OS Assignment 3 - Report for Q3

By: Arnav Goel (2021519), Section B, Branch: CSAI

In this question, we are asked to implement a syscall as a kernel module for taking a running process as input from the command line, reading its **task_struct** and printing out the following fields: PID (Process ID), PGID (process group ID), UID (user ID) and command path.

I referred to Tutorial 5 and the references given there to understand how the kernel module works and what all header files we need, the need for a **__init() and a __exit()** method and the commands for loading and unloading a module.

I have defined a method in my module called process_info_formatter, which returns 0 if there are no errors and returns some other integer value in case of errors. I used the **<linux/moduleparam.h>** header for taking input from the user. I am taking the PID of the process as an integer input and typecasting it to the pid_t type. Then I used the **find_vpid()** syscall for finding and returning a pointer to the task struct of the corresponding process.

Once I have a pointer to the task_struct of the process, I print its name, PID, directly. To print the PPID and PGID, I need to go into the **real_parent and group_leader** portions of the task_struct, respectively. For printing the UID, I need to include the header file **<linux/cred.h>** to go into the **cred** portion of the task_struct and print the UID from there.

For printing, I **kmalloc memory** in the kernel, and I freed it using **kfree()** in the method itself. I then called this function in the __init function, which means that this method will be called whenever the module is loaded.

Thus the commands for running the module are:

```
make
sudo insmod kernel_module.ko pid_input=1
sudo dmesg
sudo rmmod kernel_module.ko
```

- make will compile the module using the submitted makefile
- insmod will load the module, not for taking inputs from the command line in kernel modules, we need to pass in the argument with the name of the variable it was declared inside with thus we pass in 1 to pid_input in this example.
- dmesg shows us the kernel buffer with all the process details for PID=1
- rmmod then simply unloads the kernel module. We can make clean to clean all the files made when we called “make.”