**编译内核**

导言：

**核心文件：**

核心 (kernel)是整个操作系统的最底层， 他负责了整个硬件的驱动，以及提供各种系统所需的核心功能，包括防火墙机制、是否支持 LVM 或 Quota 等文件系统等等，这些都是核心所负责的。在开机流程中，**MBR内的loader加载核心文件来驱动整个系统的硬件**。

计算机真正在工作的东西其实是硬件， 例如数值运算要使用到 CPU、数据储存要使用到硬盘、图形显示会用到显示适配器、音乐发声要有音效芯片、 连接 Internet 可能需要网络卡等等。**如何控制这些硬件，那这就是核心的工作。**

**你想要让计算机进行的工作，都必须要核心有支持才可以！**这个标准不论在 Windows 或 Linux 这几个操作系 统上都相同！如果有一个人开发出来一个全新的硬件，目前的核心不论 Windows 或 Linux 都不支持，那么不论你用什么系统，这个硬件都是英雄无用武之地啦！

那么核心到底是什么啊？**其实核心就是系统上面的一个文件(透过源码编译而成的文件)而已， 这个文件包含了驱动主机各项硬件的侦测程序与驱动模块**。

当系统读完 BIOS 并加载 MBR 内的开机管理程序后，就能够加载核心到内存当中。然后核心开始侦测硬件，挂载根目录并取得核心模块来驱动所有的硬件，之后呼叫 systemd 就能够依序启动所有系统所需要的服务了！

**这个核心文件通常被放置成 /boot/vmlinuz-xxx**，不过也不见得，因为一部主机上面可以拥有多个核心文件，只是开机的时候仅能选择一个来加载而已。甚至我们也可以在一个 distribution上面放置多个核心，然后以这些核心来做成多重引导呢！

**核心模块：**

将一些不常用的类似驱动程序的咚咚独立出核心，编译成为模块，然后，核心可以在系统正常运作的过程当中加载这个模块到核心的支持。如此一来， 我在不需要更动核心的前提之下，只要编译出适当的核心模块，并且加载他，我的Linux就可以使用这个硬件啦！

**我的模块放在哪里呀？ 放在/lib/modules/$(uname -r)/kernel/当中。**

**驱动程序：**

硬件的驱动程序可以编译成为核心模块，所以可以在不改变核心的前提下驱动你的新硬件，不能说Linux的内核支持度不足，因为在这个驱动程序开发的工作上面来说，应该是属于硬件发展厂商的问题。若硬件开发商发展Linux上的驱动程序，那该硬件便可以在Linux上面运行。

**更新核心的目的：**

核心的编译重点在于你要你的 Linux 作什么？如果没有必要的工作，就干脆不要加在你的核心当中了！这样才能让你的 Linux 跑得更稳、更顺畅！ 这也是为什么我们要编译核心的最主要原因了！

新功能的需求：新核心中含有你需要的功能，则需要你重新编译核心。

原本核心太过臃肿：若需要稳定性高，则对于原本选择了很多功能的核心，你可以重新编译来取消该功能。

与硬件搭配的稳定性：由于原本核心大多针对Intel的CPU，若你的CPU为AMD的系统，那就需要重新编译核心来让系统取得正确的模块。

其他需求(如嵌入式系统)：特殊要求时就得自行设计核心了，像商业的软件包系统，需要小而美的操作系统，此时就需要核心更加简洁有力。

一、下载内核源码包

以linux.5.8.12.stable为例，源码包获取官网如下：

<https://www.kernel.org/>

点击tarball下载



二、挑选核心功能以及编译、安装

1. 下载完内核源码包后进行如下操作：

在家目录使用$ mkdir workspace 新建工作目录，并把下载的源码包linux-5.8.12.tar.xz 放到该目录下。

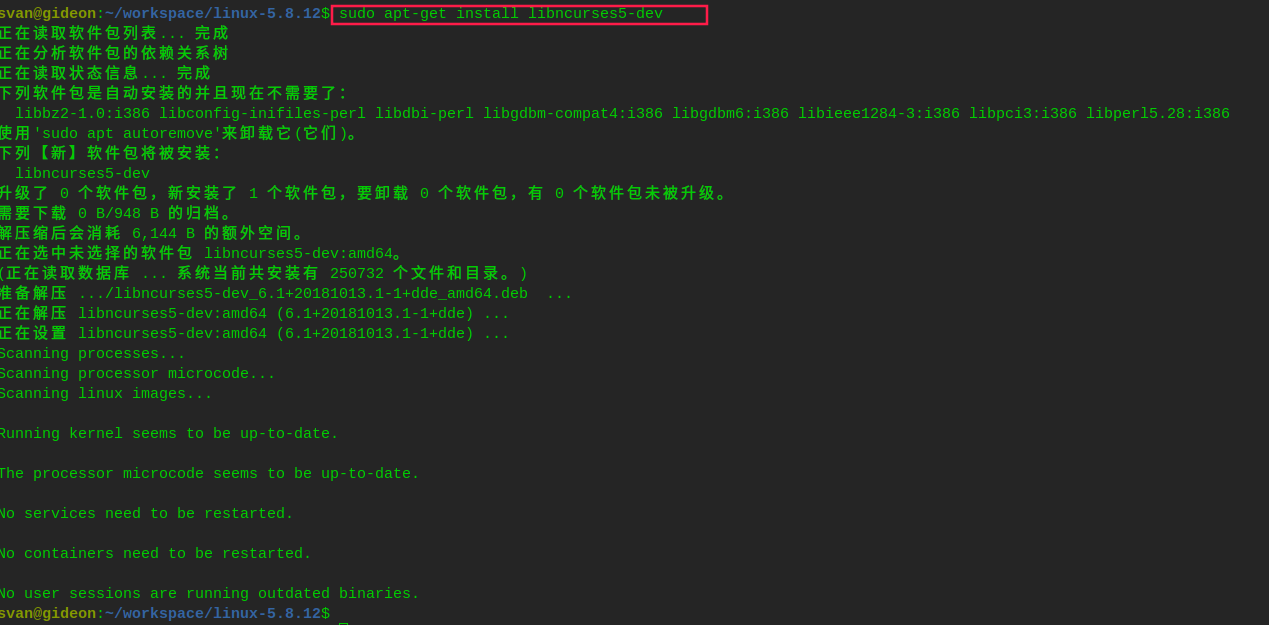
(注意：需要约25G的空间，刚开始由于没有考虑到空间的大小导致后面分出了一个区并且把工作移到了根目录下进行)

1. 使用$ tar Jx -f linux-5.8.12.tar.xz 进行解压(此时稍微等一会)
2. 使用$ make mrproper 处理掉源码中我们不清楚有没有保留的目标文件以及配置文件，另外此动作会将你以前进行过的核心功能选择文件删除掉，首次编译内核前下达此命令。
3. 使用$ make clean 仅删除编译过程中产生类似目标文件的中间文件，不会删除配置文件。
4. 使用$ make menuconfig 可以显示类似图形接口的方式来选择核心功能模块，核心功能挑选完毕后，会在当前目录当前目录(/home/gideon/workspace/linux-5.8.12/)下生成一个名为.config的隐藏文件。

