**实验报告**

**1.实验目的：**

1、建立对系统调用接口的深入认识  
2、掌握系统调用的基本过程  
3、能完成系统调用的全面控制

**2.实验内容:**

1. 添加系统调用要知道是对操作系统的内核进行操作；
2. 编译内核重新生成操作系统，安装之后重启电脑进入你重新生成的系统；
3. 测试添加的系统调用。

**3.实验流程**

1.下载linux-5.3.4.tar.xz

2.cd Desktop *//进入桌面*

*//tar jxvf linux-5.3.4.tar.bz2 //解压.bz2文件*

xz -d linux-5.3.4.tar.xz

tar xvf linux-5.3.4.tar *//解压.xz文件*

cp -r linux-5.3.4 /usr/src *//文件拷到/usr/src目录中*



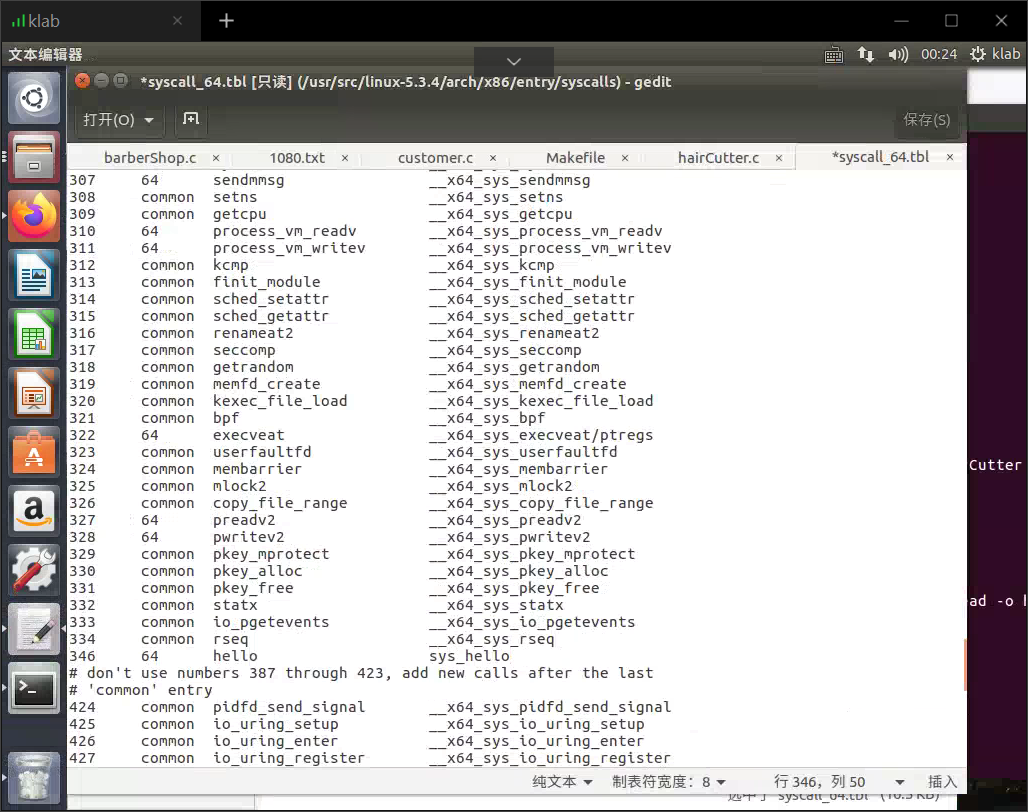
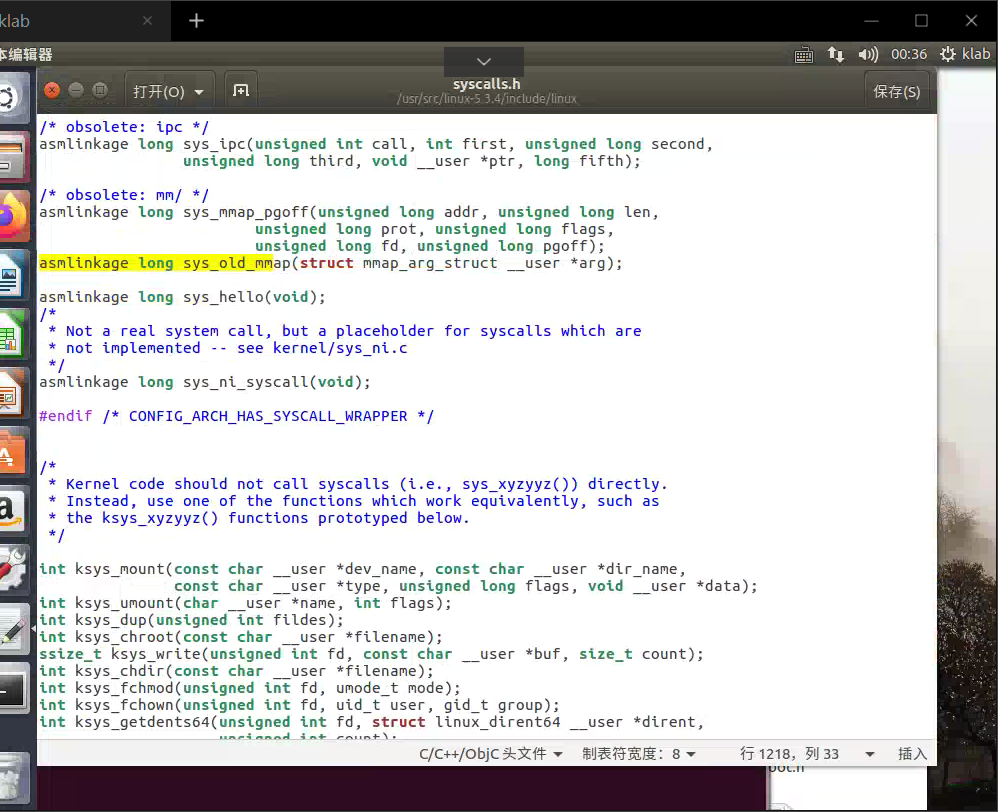
3.修改内核的代码了

