使用CPS解決高記憶體利用率問題

目錄

簡介

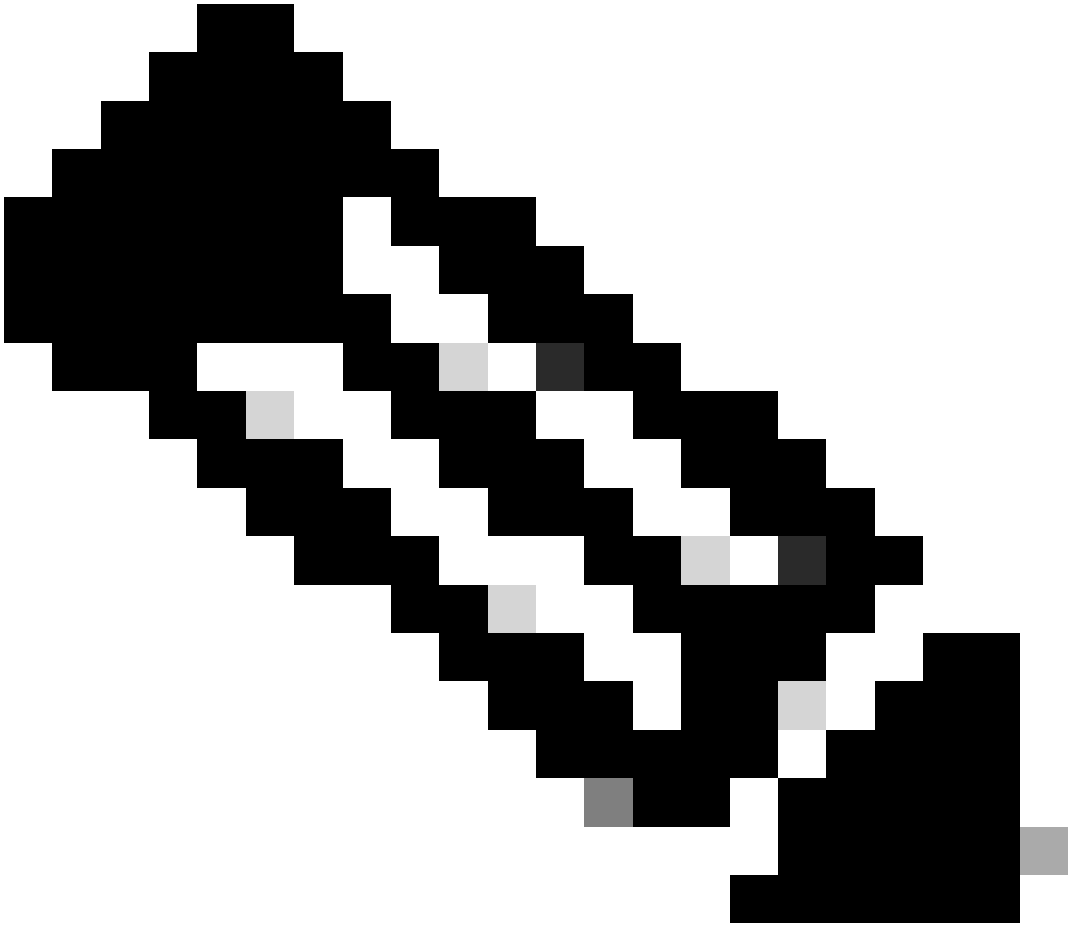
本文描述對Cisco Policy Suite (CPS)的高記憶體利用率問題進行故障排除的過程。

必要條件

需求

思科建議您瞭解以下主題：

- Linux
- CPS
- MongoDB



注意：Cisco建議對CPS CLI具有超級使用者訪問許可權。

採用元件

本文中的資訊係根據以下軟體和硬體版本：

- CPS 20.2
- 整合運算系統(UCS)-B
- MongoDB v3.6.17

本文中的資訊是根據特定實驗室環境內的裝置所建立。文中使用到的所有裝置皆從已清除（預設）的組態來啟動。如果您的網路運作中，請確保您瞭解任何指令可能造成的影響。

背景資訊

Linux擁有廣泛的工具來支援、管理、監控和部署軟體應用程式。

增加到產品應用程式的服務和功能可能會佔用大量記憶體。針對Linux伺服器的記憶體最佳化，不但讓應用程式的執行更順暢、快速，也減少了資料遺失和伺服器當機的風險。

若要為Linux機器最佳化記憶體，您首先需要瞭解記憶體在Linux中的運作方式。您首先從一些記憶體術語開始，討論Linux如何處理記憶體，然後學習如何排除記憶體故障並防止記憶體問題。

