OPITZ CONSULTING

Menschen. Innovationen. Lösungen.

Linux

Memory-Verwaltung unter Linux - Mythen aus der Praxis

Thorsten Bruhns

Solution Architect

OPITZ CONSULTING GmbH

Nürnberg, 17.11.2016

Menschen. Innovationen. Lösungen.

25 Jahre

OPITZ CONSULTING

Der Referent ☺

Thorsten Bruhns

Solution Architect

1999 – 2003 **Oracle Deutschland GmbH**

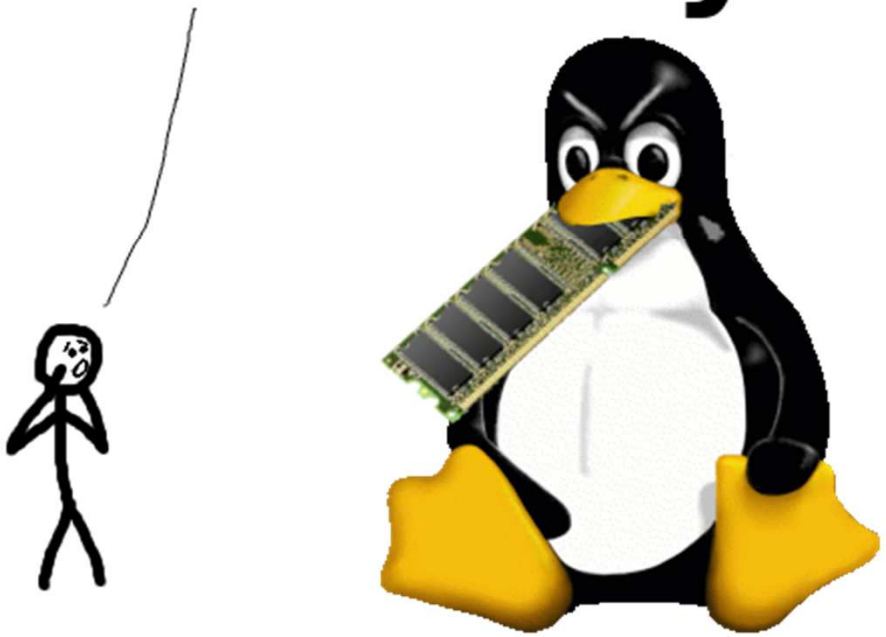
2003 – **OPITZ CONSULTING Deutschland GmbH**

Spezialist für:

Performance-Analysen und SQL-Tuning
Hochverfügbarkeit mit RAC + Data-Guard

Check_MK – Entwickler von mk_oracle

Linux ate my ram!



Quelle: http://www.linuxatemyram.com/

Agenda

1. Übersicht
2. PageTables / HugePagess
3. Swap
4. Tools
5. Fazit

1 Übersicht

Was ist Memory?

- **Irgendeine Form von Speicher**
- **Spezifikation**
  - Kapazität
  - Geschwindigkeit
  - Flüchtigkeit
- **Geschwindigkeit**
  - 1st/2nd Level Cache
    - 3 – 20 Zyklen
  - RAM
    - 250 Zyklen
  - Swap
    - > 30 Millionen Zyklen

Physical Memory

- **RAM**
  - ‚wird in Pages segmentiert
    - Typischerweise 4kB
  - Kernel reserviert einen Teil für interne Strukturen
  - Prozesse haben keinen direkten Zugriff auf Physical Memory
  - Wird vollständig vom Kernel verwaltet
- **Swap**
  - Immer gleiche Pagesize
    - HugePages nicht in swap auslagerbar
  - Ist relativ langsam
  - Systeme ohne swap sind für OOM gefährdet
  - Verwaltung durch den Kernel

Virtual Memory

- **Prozesse**
  - Nutzen ausschließlich virtuellen Speicher
  - Prozesse fordern vom Kernel Speicher an
    - Kernel verwaltet dann die notwendigen PageTables
  - Jeder Prozeß hat seine eigenen Adressbereich
    - Verwaltung erfolgt über den Kernel mit Hilfe von PageTables
    - Prozesse können gegenseitig nichts überschreiben
      - Shared-Memory ausgenommen
  - Meist anonymous pages
    - Speicher der von Prozessen angefordert wurde
    - NICHT Shared-Memory

