

上海交通大学

SHANGHAI JIAOTONG UNIVERSITY



操作系统课程设计报告 - Project1

Introduction to Linux kernel

姓名：薛春宇

学号：518021910698

完成时间：2020/5/22

一、实验目的

1. 了解 Linux kernel 的编译方法和过程，加深自身对 Linux 原理的了解
2. 学习如何编写 Linux kernel module，以及如何将这些模块插入内核中，以扩展操作系统的功能

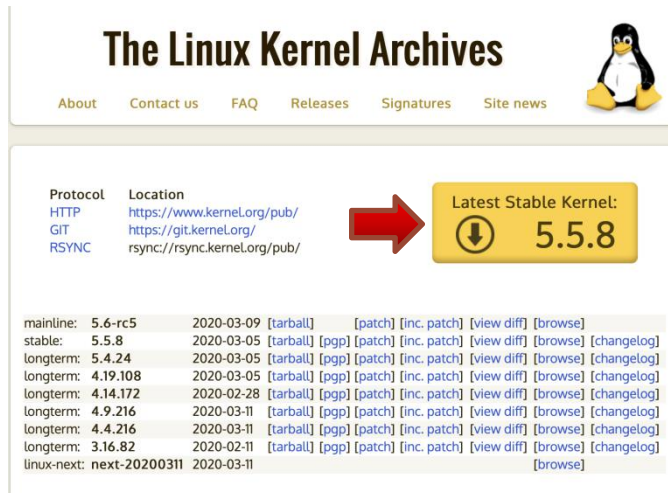
二、实验内容

1. 在虚拟机上编译新的 Linux 内核并安装、启动；
2. 分别编写两个 kernel modules:
 - (1) 报告 jiffies 的当前值；
 - (2) 报告当前运行秒数；

三、内容一：编译安装 Linux 内核

1. 下载内核源码：

打开 Ubuntu，前往网站 <https://www.kernel.org> 下载内核源码 (.tar.xz) 到桌面上。



注意：本次试验 Ubuntu 版本为 ubuntu-19.10-desktop-amd64, Linux kernel 版本为 linux-5.5.8

2. 解压：

在桌面右键打开 Ubuntu 终端，执行命令将文件解压到桌面：

```
apple@ubuntu:~$ sudo tar -xavf linux-5.5.8.tar.xz
[sudo] apple 的密码： █
```

3. 安装必要工具：

在终端内执行以下安装命令：

```
apple@ubuntu:~$ sudo apt-get install gcc make libncurses5-dev openssl libssl-dev
```

```
apple@ubuntu:~$ sudo apt-get install build-essential
```

```
apple@ubuntu:~$ sudo apt-get install pkg-config
```

```
apple@ubuntu:~$ sudo apt-get install libc6-dev
```

```
apple@ubuntu:~$ sudo apt-get install bison
```

```
apple@ubuntu:~$ sudo apt-get install flex
```

```
apple@ubuntu:~$ sudo apt-get install libelf-dev
```

注意：若出现如下情况，说明该组件已经安装到最新版本。

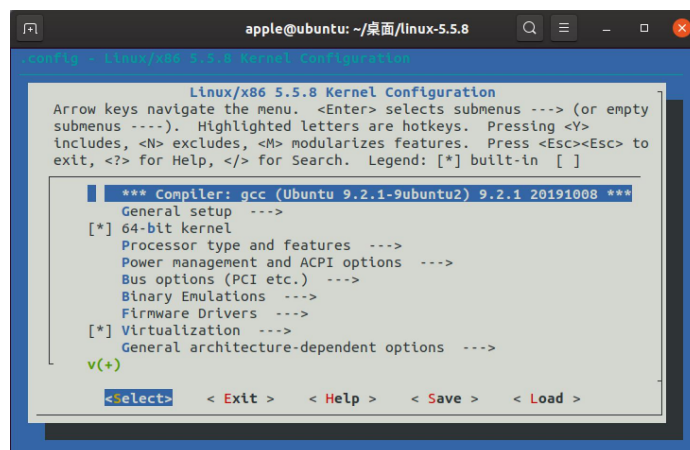
```
apple@ubuntu:~$ sudo apt-get install libelf-dev
[sudo] apple 的密码：
正在读取软件包列表... 完成
正在分析软件包的依赖关系树
正在读取状态信息... 完成
libelf-dev 已经是最新版 (0.176-1.1)。
升级了 0 个软件包，新安装了 0 个软件包，要卸载 0 个软件包，有 50 个软件包未被升级。
```

