Oracle, Memory & Linux

Presented by: Christo Kutrovsky

Pythian

love your data

Who Am I


"oracle pinup"

- Oracle ACE
- 12 years in Oracle field
- Joined Pythian 2003
- Part of Pythian Consulting Group
  - Special projects
  - Performance tuning
  - Critical services
- Presenter at: IOUG, RMOUG, UKOUG, OpenWorld

Pythian

Pythian provides services to companies and organizations that are dependent on data

Pythian Facts

- Founded in 1997, over 15 years as a profitable, private company
- 200 employees and growing
- 200 customers worldwide, 34 customers more than $1 billion in revenue
- 5 offices in 5 countries
- Employ 8 Oracle ACEs (Including 2 ACE Directors) and 2 Microsoft MVPs
- **Platinum** level partner in the Oracle Partner Network
- **Gold** level partner in the Microsoft Partner Network
- Winner of 2011 Oracle Partner Network Titan Award
- Ranked among Canada's Fastest Growing Companies in the Profit 200 in 2010, 2011 & 2012
- Ranked in the Top 250 Canadian ITC Companies in Branham300
- Average response time to alerts under 5 minutes

Why Pythian

Recognized Leader:

- Global industry leader in data infrastructure managed services and consulting with expertise in Oracle, Oracle Applications, Microsoft SQL Server, MySQL, big data and systems administration
- Work with over 200 multinational companies such as Forbes.com, Fox Sports, Nordion and Western Union to help manage their complex IT deployments

Expertise:

- One of the world's largest concentrations of dedicated, full-time DBA expertise. Employ 8 Oracle ACEs/ACE Directors
- Hold 7 Specializations under Oracle Platinum Partner program, including Oracle Exadata, Oracle GoldenGate & Oracle RAC

Global Reach & Scalability:

- 24/7/365 global remote support for DBA and consulting, systems administration, special projects or emergency response

Agenda

- Types of Physical Memory
- Virtual Memory
  - Types of memory
- How to monitor memory usage
  - Oracle specifics
- HugePages effect
- Oracle Views

Questions for Audience

- How many developers
- How many managing Linux
- How many managing Solaris
- How many managing AIX,HPUX etc.
- How many have root access
- How many have control of database memory usage
- How many still run 32 bit systems

Types of memory

Pythian
love your data

It's all memory

- CPU registers
- CPU Cache L1
- CPU Cache L2
- CPU Cache L3
- Main Memory (RAM)
  - Remote memory in NUMA
- SSD Cache (*new*)
- Magnetic Storage ("DISK")
- Tape

The Difference

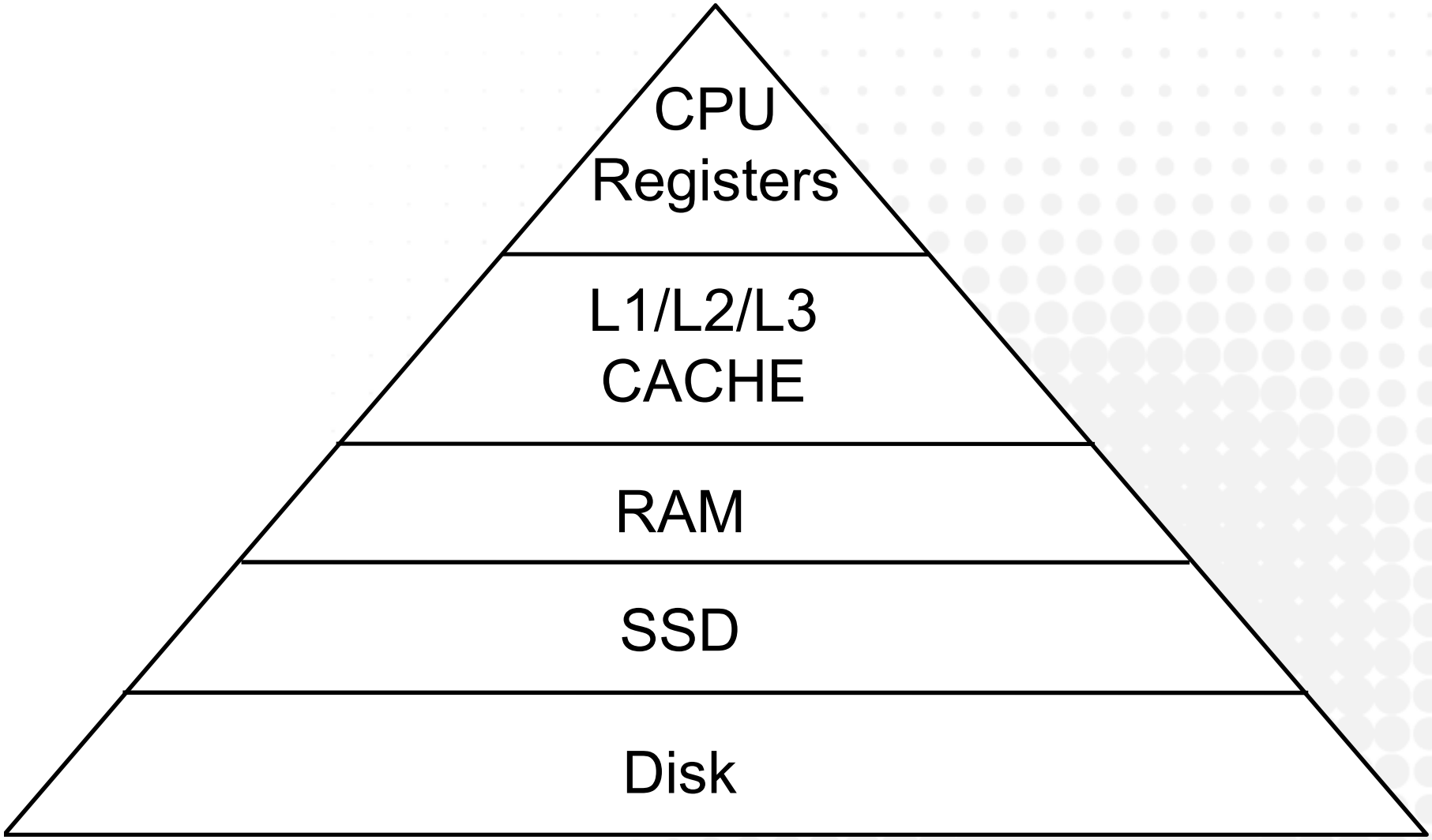
- CPU registers
- CPU Cache L1
- CPU Cache L2
- CPU Cache L3
- Main Memory (RAM)
  - Remote memory in NUMA
- SSD Cache (*new*)
- Magnetic Storage (“DISK”)
- Tape

PERFORMANCE

What is PERFORMANCE

- Performance is defined by:
  - Latency – the amount of time from data request to data receive
  - Bandwidth – how much data can flow in best case scenario
  - Cost – pure $$$

