Memory-Verwaltung unter Linux - Mythen aus der Praxis

Thorsten Bruhns
OPITZ CONSULTING Deutschland GmbH
Bad Homburg

Schlüsselworte
Oracle Linux, DBA, Speicherverwaltung

Einleitung

Die Speicherverwaltung unter Linux ist für viele Administratoren ein Albtraum, da unklar ist, auf was denn genau zu achten ist. Darüber hinaus haben sich über die Jahren Mythen entwickelt, die falsch sind und in der Folge zu Fehlentscheidungen führen.

Dieser Vortrag wird sich mit diesem Thema auseinander setzen und von Grund auf zeigen, wie und - vor allem - mit welchen Tools, eine Hauptspeicherauswertung möglich ist. Darüber hinaus werden HugePages behandelt, da diese zunehmend wichtiger werden, weil die Hauptspeichergrenzen stetig weiter zunehmen.

Memoryinterfache

An den Mythen ist das Memoryinterface von Linux nicht ganz unschuldig. Die Dokumentation und das Verhalten der Informationen in meminfo sind nicht einheitlich und haben sich im Laufe der Jahre verändert. Zudem wurden wichtige Informationen erst in neuen Kernelversionen verfügbar, was eine Auswertung deutlich komplizierter macht, weil zusätzliche Tools genutzt werden müssen, um alle notwendigen Informationen ermitteln zu können.

Der Vortrag wird einen kleinen Einblick in das Memoryinterface geben, um Begriffe und das Verhalten von hugePages aufzeigen zu können.

Eine Überwachung des Hauptspeichers sollte eigentlich üblich sein, erfolgt in sehr vielen Umgebungen aber völlig falsch, weil entweder der Hauptspeicher oder Swap betrachtet werden, was praktisch sinnlos ist, da beide Komponenten voneinander abhängig sind und somit eine gemeinsame Überwachung erfordern.

Tools

In der Praxis ist häufig zu beobachten, wie mit Hilfe von top bzw. free eine Analyse der Speicherauslastung erfolgt. Hier wird leider übersehen, das insbesondere beim Betrieb von Datenbankservern mit Oracle der Shared-Memory üblicherweise – jedoch nicht immer - Bestandteil von cached ist, was schnell zu Fehlinterpretationen führt

```
linux01:~ # free
             total       used       free     shared    buffers     cached
Mem:       3928228    3859076      69152          0      26896    2790776
-/+ buffers/cache:    1041404    2886824
Swap:      4192956     506816    3686140
```

Aus der Anzeige können keinerlei Rückschlüsse auf den Shared-Memory oder Probleme durch Swapping wegen Speichermangel gezogen werden. Für beide Betrachtungen ist eine tiefere Kenntnis über die Speicherverwaltung von Linux notwendig.

Die Bedeutung von total, used und free wird ebenfalls gerne falsch interpretiert. Das Betriebssystem wird in Abhängigkeit von total versuchen, immer etwas freien Speicher zu halten, damit neu gestartete Prozesse beim Anfordern von Speicher, möglichst schnell eine Antwort erhalten. Es kann somit durchaus möglich sein, dass free nahe 0 ist und erst nach einigen Sekunden wieder den ursprünglichen Wert annimmt, da der Kernel im Hintergrund dafür sorgen wird, möglichst immer etwas freien Speicher zu reservieren. Der Bereich beträgt üblicherweise 5% von total, kann bei extremen Speichermangel vom Kernel aber weiter reduziert werden. Normalerweise zeigt free nur dann große Werte an, wenn der Kernel neu gestartet oder Prozesse große Speichermenge an den Kernel zurück gegeben haben.
Weitere Tools sowie Auswertungsmöglichkeiten werden im Rahmen des Vortrages vorgestellt.

Shared-Memory und Oracle

Dieser Bereich ist insbesondere beim Betrieb von Oracle Datenbanken sehr wichtig. Hier hat sich in den letzten Jahren einiges im Kernel und auch bei Oracle verändert, was in der Praxis gerne zu Problemen führt.

Mit der Einführung von ‚memory_target' mußte beim Betrieb auf Linux ein anderes Speicherinterface genutzt werden, da es möglich ist, das Shared-Memory zur Laufzeit einer Instance an den Kernel zurück gegeben werden kann. Dies ist mit dem üblicherweise genutzten Shared-Memory unter Linux nicht möglich, da hier sehr große, zusammenhängende Speichersegmente angefordert werden.