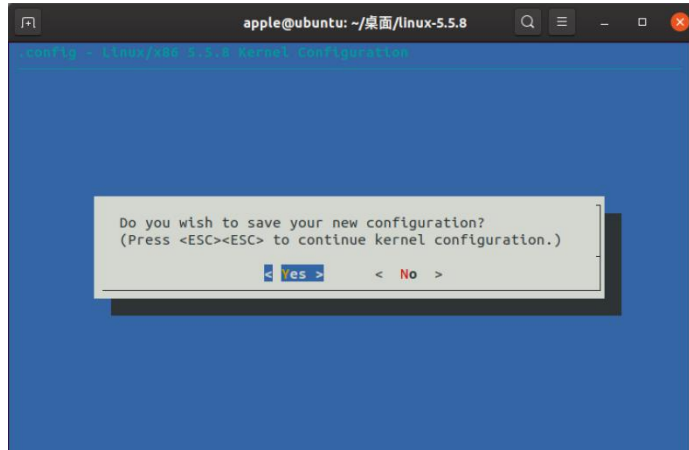
4. 开始准备编译：

在终端中打开已经解压在桌面的 linux-5.5.8 文件，并执行以下命令：

```
apple@ubuntu:~/桌面/linux-5.5.8$ sudo make menuconfig
```

输入密码后会出现下图：





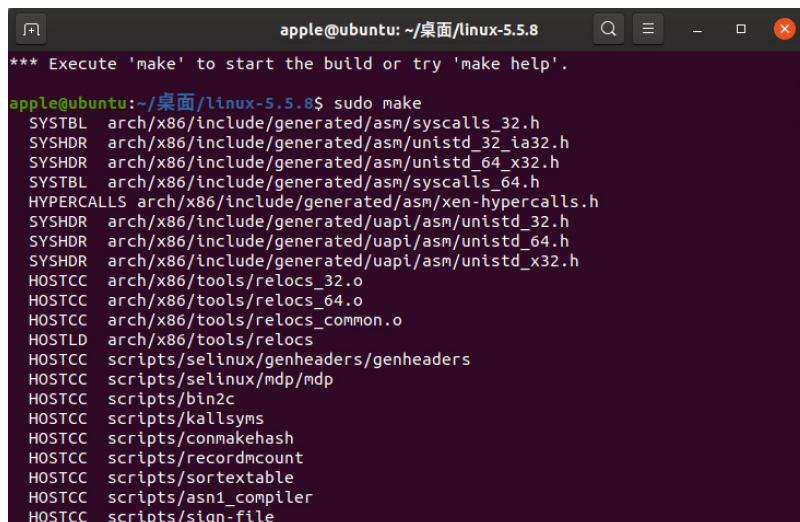
直接设置所有为默认选项, 选择 Exit 后继续选择 Yes 跳转至命令行窗口。

5. 编译

执行以下指令开始编译:

```
apple@ubuntu:~/桌面/linux-5.5.8$ sudo make
```

编译过程如下图:



注意: 编译即将完成时遇到如下错误:

```

make: *** [Makefile:1078: vmlinux] 错误 137
apple@ubuntu:~/桌面/linux-5.5.8$ ^C
apple@ubuntu:~/桌面/linux-5.5.8$ free -m
              总计          已用          空闲          共享      缓冲/缓存      可用
内存:       3908           507        3186           0          214        3176
交换:         947           559         387