Memory Hierarchy



Different memory – Intel i7

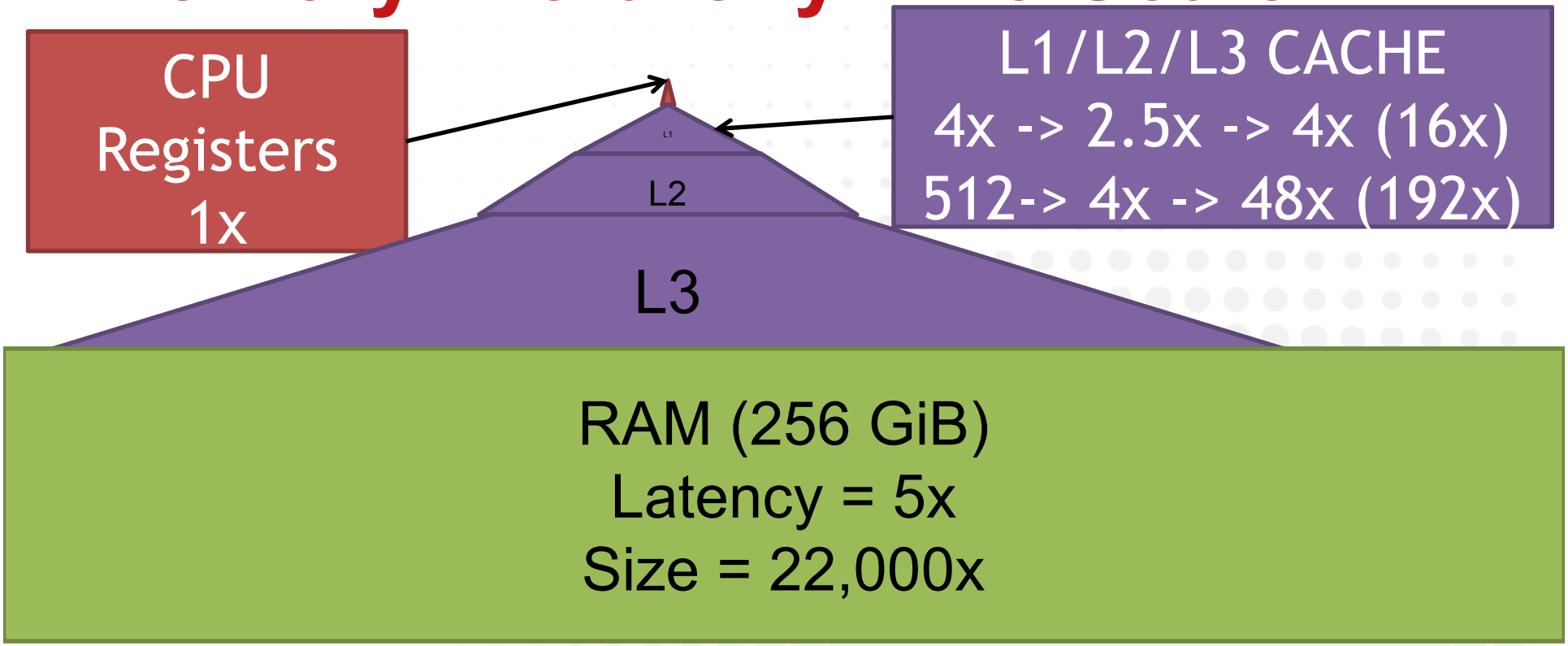
- CPU registers – **128 bytes / 0.3* ns (1 cycle)**
- CPU Cache L1 – **64 KiB / 1.2* ns**
- CPU Cache L2 – **256 KiB / 3.0* ns**
- CPU Cache L3 – **12 MiB / 12* ns**
  - Local core (1.0x), remote core (1.6x), dirty (1.9x), remote (4x)
- Main Memory (RAM) – **1 TiB or more / 60 ns**
  - Remote RAM (multi-socket) – **100 ns**
- SSD Cache (*new*) – **Any / 60,000 ns**
- Magnetic Storage ("DISK") – **Any / 3,000,000 ns**

http://www.tomshardware.com/reviews/Intel-i7-nehalem-cpu,2041-10.html
http://software.intel.com/sites/products/collateral/hpc/vtune/performance_analysis_guide.pdf
http://en.wikipedia.org/wiki/Memory_hierarchy

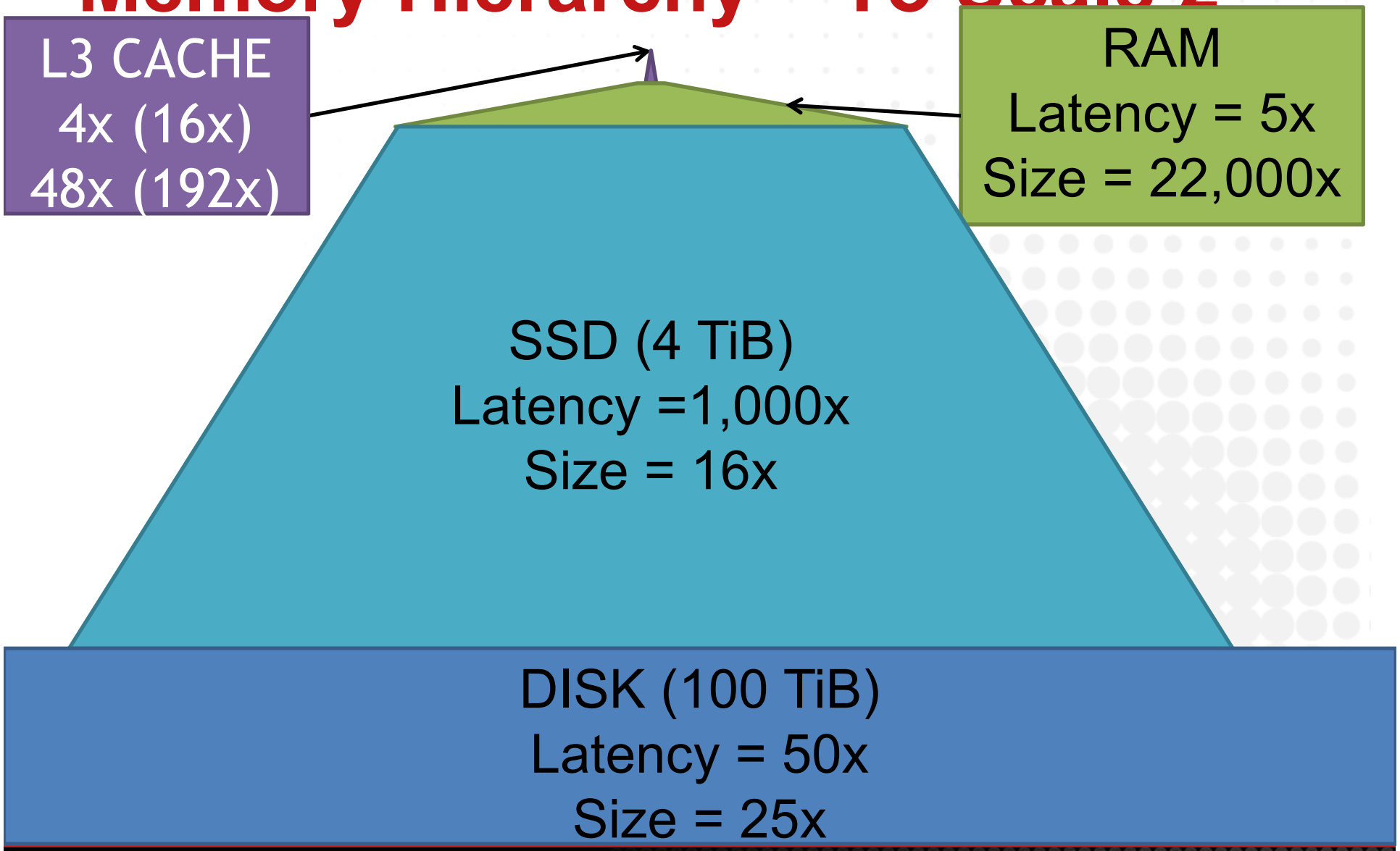
Scaled to real life

- CPU Registers – 0.3 seconds your hands
- CPU Cache – 3 seconds your desk
  - L3 cache is your desk drawer – 12 sec (48 sec if locked)
- RAM – 60 seconds - the storage room in the basement
- SSD – 16.6 Hours
- Disk – 34 days
  - Used to be 8 hours

