CS3003D: Operating Systems

Assignment – 1

Name: G.V.S. CHETAN TEJA　　　　Batch: A

Roll No: B190436CS　　　　Date: 6th September

Problem Statement

Download the latest stable Linux kernel from kernel.org, compile it, and dual boot it with your current Linux version. Your current version as well as the new version should be present in the grub-menu.

Methodology

- Dual booting can be directly done with the host OS but if something goes wrong then OS could be corrupted. Hence, it is highly important to load the code accurately and load them.
- Obtain the kernel source code from kernel.org
- Install the development dependencies
- Compile the kernel
- Install the compiled kernel, add grub entry.
- Reboot the system.

Introduction

The Linux Kernel is the foundation of the Unix-like operating systems. The kernel is responsible for communication between hardware and software and the allocation of available resources.

All Linux distributions are based on a predefined kernel. But, if you want to disable several options and drivers or try experimental patches, you need to build a Linux kernel.

The kernel has 4 jobs:

1. Memory management: Keep track of how much memory is used to store what, and where.

2. Process management: Determine which processes can use the central processing unit (CPU), when, and for how long.

3. Device drivers: Act as mediator/interpreter between the hardware and processes.

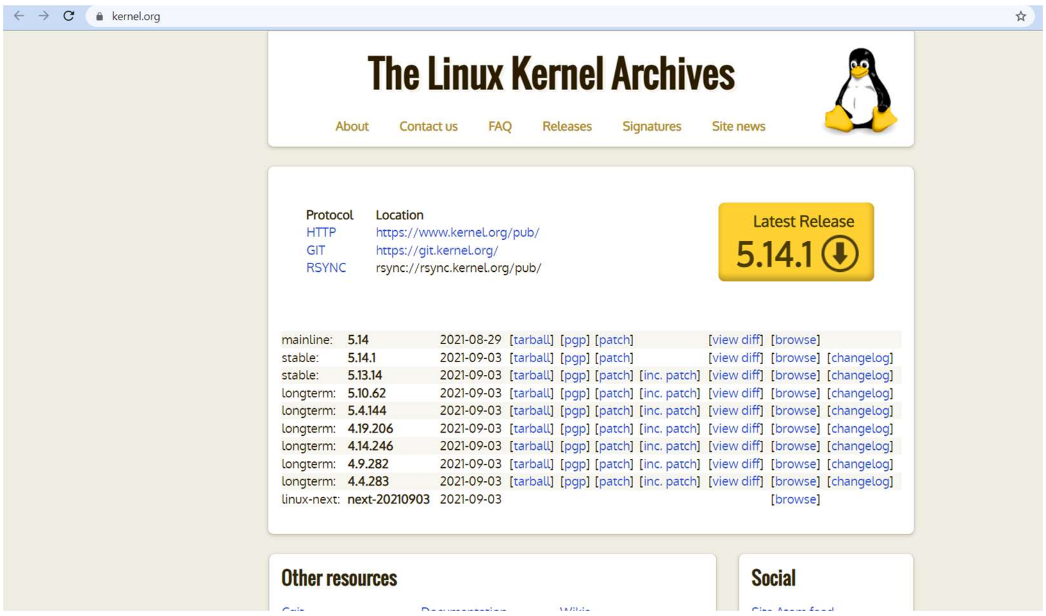
4. System calls and security: Receive requests for service from the processes.

Procedure

The process of building a Linux kernel takes seven easy steps to complete. However, the procedure requires a significant amount of time to complete, depending on the system speed.

Step 1: Get the latest Linux kernel source code

Visit the official project site and download the latest source code. Click on the big yellow button that read as "**Latest Stable Kernel**".

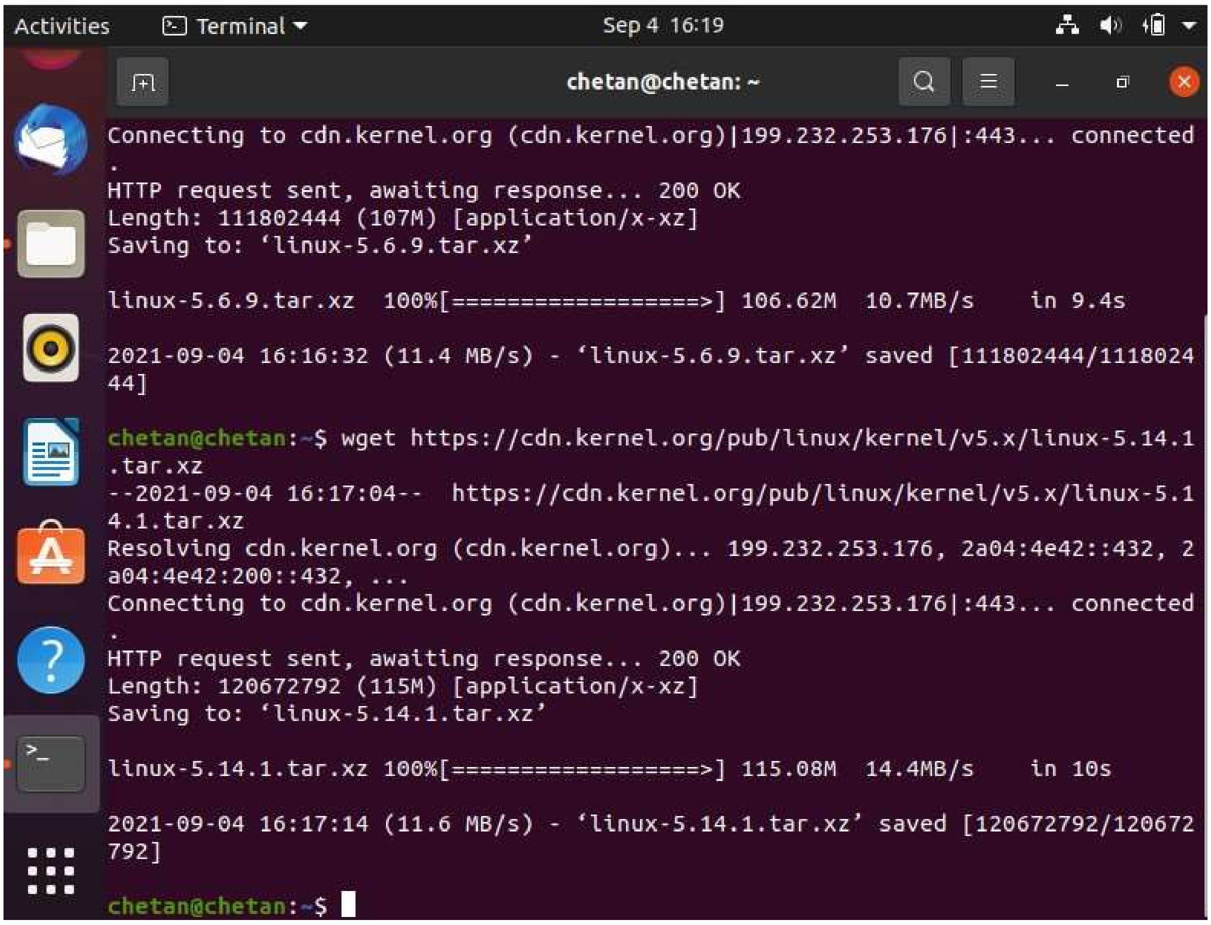


Protocol	Location
HTTP	https://www.kernel.org/pub/
GIT	https://git.kernel.org/
RSYNC	rsync://rsync.kernel.org/pub/