```

查阅相关资料后发现不是缺少 swap 交换分区（内存不够）的问题，于是尝试重新步骤 4，这次将所有可勾选的选项全部取消，问题解决。

6. 编译完成：

第一次完成结果如下图：

```

apple@ubuntu: ~/桌面/linux-5.5.8
CC arch/x86/boot/memory.o
CC arch/x86/boot/pm.o
AS arch/x86/boot/pmjump.o
CC arch/x86/boot/printf.o
CC arch/x86/boot/regs.o
CC arch/x86/boot/string.o
CC arch/x86/boot/tty.o
CC arch/x86/boot/video.o
CC arch/x86/boot/video-mode.o
CC arch/x86/boot/version.o
CC arch/x86/boot/video-vga.o
CC arch/x86/boot/video-vesa.o
CC arch/x86/boot/video-bios.o
LD arch/x86/boot/setup.elf
OBJCOPY arch/x86/boot/setup.bin
OBJCOPY arch/x86/boot/vmlinux.bin
HOSTCC arch/x86/boot/tools/build
BUILD arch/x86/boot/bzImage
Setup is 18204 bytes (padded to 18432 bytes).
System is 32928 kB
CRC 7cf88d2f
Kernel: arch/x86/boot/bzImage is ready (#3)
GEN scripts/gdb/linux/constants.py
apple@ubuntu:~/桌面/linux-5.5.8$

```

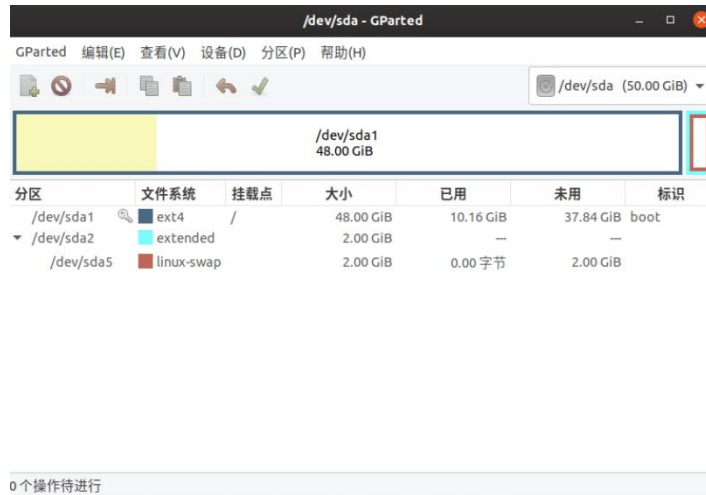
然而，在安装完 kernel 并重启 Ubuntu 并切换内核版本的时候 GRUB 崩溃，无法打开任意一版的内核。随后发现，在编译过程中有警告窗口“当前磁盘容量剩余 0KB”。

```

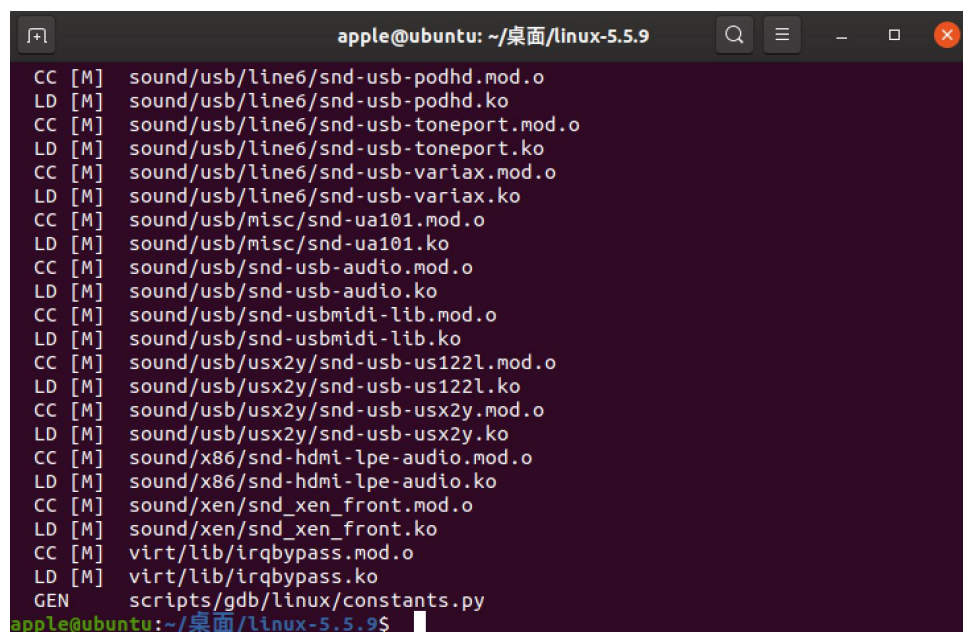
apple@ubuntu: ~/Desktop/linux-5.5.8
AR arch/x86/lib/built-in.a
AR virt/lib/built-in.a
CC [M] virt/lib/irqbypass.o
AR virt/built-in.a
GEN .version
CHK include/generated/compile.h
SKIPPED include/generated/compile.h
LD vmlinux.o
ld: vmlinux.o: final close failed: 设备上没有空间
make: *** [Makefile:1078: vmlinux] 错误 1
apple@ubuntu:~/Desktop/linux-5.5.8$ df -h
文件系统      容量  已用  可用  已用% 挂载点
udev          1.9G   0    1.9G   0% /dev
tmpfs         391M  1.8M  390M   1% /run
/dev/sda1      20G   19G   0  100% /
tmpfs         2.0G   0    2.0G   0% /dev/shm
tmpfs         5.0M  4.0K  5.0M   1% /run/lock
tmpfs         2.0G   0    2.0G   0% /sys/fs/cgroup
/dev/loop1     55M   55M   0  100% /snap/core18/1223
/dev/loop0     15M   15M   0  100% /snap/gnome-characters/317
/dev/loop2    150M  150M   0  100% /snap/gnome-3-28-1804/71
/dev/loop4     45M   45M   0  100% /snap/gtk-common-themes/1353
/dev/loop5     1.0M   1.0M   0  100% /snap/gnome-logs/81
/dev/loop6     90M   90M   0  100% /snap/core/7917