Virtual Memory

- **Adressbereich (64bit)**
  - 256TB
    - Das dürfte noch ein paar Tage reichen – hoffentlich
- **Memory Overcommit**
  - Prozesse können mehr Speicher anfordern als physisch vorhanden
    - Anfordern bedeutet nicht zwingend belegen / nutzen.
  - Achtung bei Out of Memory (OOM)
    - Kernel kann in Out of Memory getrieben werden
      - Mehr swap kann den OOM vermeiden
    - Kernel killt dann Prozesse, damit er selbst wieder Lebensraum hat

PageCache

- **Filesystemcache**
  - kann nicht in swap ausgelagert werden
  - Jeder IO auf Dateiebene geht durch den Cache
  - Kann nicht deaktiviert werden
  - Füllt üblicherweise den freien Speicher
    - Hauptverantwortlich für sehr wenig ‚free memory'
    - Was ist eigentlich ‚free memory'?

swap

- **Swap muß immer vorhanden sein**
  - Vermeidung von out of Memory
  - System ist ohne Swap nicht schneller
    - es killt nur schneller Prozesse wegen OOM
- **Nur Memory von Prozessen kann geswappt werden**
  - HugePages, PageTables, PageCache etc sind ‚non swappable

2 PageTables / HugePages

PageTables

- **Mapping zwischen virtuellen und physischen Speicher**
- **Jeder Prozeß hat eigenen Adressbereich**
  - Jeder Prozeß hat eigenen Bereich auf der PageTable
    - Isolation zwischen Speicher der Prozesse
- **Swap von Speicher nur über PageTable möglich**
  - Achtung! HugePages können nicht ausgelagert werden!

PageTables

- **Vom Kernel verwaltet**
- **Nicht auslagerbar – non swappable**
  - War in der Praxis schon mehrfach für Swapping verantwortlich
- **Größe abhängig von Anzahl Prozesse und allokiertem virtual Memory**
- **Achtung bei Shared Memory**
  - Prozesse blenden Shared-Memory in virtuellen Speicher ein
    - Speicherbedarf kann dramatisch ansteigen
    - Oracle Datenbankserver mit PageTables >20% RAM durchaus möglich
      - Der Speicher wurde beim Sizing nie berücksichtigt….
    - Hier können HugePages helfen

HugePages / LargePages

- **Große Memorypages**
  - Reduziert Bedarf an PageTableeinträgen
  - Performancegewinn wegen reduziertem Verwaltungsaufwand auf PageTable
- **Nicht auslagerbar – non swappable**
  - Speicher wird immer im physischen RAM allokiert
  - Erfordert gutes Sizing von Systemen
- **Achtung bei Transparent HugePages**
  - Automatische Verwaltung von HugePages
  - Gemäß Alert Note: 1557478.1 für Oracle Datenbanken nicht supported!
- **Nutzung bei gro0en SGAs dringend empfohlen**

HugePages / LargePages

- **Oracle Datenbank mit /dev/shm und HugePages geht nicht**
  - => memory_target <> 0 & HugePages geht NICHT
  - DB-Server mit vielen Prozessen (>1000) möglichst immer mit HugePages
- **ASM darf ab 11.2 nicht mit HugePages betrieben werden**
  - Nutzung von memory_target klar empfohlen
    Ist kein Nachteil, da SGA und Anzahl an Prozessen klein.

3 swap

Swap – ist das gefährlich?