Memory Hierarchy – To Scale



Memory Hierarchy – To Scale 2



RAM

- Not so significant latency implication
  - 4x -> 2.5x -> 4x -> 5x -> 1000x -> 50x
- Significant size up compared to previous layers
  - 512x -> 4x -> 48x -> 22,000x ->16x ->25x
- Your most important cache
  - It's a cache because it's still volatile

Virtual Memory

Pythian
love your data

How a computer works

- Read instructions from memory
  - In the old days it was from a cassette or even paper punch cards
- Execute instructions, which read some more memory
- Produce results
- stores them in memory
- display them Paper or Screen
  - (or sound)

How we use a computer

- Do many things
- Run multiple applications at the same time
- Expect them to run unaware of each other
- One application must in no way interfere with another

Virtual Memory attempts to abstract the ‘read/write’ concept and just work with “memory”

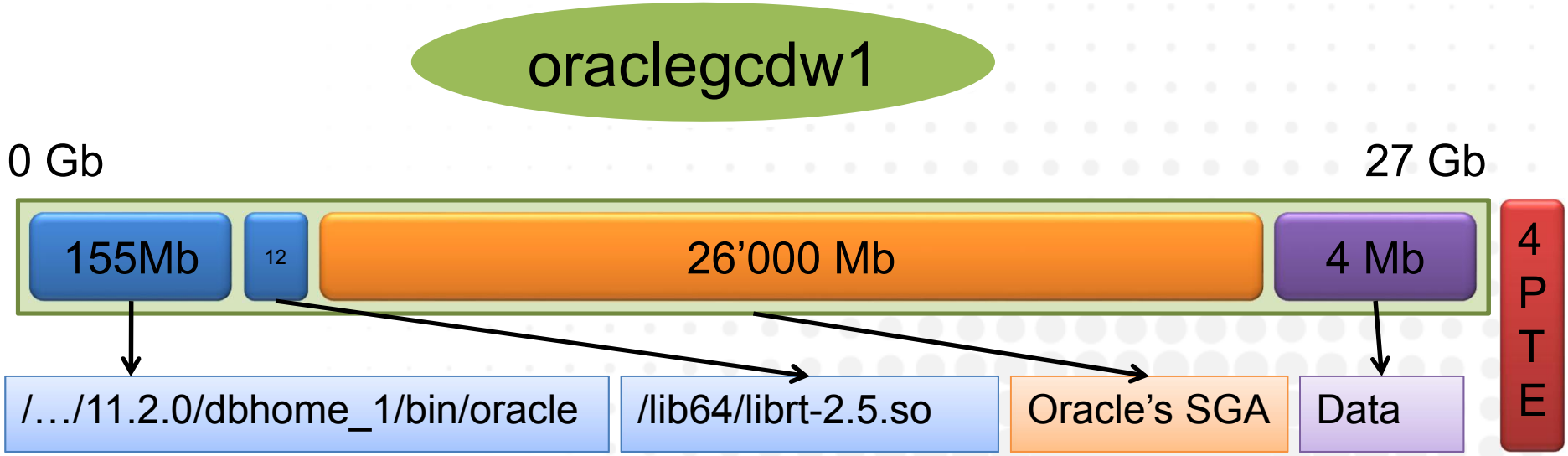
The Goal of Virtual Memory

- Write programs that do not directly depend on the amount of RAM
  - More RAM available – program runs faster
- Simplify memory management
- Run independently, safely
- OS controls what stays in RAM and what and when is evicted
- Optimal file cache usage

VM Offers

- Protection
  - Independent memory space
- Features
  - read/write/execute permissions
  - maximize memory reuse/sharing
  - mmap – memory mapped files – work with files as if they were “loaded” in memory
  - allocate more memory than available

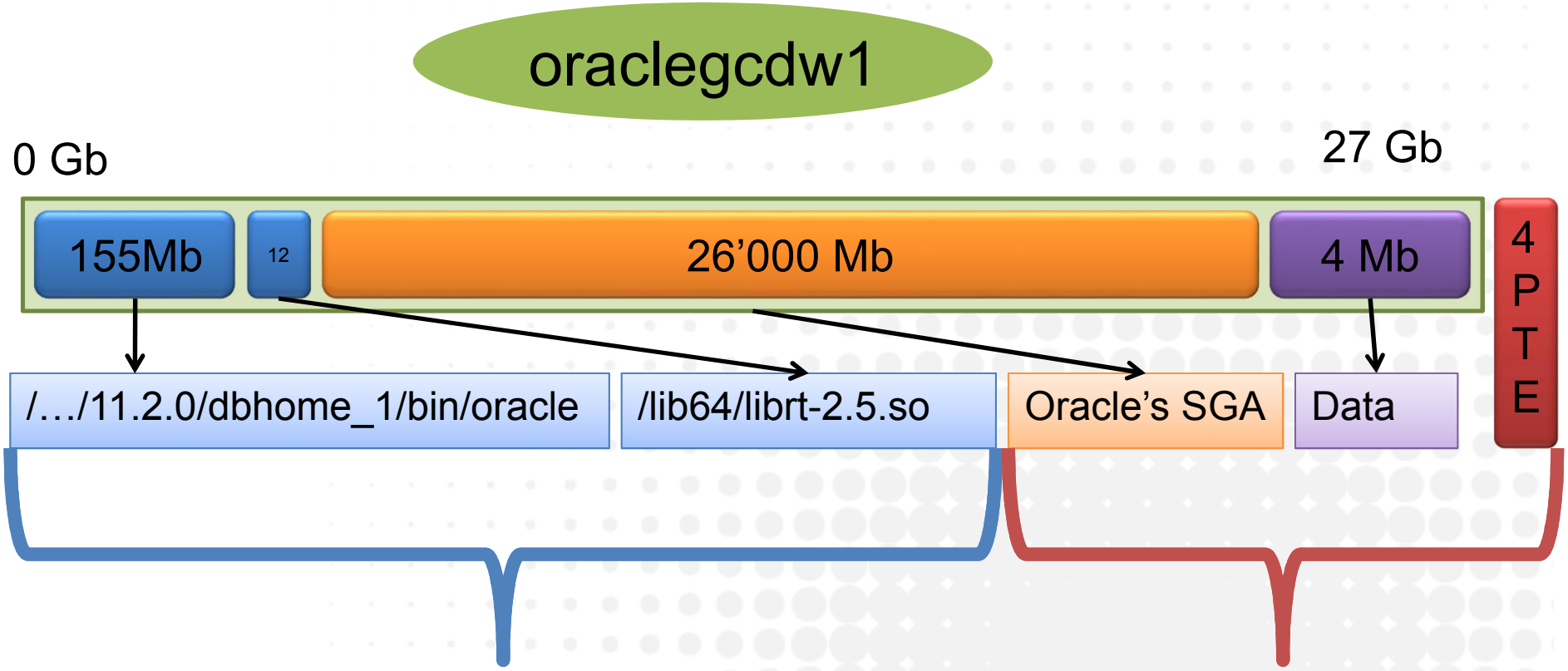
VM Visualized




```
  PID USER      PR  NI  VIRT  RES  SHR S %CPU %MEM    TIME+  COMMAND
23235 oracle    16   0 25.7g 546m 202m S 63.9  0.8 308:51.92 oraclegcdw1 (LOCAL=N
```

```
cat /proc/pid/status
…
VmPeak: 26898012 kB    VmData:     5032 kB
VmSize: 26898012 kB    VmStk:        96 kB
VmLck:          0 kB   VmExe:    155348 kB
VmHWM:     559248 kB   VmLib:     12260 kB
VmRSS:     559248 kB   VmPTE:      4328 kB
```

