

Linux Kernel Project 1

Team : 第五組

Members :

111552013 李德泰

111552019 古倫全

111552016 陳家康

實作環境 :

Ubuntu 16.04

Linux Kernel 3.10.104

程式編譯 :

1. gcc cat.c -o cat.out
2. gcc getTaskInfo.c -o getTaskInfo.out

程式操作 :

1. 執行 cat.out，使 thread 程式持續在背景執行
2. 使用 ps -a 指令查詢 cat.out 的 PID
3. 執行 getTaskInfo.out <PID>，呼叫 System call 印出該 PID 的相關的資訊
4. 輸入 sudo dmesg 查看 VMA 資訊

Thread 小程式 :

額外建立 3 個 thread 與一個 Global 變數，一個 thread 將變數改為 0，一個 thread 將變數改為 1，最後一個 thread 將變數 print 出來，若變數等於 0 則輸出 "The cat is upstairs"，反之則輸出 "The cat is downstairs"。

System Call 邏輯 :

1. 以 for_each_process 找出指定 PID 的 task
2. 抓取此 Task 使用的 memory map 並儲存起來(code、data、stack 位置)
3. 以 while_each_thread 遍歷此 Task 下的所有 thread，將各 thread 的 stack、heap 位置儲存
4. 再由 mm 取得 mmap，進而讀取 VMA 資訊
5. 最後透過 copy_to_user 將結果回傳至 User space

執行結果：

```
ted@ubuntu: ~/Desktop/my code
ted@ubuntu:~/Desktop/my code$ ps -a
  PID TTY          TIME CMD
 1938 pts/11        00:00:00 cat.out
 2183 pts/12        00:00:00 ps
ted@ubuntu:~/Desktop/my code$ ./getTaskInfo.out 1938
return code is : 1
Task User Number : 4
Task start code address : 0x8048000
Task end code address : 0x8048aa0
Task start data address : 0x8049f00
Task end data address : 0x804a038
Task start bss address : 0x84fb000
Task end bss address : 0x851c000
Task start stack address : 0xbfb8b00

main thread PID : 1938
main thread stack pos : 0xe9d95d8c
main thread file pos : 0xc161cdf3

thread1 PID : 1939
thread1 stack pos : 0xe8899e84
thread1 file pos : 0xc161cdf3

thread2 PID : 1940
thread2 stack pos : 0xe889be84
thread2 file pos : 0xc161cdf3

thread3 PID : 1941
thread3 stack pos : 0xe889de84
thread3 file pos : 0xc161cdf3

ted@ubuntu:~/Desktop/my code$
```