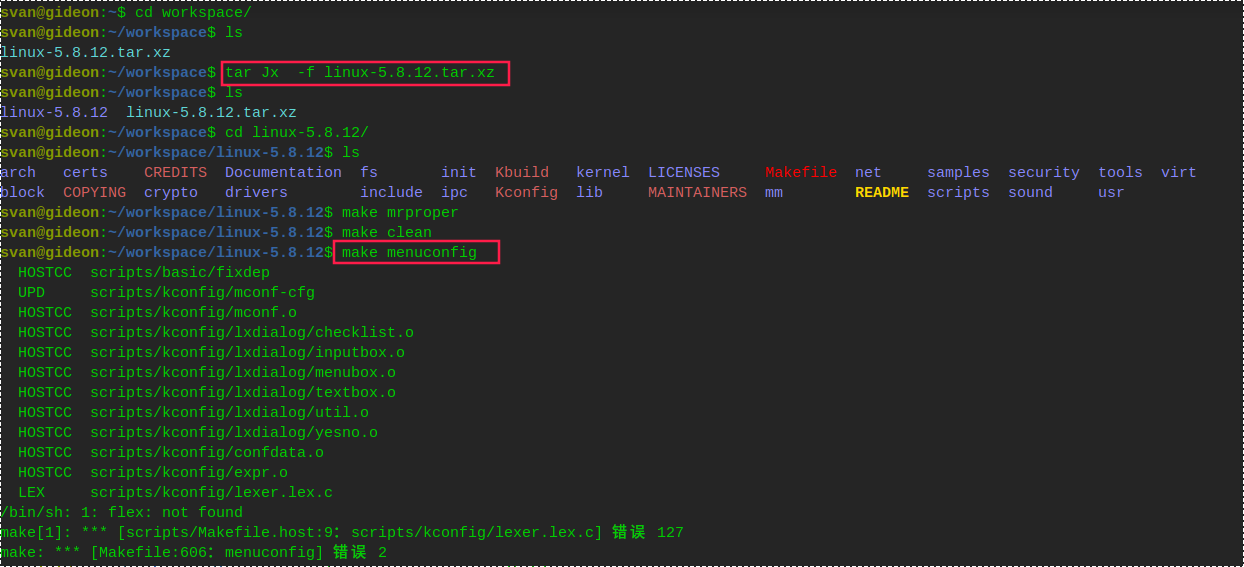
(注意：不要更改这个隐藏文件的文件名，否则无法编译核心功能，后面你会看我在挑选核心功能并保存时擅自更改了文件名，最后又手动改回来，并且(好像忘记截屏了所以没有体现出来)删掉.config\_van.old)

