel voor efficiënte pakketverwerking op basis van Data Plane Development Kit (DPDK)-framework. Omdat pakkettransport in een strakke lus wordt geïmplementeerd, blijft CPU te allen tijde op 100% of dicht bij 100%. Hoewel dit gebeurt, wordt er geen latentie door deze CPU's geïntroduceerd, omdat dit verwacht gedrag.

Achtergrondinformatie over de DPDK-enquête is hier te vinden.

De vEdge Cloud- en vEdge 5000-platforms maken gebruik van dezelfde verzendende architectuur en vertonen hetzelfde gedrag in dit opzicht. Hier is een voorbeeld van een vEdge 5000 die uit de **bovenste** uitvoer is getrokken. Hij heeft 28 kernen, waarvan 2 (Cpu0 en Cpu1) worden gebruikt voor **controle** (zoals de vEdge 2000) en 26 voor **Gegevens**.

```
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
```

Op dit punt is het laadgemiddelde altijd hoog, omdat 26 van de 28 processors 100% hebben dankzij het **kp-um** proces.

Conclusie

Het gerapporteerde CPU-gebruik in vManager voor 17.2.x-releases vóór 17.2.7 is niet het werkelijke CPU-gebruik maar wordt in plaats daarvan berekend op basis van het taakgemiddelde. Dit kan leiden tot verwarring bij het begrip van de gerapporteerde waarde en kan leiden tot valse alarmsignalen gerelateerd aan hoge CPU's terwijl het platform normaal gesproken werkt met een lage of geen feitelijke belasting van verkeer of netwerk.

Dit gedrag wordt gewijzigd/aangepast met de releases van 17.2.7 en 18.2 zodat het lezen van de CPU nu nauwkeurig kan zijn gebaseerd op het **vanaf** de **top** lezen van cpu_user.

De kwestie wordt ook genoemd in de Releaseopmerkingen van 17.2.