实验一：Linux内核编译及添加系统调用

1. 实验内容

（1）添加一个系统调用，实现对指定进程的nice值的修改及读取功能，并返回进程最新的nice值及优先级prio。

（2）写一个简单的应用测试（1）中添加的系统调用。

（3）若程序中调用了Linux的内核函数，要求深入阅读相关函数源码。

1. 实验设计方案
2. 函数接口设计

本实验要求实现两个功能，读取指定进程的nice值以及修改指定进程的nice值。程序添加的系统调用的原型为：

int mysetnice ( pid\_t pid, int flag, int nicevalue, void \_\_user\* prio, void \_\_user\* nice);

参数含义：

pid：进程ID

flag：若为0，则读取nice值；若为1，则修改nice值

prio、nice：指向进程当前的优先级及nice值

返回值：系统调用成功则返回0，失败时返回错误码EFAULT

（2）函数设计思路及程序流程图

函数中主要使用到了系统调用set\_user\_nice()来对进程的的nice值进行修改。该函数的原型为void set\_user\_nice(struct task\_struct \*p, long nice)，该程序可实现将进程p的nice值修改为nice。

由于函数需要的参数为tack\_struct \*p，因此首先需要根据用户输入的pid找到对应的task\_struct。

程序中借助find\_get\_pid和pid\_task来实现，首先借助pid调用find\_get\_pid找到pid对应的struct pid，接着根据struct pid调用pid\_task找到相应的task\_struct。

判断flag，若为0则将当前进程的nice值及prio值传回用户空间；

若为1，则将进程的nice值修改为用户输入的nice值，重新获取当前的nice值及prio值，并将其传回用户空间。



（3）系统调用添加步骤

首先在系统调用表中为新添加的系统调用分配系统调用号；



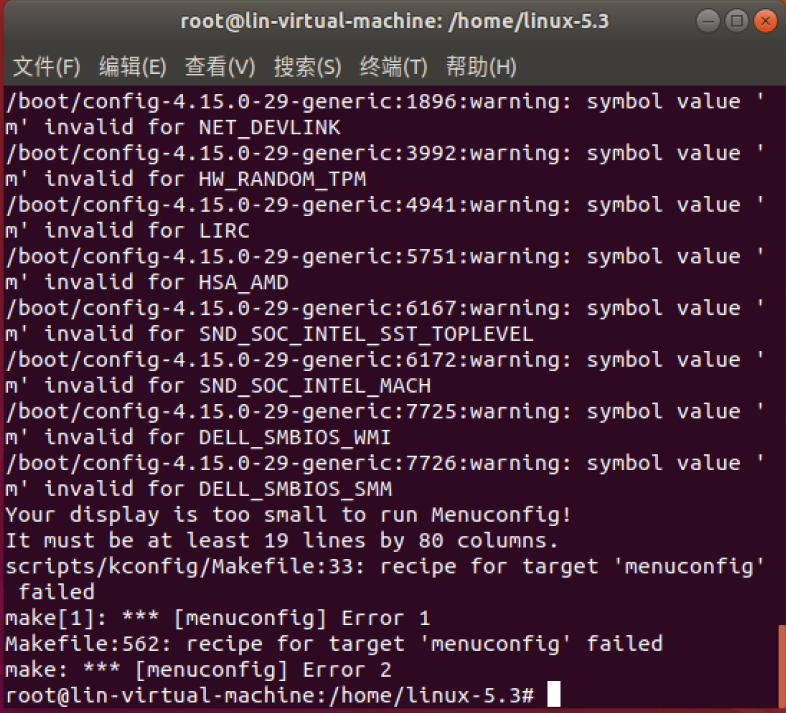
接着，声明系统调用服务例程原型：在文件./include/linux/syscalls.h末尾添加；



然后，实现系统调用服务例程，添加在/kernel/sys.c中；

最后，重新编译内核。

1. 实验中遇到的问题
2. 编译内核时，执行“make menuconfig”命令时报错



问题是由于当前terminal窗口太小，无法显示出menuconfig界面，将terminal窗口放大即可。

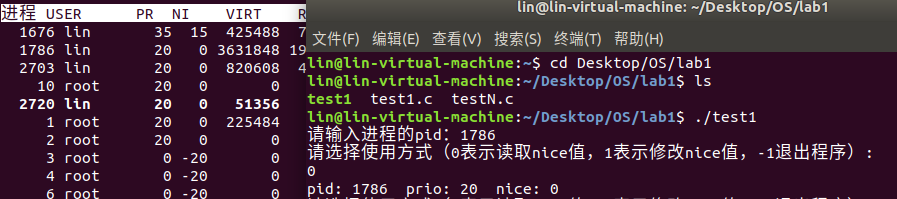
1. 最开始时实验结果，输出的进程的prio为0



程序中flag为0时，没有获取进程的prio值，导致结果出现错误。

1. 实验结果讨论

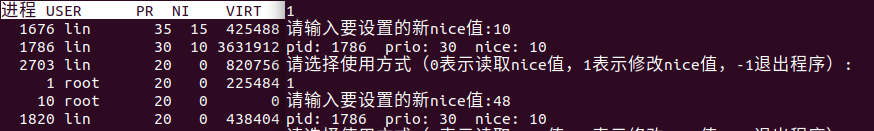
运行测试程序，实验结果的截图如下：



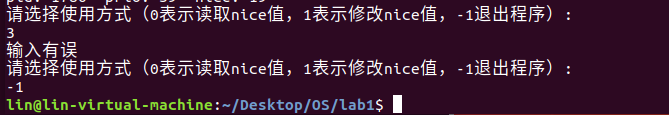
开始时，输入进程的pid为1786，输入0，获取进程的prio值及nice值，得到结果20 0，利用top命令查看进程1786的prio与nice值，看出结果正确；



接着，输入1，修改进程的nice值为10，由于10在nice值的范围（-20~19）内，进程的nice值得到修改，返回值为34 14，利用top命令看到进程1786的prio与nice值确实得到了修改。



接着，输入1，修改进程的nice值为48，由于nice值的取值范围是-20~19，超出范围根据set\_user\_nice函数的定义，进程的nice值不被修改。同样利用top命令验证结果，符合预期。



输入3，报错提示。

输入-1，退出程序。