参考文献：

知乎公众号：[Linux内核园](https://www.zhihu.com/people/linuxnei-he-yuan" \t "_blank)

Linux内核编译及添加系统调用（详细版）

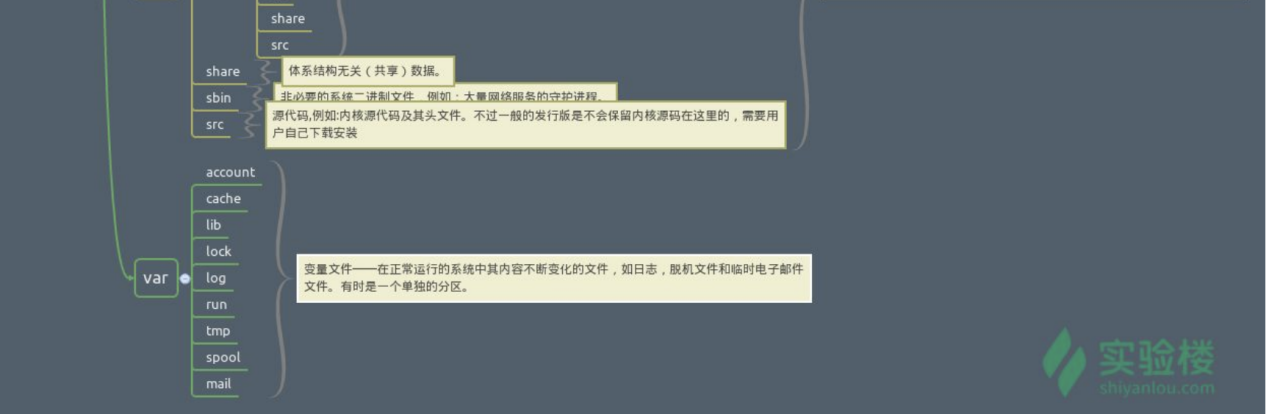
**实验一：Linux内核编译及添加系统调用（HDU）**

花了一上午的时间来写这个，良心制作，发现自己刚学的时候没有找到很详细的，就是泛泛的说了下细节地方也没有，于是自己写了这个，有点长，如果你认真的看完了，也应该是懂了。

## 前期准备工作

1. 需要准备虚拟机上安装Ubuntu，笔者安装的是**Ubuntu18.04**，安装的教程自行百度解决，教程很多。**有几点需要提一下，就是内存分配至少60G，核分配4个最好，为了在编译的时候别崩溃。**  
建议去熟悉一下Linux下面的文件目录结构，根目录下每个目录一般会存放什么样的文件





1. 然后常见命令操作也要熟悉一下。

2. 下载Linux内核[地址](https://link.zhihu.com/?target=https%3A//www.kernel.org/" \t "_blank),自行选择版本，建议选择4.xx版本，因为版本高出错的概率也大。  
下载好了之后，会放在自己的Ubuntu中的Downloads目录下，同时是一个压缩文件，到时候需要解压到放内核目录文件下。首先进入到该Downloads文件目录下，查看是否下载好了。

$cd ~/Downloads

$ls

linux-4.19.25.tar.xz

之后开始解压上面的那个压缩文件到存放内核的地方，就是Linux系统的**/usr/src**目录下，此目录用来存放内核源码的。从上图也可以了解到。

cd ~/Downloads

tar xvJf linux-4.19.25.tar.xz -C /usr/src

进入**/usr/src**目录查看是否有，如果有就可以开始后续工作了。

## 二、实验要求和内容

### 1. 内容要求：

（1）添加一个系统调用，实现对指定进程的nice值的修改或读取功能，并返回进程最新的nice值及优先级。建议调用原型是int mysetniec(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，则表示读取nice的值；若为1，则表示修改nice的值。  
nicevalue：为指定的进程设置新的nice。  
prio，nice：指向进程的优先级和nice值。  
返回值：系统调用成功时返回0；失败时返回错误码EFAULT。  
（2）写一个简单的应用程序测试（1）中添加的系统调用。  
（3）若系统调用了Linux的内核函数，要求深入阅读相关的源码。

