操作系统实验：向 Linux 内核增加一个系统调用

姓名：周嘉莹

学号：71118321

报告日期 2019/3/29

一、实验目的

通过实验，熟悉 Linux 操作系统的使用，掌握构建与启动 Linux 内核的方法；掌握用户程序如何利用系统调用与操作系统内核实现通信的方法，加深对系统调用机制的理解；进一步掌握如何向操作系统内核增加新的系统调用的方法，以扩展操作系统的功能。

二、实验内容

1. Linux环境下的C或C++编译和调试工具的使用。
2. 向Linux内核增加新的系统调用，系统调用名称和功能自行定义，但必须实现如下输出功能："My Student No. is ×××，and My Name is ×××"。
3. Linux新内核的编译、安装和配置。
4. 编写应用程序以测试新的系统调用并输出测试结果。

三、实验步骤

1.先切换到 root 用户,把下载好的内核压缩文件放到/usr/src 下，然后进入目录解压

```
xz -d linux-4.14.107.tar.xz
tar -xvf linux-4.14.107.tar
sudo mv /下载/linux-4.14.107 /usr/src
```

2.打开文件

```
/arch/x86/entry/syscalls/syscall_64.tbl
```

添加一条新的系统调用号。本机为 333，系统调用函数名为"hello"

```
328      64        pwritev2                sys_pwritev2
329      common    pkey_mprotect           sys_pkey_mprotect
330      common    pkey_alloc              sys_pkey_alloc
331      common    pkey_free               sys_pkey_free
332      common    statx                   sys_statx
333      64        hello                   sys_hello
```

3.在系统调用的头文件中添加函数声明

```
/include/linux/syscalls.h
```

```
asmlinkage long sys_statx(int dfd, const char __user *path, unsigned flags,
                          unsigned mask, struct statx __user *buffer);
asmlinkage long sys_hello(void);

#endif
```

4.在最后一个文件中定义系统调用函数

```
/kernel/sys.c
```

```
#endif /* CONFIG_COMPAT */
asmlinkage long __sys_hello(void){
        printk("My Student No.is 71118321, and My Name is 周嘉莹.");
        printk("Have fun!");
        return 1;
}
```

