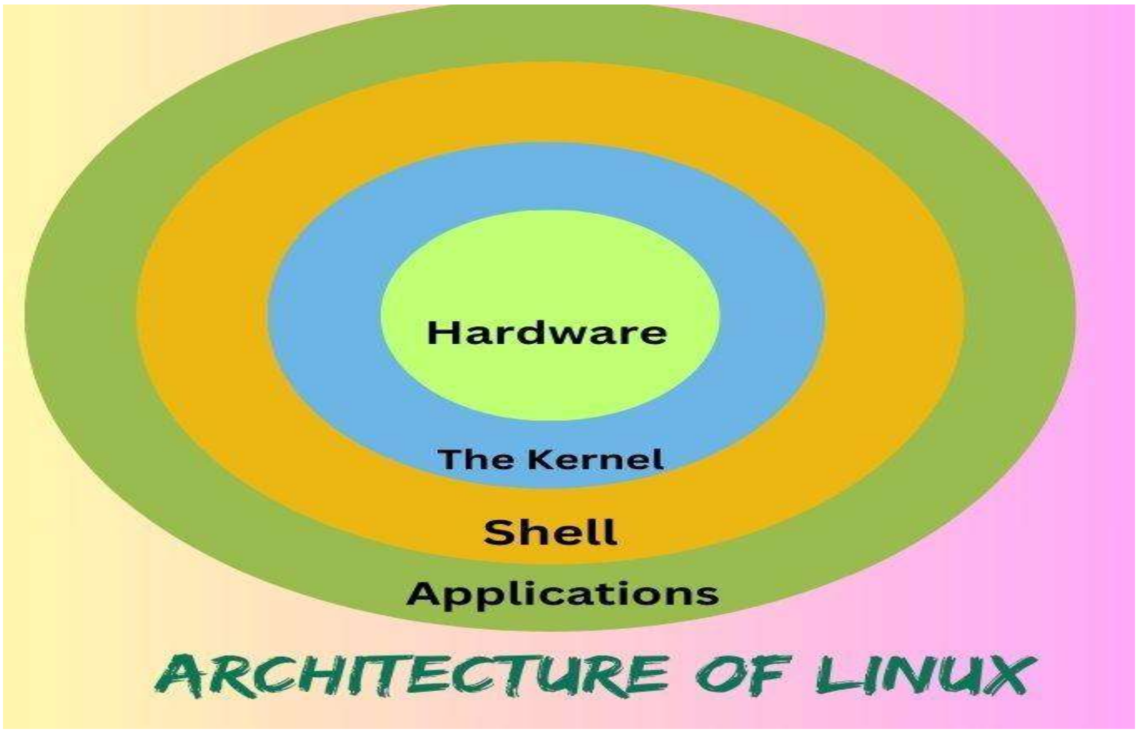
Unit 5 Getting Started with Linux, Linux Booting, Linux Admin (Ubuntu)

- History of Linux
- GNU, GPL Concept
- Open Source & Freeware
- Structure and Features of Linux
- Installation and Configuration of Linux
    - Using with Ubuntu
- Startup, Shutdown and boot loaders of Linux
- Linux Booting Process
    - LILO Configuration
    - GRUB Configuration
- Creating Linux User Account and Password
- Installing and Managing Samba Server
- Installing and Managing Apache Server
- Configure Ubuntu's Built-In Firewall
- Working with WINE

1) History of Linux

The History of Linux began in 1991 with the commencement of a personal project by a Finnish student, Linus Torvalds, to create a new free operating system kernel.

Since then, the resulting Linux kernel has been marked by constant growth throughout its history. Since the initial release of its source code in 1991, it has grown from a small number of C files under a license prohibiting commercial distribution to its state in 2009 of over 370 megabytes of source under the GNU General Public License.



Advantages of Linux

- The main advantage of Linux is it is an open-source operating system. This means the source code is easily available for everyone and you are allowed to contribute, modify and distribute the code to anyone without any permissions.
- In terms of security, Linux is more secure than any other operating system. It does not mean that Linux is 100 percent secure, it has some malware for it but is less vulnerable than any other operating system. So, it does not require any anti-virus software.
- The software updates in Linux are easy and frequent.
- Various Linux distributions are available so that you can use them according to your requirements or according to your taste.
- Linux is freely available to use on the internet.
- It has large community support.
- It provides high stability. It rarely slows down or freezes and there is no need to reboot it after a short time.

- It maintains the privacy of the user.
- The performance of the Linux system is much higher than other operating systems. It allows a large number of people to work at the same time and it handles them efficiently.
- It is network friendly.
- The flexibility of Linux is high. There is no need to install a complete Linux suite; you are allowed to install only the required components.
- Linux is compatible with a large number of file formats.
- It is fast and easy to install from the web. It can also install it on any hardware even on your old computer system.
- It performs all tasks properly even if it has limited space on the hard disk.

Disadvantages of Linux

- It is not very user-friendly. So, it may be confusing for beginners.
- It has small peripheral hardware drivers as compared to windows.

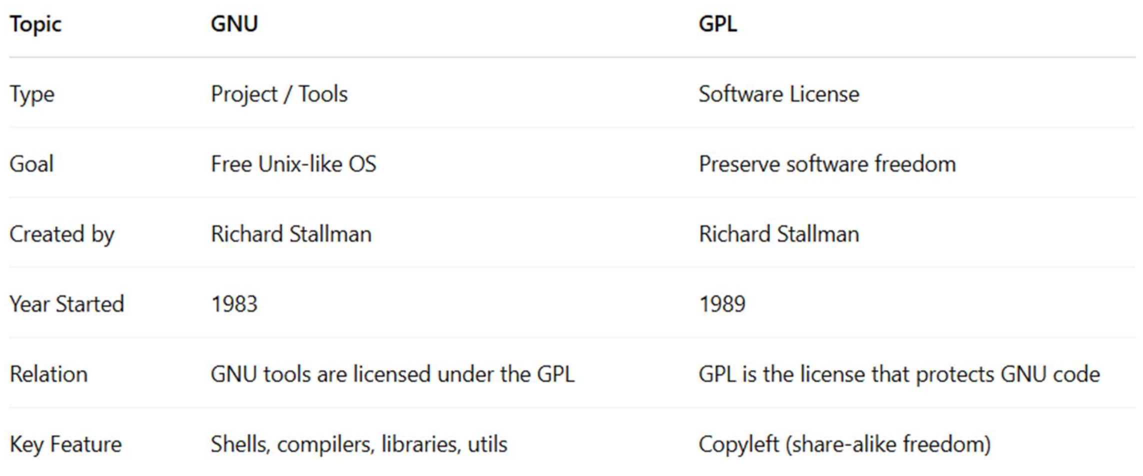
GNU, GPL Concept

What is GNU?

- **GNU** is a **free software project** started by **Richard Stallman** in 1983.
- Goal: Create a complete **Unix-like OS** that is **free software**.
- "Free" here means **freedom**, not necessarily **price**.
- It's recursive: **GNU stands for "GNU's Not Unix"**, reflecting that it's Unix-compatible but not Unix.

What is GPL?

- The **GNU General Public License** is a **free software license** created by Richard Stallman for the GNU project.
- Ensures that software remains **free** and that **freedoms are preserved** when the software is distributed.

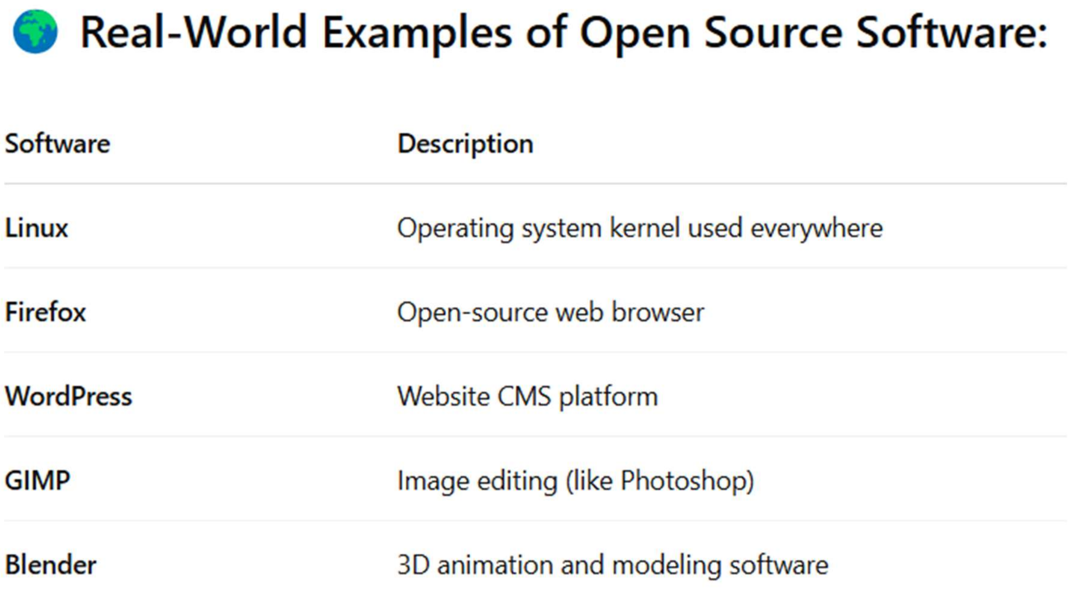


Topic	GNU	GPL
Type	Project / Tools	Software License
Goal	Free Unix-like OS	Preserve software freedom
Created by	Richard Stallman	Richard Stallman
Year Started	1983	1989
Relation	GNU tools are licensed under the GPL	GPL is the license that protects GNU code
Key Feature	Shells, compilers, libraries, utils	Copyleft (share-alike freedom)

Open Source & Freeware

Open Source means that the **source code** of a software program is **freely available** for **anyone to view, use, modify, and share**.

The term **Open-source** is closely related to Open-source software (OSS). Open-source software is a type of computer software that is released under a license, but the source code is made available to all the users. The copyright holders of such software allow the users to use it and do some valuable modifications in its source code to add some new features, to improve the existing features, and to fix bugs if there are any. Because of this reason only Open-source software is mostly developed collaboratively.



🌍 Real-World Examples of Open Source Software:

Software	Description
Linux	Operating system kernel used everywhere
Firefox	Open-source web browser
WordPress	Website CMS platform
GIMP	Image editing (like Photoshop)
Blender	3D animation and modeling software

Some famous examples of Open-source products are :

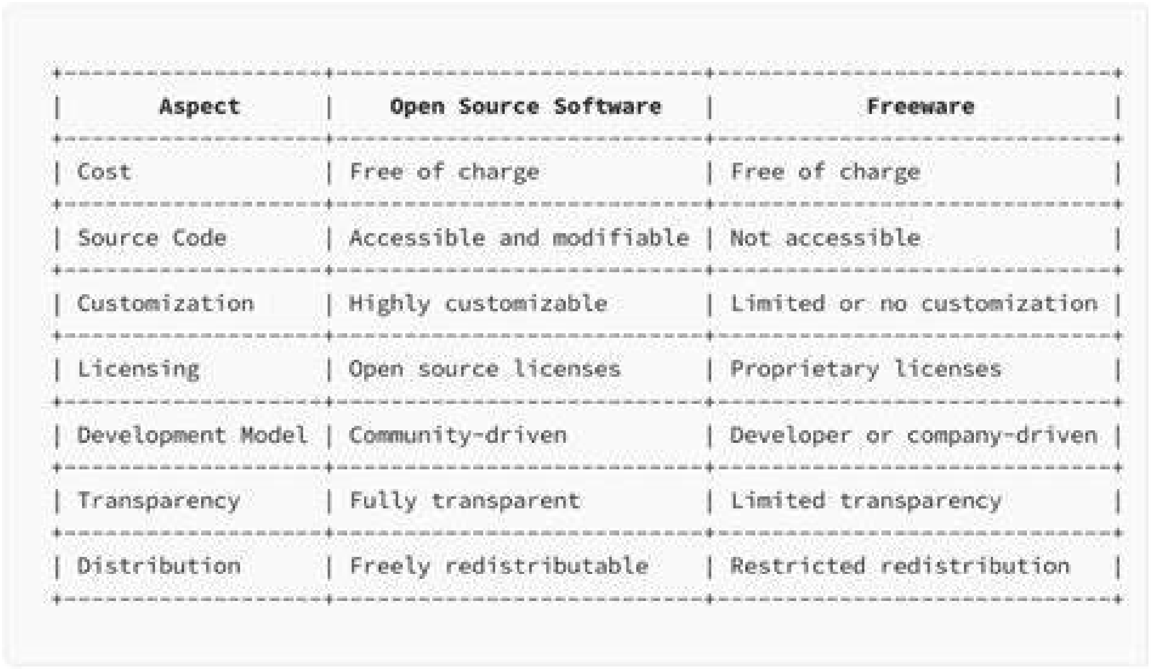
- **Operating systems** – Android, Ubuntu, Linux
- **Internet browsers** – Mozilla Firefox, Chromium
- **Integrated Development Environment (IDEs)** – Vs code (Visual Studio Code), Android Studio, PyCharm, Xcode

What is Freeware?

Freeware is a copyrighted program, application, or software that may be downloaded, installed, used, and shared without restriction. For the general public, these programs are free to use. Free software is not the same as freeware since the latter's source code can be altered by the user for republishing or program integration.

Freeware is software that has no expiration date, no payments or contributions required no limits on the number of times you may download or run it, and no requirement for purchased

licenses to use it. Freeware is software that may be downloaded for free and generally has barred features. Freeware typically limits the user's ability to use, copy, distribute, alter, create derivative works, and reverse engineer the program.
It may still be constrictive in certain respects, though. On the other side, free software has no limitations whatsoever, allowing the user to do everything they wish with the program.