VM Visualized



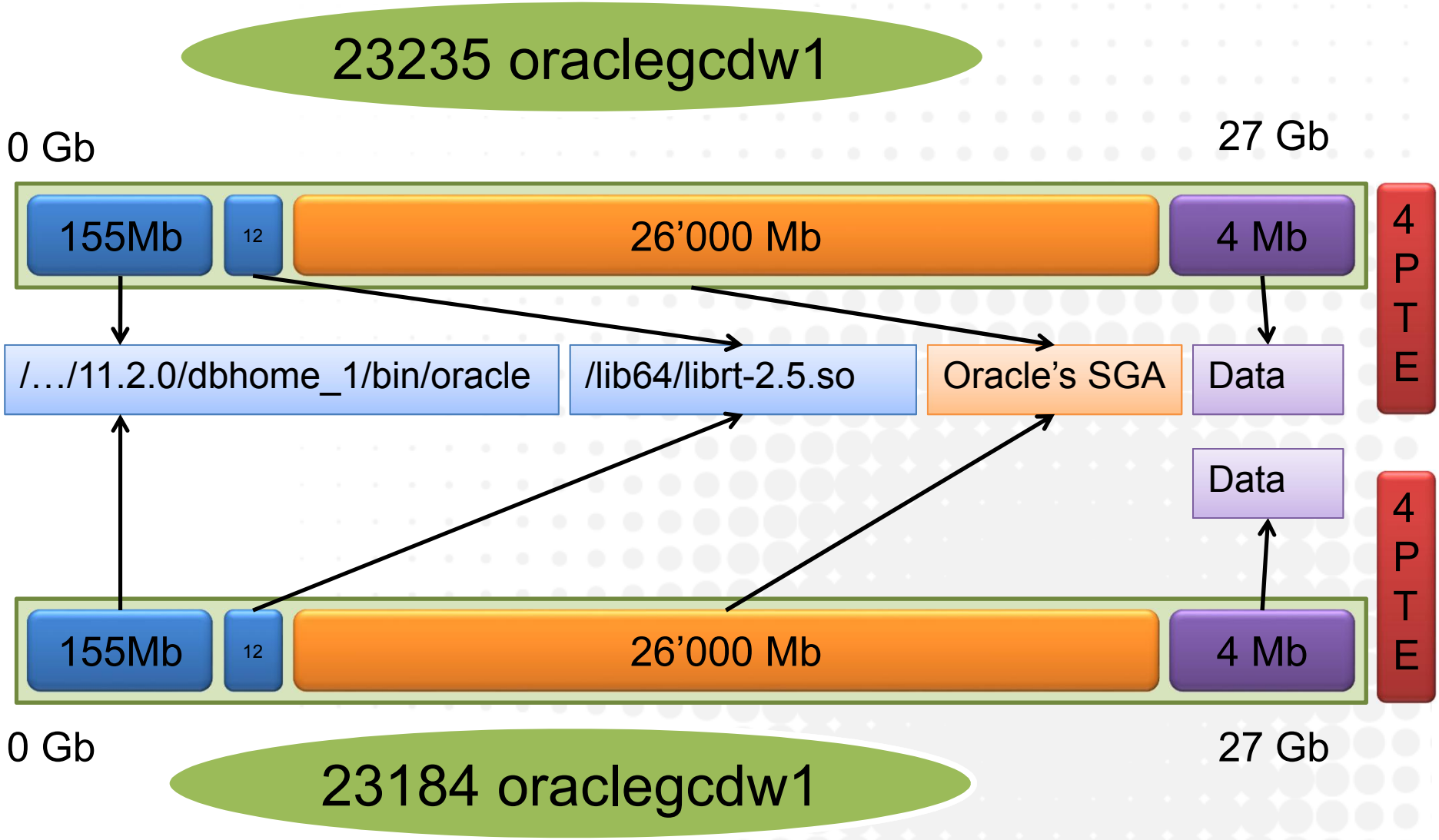
Loaded from disk
i.e.
copied from slower memory

"Anonymous"
i.e.
not copied

VM Memory types

- Two types of memory
  - With a disk representation
  - Without a disk representation
    - Depends on on OS it's called "Anonymous", "computed" and other names.

VM Visualized



Pythian
love your data

VM Memory types

- Shared
  - everything from disk
  - IPC shared memory segments
- Private
  - Anonymous memory
  - “copy on write”

/proc/meminfo

```
cat /proc/meminfo
Buffers:          613 944 kB
Cached:        38 120 544 kB
SwapCached:       10 748 kB
Active:        35 034 160 kB
Inactive:      12 114 512 kB
HighTotal:             0 kB
HighFree:              0 kB
LowTotal:      74 027 752 kB
LowFree:       22 744 448 kB
SwapTotal:     16 771 852 kB
SwapFree:      16 761 104 kB
Dirty:             9 656 kB
Writeback:             0 kB
AnonPages:      8 402 424 kB
Mapped:        18 820 948 kB
Slab:           1 165 376 kB
PageTables:     2 620 300 kB
```

