

Term Project2:

Physical Page Map Profiling

Due: 5th Dec. (TUE), 11:59 PM

Submit on I-Campus.

1. Project Introduction

You should implement a kernel module to get the physical page map of all processes via /proc.

You can refer to the data structures for the memory management in Linux Kernel.

2. Problem Specification

2.1 Environment

2.1.1 Ubuntu environment (kernel version 4.9.x)

2.1.2 You can install Ubuntu or use virtual machine for the project

2.1.3 If you use the virtual machine, allocate enough memory to the machine

2.2 Analyze the core data structures related to the projects

Your report should describe the relationship among the following kernel data structures and the management scheme of page table in Linux kernel.

: task_struct, vm_area_struct, mm_struct, file

2.3 Make physical page map

2.3.1 Implement your own loadable kernel module

```
$ ls /proc/
96 18108 213 213 265 38 3970 4036 4114 4258 4380 463 6 7512 917
58 18244 204 221 278 3839 3988 4054 4143 4272 44 4764 61 76 935
81 18324 205 222 28 3840 3993 4056 4161 4277 4402 48 62 77 941
79 18331 2053 24 288 3846 3999 4069 4162 4282 4409 49 63 78 944
71 18386 206 247 30 3849 4 4071 4192 43 4413 50 64 8 954
18 18821 207 248 31 39 40 4094 4195 4309 4420 505 65 81 955
19 18822 208 249 312 3928 4005 4103 42 4311 4452 51 66 82 958
87 19 209 25 32 3929 4007 4104 4210 4335 4462 52 67 83 962
93 2 21 251 33 394 4012 4110 4220 4340 4464 5290 68 9 985
37 20 210 256 34 3941 4020 4111 4222 4348 45 54 7 908 992
82 201 211 26 36 3959 4025 4112 4239 4356 4523 55 74 911 acpi execdomains kpagecgroup partitions
```

Figure 1 New node 'PhysicalPageMap' in proc filesystem

2.3.2 Add a file node named "PhysicalPageMap" in the /proc filesystem

2.3.3 Print the following information about a page per line for all processes within the file

"PhysicalPageMap"

- Process ID
- Process Name
- Virtual Page Address
- Physical Page Frame Number (PPN)
- VMA Range (start address ~ end address)

VMA Size (in Byte)

VMA Name (Classify the VMA into Heap, Stack, Data, Code, or Others[=(null)])

