**BÁO CÁO ĐỒ ÁN**

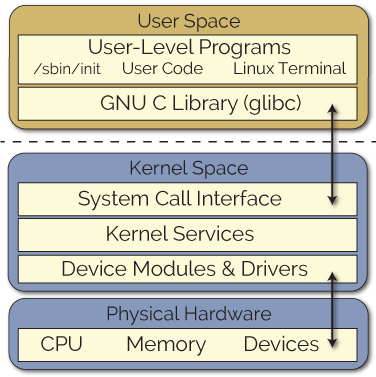
**LINUX KERNEL**

*Hoàng Thiên Nữ - 1612880*

# Linux kernel module là gì?

Kernel module là một cơ chế giúp thêm hoặc loại bỏ code ra khỏi Linux kernel ở thời gian thực. Cơ chế này thích hợp cho những driver các thiết bị, kích hoạt kernel giao tiếp với phần cứng mà không cần phải biết rõ nguyên lý hoạt động của phần cứng.

Kernel module chạy ở Kernel Space và các ứng dụng chạy ở User Space (hình 1). Cả kernel space và user space đều có những vùng địa chỉ nhớ riêng biệt mà không bị trùng lắp nhau. Phương pháp này đám bảo các ứng dụng chạy ở user space có một cái nhìn nhất quán đối với phần cứng, bất kể nền tảng nào. Các dịch vụ ở kernel được cung cấp cho user space thông qua system calls. Kernel ngăn các ứng dụng ở user space có xung đột với nhau hoặc truy cập vào tài nguyên không cho phép thông qua các mức bảo vệ.



Hình 1 Giao tiếp giữa các thành phần máy tính

# Xây dựng một kernel module

Một phương pháp để xây dựng một kernel module là sử dụng kernel code. Khác với việc viết một chương trình ứng dụng, một kerel module không có hàm main() và có một số điểm lưu ý như sau:

* Không được thực thi một cách tuần tự: một kernel module sẽ tự đăng kí để xử lý những yêu cầu sử dụng các hàm khởi tạo của chính nó. Các hàm đó sẽ chạy và dừng sau đó. Các loại yêu cầu mà nó xử lý có thể được định nghĩa bên trong module code.
* Không tự động dọn dẹp: bất cứ tài nguyên nào được cấp phát ở module code đều phải được giải phóng một cách thủ công khi mà module dừng. Nếu không nó sẽ vẫn chiếm một vùng tài nguyên cho đến khi hệ thống khởi động lại.
* Không có printf(): kernel code không thể truy cập những thư viện code được viết ở tầng user space. Kernel module chỉ chạy ở kernel space và có một vùng nhớ riêng. Có một sự giao tiếp giữa kernel space và user space được định nghĩa. Do vậy, ta sẽ dùng printk() thể xuất thông tin từ user space.
* Có thể bị gián đoạn.
* Có đặc quyền thực thi ở cấp độ cao hơn.
* Không hỗ trợ chấm động: kernel code sử dụng traps để chuyển đổi từ số nguyên sang số chấm động cho các ứng dụng ở user space.

Cần phải có Makefile để xây dựng một kernel module. Cấu trúc một Makefile như sau:

|  |  |
| --- | --- |
| 1  2  3  4  5  6 | obj-m+= modulename.o    all:  make -C /lib/modules/$(shell uname -r)/build/ M=$(PWD) modules  clean:  make -C /lib/modules/$(shell uname -r)/build/ M=$(PWD) clean |

Dòng đầu tiên là goal definition. Nó sẽ khai báo tên module để xây dựng. Sử dụng obj-m để khai báo một module. $shell uname -r là một lời gọi để trả về phiên bản kernel hiện tại. Quá trình cài đặt module được gọi qua **make**.