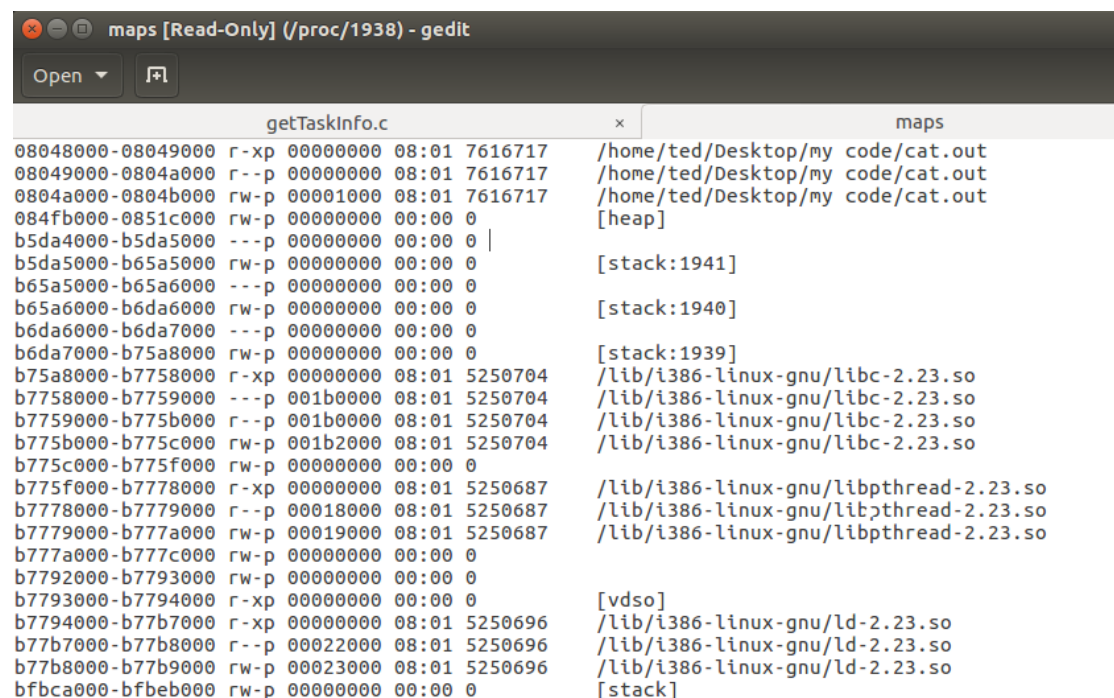
```
51.524681] VMA 0 : 0x8048000 ~ 0x8049000 /home/ted/Desktop/my code/cat.out
51.524682] VMA 1 : 0x8049000 ~ 0x804a000 /home/ted/Desktop/my code/cat.out
51.524683] VMA 2 : 0x804a000 ~ 0x804b000 /home/ted/Desktop/my code/cat.out
51.524683] VMA 3 : 0x84fb000 ~ 0x851c000 (null)
51.524684] VMA 4 : 0xb5da4000 ~ 0xb5da5000 (null)
51.524685] VMA 5 : 0xb5da5000 ~ 0xb65a5000 (null)
51.524685] VMA 6 : 0xb65a5000 ~ 0xb65a6000 (null)
51.524686] VMA 7 : 0xb65a6000 ~ 0xb6da6000 (null)
51.524687] VMA 8 : 0xb6da6000 ~ 0xb6da7000 (null)
51.524687] VMA 9 : 0xb6da7000 ~ 0xb75a8000 (null)
51.524688] VMA 10 : 0xb75a8000 ~ 0xb7758000 /lib/i386-linux-gnu/libc-2.23.so
51.524689] VMA 11 : 0xb7758000 ~ 0xb7759000 /lib/i386-linux-gnu/libc-2.23.so
51.524690] VMA 12 : 0xb7759000 ~ 0xb775b000 /lib/i386-linux-gnu/libc-2.23.so
51.524691] VMA 13 : 0xb775b000 ~ 0xb775c000 /lib/i386-linux-gnu/libc-2.23.so
51.524692] VMA 14 : 0xb775c000 ~ 0xb775f000 (null)
51.524693] VMA 15 : 0xb775f000 ~ 0xb7778000 /lib/i386-linux-gnu/libpthread-2.23.so
51.524693] VMA 16 : 0xb7778000 ~ 0xb7779000 /lib/i386-linux-gnu/libpthread-2.23.so
51.524694] VMA 17 : 0xb7779000 ~ 0xb777a000 /lib/i386-linux-gnu/libpthread-2.23.so
51.524695] VMA 18 : 0xb777a000 ~ 0xb777c000 (null)
51.524696] VMA 19 : 0xb7792000 ~ 0xb7793000 (null)
51.524696] VMA 20 : 0xb7793000 ~ 0xb7794000 (null)
51.524697] VMA 21 : 0xb7794000 ~ 0xb77b7000 /lib/i386-linux-gnu/ld-2.23.so
51.524698] VMA 22 : 0xb77b7000 ~ 0xb77b8000 /lib/i386-linux-gnu/ld-2.23.so
51.524699] VMA 23 : 0xb77b8000 ~ 0xb77b9000 /lib/i386-linux-gnu/ld-2.23.so
51.524700] thread end : 0xe9d95d8c
ted@ubuntu:~/Desktop/my code$
```

