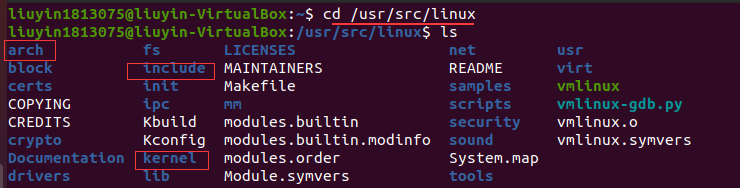
lab07 New System Call

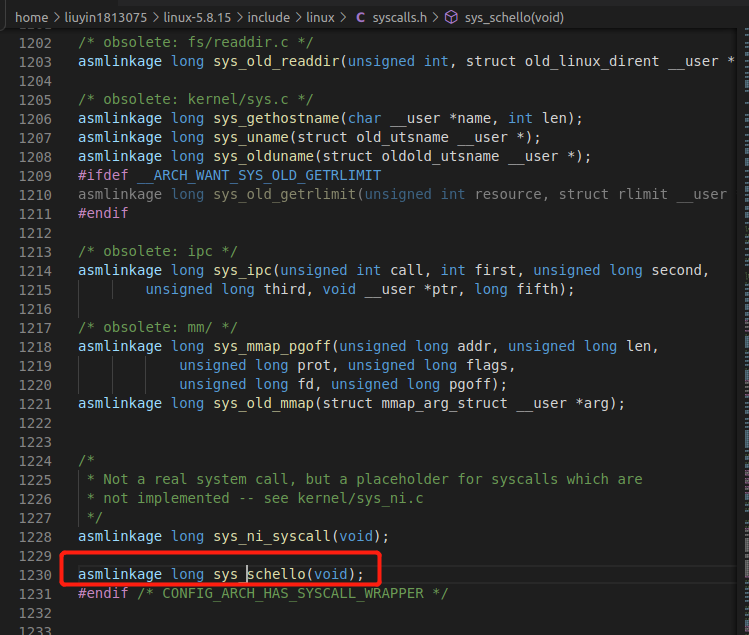
软件工程2018级 1813075 刘茵

1. 实验⽬标
2. 向linux内核中添加新的系统调用
3. 在⽤户态下尝试新的系统调⽤
4. 操作过程
5. part 1
6. 进入 usr/src/linux目录（和 home/linux5.8.15相同）



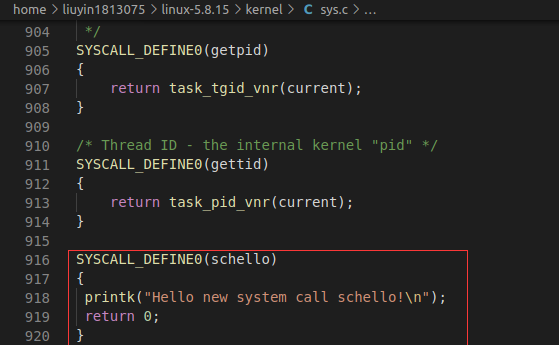
1. 用vscode打开include/linux/syscall.h ⽂件，并修改。增加1230行。