mainline:	5.14	2021-08-29	[tarball]	[pgp]	[patch]		[view diff]	[browse]	
stable:	5.14.1	2021-09-03	[tarball]	[pgp]	[patch]		[view diff]	[browse]	[changelog]
stable:	5.13.14	2021-09-03	[tarball]	[pgp]	[patch]	[inc. patch]	[view diff]	[browse]	[changelog]
longterm:	5.10.62	2021-09-03	[tarball]	[pgp]	[patch]	[inc. patch]	[view diff]	[browse]	[changelog]
longterm:	5.4.144	2021-09-03	[tarball]	[pgp]	[patch]	[inc. patch]	[view diff]	[browse]	[changelog]
longterm:	4.19.206	2021-09-03	[tarball]	[pgp]	[patch]	[inc. patch]	[view diff]	[browse]	[changelog]
longterm:	4.14.246	2021-09-03	[tarball]	[pgp]	[patch]	[inc. patch]	[view diff]	[browse]	[changelog]
longterm:	4.9.282	2021-09-03	[tarball]	[pgp]	[patch]	[inc. patch]	[view diff]	[browse]	[changelog]
longterm:	4.4.283	2021-09-03	[tarball]	[pgp]	[patch]	[inc. patch]	[view diff]	[browse]	[changelog]
linux-next:	next-20210903	2021-09-03						[browse]	

The filename would be linux-x.y.z.tar.xz, where x.y.z is actual Linux kernel version number. For example file linux-5.14.1.tar.xz represents Linux kernel version 5.14.1. Use the wget command to download Linux kernel source code:

```
$ wget https://cdn.kernel.org/pub/linux/kernel/v5.x/linux-5.14.1.tar.xz
```



```
Connecting to cdn.kernel.org (cdn.kernel.org)|199.232.253.176|:443... connected
.
HTTP request sent, awaiting response... 200 OK
Length: 111802444 (107M) [application/x-xz]
Saving to: ‘linux-5.6.9.tar.xz’

linux-5.6.9.tar.xz  100%[=================>] 106.62M  10.7MB/s    in 9.4s

2021-09-04 16:16:32 (11.4 MB/s) - ‘linux-5.6.9.tar.xz’ saved [111802444/1118024
44]

chetan@chetan:~$ wget https://cdn.kernel.org/pub/linux/kernel/v5.x/linux-5.14.1
.tar.xz
--2021-09-04 16:17:04--  https://cdn.kernel.org/pub/linux/kernel/v5.x/linux-5.1
4.1.tar.xz
Resolving cdn.kernel.org (cdn.kernel.org)... 199.232.253.176, 2a04:4e42::432, 2
a04:4e42:200::432, ...
Connecting to cdn.kernel.org (cdn.kernel.org)|199.232.253.176|:443... connected
.
HTTP request sent, awaiting response... 200 OK
Length: 120672792 (115M) [application/x-xz]
Saving to: ‘linux-5.14.1.tar.xz’

linux-5.14.1.tar.xz 100%[=================>] 115.08M  14.4MB/s    in 10s

2021-09-04 16:17:14 (11.6 MB/s) - ‘linux-5.14.1.tar.xz’ saved [120672792/120672
792]

chetan@chetan:~$
```

Either of the following can be done to download the tar file.

Step 2: Extract tar.xz file

You really don’t have to extract the source code in /usr/src. You can extract the source code in your $HOME directory or any other directory using the following unzx command or xz command:

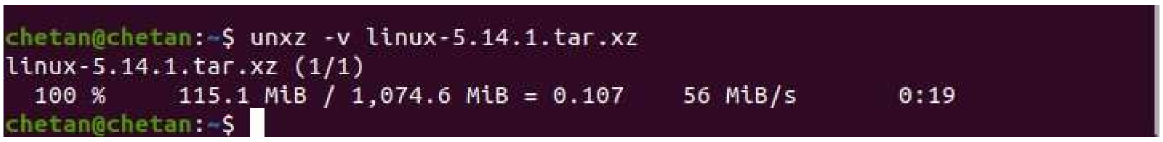
```
$ unxz -v linux-5.6.9.tar.xz
```

Or

```
$ xz -d -v linux-5.6.9.tar.xz
```

Or

tar xf archive.tar.xz



```
chetan@chetan:~$ unxz -v linux-5.14.1.tar.xz
linux-5.14.1.tar.xz (1/1)
  100 %     115.1 MiB / 1,074.6 MiB = 0.107    56 MiB/s       0:19
chetan@chetan:~$
```

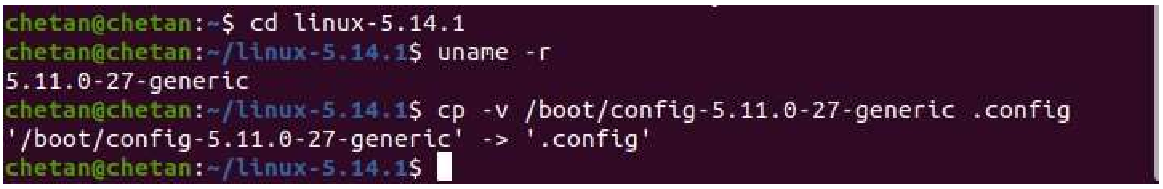
Step 3: Configure the Linux kernel features and modules

Before start building the kernel, one must configure Linux kernel features. You must also specify which kernel modules (drivers) needed for your system. The task can be overwhelming for a new user. I suggest that you copy existing config file using the cp command:

```
$ cd linux-5.6.9
$ cp -v /boot/config-$(uname -r) .config
```

Sample outputs:

```
'/boot/config-4.15.0-30-generic' -> '.config'
```



```
chetan@chetan:~$ cd linux-5.14.1
chetan@chetan:~/linux-5.14.1$ uname -r
5.11.0-27-generic
chetan@chetan:~/linux-5.14.1$ cp -v /boot/config-5.11.0-27-generic .config
'/boot/config-5.11.0-27-generic' -> '.config'
chetan@chetan:~/linux-5.14.1$
```

Step 4: Install the required compilers and other tools

You must have development tools such as GCC compilers and related tools installed to compile the Linux kernel.