### 2. Linux系统调用的基本概念

实质是指调用内核函数，于内核态中运行，Linux中的用户通过执行一条访管指令“int $0x80”来调用系统调用，该指令会产生一个访管中断，从而让系统暂停当前的进程执行，而转去执行系统调用处理程序。通过用户态传入的系统调用号从系统调用表中找到相应的服务例程的入口并执行，完成后返回。  
（1）系统调用号与系统调用表：Linux内核中设置了一张系统调用表，用于关联系统调用号及其相对应的服务例程入口地址，定义在**./arch/x86/entry/syscalls/syscall\_64.tbl**文件中，每个系统调用占一个表项，一旦分配好就不可以有任何变更。  
（2）系统调用服务例程:每个系统调用都对应一个内核服务例程来实现系统调用的功能，其命名的格式都是以**"sys\_**开头。其代码通常放在**./kernel/sys.c**中，服务例程的原型声明则是放在**./include/linux/syscall.h**中。如sys\_open,通常格式是asmlinkage long sys\_open(int flag......)。其中的amslinkage是一个必需的限定词，用于通知编译器从堆栈中提取函数的参数，而不是从寄存器中。  
在sys.c中编程时，格式是SYSCALL\_DEFINE5(mysetnice, pid\_t, pid, int, flag, int, nicevalue, void \_\_user \*, prio, void \_\_user \*, nice)**N=5**代表参数的个数。  
（3）系统调用参数传递：在X86中，Linux通过6个寄存器来传入参数，其中一个eax是传递系统调用号，后面的5个传递参数。  
（4）系统调用参数验证

## 三、开始实验

### 1. 切换到root权限下，防止权限不够，导致出错。

$ sudo passwd root

[sudo] password for leslie:

Enter new UNIX password:

Retype new UNIX password:

passwd: password updated successfully

$ su root

Password:

1. 首先，你安装Linux系统时，它会让你设置一个你的用户名和用户密码，在这里我设置的用户名是leslie，即绿色字体leslie@tp50的前半部分，后半部分tp50是我的主机名。
2. sudo放在命令首，意思是当前指令以管理员权限运行。

(2)passwd是一条命令，用来修改用户密码，参数root是超级用户名，拥有系统最高的权限。passwd root的意思是修改超级用户的密码，在创建Ubuntu时，默认超级用户是没有密码的（也可能是一个随机数之类的我不记得了），用这条命令重新设定一个密码。

(3)su是一条命令，用来切换当前用户，在第一章你会认识到，Linux是一个多用户多任务的操作系统。参数root是用户名，指示切换到的用户，在这里su root意为切换到root用户，当参数缺省时，默认切换到超级用户。

(4)你会发现，它在提示输入密码时，虽然键盘已经输入了密码，但是终端没有任何响应，不要担心，这正是Unix和Linux的特点，为了确保安全，在输入密码时不显示输入的内容，在输入密码后，直接按下回车就好了。