Aspect	Open Source Software	Freeware
Cost	Free of charge	Free of charge
Source Code	Accessible and modifiable	Not accessible
Customization	Highly customizable	Limited or no customization
Licensing	Open source licenses	Proprietary licenses
Development Model	Community-driven	Developer or company-driven
Transparency	Fully transparent	Limited transparency
Distribution	Freely redistributable	Restricted redistribution

Installation and Configuration of Linux
In this guide I will cover the installation of Ubuntu Linux 11.10, 32-bit version from a LiveCD. The installation of other 'Ubuntu versions including 64-bit will be very similar to this guide. I will try to explain certain options and provide helpful hints along the way, so rather than just following the guide, you can understand the reasoning behind the decisions.
This guide is drawn from my experience. As some of you are aware, I'm passionate about open-source software and OS', in particular Linux. I'm no professional though, just a person that enjoys spending his spare time using Linux and open-source software in- between dissecting others' computers, and somehow managing to turn my own systems into fireballs of destruction!

If there are any mistakes please draw my attention to them and I will correct as needed. I have tried to make this as simple as possible, whilst covering the vast majority of scenarios users will come across whilst installing this operating system.

Step 1:
The first thing you should do is head to http://www.ubuntu.com/download/ubuntu/download and download Ubuntu 11.10 Live CD. Just click the big orange Start download box.

Step 2:
Using your disc burning software, burn the .iso you downloaded to a CD.

Step 3:
Before you go any further, ensure all important data is backed up in case of data loss on your drives. This guide assumes you have media backups of your Windows partitioned hard drive and you are safe to proceed.

***Warning: Installing another operating system without first ensuring you have backups of your current files and operating system is a big risk. If you have no data to lose or you've backed up important data, you're ready to proceed. YOU are responsible if you lose data.For those of you using Windows, and installing Linux for the first time I recommend you either use a separate hard disk that does not contain the Windows OS, or create a partition big enough for Linux within Windows using Disk Management in the Administrative Tools menu of the control panel. 30GB of hard disk space is absolutely plenty of space for you to begin exploring Ubuntu whilst at the same time having room to grow.

Step 4:
Ensure you have a network cable connected, restart your computer, and boot from the CD drive.

Step 5:
The LiveCD will load up, and you'll be presented by the following box:
For the purposes of this guide, we will assume you've already tried Ubuntu and want to proceed with an installation, so click "Install Ubuntu."

Step 6:
You'll be greeted by the "Preparing to install Ubuntu" screen, exactly as below:

I recommend you select "Install third-party software" as I have done in the screenshot above. I prefer to do system updates once up and running, but if you have the extra time you can also select "download updates while installing" as well. Then click continue.

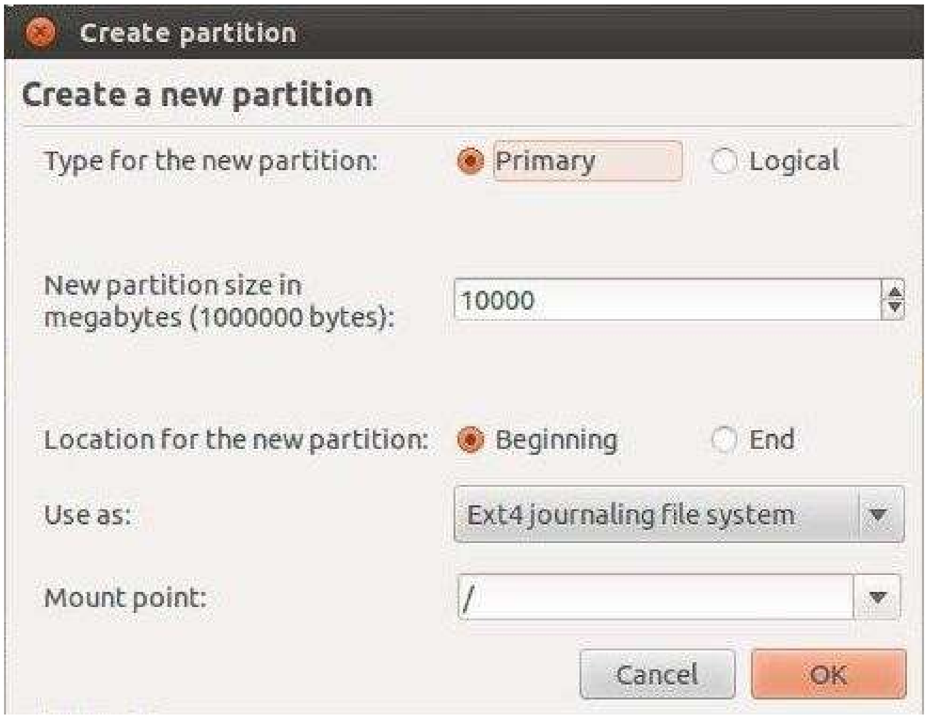
Step 7:
The next screen you will see is "installation type," what you see will be dependent on whether you have an existing Windows installation or not.
I'm going to split this into three different sub-steps, to make it as simple as possible.

Step 7-A:

For those installing in a virtual machine or to hard disks without an OS you will see the following screen:
You have two choices:
Erase the entire disk and use all of it for installation -- Ubuntu will automatically partition your disk and proceed with installation.
Select "something else" and manually create your partitions (which is covered in detail in step 7-C).
If you are choosing the first option, select the radio button and then click continue, proceeding to step 8.

Step 7-B:
Those of you that have current Windows installations or are going to dual-boot with another existing OS will be presented with a screensimilar to below:
You have three options available:

You can choose the first option and install Ubuntu alongside your existing OS.
You can opt to replace your Windows installation with Ubuntu, allowing the installer to format your current partitions and automatically create new ones for Linux.
You can choose "something else" and create your own partition scheme and sizing (covered in detail in step 7-C).



Create a new partition

Type for the new partition: [✓ Primary] Logical

New partition size in megabytes (1000000 bytes): 10000

Location for the new partition: [✓ Beginning] End

Use as: Ext4 journaling file system

Mount point: /

You will notice I have already filled out the example above to create a 10GB root partition.

You can have a maximum of 4 primary partitions, or 3 primary partitions and 1 logical (which allows for another 64 partitions) The size above is 10.00GB. e.g 1,000 = 1GB 10,000 = 10GB (Remember to leave enough free remaining space to create your SWAP partition!) Location for new partition: e.g. do you want it at the start or end of the free space. Select beginning.
Use as: Ext4 is the recommended file system for Ubuntu, much the same as NTFS is Windows. SWAP is for SWAP space. Mount point: This is where you want the partition to mount. E.g. we need a root partition, which in Linux is denoted bya "/".

Click OK once you have finished setting the partition information and you will return to your partition screen, now showing the root partition you just created. Using the same methods as before, create a SWAP partition.

I recommend you set the size of your SWAP partition to at least the size of your available RAM. If you have plenty of hard disk capacity I would suggest you use double the size. So if you have 2GB of RAM, set it to either 2GB or 4GB. For best performance it is recommended you have your SWAP partition at the beginning or end of your drive.
Once you have done that, you should be looking at something like below.

So to re-cap the above, (in my example) we have the following:

- /dev/sda1 is your Windows partition.
- /dev/sda2 is your new root partition (Windows equiv. of C .
- /dev/sda3 is your SWAP space.

Once you are happy with the changes you have made, click install now and proceed to the next step.

Step 8:
As the installation starts to copy the required files to the hard disk, you will be presented with a screen to select your locale. It should automatically find where you are, as it has for me already: Just double check it is correct, and then select continue.

Step 9:
The next screen to appear will be keyboard layout:
Ensure the correct option is selected, above you will see the correct (and default UK) selection has been automatically made for me.

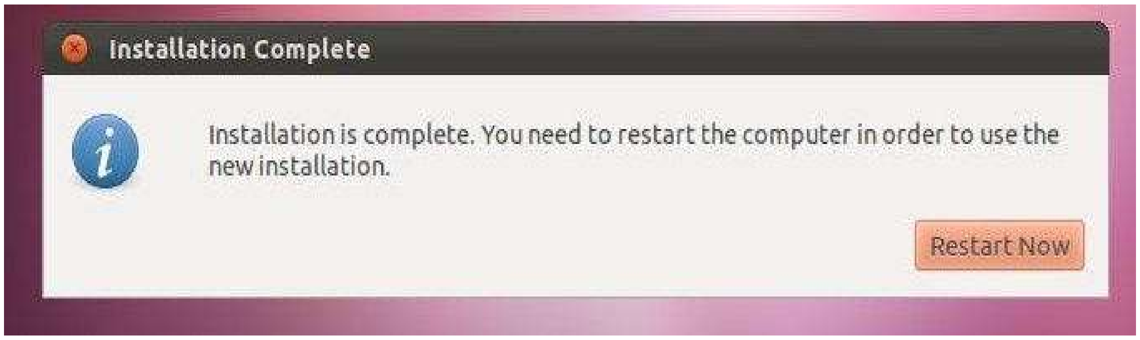
Step 10:
You will now be greeted by the "who are you" screen, ready for you to fill out with your user details:

The computers name and username will automatically populate when you type your full name. You can however edit them as you please. Fill in the details and then click continue.
You can opt to have Ubuntu automatically log in for you -- even with a password set -- or you can choose the traditional option requiring a password to log in. You really don't need to choose the encrypt option unless you're installing on a laptop and are dealing with highly secure information.

Step 11:
The installation information screens will now appear as Ubuntu continues the installation:

- Step 12:

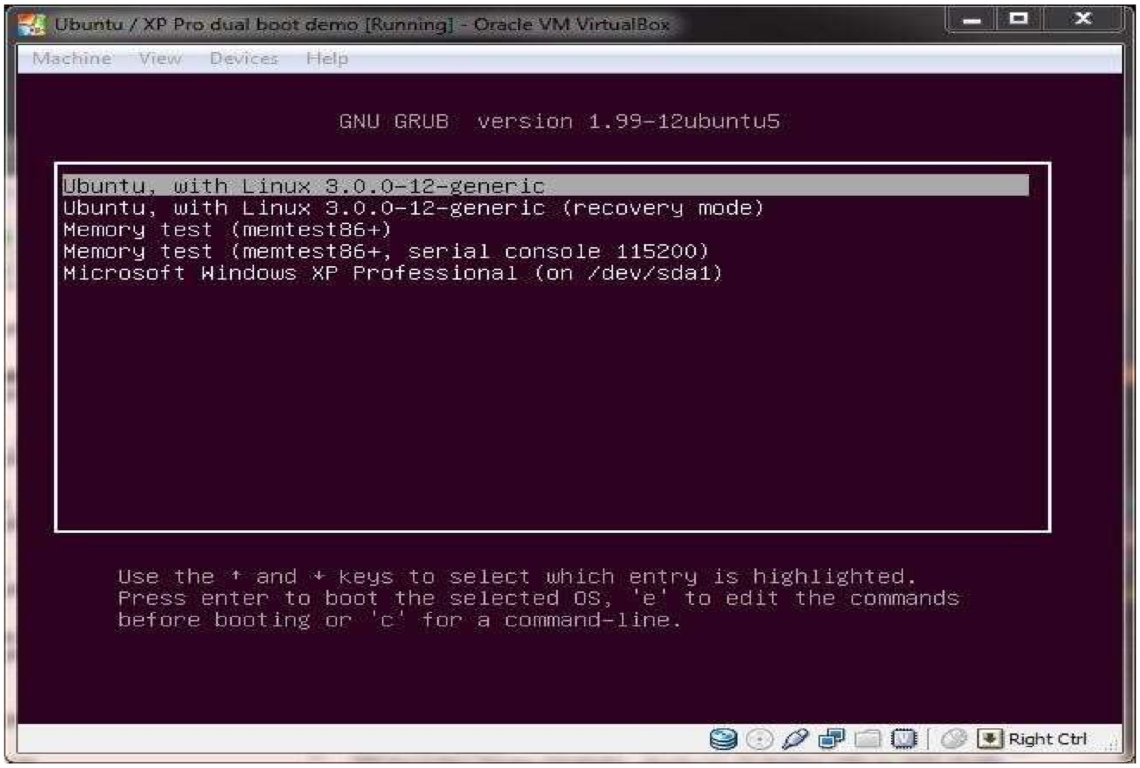
Once installation has finished, you will be presented with the following box:



Installation is complete. You need to restart the computer in order to use the new installation.

Select "restart now" and when requested, remove your installation CD, then press enter to reboot.