1. make menuconfig 是基于ncurse库编制的图形界面工具，若没有ncurse，则自行安装：

$ sudo apt-get install libncurses5-dev



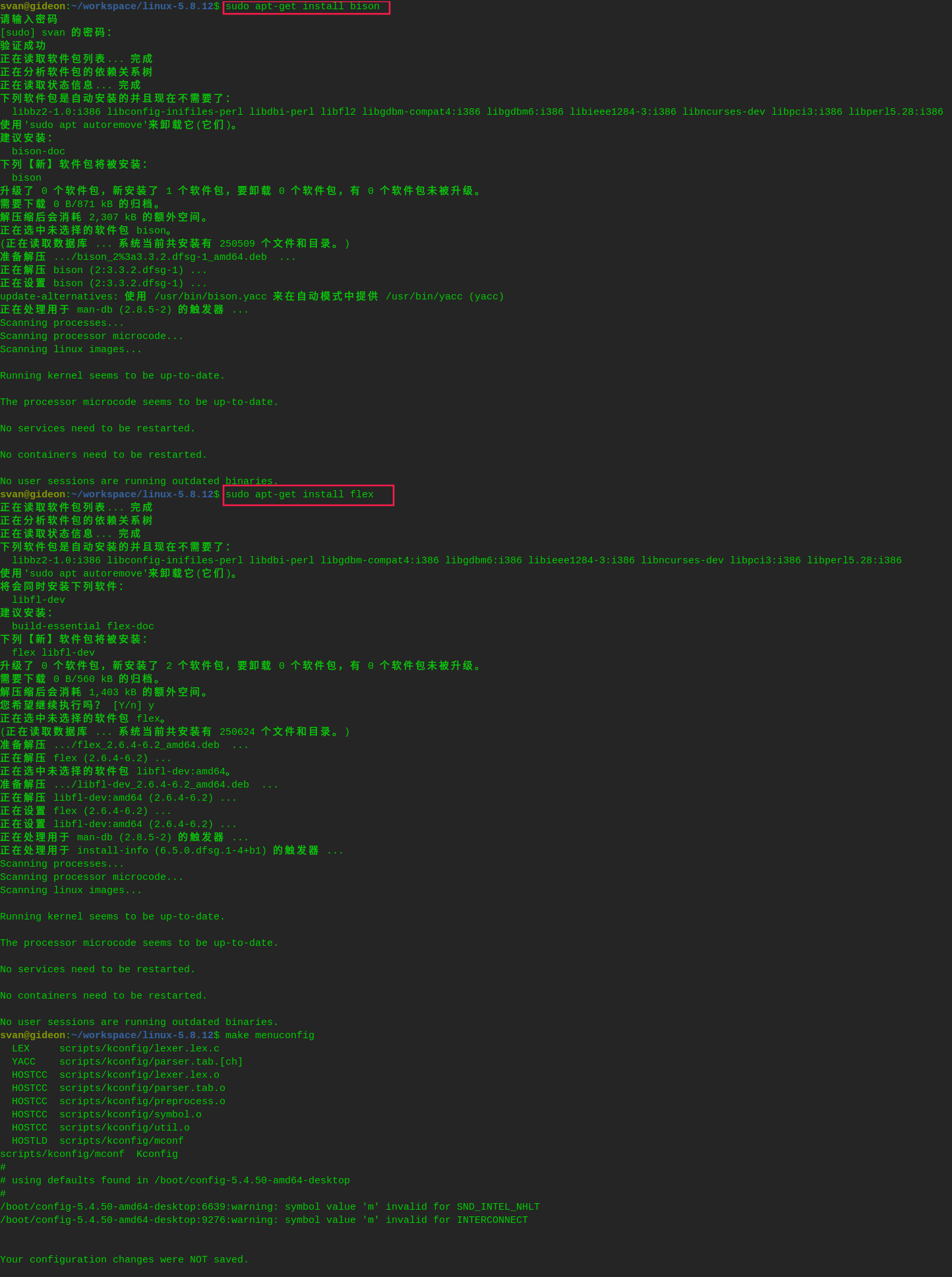
此时出现如下错误，即flex找不到。



1. 使用$ sudo apt-get install bison

与$ sudo apt-get install flex

解决上述问题



成功执行make menuconfig 后会出现如下界面，并且开始配置内核，其中：

[\*] <\*> 代表该功能编译到内核

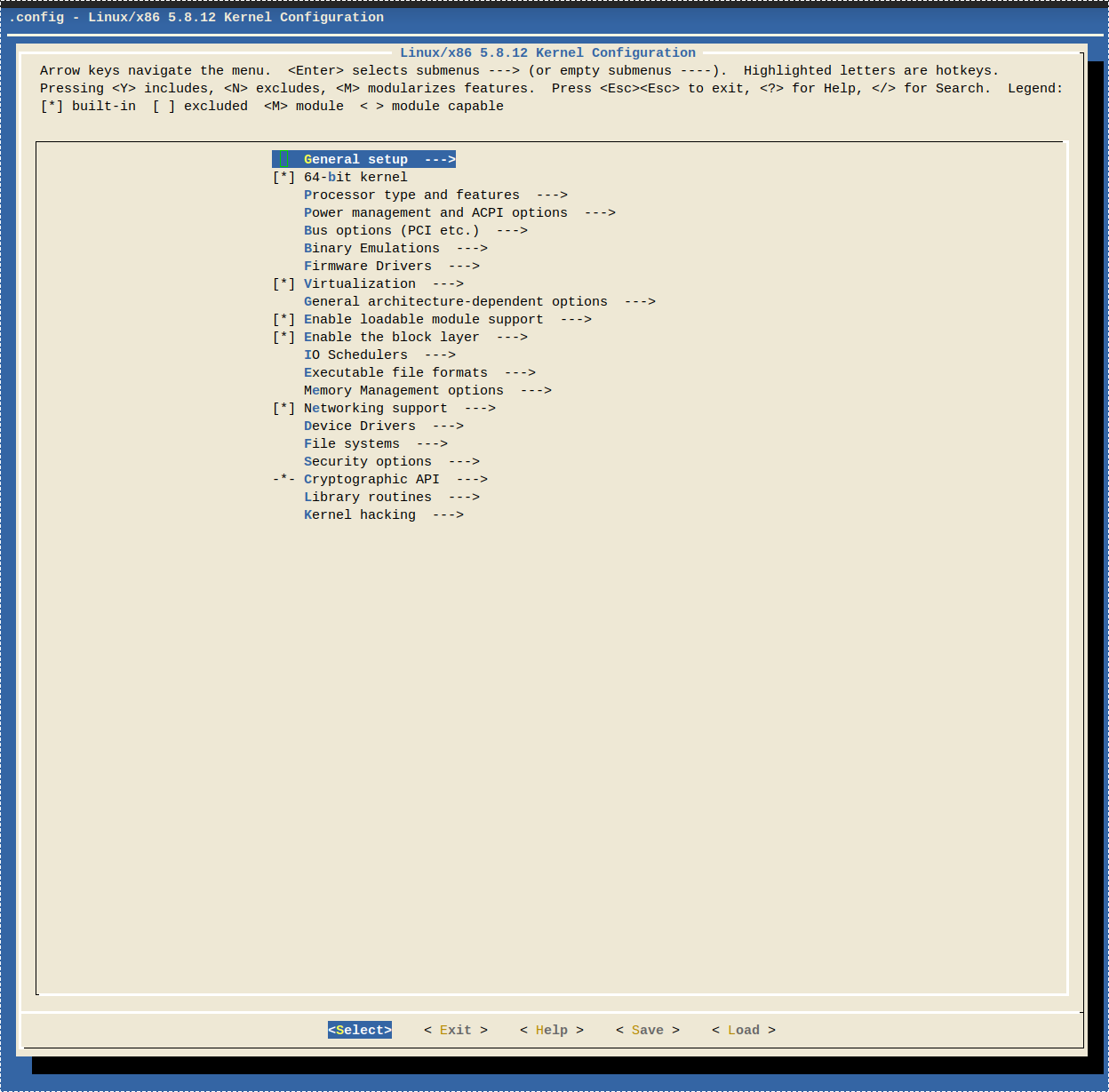
---> 代表有需要来设定的子项

1. 代表该功能编译为模块

-\*- 代表

[ ] < > 代表该功能可以选择，可以使用空格键来选择，是编译成模块还是编译到内核

使用上下键来挑选功能，左右键来选择保存、挑选、帮助等，选中后使用回车键进入，使用空格键来选择功能是否编译到内核。

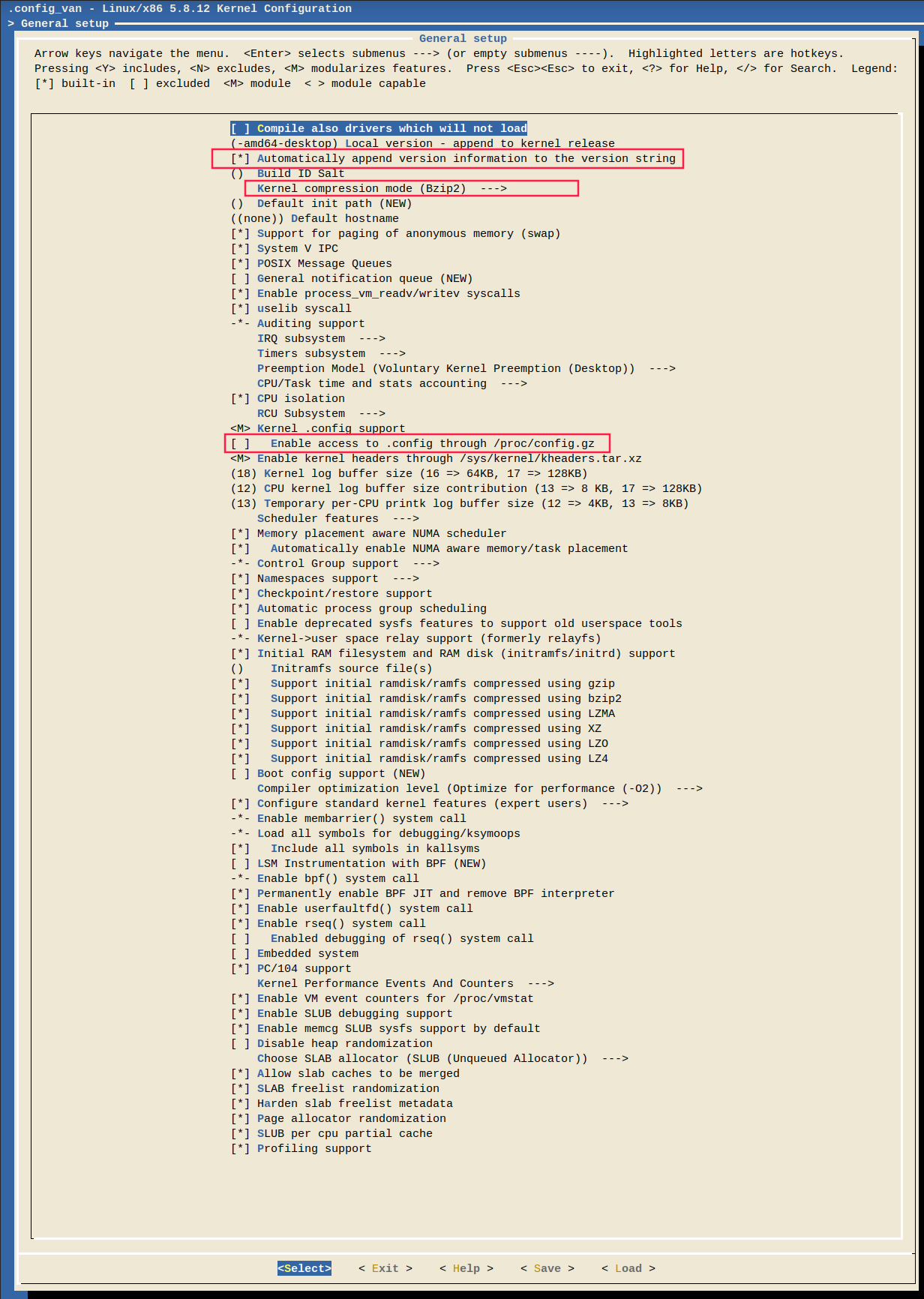


先来看general setup --->，进入后如下界面：



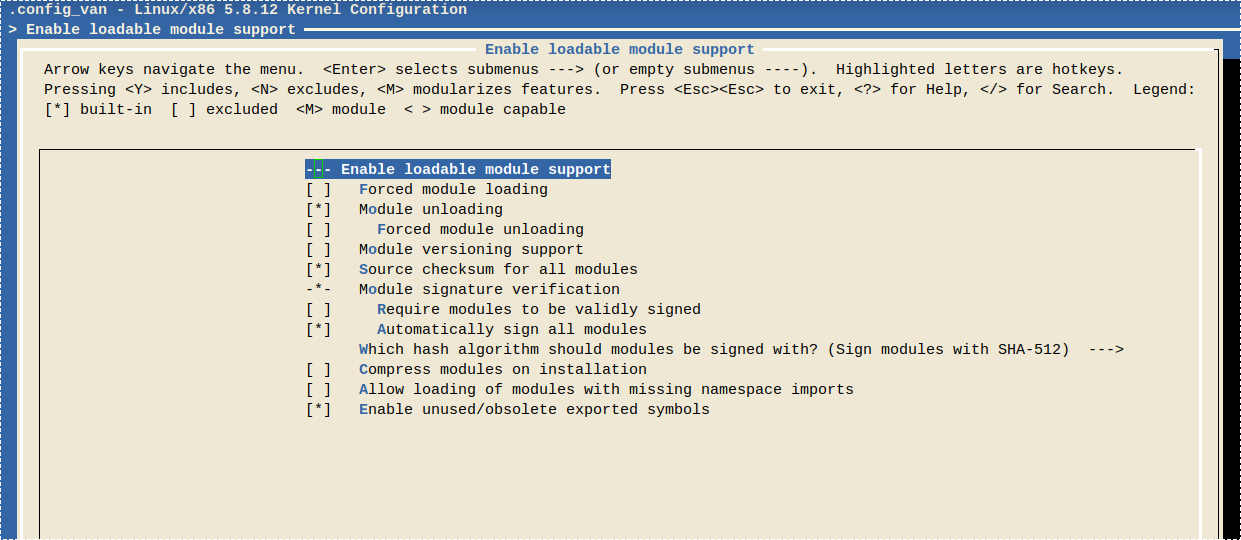
上图是未修改过的，下面图片中红色框框是改动的部分:

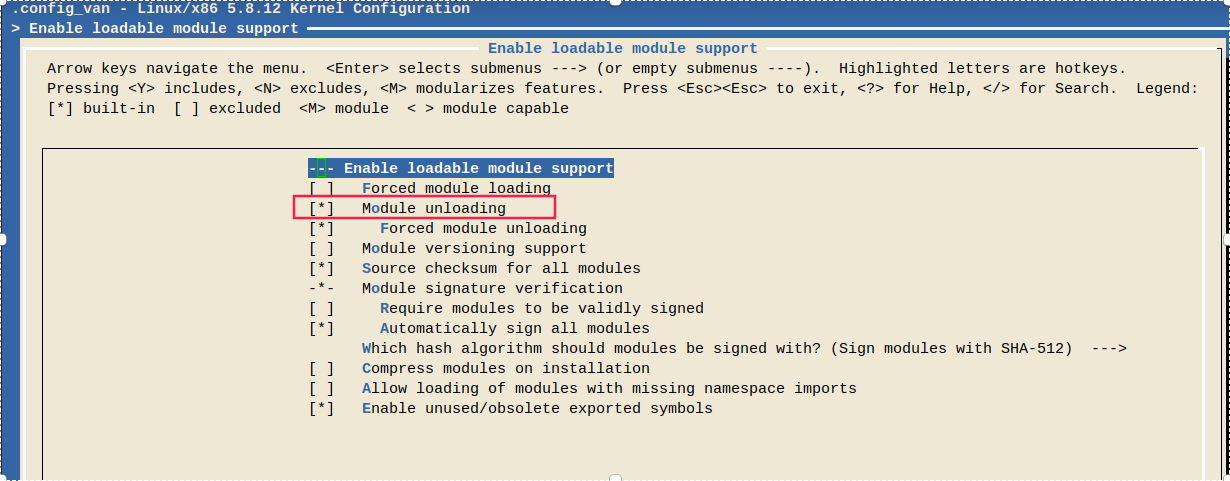
1. 自动在版本字符串后追加版本信息，自动探测你的内核并且生成相应的版本。
2. 内核压缩模式由Gzip改为Bzip2
3. 此选项功能是可以通过/proc/config.gz访问当前内核的.config，但是我没选上。



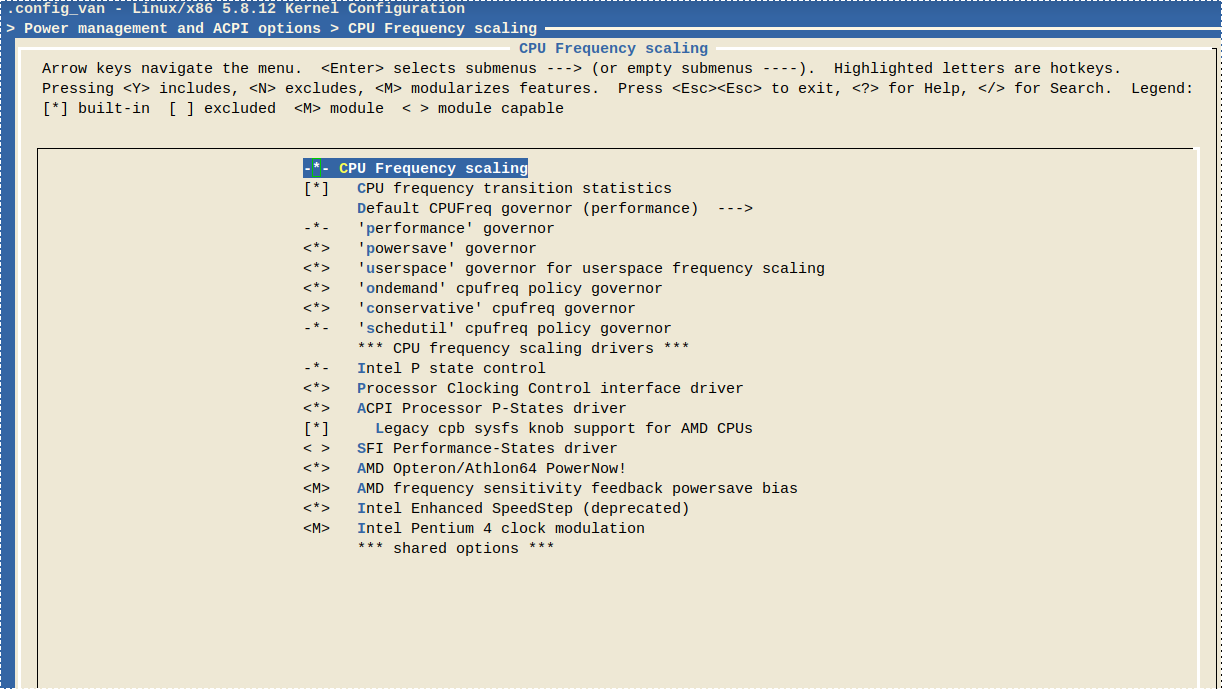
这个是enable loadable module support ，即加载模块的支持。