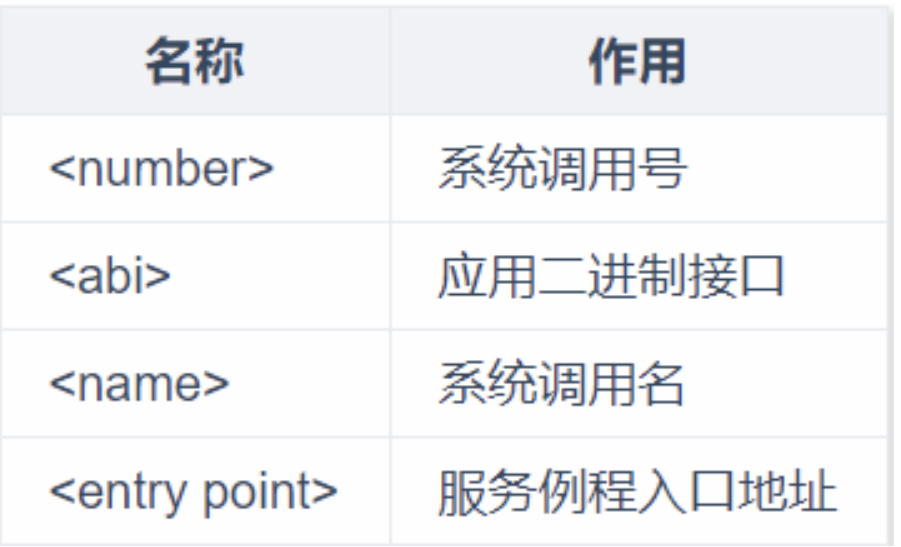
### 2. 分配系统调用号，修改系统调用表

**（1）查看系统调用表，并修改**

gedit /usr/src/linux-4.19.25/arch/x86/entry/syscalls/syscall\_64.tbl

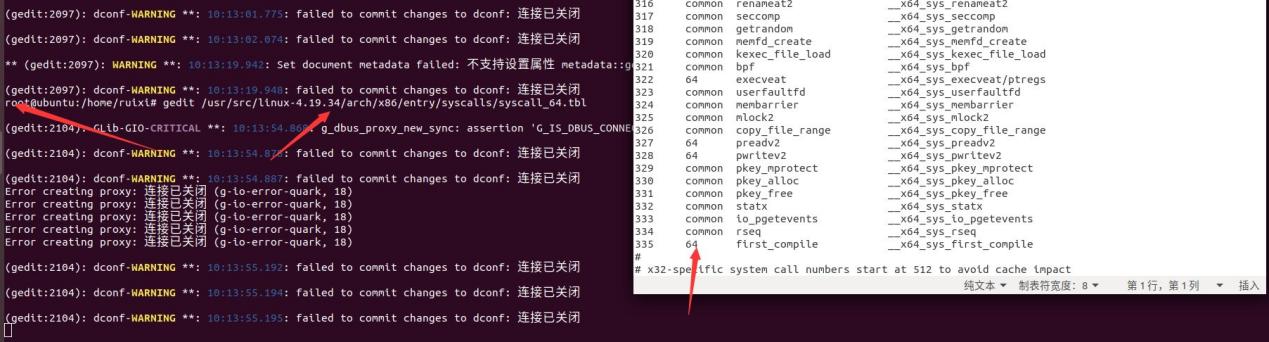
你只需要将**linux-4.19.25**换成你自己下载好的版本即可。

你会看见这个格式



应用二进制接口分为三种：64、x32和common，即三种不同的调用约定，这里不需考虑太多，三种任意选择一种即可，按照上述格式编写新的系统调用表表项如下：

335 64 first\_compile \_\_x64\_sys\_first\_compile



**（2）声明系统调用服务例程**

查看系统调用头文件