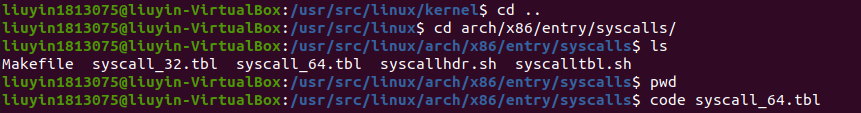


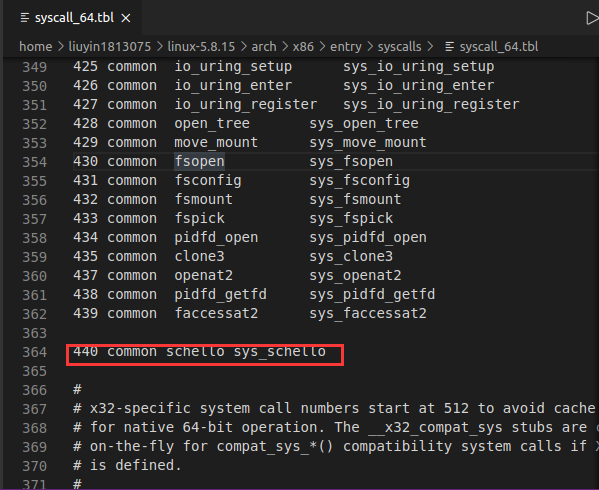
1. 修改/kernel/sys.c 文件，添加SYSCALL\_DEFINE0(schello)函数