```


在将虚拟机磁盘分别扩展至 30GB、40GB、50GB 后仍然是这个问题。查阅资料后发现未在 Ubuntu 系统内部利用 `sudo gparted` 指令重新分配磁盘空间（有 30GB 未分配的磁盘空间）。



重新分配空间（48GB 虚拟机，2GBswap 空间），在 `menuconfig` 中将所有选项勾选并编译完成后结果如下：



共 2 小时 20 分钟左右。(过程中最新版 kernel 已经从 5.5.8 升级到 5.5.9)。

7. 安装:

执行如下指令:

```
apple@ubuntu:~/桌面/linux-5.5.9$ sudo make modules_install
```

运行结果如下:

```
apple@ubuntu: ~/桌面/linux-5.5.9
INSTALL sound/soc/xtensa/snd-soc-xtfpga-i2s.ko
INSTALL sound/soc/zte/zx-tdm.ko
INSTALL sound/soundcore.ko
INSTALL sound/synth/emux/snd-emux-synth.ko
INSTALL sound/synth/snd-util-mem.ko
INSTALL sound/usb/6fire/snd-usb-6fire.ko
INSTALL sound/usb/bcd2000/snd-bcd2000.ko
INSTALL sound/usb/caiaq/snd-usb-caiaq.ko
INSTALL sound/usb/hiface/snd-usb-hiface.ko
INSTALL sound/usb/line6/snd-usb-line6.ko
INSTALL sound/usb/line6/snd-usb-pod.ko
INSTALL sound/usb/line6/snd-usb-podhd.ko
INSTALL sound/usb/line6/snd-usb-toneport.ko
INSTALL sound/usb/line6/snd-usb-variax.ko
INSTALL sound/usb/misc/snd-ua101.ko
INSTALL sound/usb/snd-usb-audio.ko
INSTALL sound/usb/snd-usbmidi-lib.ko
INSTALL sound/usb/usx2y/snd-usb-usx2l.ko
INSTALL sound/usb/usx2y/snd-usb-usx2y.ko
INSTALL sound/x86/snd-hdmi-lpe-audio.ko
INSTALL sound/xen/snd_xen_front.ko
INSTALL virt/lib/irqbypass.ko
DEPMOD 5.5.9
apple@ubuntu:~/桌面/linux-5.5.9$
```

