**ĐẠI HỌC QUỐC GIA THÀNH PHỐ HỒ CHÍ MINH**

**TRƯỜNG ĐẠI HỌC KHOA HỌC TỰ NHIÊN**

**KHOA CÔNG NGHỆ THÔNG TIN**

**~~~~~~~~~**



**BÁO CÁO ĐỒ ÁN 2**

**MÔN: HỆ ĐIỀU HÀNH**

*Sinh viên thực hiện*

**1712856**: Huỳnh Văn Tú

**1712210**: Nguyễn Xuân Vỹ

**TPHCM, ngày 03 tháng 11 năm 2019**

**1. Linux Kernel Module**

**1.1 Mục t iêu và yêu cầu**

- Mục tiêu hiểu về Linux kernel module và hệ thống quản lý file và device trong linux, giao tiếp giữa tiến trình ở user space và kernel space.

- Yêu cầu:

+ Viết một module dùng để tạo ra số ngẫu nhiên.

+ Module này sẽ tạo một character device để cho phép các tiền trình ở user space có thể open và read các số ngẫu nhiên.

**1.2 Tổ chức chương trình**

- Makefile: Build một module từ file code tên my\_mod\_rand.c

- test.c: Test yêu cầu bài toán.

- my\_mod\_rand.c: File code chính của chương trình. Gồm các thành phần chính:

+ static struct file\_operations fops: định nghĩa các hàm open, release, read cho character device.

+ static int \_\_init my\_rand\_init(void): Khởi tạo module, bao gồm đăng kí số hiệu, class và device.

+ static void \_\_exit my\_rand\_exit(void): Hủy class, device kết thúc module.

+ static int my\_rand\_open(struct inode \*, struct file \*): Hàm được gọi khi character device bị process ở userspace mở lên (open).

+ static int my\_rand\_release(struct inode \*, struct file \*): Hàm được gọi khi character device bị process ở userspace đóng lại (release).

+ static ssize\_t my\_rand\_read(struct file \*, char \*, size\_t, loff\_t \*): Hàm được gọi khi character device bị process ở userspace đọc (read). Sử dụng hàm get\_random\_bytes() và copy\_to\_user() để process ở userspace có thể lấy số ngẫu nhiên từ character device.

**2. Hook vào system call**

**2.1 Mục tiêu và yêu cầu**

Chương trình hook vào một system call:

+ syscall open =>ghi vào dmesg tên tiến trình mở file và tên file được mở.

+ syscall write => ghi vào dmesg tên tiến trình, tên file bị ghi và số byte được ghi.

**2.2 Tổ chức chương trình**

- Makefile: Build một module từ file code tên hook.c

- hook.c: File code chính của chương trình. Gồm các thành phần chính:

+ asmlinkage int new\_sys\_open(const char \_\_user \*, int, mode\_t): Viết lại system call open mới, ghi vào dmesg tên tiến trình mở file và tên file được mở.

+ asmlinkage int new\_sys\_write(unsigned int, const char \_\_user \*, size\_t): VIết lại system call write mới, ghi vào dmesg tên tiến trình, tên file bị ghi và số byte được ghi.

+ static int \_\_init init\_hook(void): Khởi tạo module. Lấy địa chỉ sys\_call\_table, sao lưu lại system call cũ và thay system call cũ bằng system call mới trong sys\_call\_table.

+ static void \_\_exit exit\_hook(void): Thay system call mới bằng system call cũ đã được sao lưu trước đó vào sys\_call\_table.

**Tài liệu tham khảo**

[1] Lập trình Linux Kernel Module (Moodle).

[2] Writing a Linux Kernel Module - Part 1: Introduce.

[3] Writing a Linux Kernel Module - Part 2: A Character Device.

[4] Hướng dẫn Hook 1 system call (Moodle).

[5] Basics of Making a Rootkit: From syscall to hook!

[6] How can I get a filename from a file descriptor inside a kernel module?

[7] Syscall Hijacking: Dynamically obtain syscall table address.