Copied from Disk

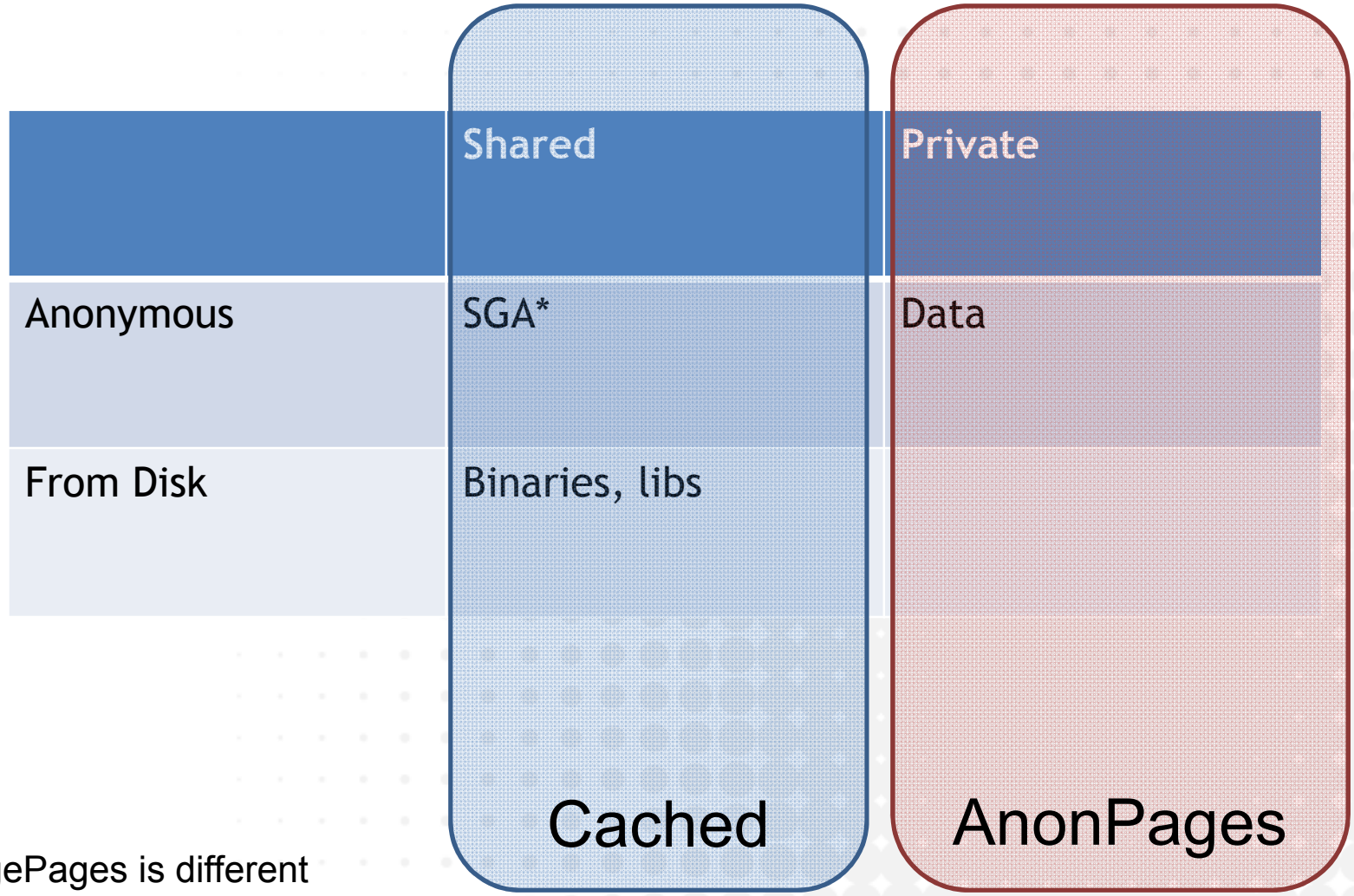
NOT from Disk

VM Metadata

VM Table

	Shared	Private
Anonymous	SGA	Data
From Disk	Binaries, libs	

VM Table



*HugePages is different

/proc/meminfo

```
cat /proc/meminfo
Buffers:          613 944 kB
Cached:        38 120 544 kB
SwapCached:       10 748 kB
Active:        35 034 160 kB
Inactive:      12 114 512 kB
HighTotal:             0 kB
HighFree:              0 kB
LowTotal:      74 027 752 kB
LowFree:       22 744 448 kB
SwapTotal:     16 771 852 kB
SwapFree:      16 761 104 kB
Dirty:             9 656 kB
Writeback:             0 kB
AnonPages:      8 402 424 kB
Mapped:        18 820 948 kB
Slab:           1 165 376 kB
PageTables:     2 620 300 kB
```

File System Cache, Binaries

Oracle SGA

Private Data, Stack, etc.

Oracle PGA

VM Metadata

/proc/meminfo - hugepages

```
cat /proc/meminfo
Buffers:          493 232 kB
Cached:        10 093 112 kB
SwapCached:            0 kB
Active:        11 311 344 kB
Inactive:       1 821 104 kB
HighTotal:             0 kB
HighFree:              0 kB
LowTotal:      82 450 640 kB
LowFree:        3 202 516 kB
SwapTotal:      2 097 144 kB
SwapFree:       2 093 676 kB
Dirty:                580 kB
Writeback:              0 kB
AnonPages:      2 636 020 kB
Mapped:           138 272 kB
Slab:             482 468 kB
PageTables:        47 132 kB
```

File System Cache, Binaries

```
HugePages_Total: 32000
HugePages_Free:   8980
HugePages_Rsvd:   1197
```

Oracle SGA

Private Data, Stack, etc.

Oracle PGA

VM Metadata

/proc/meminfo – example 2

```
cat /proc/meminfo
Buffers:          654 908 kB
Cached:        37 806 532 kB
SwapCached:       101 112 kB
Active:        34 262 824 kB
Inactive:       9 524 280 kB
HighTotal:              0 kB
HighFree:               0 kB
LowTotal:      74 027 752 kB
LowFree:       20 683 044 kB
SwapTotal:     16 771 852 kB
SwapFree:      16 670 740 kB
Dirty:             14 328 kB
Writeback:              0 kB
AnonPages:      5 222 736 kB
Mapped:        24 687 736 kB
Slab:           1 371 528 kB
PageTables:     7 843 932 kB
HugePages_Total:       0
HugePages_Free:        0
HugePages_Rsvd:        0
```

File System Cache, Binaries

Oracle SGA

Private Data, Stack, etc.

VM Metadata

Oracle's SGA

- Without HugePages
  - In CACHED section
- With HugePages
  - Not in CACHED section

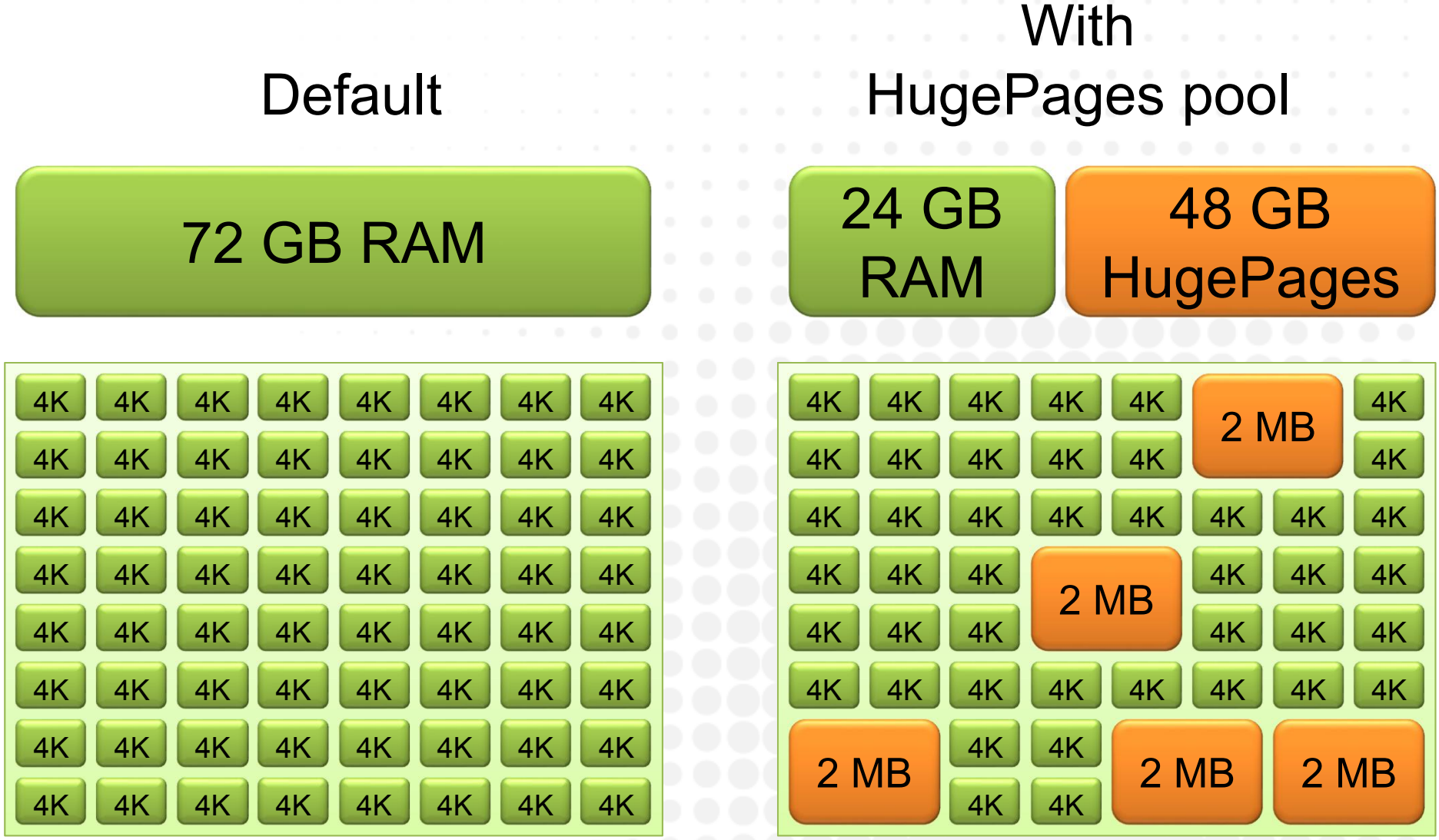
Oracle's memory

Oracle memory type	Oracle location	OS Location	OS Location (HugePages)
*pool	SGA	Cached	-
*cache_size	SGA	Cached	-
Sort/Hash	PGA	AnonPages	AnonPages
PL/SQL variables, arrays, workspace	UGA	AnonPages	AnonPages
Local cursor cache, workareas etc.	UGA	AnonPages	AnonPages
Bind variable data	UGA (+SGA)	AnonPages	AnonPages
Binaries	-	Cached	Cached

