Quick Tips on Linux HugePages

By Yong Huang

The following are a few quick and practical tips on using Linux HugePages on servers that run Oracle databases.

1. Be generous first and dynamically shrink later.

If you don't want to be accurate in calculating how much memory to allocate for HugePages, give a rough and very generous estimate. Start all Oracle instances on the box. (To save time, startup nomount is enough.) Check the difference between HugePages_Free and HugePages_Rsvd, which is the wastage, because HugePages_Free includes reserved but not actually used memory. For example:

```
$ grep Huge /proc/memory
...
HugePages_Total:  3190
HugePages_Free:   2458
HugePages_Rsvd:   2341
```

2458-2341=117 pages of HugePages or 234 MB memory (assumes 2 MB page size) will never be used. You do not have to wait until the instances have been used for a while; that would increase both HugePages_Free and HugePages_Rsvd, but not the difference between them.

Now, we can dynamically shrink HugePages to reduce wastage. Let's cut that down to, say, 10 pages. So we should decrease HugePages_Total by 117-10=107. That is, change 3190 to 3190-107=3083.

```
# echo 3083 > /proc/sys/vm/nr_hugepages
```

cat /proc/sys/vm/nr_hugepages to confirm the number has been reduced to 3083. Update vm.nr_hugepages in /etc/sysctl.conf with this number so it will take effect on the next reboot.

The advantage of over-allocating HugePages at the beginning is that it saves time in getting the memory allocation right on the first try. In addition, dynamically changing HugePages allocation ensures no memory is wasted. In the case of shutting down an Oracle instance for an extended period of time, you can even lower /proc/sys/vm/nr_hugepages to give the memory back to OS as well as Oracle PGA. But then if you start back up the previously shut down instance, you have to increase the nr_hugepages number first, and you may not be able to bring it up fully to the desired number if the available memory is no longer physically contiguous. When that happens, you may or may not be able to start the instance depending on the setting of use_large_pages. If it's set to true (default), the instance may be started, but it uses no HugePages at all and you'll waste lots of HugePages. You can lower nr_hugepages back down to release the memory back to OS and wait until the next server reboot for sufficient contiguous memory. So, think it over whenever you plan to dynamically lower the value.

2. Seeing is believing.

In older versions of Oracle, the only way to know that HugePages is used is to check /proc/memory. Later versions show the lines in alert.log (Oracle 11*g* example):

```
Total Shared Global Region in Large Pages = 2370 MB (100%)

Large Pages used by this instance: 1185 (2370 MB)
Large Pages unused system wide = 815 (1630 MB)
Large Pages configured system wide = 2000 (4000 MB)
Large Page size = 2048 KB
```

The instance in this example clearly has too much unused HugePages. I would cut configured HugePages down from 2,000 to 2,000-815+overhead, say, 1,200. (The overhead may be related to the number of shared memory segments for the instance as shown in ipcs or sysresv, among other things.)

In 12*c*, the alert.log has these lines instead (excluding the annoying timestamp lines profusely intercalated):

```
 PAGESIZE  AVAILABLE_PAGES  EXPECTED_PAGES  ALLOCATED_PAGES  ERROR(s)
      4K       Configured               5                5        NONE
   2048K            1620            1617             1617        NONE
```

This example only wastes three HugePages, corresponding to the following /proc/meminfo values where 10-7=3:

```
HugePages_Total:    1620
HugePages_Free:       10
HugePages_Rsvd:        7
```

Beginning with Linux kernel 2.6.29 or Red Hat Enterprise Linux 6 and possibly later minor releases of RHEL 5, /proc/*pid*/smaps provides clues about HugePages usage as well.

```
# cat /proc/<any pid of Oracle instance>/smaps
...
61000000-a7000000 rwxs 00000000 00:0c 1146885                            /SYSV00000000 (deleted)
Size:            1146880 kB
Rss:                   0 kB
Pss:                   0 kB
Shared_Clean:          0 kB
Shared_Dirty:          0 kB
Private_Clean:         0 kB
Private_Dirty:         0 kB
Referenced:            0 kB
Anonymous:             0 kB
AnonHugePages:         0 kB
Swap:                  0 kB
KernelPageSize:     2048 kB  ← 2MB HugePage size
MMUPageSize:        2048 kB  ← 2MB HugePage size
```

The last two lines showing 2 MB instead of 4 KB page size are the telltale sign that HugePages are used.

Beginning with Oracle 12*c*, there's yet another way to check the usage. Fixed table x$ksmssinfo (probably Kernel Service, Memory Sga OS (level) Info) not only can tell you whether the memory page size is that of HugePages, but can even map the SGA components with shared memory segments. The example below is from Oracle 12.1.0.2, where in-memory area of 112 MB is configured. (I removed the ipcs lines irrelevant to this Oracle instance in the example.)

```
SQL> select "AREA NAME", "SEGMENT SIZE", "SIZE", pagesize, shmid from x$ksmssinfo;

AREA NAME            SEGMENT SIZE       SIZE   PAGESIZE      SHMID
-------------------- ------------ ---------- ---------- ----------
imc area rdonly 0        83886080   83886080    2097152   87588873
Variable Size          3288334336 3254779904    2097152   87621642
imc area default 0     3288334336   33554432    2097152   87621642
Redo Buffers             14680064   13844480    2097152   87654411
Fixed Size                4194304    2932736    2097152   87556104
skgm overhead               20480      20480       4096   87687180

SQL> !ipcs -m

------ Shared Memory Segments --------
key        shmid      owner      perms      bytes      nattch     status
...
0x00000000 87556104   oracle     640        4194304    25
0x00000000 87588873   oracle     640        83886080   25
0x00000000 87621642   oracle     640        3288334336 25
0x00000000 87654411   oracle     640        14680064   25
0x639dac14 87687180   oracle     640        20480      25
```

As you can see, this fixed table tells us HugePages is used except for Oracle's interface to the OS in the generic memory management layer (skgm overhead), which still uses the default 4 KB page size. The largest HugePages segment of 3288334336 bytes in size maps to two areas inside Oracle: Variable Size (not the same as Variable Size shown by SQL*Plus command show sga, which excludes buffer cache) used for buffer cache and various SGA pools (shared pool, java pool, large pool, etc.), and part of the in-memory area or column store (imc area default 0). The second largest segment of 83886080 bytes contains the other part of in-memory area (imc area rdonly 0). The remaining two HugePages segments are obvious. In spite of small sizes, they are not fully used by Oracle.

For more about HugePages, see **http://yong321.freeshell.org/oranotes/HugePages.txt** and the references cited therein.

About the Author

Yong Huang is a DBA at MD Anderson Cancer Center in Houston. Before joining MD Anderson, Yong worked as an Oracle DBA and consultant at Schlumberger, Unocal Oil, Nationwide Insurance, Electronic Arts and eBay.