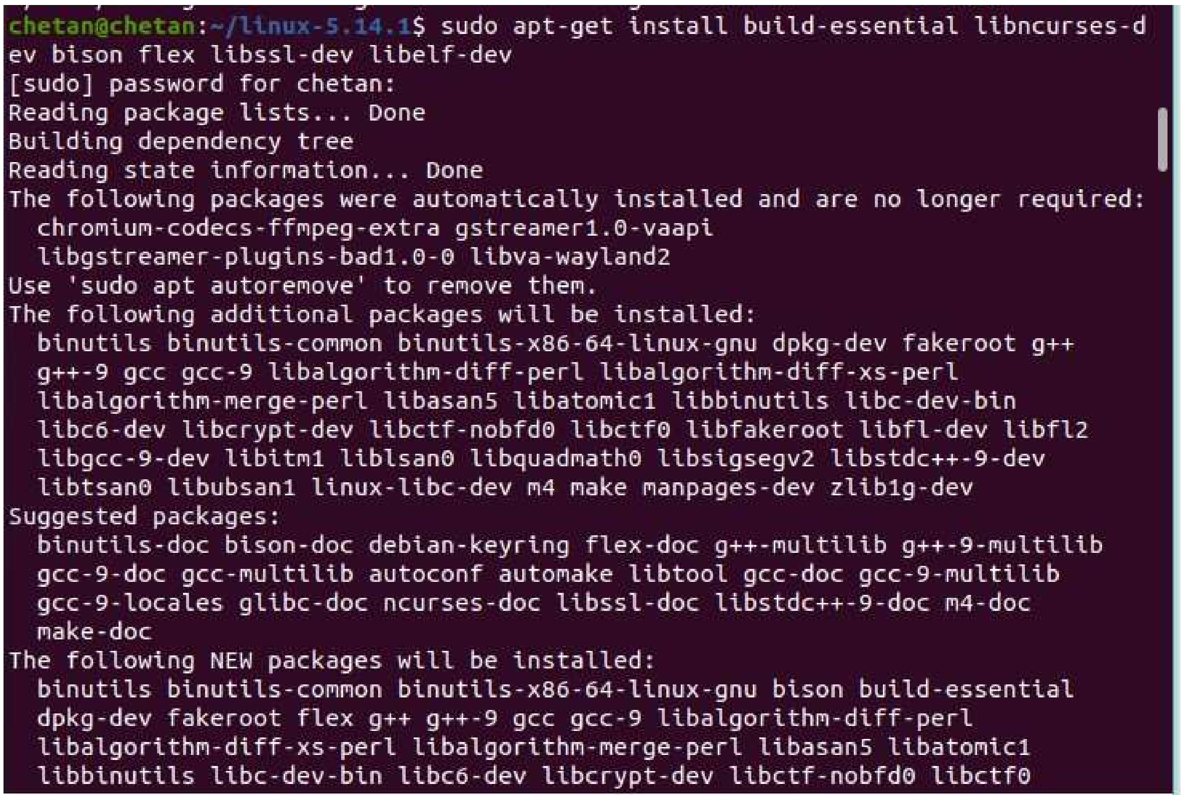
To install GCC and development tools on ubuntu linux use the following command.

```
$ sudo apt-get install build-essential libncurses-dev bison flex
libssl-dev libelf-dev
```

The command we used above installs the following packages:

Package	**Package description**

`git`	Tracks and makes a record of all changes during development in the source code. It also allows reverting the changes.
`fakeroot`	Packaging tool that makes the fake root environment.
`build-essential`	Installs development tools such as C, C++, gcc, and g++.
`ncurses-dev`	Programming library that provides API for the text-based terminals.
`xz-utils`	Provides fast file compression and decompression.
`libssl-dev`	Supports SSL and TSL that encrypt data and make the internet connection secure.
`bc` (Basic Calculator)	A mathematical scripting language that supports the interactive execution of statements.
`flex` (Fast Lexical Analyzer Generator)	Generates lexical analyzers that convert characters into tokens.
`libelf-dev`	Issues a shared library for managing ELF files (executable files, core dumps and object code)
`bison`	GNU parser generator that converts grammar description to a C program.



```
chetan@chetan:~/linux-5.14.1$ sudo apt-get install build-essential libncurses-d
ev bison flex libssl-dev libelf-dev
[sudo] password for chetan:
Reading package lists... Done
Building dependency tree
Reading state information... Done
The following packages were automatically installed and are no longer required:
  chromium-codecs-ffmpeg-extra gstreamer1.0-vaapi
  libgstreamer-plugins-bad1.0-0 libva-wayland2
Use 'sudo apt autoremove' to remove them.
The following additional packages will be installed:
  binutils binutils-common binutils-x86-64-linux-gnu dpkg-dev fakeroot g++
  g++-9 gcc gcc-9 libalgorithm-diff-perl libalgorithm-diff-xs-perl
  libalgorithm-merge-perl libasan5 libatomic1 libbinutils libc-dev-bin
  libc6-dev libcrypt-dev libctf-nobfd0 libctf0 libfakeroot libfl-dev libfl2
  libgcc-9-dev libitm1 liblsan0 libquadmath0 libsigsegv2 libstdc++-9-dev
  libtsan0 libubsan1 linux-libc-dev m4 make manpages-dev zlib1g-dev
Suggested packages:
  binutils-doc bison-doc debian-keyring flex-doc g++-multilib g++-9-multilib
  gcc-9-doc gcc-multilib autoconf automake libtool gcc-doc gcc-9-multilib
  gcc-9-locales glibc-doc ncurses-doc libssl-doc libstdc++-9-doc m4-doc
  make-doc
The following NEW packages will be installed:
  binutils binutils-common binutils-x86-64-linux-gnu bison build-essential
  dpkg-dev fakeroot flex g++ g++-9 gcc gcc-9 libalgorithm-diff-perl
  libalgorithm-diff-xs-perl libalgorithm-merge-perl libasan5 libatomic1
  libbinutils libc-dev-bin libc6-dev libcrypt-dev libctf-nobfd0 libctf0
```

