Statutory note

- This course is on a no plagiarism diet.
- All parties involved in plagiarism harakiri will be penalized to the maximum extent.
- `The moss detective agency` has agreed to conduct all investigations. https://theory.stanford.edu/~aiken/moss/
  Byomkesh, Sherlock and Hercule are on standby.

0. Access to a machine with any Linux distribution and root access.

1. Hello LKM!

A Linux kernel module is a piece of code (kernel functionality) that can be loaded and unloaded into the kernel on demand. Kernel modules extend the functionality of the kernel without the need to reboot the system and are typically used for on-demand loading of drives to handle plug-and-play devices, add custom system call like binary functionality to the kernel etc.

Follow the link https://linux-kernel-labs.github.io/refs/heads/master/labs/kernel_modules.html
to get pointers on how to write, load and unload linux kernel modules.

Make sure you have a stable Linux OS distribution installed on your machine. Following are the commands for installing packages on Ubuntu for building and installing kernel modules

sudo apt install linux-headers-$(uname -r) build-essential

Make sure that you implement the hello world kernel module which prints a message when the module is loaded and an exit message when the kernel is unloaded.

2. Peeking into Linux

Implement the following functionalities to peek into the Linux kernel implementation.

a. Write a kernel module to list all processes either in running or runnable state and then compare it with the output from ps command.
b. Write a kernel module to print the **pid** of the task which has the largest in use heap memory. Print the **pid** and the amount of heap memory as output.
c. Write a kernel module to print the kernel stack pointer of the task with pid 1.
d. Given a process ID and a virtual address in its address space, determine if it is mapped and if so determine its physical address (pseudo physical address if running in a VM).
   Print the **pid**, virtual address and corresponding physical address. Note that the kernel virtual address mappings will be same across processes, and for 64 bit systems (only 48 bits used for addressing) the kernel virtual address space starts at virtual address **ffff_8000_0000_0000** and goes upto **ffff_ffff_ffff_ffff** inside each process address space.
e. For a specified process (via its pid), determine the size of the allocated virtual address space (sum of all vmas) and the physical address space. Does Linux implement lazy memory allocation and if so demonstrate via your LKM implementation.

Following is a list of important Linux kernel data structures/definitions that will be useful:
task_struct
mm_struct
vma_struct
current

References

Following is a good source for easily navigating through the kernel source code

https://elixir.bootlin.com/linux/latest/source/kernel

Submission instructions:

Add all files in a directory with the following format …
/<rollno-cs695-a2>
    |_ lkm1.c
    |_ lkm2.c
    |_ lkm3.c
    |_ lkm4.c
    |_ lkm5.c
    |_ README (Paste output of all LKMs in this file explaining the rationale behind the output).

Upload the zipped file `<rollno-cs695-a2>.tar.gz` of the submission folder on Moodle.

Deadline: 27th Jan 2023, 5pm