gedit /usr/src/linux-4.19.25/include/linux/syscalls.h

compat\_sys\_rt\_tgsigqueueinfo ksys\_personality

同样将linux-4.19.25换成你自己的版本即可。

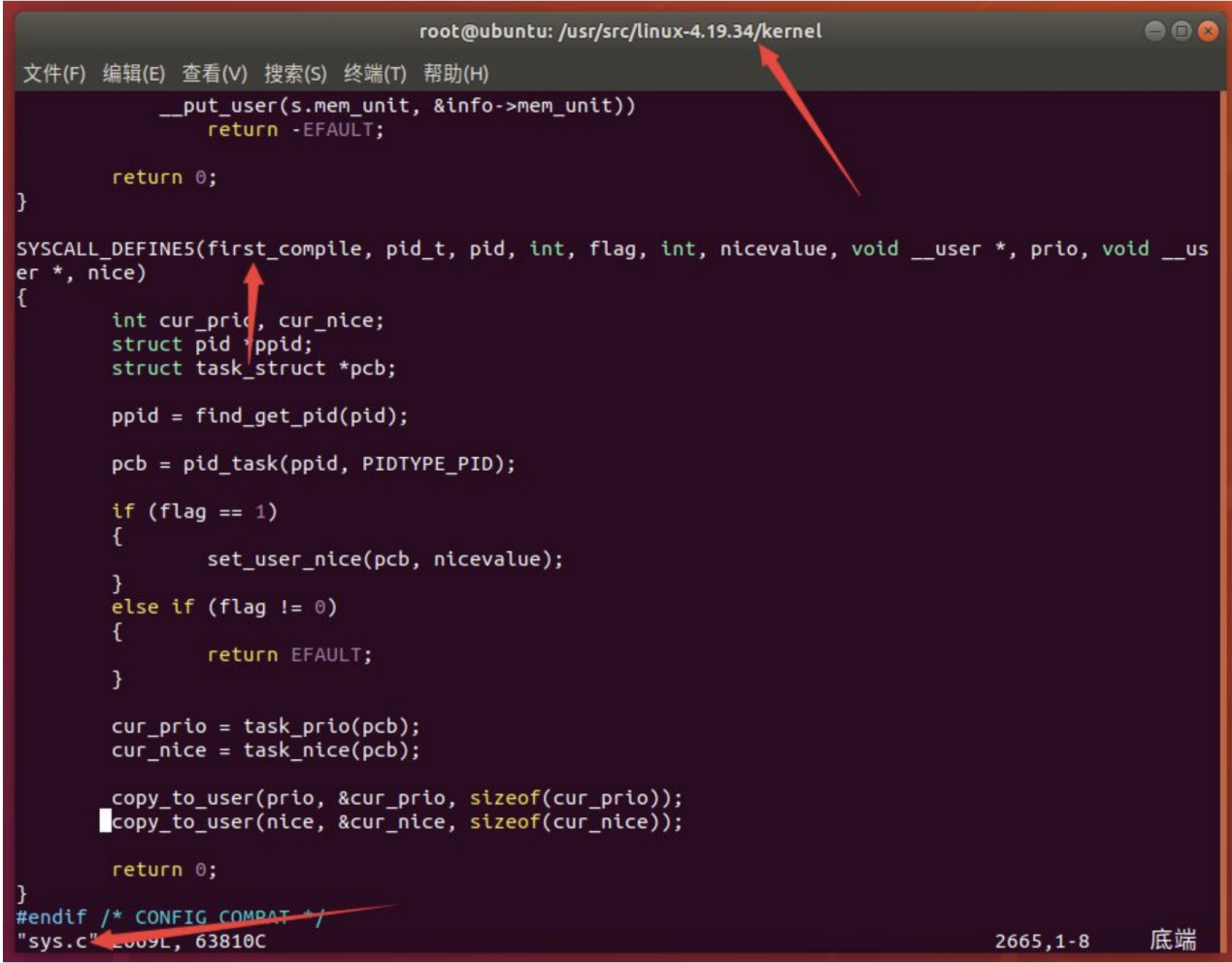


**（3）实现自己的系统调用服务例程**

首先进入解压后的文件目录，就是开始解压放入的目录，如下图：

$cd /usr/src/linux-4.19.34(换成自己的版本即可)/kernel

vim sys.c



函数说明：  
这一步与上一步的关系，就是C语言中头文件与实现文件的关系，上一步我们对函数进行了声明，这里给函数一个具体的实现。

首先要明确，我们要实现一个什么样的功能，根据内容要求可知，这个系统调用需要具备对指定进程的nice值的修改及读取的功能，同时返回进程最新的nice值及优先级prio。