What is HugePages

- A separate memory area that is
  - Used only for shared memory segments*
  - Non-swappable – locked in memory
    - Thus not managed by VM memory
  - Managed in 2 Mb continuous memory segments

What is HugePages



VM Tricks – untouched memory

- Untouched memory does not exists
  - No PTE entires
  - No memory consumed

VM Tricks – untouched memory

```
cat grab.c
main() {void *p;
p=malloc(1073741824);
sleep(60);}

cat /proc/meminfo
...
MemFree:         3 230 592 kB
...
Committed_AS:       49 972 kB

./grab
cat /proc/meminfo
...
MemFree:         3 230 464 kB
...
Committed_AS:    1 098 808 kB
```

VM Tricks – untouched memory

- What does that mean for Oracle?
  - Non-initialized SGA (db_cache) takes no memory
  - Untouched SGA by a new process consumes no PTE entries

```
cat /proc/pid/status
…
VmPeak:  26 898 012 kB
VmSize:  26 898 012 kB
VmLck:             0 kB
VmHWM:       559 248 kB
VmRSS:       559 248 kB
```

```
VmData:      5032 kB
VmStk:         96 kB
VmExe:     155348 kB
VmLib:      12260 kB
VmPTE:       4328 kB
```

VM Tricks - swap

- Memory can be both swapped and not-swapped
  - Only anonymous memory is swapped
    - Hugepages cannot be swapped, ever
- Linux swaps recently unused memory in anticipation of it's eviction
  - Only "touched" memory needs to be swapped
- SwapCached shows memory that exists in both swap and RAM
  - Actual swapping:
    SwapTotal – SwapFree - SwapCached

VM Tricks - OverCommit

- Linux can allocate more memory that is available
  - Available = RAM + Total SWAP
- True “out of memory” on linux very rare
  - Controlled via
    /proc/sys/vm/overcommit_memory
    /proc/sys/vm/overcommit_ratio
- OOM Killer – Out Of Memory Killer
  - A kernel thread that kills abusers

VM Tricks – Solaris

- Solaris is different
- VM Concept similar
- Difference in implementation
  - No Overcommit
    - No OOM Killer
  - All anonymous pages MUST* have the available swap, should they need to be swapped
    - Oracle's SGA is anonymous

* Unless using ISM

VM Tricks – Solaris Pages

- Hugepages are automatic
  - Multiple sizes: 64K, 2Mb, 64 Mb
  - If available…
  - Shutdown database, copy files, startup – uses more smaller pages

VM Tricks – Solaris ISM/DISM

- PTE Tables are shareable
  - ISM – Intimate Shared Memory
    - Memory locked
  - DISM – Dynamic Intimate Shared Memory
    - Memory lockable
    - Requires SGA size in available SWAP space
- If sga_max_size > sga_target
  - DISM is used
  - **Needs ORADISM to run as root, to lock SGA**
    - **And project limits relaxed**

Monitoring Memory

Pythian
love your data

Linux memory tools

- Global
  - /proc/meminfo
  - vmstat
  - Ipcs
  - cgroups (**)
- Per process
  - top
  - /proc/pid/status
  - /proc/pid/maps

Solaris memory tools

- Global
  - prstat
  - vmstat
  - ipcs
  - “memstat” kernel debugger call
    - echo “::memstat”|mdb -k
- Per process
  - pmap -x

prstat

```
  PID USERNAME  SIZE   RSS STATE  PRI NICE      TIME  CPU PROCESS/NLWP
 23042 oracle   8134M 8079M cpu0     0    0   0:00:03 0.2% oracle/2
  1027 oracle    122M   77M sleep   59    0  14:33:26 0.0% oraagent.bin/34
  1254 oracle   8136M 8078M sleep   59    0   6:41:32 0.0% oracle/1
  1069 oracle     74M   51M sleep   49    0   4:37:37 0.0% ocssd.bin/15
  1007 oracle    119M   84M sleep   59    0   3:08:58 0.0% ohasd.bin/39
 23032 root     4112K 3824K cpu46   59    0   0:00:00 0.0% prstat/1
 23041 oracle     44M   14M sleep   59    0   0:00:00 0.0% sqlplus/1
 23017 root     7544K 6240K sleep   59    0   0:00:00 0.0% sshd/1
  1276 oracle   8134M 8077M sleep   59    0   1:10:51 0.0% oracle/1
  1092 oracle     56M   42M sleep   59    0   1:28:33 0.0% diskmon.bin/6
```

Vmstat Solaris

- No way to tell if the system is swaping or writing to disk
  - Vmstat –s “so” and “si” – always zero
  - Best guess is a combination of non-zero “sr” and “po”/”pi” and activity on swap device

Vmstat – Solaris

vmstat 2

```
 kthr      memory            page            disk          faults      cpu
 r b w   swap  free  re  mf pi po fr de sr s0 s1 s2 s3   in   sy   cs us sy id
 3 0 0 15940984 1785088 8 23 0  5  5  0  0 10 28  7  5 8296 28180 10890 6 2 92
 0 0 0 15411984 1412504 4 11 0  0  0  0  0  0  0  2  0 4487 1919 2483  0  0 100
 0 0 0 15411664 1412248 0 1  0  0  0  0  0  0  0  1  0 4476 1530 2438  0  0 100
^C
```

vmstat -S 2

```
 kthr      memory            page            disk          faults      cpu
 r b w   swap  free  si  so pi po fr de sr s0 s1 s2 s3   in   sy   cs us sy id
 3 0 0 15940968 1785080 0 0  0  5  5  0  0 10 28  7  5 8296 28179 10890 6 2 92
 0 0 0 15411960 1412480 0 0  0  0  0  0  0  0  0  2  0 4446 1560 2410  0  0 100
 0 0 0 15411640 1412224 0 0  0  0  0  0  0  0  2  1  0 4585 1898 2642  0  0 100
```

Memstat (mdb)

```
bash-3.00# mdb -k
Loading modules: [ unix genunix specfs dtrace zfs sd mpt px ldc ip
hook neti sctp arp usba fctl nca lofs cpc random crypto fcip
logindmux ptm ufs sppp nfs ipc ]
> ::memstat
Page Summary                Pages                MB  %Tot
------------     ----------------  ----------------  ----
Kernel                     215755              1685   11%
ZFS File Data              343278              2681   17%     File Cache
Anon                      1214036              9484   59%     Oracle SGA
Exec and libs               52228               408    3%
Page cache                  32394               253    2%
Free (cachelist)            66268               517    3%
Free (freelist)            118696               927    6%

Total                     2042655             15958
Physical                  2011811             15717
```

ipcs / sysresv

- ipcs –a (OS tool)
  - Both Solaris and Linux
  - Shows all Shared Memory Segments, sempahors etc.
  - ipcrm can remove orphan segments
- sysresv (Oracle tool)
  - Reports semaphors and keys for current ORACLE_SID

ipcs

ipcs -a

```
------ Shared Memory Segments --------
key        shmid      owner      perms      bytes      nattch
status
0x00000000 0          root       644        72         2
0x00000000 32769      root       644        16384      2
0x00000000 65538      root       644        280        2
0xed304ac0 163844     oracle     660        4096       0
0x466ff99c 557061     oracle     660        26845642752 182

------ Semaphore Arrays --------
key        semid      owner      perms      nsems
0x869d3e0c 131073     oracle     660        125
0x869d3e0d 163842     oracle     660        125
```

sysresv

sysresv

```
IPC Resources for ORACLE_SID "qadw1" :
Shared Memory:
ID              KEY
557061          0x466ff99c
Semaphores:
ID              KEY
2490378         0xd9773da4
2523147         0xd9773da5
2555916         0xd9773da6
...
```