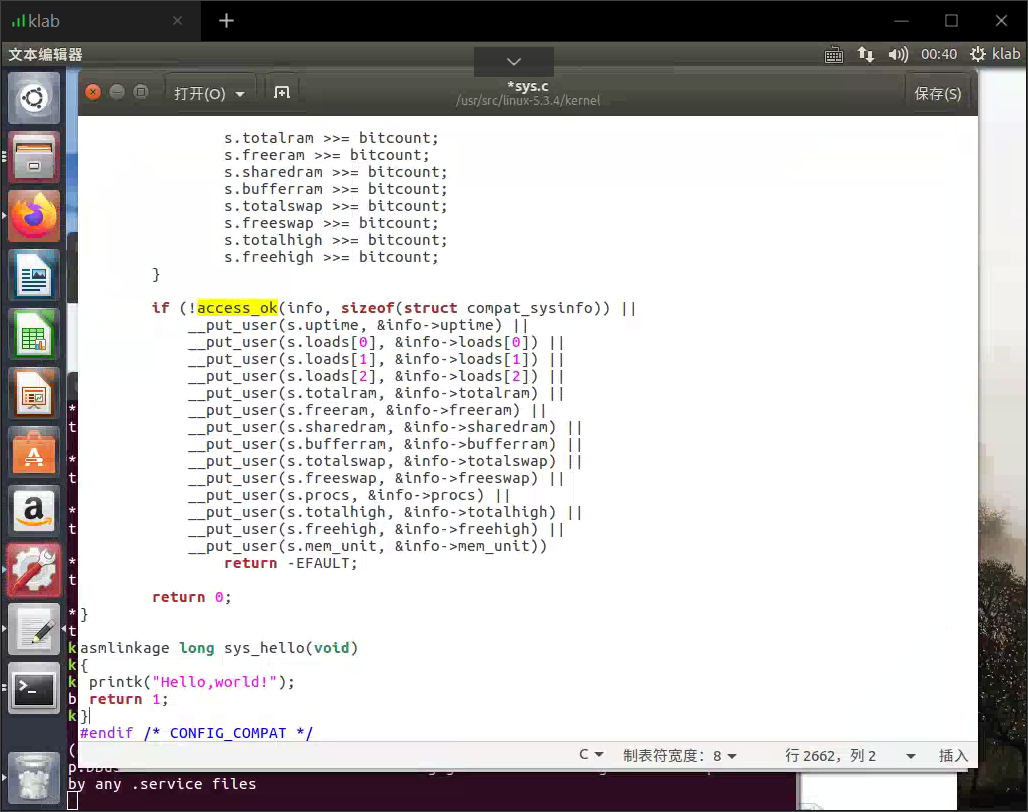
源码目录下涉及内核的三个文件有：

/arch/x86/entry/syscalls/syscall\_64.tbl      //设置系统调用号，如果是32位就用syscall\_32.tbl

/include/linux/syscalls.h                     //系统调用的头文件

/kernel/sys.c                                         //定义系统调用



安装 gcc make

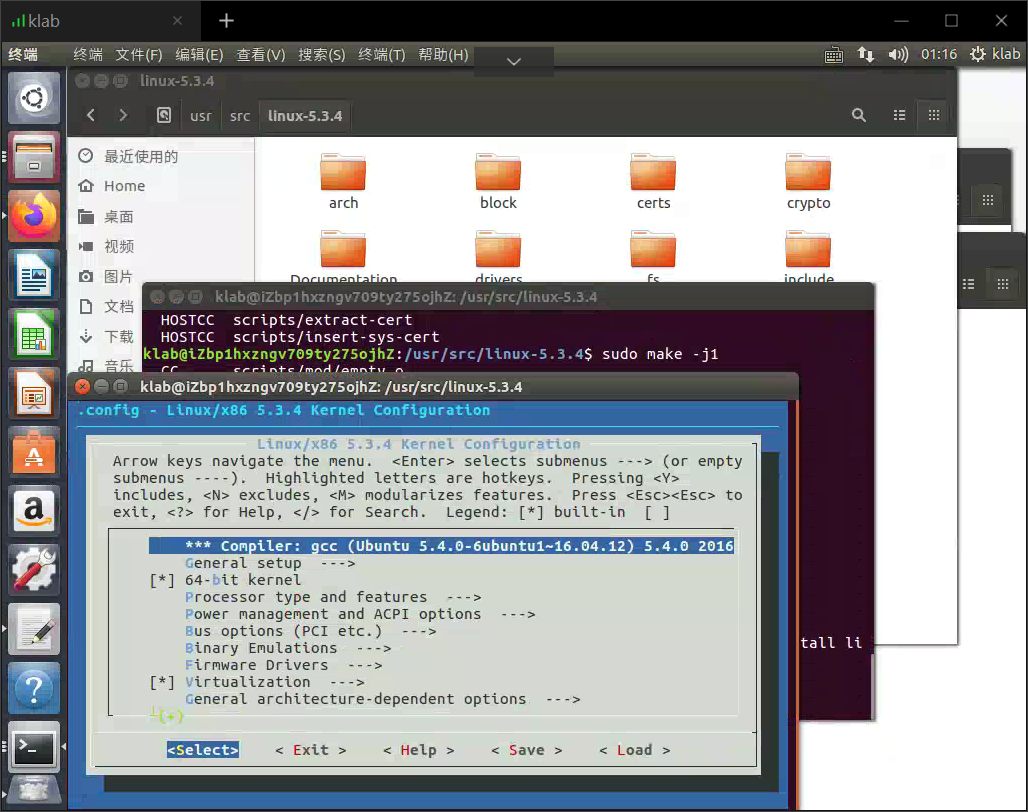
sudo apt-get install gcc make flex bison

