

CHAPTER - 4

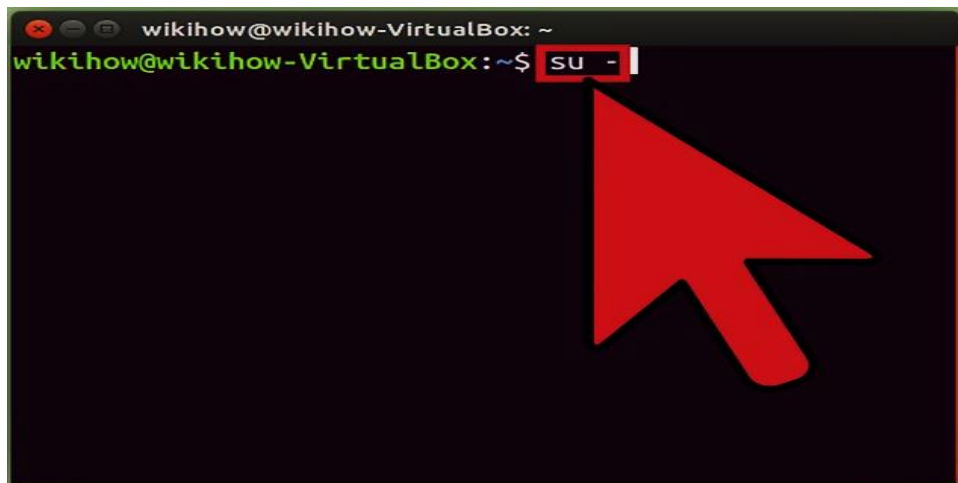
SYSTEM ADMINISTRATION

ROOT LOGIN:

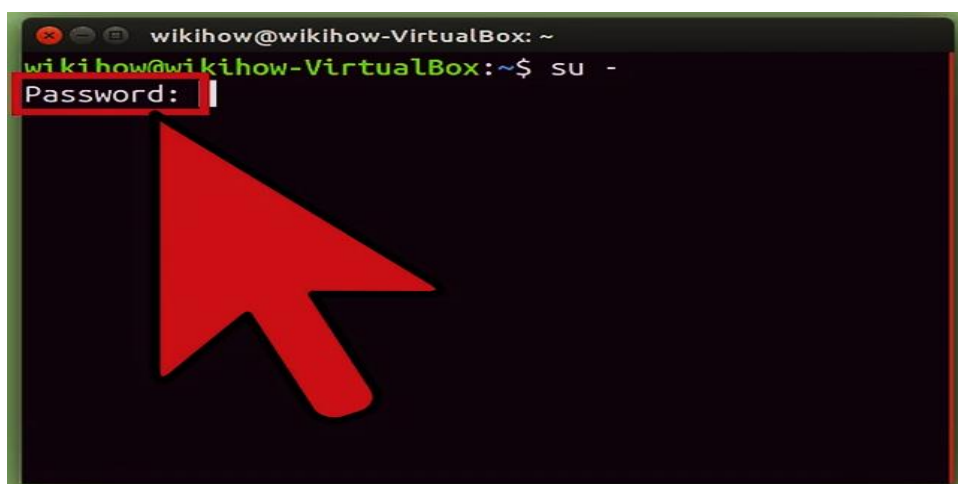
The "root" account on a Linux computer is the account with full privileges. Root access is often necessary for performing commands in Linux, especially commands that affect system files. Because root is so powerful, it's recommended to only request root access when necessary, as opposed to logging in as the root user. This can help prevent accidental damage to important system files.

