Introduction of runtime memory instrumentation with kernel functions

Machida AT sm.sony.co.jp

2006.04.10

List of information

- Mem usage APIs ported from 2.4 to 2.6.11
  - All patches are included in ***20060410-runtime-mem-usage.tgz*** attached at http://tree.celinuxforum.org/CelfPubWiki/RuntimeMemoryMeasurement
  - /proc/<pid>/statrm – memory-accounting.patch
    - Summary of Resident/Shared page info
  - /proc/<pid>/pgstat – memory-accounting-1.patch
    - Detailed page info.
  - /proc/<pid>/memmap – memory-accounting.patch
    - Detailed page info of Shared mem.
  - /proc/memmap – memory-accounting.patch
    - Usage of phy mem.
  - /proc/freemem – freemem-1.patch
    - Accurate memory counting API
- Other reference for general info (usage of PS and/or TOP)
  - **Section 5 "Measuring memory in Linux"** in http://tree.celinuxforum.org/CelfPubWiki/RuntimeMemoryMeasurement

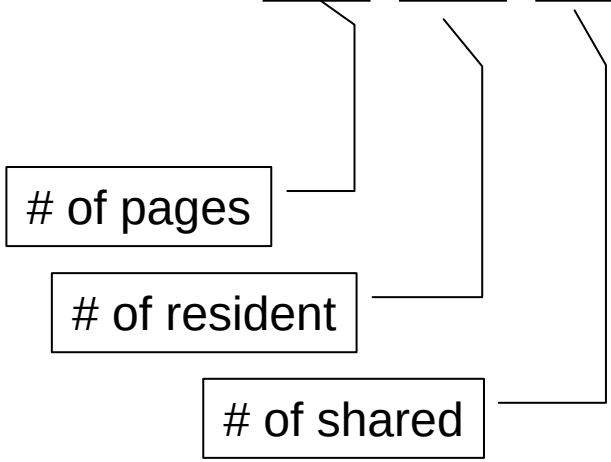
/proc/<pid>/statrm

- This entry summarizes the information provided by /proc/<pid>/memmap and combines it with some of the information provided by /proc/<pid>/maps. It displays one line for each VMA of the process.
  - virtual start address in hexadecimal (same as /proc/[pid]/maps)
  - total number of pages in the region in decimal
  - of the total number of pages in the region, the number that are resident
  - of the resident pages, the number which are shared with other processes, i.e. which have a usage count greater than one
  - of the resident pages, the number which are dirty (modified)
  - permissions (same as /proc/[pid]/maps)
  - file byte offset in hex (same as /proc/[pid]/maps)
  - file name (same as /proc/[pid]/maps)

Example & Usage

- Example

```
% cat /proc/760/statrm
00008000     129     109     109      0 r-xp 00000000  /devel/usr/bin/bash
00091000       6       6       0      0 rw-p 00081000  /devel/usr/bin/bash
00097000      39      38       0      0 rwxp 00097000
40000000      21      21      21      0 r-xp 00000000  /devel/lib/ld-2.3.3.so
40015000       1       1       0      0 rw-p 40015000
4001d000       1       1       0      0 rw-p 00015000  /devel/lib/ld-2.3.3.so
4001e000      54      29      29      0 r-xp 00000000  /devel/usr/lib/libncurses.so
```



Usage

Followings are example of guesses from statistics

- # of resident << # of region pages
  possible untouched code or data
- In Data/Bss, # of shared is close to # of resident
  possible COW and RO

For detail, need to check with /proc/<pid>/pgstat or /proc/<pid>/memmap

/proc/<pid>/pgstat

- Show detailed status per page in process
- address: [RS], [COW Zero shared], [COW shared], [shared], [Swap cache], [Swap area], [Active/Inactive], [Dirty/Clean], [Reserved/Unreserved], [Locked/Unlocked]

  – R = Resident S = Swapped
  – - ==> would indicate Not applicable
  – 1 ==> true when condition is boolean and 0 otherwise

  – E.g.

```
40000000:R - - 1 - - 1 0 0 0
40001000:R - - 1 - - 1 0 0 0
40002000:R - - 1 - - 1 0 0 0
40003000:R - - 1 - - 1 0 0 0
40004000:R - - 1 - - 1 0 0 0
…
```

/proc/<pid>/memmap

- This entry provides detailed information about whether pages are resident and how much shared within each region of the process.
  - Example output

```
% cat /proc/760/memmap
2
1
1 1 1
5 5 5 5 5 5 5 5 5 5 5 5 - - - - - 5 - - - - 5 -
1
1 1
- 1
5 5 5 5 5 5 5 5 5 5 5 5 5 5 - - - - - - - - - - - - - - - 5
  5 5 - - - - - - - - - - - - - - - - - - - - - - - - - 5 - - -
 - - - - - - - - - - - - - - - - - - - - - - - - - - - - - - -
```

…

/proc/memmap

- This entry provides a global view of the free and allocated physical memory pages.
- It provides data in a bit map format, one bit represents a single page frame. A one means the page is allocated, a zero means the page is free.

- % cat /proc/memmap

```
High water used pages =1991      Low water used pages 853
f1fffffffffffffffffffffffffffffffffffffffffffffffffffffffffffff
fffffffffffffffffffffffffffffffffffffffffffffffffffffffffffffff
fffffffffffffffffffffffffffffffffffffffffffffffffffffffffffffff
ffffffffffffffddffffffff7501dff7ffffdfffffffffdfffffffffffd5d
00000000000000000000000000000000000000000000000000000000000000
00000000000000000000000000000000000000000000000000000000000000
00000000000000000000000000000000000000000000000000000000000000
00000000000000000000000000000000000000000000000000000000000000
00000000000000000000000000000000000000000000000000000000000000
.....
```

Accurate memory counting API

- See http://tree.celinuxforum.org/CelfPubWiki/AccurateMemoryMeasurement
- Use case
  - Estimate room of memory at runtime
  - Refrain fro activating new application if current room cannot satisfy it.
- Implementation
  - Original implementation for 2.4 kernel by Panasonic
    - ARM specific system call
  - Ported to 2.6.11 by Sony
    - Architecture neutral /proc interface

```
% cat /proc/freemem
MemFree Total:          843776 B
MemFrey Total:             824 kB
Inactive Shrinkable:        84 kB
Active Shrinkable:          20 kB
Cached Shrinkable:         396 kB
```

Algorithm - quote from CELF wiki

- When the is API invoked:
  - Get the number of free pages using nr_free_pages()
  - Get the number of shrinkable page cache by inspecting active- and inactive- page cache list, and counting pages that can be free'ed.
    - The inspection logic is basically same as shrink_cache(). The Difference is whether pages are actually free'ed or not.
  - Get the number of pages in slab free list.
  - Get the number of i-node cache and directory entry cache. We do not inspect the status of those caches in detail for saving time.