Assignment 3: Memory Management and Introduction to Multithreading

---

Goal of the Assignment:

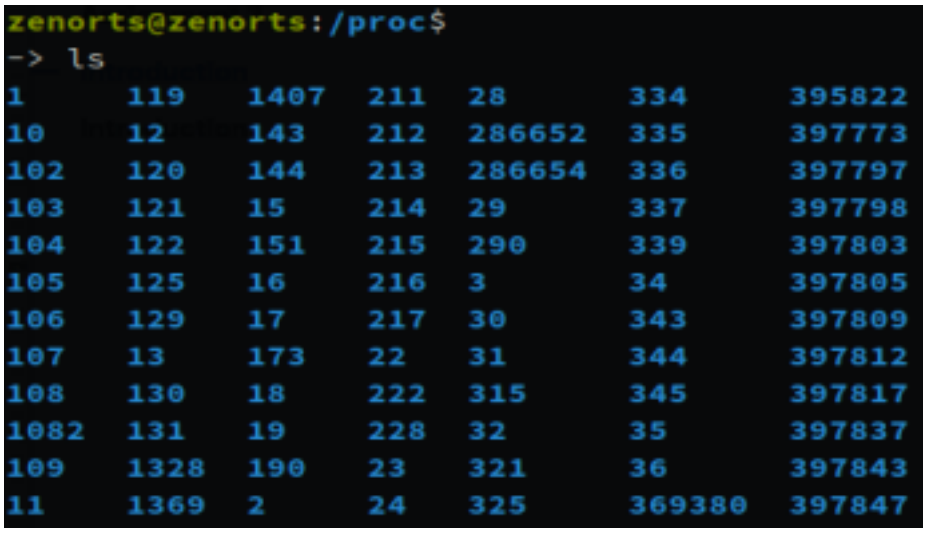
- To make you familiar with virtual address, physical address, and address translation for memory management.
- To learn and understand how threads use common data and run in parallel.

Before you start:

When we run a program and print the memory address, the printed address is just a virtual address provided by the operating system. Students need to know how these virtual address and physical address are in correlation with each other. As this will help build an understanding of the steps involved in a virtual address to physical address mapping.

Just to give a brief overview of things that we are going to do in this assignment. There are a few steps involved in this process.

1. Change directory to “cd /proc” then do “ls” This will result in showing some things like shown in the figure below



These are all process IDs that are currently available in the OS. Here these folders are just a mere representation of processes and what data they possess during execution. 2. Now change the directory to any process id, “cd PID_number”. This way you will get to

see a lot of files present in the directory related to the process presently running. We are only interested in two files, **(1) maps and (2) pagemap.**

3. The concept behind these files is very simple to understand. First let's look at the maps file. (Shown in figure below)

```
zenorts@zenorts:/proc/1$
-> sudo cat maps
564797b21000-564797b53000 r--p 00000000 08:02 54789795                   /usr/lib/systemd/systemd
564797b53000-564797c11000 r-xp 00032000 08:02 54789795                   /usr/lib/systemd/systemd
564797c11000-564797c67000 r--p 000f0000 08:02 54789795                   /usr/lib/systemd/systemd
564797c67000-564797cad000 r--p 00145000 08:02 54789795                   /usr/lib/systemd/systemd
564797cad000-564797cae000 rw-p 0018b000 08:02 54789795                   /usr/lib/systemd/systemd
```

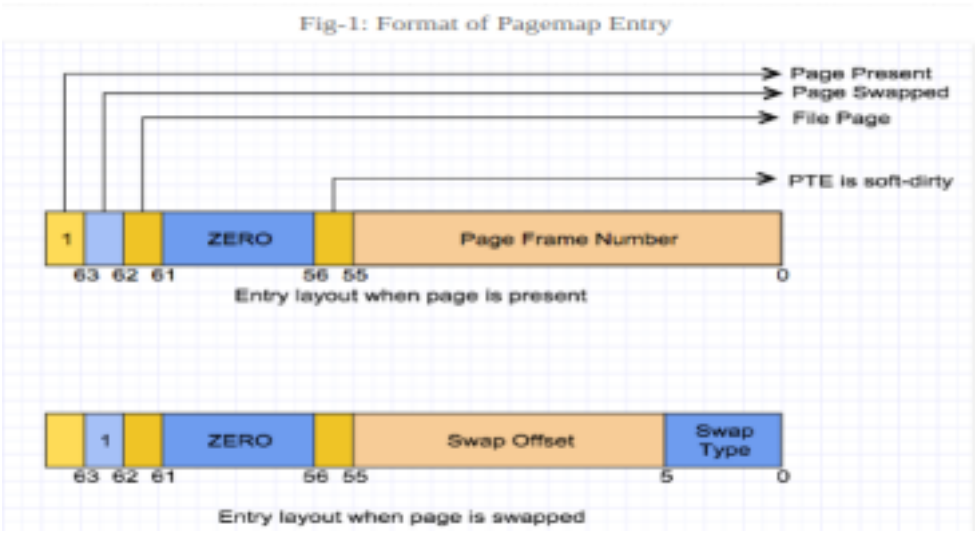
Each row in /proc/$PID/maps describes a region of contiguous virtual memory in a process or thread. Each row has the following fields:

address perms offset dev inode pathname 08048000-08056000 r-xp 00000000 08:02

5464593 /usr/sbin/abc

More details about this can be found at below link: https://man7.org/linux/man pages/man5/proc.5.html

4. Now let's dig into the pagemap file. Try not to open this file as it will take forever to open, in a few cases it may cause your OS to restart, especially if you are using a resource constrained device or VM. The file is basically having collection of page table entries mapping to physical addresses.
5. Page entry format as shown in figure.



Pagemap is a set of interfaces in the kernel that allow user space programs to examine the page tables and related information by reading files in /proc. You can follow

/proc/pid/pagemap, This file lets a user space process find out which physical frame each virtual page is mapped to. It contains one 64-bit value for each virtual page, containing the following data:

a. * Bits 0-54 page frame number (PFN) if present
b. * Bits 0-4 swap type if swapped
c. * Bits 5-54 swap offset if swapped
d. * Bit 55 pte is soft-dirty (see Documentation/vm/soft-dirty.txt)
e. * Bit 56 page exclusively mapped (since 4.2)
f. * Bits 57-60 zero
g. * Bit 61 page is file-page or shared-anon (since 3.5)
h. * Bit 62 page swapped
i. * Bit 63 page present

To know more about page map you can follow the link(Pagemap details)

Part A: Virtual address to Physical address Translation [Total: 10 Points]

Once you are familiar with the concept and usage of maps and pagemap. Now, write a program/script that takes PID as a command-line argument and prints every starting virtual address in **/proc/[pid]/maps** file to a corresponding page frame number using **/proc/[pid]/pagemap** file. As enough explanation is provided on pagemap and maps, students are free to use any online explanation as a reference as well.

Part B: Extending Virtual Address Translation [Total: 10 Points]

Once the first part is over students can start to utilize pagemap files for accessing more information. In this section, students will create another program that will be based on the following points:

- Students are expected to write a program/script that takes PID and virtual address as a command-line arguments for a process and translate it to corresponding physical address along with all auxiliary information encoded in the 64-bit PTE. Students must

print all the relevant information in proper format.

Help: You can refer to the "Pagemap Interface of Linux Explained" blog. It provides a similar python script.

Part C: Basic Thread Execution using Pthread library in C [Total: 8 Points]

Write a multithreaded program that calculates various statistical values for a list of numbers. This program will be passed a series of numbers on the command line and will then create three separate worker threads. One thread will determine the average of the numbers, the second will determine the maximum value, and the third will determine the minimum value.

The variables representing the average, minimum, and maximum values will be stored globally. The worker threads will set these values, and the parent thread will output the values once the workers have exited.

Sample Input/Output:

$ Enter the elements: // this is printed by the parent thread.
$ 90 81 78 95 79 72 85

The remaining outputs are printed by the worker threads.
$ The average value is 82 and the thread id is :<thread_id>
$ The minimum value is 72 and the thread id is :<thread_id>
$ The maximum value is 95 and the thread id is :<thread_id>

Help: You can refer to the https://www.geeksforgeeks.org/multithreading-c-2/ blog to understand how to write and compile multithreaded program in C.

Deliverables in a tar ball on GC:

- Submission Guidelines: Upload the assignment report, code files, README, etc in GC as a tar ball with file name as <your roll no>_<your name>.tar
- Readable Report [2 points for report quality] enumerating steps followed with screenshots for each of the important steps. Put the screenshots in the report for better clarity.

- You can use any (your favorite) language for Part A and Part B.

Check Web sources for more information