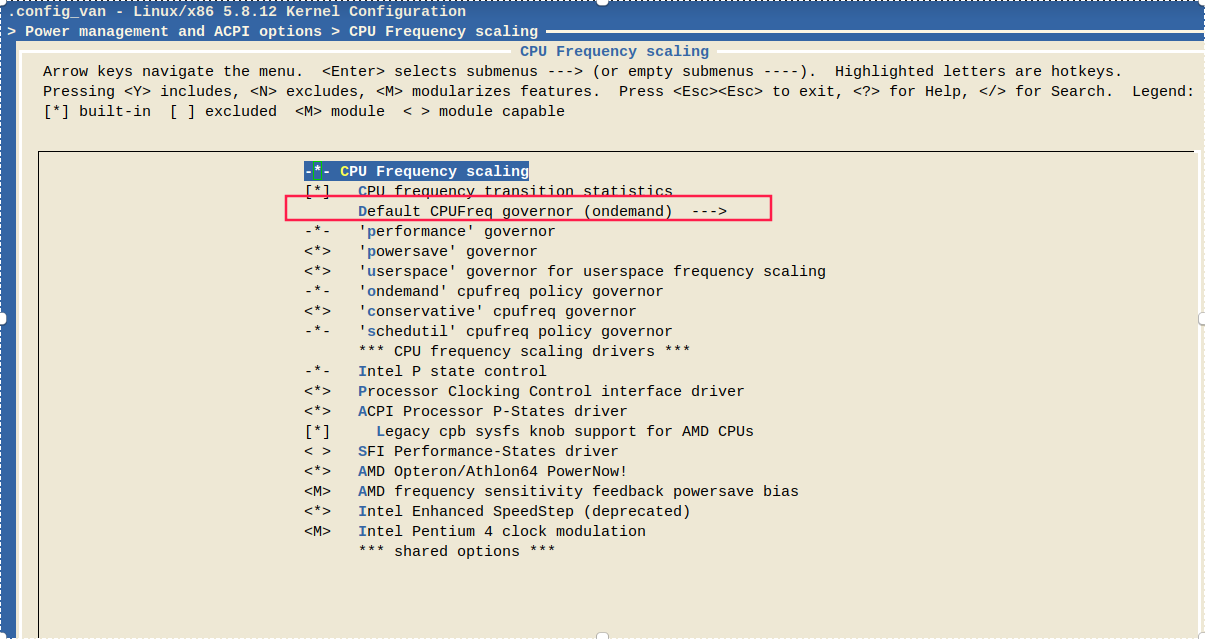
这个红色的框框画错了，本意是画的下面的一行，即支持卸载模块的功能。



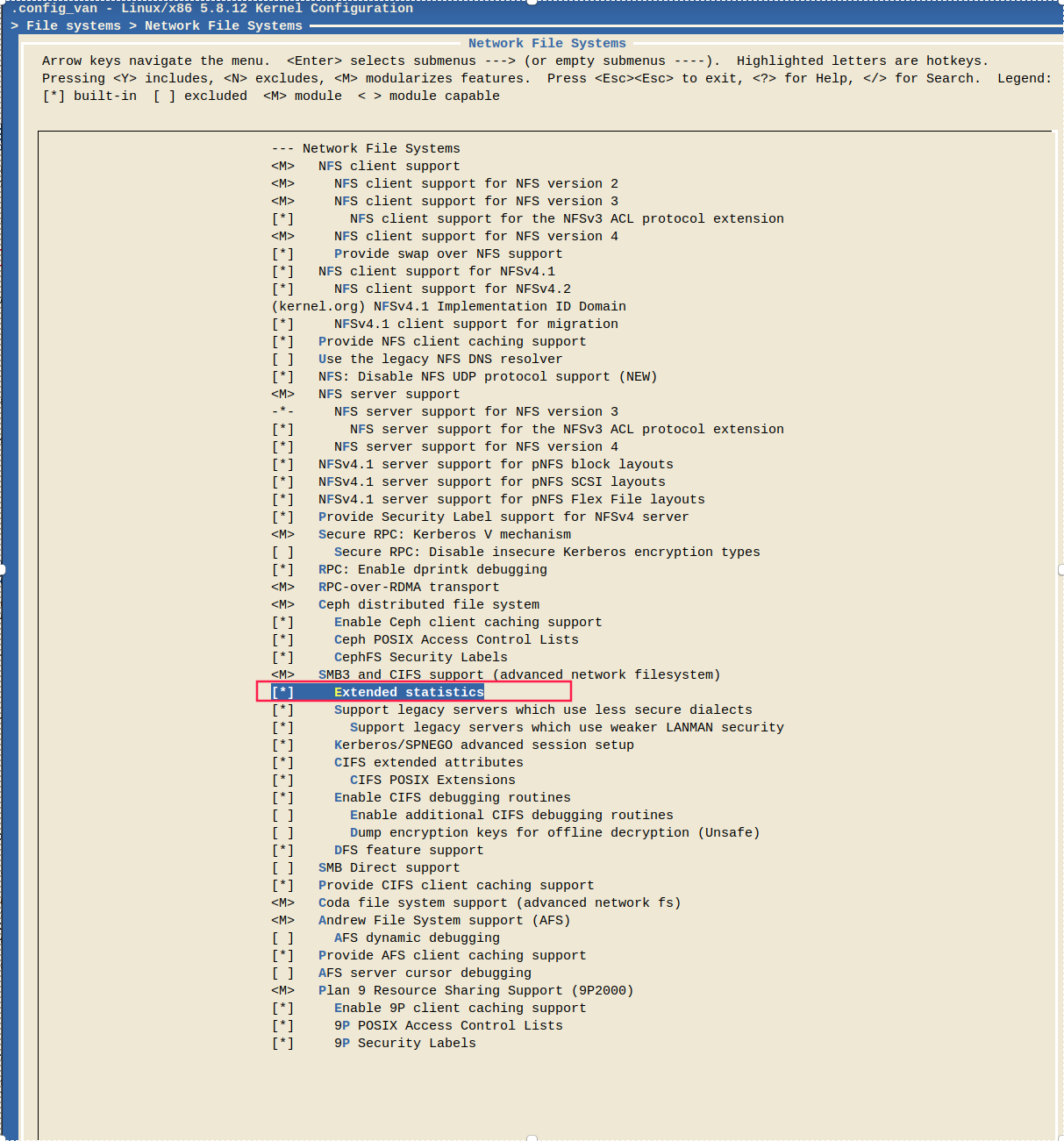


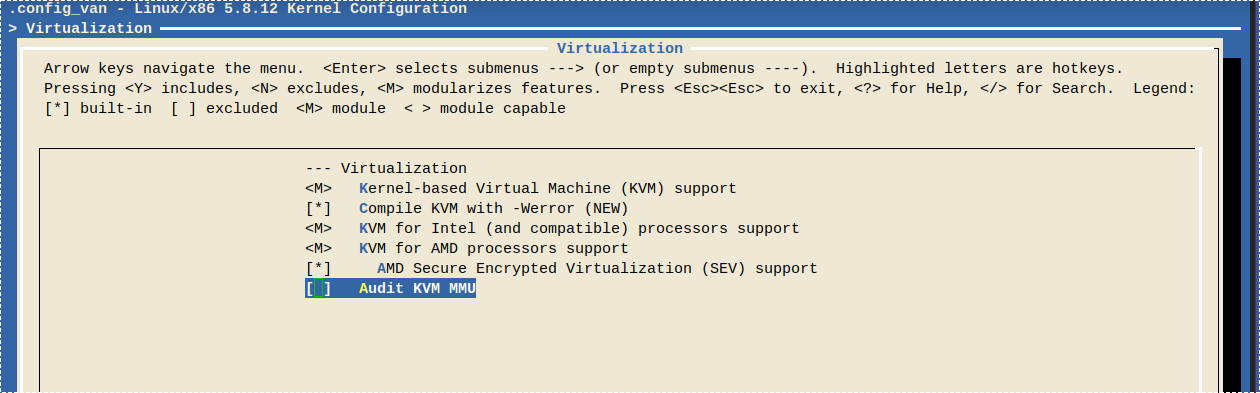
这个是power management and ACPI options 中的CPU Frequency scaling 功能。