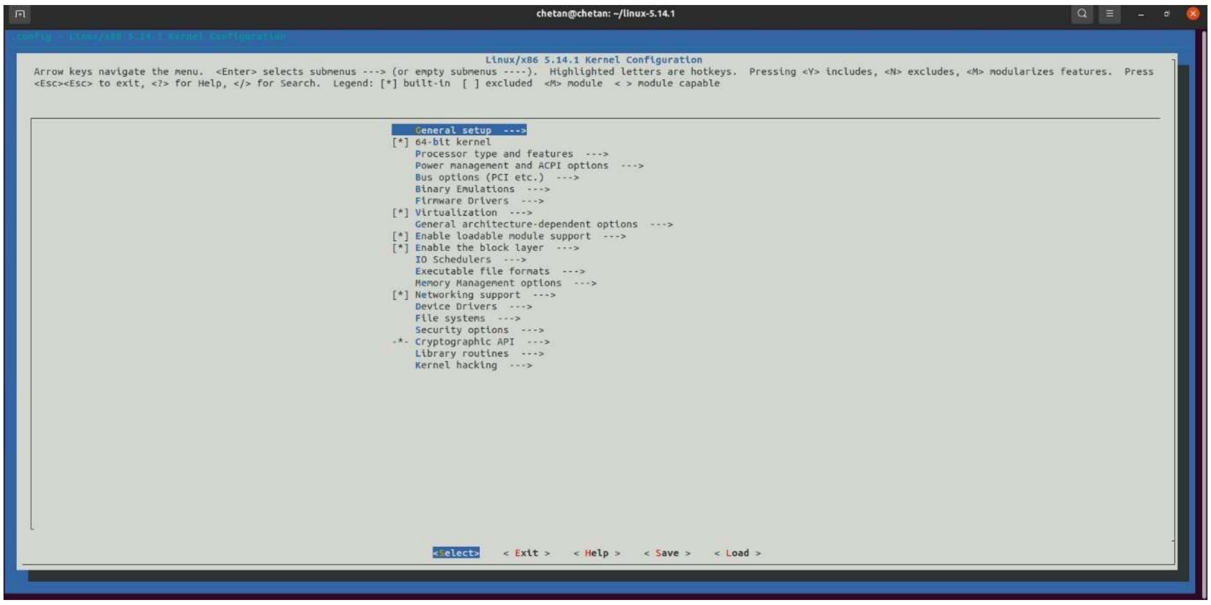
Step 5. Configuring the kernel

Now you can start the kernel configuration by typing any one of the following command in source code directory:

- $ `make menuconfig` – Text based color menus, radiolists & dialogs. This option also useful on remote server if you wanna compile kernel remotely.
- $ `make xconfig` – X windows (Qt) based configuration tool, works best under KDE desktop
- $ `make gconfig` – X windows (Gtk) based configuration tool, works best under Gnome Dekstop.

For example, run `make menuconfig` command launches following screen:

```
$ make menuconfig
```



```
chetan@chetan: ~/linux-5.14.1

                              Linux/x86 5.14.1 Kernel Configuration
 Arrow keys navigate the menu.  <Enter> selects submenus ---> (or empty submenus ----).  Highlighted letters are hotkeys.  Pressing <Y> includes, <N> excludes, <M> modularizes features.  Press
 <Esc><Esc> to exit, <?> for Help, </> for Search.  Legend: [*] built-in  [ ] excluded  <M> module  < > module capable

                    General setup  --->
                [*] 64-bit kernel
                    Processor type and features  --->
                    Power management and ACPI options  --->
                    Bus options (PCI etc.)  --->
                    Binary Emulations  --->
                    Firmware Drivers  --->
                [*] Virtualization  --->
                    General architecture-dependent options  --->
                [*] Enable loadable module support  --->
                [*] Enable the block layer  --->
                    IO Schedulers  --->
                    Executable file formats  --->
                    Memory Management options  --->
                [*] Networking support  --->
                    Device Drivers  --->
                    File systems  --->
                    Security options  --->
                -*- Cryptographic API  --->
                    Library routines  --->
                    Kernel hacking  --->

                  <Select>    < Exit >    < Help >    < Save >    < Load >
```

You have to select different options as per your need. Each configuration option has HELP button associated with it so select help button to get help. Please note that 'make menuconfig' is optional. I used it here to demonstration purpose only. You can enable or disable certain features or kernel driver with this option. It is easy to remove support for a device driver or option and end up with a broken kernel. For example, if the ext4 driver is removed from the kernel configuration file, a system may not boot. When in doubt, just leave support in the kernel.

Step 5. How to compile a Linux Kernel

Start compiling and tocreate a compressed kernel image, enter:

```
$ make
```

To speed up compile time, pass the `-j` as follows:

```
## use 4 core/thread ##
$ make -j 4
## get thread or cpu core count using nproc command ##
$ make -j $(nproc)
```

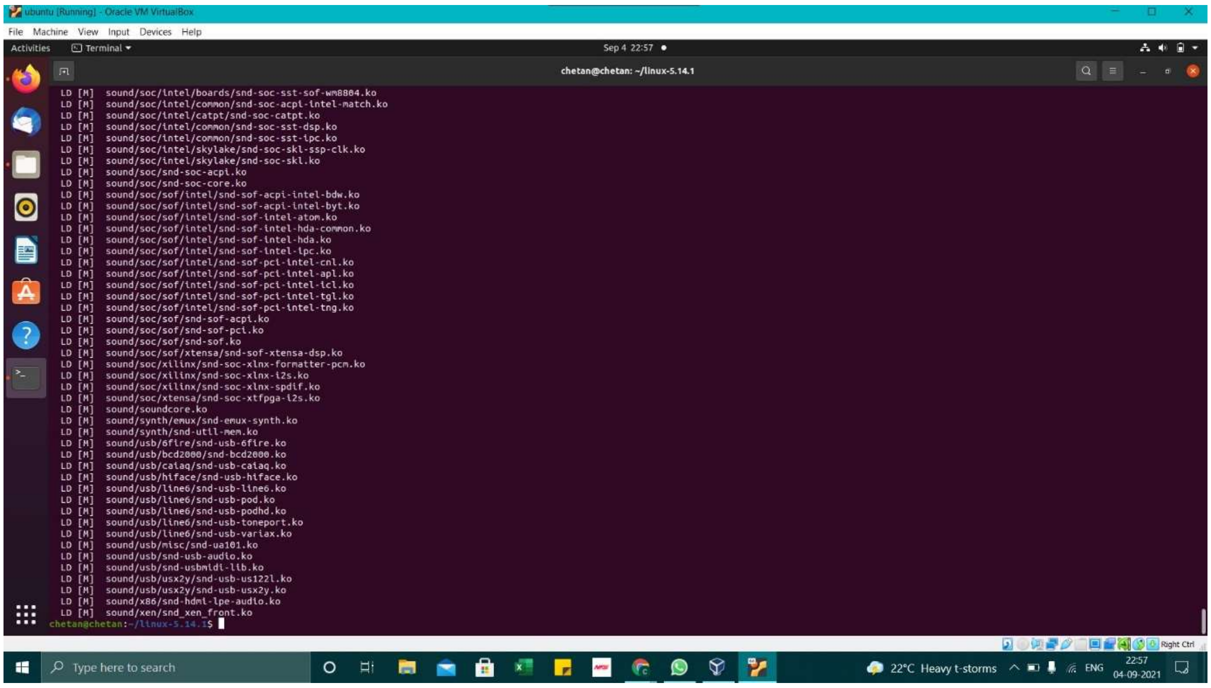
```
The $ nproc on my system gave 4 using which I performed the make
command.
```

```
$ make -j 4
```

Compiling and building the Linux kernel going take a significant amount of time. The build time depends upon your system’s resources such as available CPU core and the current system load. So we must have some patience.

The terminal lists all Linux kernel components: memory management, hardware device drivers, filesystem drivers, network drivers, and process management.