File Name (Mapped to the memory region)

```
[pid] 18324 [process_name] gnome-terminal [virt] 4448256 [phys] 3438685 [range] 4194304 - 4493312 (299008 bytes) [vname] (null) [fname] gnome-terminal-server
[pid] 18324 [process_name] gnome-terminal [virt] 4452352 [phys] 3925222 [range] 4194304 - 4493312 (299008 bytes) [vname] (null) [fname] gnome-terminal-server
[pid] 18324 [process_name] gnome-terminal [virt] 6586368 [phys] 2458758 [range] 6586368 - 6590464 (4096 bytes) [vname] data [fname] gnome-terminal-server
[pid] 18324 [process_name] gnome-terminal [virt] 6590464 [phys] 3942192 [range] 6590464 - 6602752 (12288 bytes) [vname] data [fname] gnome-terminal-server
[pid] 1099 [process_name] nvidia-persiste [virt] 4210688 [phys] 4282565 [range] 4194304 - 4227072 (32768 bytes) [vname] (null) [fname] nvidia-persistenced
[pid] 1099 [process_name] nvidia-persiste [virt] 4214784 [phys] 4282564 [range] 4194304 - 4227072 (32768 bytes) [vname] (null) [fname] nvidia-persistenced
[pid] 1099 [process_name] nvidia-persiste [virt] 4218880 [phys] 4233803 [range] 4194304 - 4227072 (32768 bytes) [vname] (null) [fname] nvidia-persistenced
[pid] 1099 [process_name] nvidia-persiste [virt] 4222976 [phys] 4233802 [range] 4194304 - 4227072 (32768 bytes) [vname] (null) [fname] nvidia-persistenced
[pid] 1099 [process_name] nvidia-persiste [virt] 6324224 [phys] 4239410 [range] 6324224 - 6328320 (4096 bytes) [vname] data [fname] nvidia-persistenced
[pid] 1099 [process_name] nvidia-persiste [virt] 10133504 [phys] 4254254 [range] 10133504 - 10268672 (135168 bytes) [vname] heap [fname] (null)
[pid] 1549 [process_name] bluetoothd [virt] 350774624256 [phys] 4159428 [range] 350773755904 - 350774853632 (1097728 bytes) [vname] (null) [fname] bluetoothd
[pid] 1549 [process_name] bluetoothd [virt] 350774628352 [phys] 4159427 [range] 350773755904 - 350774853632 (1097728 bytes) [vname] (null) [fname] bluetoothd
[pid] 1549 [process_name] bluetoothd [virt] 350774632448 [phys] 4159426 [range] 350773755904 - 350774853632 (1097728 bytes) [vname] (null) [fname] bluetoothd
[pid] 1549 [process_name] bluetoothd [virt] 350774636544 [phys] 4159425 [range] 350773755904 - 350774853632 (1097728 bytes) [vname] (null) [fname] bluetoothd
[pid] 1549 [process_name] bluetoothd [virt] 350774640640 [phys] 4159424 [range] 350773755904 - 350774853632 (1097728 bytes) [vname] (null) [fname] bluetoothd
[pid] 18331 [process_name] bash [virt] 7344224 [phys] 3654014 [range] 7327744 - 7352320 (24576 bytes) [vname] (null) [fname] (null)
[pid] 18331 [process_name] bash [virt] 7348224 [phys] 3764560 [range] 7327744 - 7352320 (24576 bytes) [vname] (null) [fname] (null)
[pid] 18331 [process_name] bash [virt] 39342080 [phys] 3033954 [range] 39342080 - 41279488 (1937408 bytes) [vname] heap [fname] (null)
[pid] 18331 [process_name] bash [virt] 39346176 [phys] 3164646 [range] 39342080 - 41279488 (1937408 bytes) [vname] (null) [fname] (null)
[pid] 919 [process_name] dbus-daemon [virt] 969147822080 [phys] 4260494 [range] 969147715584 - 969147932672 (217088 bytes) [vname] (null) [fname] dbus-daemon
[pid] 919 [process_name] dbus-daemon [virt] 969147826176 [phys] 4261316 [range] 969147715584 - 969147932672 (217088 bytes) [vname] (null) [fname] dbus-daemon
[pid] 919 [process_name] dbus-daemon [virt] 969147830272 [phys] 4257435 [range] 969147715584 - 969147932672 (217088 bytes) [vname] (null) [fname] dbus-daemon
[pid] 919 [process_name] dbus-daemon [virt] 969147834368 [phys] 4251853 [range] 969147715584 - 969147932672 (217088 bytes) [vname] (null) [fname] dbus-daemon
[pid] 919 [process_name] dbus-daemon [virt] 969147838464 [phys] 4251882 [range] 969147715584 - 969147932672 (217088 bytes) [vname] (null) [fname] dbus-daemon
[pid] 919 [process_name] dbus-daemon [virt] 969147842560 [phys] 4241377 [range] 969147715584 - 969147932672 (217088 bytes) [vname] (null) [fname] dbus-daemon
[pid] 919 [process_name] dbus-daemon [virt] 969147846656 [phys] 4241276 [range] 969147715584 - 969147932672 (217088 bytes) [vname] (null) [fname] dbus-daemon
[pid] 13379 [process_name] chrome [virt] 140730241822720 [phys] 3031739 [range] 140730241822720 - 140730241835008 (12288 bytes) [vname] (null) [fname] (null)
[pid] 13379 [process_name] chrome [virt] 140730241835008 [phys] 3030887 [range] 140730241835008 - 140730241843200 (8192 bytes) [vname] (null) [fname] (null)
[pid] 13380 [process_name] cat [virt] 4194304 [phys] 4276705 [range] 4194304 - 4243456 (49152 bytes) [vname] code [fname] cat
[pid] 13380 [process_name] cat [virt] 4198400 [phys] 4278666 [range] 4194304 - 4243456 (49152 bytes) [vname] (null) [fname] cat
```