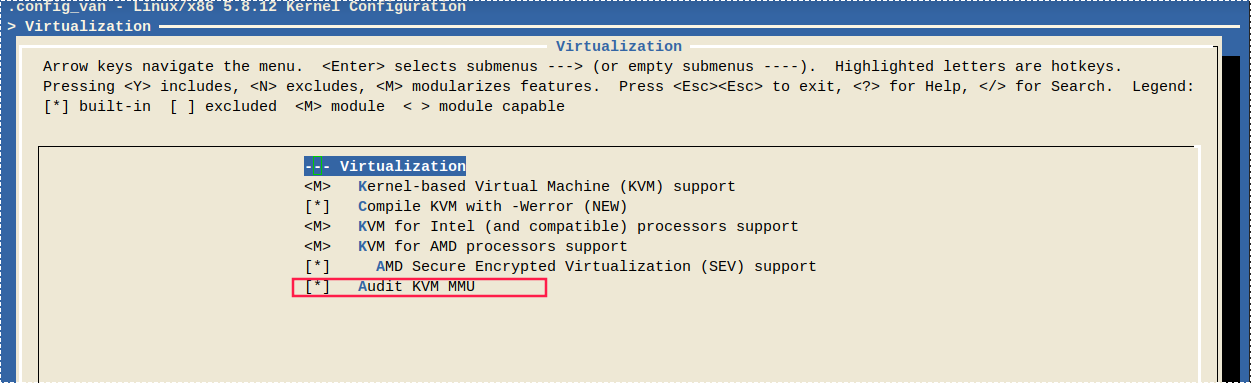






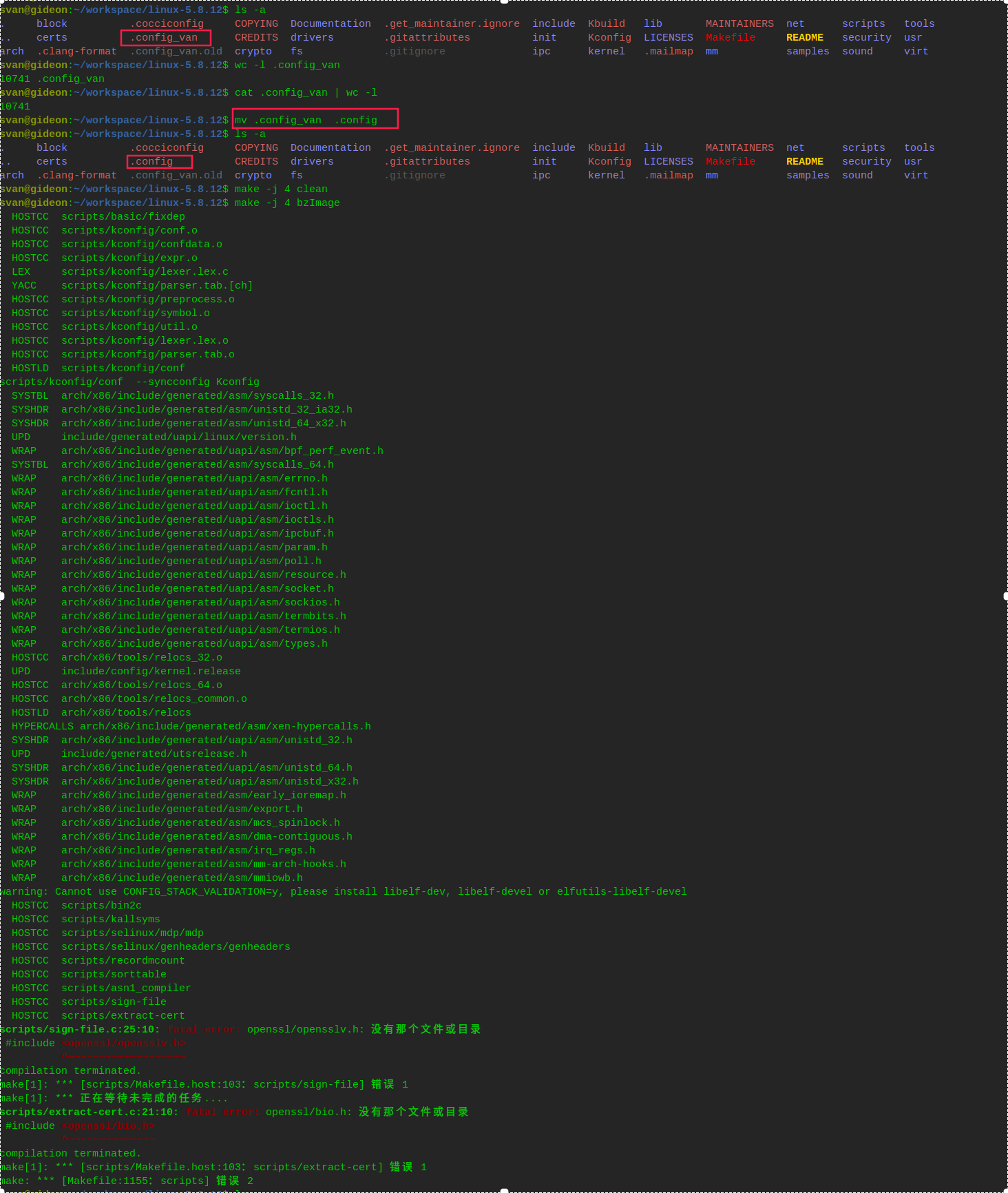




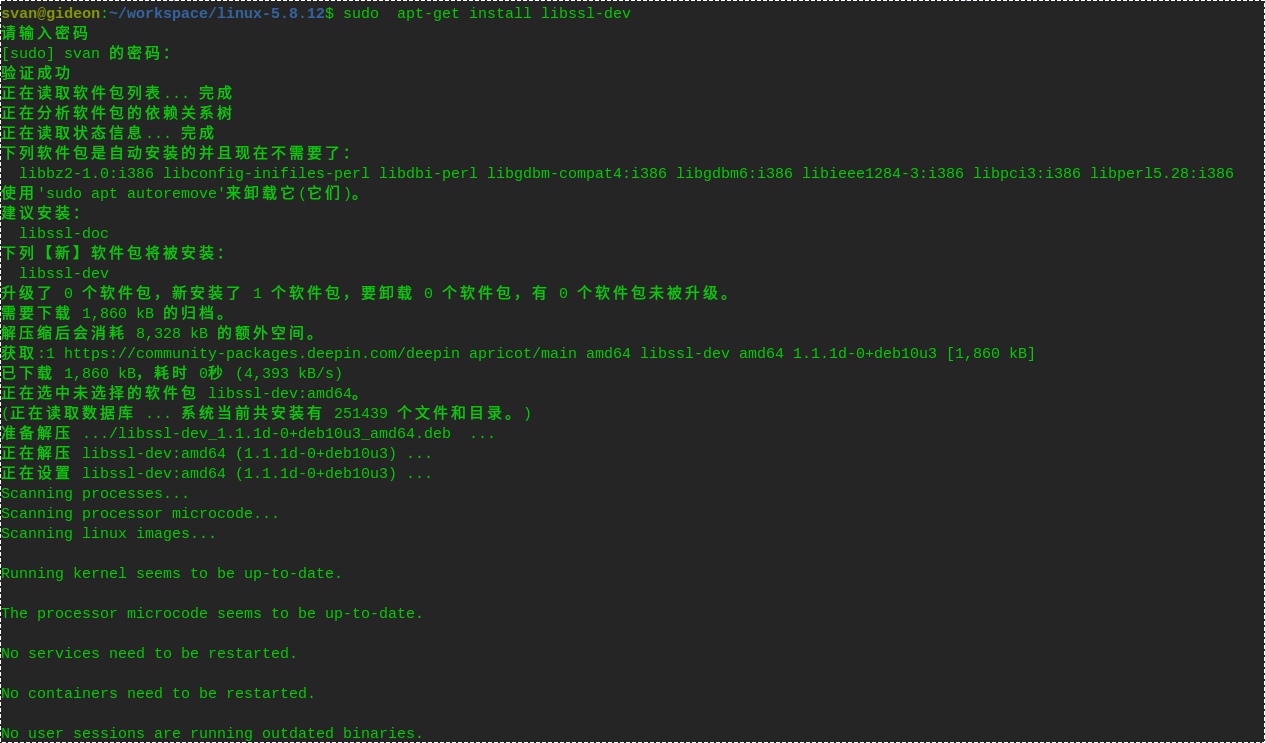


挑选核心功能后，开始核心、核心模块的编译与安装。

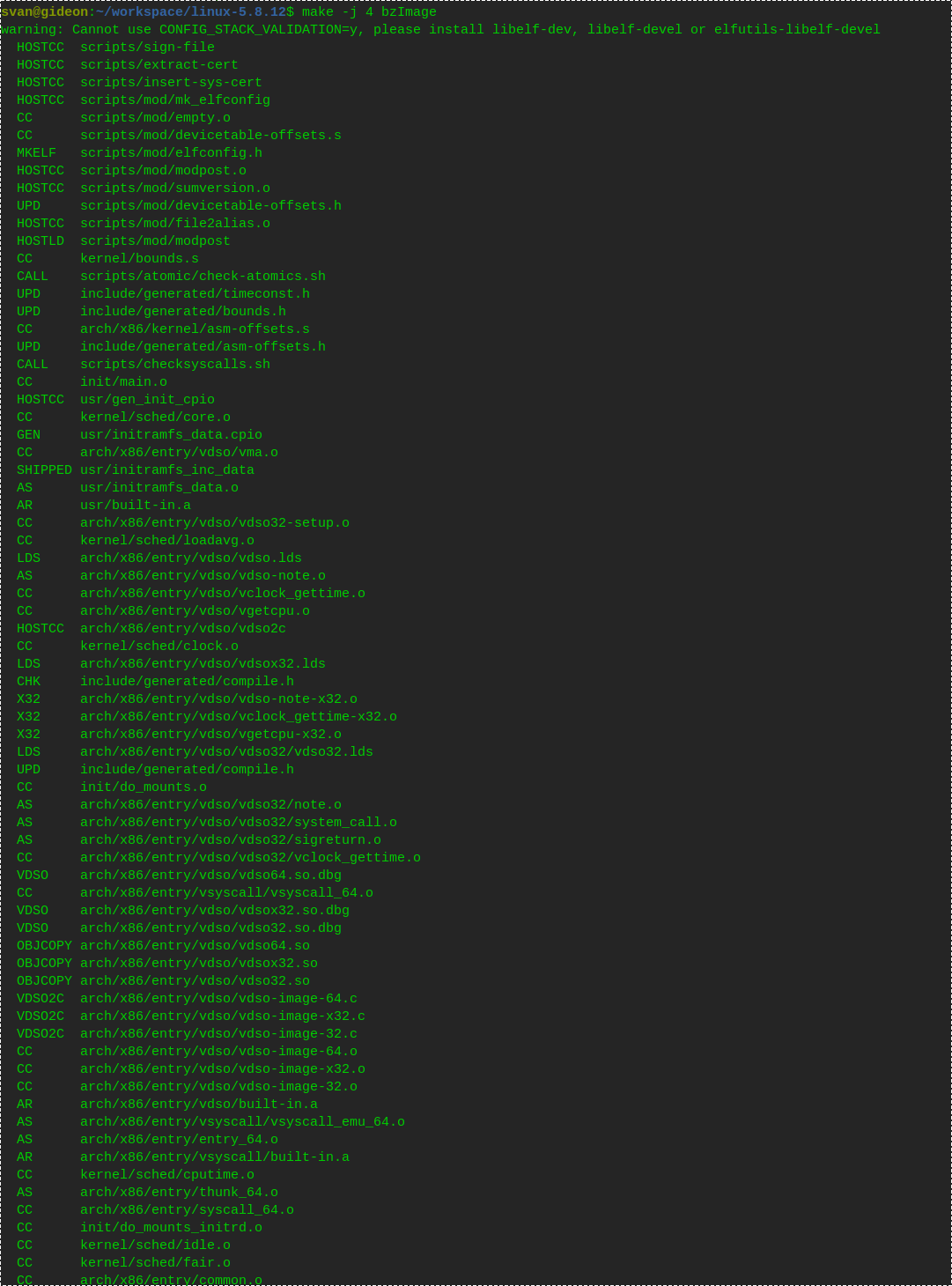
1. 使用$ make -j 4 clean 清除暂存档，使用$ make -j 4 bzImage 编译核心，此时报错。