- **Jeder Server sollte etwas swap haben**
  - OS braucht eine Chance zur Auslagerung von Memory um OOM zu vermeiden
- **Ist swap in use schlimm?**
  - Wenn OOM droht, dann ist es gefährlich!
    - Überwachung von swap in use ist sehr fehlerbehaftet
    - Was ist, wenn ein Prozeß swap in use verursacht und dann gleich viel Speicher frei gegeben hat?
- **Swapping ist performancerelevant, wenn es ständig erfolgt**
  - free liefert hier keinerlei Hilfe…
  - top ist auch nicht wirklich besser
    - kswapd als häufig aktiver Prozeß kann Anhaltspunkte liefern

Swap – aktives Swapping erkennen

- **vmstat**
  - Nur adhoc-Betrachtung möglich
  - Auf die Spalten ‚si' (swap in) und ‚so' (swap out) achten.
  - Nur so
    - Ist ungefährlich solange kein OOM erreicht wird
    - Performanceeinfluß spürbar, meist nicht kritisch
  - Nur si
    - Nicht so schlimm wie ‚si und so' aber performancerelevant..
  - viel si und so
    - Kernel muß hier so machen, um Platz für si zu schaffen.
    - Das geht dramatisch auf die Performance des Systems
    - Typisches Anzeichen für extremen Speichermangel.
    - Führt schnell zum Systemstillstand

Swap – aktives Swapping mit vmstat erkennen

```
[root@vsdlmsilinuxtest ~]# vmstat 5


procs -----------memory---------- ---swap-- -----io---- --system-- -
----cpu-----
 r  b   swpd   free   buff  cache   si   so    bi    bo   in   cs us
sy id wa st
 3  1 1751364  57744  12416 240024 7302    0 58485  1174 5634 4876
20 17 34 29  0
 0  1 1741240  83484  12016 200420 4712    0 25458   226 3931 3425
13 12 50 25  0
 1  1 1604276  67724  11276 136440 4039 1726  4146  1805 1702 1635
1  1 74 23  0
17  4 1609444  61604   8896 115868 4506 5088  6560  5583 3295 4786
7 11 62 20  0
 0  1 1621588  55388  10176  78516 4021 4894  9727  6946 5129 12181
28 11 44 18  0
```

p

Swap – historische Betrachtung mittels sar -B

- **sar -B**
  - Liefert viel mehr Informationen – wirkt unübersichtlich
  - Erfordert sysstat.rpm
    - Ist mittlerweile in allen Distributionen verfügbar
    - Auf Oracle Datenbankservern immer vorhanden
  - Betrachtung: majflt/s
    - *Number of major faults the system has made per second, those which have required loading a memory page from disk.*
    - Große Werte meist viel Swapping
      - Performanceeinfluß spürbar
      - Nach Reboot können die Speicherseiten nicht im RAM sein. ☺

Swap – historische Betrachtung mittels sar -B

```
06:20:01 PM  pgpgin/s pgpgout/s   fault/s  majflt/s … …

06:30:01 PM      0.00    102.11  10521.92      0.00
06:40:01 PM      0.00    105.41  11493.14      0.00
06:50:01 PM      0.03     87.09   9695.53      0.00
07:00:01 PM      0.43    102.94  10501.45      0.00


07:30:01 PM    402.18    730.10  10187.43     93.83
07:40:02 PM    402.04    717.38  10246.94     95.14
07:50:02 PM    407.26    756.35   9044.68     95.51
08:00:01 PM    406.65    743.72   8348.80     96.77
```

4 Tools

free und top – hilfreiche Tools?

- **Alles bzgl. free gilt auch für top**
- **free , clear caches und nochmal free**
  - `free && sync && echo 3 > /proc/sys/vm/drop_caches && free`
  - ‚Dirty hack' für verzweifelte, wenn die Frage nach freiem RAM unklar ist
    - Kann erheblichen Performanceeinfluß haben!
  - Ergebnis:

free und top – hilfreiche Tools?