一台電腦可包含的記憶體總量取決於作業系統的體系結構。

Linux中的整個記憶體稱為虛擬記憶體—它包括實體記憶體（通常稱為RAM -隨機訪問記憶體）和交換空間。除非增加更多RAM，否則無法增加系統的實體記憶體。不過，虛擬記憶體可以透過使用硬碟的交換空間來增加。

RAM會決定您的電腦是否能處理高記憶體耗用的程式。

來自使用者、電腦進程和硬碟驅動器(HDD)的資料被傳送到RAM。如果需要，RAM會將其儲存並傳送回使用者或HDD。如果資料需要永續性，RAM會將其傳送到中央處理器(CPU)。

若要檢查電腦中是否有可用的可用空間，您可以使用free指令。

```
[root@installer ~]# free -h
total used free shared buff/cache available
Mem: 11Gi 1.3Gi 2.9Gi 105Mi 7.4Gi 10Gi
Swap: 0B 0B 0B
[root@installer ~]#
```

問題

Linux伺服器會因各種原因消耗大量記憶體。為了快速有效地進行故障排除，首先，您需要排除最可能的原因。

Java程式：

有數個應用程式是使用Java來實作，其不正確的實作或組態可能會導致伺服器中的記憶體使用率高。最常見的兩個原因是快取和會話快取反模式中的配置錯誤。

快取記憶體是應用程式達到高效能的一種常見方式，但如果套用不正確，最終可能會損害系統效能。錯誤的配置可能會使快取成長過快，並且為系統中運行的其他進程留下更少的記憶體。

儲存應用程式的中間狀態時，通常會使用階段作業快取。它允許開發人員按會話儲存使用者，並便於儲存或獲取資料對象值。但是，開發人員往往忘記在以後清除會話快取資料。

使用Java中的資料庫時，通常會使用休眠階段作業來建立連線，並管理伺服器與資料庫之間的階段作業。但是當開發人員使用休眠工作階段時，經常會發生錯誤。休眠工作階段不是為了執行緒安全而隔離，而是包含在相同的超文字傳輸通訊協定(HTTP)工作階段中。這使得應用程式儲存的狀態多於必要，而且只有少數使用者，記憶體使用量會大幅增加。

資料庫：

當討論高記憶體消耗過程時，必須提及資料庫。應用程式在處理使用者請求時，對資料庫進行了多次讀取和寫入，因此我們的資料庫可能會佔用大量的記憶體。

以MongoDB資料庫為例：為了實現高效能，它應用了一個緩衝機制來快取和索引資料。如果您將資料庫設定成在您對資料庫有數個要求時使用最大記憶體，Linux伺服器中的記憶體很快就會被淹沒。

CPS記憶體消耗可以透過使用Grafana圖中的適當KPI或其他工具進行監控。如果任何CPS虛擬機器(VM)上的記憶體消耗增加超過預設閾值90%，CPS可能會為該虛擬機器生成記憶體不足警報。此閾值可在CPS部署模板中使用free_mem_per設定進行配置。