sudo apt-get install libncurses5-dev libssl-dev



运行

sudo make menuconfig

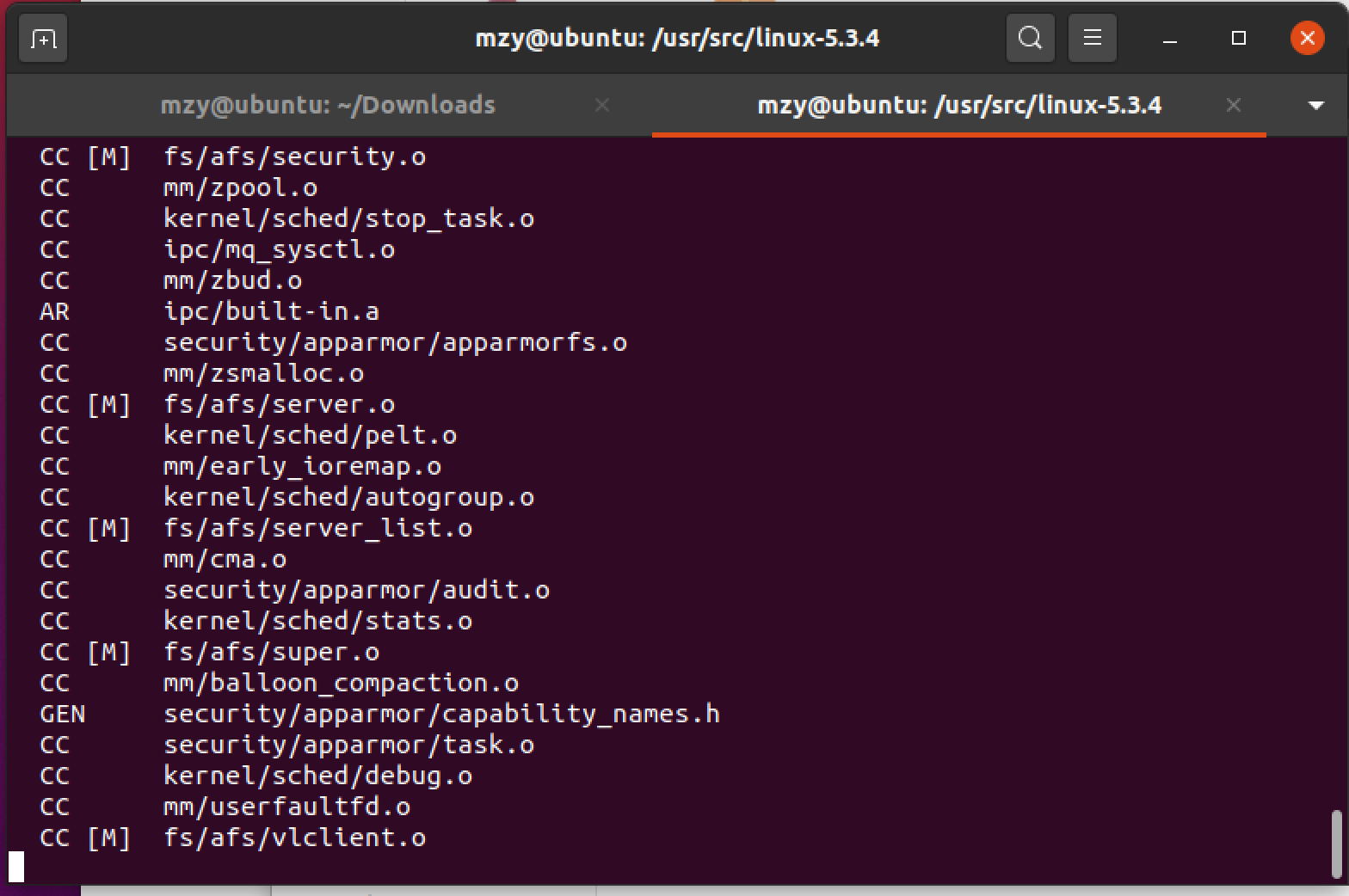


4.编译内核

make –j4 *//内核配置，比较耗时间*

使用多线程加速

等待很久



make install

make modules *//编译模块*