Linux /proc/meminfo

```
MemTotal:       74 027 752 kB
MemFree:        20 683 044 kB
Buffers:           654 908 kB
Cached:         37 806 532 kB
SwapCached:        101 112 kB
Active:         34 262 824 kB
Inactive:        9 524 280 kB
HighTotal:               0 kB
HighFree:                0 kB
LowTotal:       74 027 752 kB
LowFree:        20 683 044 kB
SwapTotal:      16 771 852 kB
SwapFree:       16 670 740 kB
Dirty:             14 328 kB
Writeback:               0 kB
AnonPages:       5 222 736 kB
Mapped:         24 687 736 kB
Slab:            1 371 528 kB
PageTables:      7 843 932 kB
NFS_Unstable:            0 kB
Bounce:                  0 kB
CommitLimit:    53 785 728 kB
Committed_AS:   43 555 536 kB
VmallocTotal:   34 359 738 367 kB
VmallocUsed:       290 700 kB
VmallocChunk:   34 359 447 655 kB
HugePages_Total:         0
HugePages_Free:          0
HugePages_Rsvd:          0
Hugepagesize:         2048 kB
```

Linux /proc/meminfo Newer

```
MemTotal:       16 344 972 kB
MemFree:        13 634 064 kB
Buffers:            3 656 kB
Cached:          1 195 708 kB
SwapCached:             0 kB
Active:           891 636 kB
Inactive:       1 077 224 kB
HighTotal:     15 597 528 kB
HighFree:      13 629 632 kB
LowTotal:         747 444 kB
LowFree:            4 432 kB
SwapTotal:              0 kB
SwapFree:               0 kB
Dirty:                968 kB
Writeback:              0 kB
AnonPages:        861 800 kB
Mapped:           280 372 kB
Slab:             284 364 kB
SReclaimable:     159 856 kB
SUnreclaim:       124 508 kB
PageTables:        24 448 kB
NFS_Unstable:           0 kB
Bounce:                 0 kB
WritebackTmp:           0 kB
CommitLimit:    7 669 796 kB
Committed_AS:     100 056 kB
VmallocTotal:     112 216 kB
VmallocUsed:          428 kB
VmallocChunk:     111 088 kB
```

MemTotal

- Total memory available to Linux
  - Excludes reserved region
- If not what you expect – check the DIMM's. They do occasionally die.

MemFree

- Wasted memory
  - Memory not currently in use by anything
  - May be removed from the system as it provides no benefit
- Memory immediately available to be used by a process touching memory
- Will be used by any non-directIO filesystem read
- Will be consumed until approaches /proc/sys/vm/min_free_kbytes

MemFree – example

```
grep MemFree /proc/meminfo
MemFree:          26 568 kB

echo 900000 > /proc/sys/vm/min_free_kbytes

grep MemFree /proc/meminfo
MemFree:         210 056 kB
```

Buffers

- Cache of raw disk blocks
  - Usually occupied with ext3 metadata
  - Mostly ext3 pointers (extent management)
  - Not the cache of actual user data
- Should be relatively low (400Mb) on ASM systems as filesystem metadata is inside ASM
  - Unless filesystem used for backups, or other data

Cached

- Memory that is copied from disk to RAM
  - File system cache
  - Binaries been executed
- Can be dirty i.e. Requires disk writes to be released
- If not dirty, can be very quickly released when programs request memory
- **Will include the Oracle SGA if not using hugepages**

Cached – example 1

```
[root@ ~]#  cat /proc/meminfo
…
MemFree:        8232512 kB
Buffers:           9328 kB
Cached:           28372 kB
…
du -smc indx01_*
1714    indx01_01.dbf
1761    indx01_02.dbf
1722    indx01_03.dbf
5197    total
…
```

cat indx01_* > /dev/null

Cached – example 2

```
[root@ ~]# vmstat 2
procs -----------memory---------- ---swap-- -----io---- --system-- ----cpu----
 r  b   swpd   free   buff  cache   si   so    bi    bo   in    cs us sy id wa
 0  0      0 8093888  10808 163392    0    0     0     0 1012    17  0  0 100  0
 0  0      0 8093952  10808 163392    0    0     0     0 1012    16  0  0 100  0
 0  1      0 7956736  10948 300272    0    0 68602     0 1567  1126  0  2 76 22
 0  1      0 7808576  11092 448068    0    0 73992    80 1623  1210  0  2 75 23
…
 0  1      0 2847616  16104 5397616   0    0 65792     0 1542  1076  0  2 75 23
 0  0      0 2766272  16180 5479180   0    0 40698     0 1341   675  0  1 85 14
 0  0      0 2766208  16192 5479168   0    0     0   114 1033    22  0  0 100
0
cat /proc/meminfo
…
MemFree:        2766464 kB
Buffers:          16192 kB
Cached:         5479168 kB
…
```

Cached writing – example

```
cat indx01_* >newfile
vmstat 2
procs -----------memory---------- ---swap-- -----io---- --system-- ----cpu----
 r  b   swpd   free   buff  cache   si   so    bi    bo   in    cs us sy id wa
 0  0      0 2765312  17044 5479356    0    0     0     0 1012    17  0  0 100  0
 0  3      0 2405376  17428 5833612    0    0    16 36866 1324   144  1 18 76  6
 0  2      0 2143616  17688 6091532    0    0     4 111748 2000   213  0 16 50 34
…
 0  1      0  16832   6784 8198556    0    0  8556 26684 1942  1267  0  2 74 24
 1  1      0  16832   6856 8198744    0    0 12518 20720 2130  1767  0  3 74 23
…
cat /proc/meminfo
…
MemFree:           16768 kB
Buffers:            2192 kB
Cached:          8196908 kB
…
Dirty:            277468 kB
Writeback:             0 kB
…
```

Cached removing file

cat /proc/meminfo

```
…
MemFree:           20672 kB
Buffers:            3300 kB
Cached:          8191900 kB
…
Dirty:                0 kB
Writeback:            0 kB
…
```

rm newfile

```
procs -----------memory---------- ---swap-- -----io---- --system-- ----cpu----
 r  b   swpd   free   buff  cache   si   so    bi    bo   in    cs us sy id wa
 0  0      0  23296   3380 8189480    0    0     0    28 1015    18  0  0 100  0
 0  1      0 3257472   3948 4996372    0    0   284     0 1084   160  0 14 78  8
 0  1      0 3255552   5828 4996572    0    0   940     0 1247   485  0  1 75 24
 0  1      0 3253696   7616 4996344    0    0   884    96 1237   470  0  2 75 23
 0  0      0 3253440   7988 4996492    0    0   186     0 1061   112  0  0 95  4
 0  0      0 3253440   7988 4996492    0    0     0     0 1012    14  0  0 100
0
```