把功能分拆成一个一个小块，我们需要做到的有以下几点：

根据进程号pid找到相应的进程控制块PCB（因为进程控制块中记录了用于描述进程情况及控制进程运行所需要的全部信息，nice值和优先级正是其中的一部分）；

根据PCB读取它的nice值和优先级prio；

根据PCB对相应进程的nice值进行修改；

将得到的nice值和优先级prio进行返回。

SYSCALL\_DEFINE5(first\_compile, pid\_t, pid, int, flag, int, nicevalue, void \_\_user \*, prio, void \_\_user \*, nice)

{

int cur\_prio, cur\_nice;

struct pid \*ppid;

struct task\_struct \*pcb;

ppid = find\_get\_pid(pid);

pcb = pid\_task(ppid, PIDTYPE\_PID);

if (flag == 1)

{

set\_user\_nice(pcb, nicevalue);

}

else if (flag != 0)

{

return EFAULT;

}

cur\_prio = task\_prio(pcb);

cur\_nice = task\_nice(pcb);

copy\_to\_user(prio, &cur\_prio, sizeof(cur\_prio));

copy\_to\_user(nice, &cur\_nice, sizeof(cur\_nice));

return 0;

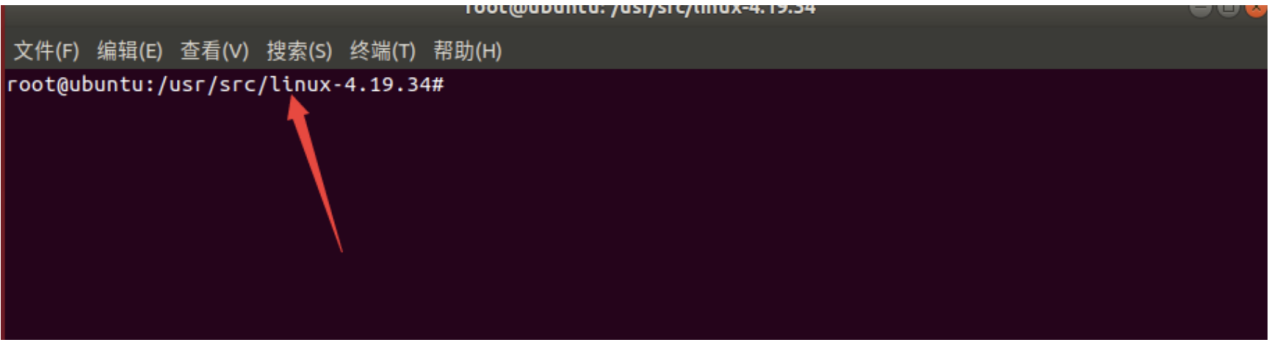
}

**（4）开始编译内核**

首先，用下面这条命令查漏补缺，很有用处，用来它，我编译是一次通过的，没有遇见什么其他麻烦。

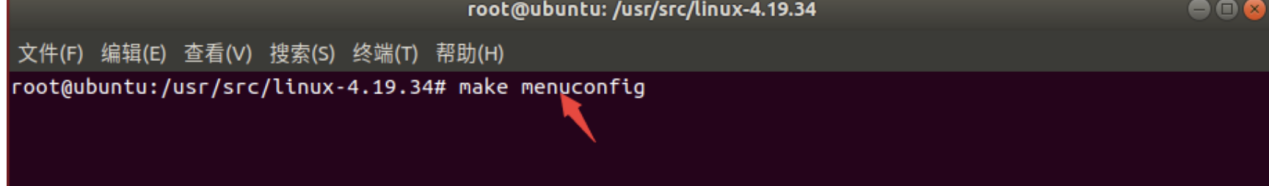
sudo apt-get install libncurses5-dev make openssl libssl-dev bison flex