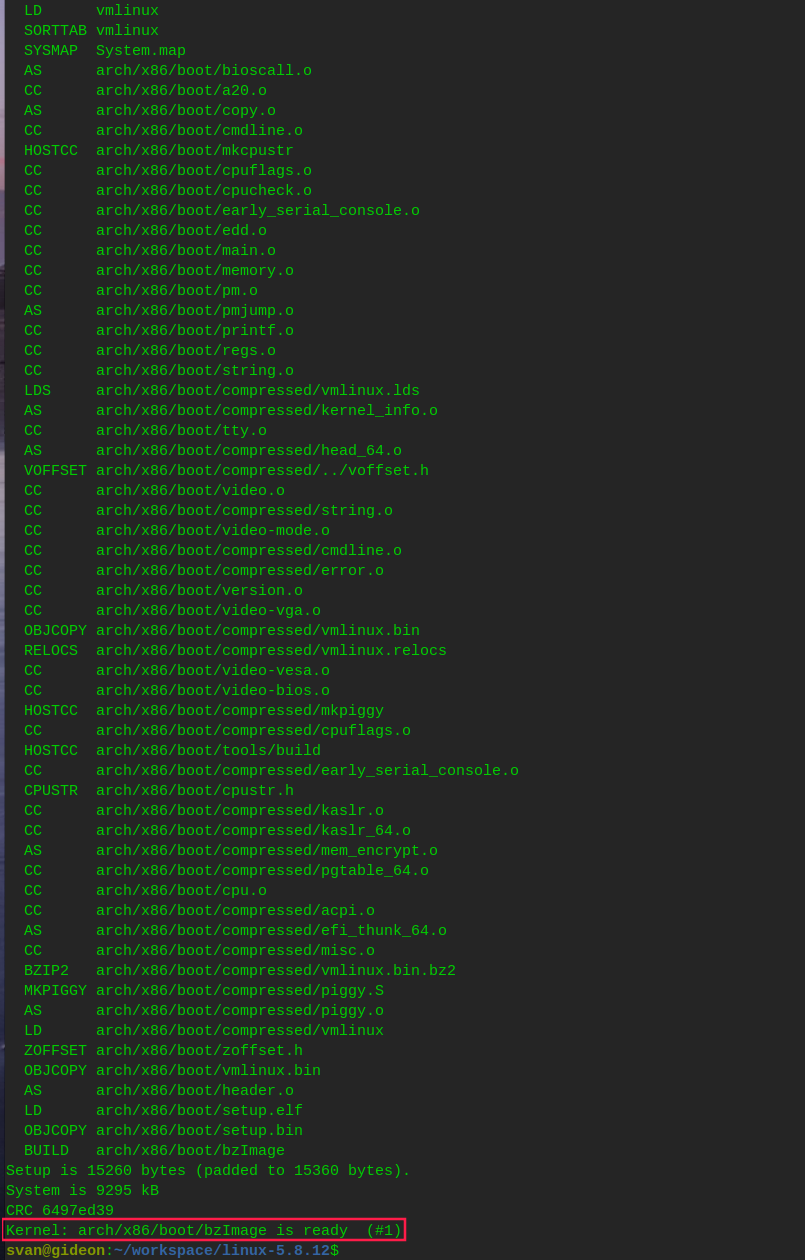


1. 使用$ sudo apt-get install libssl-dev 解决上面的问题。



此时再次使用$ make -j 4 bzImage 编译核心，经过漫长的等待后顺利完成。

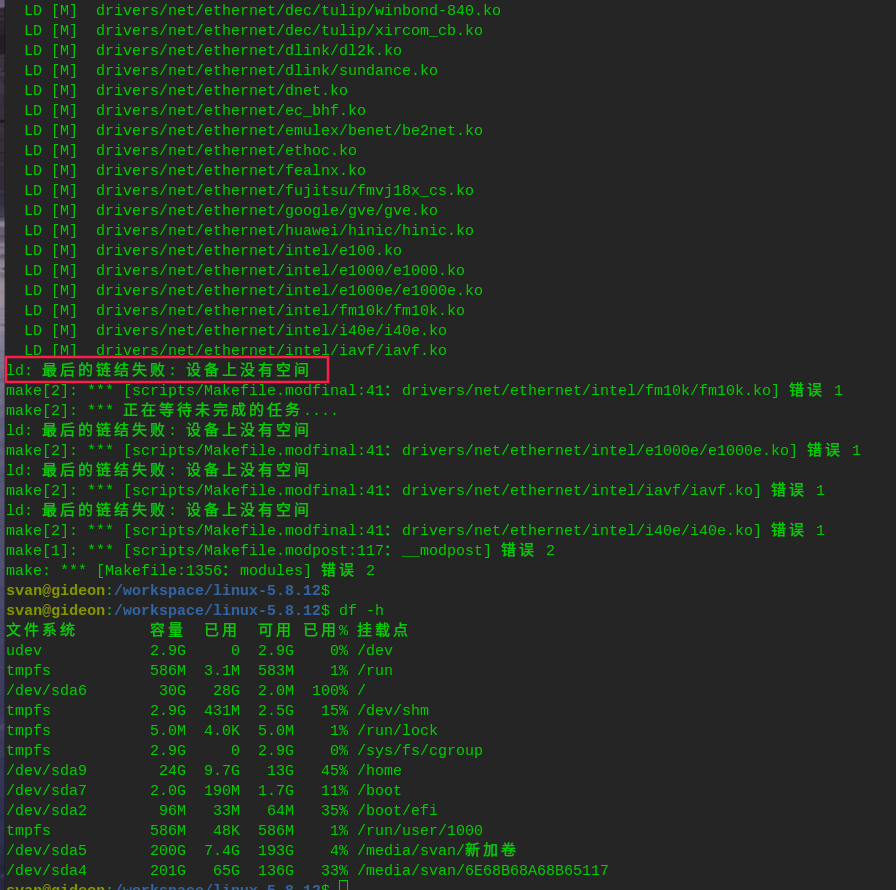




由上面图片中的make -j 4 bzImage 的结果可知bzImage 文件在arch/x86/boot/中。

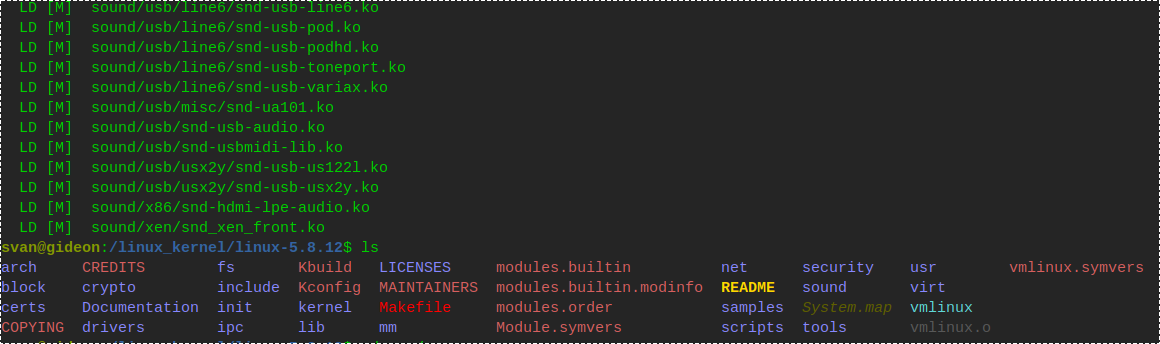


1. 使用$ make -j 4 modules 来编译模块(之前也编译过一次模块，但是由于后来电脑好像黑屏了就没看到最后编译的结果)，发现如下问题：



由上面的错误可知空间不足，解决好空间问题后(我将目录放到了根目录下)。

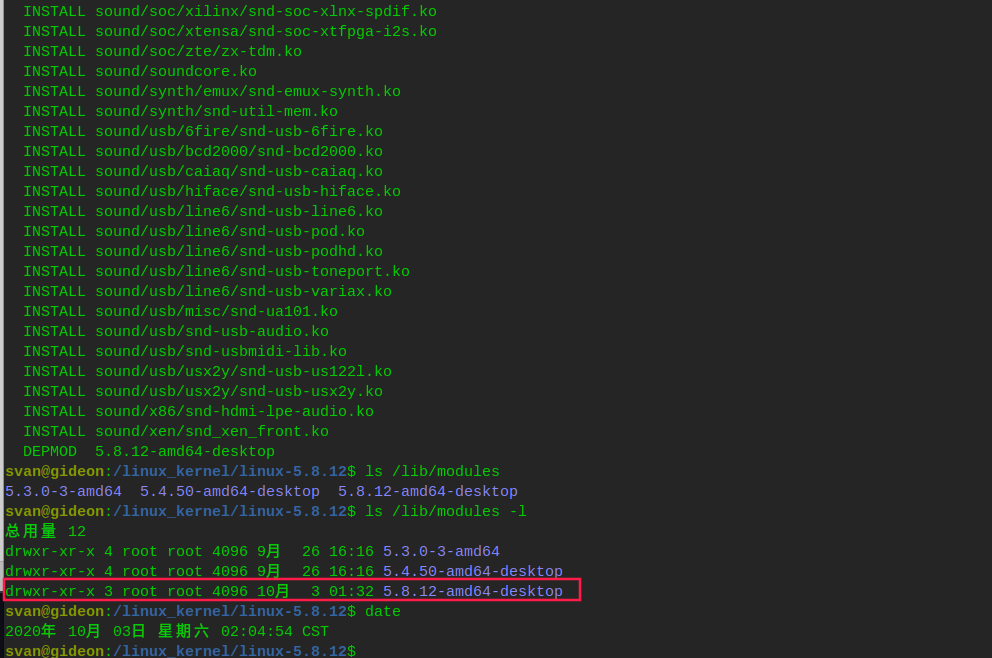
成功执行make -j 4 modules ，并且经过漫长的等待(约两个小时)，终于编译完成。



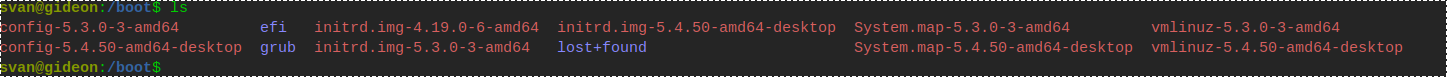
1. 使用$ make modules\_install 安装模块。



经过漫长的等待(约半个小时)，终于安装完成，并且可在/lib/modules/中查看到。



1. 现在将核心文件安装到系统核心文件的目录下，首先查看系统核心文件目录。



安装核心文件到系统核心目录下有两种方法，我们现在依次使用：

首先是方法一(执行如下命令)：

$ cd /linux\_kernel/linux-5.8.12/

