

# LINUX KERNEL COMPILATION

## Objective:

- Compile the stable linux kernel downloaded from [kernel.org](https://kernel.org) .
- Dual boot it with the current linux version.

## Step - 1:

Grab the current linux kernel version, download it and extract the source code file from tar.xz

```
mohith@mohith-virtual-machine:~$ tar xvf linux-5.8.14.tar
linux-5.8.14/
linux-5.8.14/.clang-format
linux-5.8.14/.coccinconfig
linux-5.8.14/.get_maintainer.ignore
linux-5.8.14/.gitattributes
```

## Step - 2:

Before building the kernel, one must configure Linux kernel features. Copy the existing kernel's configuration file to .config. Here **uname -r** is used to get the existing kernel's version.

```
mohith@mohith-virtual-machine:~/linux-5.8.14$ cp -v /boot/config-$(uname -r) .config
'/boot/config-5.8.14' -> '.config'
mohith@mohith-virtual-machine:~/linux-5.8.14$
```

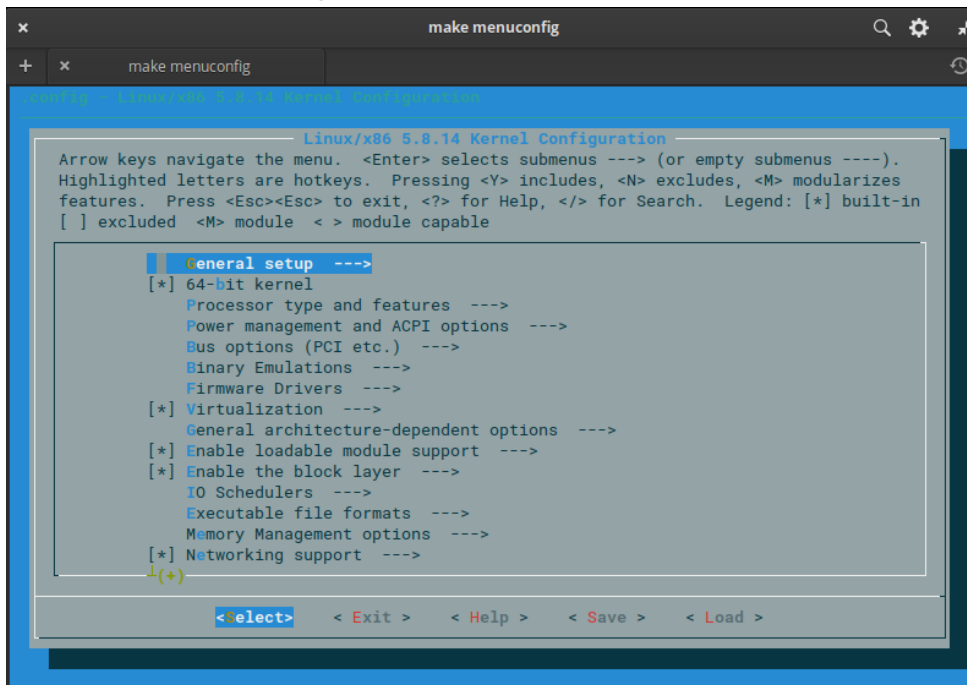
## Step - 3:

Development tools such as GCC compilers and other necessary tools installed to compile the Linux kernel.

```
mohith@mohith-virtual-machine:~/linux-5.8.14$ sudo apt-get install build-essential libncurses-dev bison flex libssl-dev libelf-dev
[sudo] password for mohith:
Reading package lists... Done
Building dependency tree
Reading state information... Done
Note, selecting 'libncurses5-dev' instead of 'libncurses-dev'
```

## Step - 4: (optional)

Start configuring the kernel using test based color menus, radiolists and dialogs.



## Step - 5:

Start compiling using **make** command. This process can also be speeded-up by using multiple threads

**\$ make -j \$(nproc)**

```
mohith@mohith-virtual-machine:~/linux-5.8.14$ nproc
4
mohith@mohith-virtual-machine:~/linux-5.8.14$ make -j 4
DESCEND objtool
CALL scripts/atomic/check-atomics.sh
CALL scripts/checksyscalls.sh
CALL scripts/prepare-patch-for-apply.sh
```

### Step - 6:

Install the linux kernel modules .

```
mohith@mohith-virtual-machine:~/linux-5.8.14$ sudo make modules_install  
[sudo] password for mohith:  
INSTALL arch/x86/crypto/aegis128-aesni.ko  
INSTALL arch/x86/crypto/aesni-intel.ko  
INSTALL arch/x86/crypto/blowfish-x86_64.ko  
INSTALL arch/x86/crypto/camellia-aesni-avx-x86_64.ko  
INSTALL arch/x86/crypto/camellia-aesni-avx2.ko  
INSTALL arch/x86/crypto/camellia-x86_64.ko  
INSTALL arch/x86/crypto/cast5-avx-x86_64.ko  
INSTALL arch/x86/crypto/cast6-avx-x86_64.ko  
INSTALL arch/x86/crypto/chacha-x86_64.ko  
INSTALL arch/x86/crypto/crc32-pclmul.ko  
INSTALL arch/x86/crypto/crct10dif-pclmul.ko  
INSTALL arch/x86/crypto/des3_edc-x86_64.ko  
INSTALL arch/x86/crypto/ghash-clmulni-intel.ko  
INSTALL arch/x86/crypto/glue_helper.ko
```

### Step - 7:

## Install the kernel.

```
mohith@mohith-virtual-machine:~/linux-5.8.14$ sudo make install
[sudo] password for mohith:
sh ./arch/x86/boot/install.sh 5.8.14 arch/x86/boot/bzImage \
    System.map "/boot"
run-parts: executing /etc/kernel/postinst.d/apt-auto-removal 5.8.14 /boot/vmlinuz-5.8.14
run-parts: executing /etc/kernel/postinst.d/initramfs-tools 5.8.14 /boot/vmlinuz-5.8.14
update-initramfs: Generating /boot/initrd.img-5.8.14
```

### Step - 8:

Reboot the system and select the new kernel from the grub menu.

```
GNU GRUB version 2.02

elementary, with Linux 5.8.14
elementary, with Linux 5.8.14 (recovery mode)
elementary, with Linux 5.8.14.old
elementary, with Linux 5.8.14.old (recovery mode)
*elementary, with Linux 5.4.0-42-generic
elementary, with Linux 5.4.0-42-generic (recovery mode)
```

### Step - 9:

Check for the current kernel version.

```
mohith@mohith-virtual-machine:~/linux-5.8.14$ uname -r
5.8.14
mohith@mohith-virtual-machine:~/linux-5.8.14$
```