

Universidad Autónoma de Chihuahua



Trabajo Sistemas Operativos 1

final:

Explicación módulo de Kernel

Iván Miguel Chavero Jurado

Jesús Alan González Murillo; 357600

Angel Torres Loya; 365354

Efraín Gilberto Domínguez Banda; 348453

05/06/2024

Explicación del módulo de Kernel

Código fuente de nuestro modulo

Comenzamos definiendo las partes del código fuente de nuestro Kernel.

Archivos de cabecera y algunas definiciones que forman parte del boiler plate de cualquier módulo de Kernel.

```
Modulo > C Bohemian_Rhapsody.c > ...
1  #include <linux/init.h>
2  #include <linux/module.h>
3  #include <linux/kernel.h>
4  #include <linux/fs.h>
5  #include <linux/cdev.h>
6  #include <linux/miscdevice.h>
7
8  MODULE_LICENSE("GPL");
9  MODULE_AUTHOR("Angel");
10 MODULE_DESCRIPTION("Farrokh Bulsara");
11 MODULE_VERSION("0.01");
12
13 #define DEVICE_NAME "Bohemian_Rhapsody"
14
```

Prototipos de nuestras funciones que son funciones básicas de Kernel: abrir, leer, escribir y cerrar.

```
18 static int Bohemian_Rhapsody_open(struct inode*, struct file*);
19 static int Bohemian_Rhapsody_release(struct inode*, struct file*);
20 static ssize_t Bohemian_Rhapsody_read(struct file*, char*, size_t, loff_t*);
21 static ssize_t Bohemian_Rhapsody_write(struct file*, const char*, size_t, loff_t*);
22 int write_to_user(char *, char *);
23
```

Especificábamos mediante la estructura file_operations de cuales serán las funciones que ha de ejecutar cuando quiera realizar cada una de las funciones básicas de archivo.

```
24 static struct file_operations fops = {
25     .owner = THIS_MODULE,
26     .open = Bohemian_Rhapsody_open,
27     .release = Bohemian_Rhapsody_release,
28     .read = Bohemian_Rhapsody_read,
29     .write = Bohemian_Rhapsody_write,
30 };
```

La definición de nuestras funciones.

```
34 static int __init Bohemian_Rhapsody_init(void) {
35     int result;
36     major = register_chrdev(0, DEVICE_NAME, &fops);
37
38     if (major < 0) {
39         printk(KERN_ALERT "Bohemian_Rhapsody Module load failed\n");
40         return major;
41     }
42
43     printk(KERN_INFO "Bohemian_Rhapsody Module has been loaded\n");
44
45     printk(KERN_INFO "I was assigned major number %d. To talk to\n", major);
46     printk(KERN_INFO "the driver, create a dev file with\n");
47     printk(KERN_INFO "'mknod /dev/%s c %d 0'.\n", DEVICE_NAME, major);
48     printk(KERN_INFO "Try various minor numbers. Try to cat and echo to\n");
49     printk(KERN_INFO "the device file.\n");
50     printk(KERN_INFO "Remove the device file and module when done.\n");
51
52 > /*cdev_init(&char_device, &fops);...
64     return 0;
65 }
66
67 static void __exit Bohemian_Rhapsody_exit(void) {
68     printk(KERN_INFO "Ya no hay rola, removiendo módulo!!\n");
69 }
70
71 static int Bohemian_Rhapsody_open(struct inode *inodep, struct file *filep) {
72     printk(KERN_INFO "Super Module device opened\n");
73     return 0;
74 }
75
76 static ssize_t Bohemian_Rhapsody_write(struct file *filep, const char *buffer, size_t len, loff_t *offset) {
77     //int errors = 0;
78     static char message[256] = {0};
79     if (copy_from_user(message, (void *)buffer, len))
80 >         return -EFAULT;
81     /*if (len > 1024) { ...
84     strcpy(user_message, message);
85     printk(KERN_INFO "From userland: %s\n", (char *)message);
86     return len;
87 }
88
89 static int Bohemian_Rhapsody_release(struct inode *inodep, struct file *filep) {
90     printk(KERN_INFO "Super Module device closed\n");
91     return 0;
92 }
93
94 static ssize_t Bohemian_Rhapsody_read(struct file *filep, char *buffer, size_t len, loff_t *offset) {
95     int errors = 0;
96     char *message = "Is this the real life? Is this just fantasy?\nCaught in a landslide, no escape from reali
97     int message_len = strlen(message);
98     errors = copy_to_user(buffer, message, message_len);
99     printk(KERN_INFO "user_message: %s\n", message);
100     return errors == 0 ? message_len : -EFAULT;
101 }
102
103 module_init(Bohemian_Rhapsody_init);
104 module_exit(Bohemian_Rhapsody_exit);
105
```

