TRƯỜNG ĐẠI HỌC BÁCH KHOA TP HCM
**KHOA KHOA HỌC MÁY TÍNH**


# HỆ ĐIỀU HÀNH

## Báo cáo Bài tập lớn số 01

# System Call

GVHD: Phạm Kiên
SV thực hiện: Nguyễn Xuân Hiến 1652192

Tp. Hồ Chí Minh, Tháng 03/2018

# 1 Chuẩn bị

- **Cài đặt các gói cho Ubuntu:**

```
$ sudo apt-get update
$ sudo apt-get install build-essential
$ sudo apt-get install kernel-package
```

**Câu hỏi :**Tại sao chúng ta cần phải cài đặt kernel-package ?
**Trả lời :**Chúng ta cần cài đặt gói kernel-pacage, nó cung cấp khả năng tạo một ảnh hạt nhân Debian, nó cần thiết để xây dựng các gói Debian của hạt nhân Linux.

- **Tải và giải nén Linux kernel:**

```
$ sudo mkdir ~/kernelbuild
$ cd ~/kernelbuild
$ wget https://cdn.kernel.org/pub/linux/kernel/v4.x/linux-4.4.56.tar.xz
$ tar -xvJf linux-4.4.56.tar.xz
```

**Câu hỏi:** Tại sao chúng ta sử dụng một nguồn kernel khác thay vì sử dụng hạt nhân gốc ?
**Trả lời:** Bởi vì ta không thể biên dịch một kernel trực tiếp từ chính nó, mà chỉ có thể biên dịch một phiên bản kernel tương tự với bản hiện tại.

# 2 Cấu hình hệ thống và thêm lời gọi hệ thống mới

- **Đổi tên phiên bản kernel**

```
$ cp /boot/config-x.x.x-x-generic ~/kernelbuild/linux-4.4.56/.config
```

Trong đó x.x.x-x-generic là phiên bản hạt nhân gốc trên máy ảo của bản. Sử dụng uname -r để kiểm tra phiên bản kernel gốc trên máy của bạn.

Đổi tên Linux kernel của bạn:

```
$ sudo apt-get install libncurses5-dev
$ cd ~/kernelbuild/linux-4.4.56
$ make nconfig // or make menuconfig
```

Chọn General setup —> local version –> Nhập .1652192 –> Nhấn F6 để lưu và F9 để thoát.

Cài đặt gói openssl:

```
$ sudo apt-get install openssl libssl-dev
```

- **Thêm lời gọi hệ thống mới**
Đi đến thư mục arch/x86/entry/syscalls chứa danh sách các cuộc gọi hệ thống.
Vì ubuntu được cài là phiên bản x86 64-bit nên cần thêm cuộc gọi vào file syscall_64.tbl
Thêm dòng:

```
[number] x32 procmem sys_procmem
```

**Câu hỏi :**Ý nghĩa của các phần tử trong dòng trên là gì?
**Trả lời :**

- number là số thứ tự cuộc gọi trong danh sách. Trong trường hợp này bằng cuối cùng trong danh sách cộng với 1
- x32 là abi[1] của hệ thống
- procmem là tên của cuộc gọi hệ thống
- sys_procmem là một điểm truy nhập (entry point)[2].

Đi đến include/linux/ và thêm vào cuối file syscall.h
Thêm dòng:

```
struct proc_segs;
asmlinkage long sys_procmem( int pid, struct proc_segs * info);
```

**Câu hỏi :**Ý nghĩa của hai dòng trên là gì ?
**Trả lời :**

- dòng thứ nhất là lệnh khai báo cấu trúc tự định nghĩa proc_segs.
- dòng thứ hai khai báo một lời gọi hệ thống sys_procmem nhận vào hai giá trị pid và con trỏ info.

- **Hiện thực lời gọi hệ thống**
  Đi đến thư mục arch/x86/kernel tạo mới file sys_procmem.c chứa mã nguồn của cuộc gọi hệ thống.
  Thêm dòng:

```
obj-y += sys_procmem.o # name of syscall object file
```

vào cuối file Makefile trong thư mục arch/x86/kernel

# 3 Biên dịch và cài đặt hệ thống

- **Biên dịch Linux kernel**

```
$ sudo make -j 4
$ sudo make -j 4 modules
```

**Câu hỏi :**Ý nghĩa của hai lệnh trên ?
**Trả lời :**

- Lệnh make biên dịch và liên kết hạt nhân. Kết quả của lệnh này là tập tin nhị phân vmlinuz.

[1] https://en.wikipedia.org/wiki/Application_binary_interface
[2] https://en.wikipedia.org/wiki/Entry_point

  - Lệnh make modules dùng để biên dịch các modules. Các module là các bộ phân của hạt nhân được nạp trực tiếp, khi nó cần thiết.

- **Cài đăt Linux kernel**

```
$ sudo make -j 4 modules_install
$ sudo make -j 4 install
$ sudo reboot
```

Trong lúc hệ thống mới khởi động nhấn nhanh nút esc trên bàn phím –> chọn Advanced options for Ubuntu –> chọn phiên bản kernel bạn mới cài đặt. Hoặc bạn có thể chạy lênh sudo update-grub. Sau khi khởi động hệ thống mở terminal nhập:

```
$ uname -r
```

Nếu phiên bản kernel có chứa mã số sinh viên của bạn thì bạn đã cài đặt thành công.

- **Kiểm tra hệ thống**
Tạo mới một file text để kiểm tra hệ thống.
Trong đó number_32 là số thứ tự của cuộc gọi đã thêm vào trong file syscall_64.tbl.
**Câu hỏi :**Tại sao chương trình trên cho biết hệ thống có hoạt động hay không ?
**Trả lời :**Chương trình trên sẽ gọi lời gọi hệ thống đã thêm vào và gọi hàm này được hiện thực trong file sys_procmem.c nên nó kiểm tra xem hệ thống có hoạt động hay không.

# 4 Tạo API cho lời gọi hệ thống

- **Tạo file procmem.h và procmem.c trong thư mục a**

```
$ mkdir ~/Desktop/a
$ cd ~/Desktop/a
```

Tạo file procmem.h và procmem.c trong thư mục a.
**Câu hỏi:** Tại sao phải định nghĩa lại cấu trúc proc_segs ?
**Trả lời:** Cần định nghĩa lại cấu trúc proc_segs bới vì chương trình trên không sử dụng mã nguồn hạt nhân.
Trong đó number_32 là số thứ tự của cuộc gọi đã thêm vào trong file syscall_64.tbl.

- **Dịch và chạy chương trình**

```
$ sudo cp ~/Desktop/a/procmem.h /usr/include
$ gcc -share -fpic procmem.c -o libprocmem.so
$ sudo cp ~/Desktop/a/libprocmem.so /usr/lib
```

**Câu hỏi:** Tại sao cần thêm sudo trước lênh cp ?
**Trả lời:** Bởi vì thư mục /usr/include thuộc quyền sở hữu của root.
**Câu hỏi:** Tại sao cần đặt -share -fpic vào lênh gcc ?
**Trả lời:** Mã này được tích hợp vào các thư viện chia sẻ cho phép tạo ra mã vị trí độc lập, để thư viện chia có thể dễ dàng nạp vào bất kỳ địa chỉ nào trong bộ nhớ.
Tạo file source code trong thư mục a.
Biên dịch và chạy source code trên với tùy chọn -lprocmem.