然后定位到，源码在的目录，也就是解压后放的目录。

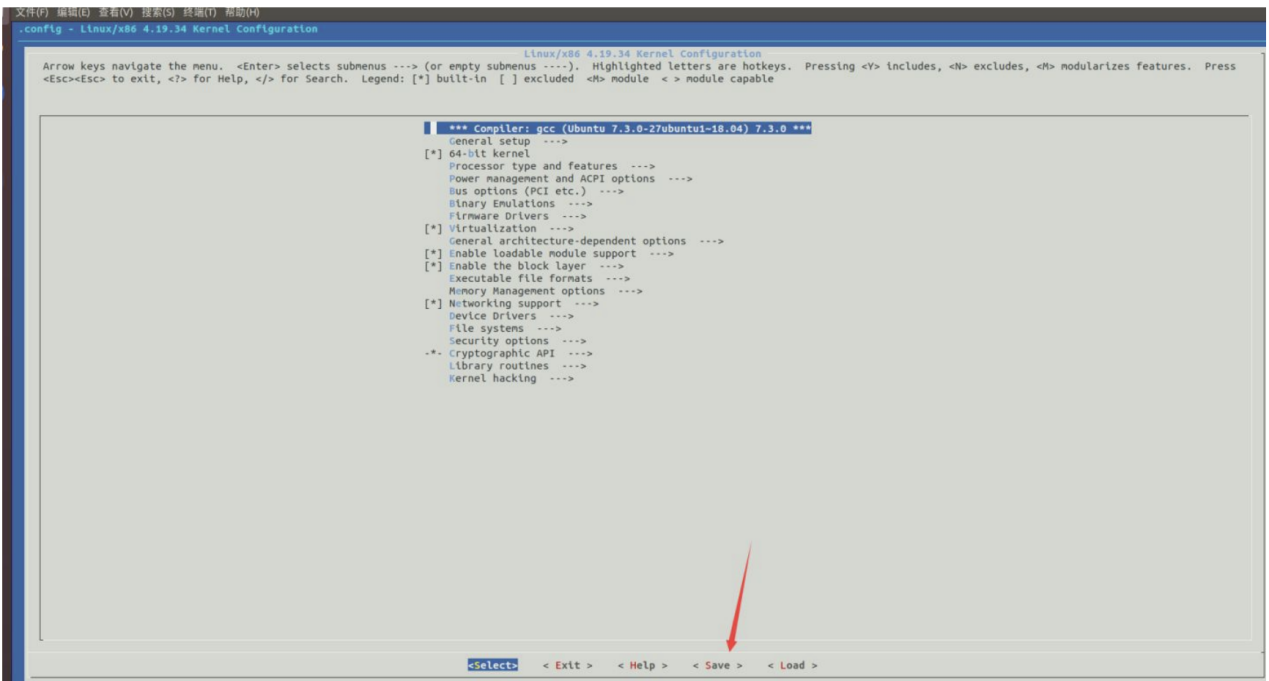


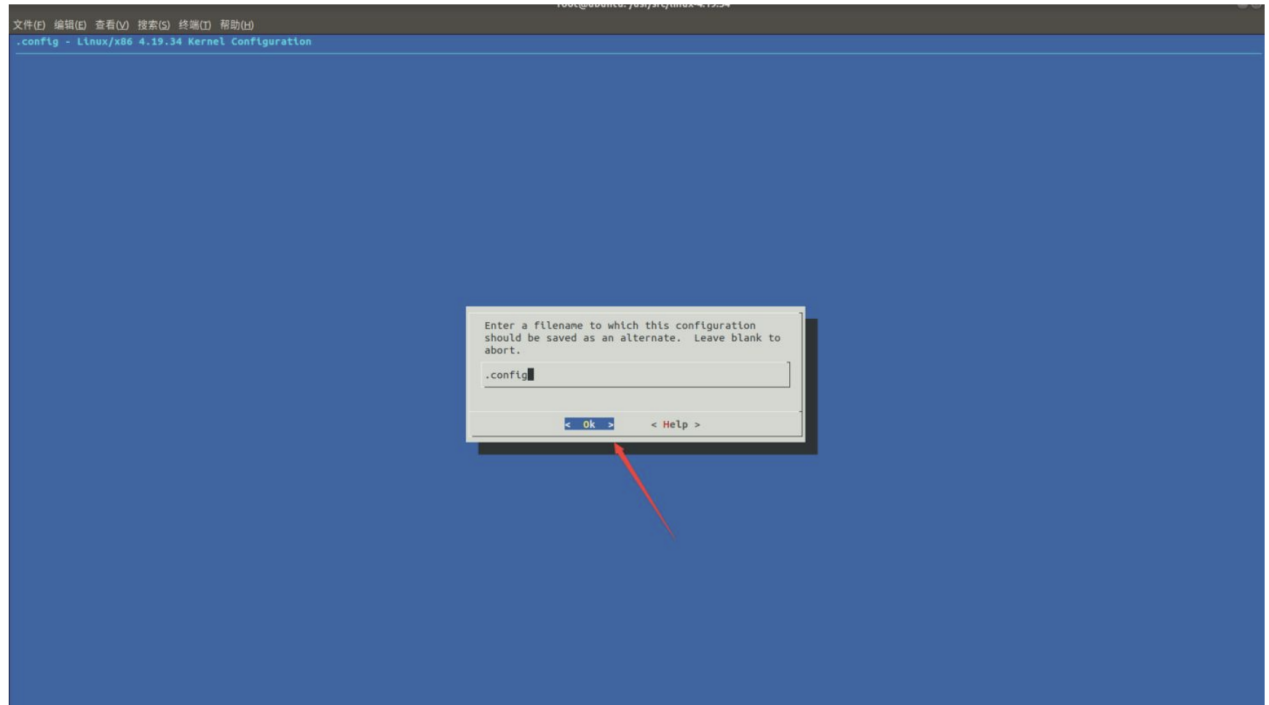
然后运行命令

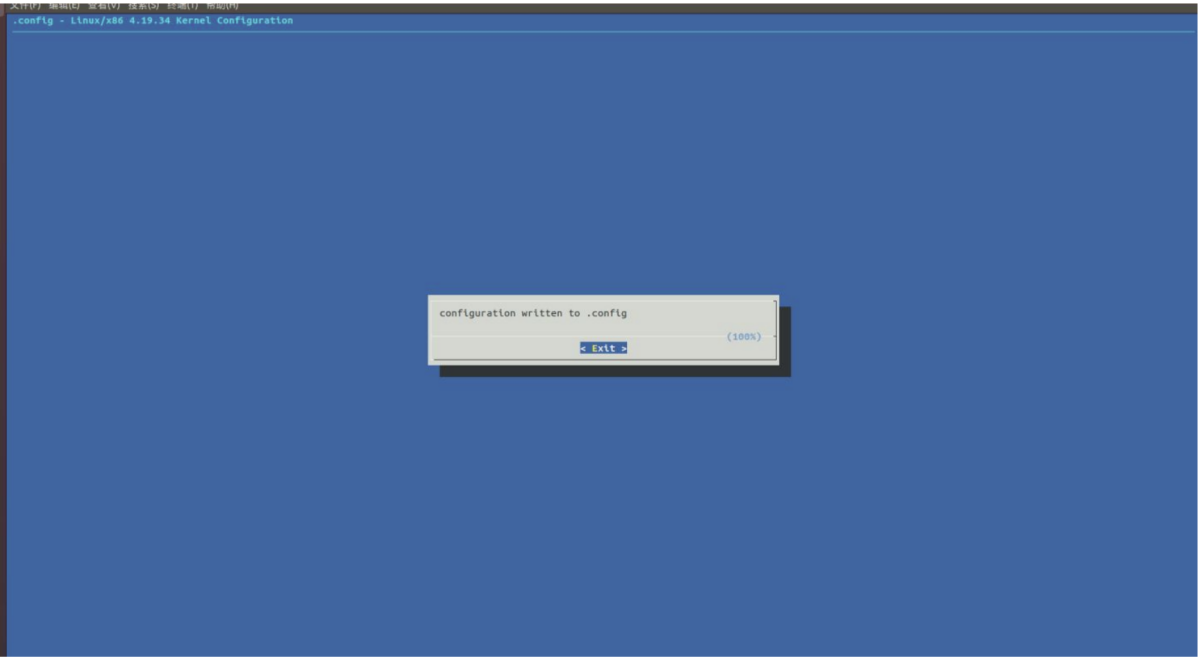