1. GAINING ROOT ACCESS IN THE TERMINAL:

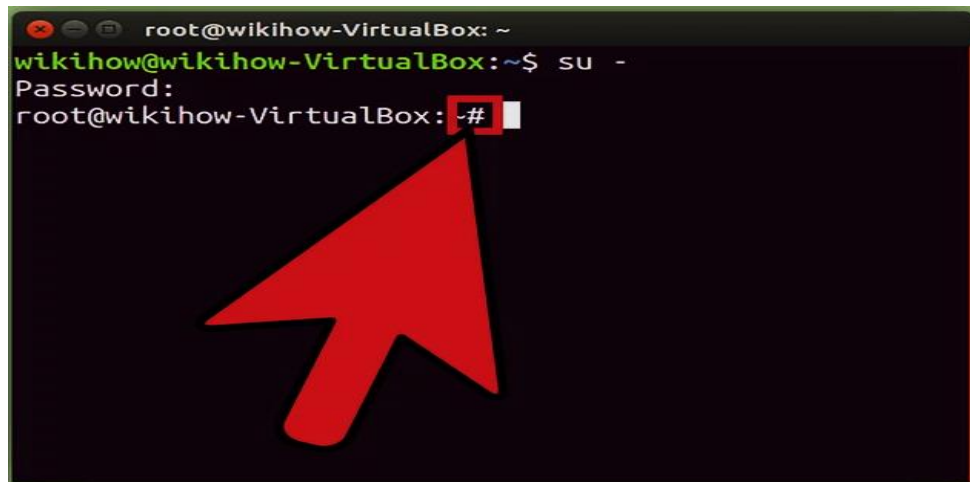
Open The Terminal: If the terminal is not already open, open it. Many distributions allow us to open it by pressing Ctrl + Alt + T.



Type su - and press Enter: This will attempt to log us in as "super user." We can actually use this command to log in as any user on the machine, but when left blank it will attempt to log in as root.

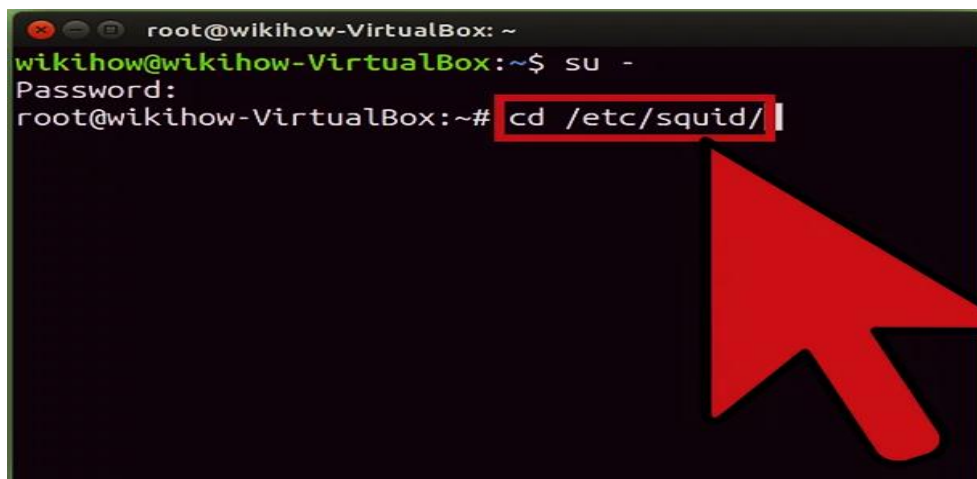


Enter The Root Password When Prompted: After typing `su -` and pressing ↵ Enter, we'll be prompted for the root password. If we get an "authentication error" message, our root account is likely locked. See the next section for instructions on unlocking it.

A terminal window titled 'root@wikihow-VirtualBox: ~' shows the command 'wikihow@wikihow-VirtualBox:~\$ su -' being entered. The prompt changes to 'Password:' and then back to 'root@wikihow-VirtualBox:~#'. A red box highlights the '#' character in the prompt, and a large red arrow points to it from the bottom left.

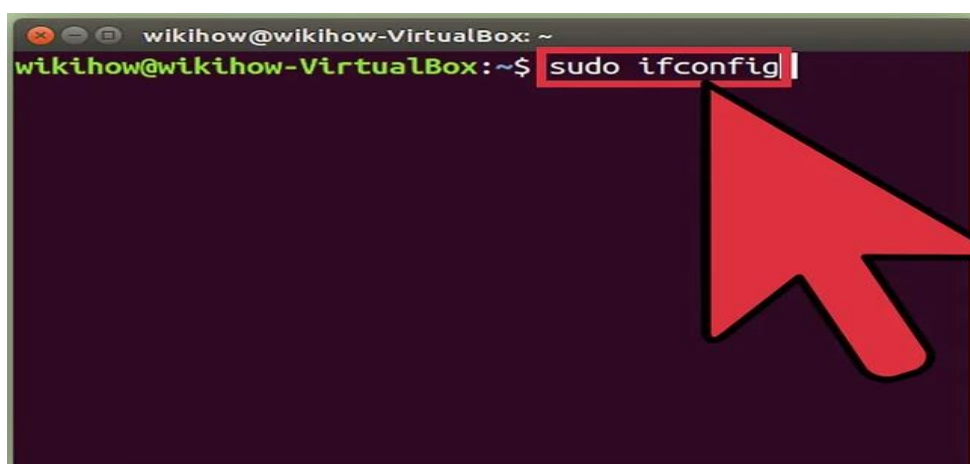
```
root@wikihow-VirtualBox: ~
wikihow@wikihow-VirtualBox:~$ su -
Password:
root@wikihow-VirtualBox:~#
```