Figure 2 Page Map example

2.3.4 Reference

```
(fs/proc/task_mmu.c)
proc_pagemap_operations (pagemap_read, pagemap_open)
proc_pid_smaps_operations (pid_smaps_open)
proc_pid_smaps_op (show_pid_smap)
```

2.4 Display the Page Map and analyze the result

- 2.4.1 Implement your own user application which allocates at least 5MB of memory with malloc() and also maps a file into memory with mmap().
- 2.4.2 While running your application (2.4.1), print the file 'PhysicalPageMap' into the file 'os_pagemap.txt' like Figure 2. The result must include the information about your application.
- 2.4.3 Draw the memory footprint of **systemd** process (pid 1) and your application like Figure 3, and attach the graph to the report (y-axis: process ID, x-axis: PPN)
- 2.4.4 You can use GNU plot or Excel to draw the memory footprint.

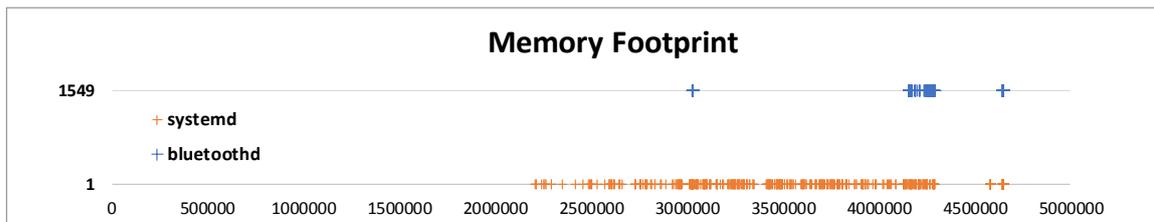


Figure 3 Memory Footprint Example

3. Evaluation Policy

- 3.1 Analyze the relationship among the data structures mentioned in section 2.2.1 (5 point)
- 3.2 Describe the management scheme of page table in Linux Kernel (10 point)
- 3.3 Implement Loadable Kernel Module (5 point)
- 3.4 Create the file node of "PhysicalPageMap" in the /proc filesystem (5 point)
- 3.5 Print the process IDs and process names (5 point)
- 3.6 Print the virtual page addresses, VMA Ranges, and VMA sizes (10 point)
- 3.7 Print the VMA names (10 point)
- 3.8 Print the Physical Page Frame Numbers (20 point)
- 3.9 Print the file name mapped to each memory region (10 point)
- 3.10 Draw the memory footprint of two processes (10 point)
- 3.11 Report (10 point)

Job	Data structure analysis	Page table management description	LKM impl.	New file 'PhysicalPageMap'	Process ID Process Name		
Score	5	10	5	5	5		
Job	Virtual Page Address VMA Range VMA Size	VMA Name	PPN	File Name	Memory Footprint	Report	Total
Score	10	10	20	10	10	10	100

4. Submission

- The final output should be a compressed zip file named **[student_id]_proj2.zip**, which includes **[student_id]_report.pdf**, **your_module_code.c**, **Makefile_for_module**, **your_application_code.c** and **os_pagemap.txt**
- Please describe the key parts of your code in the report.
Don't attach the whole source codes in the report. Just key parts. (you will submit your codes too.)
- Please submit on i-Campus.

5. Notice

- You should do the term project for yourself
- If you have a question, send email to jjysienna@gmail.com with the title of [CSE3047-41] Term project1 Question, [Student ID], [Your Name]
- Your email questions should describe the problem in detail with screenshots.

- If you cheat on the project, you will finally fail this course. (F grade)
 - If I find two similar source codes or reports, both students will fail this course.
- You will get -15 points per one day delay.
- No need to submit after 3 days (We will not get more submissions after 3 days)

Have fun!

OS Term Project