Ärgerlich ist, das sich die Verwendung von ‚memory_target' und HugePages ausschließen, weshalb der Autor unter Linux grundsätzlich von der Nutzung von ‚memory_target' unter Linux abrät. Hiervon ausgenommen sind ASM-Instancen, da hier seitens Oracle eine klare Empfehlung für memory_target ausgesprochen wird, was unter den Rahmenbedingungen einer ASM-Instance in Ordnung ist, da diese relativ wenig Prozesse und eine vergleichsweise kleine SGA haben.

Gemein wird die Betrachtung der Shared-Memory Segmente. Je nach Kernelversion ist in meminfo eine Ausgabe von shmem verfügbar:

```
Linux01:~ # cat /proc/meminfo  | grep -i shmem
Shmem:            547532 kB
```

Sollte der Wert nicht verfügbar sein, dann muß mit ipcs und df gearbeitet werden, was insbesondere bei ipcs sehr unpraktisch ist, da hier die Memorysegmente ausgegeben und eine Berechnung des Platzbedarf vorgenommen werden muß.

df hingegen ist nur nutzbar, wenn ‚memory_target' zum Einsatz kommt.

```
Linus01:~ # df /dev/shm

Filesystem           1K-blocks      Used Available Use% Mounted on
tmpfs                  1964112    627048   1337064  32% /dev/shm
```

Aufgrund dieser Besonderheiten bei Shared-Memory-Segmenten sind nur wenige Monitoringtools in der Lage, hier eine Überwachung zu ermöglichen. Glücklicherweise ist SHmem in meminfo in aktuell üblichen Linuxdistributionen für Oracle-Datenbanken verfügbar, was eine Betrachtung in der Praxis stark vereinfacht.

Wichtig! Free zeigt je nach Version und Distribution unter ‚shared' den Shared-Memory an. Nach Erfahrungen des Autors jedoch nur, wenn /dev/shm nicht genutzt wird. Aus diesem Grund ist der Wert im obigen Beispiel 0.

HugePages

Damit allein kann ein Vortrag gefüllt werden, weshalb hier nicht vollumfänglich draus eingegangen werden kann. Es wird zunehmend wichtiger, da die Speicherverwaltung im Kernel sogenannte PageTables für ein Mapping vom physischen oder in swap verwalteten Speicher zum Adressbereich der Prozesse benötigt. Diese Verwaltung wird mit zunehmender Hauptspeichergröße immer kritischer, da der Kernel einen nennenswerten Teil des Hauptspeichers für diese Verwaltungsstrukturen reservieren muß, die mittels free nicht ersichtlich sind.

HugePages oder LargePages sind hier eine Möglichkeit, diesen Overhead zu reduzieren, erfordern jedoch eine gute Kenntnis der eingesetzten Anwendungen bzw. der entsprechenden Parametrisierung, weshalb administrativ häufig darauf verzichtet wird, weil man Angst vor dem Konfigurationsaufwand hat. Dazu kommt, das Fehlkonfigurationen dramatische Folgen für die Performance bis zum Stillstand des Betriebssystems haben können. Gemein ist, das PageTables zu nonswapable Memory gehört. Der Kernel muß diese Strukturen im Hauptspeicher halten, weshalb es so wichtig ist, hier immer drauf zu achten, das diese nicht zu groß werden. Betrachtet man nur den Speicherbedarf der Prozesse, so kann das schnell zu Speichermangel führen, da unter Extrembedingungen schon Systeme beobachtet wurden, die >20% des Hauptspeichers für PageTables reserviert haben.

HugePages können helfen, den Speicherbedarf für PageTables drastisch zu reduzieren. Zusätzlich kann der Kernel entlastet werden, weil der Zugriff auf viele Speicherseiten – bei Oracle typischerweise der Shared-Memory – viele Zugriffe auf die PageTables zur Folge hat. Werden diese möglichst klein gehalten, dann wird der Overhead für die Speicherverwaltung im Kernel reduziert,was die Performance des Gesamtsystem verbessern kann.

Wichtig ist, das nicht nur Datenbankserver sondern auch Java-Container zunehmen häufig HugePages nutzen können, was bei JVMs > 16GB erhebliche Vorteile durch Reduzierung der PageTables mit sich bringen kann.

Kontaktadresse:
Thorsten Bruhns
OPITZ CONSULTING Deutschland GmbH
Norsk-Data-Strasse 2
D-00000 Bad Homburg

Telefon: +49 (0) 6172 66260 1541

E-Mail thorsten.bruhns@opitz-consulting.com
Internet: www.opitz-consulting.de