辨識造成高記憶體使用率的程式/公用程式：

1. 登入已引發記憶體不足警報的虛擬機器。

2. 瀏覽至/var/log目錄，並存回**top_memory_consuming_processes**檔案，以辨識記憶體使用率較高的「處理ID (PID)」。

```
********** Date: Tue May 16 05:06:01 UTC 2023 **********
PID PPID CMD %MEM %CPU RSS PRI STAT PSR WCHAN NI P
9435 1 /usr/bin/java -XX:OnOutOfMe 26.7 77.9 4353796 5 S<l 2 - -15 *
24139 1 /usr/java/default/bin/java 1.0 0.0 174636 20 Sl 3 - 0 *
2905 2862 /usr/sbin/collectd -C /etc/ 1.0 0.2 169104 20 Sl 1 hrtimer_nanosl 0 *
913 1 /usr/lib/systemd/systemd-jo 0.4 0.1 69364 20 Ss 5 do_epoll_wait 0 *
1513 1 /usr/libexec/platform-pytho 0.1 0.0 27912 20 Ssl 5 - 0 *
3379 2905 /usr/sbin/collectd -C /etc/ 0.1 0.0 23716 20 Sl 3 - 0 *
3377 2905 /usr/sbin/collectd -C /etc/ 0.1 0.0 23712 20 Sl 4 - 0 *
3378 2905 /usr/sbin/collectd -C /etc/ 0.1 0.0 23712 20 Sl 5 - 0 *
3380 2905 /usr/sbin/collectd -C /etc/ 0.1 0.0 23712 20 Sl 5 - 0 *
********** END **********
```

3. 使用此命令驗證流程，無論它是應用程式流程還是資料庫流程。

<#root>

#ps -ef | grep <PID>

解決CPS高記憶體使用率問題的程式

在Linux中最佳化記憶體是複雜的，修復超載記憶體需要大量的工作。

方法1.

偵測並回收快取的記憶體：

在某些情況下，低記憶體警示可能是Linux記憶體管理在快取中配置物件的結果。

評估VM已快取的記憶體量，並觸發Linux釋放部分快取記憶體。

1. 比較兩個或多個CPS虛擬機器上快取的記憶體量，以便在每個VM上運行free -m命令。

[root@dc1-qns01 ~]# free -m
total used free shared buff/cache available
Mem: 15876 5262 4396 808 6217 9628
Swap: 4095 0 4095
[root@dc1-qns01 ~]#

2. 若要回收部分非使用中的快取記憶體，請執行此命令。

#free && sync && echo 3 > /proc/sys/vm/drop_caches && echo "" && free

[root@dc1-qns01 ~]# free -m
total used free shared buff/cache available
Mem: 15876 5016 8782 872 2076 9809
Swap: 4095 0 4095
[root@dc1-qns01 ~]#

請注意：
1. 此命令丟棄可能導致輸入輸出(IO)和中央處理器(CPU)使用率臨時增加的快取對象，因此建議在非高峰時間/維護時段運行此命令。
2. 這是一個非破壞性命令，並且只有未使用的空閒記憶體。

如果記憶體不足警報仍未解決，則繼續執行方法2。

方法2.

如果高記憶體消耗是由任何應用程式進程（如QNS等）造成的。

1. 重新啟動處理。

<#root>

Command Syntax:

#monit restart <process name>

2. 透過free-m命令驗證記憶體使用率是否下降。

如果記憶體不足警報仍未解決，則繼續執行方法3。

方法3.

重新啟動已生成警報的虛擬機器，因為通常重新啟動虛擬機器是為了增加虛擬機器的資源（磁碟記憶體CPU）。

如果發現sessionmgr VM的記憶體利用率很高，則繼續執行方法4。

方法4.

