**BỘ MÔN HỆ THỐNG THÔNG TIN – KHOA CÔNG NGHỆ THÔNG TIN**

**ĐẠI HỌC KHOA HỌC TỰ NHIÊN THÀNH PHỐ HỒ CHÍ MINH, ĐẠI HỌC QUỐC GIA TP HCM**

Sinh viên thực hiện: 1712824 – Thái Vĩnh Toàn

GV phụ trách: Trần Trung Dũng

Đồ án/bài tập môn học  - HỆ ĐIỀU HÀNH

HỌC KỲ I – NĂM HỌC 2019-2020

**MÔn HỆ ĐIỀU HÀNH**

Mục lục

[**BẢNG THÔNG TIN CHI TIẾT NHÓM 2**](#_Toc24314046)

[**MỨC ĐỘ HOÀN THÀNH 2**](#_Toc24314047)

[**MÔ TẢ ĐỒ ÁN 2**](#_Toc24314048)

[**CHƯƠNG TRÌNH 3**](#_Toc24314049)

[**Tạo kernel module phát sinh số ngẫu nhiên 3**](#_Toc24314050)

[**Tạo chương trình ở user space gửi yêu cầu đến kernel 3**](#_Toc24314051)

[**Tài liệu tham khảo 4**](#_Toc24314052)

NHÓM

# BẢNG THÔNG TIN CHI TIẾT NHÓM

|  |  |  |  |  |
| --- | --- | --- | --- | --- |
| **MSSV** | **Họ tên** | **Email** | **Điện thoại** | **Hình ảnh** |
| 1712824 | Thái Vĩnh Toàn | 1712824@student.hcmus.edu.vn | 0397111575 |  |
|  |  |  |  |  |
|  |  |  |  |  |

# MỨC ĐỘ HOÀN THÀNH

|  |  |  |
| --- | --- | --- |
| **Công việc thực hiện** | **Người thực hiện** | **Hoàn thành** |
| Modul tạo số ngẫu nhiên | Thái Vĩnh Toàn - 1711814 | X |
| Tạo character device driver cho phép tiến trình ở user space open và read các số ngẫu nhiên được tạo | Thái Vĩnh Toàn - 1712824 | X |
|  |  |  |

BÁO CÁO

# MÔ TẢ ĐỒ ÁN

* Kernel Module là tên gọi chung của các đoạn code có khả năng nạp vào và gỡ ra khỏi kernel khi cần thiết mà không cần phải reboot lại toàn bộ hệ thống 🡪 Chính vì sự linh hoạt này mà nó còn có tên gọi khác là Loadable Kernel Module (LKM).
* Sự xuất hiện của LKM mang nhiều ý nghĩa lớn: giúp thu gọn kích thước của kernel, đồng thời đem đến sự linh hoạt khi thêm mới hoặc thay đổi driver trong hệ thống, đặc biệt đối các máy server. LKM được chia làm 3 loại chính: device driver, system call và file system. Mô tả một cách đơn giản, device driver là một trình điều khiển cho phép quản lý, giám sát việc vận hành một thiết bị phần cứng cụ thể gắn với máy tính (chẳng hạn bàn phím, chuột).
* Mỗi device driver cung cấp cho người dùng một giao diện gọi các hàm hệ thống system call đến tầng ứng dụng. Đây cũng là ranh giới giữa user space và kernel space
* Character Device Driver là device driver mà ở đó việc vận hành các thiết bị (chẳng hạn thiết bị đọc, viết) được thực hiện tuần tự từng byte một (byte-oriented). Một character device driver tiếp nhận một lời gọi systemcall từ user thông qua file thiết bị gọi là character device file, nói chính xác hơn là thông qua tên file của thiết bị đó.
* **Phần 1** của đồ án này tập trung tìm hiểu về cách xây dựng một LKM phát sinh số ngẫu nhiên; đồng thời tạo một character device driver cho phép các tiến trình ở **user space** *open* và *read* các số ngẫu nhiên đó.

# CHƯƠNG TRÌNH

* Cấu trúc chương trình sẽ gồm 2 tập tin:
  + Kernel module phát sinh số ngẫu nhiên
  + Chương trình gọi từ user space xuống kernel.

## Tạo kernel module phát sinh số ngẫu nhiên

* Để viết kernel module, ta cần biết nó được viết bằng C thuần và không thể sử dụng các thư viện truyền thống trong user space như stdlib.h. Chính vì vậy, Linux kernel sẽ đóng vai trò là thư viện, cung cấp các hàm và macro để ta phát triển kernel module. Các thư viện này sẽ có dạng <linux/tên\_thư\_viện.h>.
* Kernel module không có hàm main, nhưng vẫn tuân theo lập trình hướng đối tượng. Do đó, một kernel module bao gồm ít nhất 2 hàm cơ bản là constructor (có macro tương ứng là \_\_init) và destructor (có macro tương ứng là \_\_exit). Sau đó, cần truyền chúng vào module\_init và module\_\_exit để thực hiện khi lắp/tháo module vào/ra khỏi kernel.
* Để phát sinh số ngẫu nhiên, ta sử dụng hàm get\_random\_bytes nằm trong thư viện <linux/random.h>. Ngoài ra, để giới hạn số ngẫu nhiên phát sinh không vượt quá một giá trị MAX, ta thực hiện chia lấy dư số được phát sinh cho MAX
* Tạo các hàm toán tử muốn thực hiện đối với file thiết bị (ở đồ án này ta quan tâm tới toán tử open​, ​release​ và ​read​)

static struct file\_operations fops =

{

.open = dev\_open,

.release = dev\_release,

.read = dev\_read,

};

* Các hàm chính

static int \_\_init RandomGenerator\_init(void);

static void \_\_exit RandomGenerator\_exit(void);

## Tạo chương trình ở user space gửi yêu cầu đến kernel

* Viết chương trình đơn giản với hàm main thực hiện việc mở file device driver đã tạotrong folder ​/dev/<device\_name>​.
* Lưu ý: Chế độ mở file trong đồ án là *readOnly*

read(fd, &randNum, sizeof(randNum))

# Tài liệu tham khảo

* <https://www.youtube.com/watch?v=-O6GsrmOUgY&list=PL16941B715F5507C5&index=3>
* <https://uwnthesis.wordpress.com/2016/12/26/basics-of-making-a-rootkit-from-syscall-to-hook/>