# Project 1 Report

Litao Zhou, 518030910407

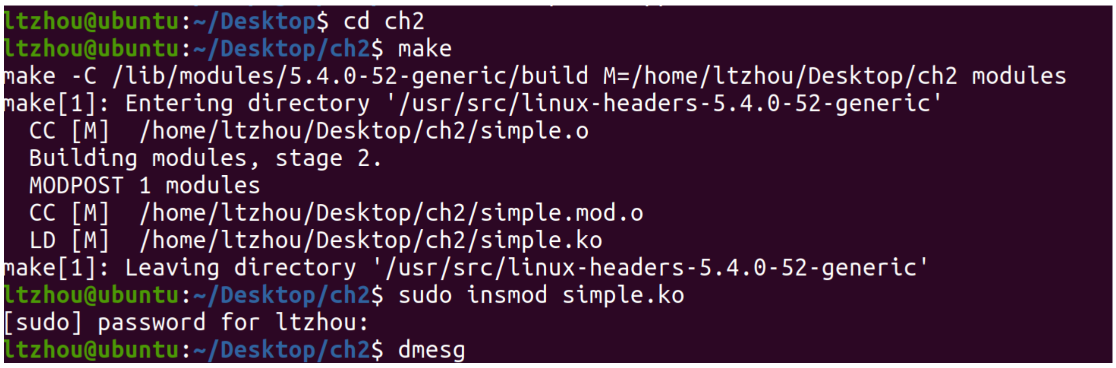
## Project Description

In this project, we will create some kernel modules and load them into the Linux kernel. Then we modify the kernel module so that it creates an entry in the /proc file system. There are three parts of experiments. First we try to load the simple.c kernel module into a Linux Virtual Machine. Then based on simple.c we modify the program by calling several kernel functions. The modified programs are in number.c and jifhz.c. Finally, we will try to load hello.c, which will create an entry in the /proc file system, and further develop other two kernel modules jiffies.c and seconds.c.

## Loading and Removing Kernel Modules

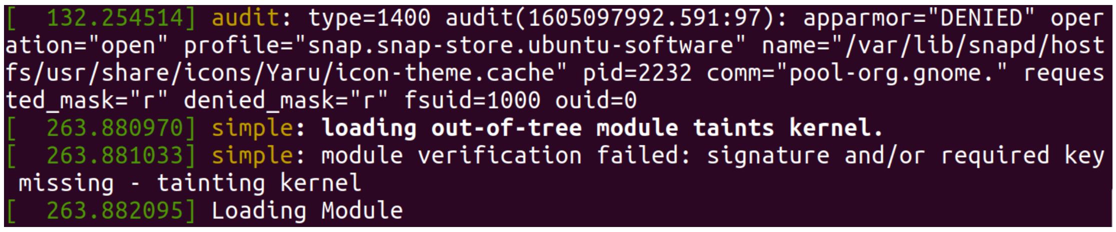
### Experiment Results

1. Compile the given simple.c into kernel module and load it into the kernel.



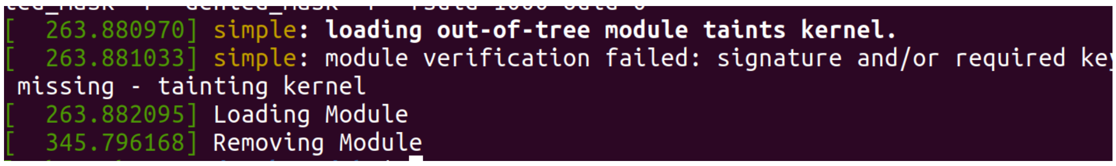
```
ltzhou@ubuntu:~/Desktop$ cd ch2
ltzhou@ubuntu:~/Desktop/ch2$ make
make -C /lib/modules/5.4.0-52-generic/build M=/home/ltzhou/Desktop/ch2 modules
make[1]: Entering directory '/usr/src/linux-headers-5.4.0-52-generic'
  CC [M]  /home/ltzhou/Desktop/ch2/simple.o
  Building modules, stage 2.
  MODPOST 1 modules
  CC [M]  /home/ltzhou/Desktop/ch2/simple.mod.o
  LD [M]  /home/ltzhou/Desktop/ch2/simple.ko
make[1]: Leaving directory '/usr/src/linux-headers-5.4.0-52-generic'
ltzhou@ubuntu:~/Desktop/ch2$ sudo insmod simple.ko
[sudo] password for ltzhou:
ltzhou@ubuntu:~/Desktop/ch2$ dmesg
```