執行結果說明：

透過 Task struct 取得的 memory map 為該 process 執行的記憶體資訊。Task User Number 代表此空間有被多少數量的 thread 所共有，印出的結果為 4 是由於 cat.out 除了額外撰寫的 3 個 thread 還有本身的 main 的執行序，也代表著這個記憶體空間為該 procrss 下所有 thread 都可以進行存取。

讀取 VMA 並拋回到 User Space 出了點錯誤沒有時間修正，故這邊以 dmesg 將 VMA 資料呈現。VMA 的輸出格式分別為起始位置、結束位置、實體檔案路徑。

VMA 0	Code Segment
VMA 1 & 2	Data Segment
VMA 3	BSS Segment
VMA 10 ~ 13	Library Segment
VMA 15~ 17	Thread Stack Segment
VMA 21 ~ 23	Dynamic Loader Segment



getTaskInfo.c		x	maps
08048000-08049000	r-xp 00000000 08:01 7616717		/home/ted/Desktop/my code/cat.out
08049000-0804a000	r--p 00000000 08:01 7616717		/home/ted/Desktop/my code/cat.out
0804a000-0804b000	rw-p 00001000 08:01 7616717		/home/ted/Desktop/my code/cat.out
084fb000-0851c000	rw-p 00000000 00:00 0		[heap]
b5da4000-b5da5000	---p 00000000 00:00 0		
b5da5000-b65a5000	rw-p 00000000 00:00 0		[stack:1941]
b65a5000-b65a6000	---p 00000000 00:00 0		
b65a6000-b6da6000	rw-p 00000000 00:00 0		[stack:1940]
b6da6000-b6da7000	---p 00000000 00:00 0		
b6da7000-b75a8000	rw-p 00000000 00:00 0		[stack:1939]
b75a8000-b7758000	r-xp 00000000 08:01 5250704		/lib/i386-linux-gnu/libc-2.23.so
b7758000-b7759000	---p 001b0000 08:01 5250704		/lib/i386-linux-gnu/libc-2.23.so
b7759000-b775b000	r--p 001b0000 08:01 5250704		/lib/i386-linux-gnu/libc-2.23.so
b775b000-b775c000	rw-p 001b2000 08:01 5250704		/lib/i386-linux-gnu/libc-2.23.so
b775c000-b775f000	rw-p 00000000 00:00 0		
b775f000-b7778000	r-xp 00000000 08:01 5250687		/lib/i386-linux-gnu/libpthread-2.23.so
b7778000-b7779000	r--p 00018000 08:01 5250687		/lib/i386-linux-gnu/libpthread-2.23.so
b7779000-b777a000	rw-p 00019000 08:01 5250687		/lib/i386-linux-gnu/libpthread-2.23.so
b777a000-b777c000	rw-p 00000000 00:00 0		
b777c000-b7793000	rw-p 00000000 00:00 0		
b7793000-b7794000	r-xp 00000000 00:00 0		[vdso]
b7794000-b77b7000	r-xp 00000000 08:01 5250696		/lib/i386-linux-gnu/ld-2.23.so
b77b7000-b77b8000	r--p 00022000 08:01 5250696		/lib/i386-linux-gnu/ld-2.23.so
b77b8000-b77b9000	rw-p 00023000 08:01 5250696		/lib/i386-linux-gnu/ld-2.23.so
bfbca000-bfbcb000	rw-p 00000000 00:00 0		[stack]

和/prop/<pid>/maps 的結果相對應可以發現，VMA 5、7、9 分別代表了各 thread stack 的位置，這些無實體位置的 Virtual memory area 可解讀為是不被其他 thread 所共用的。

Virtual Memory Figure :