Step 13:
For those of you that have Ubuntu as the only OS the computer will boot directly into Linux. If you're dual-booting, you will see the GRUB menu appear similar to below:


Ubuntu / XP Pro dual boot demo [Running] - Oracle VM VirtualBox
Machine View Devices Help
Right Ctrl

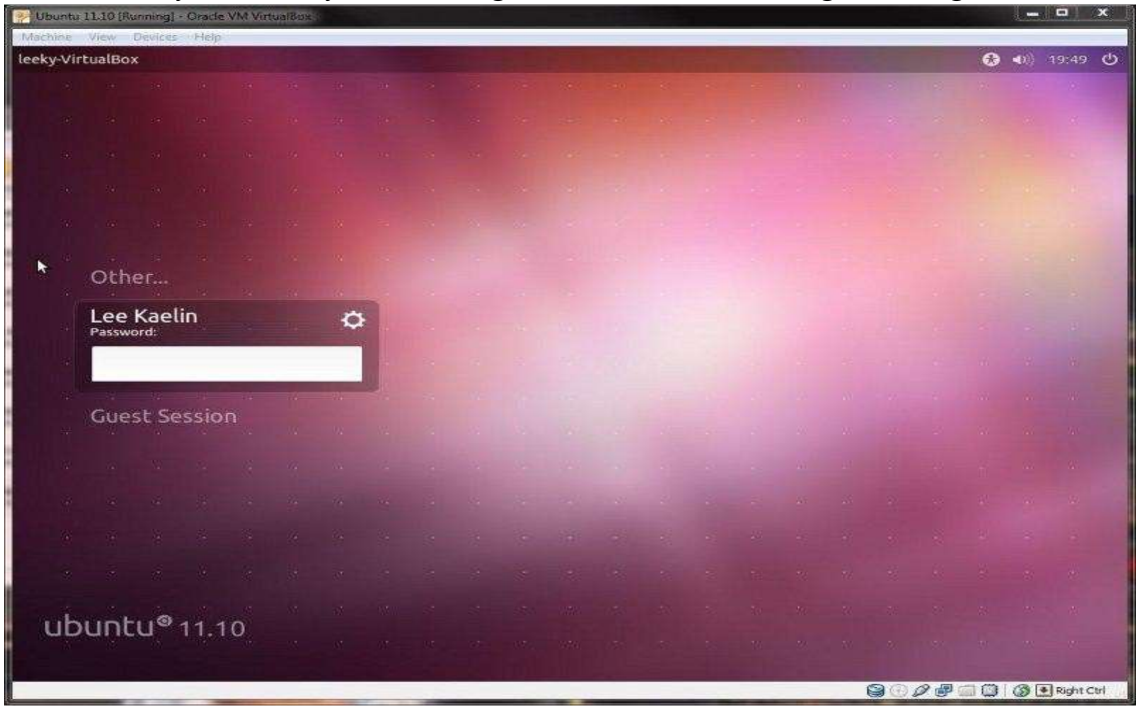
```
                 GNU GRUB  version 1.99-12ubuntu5

 Ubuntu, with Linux 3.0.0-12-generic
 Ubuntu, with Linux 3.0.0-12-generic (recovery mode)
 Memory test (memtest86+)
 Memory test (memtest86+, serial console 115200)
 Microsoft Windows XP Professional (on /dev/sda1)

    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.
```

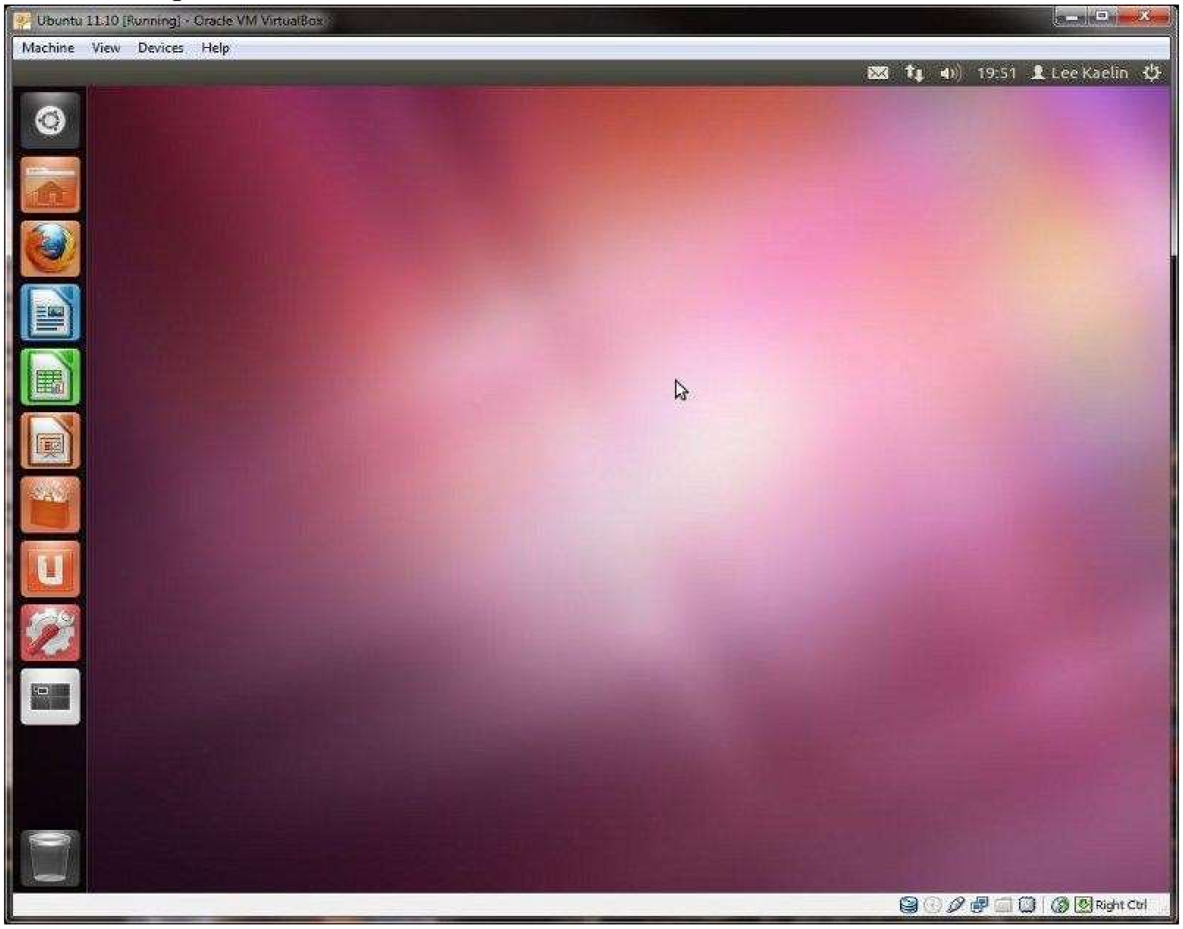
Hit enter, to select the first option and load your newly installed Ubuntu OS.

Step 14:
For those that elected to automatically log into Ubuntu, you will go straight to the desktop in Step 15. For everyone else, you will be greeted with the new login manager for Ubuntu:



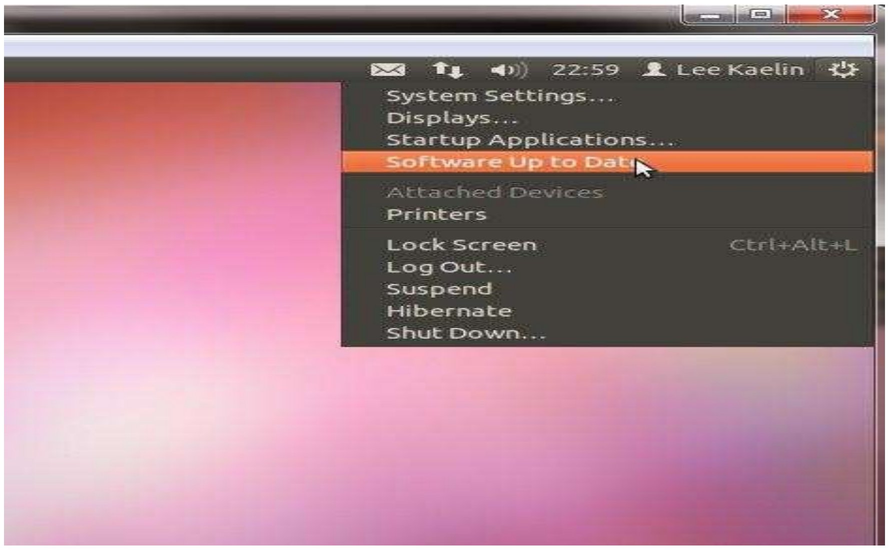
Enter your password, and hit enter to login to your desktop.

Step 15: Your desktop should look like this:



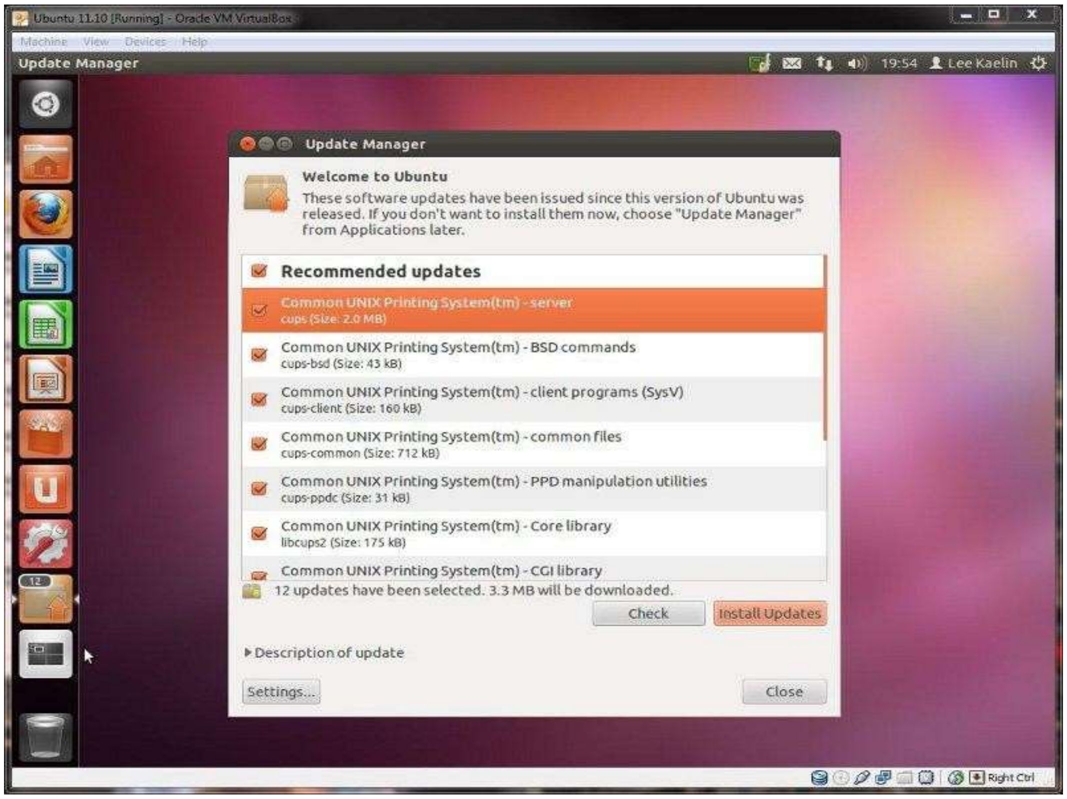
Step 16:

Before we proceed further, let's check for updates. Click on the power button on the top right corner of the screen and select "check for updates," or words to that effect.



I'd already run updates on this install so the example above is displaying "software up to date," but the picture highlights where you need to select anyway.

Upon selecting the update option, the update manager will appear, as below:



Update Manager

Welcome to Ubuntu

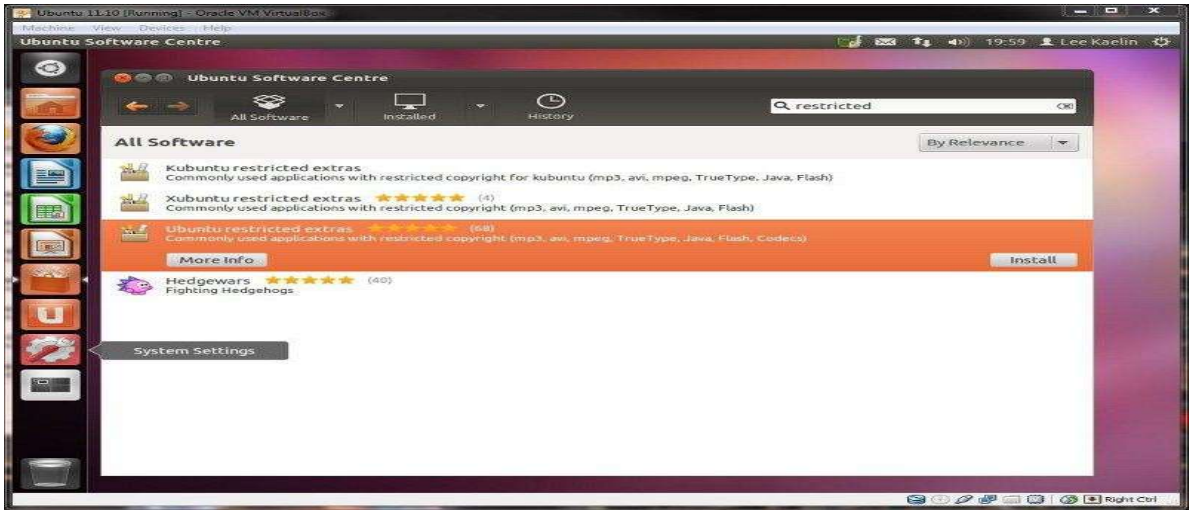
These software updates have been issued since this version of Ubuntu was released. If you don't want to install them now, choose "Update Manager" from Applications later.

Recommended updates

- Common UNIX Printing System(tm) - server
  cups (Size: 2.0 MB)
- Common UNIX Printing System(tm) - BSD commands
  cups-bsd (Size: 43 kB)
- Common UNIX Printing System(tm) - client programs (SysV)
  cups-client (Size: 160 kB)
- Common UNIX Printing System(tm) - common files
  cups-common (Size: 712 kB)
- Common UNIX Printing System(tm) - PPD manipulation utilities
  cups-ppdc (Size: 31 kB)
- Common UNIX Printing System(tm) - Core library
  libcups2 (Size: 175 kB)
- Common UNIX Printing System(tm) - CGI library

12 updates have been selected. 3.3 MB will be downloaded.

If it comes up with no available updates, just select "check" again to verify that it is correct. Having done the same thing myself, I was presented with the updates you see above. For those that opted to install updates during installation it is unlikely there will be further updates required.



Kubuntu restricted extras
Commonly used applications with restricted copyright for kubuntu (mp3, avi, mpeg, TrueType, Java, Flash)

Xubuntu restricted extras ★★★★★ (4)
Commonly used applications with restricted copyright (mp3, avi, mpeg, TrueType, Java, Flash)

Ubuntu restricted extras ★★★★★ (68)
Commonly used applications with restricted copyright (mp3, avi, mpeg, TrueType, Java, Flash, Codecs)

Hedgewars ★★★★★ (40)
Fighting Hedgehogs

You might be asked to enter your password to confirm changes. If prompted, enter your password and click OK. The same is true of any notifications that may appear during updating the OS.