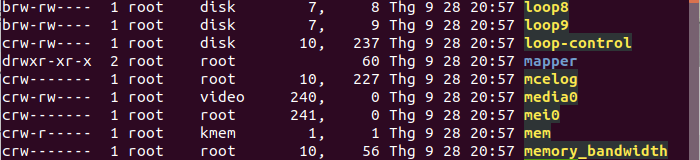
# Hệ thống quản lý file và device trong Linux

## Character Device

Một character device sẽ giúp chuyển dữ liệu qua lại giữa kernel và ứng dụng ở user space. Chúng giống như một cổng để kết nối để đọc hoặc ghi các byte dữ liệu theo một luồng từng kí tự. Chúng cung cấp một nền tảng cho nhiều driver, ví dụ như video hay âm thanh. Một sự thay thế cho character device là block device. Block device cho phép một dãy buffer của cache điều khiển những thao tác như đọc, viết và tìm kiếm. Tất cả các thiết bị đều có thể được truy cập thông qua hệ thống file được đính kèm vào cây file hệ thống. Ví dụ, với đồ án này, toàn bộ code của chương trình khi chạy sẽ trở thành một device /dev/generDev

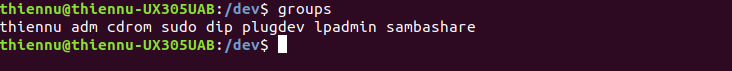
## Major and minor number

Major number được dùng trong kernel để xác định đúng driver của thiết bị khi thiết bị được truy cập. Mặt khác, minor number phụ thuộc vào thiết bị, và chỉ sử dụng bên trong driver. Ta có thể thấy được các cặp major/minor của mỗi device trong thư mục /dev.



Hình 2 Major và minor number của các thiết bị trong Linux

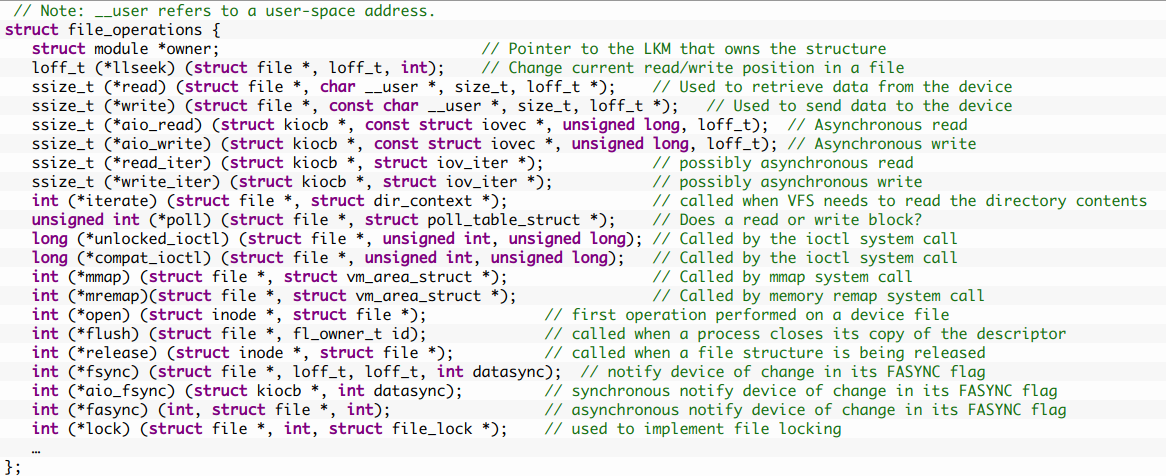
Character device sẽ được xác định bởi chữ ‘c’ ở cột đầu tiên trong danh sách, và block device là chữ ‘b’. Quyền truy cập, chủ sở hữu và nhóm các thiết bị đều được cung cấp với mỗi thiết bị.



Hình 3 Các groups có trong HĐH

## Cấu trúc quản lý dữ liệu file

Cấu trúc quản lý dữ liệu file được khai bảo ở /linux/fs.h. Cấu trúc này chứa những con trỏ hàm trong một driver, cho phép định nghĩa các thao tác xử lý. Hình 4 à một số ví dụ thao tác trên cấu trúc này.



Hình 4 Cấu trúc quản lý dữ liệu file

## Soure code cho driver thiết bị

Souce code này được đính kèm và có file README hướng dẫn đi kèm.