再执行命令:

```
apple@ubuntu:~/桌面/linux-5.5.9$ sudo make install
sh -c 'arch/x86/boot/install sh 5.5.9 arch/x86/boot/bzImage \
```

结果如下:

```
apple@ubuntu: ~/桌面/linux-5.5.9
run-parts: executing /etc/kernel/postinst.d/apt-auto-removal 5.5.9 /boot/vmlinuz-5.5.9
run-parts: executing /etc/kernel/postinst.d/initramfs-tools 5.5.9 /boot/vmlinuz-5.5.9
update-initramfs: Generating /boot/initrd.img-5.5.9
run-parts: executing /etc/kernel/postinst.d/unattended-upgrades 5.5.9 /boot/vmlinuz-5.5.9
run-parts: executing /etc/kernel/postinst.d/update-notifier 5.5.9 /boot/vmlinuz-5.5.9
run-parts: executing /etc/kernel/postinst.d/zz-update-grub 5.5.9 /boot/vmlinuz-5.5.9
Sourcing file '/etc/default/grub'
Sourcing file '/etc/default/grub.d/init-select.cfg'
Generating grub configuration file ...
Found linux image: /boot/vmlinuz-5.5.9
Found initrd image: /boot/initrd.img-5.5.9
Found linux image: /boot/vmlinuz-5.3.0-40-generic
Found initrd image: /boot/initrd.img-5.3.0-40-generic
Found linux image: /boot/vmlinuz-5.3.0-18-generic
Found initrd image: /boot/initrd.img-5.3.0-18-generic
Found memtest86+ image: /boot/memtest86+.elf
Found memtest86+ image: /boot/memtest86+.bin
done
apple@ubuntu:~/桌面/linux-5.5.9$
```

安装完成。

8. 重启验证:

在未安装最新版本 Linux Kernel 之前, Ubuntu19.10 自带内核版本为 Linux-5.3.0-40-generic, 如下图:

```
apple@ubuntu:~/桌面/linux-5.5.9$ uname -a
Linux ubuntu 5.3.0-40-generic #32-Ubuntu SMP Fri Jan 31 20:24:34 UTC 2020 x86_64
x86_64 x86_64 GNU/Linux
apple@ubuntu:~/桌面/linux-5.5.9$
```

安装完成并重启后, 可以看到系统已更新到最新版本的内核 Linux-5.5.9。

```
apple@ubuntu:~$ uname -a
Linux ubuntu 5.5.9 #1 SMP Thu Mar 12 03:00:11 PDT 2020 x86_64 x86_64 x86_64 GNU
/Linux
apple@ubuntu:~$ uname -r
5.5.9
apple@ubuntu:~$
```

四、内容二: 编写 Kernel Modules

(部分过程待实机展示)

1. simple.c:

(1) 代码实现:

```

1  #include <linux/init.h>
2  #include <linux/kernel.h>
3  #include <linux/module.h>
4  #include <linux/hash.h>
5  #include <linux/gcd.h>
6  #include <asm/param.h>
7  #include <linux/jiffies.h>
8
9  static int startTime, endTime;
10
11 /* This function is called when the module is loaded. */
12 int simple_init(void)
13 {
14     printk(KERN_INFO "Loading Kernel Module\n");
15     // Print values of GOLDEN_RATIO_PRIME
16     printk(KERN_INFO "The value GOLDEN_RATIO_PRIME is: %lu\n", GOLDEN_RATIO_PRIME);
17     // Record the start time
18     startTime = jiffies;
19     // Print values of jiffies and HZ
20     printk(KERN_INFO "The value of jiffies when init is: %lu\n", jiffies);
21     printk(KERN_INFO "The value of HZ is: %lu\n", HZ);
22     return 0;
23 }
24
25 /* This function is called when the module is removed. */
26 void simple_exit(void)
27 {
28     printk(KERN_INFO "Removing Kernel Module\n");
29     // Print the result of gcd
30     printk("The result of gcd is: %lu\n", gcd(3300, 24));
31     // Print values of jiffies when exit
32     printk("The value of jiffies when exit is: %lu\n", jiffies);
33     // Record the end time
34     endTime = jiffies;
35     // Print the total number of seconds that have elapsed
36     printk("The total elapsing time is: %lu\n", (endTime - startTime) / HZ);
37 }
38
39 /* Macros for registering module entry and exit points. */
40 module_init(simple_init);
41 module_exit(simple_exit);
42
43 MODULE_LICENSE("GPL");
44 MODULE_DESCRIPTION("Simple Module");
45 MODULE_AUTHOR("Chunyu Xue");