Once complete select close, and restart Ubuntu. The power button is located on the top right corner of the screen. Click this and select shutdown.

Step 17:
No install is complete without full support for mp3s, core MS fonts, DVD playback codec‘s, Flash and Java, so let‘s go ahead and sort this now.

Click the black Ubuntu menu button at the top left corner of the screen and in the menu that appears type "software centre" and select the Ubuntu Software Centre. Once open, click the search bar, type "restricted" and the following should appear:

Select Ubuntu restricted extras, and then click on the install button.

Authentication is much like UAC (user access control) in windows Vista and 7. It is required to elevate your user privileges to that of root (Linux administrator).

Ubuntu restricted extras will now download, sort any dependencies and install. You can check its progress by viewing the progress bar above the install button. Once finished the In Progress tab will disappear -- restart Linux. It‘s not strictly necessary, but I always do it after installing this package so everything can start up properly.

Now you can enjoy your new OS and begin exploring its features.

1. Linux Startup Process (Boot Process)

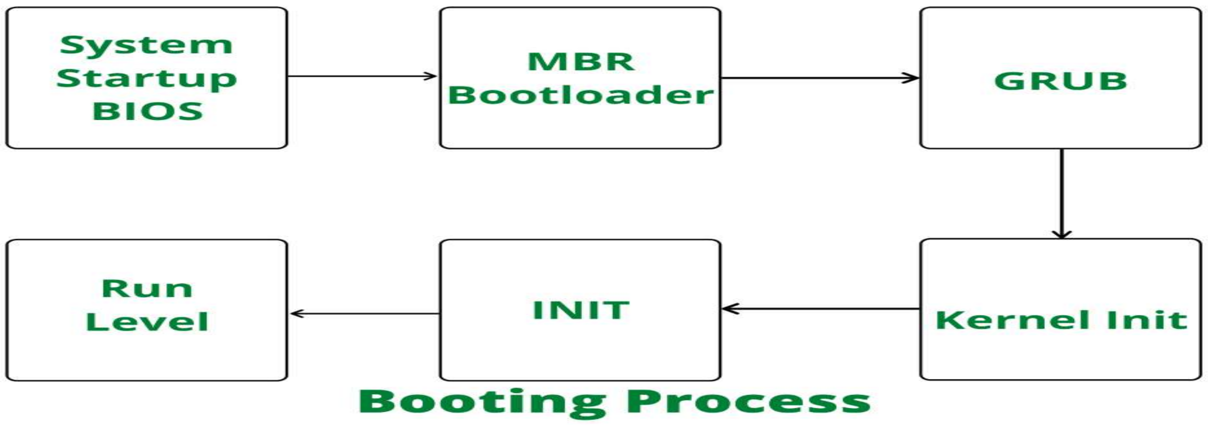
The **Linux boot process** is a multi-stage process that goes from hardware power-on to the login screen or terminal.

Many processes are running in the background when we press the system's power button. It is very important to learn the Linux boot process to understand the workings of any operating system. Knowing how the kernel boots is a must to solve the booting error.

Stages of Linux Boot Process:

Key Steps in the Linux Startup Proces:

1. The machine's BIOS (Basic Input/Output System) or boot microcode hundreds or **UEFI(Unified Extensible Firmware Interface)** initializes hardware and runs a Power-On Self-Test (POST) or a boot loader.
2. Boot loader finds the kernel image on the disk and loads it into memory, to start the system.
3. The kernel initializes the devices and their drivers.
4. The kernel mounts the basis filesystem.
5. The kernel starts a program referred to as init with a method ID zero
6. init sets the remainder of the system processes in motion.
7. For some purpose, init starts a method permitting you to log in, typically at the top or close to the top of the boot sequence.



Shutdown of Linux

The `**shutdown**` operation in Linux is a crucial command for managing system power states, allowing administrators to halt safely, power off, or reboot the system. This command provides flexibility in scheduling downtimes, ensuring minimal disruption to users and processes. In this article, we will explore the `shutdown` command with practical examples, illustrating how to use its various options to control system behavior effectively.

Syntax:

```
shutdown [OPTIONS] [TIME] [MESSAGE]
```

- **OPTIONS**: Various options to control the shutdown process, such as halting, powering off, or rebooting the system.
- **TIME**: Specifies when to perform the shutdown, either immediately, after a specified delay, or at a specific time.
- **MESSAGE**: An optional message that is broadcasted to all logged-in users, informing them of the scheduled shutdown

- `sudo shutdown now`

boot loaders of Linux

Boot Loader:

Boot Loader is a software program that is responsible for "**actually loading**" the operating system once the Boot Manager has finished its work. And by loading Operating System we mean **"loading the kernel of the Operating System".** The Boot Loader is typically a part of the Operating System itself. Till the point Boot Loader starts loading the OS, there is nothing in the Main Memory of the machine.

Following are a series of tasks that a typical Boot Loader is expected to perform:

1. Loading and parsing the Boot Configuration File.
2. Loading and initializing the OS's kernel into the main memory.
3. Loading and initializing the other system components and system drivers.
4. Finally, finish up the system environment setup and transfer the control to the kernel.

Some examples of Boot Loaders are :

1. LILO (Linux Loader): Most popular Boot Loader for Linux – Based Machines
2. Windows Boot Loader: Specific up to Windows Machines
3. GRub (Boot Loader part): It is a sub-part of GRub Boot Manager.

Linux Booting Process

1) LILO(Linux Loader) Configuration

The LILO configuration file is /etc/lilo.conf. The /sbin/lilo commands uses this file to determine what information to write to the MBR.

Warning Before editing /etc/lilo.conf, be sure to make a backup copy of the file. Also, have a working boot disk available so that changes can be made to the MBR if there is a problem. Refer To the man page for mkbootdisk for more information on creating a boot disk.

The /etc/lilo.conf file is used by the /sbin/lilo command to determine which operating system or kernel to load and where it should be installed.

2.9.1. Sample /etc/lilo.conf

The following is a sample /etc/lilo.conf for a system configured to boot two operating systems, Red Hat Enterprise Linux and DOS:

```
boot=/dev/hda
map=/boot/map
install=/boot/boot.b
prompt
timeout=50
message=/boot/message
lba32
default=linux

image=/boot/vmlinuz-2.4.0-0.43.6
        label=linux
        initrd=/boot/initrd-2.4.0-0.43.6.img
        read-only
        root=/dev/hda5

other=/dev/hda1
        label=dos
```

The following is a more detailed look at the lines of this file:

- boot=/dev/hda — Instructs LILO to be installed on the first hard disk of the first IDE controller.
- map=/boot/map — Locates the map file. In normal use, this should not be modified.
- install=/boot/boot.b — Instructs LILO to install the specified file as the new boot sector. In normal use, this should not be altered. If the install line is missing, LILO assumes a default of /boot/boot.b as the file to be used
- prompt — Instructs LILO to show you whatever is referenced in the message line. While it is not recommended that you remove the prompt line, if you do remove it, you can still access a prompt by holding down the [Shift] key while your machine starts to boot.
- timeout=50 — Sets the amount of time that LILO waits for user input before proceeding with booting the default line entry. This is measured in tenths of a second, with 50 as the default.

• message=/boot/message — Refers to the screen that LILO displays to let you select the operating system or kernel to boot.

• lba32 — Describes the hard disk geometry to LILO. Another common entry here is linear. You should not change this line unless you are very aware of what you are doing. Otherwise, you could put your system in an unbootable state.

• default=linux — Refers to the default operating system for LILO to boot as seen in the options listed below this line. The name linux refers to the label line below in each of the boot options.

• image=/boot/vmlinuz-2.4.0-0.43.6 — Specifies which Linux kernel to boot with this particular boot option.

• label=linux — Names the operating system option in the LILO screen. In this case, it is also the name referred to by the default line.

• initrd=/boot/initrd-2.4.0-0.43.6.img — Refers to the initial ram disk image that is used at boot time to initialize and start the devices that makes booting the kernel possible. The initial ram disk is a collection of machinespecific drivers necessary to operate a SCSI card, hard drive, or any other device needed to load the kernel. You should never try to share initial ram disks between machines.
• read-only — Specifies that the root partition (refer to the root line below) is read-only and cannot be altered during the boot process.
• root=/dev/hda5 — Specifies which disk partition to use as the root partition.
• other=/dev/hda1 — Specifies the partition containing DOS.

2) GRUB Configuration
GRUB is a boot loader designed to boot a wide range of operating systems from a wide range of file systems. GRUB is becoming popular due to the increasing number of possible root file systems that can Linux can reside upon.

GRUB is documented in a GNU info file. Type info grub to view the documentation.

The GRUB configuration file is /boot/grub/menu.lst. Some distributions use another configuration file; for example, Red Hat Linux uses the file /boot/grub/grub.conf.

GRUB configuration files are interpreted. Syntax errors will not be detected until the machine is rebooted, so take care not to make typing errors.

Edit the GRUB configuration file and remove any splash image entries. If these entries are not removed GRUB 0.90 behaves very oddly, transferring control between the serial console and the attached monitor and keyboard

If there is not already a password command in the GRUB configuration file then create a hashed password, see Figure 4-4. The password should be good, as it can be used to gain root access.

Figure 4-4. Using md5crypt to create a hashed password for GRUB

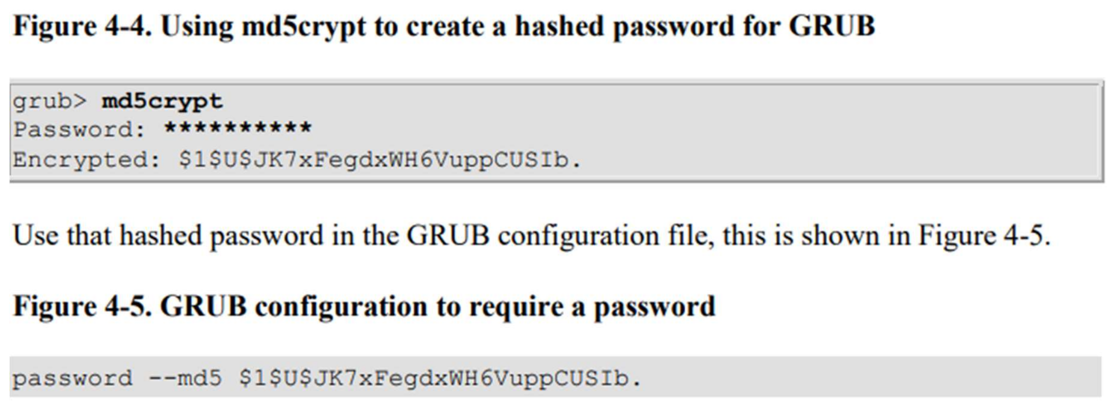


Figure 4-4. Using md5crypt to create a hashed password for GRUB

```
grub> md5crypt
Password: **********
Encrypted: $1$U$JK7xFegdxWH6VuppCUSIb.
```

Use that hashed password in the GRUB configuration file, this is shown in Figure 4-5.

Figure 4-5. GRUB configuration to require a password

```
password --md5 $1$U$JK7xFegdxWH6VuppCUSIb.
```

Define the serial port and configure GRUB to use the serial port, as shown in Figure 4-6.

Figure 4-6. GRUB configuration for serial console

```
serial --unit=0 --speed=9600 --word=8 --parity=no --stop=1
terminal serial
```

`--unit` is the number of the serial port, counting from zero, unit 0 being `COM1`.

Note that the values of `--parity` are spelt out in full: `no`, `even` and `odd`. The common abbreviations `n`, `e` and `o` are *not* accepted.

If there is mysteriously no output on the serial port then suspect a syntax error in the **serial** or **terminal** commands.

If you also want to use and attached monitor and keyboard as well as the serial port to control the GRUB boot loader then use the alternative configuration in Figure 4-7.

Figure 4-7. GRUB configuration for serial console and attached monitor and keyboard console

```
password --md5 $1$U$JK7xFegdxWH6VuppCUSIb.
serial --unit=0 --speed=9600 --word=8 --parity=no --stop=1
terminal --timeout=10 serial console
```

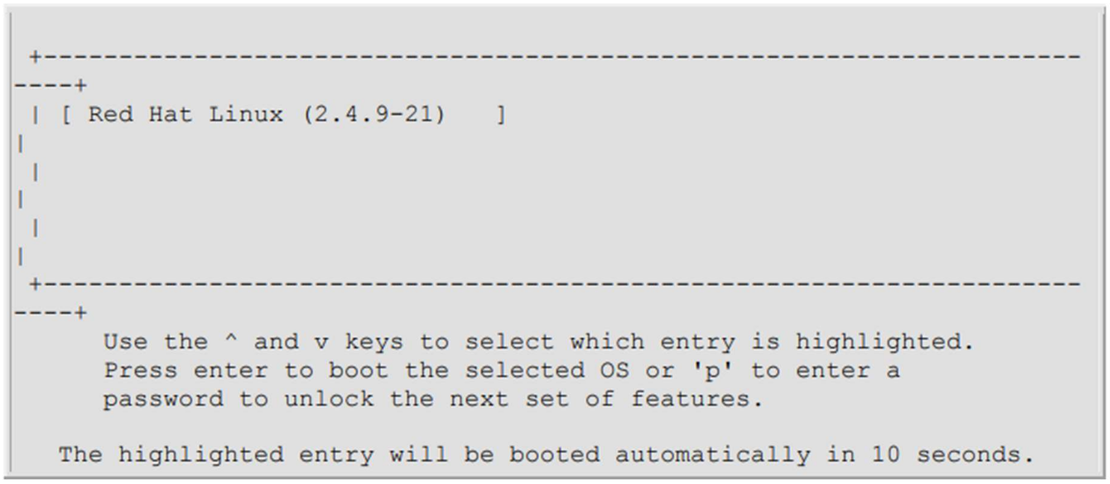
When both the serial port and the attached monitor and keyboard are configured they will both ask for a key to be pressed until the timeout expires. If a key is pressed then the boot menu is displayed to that device. Disconcertingly, the other device sees nothing.