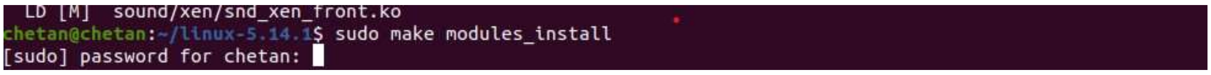
The end of this step looks like this



```
  LD [M]  sound/soc/intel/boards/snd-soc-sst-sof-wm8804.ko
  LD [M]  sound/soc/intel/common/snd-soc-acpi-intel-match.ko
  LD [M]  sound/soc/intel/catpt/snd-soc-catpt.ko
  LD [M]  sound/soc/intel/common/snd-soc-sst-dsp.ko
  LD [M]  sound/soc/intel/common/snd-soc-sst-ipc.ko
  LD [M]  sound/soc/intel/skylake/snd-soc-skl-ssp-clk.ko
  LD [M]  sound/soc/intel/skylake/snd-soc-skl.ko
  LD [M]  sound/soc/snd-soc-acpi.ko
  LD [M]  sound/soc/snd-soc-core.ko
  LD [M]  sound/soc/sof/intel/snd-sof-acpi-intel-bdw.ko
  LD [M]  sound/soc/sof/intel/snd-sof-acpi-intel-byt.ko
  LD [M]  sound/soc/sof/intel/snd-sof-intel-atom.ko
  LD [M]  sound/soc/sof/intel/snd-sof-intel-hda-common.ko
  LD [M]  sound/soc/sof/intel/snd-sof-intel-hda.ko
  LD [M]  sound/soc/sof/intel/snd-sof-intel-ipc.ko
  LD [M]  sound/soc/sof/intel/snd-sof-pci-intel-cnl.ko
  LD [M]  sound/soc/sof/intel/snd-sof-pci-intel-apl.ko
  LD [M]  sound/soc/sof/intel/snd-sof-pci-intel-icl.ko
  LD [M]  sound/soc/sof/intel/snd-sof-pci-intel-tgl.ko
  LD [M]  sound/soc/sof/intel/snd-sof-pci-intel-tng.ko
  LD [M]  sound/soc/sof/snd-sof-acpi.ko
  LD [M]  sound/soc/sof/snd-sof-pci.ko
  LD [M]  sound/soc/sof/snd-sof.ko
  LD [M]  sound/soc/sof/xtensa/snd-sof-xtensa-dsp.ko
  LD [M]  sound/soc/xilinx/snd-soc-xlnx-formatter-pcm.ko
  LD [M]  sound/soc/xilinx/snd-soc-xlnx-i2s.ko
  LD [M]  sound/soc/xilinx/snd-soc-xlnx-spdif.ko
  LD [M]  sound/soc/xtensa/snd-soc-xtfpga-i2s.ko
  LD [M]  sound/soundcore.ko
  LD [M]  sound/synth/emux/snd-emux-synth.ko
  LD [M]  sound/synth/snd-util-mem.ko
  LD [M]  sound/usb/6fire/snd-usb-6fire.ko
  LD [M]  sound/usb/bcd2000/snd-bcd2000.ko
  LD [M]  sound/usb/caiaq/snd-usb-caiaq.ko
  LD [M]  sound/usb/hiface/snd-usb-hiface.ko
  LD [M]  sound/usb/line6/snd-usb-line6.ko
  LD [M]  sound/usb/line6/snd-usb-pod.ko
  LD [M]  sound/usb/line6/snd-usb-podhd.ko
  LD [M]  sound/usb/line6/snd-usb-toneport.ko
  LD [M]  sound/usb/line6/snd-usb-variax.ko
  LD [M]  sound/usb/misc/snd-ua101.ko
  LD [M]  sound/usb/snd-usb-audio.ko
  LD [M]  sound/usb/snd-usbmidi-lib.ko
  LD [M]  sound/usb/usx2y/snd-usb-us122l.ko
  LD [M]  sound/usb/usx2y/snd-usb-usx2y.ko
  LD [M]  sound/x86/snd-hdmi-lpe-audio.ko
  LD [M]  sound/xen/snd_xen_front.ko
chetan@chetan:~/linux-5.14.1$
```

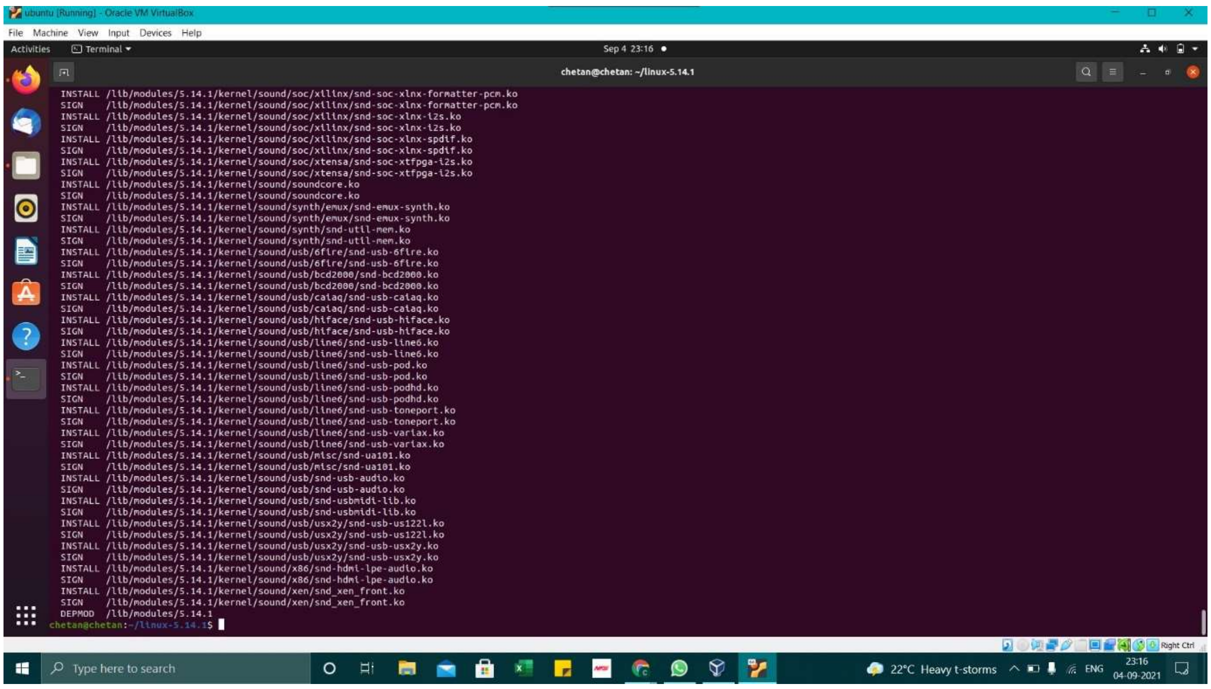
Install the Linux kernel modules

```
$ sudo make modules_install
```



```
  LD [M]  sound/xen/snd_xen_front.ko
chetan@chetan:~/linux-5.14.1$ sudo make modules_install
[sudo] password for chetan:
```

The end looks like this.

