UNC Universidad Nacional de Córdoba

FCEFyN Facultad de Ciencias Exactas, Fisicas y Naturales

TRABAJO PRÁCTICO Nº 3:

"Desarrollo de módulos para el kernel de Linux - Char device drivers"

Cátedra: Sistemas Operativos 1

Año: 2019

Integrantes:

- Ferrero Alejandro (Matrícula n° 40054394)
- Pichetti Augusto (Matrícula n° 41018392)

Carrera: Ingeniería en Computación.

Profesores:

- Martínez Pablo
- Alonso Martín

Introducción

En el presente trabajo práctico se desarrollan dos simples módulos para insertar en el kernel de linux. Estos módulos van a emular el funcionamiento de dos dispositivos de caracteres. Uno de los dispositivos realizará una encriptación simple de los caracteres que se escriben en el. El otro de los módulos realizará la desencriptación de los caracteres que se escriben en el.

Desarrollo del proyecto

El módulo encriptador consta de dos archivos: encriptador.c y su archivo de encabezado encriptador.h.

Ambos módulos son análogos en su implementación, constan de las mismas funciones. La única diferencia es en su función device_write() en la cual el driver encriptador suma un número fijo (5) para codificar los caracteres ASCII escritos en el dispositivo de caracter ubicado en /dev/encriptador. En cambio, el driver desencriptador resta la misma cantidad para decodificar el dato introducido en el dispositivo /dev/decodificador.

Para cargar el módulo se hace uso del programa insmod el cual directamente llama a la función init_module() implementada. Dicha función registra al driver mediante register_chrdev(Major, DEVICE_NAME, &fops) asignándole un número mayor único; esto lo hace de forma dinámica. Para descargar el módulo se utiliza rmmod, el cual hace uso de la función cleanup_module() que desregistra el driver mediante unregister_chrdev(Major, DEVICE_NAME).

Las funciones implementadas por cada módulo, las cuales están definidas en la estructura file_operations, son las siguientes:

- device_read()
- device_write()
- device_open()
- device_release()

Device_open() es llamada cuando un proceso trata de abrir el archivo de dispositivo (device file) como por ej: "cat /dev/encriptador". En la misma se incrementa un contador de usos mediante try_module_get(THIS_MODULE).

Device_release() es llamada cuando un proceso cierra el device file. En ella se decrementa el contador de usos con module_put(THIS_MODULE).

Device_read() es llamada cuando un proceso, el cual ya abrió el dev file, intenta leer de él. Se utiliza la función put_user() que copia un único valor del espacio del kernel al espacio de usuario. Retorna la cantidad de bytes leídos.

Device_write() es llamada cuando un proceso escribe al archivo de dispositivo (dev file) como por ej. echo "hi" > /dev/encriptador. En la misma se copian los datos del espacio de usuario al espacio del kernel mediante get_user() y luego se recorre el arreglo sumando (o restando) un número entero fijo para encriptar (o desencriptar). Retorna el número de caracteres escritos.

Cómo utilizar el driver

Pasos:

1. Posicionarse en el directorio donde se encuentra los archivos .c, .h y el Makefile y compilar mediante el comando “make”.
2. Cargar el módulo mediante sudo insmod encriptador.ko
3. Ver módulos cargados con lsmod
4. Ver los mensajes del kernel (aquellos que se imprimen con printk()) con dmesg --color=auto | tail. De ahi se podrá ver cuál fue el n° mayor asignado.
5. Crear el dispositivo de caracter con sudo mknod /dev/encriptador c numero_mayor 0
6. Agregarle permisos de escritura y lectura al dispositivo con sudo chmod 666 /dev/encriptador
7. Ver dispositivo con ls -l /dev
8. Escribir cadena de caracter con echo “texto a encriptar” > /dev/encriptador
9. Ver cadena encriptada mediante cat /dev/encriptador
10. Quitar el modulo con sudo rmmod encriptador
11. Borrar dispositivo de caracter con sudo rm /dev/encriptador