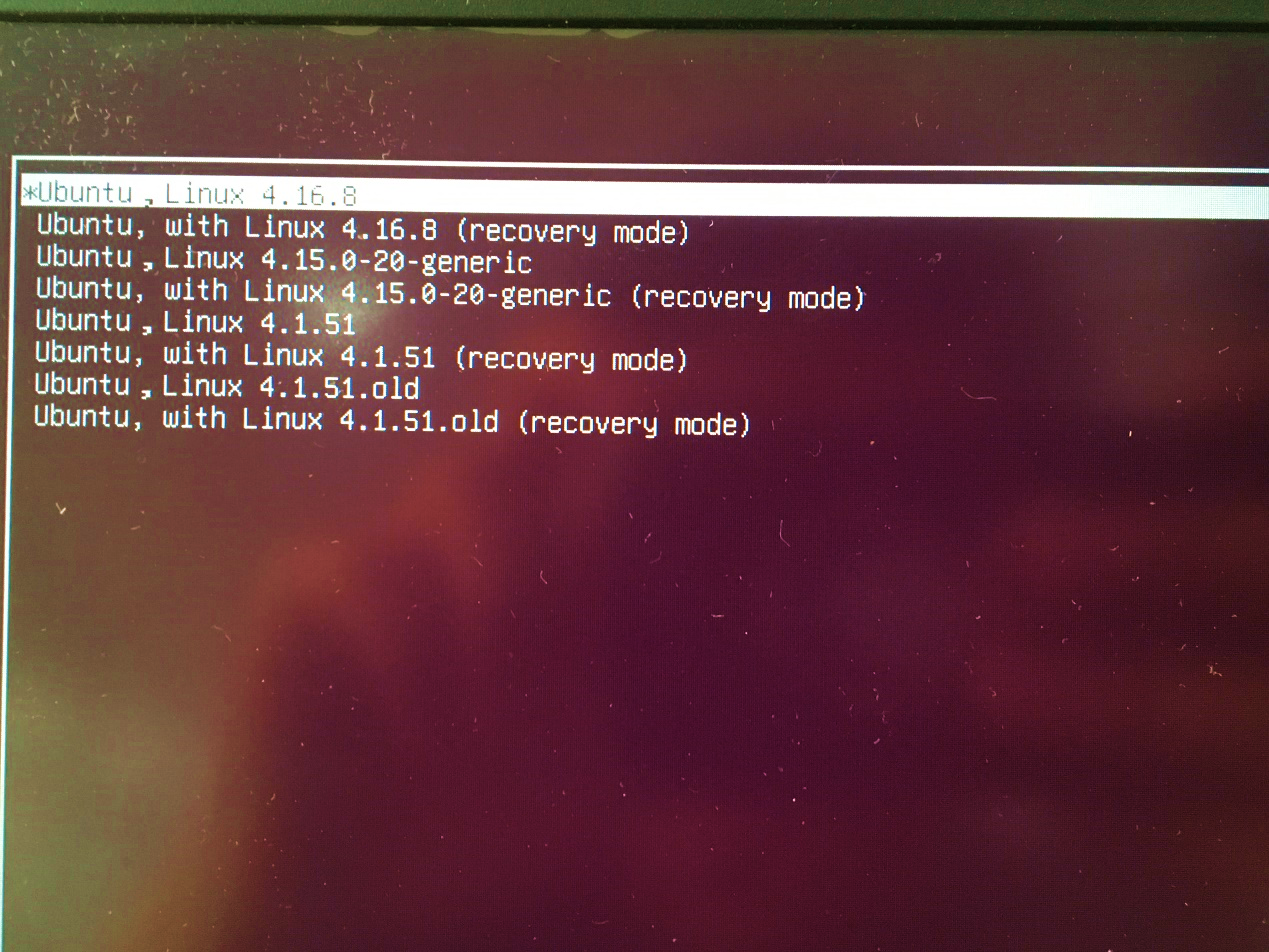
make modules\_install *//安装模块*

mkinitramfs -o /boot/initrd.img-5.3.4 *//最后创建initrd文件*

update-grub

reboot //重启，<span style="color:#FF0000;">出错信息请忽略</span>

uname –r //即可看到版*本号已经发生变化*



6.编写测试程序