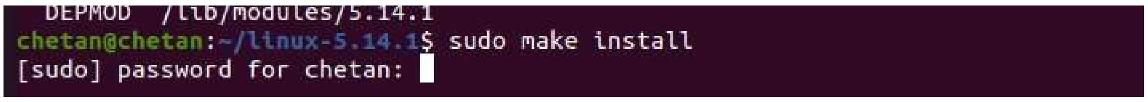


```
 INSTALL /lib/modules/5.14.1/kernel/sound/soc/xilinx/snd-soc-xlnx-formatter-pcm.ko
 SIGN    /lib/modules/5.14.1/kernel/sound/soc/xilinx/snd-soc-xlnx-formatter-pcm.ko
 INSTALL /lib/modules/5.14.1/kernel/sound/soc/xilinx/snd-soc-xlnx-i2s.ko
 SIGN    /lib/modules/5.14.1/kernel/sound/soc/xilinx/snd-soc-xlnx-i2s.ko
 INSTALL /lib/modules/5.14.1/kernel/sound/soc/xilinx/snd-soc-xlnx-spdif.ko
 SIGN    /lib/modules/5.14.1/kernel/sound/soc/xilinx/snd-soc-xlnx-spdif.ko
 INSTALL /lib/modules/5.14.1/kernel/sound/soc/xtensa/snd-soc-xtfpga-i2s.ko
 SIGN    /lib/modules/5.14.1/kernel/sound/soc/xtensa/snd-soc-xtfpga-i2s.ko
 INSTALL /lib/modules/5.14.1/kernel/sound/soundcore.ko
 SIGN    /lib/modules/5.14.1/kernel/sound/soundcore.ko
 INSTALL /lib/modules/5.14.1/kernel/sound/synth/emux/snd-emux-synth.ko
 SIGN    /lib/modules/5.14.1/kernel/sound/synth/emux/snd-emux-synth.ko
 INSTALL /lib/modules/5.14.1/kernel/sound/synth/snd-util-mem.ko
 SIGN    /lib/modules/5.14.1/kernel/sound/synth/snd-util-mem.ko
 INSTALL /lib/modules/5.14.1/kernel/sound/usb/6fire/snd-usb-6fire.ko
 SIGN    /lib/modules/5.14.1/kernel/sound/usb/6fire/snd-usb-6fire.ko
 INSTALL /lib/modules/5.14.1/kernel/sound/usb/bcd2000/snd-bcd2000.ko
 SIGN    /lib/modules/5.14.1/kernel/sound/usb/bcd2000/snd-bcd2000.ko
 INSTALL /lib/modules/5.14.1/kernel/sound/usb/caiaq/snd-usb-caiaq.ko
 SIGN    /lib/modules/5.14.1/kernel/sound/usb/caiaq/snd-usb-caiaq.ko
 INSTALL /lib/modules/5.14.1/kernel/sound/usb/hiface/snd-usb-hiface.ko
 SIGN    /lib/modules/5.14.1/kernel/sound/usb/hiface/snd-usb-hiface.ko
 INSTALL /lib/modules/5.14.1/kernel/sound/usb/line6/snd-usb-line6.ko
 SIGN    /lib/modules/5.14.1/kernel/sound/usb/line6/snd-usb-line6.ko
 INSTALL /lib/modules/5.14.1/kernel/sound/usb/line6/snd-usb-pod.ko
 SIGN    /lib/modules/5.14.1/kernel/sound/usb/line6/snd-usb-pod.ko
 INSTALL /lib/modules/5.14.1/kernel/sound/usb/line6/snd-usb-podhd.ko
 SIGN    /lib/modules/5.14.1/kernel/sound/usb/line6/snd-usb-podhd.ko
 INSTALL /lib/modules/5.14.1/kernel/sound/usb/line6/snd-usb-toneport.ko
 SIGN    /lib/modules/5.14.1/kernel/sound/usb/line6/snd-usb-toneport.ko
 INSTALL /lib/modules/5.14.1/kernel/sound/usb/line6/snd-usb-variax.ko
 SIGN    /lib/modules/5.14.1/kernel/sound/usb/line6/snd-usb-variax.ko
 INSTALL /lib/modules/5.14.1/kernel/sound/usb/misc/snd-ua101.ko
 SIGN    /lib/modules/5.14.1/kernel/sound/usb/misc/snd-ua101.ko
 INSTALL /lib/modules/5.14.1/kernel/sound/usb/snd-usb-audio.ko
 SIGN    /lib/modules/5.14.1/kernel/sound/usb/snd-usb-audio.ko
 INSTALL /lib/modules/5.14.1/kernel/sound/usb/snd-usbmidi-lib.ko
 SIGN    /lib/modules/5.14.1/kernel/sound/usb/snd-usbmidi-lib.ko
 INSTALL /lib/modules/5.14.1/kernel/sound/usb/usx2y/snd-usb-us122l.ko
 SIGN    /lib/modules/5.14.1/kernel/sound/usb/usx2y/snd-usb-us122l.ko
 INSTALL /lib/modules/5.14.1/kernel/sound/usb/usx2y/snd-usb-usx2y.ko
 SIGN    /lib/modules/5.14.1/kernel/sound/usb/usx2y/snd-usb-usx2y.ko
 INSTALL /lib/modules/5.14.1/kernel/sound/x86/snd-hdmi-lpe-audio.ko
 SIGN    /lib/modules/5.14.1/kernel/sound/x86/snd-hdmi-lpe-audio.ko
 INSTALL /lib/modules/5.14.1/kernel/sound/xen/snd_xen_front.ko
 SIGN    /lib/modules/5.14.1/kernel/sound/xen/snd_xen_front.ko
 DEPMOD  /lib/modules/5.14.1
chetan@chetan:~/linux-5.14.1$
```

Install the Linux kernel

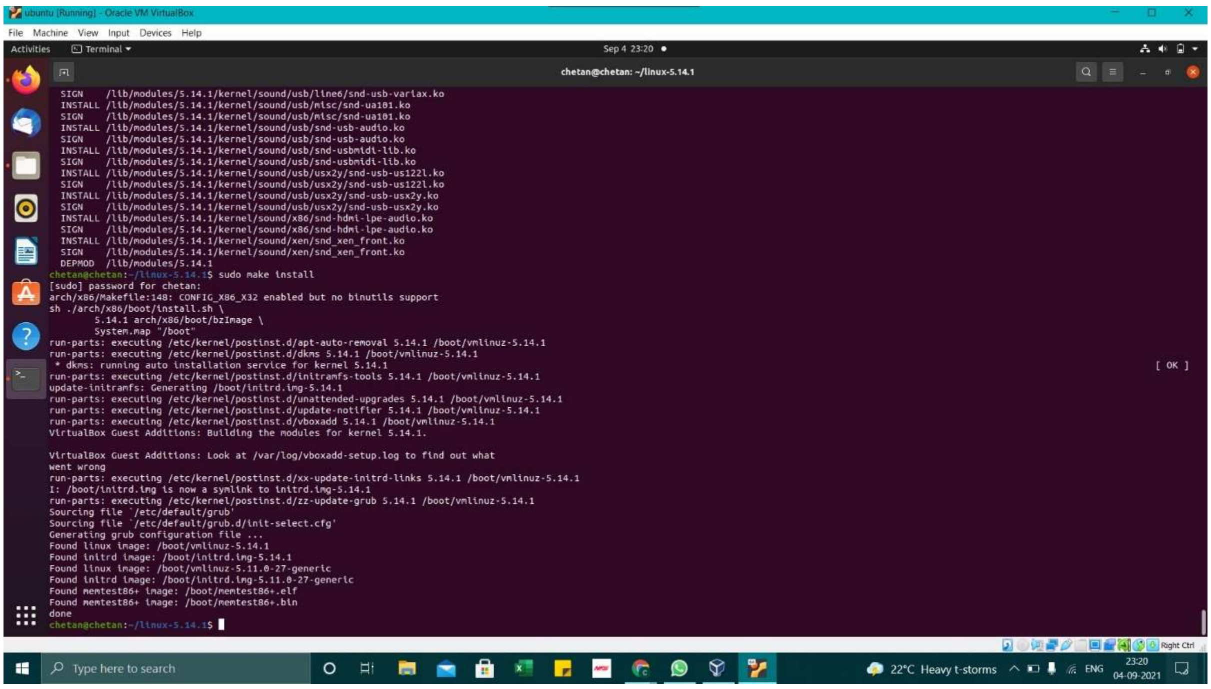
So far we have compiled the Linux kernel and installed kernel modules. It is time to install the kernel itself:

```
$ sudo make install
```



```
  DEPMOD  /lib/modules/5.14.1
chetan@chetan:~/linux-5.14.1$ sudo make install
[sudo] password for chetan:
```

