

# CSCI 5573: Advanced Operating Systems

## Fall 2019

### Homework Two

*Due Date and Time: 11:55 PM, Monday, October 14, 2019*

The goal of this homework is to get familiar with Linux Kernel Modules as well as some kernel debugging tools.

You must work in a virtual machine for this assignment, otherwise you may corrupt your operating system. You will need super user privileges to do this assignment.

You may work in teams of size two students.

#### 1. Inserting a new system call using LKM

Write a kernel module to add a new system call *getprocinfo(pid\_t pid, struct procinfo \*inf)* that returns the following data about the specified process and prints it on the standard output:

```
struct procinfo {
    pid_t pid;           /* Process ID */
    pid_t ppid;          /* Parent process ID */
    struct timespec start_time; /* Process start time */
    int num_sib;          /* Number of siblings */
};
```

You can find all of the information you need in *task\_struct*, which is defined in *include/linux/sched.h*. If *pid* is greater than 0, it represents a processID whose information is sought. If it's zero, return the information for the current process. If it's less than zero, return the information for the parent of the current process. As always, check inputs for validity. Return error EINVAL if necessary. Report the time it takes to execute your system call.

#### 2. Experience with rootkits and countermeasures

The goal of this exercise is to gain experience with rootkits and how to counter them. A rootkit is a collection of computer software, typically malicious, designed to enable access to a computer or areas of its software that is not otherwise allowed (for example, to an unauthorized user) and often masks its existence or the existence of other software. A common way rootkits work is by manipulating systems calls.

(a) Your first task is to write a very simple rootkit module that affects *ls* such that files/folders whose name begin with prefix *abc* are not listed when *ls* is run. These files/folders should still be available and you should be able to access them, e.g. by using an editor.

To write this rootkit, you will first identify the system call that *ls* uses to find the file listings. Use a tool called *strace* to do this. Next, you will write an LKM to intercept that system call. Use the methodology discussed in class to do this.

Show your work about how you identified the target system call. Further, identify at least one other command that is affected by this rootkit? How is that command affected?

## CSCI 5573: Advanced Operating Systems

### Fall 2019

(b) While this rootkit hides all files whose names start with *abc*, the name of the rootkit (the LKM module you built) is still visible, e.g. by running *lsmod* command. Your second task is to extend your rootkit so that rootkit is no longer visible from *lsmod*.

(c) Since system call manipulation is a common techniques utilized by rootkits, one useful tool would be to identify every system call that is being intercepted. So, your third task is to build a kernel module that can identify all system calls that are being intercepted as well as all new system calls introduced in the kernel after your new kernel module has been installed in the kernel. When invoked from the user space (as a new system call), this tool should print out a list of all such system calls.

(d) A key concern with this kernel module is that a rootkit could intercept it and modify it. Explain how this could be done.

(e) Your final task is to address the concern of part (d) by building a kernel module and a utility function *check\_sys\_calls*. The utility function simply loads the kernel module in the kernel and then removes the module from the kernel. When the kernel module is loaded for the first time, it records the current state of all system calls. Afterwards, whenever this kernel module is loaded, it reports all system calls that are being intercepted as well as all new system calls introduced in the kernel after the kernel module was first loaded in the kernel.