If no key is pressed then the boot menu is displayed on the whichever of `serial` or `console` is listed first in the **terminal** command. After the timeout set by the **timeout** the default option set by **default** is booted.

Figure 4-8. GRUB output to default device when configured for serial and attached monitor output

```
Press any key to continue.
Press any key to continue.
Press any key to continue.
Press any key to continue.
Press any key to continue.
Press any key to continue.
Press any key to continue.
Press any key to continue.
Press any key to continue.
Press any key to continue.

    GRUB  version 0.90  (639K lower / 162752K upper memory)
```



```
 +-------------------------------------------------------------------
----+
 | [ Red Hat Linux (2.4.9-21)   ]
|
 |
|
 |
|
 +-------------------------------------------------------------------
----+
      Use the ^ and v keys to select which entry is highlighted.
      Press enter to boot the selected OS or 'p' to enter a
      password to unlock the next set of features.

   The highlighted entry will be booted automatically in 10 seconds.
```

Creating Linux User Account and Password

To create a new user account and set a password on a Linux system, you can follow these steps. This process requires **root** privileges or a user with **sudo** access.

1. **Create a New User**
   sudo adduser username

2. **Set a Password for the User**
   sudo passwd username

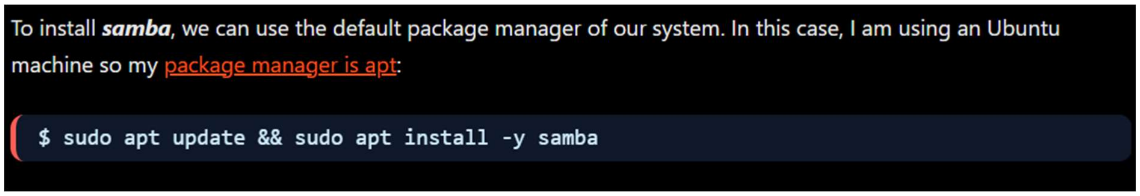
What is Samba?

Samba is the standard Windows interoperability suite of programs for Linux and Unix.

Since 1992, Samba has provided secure, stable and fast file and print services for all clients using the SMB/CIFS protocol, such as all versions of DOS and Windows, OS/2, Linux and many others.

Samba is an important component to seamlessly integrate Linux/Unix Servers and Desktops into Active Directory environments. It can function both as a domain controller or as a regular domain member.

Samba is a software package that gives network administrators flexibility and freedom in terms of setup, configuration, and choice of systems and equipment. Because of all that it offers, Samba has grown in popularity, and continues to do so, every year since its release in 1992.



To install ***samba***, we can use the default package manager of our system. In this case, I am using an Ubuntu machine so my package manager is apt:

```
$ sudo apt update && sudo apt install -y samba
```

How to Manage a Samba Server (Step by Step)

1. **Start / Stop / Restart Samba Services**
   These commands keep the Samba server running:

```
sudo systemctl start smbd     # Start the service
sudo systemctl stop smbd      # Stop the service
sudo systemctl restart smbd   # Restart (after changes)
sudo systemctl enable smbd    # Start at boot
sudo systemctl status smbd    # Check if it's running
```

Main config file: /etc/samba/smb.conf

Add a Samba user: sudo smbpasswd -a user1

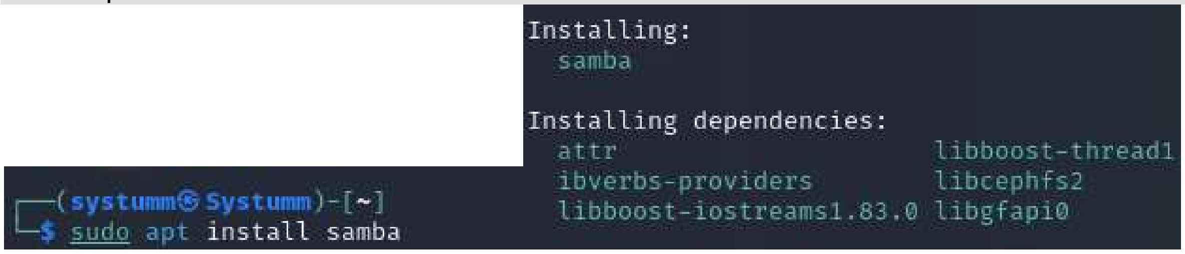
The user must also exist on the system. You can add them like this: sudo adduser user1

Remove a Samba user: sudo smbpasswd -x user1

Steps to install Samba in Ubuntu

Step 1: Enter the following command to install samba.

```
sudo apt install samba
```



```
┌──(systumm㉿Systumm)-[~]
└─$ sudo apt install samba
```

```
Installing:
  samba

Installing dependencies:
  attr                      libboost-thread1
  ibverbs-providers         libcephfs2
  libboost-iostreams1.83.0  libgfapi0
```

Step 2: Setting up Samba Now that Samba is installed, we need to create a directory for it to share:
Enter the following command to do so.

```
mkdir /home/<username>/sambashare/
```

It will create a directory as samba share inside the home directory of the user <username>. This is the directory in which files and folders that you would like to share through Samba will reside.
Note: In the current draft of the script, the <username> must be substituted with the actual username of the user who owns the home directory.

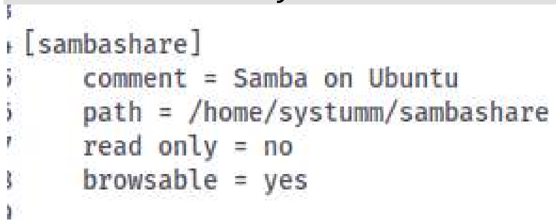
```
┌──(systumm㉿Systumm)-[~]
└─$ mkdir /home/systumm/sambashare/
```

Step 3: Now we have to edit the config file of samba. The configuration file for Samba is located at /etc/samba/smb.conf. To add the new directory as a share, we edit the file by running:

```
sudo nano /etc/samba/smb.conf
```

At the bottom of the file, add the following lines and save it.

```
[sambashare]
    comment = Samba on Ubuntu
    path = /home/username/sambashare
    read only = no
    browsable = yes
```

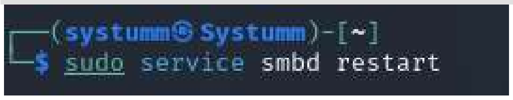


```
[sambashare]
    comment = Samba on Ubuntu
    path = /home/systumm/sambashare
    read only = no
    browsable = yes
```

This configuration section specifies that, sambashare which is located at /home/username/sambashare, should not be read-only and can be browsed. This means that users can mount the share, view its contents and write into it using a Samba client.

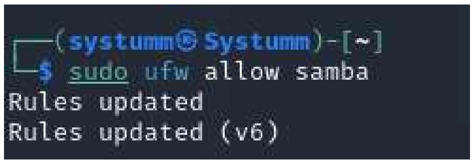
Step 4: After saving the config file, restart the samba service.

```
sudo service smbd restart
```



```
┌──(systumm㉿Systumm)-[~]
└─$ sudo service smbd restart
```

Since there may be a firewall which can block samba traffic due to security reasons. You can add firewall rules to allow samba traffic. Enter the following command to do so.
sudo ufw allow samba

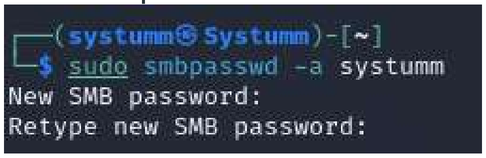


```
┌──(systumm㉿Systumm)-[~]
└─$ sudo ufw allow samba
Rules updated
Rules updated (v6)
```

Step 5: Setting up User Accounts and Connecting to Share
As Samba doesn't use the system account password, we need to set up a Samba password for our user account.
We can do this by entering the following command:

```
sudo smbpasswd -a username
```

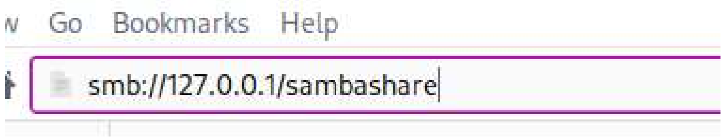
Note: Replace username with the username of your system account.



```
┌──(systumm㉿Systumm)-[~]
└─$ sudo smbpasswd -a systumm
New SMB password:
Retype new SMB password:
```

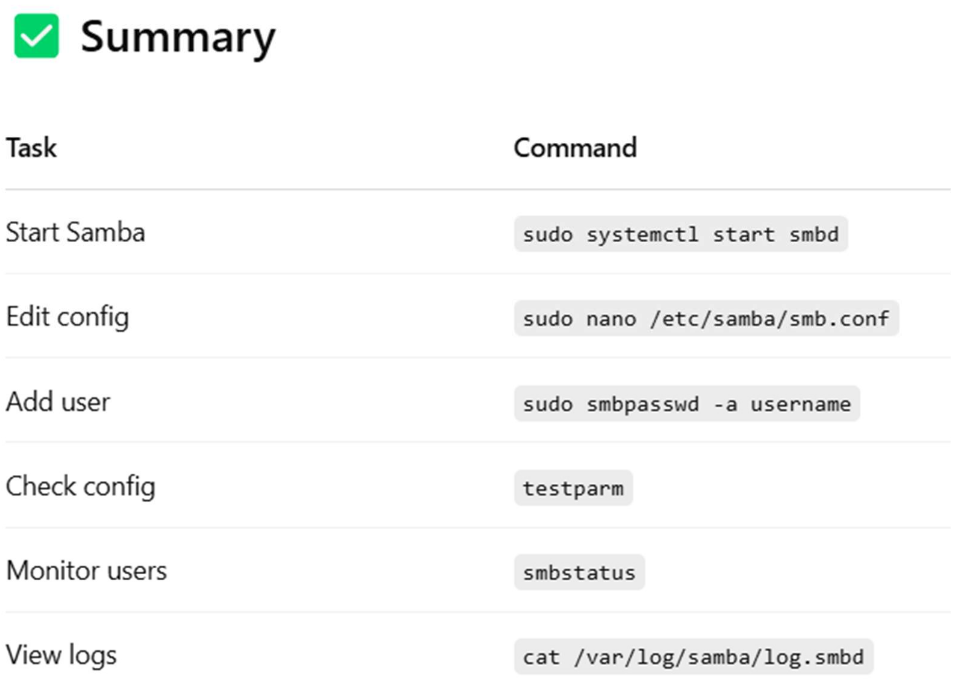
Step 6: Connecting to Share,Open the default file manager and type the following into the search bar.

```
smb://ip-address/sambashare
```



```
smb://127.0.0.1/sambashare
```

Press enter to execute the command, this will prompt you to enter your credentials.



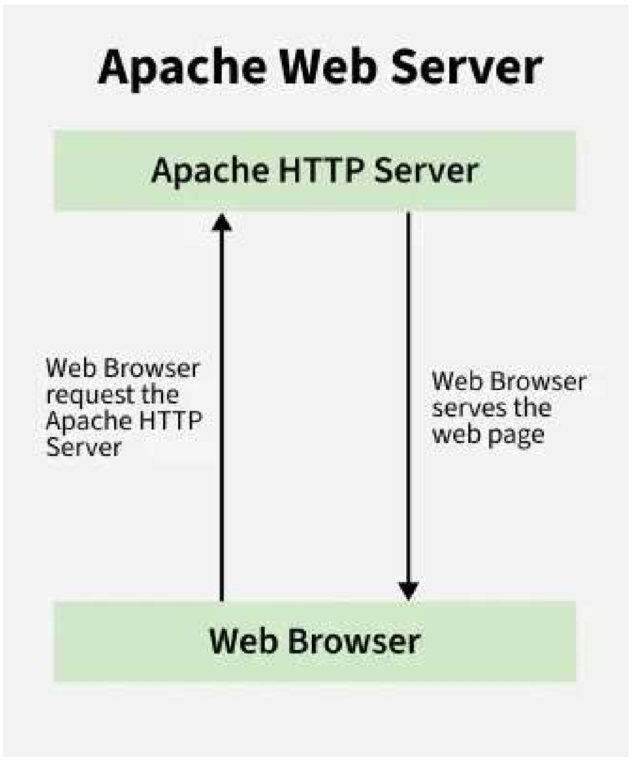
✅ Summary

Task	Command
Start Samba	`sudo systemctl start smbd`
Edit config	`sudo nano /etc/samba/smb.conf`
Add user	`sudo smbpasswd -a username`
Check config	`testparm`
Monitor users	`smbstatus`
View logs	`cat /var/log/samba/log.smbd`

Installing and Managing Apache Server

Apache HTTP Server (commonly just called **Apache**) is **free, open-source software** that runs on a web server and delivers web content to users over the Internet.

Apache HTTP Server is an open-source and free web server that is written by the Apache Software Foundation (ASF). It is among the most widely used web servers, serving millions of websites on various platforms such as Linux, Windows, and macOS, and forms an integral part of the LAMP stack (Linux, Apache, MySQL, PHP), which is commonly utilized for web hosting and development.


Apache Web Server
Apache HTTP Server
Web Browser request the Apache HTTP Server
Web Browser serves the web page
Web Browser

Uses of Apache Server are:

- It supports cross-platform support as it runs on Linux, Windows, and macOS
- Modular design (Extension support through modules.)
- Dynamic content support (PHP, Python, Perl, etc.)
- Virtual hosting (Hosts many sites in one server)
- Security support (Supports SSL/TLS, authentication, and access control)
- Customizable settings (Makes use of .htaccess files)

Step 1 — Installing Apache

Apache is available within Ubuntu's default software repositories, making it possible to install it using conventional package management tools.

Let's begin by updating the local package index to reflect the latest upstream changes:

```
sudo apt update

```

Copy

Then, install the `apache2` package:

```
sudo apt install apache2

```

Copy

After confirming the installation, `apt` will install Apache and all required dependencies.

