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

Bộ môn: Hệ Điều Hành

1712856 - Huỳnh Văn Tú.

1712210 - Nguyễn Xuân Vỹ.

TỔNG QUAN

Tên thành viên

Huỳnh Văn Tú. Sinh viên lớp 17CNTT Cử nhân Tài năng. Mã số sinh viên: 1712856.

Nguyễn Xuân Vỹ. Sinh viên lớp 17CNTT Cử nhân Tài năng. Mã số sinh viên: 1712210.

Đánh giá mức độ hoàn thành

Tên công việc	Mức độ hoàn thành
Lập trình module tạo số ngẫu nhiên	100%
Hook một system call	100%
Tổng kết	**100%**

PHẦN 1 - MODULE TẠO SỐ NGẪU NHIÊN

Mục tiêu

- Tìm hiêu về Linux kernel module, hệ thống quản lý tập tin và device.
- Tìm hiểu về giao tiếp giữa tiến trình ở user space và kernel space.

Kết quả cần đạt được

- Viết một module tạp ra số ngẫu nhiên.
- Module này sẽ tạo một character device để cho phép các tiến trình ở user space có thể "open" và "read" các số ngẫu nhiên.

Thành phần mã nguồn

- Makefile: Build một module từ my_mod_rand.c
- test.c: Thử nghiệm các yêu cầu bài toán.
- my_mod_rand.c: Mã nguồn 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.

Các bước cài đặt

1. Di chuyển đến thư mục LinuxKernelModule

2. Cài đặt: “make”, “sudo insmod my_mod_rand.ko”
3. Kiểm tra: “make test”, “sudo ./test.o”
4. Gỡ cài đặt: “sudo rmmod my_mod_rand.ko”, “make clean”

PHẦN 2 - HOOK MỘT SYSTEM CALL

Mục tiêu

- Tìm hiểu và viết một chương trình hook vào một system call.

Kết quả cần đạt được

- 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.

Thành phần mã nguồn

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

Các bước cài đặt

1. Di chuyển đến thư mục SystemCallHook
2. Test hook trước khi tải vào kernel: “make”
3. Mở terminal thứ 2 để quan sát kernel messages: “sudo dmesg –C", “dmesg -wH”
4. Chạy trên terminal thứ 1. Quan sát terminal thứ 2 để thấy được kết quả: “sudo insmod hook.ko”, “sudo rmmod hook.ko”

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.