Active/Inactive

- Active – recently used memory
  - Includes all types of memory (cached, buffers, anonymous)
  - OS will try to keep it in RAM
- Inactive – memory that will be first reused
  - “free” memory
  - Can be used to gauge the “working set”

Dirty / Writeback

- Dirty – cache/buffers memory that requires to be written to disk
  - thresholds can be adjusted
- Writeback – memory actively been written to disk
  - Can reach high values with async writes with large queue

Commited_AS / CommitLimit

- Committed_AS
  - Total memory requested on the system
  - Not used, just requested
  - If every process in the system is to touch and use the memory it has requested, this is how much would be used
- CommitLimit
  - Total memory that can be requested
  - Should factor over allocate
  - Memory allocation errors* when you reach limit – (overcommit_memory)

Commit_AS example

```
cat grab.c
main() {void *p;
p=malloc(1073741824);
sleep(60);}

cat /proc/meminfo
...
MemFree:         3 230 592 kB
...
Committed_AS:       49 972 kB

./grab
cat /proc/meminfo
...
MemFree:         3 230 464 kB
...
Committed_AS:    1 098 808 kB
```

Slab

- Slab – “in-kernel data structures cache”
  - similar to Oracle’s “shared_pool”
  - designed to prevent memory fragmentation
  - detailed monitoring:
    /proc/slabinfo
    slabtop
- Basically “system space”

slabtop example

```
Active / Total Objects (% used)    : 88874 / 139343 (63.8%)
Active / Total Slabs (% used)      : 5839 / 5846 (99.9%)
Active / Total Caches (% used)     : 90 / 132 (68.2%)
Active / Total Size (% used)       : 17286.03K / 23311.27K (74.2%)
Minimum / Average / Maximum Object : 0.01K / 0.17K / 128.00K

  OBJS ACTIVE  USE OBJ SIZE  SLABS OBJ/SLAB CACHE SIZE NAME
32382  24900  76%    0.27K   2313       14       9 252K radix_tree_node
56925  40013  70%    0.05K    759       75       3 036K buffer_head
  364    363  99%    4.00K    364        1       1 456K size-4096
 2485   2471  99%    0.54K    355        7       1 420K ext3_inode_cache
 2376    413  17%    0.50K    297        8       1 188K size-512
  256    256 100%    3.00K    128        2       1 024K biovec-(256)
 4576   4481  97%    0.15K    176       26         704K dentry_cache
10248   4548  44%    0.06K    168       61         672K size-64
 4340   1215  27%    0.12K    140       31         560K size-128
 1980    316  15%    0.25K    132       15         528K size-256
...
```

HugePages

- HugePages_Total
- HugePages_Free
- HugePages_Rsvd
  - Allocated but untouched pages
  - Must have Rsvd amount FREE hugepages or bad things happen to Oracle
- Hugepages are:
  - Locked in memory
  - 512 larger than regular 4KiB pages
    - Requires 512 times less PTE Entries

HugePages - Example

- Config:
  - 1.7 Gb sga (max on 32 bit without VLM)
  - 1400 Mb in db_cache_size
  - table sized to fit exactly in cache
- Test
  - Start 100 sessions,
  - full scan test table (cached) in order to touch the memory and allocate the PTEs
  - Sessions will wait via dbms_lock.allocate to be released
- Show before and after PageTables usage

HugePages - Example

```
Before starting the sessions (db is UP)

cat /proc/meminfo

…

MemFree:        1 070 472 kB
PageTables:         4 932 kB


After sessions have finished touching the memory

cat /proc/meminfo

…

MemFree:          473 496 kB
PageTables:       295 068 kB
```

PTE Tables

- Page Table Entries
  - Per process non-shareable (except ISM solaris)
  - Minimum 16 bytes in size*
- 100 processes (read sessions) mapping 200 GiB of SGA

  200*1024*1024/4 = 52'428'800 (4KiB pages) * 16 bytes

  800 MiB **PER PROCESS**

  Total: 80'000 MiB
- With HugePages
  - 1600 KiB **per process**

  Total: 156 Mib

A New Page Table for 64-bit Address Spaces
Madhusudhan Talluri, Sun,
http://citeseerx.ist.psu.edu/viewdoc/download?doi=10.1.1.110.4178&rep=rep1&type=pdf

CGROUPS

Pythian
love your data

CGROUPS

- Linux resource manager Supports:

blkio – this subsystem sets limits on input/output access to and from block devices such as physical drives (disk, solid state, USB, etc.).

cpu – this subsystem uses the scheduler to provide cgroup tasks access to the CPU.

cpuacct – this subsystem generates automatic reports on CPU resources used by tasks in a cgroup.

cpuset – this subsystem assigns individual CPUs (on a multicore system) and memory nodes to tasks in a cgroup.

devices – this subsystem allows or denies access to devices by tasks in a cgroup.

freezer – this subsystem suspends or resumes tasks in a cgroup.

memory – this subsystem sets limits on memory use by tasks in a cgroup, and generates automatic reports on memory resources used by those tasks.

net_cls – this subsystem tags network packets with a class identifier (classid) that allows the Linux traffic controller (tc) to identify packets originating from a particular cgroup task.

net_prio – this subsystem provides a way to dynamically set the priority of network traffic per network interface.

CGROUPS – memory

- Hierarchical memory control
  - Inherited
  - Includes shared pages
  - Includes file system cache
  - Includes SWAP
  - Can include KERNEL (slab, sockets, stack)

CGROUPS – memory

- memory.usage_in_bytes
- memory.memsw.usage_in_bytes
- memory.oom_control
- memory.soft_limit_in_bytes
- Notifier
  - low/medium/critical

CGROUPS – testing

- Mount memory cgroup module (doc)
- Create cgroup “0”
- echo “$$” of bash to 0/task
- startup database
- Run:

```
wile [[ 1 ]] ; do echo $((`cat
 memory.memsw.usage_in_bytes`/1000000
 )) ; sleep 1 ; done
```

CGROUPS – example 1

867

```
SQL> create tablespace mytest
 datafile size 1g;


Tablespace created.


SQL>
```

1871

CGROUPS – shutdown

```
1872
SQL> shutdown immediate;
Database closed.
Database dismounted.
ORACLE instance shut down.
1077
```

CGROUPS – file delete

1862

```
SQL> drop tablespace mytest;

Tablespace dropped.
```

789

CGROUPS – direct IO

798

```
SQL> create tablespace mytest
 datafile size 1g;


Tablespace created.
```

798

CGROUPS – direct path

```
799
SQL> insert /*+APPEND*/ into t1
 select * from t1;


331040 rows created.


SQL> commit;


Commit complete.
799
```

CGROUPS – 4g

```
alter system set memory_target=4g
 scope=spfile;


System altered.

startup
```

2597

CGROUPS – sga init

```
4939 622
SQL> select count(*) from t1;
--"_serial_direct_read" = never;

5158 664
5158 680
…
5158 844
5158 904
5159 973
```

Monitoring Memory from Oracle

Pythian
love your data

Oracle views

- V$PROCESS
  - PGA_USED_MEM,PGA_ALLOC_MEM, PGA_FREABLE_MEM, PGA_MAX_MEM
- V$PROCESS_MEMORY
  - PL/SQL vs SQL vs Other
- V$PROCESS_MEMORY_DETAIL
  - Not populated
  - Requires event “PGA_DETAIL_GET“ to be set
  - Only for testing

Thank you and Q&A

To contact us…


sales@pythian.com


1-877-PYTHIAN

To follow us…


http://www.pythian.com/news/


http://www.facebook.com/pages/The-Pythian-Group/163902527671


@pythian


@pythianjobs


http://www.linkedin.com/company/pythian

Pythian
love your data

Transition/Section Marker

Pythian
love your data