- `Bohemian_Rhapsody_init`: nos imprimirá por consola información básica de nuestro módulo de Kernel. Registra el dispositivo de caracteres y asigna un número mayor. Añade el dispositivo de caracteres al sistema.
- `Bohemian_Rhapsody_exit`: Desregistra el dispositivo y libera recursos.
- `Bohemian_Rhapsody_open`: Simplemente imprime un mensaje al abrir el dispositivo.
- `Bohemian_Rhapsody_release`: Imprime un mensaje al cerrar el dispositivo.
- `Bohemian_Rhapsody_read`: Copia un mensaje predefinido al espacio de usuario.
- `Bohemian_Rhapsody_write`: Copia un mensaje del espacio de usuario al Kernel y lo guarda en `user_message`.

El registro y desregistro del módulo se realiza mediante

```
105  
106 module_init(Bohemian_Rhapsody_init);  
107 module_exit(Bohemian_Rhapsody_exit);  
108
```

Makefile

Este se encarga de automatizar el proceso de creación y compilación del módulo de Kernel.

```
obj-m += Bohemian_Rhapsody.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

old:
    make -C /lib/modules/6.1.13-200.fc37.x86_64/build M=$(PWD) modules
```

Nuestro modulo se compone de las siguientes partes:

- `obj-m += Bohemian_Rhapsody.o`: Indica que el archivo objeto 'Bohemian_Rhapsody.o' debe ser compilado como un módulo del Kernel.
- `make -C /lib/modules/$(shell uname -r)/build M=$(PWD) modules`: Cambia al directorio que contiene los archivos de cabecera e invoca el objetivo modulo, lo que desencadena el proceso de compilación de los módulos en el directorio.
- `make -C /lib/modules/$(shell uname -r)/build M=$(PWD) clean`: Cambia al directorio que contiene los archivos de cabecera y scripts necesarios para compilar módulos e invoca el objetivo 'clean', lo que desencadena el proceso de limpieza de los archivos generados durante la compilación en el directorio.
- `make -c /lib/modules/6.1.13-200.fc37.x86_64/build M=$(PWD) modules`: Cambia al directorio que contiene los archivos de cabecera y scripts necesarios para compilar módulos e invoca el objetivo 'modules' para compilar los módulos del Kernel ubicados en el directorio actual ('/lib/modules/6.1.13-200.fc37.x86_64/build').

Compilación del modulo

Nos posicionamos en el directorio que contiene nuestro archivo Makefile y el código fuente de nuestro modulo.

Ejecutamos el comando make lo que nos compilara nuestro modulo en base a las instrucciones del archivo Makefile.