Bezeichnung	Vorher	Nachher
Total	3785376	3785376
Free	113436	1716836
Shared	2467112	1071936
Cached	2788760	1198584

- **Caches enthalten häufig Shared Memory**
  - Leider nicht immer…
- **Shared = 0 bei Nutzung von Shared Memory möglich…**
- **=> Aussagekraft von free führt schnell zu Fehlinterpretationen!**

/proc/meminfo

- **cat /proc/meminfo**
  - Memoryinformationen vom Kernel
  - Sehr viele Details – wird schnell unübersichtlich
  - Mässige bis keine Dokumentation vorhanden
  - Einzige hilfreiche Quelle für HugePages
    - `cat /proc/meminfo | grep –i huge`
  - Einfache Quelle für Shared Memory
    - Erst ab OL/RHEL6 verfügbar
      - `cat /proc/meminfo | grep –i hmem`

/proc/meminfo

```
[linuxtest~]# cat /proc/meminfo | grep -i shmem
Shmem:             951492 kB

[root@linuxtest ~]# cat /proc/meminfo | grep -i huge
HugePages_Total:       0
HugePages_Free:        0
HugePages_Rsvd:        0
HugePages_Surp:        0
Hugepagesize:       2048 kB
```

5 Fazit

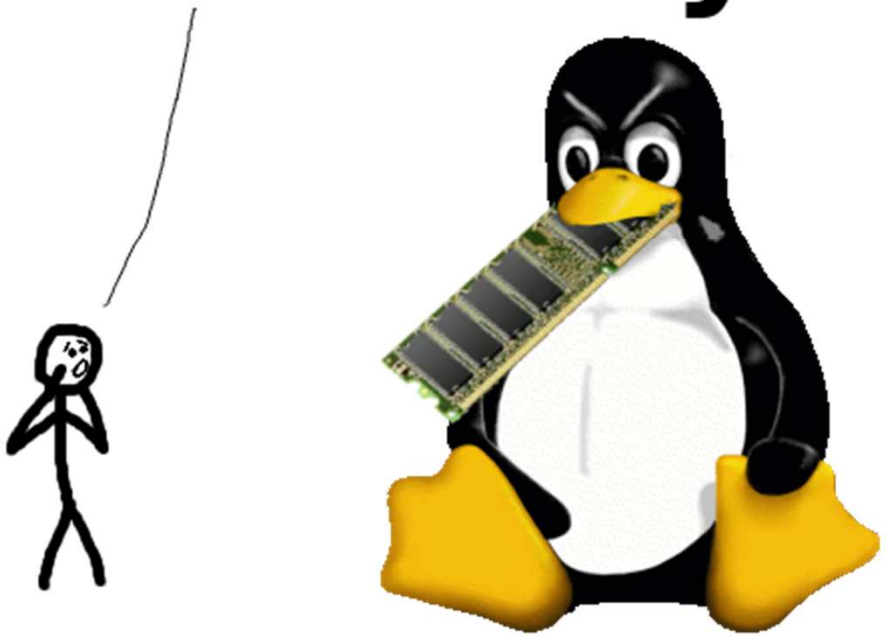
Fazit Speicherbereiche

- **HugePages**
  - Ermöglichen viel Speicherersparnis in PageTables
  - Können aufwändig in Sizing und Konfiguration sein
- **PageTables**
  - Dürfen nicht aus den Augen verloren werden
- **Swapping**
  - Ist relativ einfach identifizierbar
  - Sollte niemals deaktiviert werden

Fazit Tools

- **free und top**
  - Sind hilfreich aber sehr ungenau
  - Werden häufig falsch interpretiert
- **sar / vmstat**
  - Zuverlässig und sehr hilfreich
- **/proc/meminfo**
  - Einzig hilfreiches Tool für HugePages
- **cat /proc/meminfo**
  - Für HugePages zwingend benötigt
  - Nicht wirklich schlimm, wenn man sich dran gewöhnt hat

Linux ate my ram!



Quelle: http://www.linuxatemyram.com/

Ansprechpartner bei OPITZ CONSULTING

Thorsten Bruhns, Solution Architect

OPITZ CONSULTING Deutschland GmbH
thorsten.bruhns@opitz-consulting.de
Telefon +49 6172 66 26 0 - 1541
Mobil +49 174 30 49 64 2

youtube.com/opitzconsulting

@OC_WIRE

slideshare.net/opitzconsulting

xing.com/net/opitzconsulting