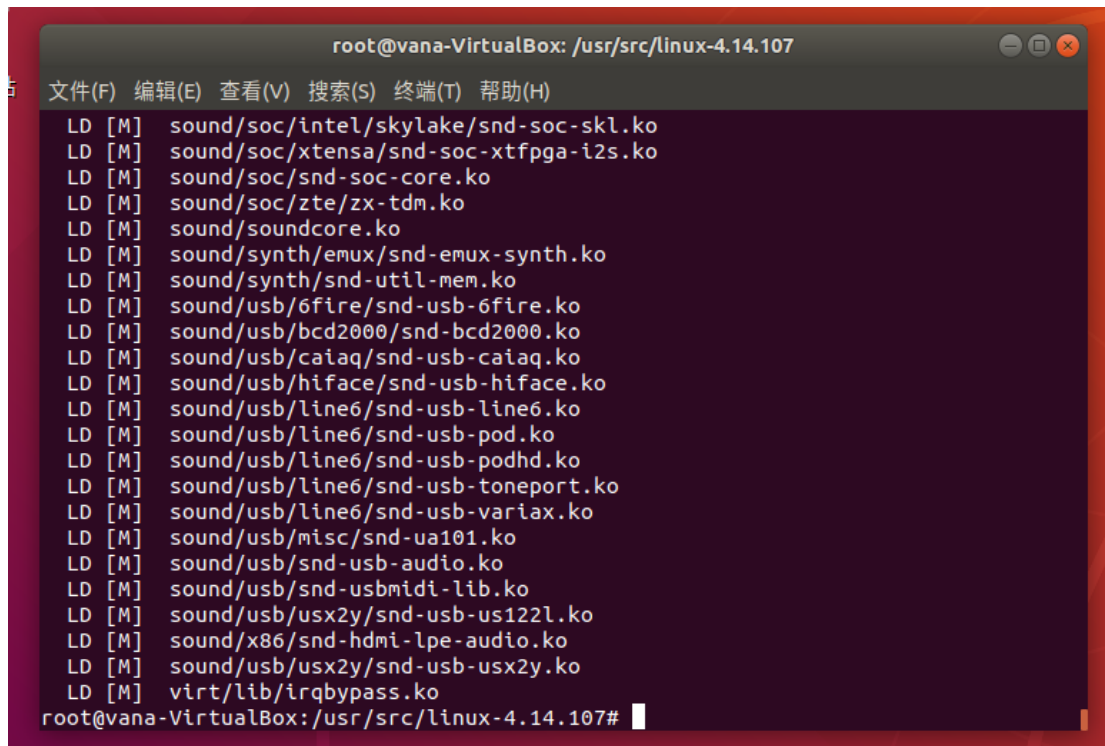"printk"为输出至系统日志，可通过 dmesg 命令显示出来

5.配置内核选项

```
make menuconfig
```

默认内容即可，这里将系统版本名称改为了"hello"
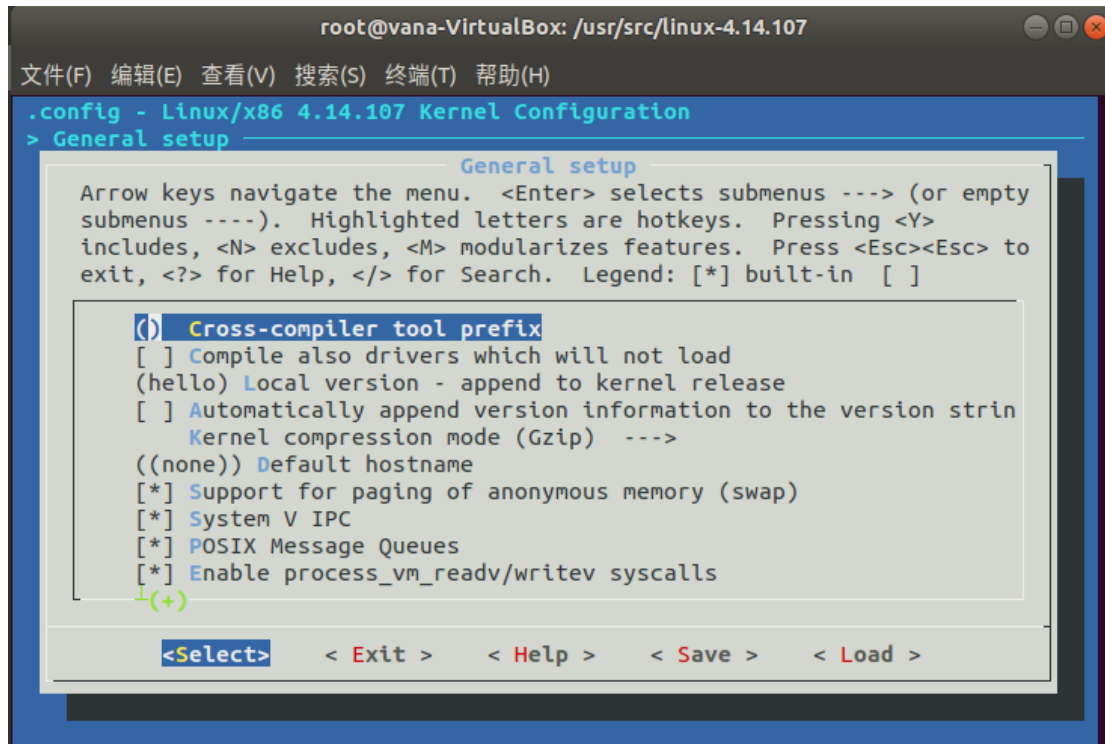
6.开始编译内核

```
make
make modules_install
make install
```

7.编译完成后重启进入"高级选项"，选择刚刚添加的内核开机

已成功切换为新的内核

8.创建"hello.c"文件，编写程序使用系统调用

```c
#include<stdio.h>
#include<linux/kernel.h>
#include<sys/syscall.h>
#include<linux/unistd.h>
int main()
{
        syscall(333);
        return 0;

}
```

9.编译运行"hello.c"，之后通过"dmesg"查看系统日志

已成功使用系统调用

五、主要数据结构及其说明

主要使用 root 权限："su root"完成相应操作
编译.c 文件使用"gcc"

六、程序运行时的初值和运行结果

```
    LD [M]   sound/soc/intel/skylake/snd-soc-skl.ko
    LD [M]   sound/soc/xtensa/snd-soc-xtfpga-i2s.ko
    LD [M]   sound/soc/snd-soc-core.ko
    LD [M]   sound/soc/zte/zx-tdm.ko
    LD [M]   sound/soundcore.ko
    LD [M]   sound/synth/emux/snd-emux-synth.ko
    LD [M]   sound/synth/snd-util-mem.ko
    LD [M]   sound/usb/6fire/snd-usb-6fire.ko
    LD [M]   sound/usb/bcd2000/snd-bcd2000.ko
    LD [M]   sound/usb/caiaq/snd-usb-caiaq.ko
    LD [M]   sound/usb/hiface/snd-usb-hiface.ko
    LD [M]   sound/usb/line6/snd-usb-line6.ko
    LD [M]   sound/usb/line6/snd-usb-pod.ko
    LD [M]   sound/usb/line6/snd-usb-podhd.ko
    LD [M]   sound/usb/line6/snd-usb-toneport.ko
    LD [M]   sound/usb/line6/snd-usb-variax.ko
    LD [M]   sound/usb/misc/snd-ua101.ko
    LD [M]   sound/usb/snd-usb-audio.ko
    LD [M]   sound/usb/snd-usbmidi-lib.ko
    LD [M]   sound/usb/usx2y/snd-usb-us122l.ko
    LD [M]   sound/x86/snd-hdmi-lpe-audio.ko
    LD [M]   sound/usb/usx2y/snd-usb-usx2y.ko
    LD [M]   virt/lib/irqbypass.ko
root@vana-VirtualBox:/usr/src/linux-4.14.107#
```

```
  INSTALL sound/soc/snd-soc-core.ko
  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-us122l.ko
  INSTALL sound/usb/usx2y/snd-usb-usx2y.ko
  INSTALL sound/x86/snd-hdmi-lpe-audio.ko
  INSTALL virt/lib/irqbypass.ko
  DEPMOD  4.14.107hello
root@vana-VirtualBox:/usr/src/linux-4.14.107#
```

七、实验体会（实验中遇到的问题及解决方法、实验中产生的错误及原因分析、实验的体会及收获、对做好今后实验提出建设性建议等）

1.下载的内核版本高于系统支持的内核版本，导致不兼容，编译时出错。使用低版本内核后解决问题

2.加快编译速度可以使用"sudo make -j4"指令，使用 4 线程。

3.系统调用的输出函数为"printk"而非"printf"，输出到系统日志，需要使用"dmesg"指令查看。

八、源程序并附上注释

```
#endif /* CONFIG_COMPAT */
asmlinkage long __sys_hello(void){
        printk("My Student No.is 71118321, and My Name is 周嘉莹.");
        printk("Have fun!");
        return 1;
}
```