The end for this process



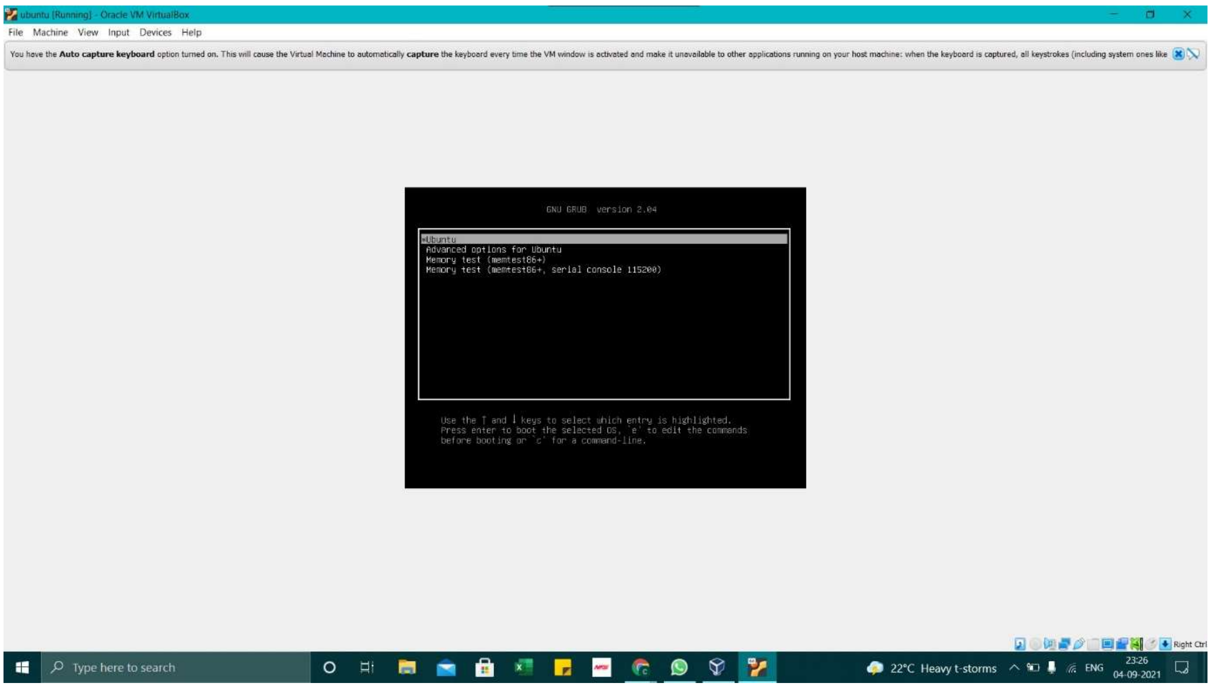
```
  SIGN    /lib/modules/5.14.1/kernel/sound/usb/line6/snd-usb-variax.ko
  INSTALL /lib/modules/5.14.1/kernel/sound/usb/misc/snd-ua101.ko
  SIGN    /lib/modules/5.14.1/kernel/sound/usb/misc/snd-ua101.ko
  INSTALL /lib/modules/5.14.1/kernel/sound/usb/snd-usb-audio.ko
  SIGN    /lib/modules/5.14.1/kernel/sound/usb/snd-usb-audio.ko
  INSTALL /lib/modules/5.14.1/kernel/sound/usb/snd-usbmidi-lib.ko
  SIGN    /lib/modules/5.14.1/kernel/sound/usb/snd-usbmidi-lib.ko
  INSTALL /lib/modules/5.14.1/kernel/sound/usb/usx2y/snd-usb-us122l.ko
  SIGN    /lib/modules/5.14.1/kernel/sound/usb/usx2y/snd-usb-us122l.ko
  INSTALL /lib/modules/5.14.1/kernel/sound/usb/usx2y/snd-usb-usx2y.ko
  SIGN    /lib/modules/5.14.1/kernel/sound/usb/usx2y/snd-usb-usx2y.ko
  INSTALL /lib/modules/5.14.1/kernel/sound/x86/snd-hdmi-lpe-audio.ko
  SIGN    /lib/modules/5.14.1/kernel/sound/x86/snd-hdmi-lpe-audio.ko
  INSTALL /lib/modules/5.14.1/kernel/sound/xen/snd_xen_front.ko
  SIGN    /lib/modules/5.14.1/kernel/sound/xen/snd_xen_front.ko
  DEPMOD  /lib/modules/5.14.1
chetan@chetan:~/linux-5.14.1$ sudo make install
[sudo] password for chetan:
arch/x86/Makefile:148: CONFIG_X86_X32 enabled but no binutils support
sh ./arch/x86/boot/install.sh \
        5.14.1 arch/x86/boot/bzImage \
        System.map "/boot"
run-parts: executing /etc/kernel/postinst.d/apt-auto-removal 5.14.1 /boot/vmlinuz-5.14.1
run-parts: executing /etc/kernel/postinst.d/dkms 5.14.1 /boot/vmlinuz-5.14.1
 * dkms: running auto installation service for kernel 5.14.1                                        [ OK ]
run-parts: executing /etc/kernel/postinst.d/initramfs-tools 5.14.1 /boot/vmlinuz-5.14.1
update-initramfs: Generating /boot/initrd.img-5.14.1
run-parts: executing /etc/kernel/postinst.d/unattended-upgrades 5.14.1 /boot/vmlinuz-5.14.1
run-parts: executing /etc/kernel/postinst.d/update-notifier 5.14.1 /boot/vmlinuz-5.14.1
run-parts: executing /etc/kernel/postinst.d/vboxadd 5.14.1 /boot/vmlinuz-5.14.1
VirtualBox Guest Additions: Building the modules for kernel 5.14.1.

VirtualBox Guest Additions: Look at /var/log/vboxadd-setup.log to find out what
went wrong
run-parts: executing /etc/kernel/postinst.d/xx-update-initrd-links 5.14.1 /boot/vmlinuz-5.14.1
I: /boot/initrd.img is now a symlink to initrd.img-5.14.1
run-parts: executing /etc/kernel/postinst.d/zz-update-grub 5.14.1 /boot/vmlinuz-5.14.1
Sourcing file `/etc/default/grub'
Sourcing file `/etc/default/grub.d/init-select.cfg'
Generating grub configuration file ...
Found linux image: /boot/vmlinuz-5.14.1
Found initrd image: /boot/initrd.img-5.14.1
Found linux image: /boot/vmlinuz-5.11.0-27-generic
Found initrd image: /boot/initrd.img-5.11.0-27-generic
Found memtest86+ image: /boot/memtest86+.elf
Found memtest86+ image: /boot/memtest86+.bin
done
chetan@chetan:~/linux-5.14.1$ 
```