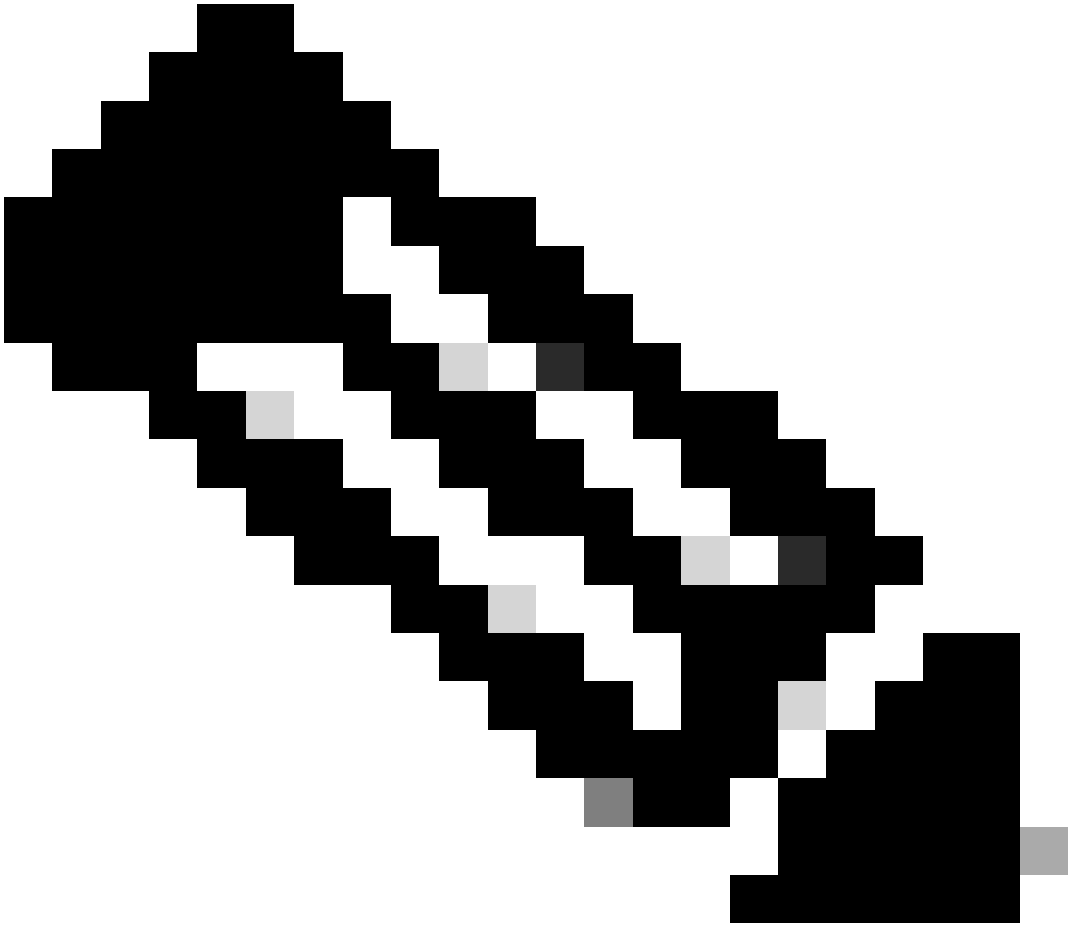
1. 登入已注意到高記憶體使用率的VM。

2. 切換作業選項至/var/log目錄，並存回mongodb-<xxxx>.log檔案，以取得與記憶體消耗和writeConcernMajorityJournalDefault引數相關的警告/訊息。

```
2022-12-13T00:30:39.012+0200 I REPL [replexec-0] ** WARNING: This replica set node is running without journaling enabled but the
2022-12-13T00:30:39.012+0200 I REPL [replexec-0] ** writeConcernMajorityJournalDefault option to the replica set config
2022-12-13T00:30:39.012+0200 I REPL [replexec-0] ** is set to true. The writeConcernMajorityJournalDefault
2022-12-13T00:30:39.012+0200 I REPL [replexec-0] ** option to the replica set config must be set to false
2022-12-13T00:30:39.012+0200 I REPL [replexec-0] ** or w:majority write concerns will never complete.
2022-12-13T00:30:39.012+0200 I REPL [replexec-0] ** In addition, this node's memory consumption may increase until all
2022-12-13T00:30:39.012+0200 I REPL [replexec-0] ** available free RAM is exhausted.
```