$ sudo cp arch/x86/boot/bzImage /boot/vmlinuz-5.8.12-amd64-desktop

$ sudo cp .config /boot/config-5.8.12-amd64-desktop

$ sudo cp System.map /boot/System.map-5.8.12-amd64-desktop

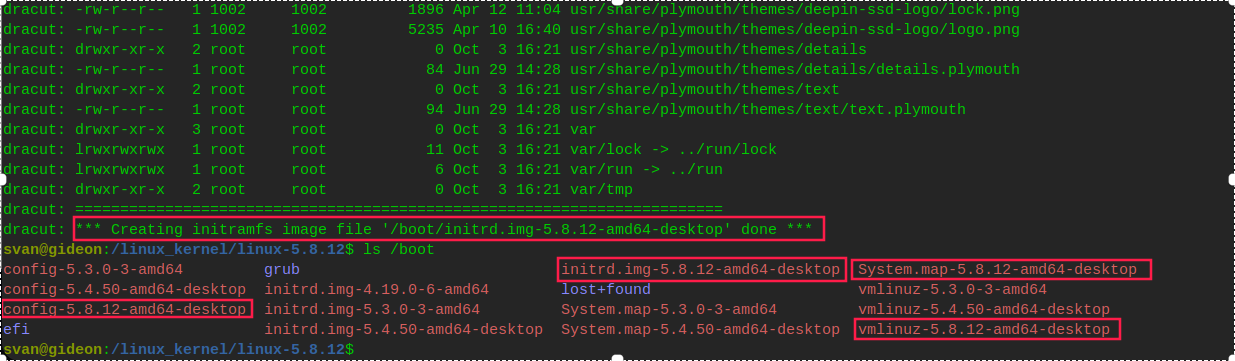
$ sudo dracut -v /boot/initrd.img-5.8.12-amd64-desktop

若没有dracut命令，则自行安装。



下面有很多输出的内容不再显示了，大概一分钟不到就好了。

查看目录 /boot下的内容发现多出来四个。



现在来看方法二(执行如下命令)：

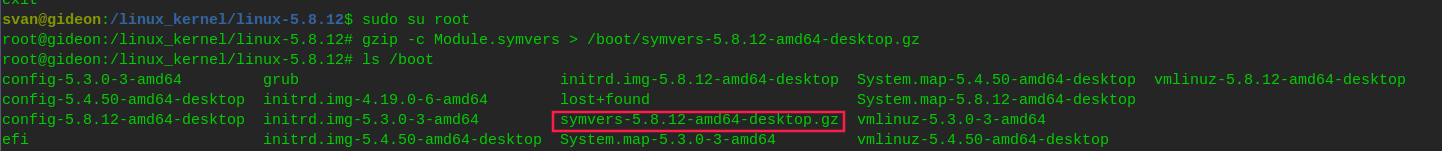
$ sudo make install



上述命令执行完后，还需执行如下命令：

$ sudo su root

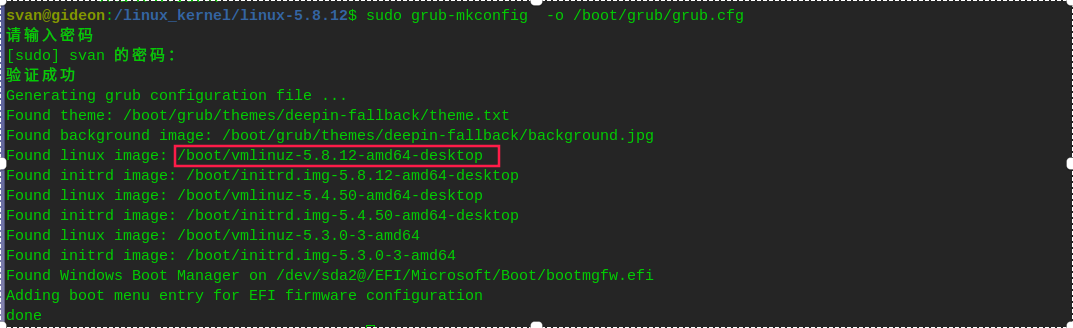
$ gzip -c Module.symvers > /boot/symvers-5.8.12-amd64-desktop.gz



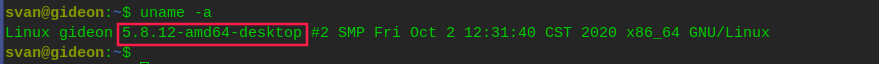
1. 配置开机菜单，使用如下命令：

$ grub-mkconfig -o /boot/grub/grub.cfg

可查看到 vmlinuz-5.8.12-amd64-desktop 应该是最早出现的。



1. 重启后可查看到内核版本为5.8.12。



1. 单一核心模块编译