2. The loading info can be found in dmesg



```
[  132.254514] audit: type=1400 audit(1605097992.591:97): apparmor="DENIED" oper
ation="open" profile="snap.snap-store.ubuntu-software" name="/var/lib/snapd/host
fs/usr/share/icons/Yaru/icon-theme.cache" pid=2232 comm="pool-org.gnome." reques
ted_mask="r" denied_mask="r" fsuid=1000 ouid=0
[  263.880970] simple: loading out-of-tree module taints kernel.
[  263.881033] simple: module verification failed: signature and/or required key
 missing - tainting kernel
[  263.882095] Loading Module
```

3. Remove our module from the kernel, and we can find the removing info from dmesg



```
[  263.880970] simple: loading out-of-tree module taints kernel.
[  263.881033] simple: module verification failed: signature and/or required ke
 missing - tainting kernel
[  263.882095] Loading Module
[  345.796168] Removing Module
```

## Golden Ratio Prime and GCD

### Implementation

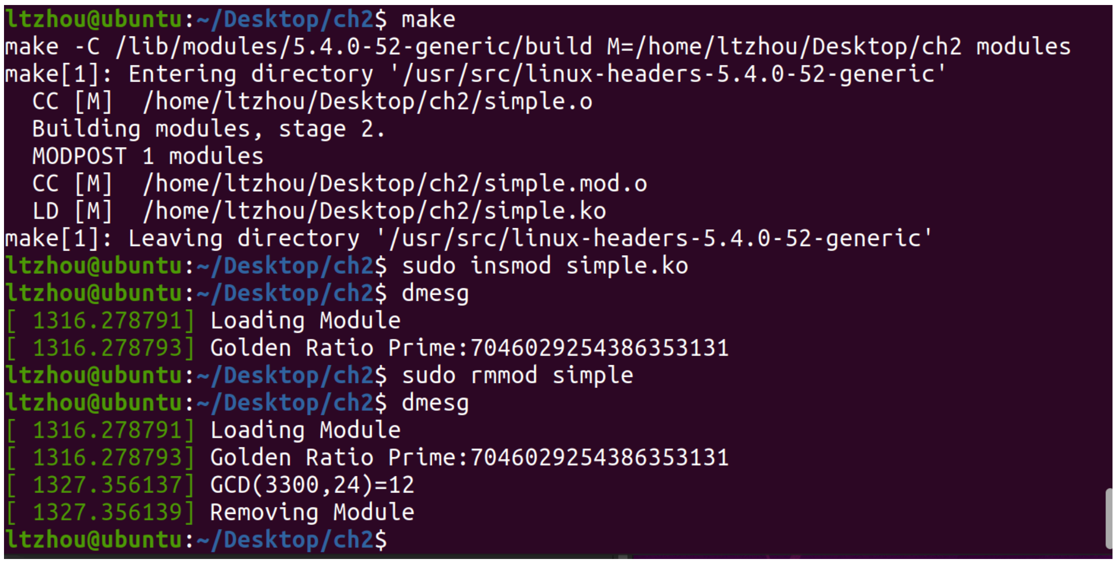
1. First include the kernel libraries that will be used. In this part, we use <linux/hash.h> <linux/gcd.h>.
2. Then we print the data required in the kernel using printk(). Note that the GOLDEN_RATIO_RATE is a long long unsigned number, and gcd() returns a long unsigned number. The datatype should be matched when printing.

The code we edited is shown below

```
#include <linux/hash.h>
#include <linux/gcd.h>

/* This function is called when the module is loaded. */
static int simple_init(void)
{
      printk(KERN_INFO "Loading Module\n");
      printk(KERN_INFO "Golden Ratio Prime:%llu\n", GOLDEN_RATIO_PRIME);
      return 0;
}

/* This function is called when the module is removed. */
static void simple_exit(void) {
      printk(KERN_INFO "GCD(3300,24)=%lu\n", gcd(3300,24));
      printk(KERN_INFO "Removing Module\n");
}
```

### Result

Our module printed out the expected result when loading and removing the module



```
ltzhou@ubuntu:~/Desktop/ch2$ make
make -C /lib/modules/5.4.0-52-generic/build M=/home/ltzhou/Desktop/ch2 modules
make[1]: Entering directory '/usr/src/linux-headers-5.4.0-52-generic'
  CC [M]  /home/ltzhou/Desktop/ch2/simple.o
  Building modules, stage 2.
  MODPOST 1 modules
  CC [M]  /home/ltzhou/Desktop/ch2/simple.mod.o
  LD [M]  /home/ltzhou/Desktop/ch2/simple.ko
make[1]: Leaving directory '/usr/src/linux-headers-5.4.0-52-generic'
ltzhou@ubuntu:~/Desktop/ch2$ sudo insmod simple.ko
ltzhou@ubuntu:~/Desktop/ch2$ dmesg
[ 1316.278791] Loading Module
[ 1316.278793] Golden Ratio Prime:7046029254386353131
ltzhou@ubuntu:~/Desktop/ch2$ sudo rmmod simple
ltzhou@ubuntu:~/Desktop/ch2$ dmesg
[ 1316.278791] Loading Module
[ 1316.278793] Golden Ratio Prime:7046029254386353131
[ 1327.356137] GCD(3300,24)=12
[ 1327.356139] Removing Module
ltzhou@ubuntu:~/Desktop/ch2$
```