Step 2 — Adjusting the Firewall

Before testing Apache, it's necessary to modify the firewall settings to allow outside access to the default web ports. Assuming that you followed the instructions in the prerequisites, you should have a UFW firewall configured to restrict access to your server.

During installation, Apache registers itself with UFW to provide a few application profiles that can be used to enable or disable access to Apache through the firewall.

List the `ufw` application profiles by typing:

```
sudo ufw app list

```

Copy

You will receive a list of the application profiles:

```
Output
Available applications:
  Apache
  Apache Full
  Apache Secure
  OpenSSH
```

As indicated by the output, there are three profiles available for Apache:

- **Apache**: This profile opens only port 80 (normal, unencrypted web traffic)
- **Apache Full**: This profile opens both port 80 (normal, unencrypted web traffic) and port 443 (TLS/SSL encrypted traffic)
- **Apache Secure**: This profile opens only port 443 (TLS/SSL encrypted traffic)

It is recommended that you enable the most restrictive profile that will still allow the traffic you've configured. Since we haven't configured SSL for our server yet in this guide, we will only need to allow traffic on port 80:

```
sudo ufw allow 'Apache'

```

Copy

You can verify the change by typing:

```
sudo ufw status

```

Copy

The output will provide a list of allowed HTTP traffic:

```
Output
Status: active

To                         Action      From
--                         ------      ----
OpenSSH                    ALLOW       Anywhere
Apache                     ALLOW       Anywhere
OpenSSH (v6)               ALLOW       Anywhere (v6)
Apache (v6)                ALLOW       Anywhere (v6)
```

As indicated by the output, the profile has been activated to allow access to the Apache web server.

Step 3 — Checking your Web Server

At the end of the installation process, Ubuntu 20.04 starts Apache. The web server should already be up and running.

Check with the `systemd` init system to make sure the service is running by typing:

```
sudo systemctl status apache2
```

Copy

```
Output
● apache2.service - The Apache HTTP Server
     Loaded: loaded (/lib/systemd/system/apache2.service; enabled; vendor preset:
enabled)
     Active: active (running) since Thu 2020-04-23 22:36:30 UTC; 20h ago
       Docs: https://httpd.apache.org/docs/2.4/
   Main PID: 29435 (apache2)
      Tasks: 55 (limit: 1137)
     Memory: 8.0M
     CGroup: /system.slice/apache2.service
             ├─29435 /usr/sbin/apache2 -k start
             ├─29437 /usr/sbin/apache2 -k start
             └─29438 /usr/sbin/apache2 -k start
```

As confirmed by this output, the service has started successfully. However, the best way to test this is to request a page from Apache.

You can access the default Apache landing page to confirm that the software is running properly through your IP address. If you do not know your server's IP address, you can get it a few different ways from the command line.

Try typing this at your server's command prompt:

```
hostname -I
```

Copy

You will get back a few addresses separated by spaces. You can try each in your web browser to determine if they work.

Another option is to use the Icanhazip tool, which should give you your public IP address as read from another location on the internet:

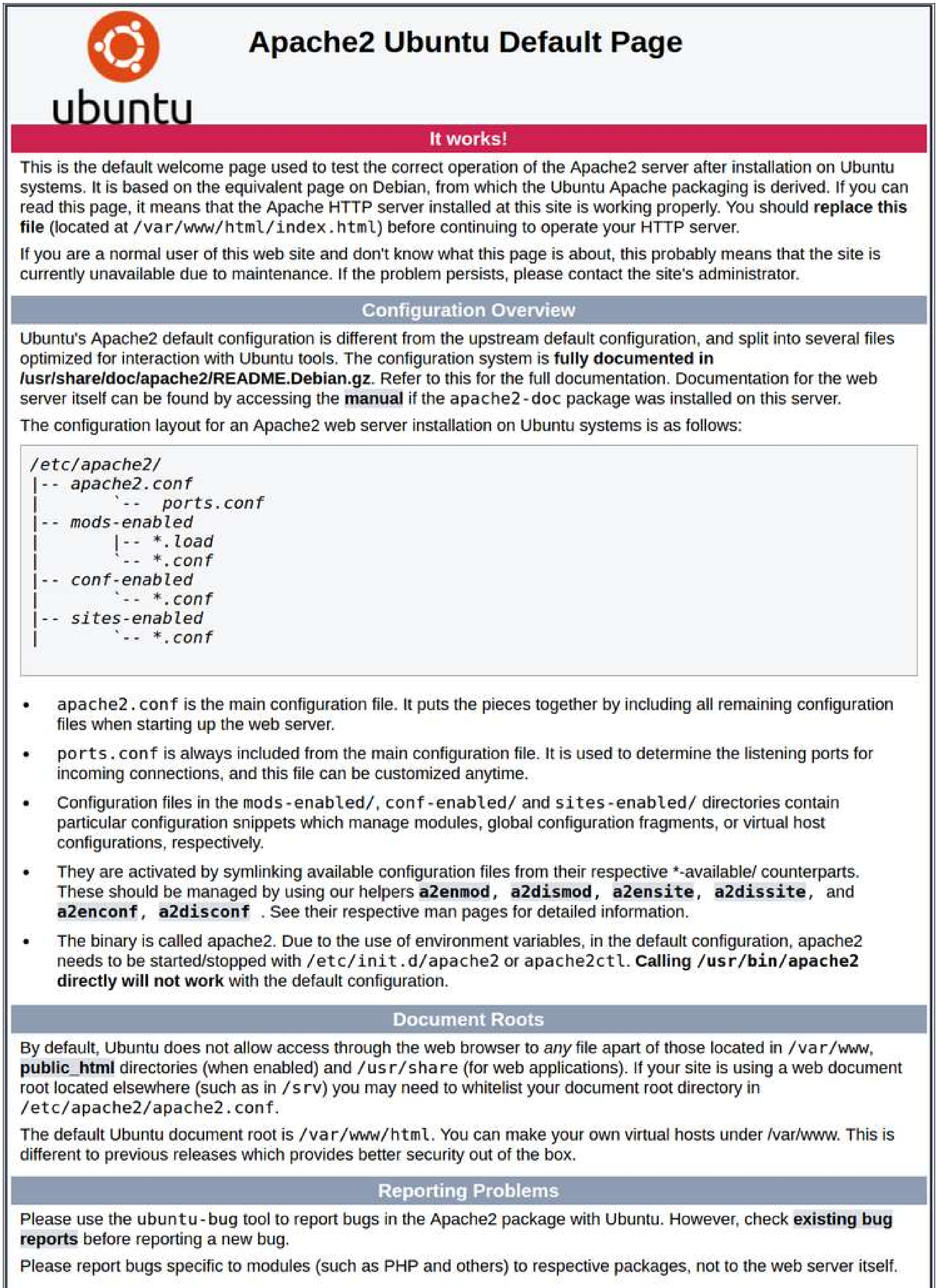
```
curl -4 icanhazip.com
```

Copy

When you have your server's IP address, enter it into your browser's address bar:

```
http://your_server_ip
```

You should see the default Ubuntu 20.04 Apache web page:



Apache2 Ubuntu Default Page

ubuntu

It works!

This is the default welcome page used to test the correct operation of the Apache2 server after installation on Ubuntu systems. It is based on the equivalent page on Debian, from which the Ubuntu Apache packaging is derived. If you can read this page, it means that the Apache HTTP server installed at this site is working properly. You should **replace this file** (located at `/var/www/html/index.html`) before continuing to operate your HTTP server.

If you are a normal user of this web site and don't know what this page is about, this probably means that the site is currently unavailable due to maintenance. If the problem persists, please contact the site's administrator.

Configuration Overview

Ubuntu's Apache2 default configuration is different from the upstream default configuration, and split into several files optimized for interaction with Ubuntu tools. The configuration system is **fully documented in /usr/share/doc/apache2/README.Debian.gz**. Refer to this for the full documentation. Documentation for the web server itself can be found by accessing the **manual** if the `apache2-doc` package was installed on this server.

The configuration layout for an Apache2 web server installation on Ubuntu systems is as follows:

```
/etc/apache2/
|-- apache2.conf
|       `--  ports.conf
|-- mods-enabled
|       |-- *.load
|       `-- *.conf
|-- conf-enabled
|       `-- *.conf
|-- sites-enabled
|       `-- *.conf
```

- `apache2.conf` is the main configuration file. It puts the pieces together by including all remaining configuration files when starting up the web server.
- `ports.conf` is always included from the main configuration file. It is used to determine the listening ports for incoming connections, and this file can be customized anytime.
- Configuration files in the `mods-enabled/`, `conf-enabled/` and `sites-enabled/` directories contain particular configuration snippets which manage modules, global configuration fragments, or virtual host configurations, respectively.
- They are activated by symlinking available configuration files from their respective *-available/ counterparts. These should be managed by using our helpers **`a2enmod`**, **`a2dismod`**, **`a2ensite`**, **`a2dissite`**, and **`a2enconf`**, **`a2disconf`** . See their respective man pages for detailed information.
- The binary is called apache2. Due to the use of environment variables, in the default configuration, apache2 needs to be started/stopped with `/etc/init.d/apache2` or `apache2ctl`. **Calling `/usr/bin/apache2` directly will not work** with the default configuration.

Document Roots

By default, Ubuntu does not allow access through the web browser to *any* file apart of those located in `/var/www`, **public_html** directories (when enabled) and `/usr/share` (for web applications). If your site is using a web document root located elsewhere (such as in `/srv`) you may need to whitelist your document root directory in `/etc/apache2/apache2.conf`.

The default Ubuntu document root is `/var/www/html`. You can make your own virtual hosts under /var/www. This is different to previous releases which provides better security out of the box.

Reporting Problems

Please use the `ubuntu-bug` tool to report bugs in the Apache2 package with Ubuntu. However, check **existing bug reports** before reporting a new bug.

Please report bugs specific to modules (such as PHP and others) to respective packages, not to the web server itself.

This page indicates that Apache is working correctly. It also includes some basic information about important Apache files and directory locations.

Step 4 — Managing the Apache Process

Now that you have your web server up and running, let's go over some basic management commands using `systemctl`.

To stop your web server, type:

```
sudo systemctl stop apache2

```

Copy

To start the web server when it is stopped, type:

```
sudo systemctl start apache2

```

Copy

To stop and then start the service again, type:

```
sudo systemctl restart apache2

```

Copy

If you are simply making configuration changes, Apache can often reload without dropping connections. To do this, use this command:

```
sudo systemctl reload apache2

```

Copy

By default, Apache is configured to start automatically when the server boots. If this is not what you want, disable this behavior by typing:

```
sudo systemctl disable apache2

```

Copy

To re-enable the service to start up at boot, type:

```
sudo systemctl enable apache2

```

Copy

Apache should now start automatically when the server boots again.

Step 5 — Setting Up Virtual Hosts (Recommended)

When using the Apache web server, you can use *virtual hosts* (similar to server blocks in Nginx) to encapsulate configuration details and host more than one domain from a single server. We will set up a domain called **your_domain**, but you should **replace this with your own domain name**. If you are setting up a domain name with DigitalOcean, please refer to our Networking Documentation.

Apache on Ubuntu 20.04 has one server block enabled by default that is configured to serve documents from the `/var/www/html` directory. While this works well for a single site, it can become unwieldy if you are hosting multiple sites. Instead of modifying `/var/www/html`, let's create a directory structure within `/var/www` for a **your_domain** site, leaving `/var/www/html` in place as the default directory to be served if a client request doesn't match any other sites.
Create the directory for **your_domain** as follows:

```
sudo mkdir /var/www/your_domain
```
Copy

Next, assign ownership of the directory with the `$USER` environment variable:

```
sudo chown -R $USER:$USER /var/www/your_domain
```
Copy

The permissions of your web roots should be correct if you haven't modified your umask value, which sets default file permissions. To ensure that your permissions are correct and allow the owner to read, write, and execute the files while granting only read and execute permissions to groups and others, you can input the following command:

```
sudo chmod -R 755 /var/www/your_domain
```
Copy

Next, create a sample `index.html` page using `nano` or your favorite editor:

```
sudo nano /var/www/your_domain/index.html
```
Copy

Inside, add the following sample HTML:

/var/www/your_domain/index.html

```
<html>
    <head>
        <title>Welcome to Your_domain!</title>
    </head>
    <body>
        <h1>Success!  The your_domain virtual host is working!</h1>
    </body>
</html>
```
Copy

Save and close the file when you are finished.

In order for Apache to serve this content, it's necessary to create a virtual host file with the correct directives. Instead of modifying the default configuration file located at `/etc/apache2/sites-available/000-default.conf` directly, let's make a new one at `/etc/apache2/sites-available/your_domain.conf`:

```
sudo nano /etc/apache2/sites-available/your_domain.conf
```
Copy

Paste in the following configuration block, which is similar to the default, but updated for our new directory and domain name:

/etc/apache2/sites-available/your_domain.conf

```
<VirtualHost *:80>
    ServerAdmin webmaster@localhost
    ServerName your_domain
    ServerAlias www.your_domain
    DocumentRoot /var/www/your_domain
    ErrorLog ${APACHE_LOG_DIR}/error.log
    CustomLog ${APACHE_LOG_DIR}/access.log combined
</VirtualHost>
```

Copy

Notice that we've updated the `DocumentRoot` to our new directory and `ServerAdmin` to an email that the **your_domain** site administrator can access. We've also added two directives: `ServerName`, which establishes the base domain that should match for this virtual host definition, and `ServerAlias`, which defines further names that should match as if they were the base name.

Save and close the file when you are finished.

