Lab Report of Project 1: Introduction to Linux Kernel Modules

Name: Li Dongyue Student Number: 516030910502
Computer System Engineering (Undergraduate), Fall 2018

October 21, 2018

1 Introduction

In this project, we learn how to create a kernel module and load it into the Linux kernel. We then modify the kernel module so that it creates an entry in the `/proc` file system. We enter commands on the command line to manage the modules in the kernel.

1.1 Creating Kernel Module

We first learn how to create a simple basic kernel module. It prints appropriate messages when it is loaded and unloaded. We use the function *simple_init()* as the module entry point, which represents the function that is invoked when the module is loaded into the kernel. Similarly, the *simple_exit()* function is the module exit point, the function that is called when the module is removed from the kernel. We add the `MODULE LICENSE(), MODULE DESCRIPTION()`, and `MODULE AUTHOR()` in the final lines to represent details regarding the software license, description of the module, and author. We combine the C program with a `Makefile` and use the command `make` to compile a kernel module.

1.2 Loading Kernel Module

Kernel modules are loaded using the `insmod` command and removed by the `rmmod` command. As kernel modules are running within the kernel, it is possible to obtain values and call functions that are available only in the kernel and not to regular user applications. Once we are able to correctly load and unload your module, we can operate some value in the kernel.

1.3 The /proc File System

The `/proc` file system is a "pseudo" file system that exists only in kernel memory and is used primarily for querying various kernel and per-process statistics. We can design kernel modules that create additional entries in the `/proc` file system involving both kernel statistics and information related to specific processes.

2 Procedure

2.1 Jiffies

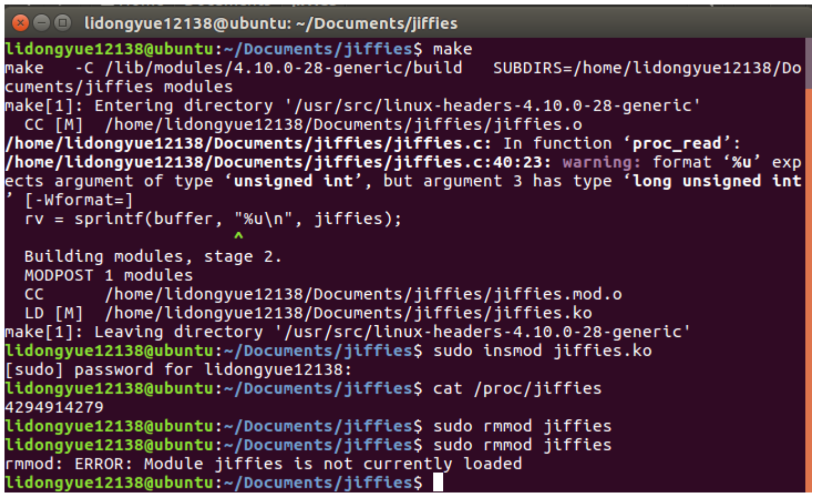
Follow the steps given by the textbook and we make a small change to the example code so that we can print the current value of `jiffies`. We show part of the function below:

```c
#include <linux/init.h>
#include <linux/kernel.h>
#include <linux/module.h>
#include <linux/proc_fs.h>
#include <linux/uaccess.h>
#include <linux/jiffies.h>
#define BUFFER_SIZE 128
#define PROC_NAME "jiffies"
ssize_t proc_read(struct file *file, char __user *usr_buf,
size_t count, loff_t *pos);
static struct file_operations proc_ops = {
  .owner = THIS_MODULE,
  .read = proc_read,
};
/* This function is called when the module is loaded. */
int proc_init(void)
{
```

```
/* creates the /proc/jiffies entry */
  proc_create(PROC_NAME, 0666, NULL, &proc_ops);
  return 0;
}
/* This function is called when the module is removed. */
void proc_exit(void)
{
  /* removes the /proc/jiffies entry */
  remove_proc_entry(PROC_NAME, NULL);
}
/* This function is called each time /proc/jiffies is read */
ssize_t proc_read(struct file *file, char __user *usr_buf,
size_t count, loff_t *pos)
{
  int rv = 0;
  char buffer[BUFFER_SIZE];
  static int completed = 0;
  if (completed) {
    completed = 0;
    return 0;
  }
  completed = 1;
  rv = sprintf(buffer, "%u\n", jiffies);
  /* copies kernel space buffer to user space usr buf */
  copy_to_user(usr_buf, buffer, rv);
  return rv;
}
module_init(proc_init);
module_exit(proc_exit);
MODULE_LICENSE("GPL");
MODULE_DESCRIPTION("jiffies Module");
MODULE_AUTHOR("SGG");
```