## Jiffies and HZ

### Implementation

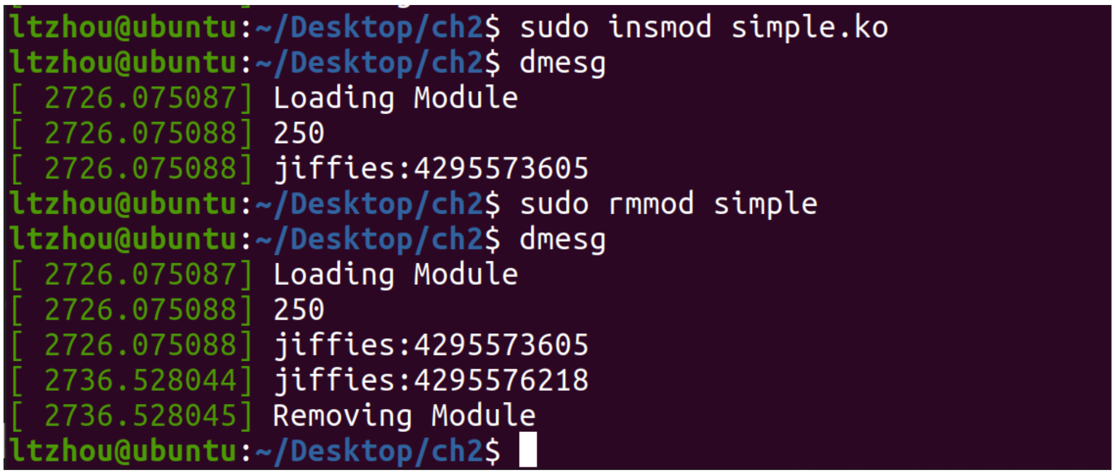
Similar as above, we include the related libraries so that we can access jiffies and HZ value. Then we print them out into the kernel message buffer. The code we edited is shown below

```c
#include <linux/jiffies.h>
#include <linux/param.h>


/* This function is called when the module is loaded. */
static int simple_init(void)
{
      printk(KERN_INFO "Loading Module\n");
      printk(KERN_INFO "%d", HZ);
      printk(KERN_INFO "jiffies:%llu\n", get_jiffies_64());
      return 0;
}

/* This function is called when the module is removed. */
static void simple_exit(void) {
      printk(KERN_INFO "jiffies:%llu\n", get_jiffies_64());
      printk(KERN_INFO "Removing Module\n");
}
```

## Result

Our module printed out the expected result when loading and removing the module. Since jiffies is the interrupt count since booting, and HZ is the frequency of the interrupt. We can calculate the running time of the module as follows

$$\Delta t = \frac{jiffies_2 - jiffies_1}{HZ}$$



```
ltzhou@ubuntu:~/Desktop/ch2$ sudo insmod simple.ko
ltzhou@ubuntu:~/Desktop/ch2$ dmesg
[ 2726.075087] Loading Module
[ 2726.075088] 250
[ 2726.075088] jiffies:4295573605
ltzhou@ubuntu:~/Desktop/ch2$ sudo rmmod simple
ltzhou@ubuntu:~/Desktop/ch2$ dmesg
[ 2726.075087] Loading Module
[ 2726.075088] 250
[ 2726.075088] jiffies:4295573605
[ 2736.528044] jiffies:4295576218
[ 2736.528045] Removing Module
ltzhou@ubuntu:~/Desktop/ch2$ 
```

The difference of the two jiffies value above in my experiment is $4295576218 - 4295573605 = 2613$. The running time of our module is $\frac{2613}{250} = 10.452$, which coincides with the running time indicated by dmesg.