Let's enable the file with the `a2ensite` tool:

```
sudo a2ensite your_domain.conf
```

Copy

Disable the default site defined in `000-default.conf`:

```
sudo a2dissite 000-default.conf
```

Copy

Next, let's test for configuration errors:

```
sudo apache2ctl configtest
```

Copy

You should receive the following output:

```
Output
Syntax OK
```

Restart Apache to implement your changes:

```
sudo systemctl restart apache2
```

Copy

Apache should now be serving your domain name. You can test this by navigating to `http://your_domain`, where you should see something like this:

Success! The your_domain virtual host is workin

Step 6 – Getting Familiar with Important Apache Files and Directories

Now that you know how to manage the Apache service itself, you should take a few minutes to familiarize yourself with a few important directories and files.

Content

- `/var/www/html`: The actual web content, which by default only consists of the default Apache page you saw earlier, is served out of the `/var/www/html` directory. This can be changed by altering Apache configuration files.

Server Configuration

- `/etc/apache2`: The Apache configuration directory. All of the Apache configuration files reside here.
- `/etc/apache2/apache2.conf`: The main Apache configuration file. This can be modified to make changes to the Apache global configuration. This file is responsible for loading many of the other files in the configuration directory.
- `/etc/apache2/ports.conf`: This file specifies the ports that Apache will listen on. By default, Apache listens on port 80 and additionally listens on port 443 when a module providing SSL capabilities is enabled.
- `/etc/apache2/sites-available/`: The directory where per-site virtual hosts can be stored. Apache will not use the configuration files found in this directory unless they are linked to the `sites-enabled` directory. Typically, all server block configuration is done in this directory, and then enabled by linking to the other directory with the `a2ensite` command.
- `/etc/apache2/sites-enabled/`: The directory where enabled per-site virtual hosts are stored. Typically, these are created by linking to configuration files found in the `sites-available` directory with the `a2ensite`. Apache reads the configuration files and links found in this directory when it starts or reloads to compile a complete configuration.
- `/etc/apache2/conf-available/`, `/etc/apache2/conf-enabled/`: These directories have the same relationship as the `sites-available` and `sites-enabled` directories, but are used to store configuration fragments that do not belong in a virtual host. Files in the `conf-available` directory can be enabled with the `a2enconf` command and disabled with the `a2disconf` command.
- `/etc/apache2/mods-available/`, `/etc/apache2/mods-enabled/`: These directories contain the available and enabled modules, respectively. Files ending in `.load` contain fragments to load specific modules, while files ending in `.conf` contain the configuration for those modules. Modules can be enabled and disabled using the `a2enmod` and `a2dismod` command.

Configure Ubuntu's Built-In Firewall

Ubuntu includes its own firewall, known as ufw -- short for "uncomplicated firewall." Ufw is an easier-to-use frontend for the standard Linux iptables commands. You can even control ufw from a graphical interface.

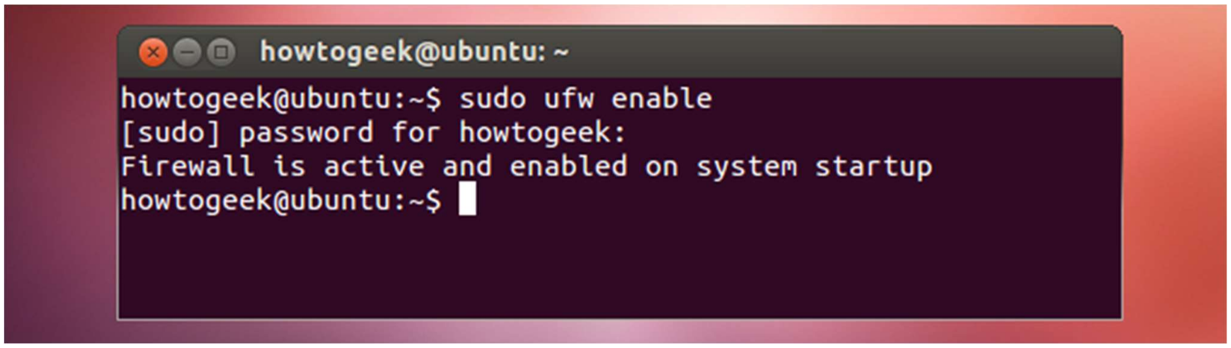
Ubuntu's firewall is designed as an easy way to perform basic firewall tasks without learning iptables. It doesn't offer all the power of the standard iptables commands, but it's less complex.

Terminal Usage

The firewall is disabled by default. To enable the firewall, run the following command from a terminal:

```
sudo ufw enable
```

You don't necessarily have to enable the firewall first. You can add rules while the firewall is offline, and then enable it after you're done configuring it.



```
howtogeek@ubuntu: ~
howtogeek@ubuntu:~$ sudo ufw enable
[sudo] password for howtogeek:
Firewall is active and enabled on system startup
howtogeek@ubuntu:~$
```

Working With Rules

Let's say you want to allow SSH traffic on port 22. To do so, you can run one of several commands:

sudo ufw allow 22 (Allows both TCP and UDP traffic -- not ideal if UDP isn't necessary.)

sudo ufw allow 22/tcp (Allows only TCP traffic on this port.)

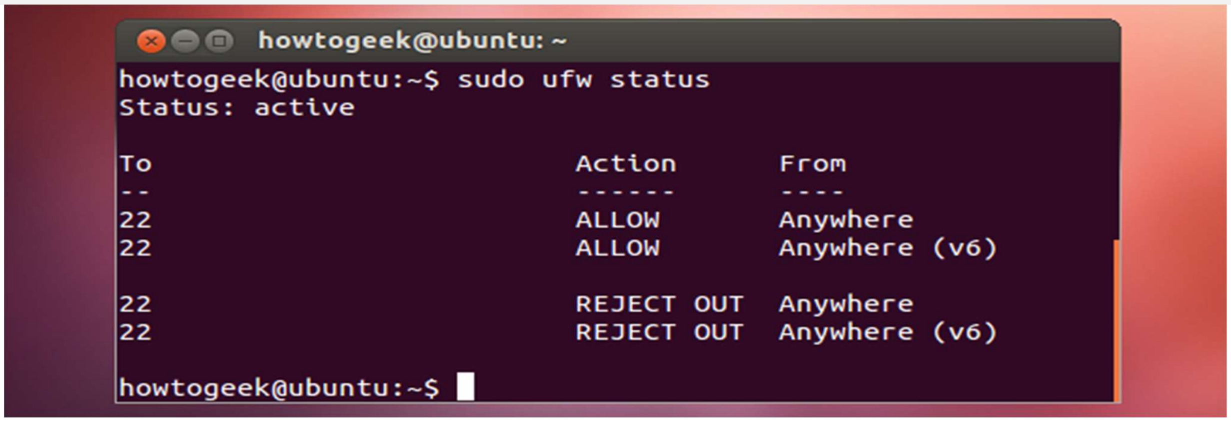
sudo ufw allow ssh (Checks the /etc/services file on your system for the port that SSH requires and allows it. Many common services are listed in this file.)

Ufw assumes you want to set the rule for incoming traffic, but you can also specify a direction. For example, to block outgoing SSH traffic, run the following command:

sudo ufw reject out ssh

You can view the rules you've created with the following command:

sudo ufw status



```
howtogeek@ubuntu: ~
howtogeek@ubuntu:~$ sudo ufw status
Status: active

To                         Action      From
--                         ------      ----
22                         ALLOW       Anywhere
22                         ALLOW       Anywhere (v6)

22                         REJECT OUT  Anywhere
22                         REJECT OUT  Anywhere (v6)

howtogeek@ubuntu:~$
```

To delete a rule, add the word delete before the rule. For example, to stop rejecting outgoing ssh traffic, run the following command:

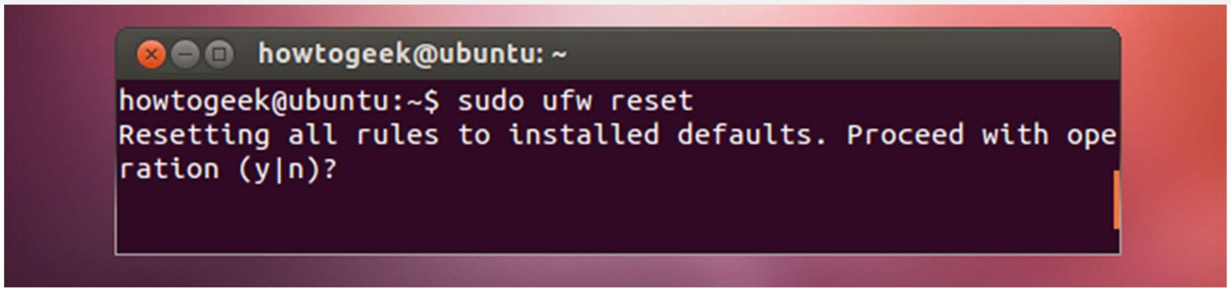
```
sudo ufw delete reject out ssh
```

Ufw's syntax allows for fairly complex rules. For example, this rule denies TCP traffic from the IP 12.34.56.78 to port 22 on the local system:

```
sudo ufw deny proto tcp from 12.34.56.78 to any port 22
```

To reset the firewall to its default state, run the following command:

```
sudo ufw reset
```



```
howtogeek@ubuntu:~$ sudo ufw reset
Resetting all rules to installed defaults. Proceed with ope
ration (y|n)?
```

Working with WINE

What is Wine?

Wine is a program that allows you to run Windows software on your Linux computer. Linux is a different kind of operating system than Windows. Normally, programs made for Windows cannot work directly on Linux. Wine acts like a translator between the Windows program and your Linux system. It tricks the Windows program into thinking it is running on a Windows computer. This way, you can install and use many Windows programs and games right on your Linux machine, without needing a Windows operating system. Wine is free and open-source software. It makes it possible to use Windows applications on Linux without having to pay for a Windows license.

Uses of Wine

- **Run Windows programs on Linux:** The primary use of Wine is to allow you to run software designed for Windows on your Linux operating system. This lets you access Windows-only programs without having to actually install Windows.
- **Use favorite Windows apps:** With Wine, you can keep using your preferred Windows applications and games after switching to Linux. No need to find Linux alternatives.
- **No Windows license needed:** Wine enables running Windows programs on Linux without purchasing a Windows operating system license. This saves money.

- **Test Windows software:** Developers can use Wine to test how their Windows programs run on Linux systems before releasing them officially for Linux.
- **Play Windows games:** Many popular Windows games, both old and new, can be played on Linux by using Wine to run the game's Windows version.
- **Access legacy Windows software:** Wine allows using older, legacy Windows software that may no longer work well on newer Windows versions.

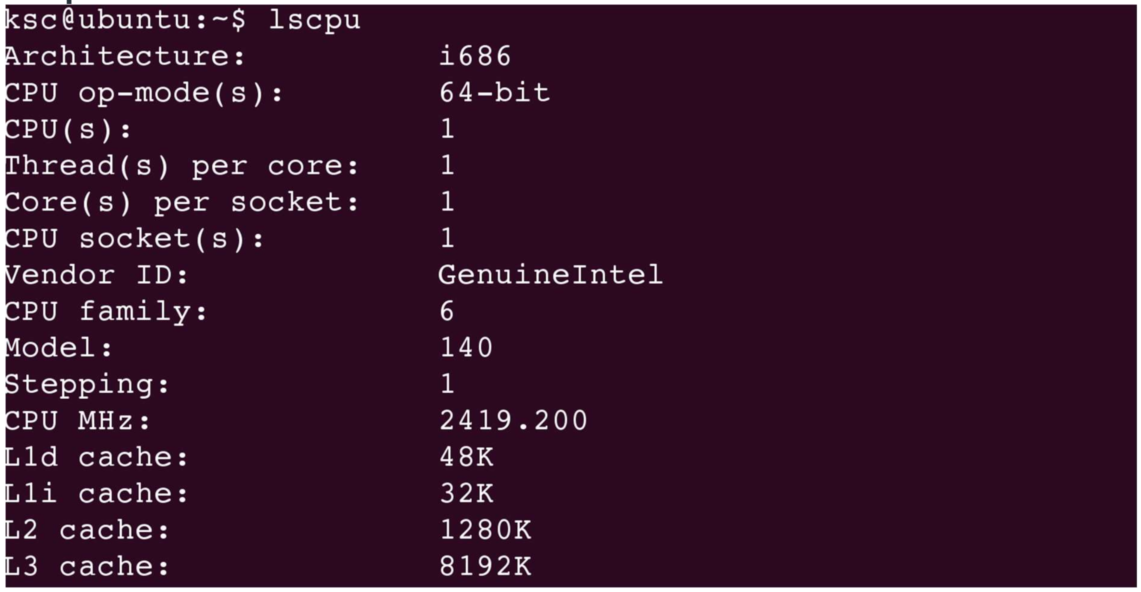
Step 1: Check your computer's architecture

You need to know if your Linux is running a 32-bit or 64-bit version. This is important because you'll need to install the right Wine version for your system. To check your computer's architecture use the below command.