Con esto se crea una serie de archivo entre ellos Bohemina_Rhapsody.ko que será el archivo que podrá leer nuestro Kernel de Linux.

```
colosus762@fedora ~/Modulo
[colosus762@fedora Modulo]$ ls
Bohemian_Rhapsody.c  Makefile
[colosus762@fedora Modulo]$ make
make -C /lib/modules/6.8.11-200.fc39.x86_64/build M=/home/colosus762/Modulo modules
make[1]: Entering directory '/usr/src/kernels/6.8.11-200.fc39.x86_64'
  CC [M] /home/colosus762/Modulo/Bohemian_Rhapsody.o
/home/colosus762/Modulo/Bohemian_Rhapsody.c: In function 'Bohemian_Rhapsody_init':
/home/colosus762/Modulo/Bohemian_Rhapsody.c:38:9: warning: unused variable 'result' [-Wunused-variable]
   38 |     int result;
       |         ^~~~~~
  MODPOST /home/colosus762/Modulo/Module.symvers
  CC [M] /home/colosus762/Modulo/Bohemian_Rhapsody.mod.o
  LD [M] /home/colosus762/Modulo/Bohemian_Rhapsody.ko
  BTF [M] /home/colosus762/Modulo/Bohemian_Rhapsody.ko
Skipping BTF generation for /home/colosus762/Modulo/Bohemian_Rhapsody.ko due to unavailability of vmlinux
make[1]: Leaving directory '/usr/src/kernels/6.8.11-200.fc39.x86_64'
[colosus762@fedora Modulo]$ ls
Bohemian_Rhapsody.c  Bohemian_Rhapsody.mod  Bohemian_Rhapsody.mod.o  Makefile      Module.symvers
Bohemian_Rhapsody.ko Bohemian_Rhapsody.mod.c Bohemian_Rhapsody.o      modules.order
[colosus762@fedora Modulo]$
```

Con el comando sudo insmod Bohemian_Rhapsody.ko estamos indicando que deseamos cargar nuestro modulo a espacio de Kernel. Una vez hecho esto con el comando lsmod | grep Bohemian_Rhapsody podemos verificar si nuestro modulo se encuentra cargado en espacio de Kernel.

```
colosus762@fedora ~/Modulo
[colosus762@fedora Modulo]$ ls
Bohemian_Rhapsody.c  Makefile
[colosus762@fedora Modulo]$ make
make -C /lib/modules/6.8.11-200.fc39.x86_64/build M=/home/colosus762/Modulo modules
make[1]: Entering directory '/usr/src/kernels/6.8.11-200.fc39.x86_64'
  CC [M] /home/colosus762/Modulo/Bohemian_Rhapsody.o
/home/colosus762/Modulo/Bohemian_Rhapsody.c: In function 'Bohemian_Rhapsody_init':
/home/colosus762/Modulo/Bohemian_Rhapsody.c:38:9: warning: unused variable 'result' [-Wunused-variable]
   38 |     int result;
       |         ^~~~~~
  MODPOST /home/colosus762/Modulo/Module.symvers
  CC [M] /home/colosus762/Modulo/Bohemian_Rhapsody.mod.o
  LD [M] /home/colosus762/Modulo/Bohemian_Rhapsody.ko
  BTF [M] /home/colosus762/Modulo/Bohemian_Rhapsody.ko
Skipping BTF generation for /home/colosus762/Modulo/Bohemian_Rhapsody.ko due to unavailability of vmlinux
make[1]: Leaving directory '/usr/src/kernels/6.8.11-200.fc39.x86_64'
[colosus762@fedora Modulo]$ ls
Bohemian_Rhapsody.c  Bohemian_Rhapsody.mod  Bohemian_Rhapsody.mod.o  Makefile      Module.symvers
Bohemian_Rhapsody.ko Bohemian_Rhapsody.mod.c Bohemian_Rhapsody.o      modules.order
[colosus762@fedora Modulo]$ sudo insmod Bohemian_Rhapsody.ko
[sudo] password for colosus762:
[colosus762@fedora Modulo]$ lsmod | grep Bohemian_Rhapsody
Bohemian_Rhapsody      16384  0
[colosus762@fedora Modulo]$
```