```

(2) 运行结果:

```

apple@ubuntu:~/桌面/OSProject1$ sudo dmesg -c
[sudo] apple 的密码:
apple@ubuntu:~/桌面/OSProject1$ sudo insmod simple.ko
apple@ubuntu:~/桌面/OSProject1$ dmesg
[ 6506.619578] Loading Kernel Module...
[ 6506.619579] The value GOLDEN_RATIO_PRIME is: 7046029254386353131
[ 6506.619580] The value of jiffies when init is: 4296518706
[ 6506.619580] The value of HZ is: 250
apple@ubuntu:~/桌面/OSProject1$ sudo rmmod simple
apple@ubuntu:~/桌面/OSProject1$ dmesg
[ 6506.619578] Loading Kernel Module...
[ 6506.619579] The value GOLDEN_RATIO_PRIME is: 7046029254386353131
[ 6506.619580] The value of jiffies when init is: 4296518706
[ 6506.619580] The value of HZ is: 250
[ 6518.121888] Removing Kernel Module...
[ 6518.121890] The result of gcd is: 12
[ 6518.121891] The value of jiffies when exit is: 4296521581
[ 6518.121891] The total elapsing time is: 11
apple@ubuntu:~/桌面/OSProject1$

```

2. jiffies、seconds:

(1) 代码实现:

jiffies:

```
1  #include <linux/init.h>
2  #include <linux/kernel.h>
3  #include <linux/module.h>
4  #include <linux/proc_fs.h>
5  #include <asm/uaccess.h>
6  #include <linux/jiffies.h>
7  #include <linux/uaccess.h>
8
9  #define BUFFER_SIZE 128
10 #define PROC_NAME "jiffies"
11
12 ssize_t proc_read(struct file *file, char __user *usr_buf, size_t count, loff_t *pos);
13
14 static struct file_operations proc_ops = {
15     .owner = THIS_MODULE,
16     .read = proc_read,
17 };
18
19 /* This function is called when the module is loaded. */
20 int proc_init(void)
21 {
22     /* Create the /proc/jiffies entry */
23     proc_create(PROC_NAME, 0666, NULL, &proc_ops);
24     printk(KERN_INFO "Successfully create /proc file...\n", PROC_NAME);
25
26     return 0;
27 }
28
29 /* This function is called when the module is removed. */
30 void proc_exit(void)
31 {
32     /* Remove the /proc/hello entry */
33     remove_proc_entry(PROC_NAME, NULL);
34     printk(KERN_INFO "Successfully remove /proc file...\n", PROC_NAME);
35 }
36
37 /* This function is called each time /proc/hello is read */
38 ssize_t proc_read(struct file *file, char __user *usr_buf, size_t count, loff_t *pos)
39 {
40     int rv = 0;
41     char buffer[BUFFER_SIZE];
42     static int completed = 0;
43
44     if(completed){
45         completed = 0;
```

```
46         return 0;
47     }
48
49     completed = 1;
50
51     rv = sprintf(buffer, "The value of jiffies is: %lu\n", jiffies);
52
53     /* Copies kernel space buffer to user space usr_buf */
54     copy_to_user(usr_buf, buffer, rv);
55
56     return rv;
57 }
58
59 module_init(proc_init);
60 module_exit(proc_exit);
61
62 MODULE_LICENSE("GPL");
63 MODULE_DESCRIPTION("Hello Module");
64 MODULE_AUTHOR("Chunyu Xue");
65
66
```