Command:

```
lscpu
```

Output:



```
ksc@ubuntu:~$ lscpu
Architecture:          i686
CPU op-mode(s):        64-bit
CPU(s):                1
Thread(s) per core:    1
Core(s) per socket:    1
CPU socket(s):         1
Vendor ID:             GenuineIntel
CPU family:            6
Model:                 140
Stepping:              1
CPU MHz:               2419.200
L1d cache:             48K
L1i cache:             32K
L2 cache:              1280K
L3 cache:              8192K
```

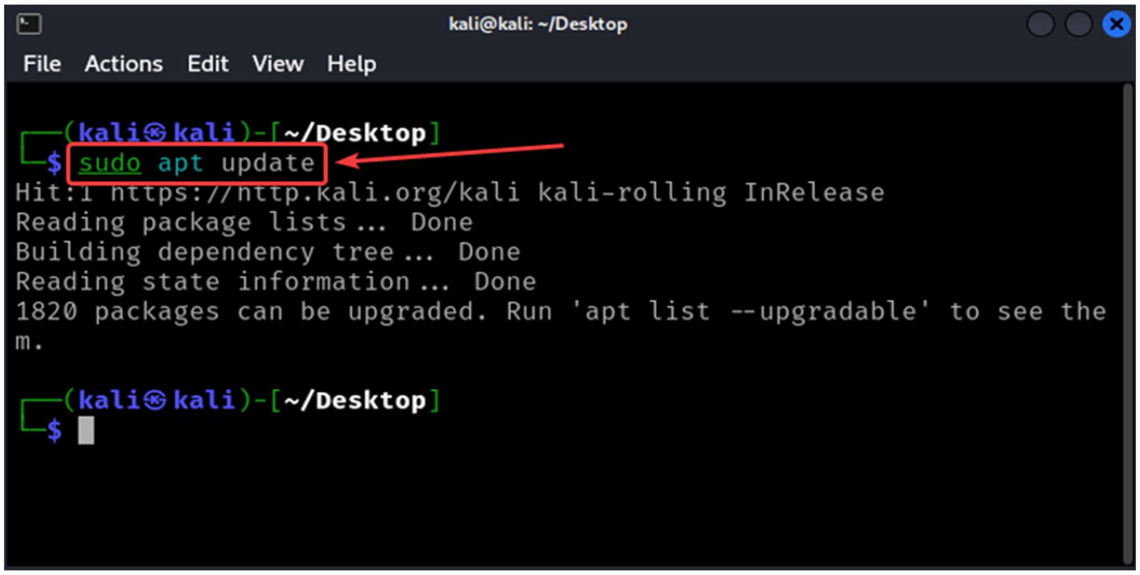
Step 2: Update your system

To get the latest software updates, you need to update the software repository on your Linux computer. Type the below command to update the system.

Command:

```
sudo apt update
```

Output:



```
┌──(kali㉿kali)-[~/Desktop]
└─$ sudo apt update
Hit:1 https://http.kali.org/kali kali-rolling InRelease
Reading package lists... Done
Building dependency tree... Done
Reading state information... Done
1820 packages can be upgraded. Run 'apt list --upgradable' to see the
m.

┌──(kali㉿kali)-[~/Desktop]
└─$ 
```

Step 3: Install the Wine

After updating your system, you need to install the Wine program itself. The command you use depends on whether your Linux is 32-bit or 64-bit.

Command:

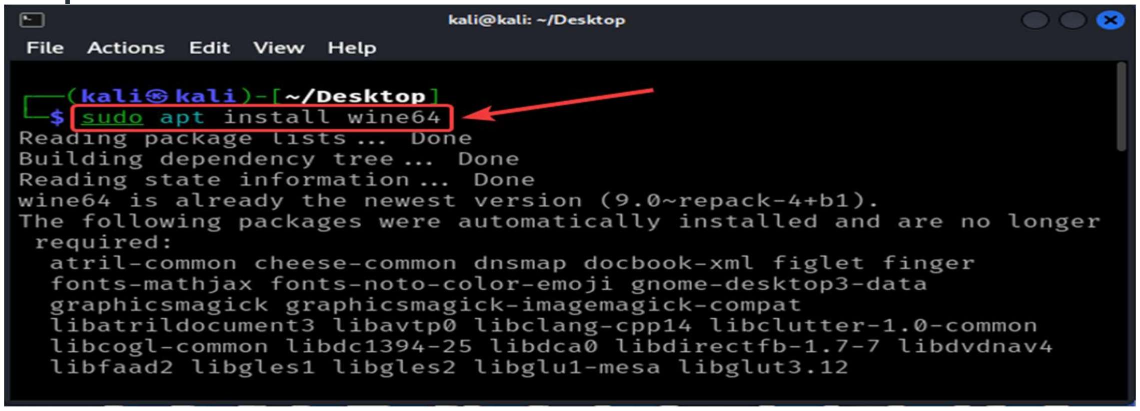
For 64-bit Linux systems use the below command :

```
sudo apt install wine64
```

For 32-bit Linux systems use the below command :

```
sudo apt install wine32
```

Output:



```
┌──(kali㉿kali)-[~/Desktop]
└─$ sudo apt install wine64
Reading package lists... Done
Building dependency tree... Done
Reading state information... Done
wine64 is already the newest version (9.0~repack-4+b1).
The following packages were automatically installed and are no longer
 required:
  atril-common cheese-common dnsmap docbook-xml figlet finger
  fonts-mathjax fonts-noto-color-emoji gnome-desktop3-data
  graphicsmagick graphicsmagick-imagemagick-compat
  libatrildocument3 libavtp0 libclang-cpp14 libclutter-1.0-common
  libcogl-common libdc1394-25 libdca0 libdirectfb-1.7-7 libdvdnav4
  libfaad2 libgles1 libgles2 libglu1-mesa libglut3.12
```

Type y to confirm the installation. When prompted to enter the "y" or "n" and press "y" to confirm the wine installation.

Setting up the Wine Windows environment

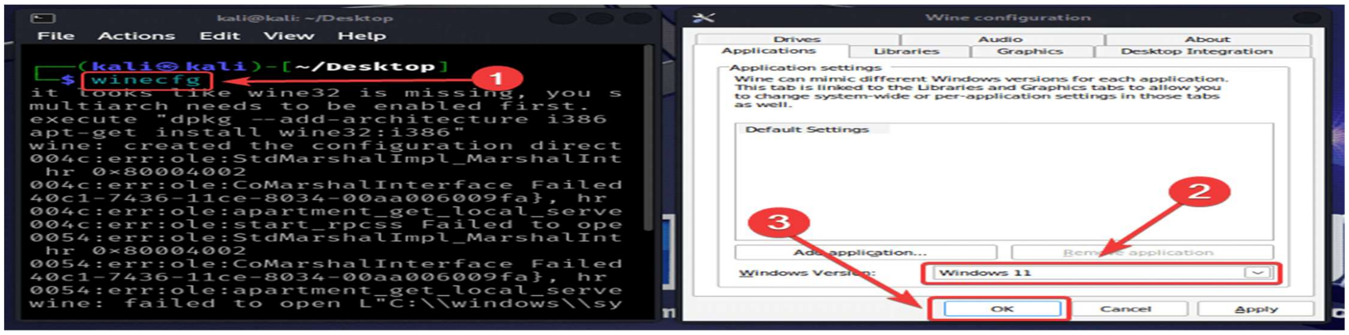
After installing Wine, you need to create a Windows home directory for it to run Windows programs. To do that Enter the following command in the terminal and then wine configuration menu will pop up select the Windows version according to your system. Check the Step 1 of the Installing the wine.

Command:

```
winecfg
```

Output:

Setting environments



```
┌──(kali㉿kali)-[~/Desktop]
└─$ winecfg
it looks like wine32 is missing, you s
multiarch needs to be enabled first.
execute "dpkg --add-architecture i386
apt-get install wine32:i386"
wine: created the configuration direct
004c:err:ole:StdMarshalImpl_MarshalInt
 hr 0×80004002
004c:err:ole:CoMarshalInterface Failed
40c1-7436-11ce-8034-00aa006009fa}, hr
004c:err:ole:apartment_get_local_serve
004c:err:ole:start_rpcss Failed to ope
0054:err:ole:StdMarshalImpl_MarshalInt
 hr 0×80004002
0054:err:ole:CoMarshalInterface Failed
40c1-7436-11ce-8034-00aa006009fa}, hr
0054:err:ole:apartment_get_local_serve
wine: failed to open L"C:\\windows\\sy
```

Wine configuration

Application settings

Wine can mimic different Windows versions for each application. This tab is linked to the Libraries and Graphics tabs to allow you to change system-wide or per-application settings in those tabs as well.

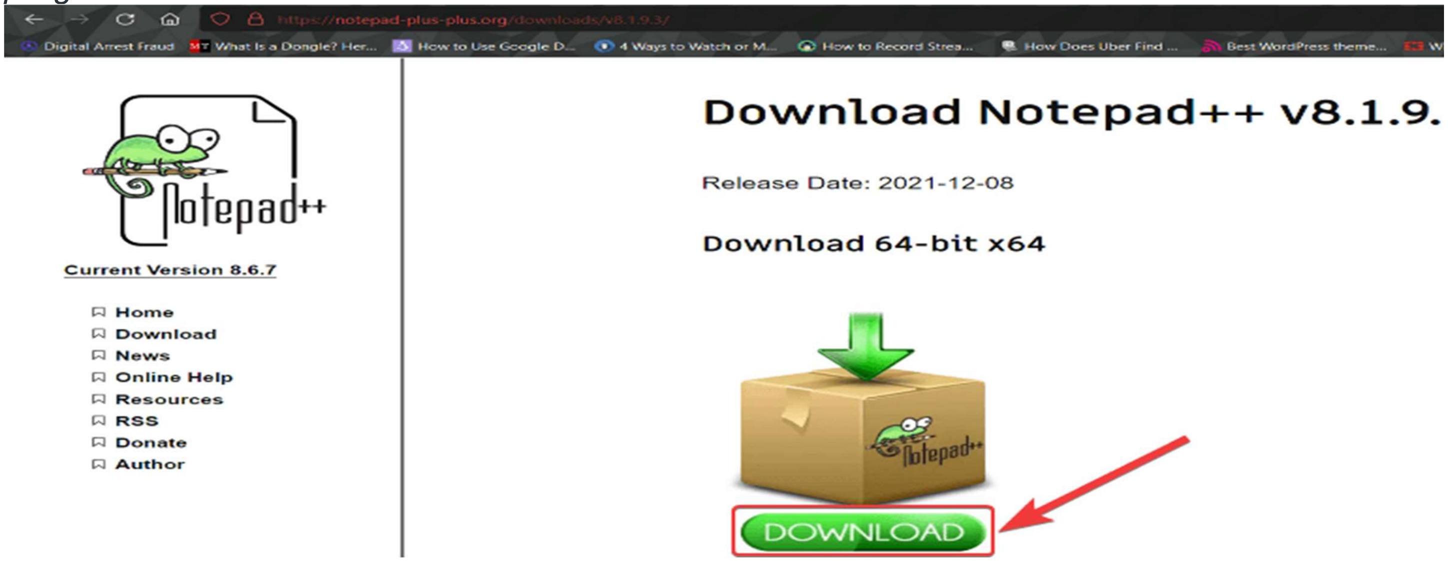
Windows Version: Windows 11

Installing a Program using Wine

Step 1: Download a Windows program

Now you need to get the Windows program you want to run on your Linux system. Go to the official website of the Windows program you want to use. Look for the download section. Download the installation file in either ".exe" or ".msi" format. These are the types of installation files Windows uses.

You can check the Wine website (https://www.winehq.org) to see a big list of Windows programs that are known to work well with Wine.



Downloading wine

Step 2: Switch to Downloads Directory

After downloading the Windows program installer file (.exe or .msi), you need to go to the Downloads folder to access it. To go to the Downloads directory use the below command.

Command:

```
cd Downloads
```

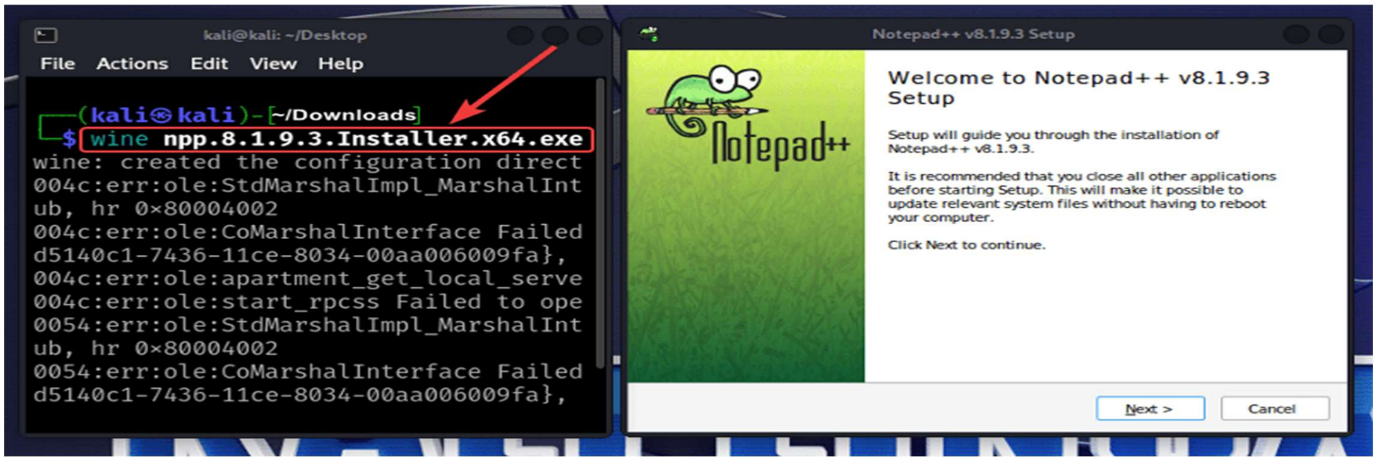
Step 3: Installing the Program

Now that you are in the Downloads folder, you can run the Windows installer file using Wine. Then in the terminal type the below with your program name.

Command:

```
wine (Program_Name)
```

Output:



```
┌──(kali㉿kali)-[~/Downloads]
└─$ wine npp.8.1.9.3.Installer.x64.exe
wine: created the configuration direct
004c:err:ole:StdMarshalImpl_MarshalInt
ub, hr 0×80004002
004c:err:ole:CoMarshalInterface Failed
d5140c1-7436-11ce-8034-00aa006009fa},
004c:err:ole:apartment_get_local_serve
004c:err:ole:start_rpcss Failed to ope
0054:err:ole:StdMarshalImpl_MarshalInt
ub, hr 0×80004002
0054:err:ole:CoMarshalInterface Failed
d5140c1-7436-11ce-8034-00aa006009fa},
```

Welcome to Notepad++ v8.1.9.3 Setup

Setup will guide you through the installation of Notepad++ v8.1.9.3.

It is recommended that you close all other applications before starting Setup. This will make it possible to update relevant system files without having to reboot your computer.

Click Next to continue.