cd Desktop

gedit test.c *//用记事本编写一个c程序*

#include <linux/unistd.h>

#include <sys/syscall.h>

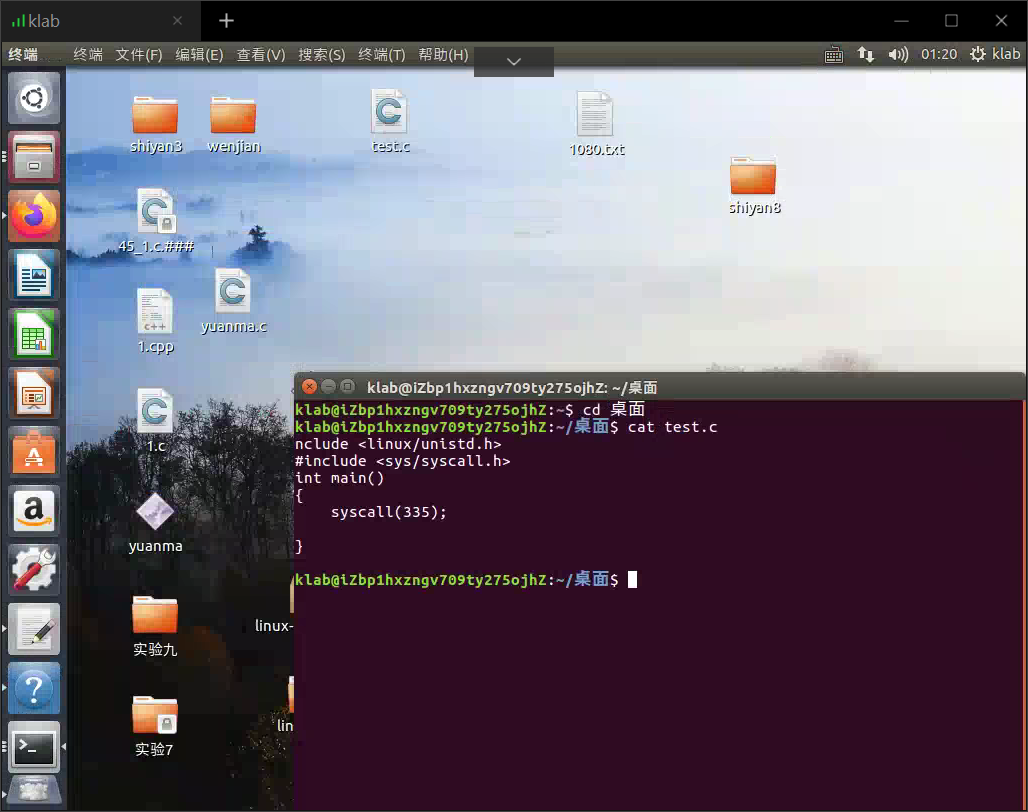
int main()

{

syscall(335);

}

*//编译运行*



gcc -o test test.c

./test

dmesg *//查看进程信息*