# Jiffies in /proc

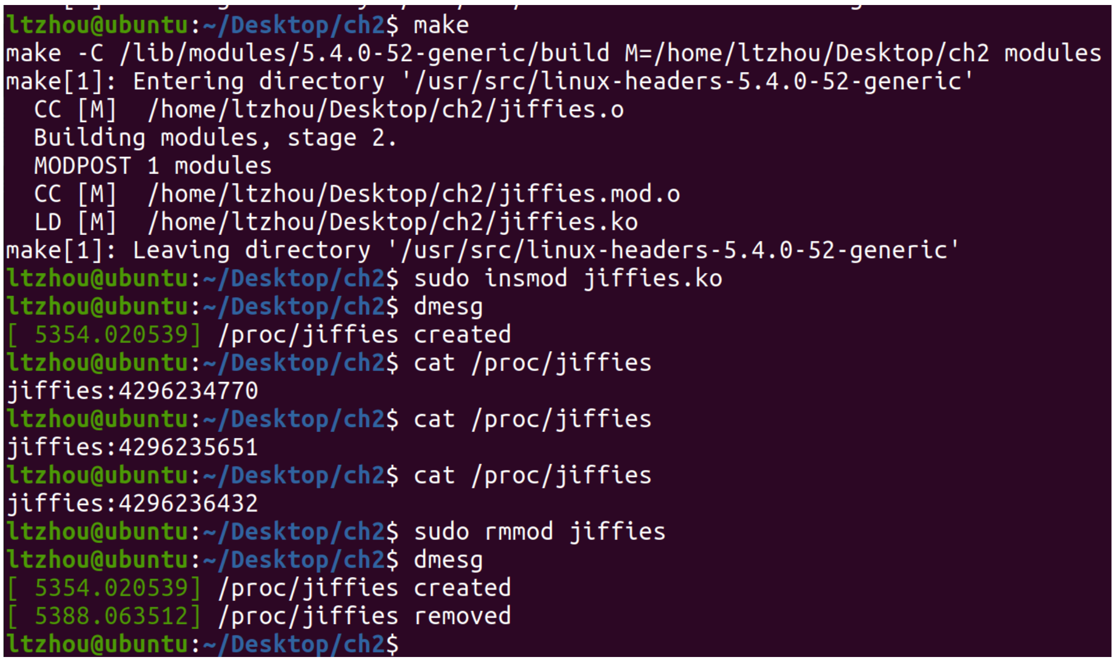
## Implementation

1. We import the kernel libraries related to jiffies. Then we modify the content to be printed out to the buffer in proc_read() function, as has been shown below.
2. The contents to be printed is just the present jiffies value, which can be implemented by calling get_jiffies_64() kernel function.
3. On my virtual machine, I have to rename the original copy_to_user() function into raw_copy_to_user() so that the program can compile.

```
#include <linux/jiffies.h>

#define BUFFER_SIZE 128

#define PROC_NAME "jiffies"

......

static 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, "jiffies:%llu\n", get_jiffies_64());

        // copies the contents of buffer to userspace usr_buf
        raw_copy_to_user(usr_buf, buffer, rv);

        return rv;
}
```

## Result

We load the jiffies module we have created into the kernel. After checking that the module is successfully loaded, we read the /proc/jiffies several times, and it can be seen that the jiffies value printed out are dynamically increasing depending on the time we make a request.



```
ltzhou@ubuntu:~/Desktop/ch2$ make
make -C /lib/modules/5.4.0-52-generic/build M=/home/ltzhou/Desktop/ch2 modules
make[1]: Entering directory '/usr/src/linux-headers-5.4.0-52-generic'
  CC [M]  /home/ltzhou/Desktop/ch2/jiffies.o
  Building modules, stage 2.
  MODPOST 1 modules
  CC [M]  /home/ltzhou/Desktop/ch2/jiffies.mod.o
  LD [M]  /home/ltzhou/Desktop/ch2/jiffies.ko
make[1]: Leaving directory '/usr/src/linux-headers-5.4.0-52-generic'
ltzhou@ubuntu:~/Desktop/ch2$ sudo insmod jiffies.ko
ltzhou@ubuntu:~/Desktop/ch2$ dmesg
[ 5354.020539] /proc/jiffies created
ltzhou@ubuntu:~/Desktop/ch2$ cat /proc/jiffies
jiffies:4296234770
ltzhou@ubuntu:~/Desktop/ch2$ cat /proc/jiffies
jiffies:4296235651
ltzhou@ubuntu:~/Desktop/ch2$ cat /proc/jiffies
jiffies:4296236432
ltzhou@ubuntu:~/Desktop/ch2$ sudo rmmod jiffies
ltzhou@ubuntu:~/Desktop/ch2$ dmesg
[ 5354.020539] /proc/jiffies created
[ 5388.063512] /proc/jiffies removed
ltzhou@ubuntu:~/Desktop/ch2$
```

## Elapsed time in /proc

### Implementation

We can calculate the elapsed time by using the formula proposed above.