make menuconfig

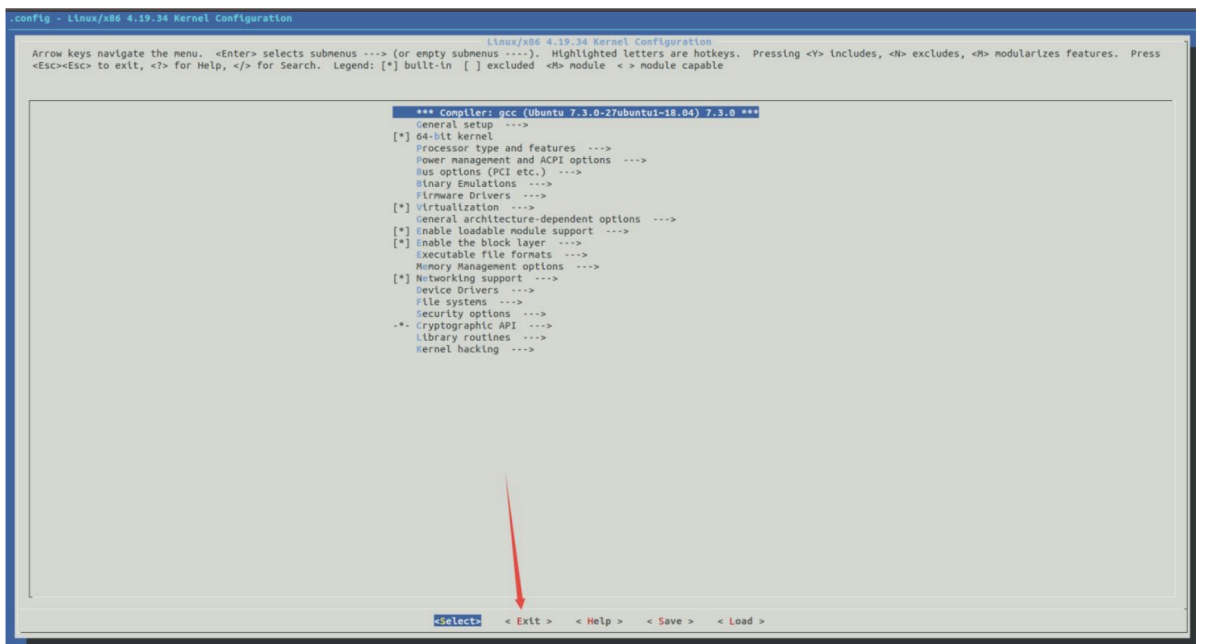


然后会出现以下界面，根据下面的图片所指的按钮来，通过左右键来确定光标停在选的按钮上，enter键是确定键









**准备工作做好后，开始编译，耗时最长**

sudo make -j4 2> error.log

-j4表示使用四线程进行编译，这个过程大概持续一个小时，后面的重定向将错误信息输出到了error.log这个文件里面，方便我们之后进行错误排查，不至于一两个小时坐在电脑面前盯着信息输出生怕出现一个错误而自己错过了，之后修改只能靠两眼排查，相信我，那不是一种好的体验。  
开始等待吧，结束后就可以安装内核了。