seconds:

```

1  #include <linux/init.h>
2  #include <linux/kernel.h>
3  #include <linux/module.h>
4  #include <linux/proc_fs.h>
5  #include <asm/uaccess.h>
6  #include <asm/param.h>
7  #include <linux/jiffies.h>
8  #include <linux/uaccess.h>
9
10 #define BUFFER_SIZE 128
11 #define PROC_NAME "seconds"
12
13 static long unsigned int startTime;
14
15 ssize_t proc_read(struct file *file, char __user *usr_buf, size_t count, loff_t *pos);
16
17 static struct file_operations proc_ops = {
18     .owner = THIS_MODULE,
19     .read = proc_read,
20 };
21
22 /* This function is called when the module is loaded. */
23 int proc_init(void)
24 {
25     // Record the start time
26     startTime = jiffies;
27
28     /* Create the /proc/jiffies entry */
29     proc_create(PROC_NAME, 0666, NULL, &proc_ops);
30     printk(KERN_INFO "Successfully create /proc file...\n", PROC_NAME);
31
32     return 0;
33 }
34
35 /* This function is called when the module is removed. */
36 void proc_exit(void)
37 {
38     /* Remove the /proc/hello entry */
39     remove_proc_entry(PROC_NAME, NULL);
40     printk(KERN_INFO "Successfully remove /proc file...\n", PROC_NAME);
41 }
42
43 /* This function is called each time /proc/hello is read */
44 ssize_t proc_read(struct file *file, char __user *usr_buf, size_t count, loff_t *pos)
45 {
46     int rv = 0;
47     char buffer[BUFFER_SIZE];
48     static int completed = 0;
49
50     if(completed){
51         completed = 0;
52         return 0;
53     }
54
55     completed = 1;
56
57     rv = sprintf(buffer, "The running time is: %lu\n", (jiffies - startTime) / HZ);
58
59     /* Copies kernel space buffer to user space usr_buf */
60     copy_to_user(usr_buf, buffer, rv);
61
62     return rv;
63 }
64
65 module_init(proc_init);
66 module_exit(proc_exit);
67
68 MODULE_LICENSE("GPL");
69 MODULE_DESCRIPTION("Hello Module");
70 MODULE_AUTHOR("Chunyu Xue");
71

```

(2) 运行结果:

```
apple@ubuntu:~/桌面/OSProject1$ sudo insmod jiffies.ko
apple@ubuntu:~/桌面/OSProject1$ sudo insmod seconds.ko
apple@ubuntu:~/桌面/OSProject1$ dmesg
[ 6586.054207] Successfully create /proc file...
[ 6594.655264] Successfully create /proc file...
apple@ubuntu:~/桌面/OSProject1$ cat /proc/jiffies
The value of jiffies is: 4296544101
apple@ubuntu:~/桌面/OSProject1$ cat /proc/seconds
The running time is: 20
apple@ubuntu:~/桌面/OSProject1$ sudo rmmod jiffies
apple@ubuntu:~/桌面/OSProject1$ sudo rmmod seconds
apple@ubuntu:~/桌面/OSProject1$ dmesg
[ 6586.054207] Successfully create /proc file...
[ 6594.655264] Successfully create /proc file...
[ 6627.898718] Successfully remove /proc file...
[ 6634.718404] Successfully remove /proc file...
apple@ubuntu:~/桌面/OSProject1$
```

3. Makefile:

```
1  obj-m += simple.o jiffies.o seconds.o
2
3  all :
4      make -C /lib/modules/$(shell uname -r)/build M=$(PWD) modules
5
6  clean:
7      make -C /lib/modules/$(shell uname -r)/build M=$(PWD) clean
8
```

五、实验结果

在解决遇到的一系列问题，经过两天的不断尝试之后，终于顺利在 Ubuntu19.10 上编译并安装了最新版的 Linux-5.5.9 内核。

按照指示编写 simple.c、jiffies.c、seconds.c 等三个 c 文件，以及一个 Makefile，运行后能够成功完成相应的任务。

六、实验反思

1. 遇到问题时多利用网络资源（如 CSDN、博客园等网站）；
2. 就算无法直接查询到结果，也可借鉴别人类似的方法多加尝试；
3. 在解决问题时要有耐心。