jiffies.c

The running result is shown in Figure 1.



```
lidongyue12138@ubuntu:~/Documents/jiffies$ make
make  -C /lib/modules/4.10.0-28-generic/build  SUBDIRS=/home/lidongyue12138/Do
cuments/jiffies modules
make[1]: Entering directory '/usr/src/linux-headers-4.10.0-28-generic'
  CC [M]  /home/lidongyue12138/Documents/jiffies/jiffies.o
/home/lidongyue12138/Documents/jiffies/jiffies.c: In function 'proc_read':
/home/lidongyue12138/Documents/jiffies/jiffies.c:40:23: warning: format '%u' exp
ects argument of type 'unsigned int', but argument 3 has type 'long unsigned int
' [-Wformat=]
  rv = sprintf(buffer, "%u\n", jiffies);
                       ^
  Building modules, stage 2.
  MODPOST 1 modules
  CC      /home/lidongyue12138/Documents/jiffies/jiffies.mod.o
  LD [M]  /home/lidongyue12138/Documents/jiffies/jiffies.ko
make[1]: Leaving directory '/usr/src/linux-headers-4.10.0-28-generic'
lidongyue12138@ubuntu:~/Documents/jiffies$ sudo insmod jiffies.ko
[sudo] password for lidongyue12138:
lidongyue12138@ubuntu:~/Documents/jiffies$ cat /proc/jiffies
4294914279
lidongyue12138@ubuntu:~/Documents/jiffies$ sudo rmmod jiffies
lidongyue12138@ubuntu:~/Documents/jiffies$ sudo rmmod jiffies
rmmod: ERROR: Module jiffies is not currently loaded
lidongyue12138@ubuntu:~/Documents/jiffies$
```

Figure 1: The running results of `jiffies`

2.2 Seconds

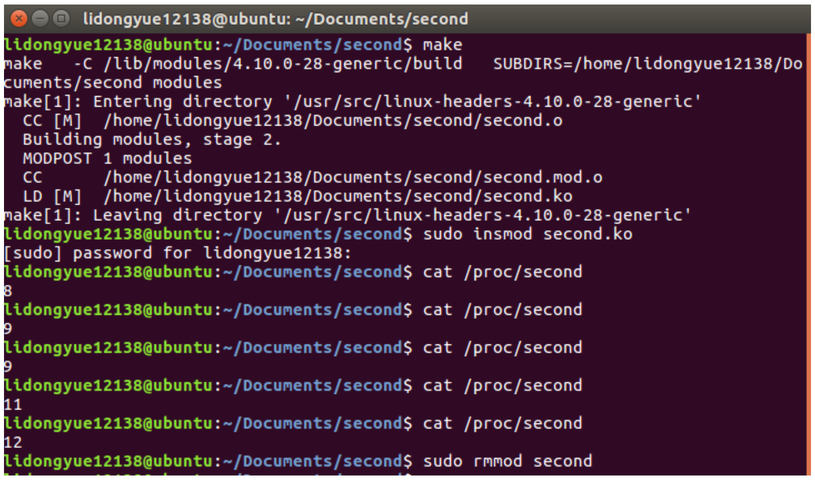
In this program, we have to do a calculation. We take a record of `jiffies` when we initialize the module and do a minus step and divide step to calculate the seconds. We show part of the codes below:

```
/* This function is called each time /proc/second is read */
ssize_t proc_read(struct file *file, char __user *usr_buf,
size_t count, loff_t *pos)
{
  int rv = 0;
  char buffer[BUFFER_SIZE];
  static int completed = 0;
  if (completed) {
    completed = 0;
```

```
        return 0;
    }
    completed = 1;
    rv = sprintf(buffer, "%lu\n", (jiffies-t)/HZ);
    /* copies kernel space buffer to user space usr buf */
    copy_to_user(usr_buf, buffer, rv);
    return rv;
}
```

seconds.c

The running result is shown in Figure 2.



```
lidongyue12138@ubuntu:~/Documents/second$ make
make   -C /lib/modules/4.10.0-28-generic/build   SUBDIRS=/home/lidongyue12138/Do
cuments/second modules
make[1]: Entering directory '/usr/src/linux-headers-4.10.0-28-generic'
  CC [M]  /home/lidongyue12138/Documents/second/second.o
  Building modules, stage 2.
  MODPOST 1 modules
  CC      /home/lidongyue12138/Documents/second/second.mod.o
  LD [M]  /home/lidongyue12138/Documents/second/second.ko
make[1]: Leaving directory '/usr/src/linux-headers-4.10.0-28-generic'
lidongyue12138@ubuntu:~/Documents/second$ sudo insmod second.ko
[sudo] password for lidongyue12138:
lidongyue12138@ubuntu:~/Documents/second$ cat /proc/second
8
lidongyue12138@ubuntu:~/Documents/second$ cat /proc/second
9
lidongyue12138@ubuntu:~/Documents/second$ cat /proc/second
9
lidongyue12138@ubuntu:~/Documents/second$ cat /proc/second
11
lidongyue12138@ubuntu:~/Documents/second$ cat /proc/second
12
lidongyue12138@ubuntu:~/Documents/second$ sudo rmmod second
```

Figure 2: The running results of `second`

3 Conclusion

In this project, we learn how to create a kernel module and load it into the Linux kernel, how to modify the kernel module so that it creates an entry in the `/proc` file system, and how to enter commands on the command line to manage the modules in the kernel. Source codes are attached with this file.