Grub Menu

You should ideally reboot the virtual system after the completion of this stage.

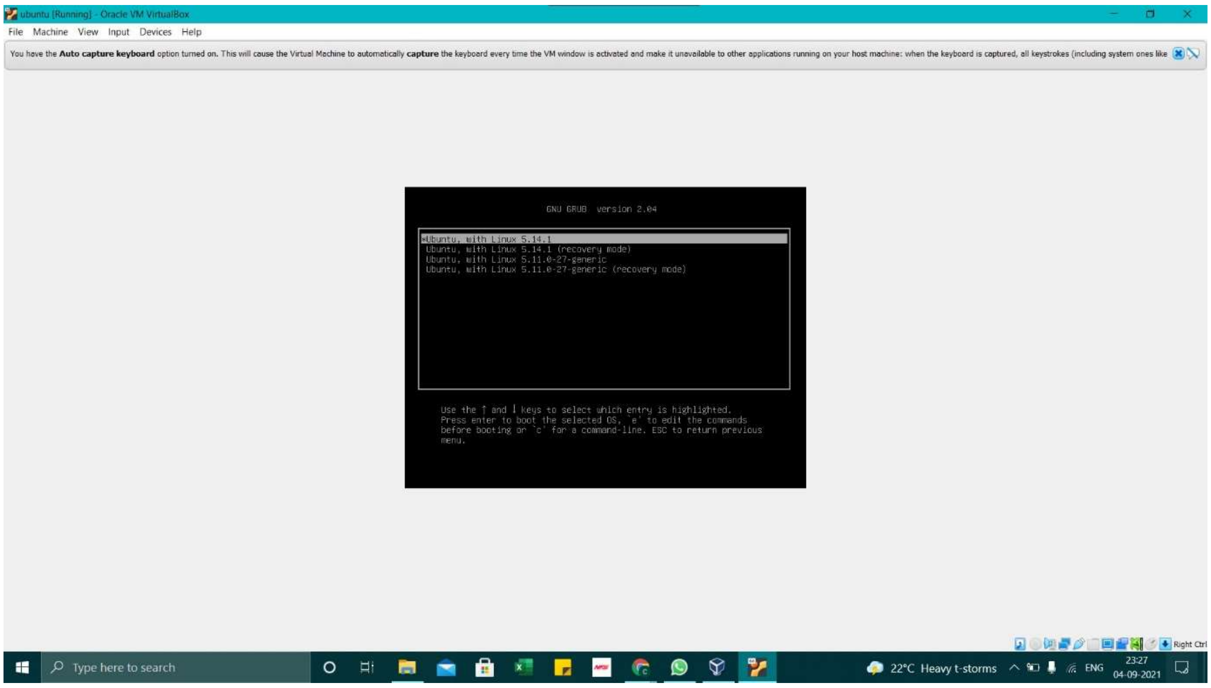
On turning on hold the right shift key during boot up to access the GRUB menu, which looks like this as shown below-



```
                         GNU GRUB  version 2.04

 *Ubuntu
  Advanced options for Ubuntu
  Memory test (memtest86+)
  Memory test (memtest86+, serial console 115200)

     Use the ↑ and ↓ keys to select which entry is highlighted.
     Press enter to boot the selected OS, `e' to edit the commands
     before booting or `c' for a command-line.
```


ubuntu [Running] - Oracle VM VirtualBox
File Machine View Input Devices Help
Type here to search
22°C Heavy t-storms
ENG
23:27
04-09-2021
Right Ctrl

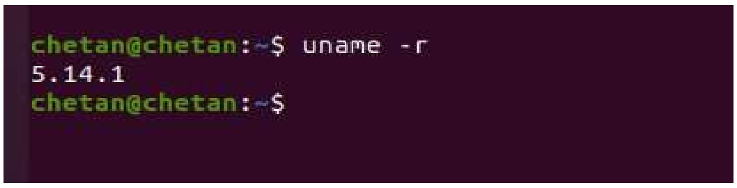
You have the **Auto capture keyboard** option turned on. This will cause the Virtual Machine to automatically **capture** the keyboard every time the VM window is activated and make it unavailable to other applications running on your host machine: when the keyboard is captured, all keystrokes (including system ones like

```
GNU GRUB  version 2.04

*Ubuntu, with Linux 5.14.1
 Ubuntu, with Linux 5.14.1 (recovery mode)
 Ubuntu, with Linux 5.11.0-27-generic
 Ubuntu, with Linux 5.11.0-27-generic (recovery mode)

   Use the ↑ and ↓ keys to select which entry is highlighted.
   Press enter to boot the selected OS, `e' to edit the commands
   before booting or `c' for a command-line. ESC to return previous
   menu.
```

To show that the latest kernel is installed, use the command in the note given above, this will print a more specific string with actual release.

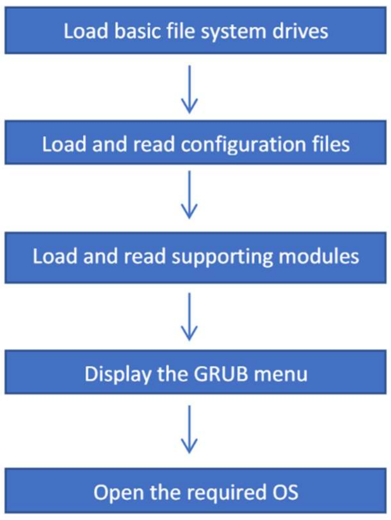


```
chetan@chetan:~$ uname -r
5.14.1
chetan@chetan:~$
```

Conclusion – Linux Compile Kernel version 5.6.9

Configurations! You completed various steps to build the Linux kernel from source code and compiled kernel should be running on your system.

Flow Chart


Load basic file system drives
Load and read configuration files
Load and read supporting modules
Display the GRUB menu
Open the required OS