1. 修改 arch/x86/entry/syscalls/syscall\_64.tbl文件，增加440





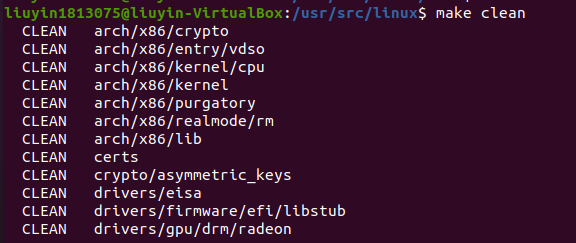
1. 依次运行以下命令编译内核。

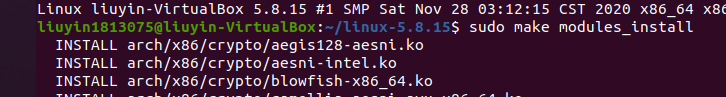
make clean

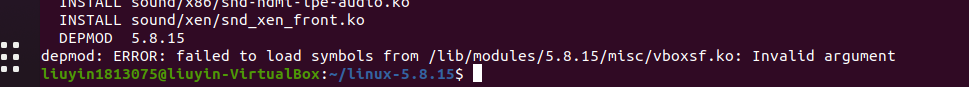
make -j5

sudo make modules\_install

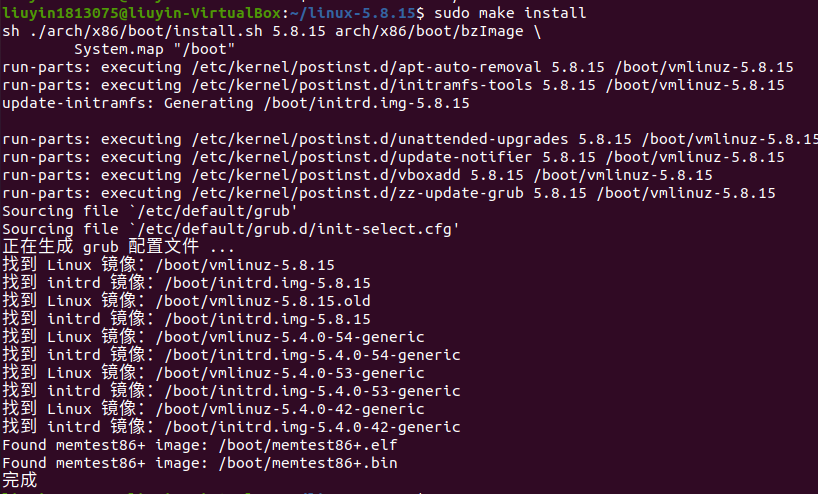
sudo make install



****

****

Ps: make modules\_install 结束后会报错error(但是对结果没有影响)，网上没有找到具体的匹配信息。

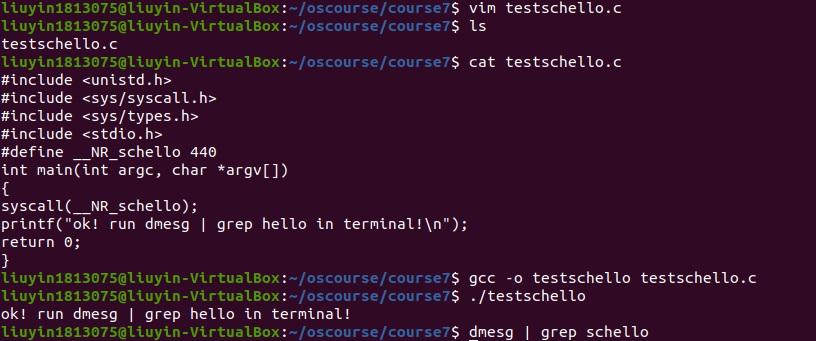


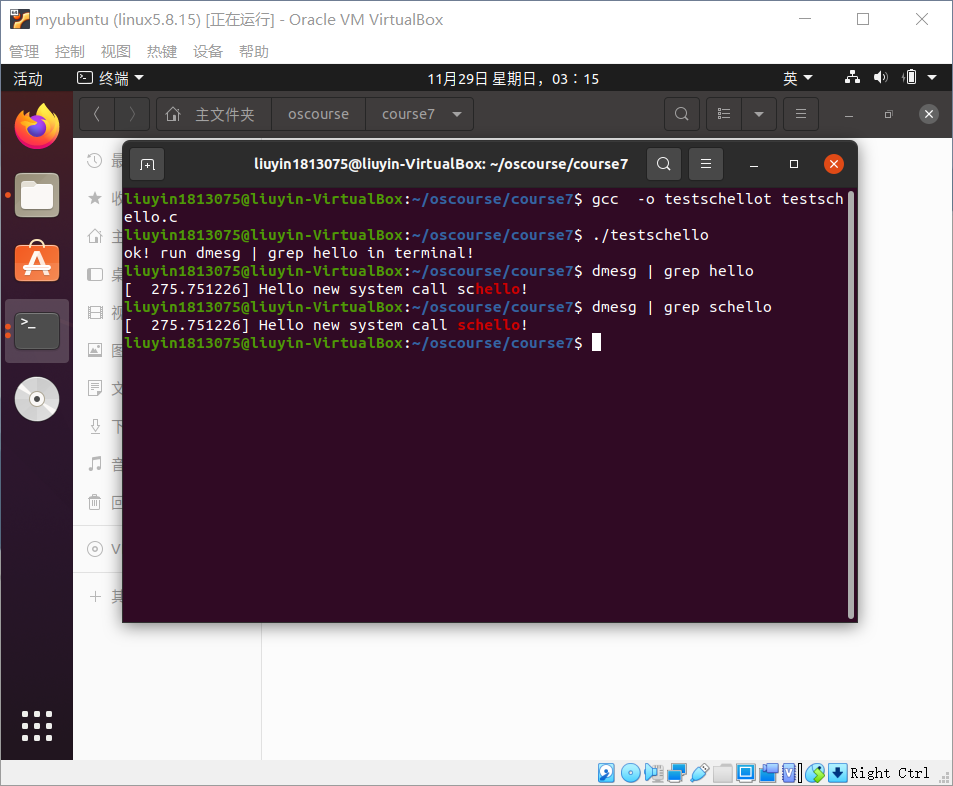
1. 编写testschello.c程序在⽤户态测试系统调⽤，并执行

gcc -o testschello testschello.c

./testschello

dmesg | grep schello （查看输出）





1. Part 2
2. 再次在kernel/sys.c⽂件中修改函数定义输出当前所有进程：

SYSCALL\_DEFINE0(schello)

{

struct task\_struct \*p;

printk("Hello new system call schello!\n");

printk("%-20s %-6s %-6s\n","Name","Pid","Stat");

for (p = &init\_task; (p = next\_task(p)) != &init\_task;)

printk("%-20s %-6d %-6ld\n",p->comm,p->pid,p->state);

return 0;

}

2. 重复<Part 1>的5-7步骤

3. 再次输出结果。