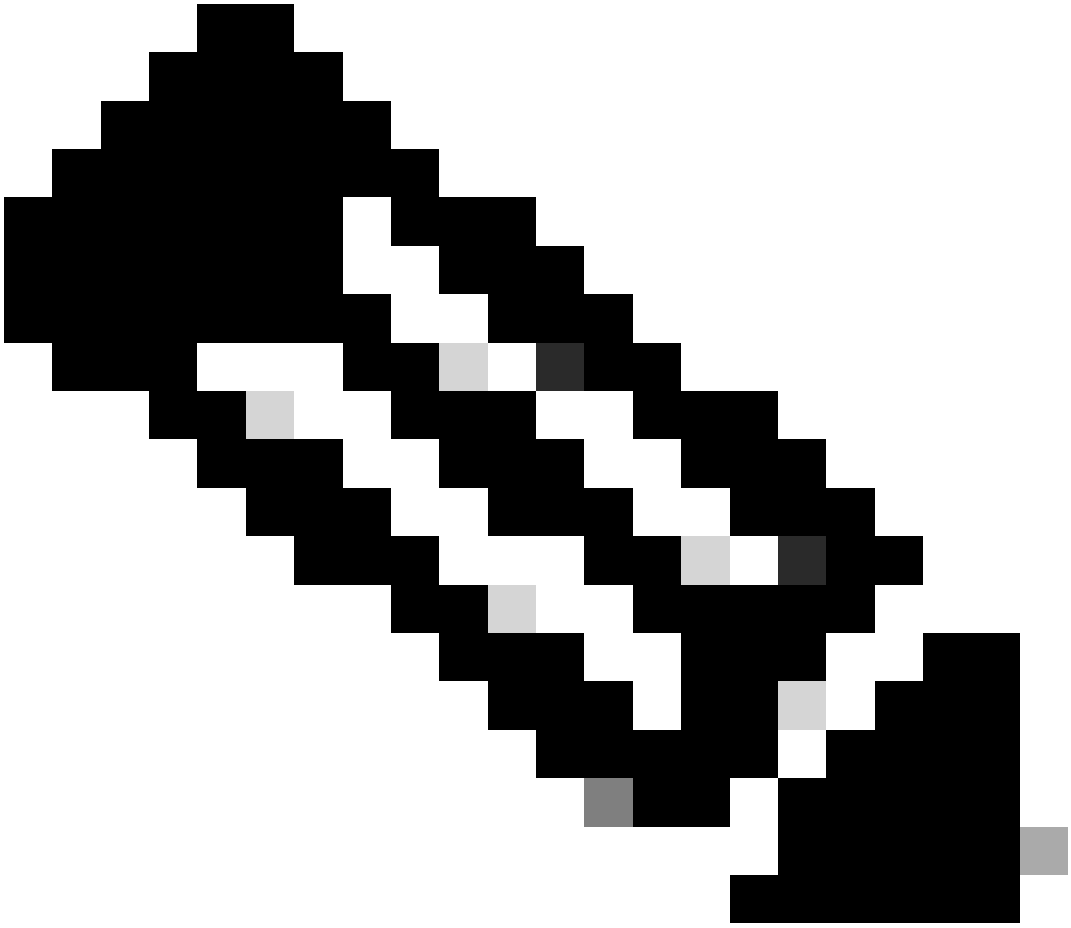
3. 登入相應的mongoShell，驗證mongo protocolVersion和writeConcernMajorityJournalDefault的當前值。

```
set04:PRIMARY> rs.status().optimes.lastCommittedOpTime.t
NumberLong(0)
set04:PRIMARY>
```



NumberLong **注意**：mongo protocol version 0的o/p中始終為負值。

```
set04:PRIMARY> rs.conf().writeConcernMajorityJournalDefault
set04:PRIMARY>
```



注意：如果輸出未傳回none，則您必須將writeConcernMajorityJournalDefault值預設設為true。

4. 如果protocolVersion是1，writeConcernMajorityJournalDefault值為true，則從各自的mongoShell運行這些命令，將writeConcernMajorityJournalDefault值修改為 false.

```
#cfg=rs.conf()
#cfg.writeConcernMajorityJournalDefault=false
#rs.reconfig(cfg)
```

5. 驗證writeConcernMajorityJournalDefault值是否已更改為false。

```
set03:PRIMARY> rs.conf().writeConcernMajorityJournalDefault
false
set03:PRIMARY>
```

6. 透過free-m命令驗證記憶體使用率是否減少。

關於此翻譯

思科已使用電腦和人工技術翻譯本文件，讓全世界的使用者能夠以自己的語言理解支援內容。請注意，即使是最佳機器翻譯，也不如專業譯者翻譯的內容準確。Cisco Systems, Inc. 對這些翻譯的準確度概不負責，並建議一律查看原始英文文件（提供連結）。