Con el comando sudo dmesg | tail muestra los últimos mensajes recibidos desde el Kernel, de aquí se extrae el mayor number (506 en este caso) que usa el espacio de Kernel y el espacio de usuario para comunicarse.

```
colosus762@fedora:~/Modulo
colosus762@fedora:~/Modulo
LD [M] /home/colosus762/Modulo/Bohemian_Rhapsody.ko
BTF [M] /home/colosus762/Modulo/Bohemian_Rhapsody.ko
Skipping BTF generation for /home/colosus762/Modulo/Bohemian_Rhapsody.ko due to unavailability of vmlinux
make[1]: Leaving directory '/usr/src/kernels/6.8.11-200.fc39.x86_64'
[colosus762@fedora Modulo]$ ls
Bohemian_Rhapsody.c  Bohemian_Rhapsody.mod  Bohemian_Rhapsody.mod.o  Makefile  Module.symvers
Bohemian_Rhapsody.ko  Bohemian_Rhapsody.mod.c  Bohemian_Rhapsody.o  modules.order
[colosus762@fedora Modulo]$ sudo insmod Bohemian_Rhapsody.ko
[sudo] password for colosus762:
[colosus762@fedora Modulo]$ lsmod | grep Bohemian_Rhapsody
Bohemian_Rhapsody 16384 0
[colosus762@fedora Modulo]$ sudo dmesg | tail
[sudo] password for colosus762:
[22103.932342] msi_wmi: Unknown event received
[22103.947847] atkbd serio0: Unknown key released (translated set 2, code 0xbd on isa0060/serio0).
[22103.947863] atkbd serio0: Use 'setkeycodes e03d <keycode>' to make it known.
[22465.485533] Bohemian_Rhapsody Module has been loaded
[22465.485535] I was assigned major number 506. To talk to
[22465.485536] the driver, create a dev file with
[22465.485537] 'mknod /dev/Bohemian_Rhapsody c 506 0'.
[22465.485537] Try various minor numbers. Try to cat and echo to
[22465.485538] the device file.
[22465.485538] Remove the device file and module when done.
[colosus762@fedora Modulo]$
```

Creamos un archivo de dispositivo en /dev, que permiten la comunicación entre el software del usuario y el hardware o los controladores del Kernel. El comando `chmod` cambia los permisos de acceso de archivos y directorios.

```
colosus762@fedora:~/Modulo
colosus762@fedora:~/Modulo
colosus762@fedora:~/Modulo
[colosus762@fedora Modulo]$ sudo mknod /dev/Bohemian_Rhapsody c 506 0
[sudo] password for colosus762:
[sudo] password for colosus762:
[colosus762@fedora Modulo]$ sudo chmod 666 /dev/Bohemian_Rhapsody
[sudo] password for colosus762:
[colosus762@fedora Modulo]$
```

Y finalmente podemos hacer que nuestro modulo cante con `cat /dev/Bohemian_Rhapsody`

```
colosus762@fedora:~/Modulo
(Galileo) Galileo, (Galileo) Galileo, Galileo Figaro, magnifico
But I'm just a poor boy, nobody loves me
He's just a poor boy from a poor family
Spare him his life from this monstrosity
Easy come, easy go, will you let me go?
نحن الله
No, we will not let you go (let him go)
نحن الله
We will not let you go (let him go)
نحن الله
We will not let you go (let me go)
Will not let you go (let me go)
Never, never, never, never let me go
No, no, no, no, no, no, no, no
Oh, mamma mia, mamma mia
Mamma mia, let me go
Beelzebub has a devil put aside for me, for me, for me
So you think you can stone me and spit in my eye?
So you think you can love me and leave me to die?
Oh, baby, can't do this to me, baby
Just gotta get out, just gotta get right outta here
Ooh
Ooh, yeah, ooh, yeah
Nothing really matters, anyone can see
Nothing really matters
Nothing really matters to me Is this the real life? Is this just fantasy?
Caught in a landslide, no escape from reality
Open your eyes, look up to the skies and see
I'm just a poor boy, I need no sympathy
Because I'm easy come, ^C
[colosus762@fedora Modulo]$ cat /dev/Bohemian_Rhapsody
```

Y finalmente retiramos nuestro modulo de los mdulos cargados con el comando `rmmmod Bohemian_Rhapsody`

```
colosus762@fedora:~/Modulo
[colosus762@fedora Modulo]$ lsmod | grep Bohemian_Rhapsody
Bohemian_Rhapsody      16384  0
[colosus762@fedora Modulo]$ sudo rmmmod Bohemian_Rhapsody
[colosus762@fedora Modulo]$ lsmod | grep Bohemian_Rhapsody
[colosus762@fedora Modulo]$
```