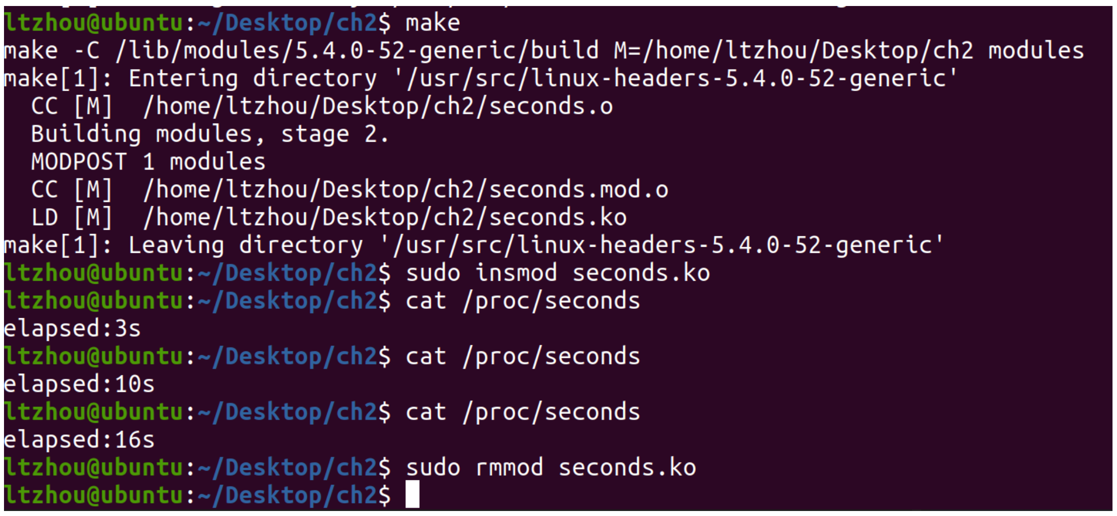
In order to obtain the difference between the current jiffies and the jiffies when the module is loaded, we need to store the loading jiffies. Therefore, we introduce a new static variable to store the initial jiffies. The load_jif variable will be updated when the module is loaded.

```c
#include <linux/jiffies.h>
#include <linux/param.h>

#define BUFFER_SIZE 128

#define PROC_NAME "seconds"

static u64 load_jif;

......

static int proc_init(void)
{
       // creates the /proc/seconds entry
       // the following function call is a wrapper for
       // proc_create_data() passing NULL as the last argument
       load_jif = get_jiffies_64(); // <--- This is new

       proc_create(PROC_NAME, 0, NULL, &proc_ops);

       printk(KERN_INFO "/proc/%s created\n", PROC_NAME);
```

```c
        return 0;
}

static 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;
        u64 elapsed_time;

        if (completed) {
                completed = 0;
                return 0;
        }

        completed = 1;

        elapsed_time = (get_jiffies_64() - load_jif) / HZ;

        rv = sprintf(buffer, "elapsed:%llus\n", elapsed_time);

        // copies the contents of buffer to userspace usr_buf
        raw_copy_to_user(usr_buf, buffer, rv);

        return rv;
}
```

## Result

We load the seconds module we have created into the kernel. After checking that the module is successfully loaded, we read the /proc/seconds several times, and it can be seen that the seconds value printed out are dynamically increasing from 0s depending on the moment we make a request.


```
ltzhou@ubuntu:~/Desktop/ch2$ make
make -C /lib/modules/5.4.0-52-generic/build M=/home/ltzhou/Desktop/ch2 modules
make[1]: Entering directory '/usr/src/linux-headers-5.4.0-52-generic'
  CC [M]  /home/ltzhou/Desktop/ch2/seconds.o
  Building modules, stage 2.
  MODPOST 1 modules
  CC [M]  /home/ltzhou/Desktop/ch2/seconds.mod.o
  LD [M]  /home/ltzhou/Desktop/ch2/seconds.ko
make[1]: Leaving directory '/usr/src/linux-headers-5.4.0-52-generic'
ltzhou@ubuntu:~/Desktop/ch2$ sudo insmod seconds.ko
ltzhou@ubuntu:~/Desktop/ch2$ cat /proc/seconds
elapsed:3s
ltzhou@ubuntu:~/Desktop/ch2$ cat /proc/seconds
elapsed:10s
ltzhou@ubuntu:~/Desktop/ch2$ cat /proc/seconds
elapsed:16s
ltzhou@ubuntu:~/Desktop/ch2$ sudo rmmod seconds.ko
ltzhou@ubuntu:~/Desktop/ch2$
```