Check The Command Prompt: When we are logged in as root, the command prompt should end with # instead of \$.

A terminal window titled 'root@wikihow-VirtualBox: ~' shows the command 'wikihow@wikihow-VirtualBox:~\$ su -' being entered. The prompt changes to 'Password:' and then back to 'root@wikihow-VirtualBox:~#'. The command 'cd /etc/squid/' is entered, and a red box highlights it. A large red arrow points to the command from the bottom right.

```
root@wikihow-VirtualBox: ~
wikihow@wikihow-VirtualBox:~$ su -
Password:
root@wikihow-VirtualBox:~# cd /etc/squid/
```

Enter The Commands That Require Root Access: Once we've used `su -` to log in as root, we can run any commands that require root access. The `su` command is preserved until the end of the session, so we don't need to keep re-entering the root password every time we need to run a command.

A terminal window titled 'wikihow@wikihow-VirtualBox: ~' shows the command 'wikihow@wikihow-VirtualBox:~\$ sudo ifconfig' being entered. A red box highlights the command, and a large red arrow points to it from the bottom right.

```
wikihow@wikihow-VirtualBox: ~
wikihow@wikihow-VirtualBox:~$ sudo ifconfig
```

Consider using sudo instead of su - `sudo` ("super user do") is a command that lets us run other commands as root temporarily. This is the best way for most users to run root commands, as the root environment is not maintained, and the user doesn't need to know the root password. Instead, the user will enter their own user password for temporary root access.

- Type `sudo command` and press ↵ Enter (e.g. `sudo ifconfig`). When prompted for the password, enter *user* password, not the root password.
- `sudo` is the preferred method for distributions like Ubuntu, where it will work even when the root account is locked.
- This command is limited to users with administrator privileges. Users can be added or removed from `/etc/sudoers`.

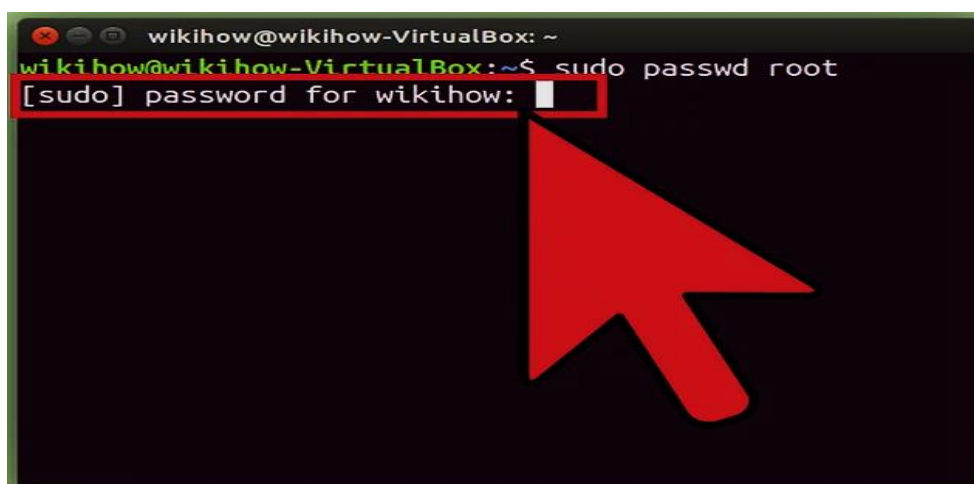
2. UNLOCKING THE ROOT ACCOUNT (UBUNTU):

Ubuntu (and several other distributions) locks the root account so that the average user can't access it. This is done because root access is rarely necessary when using the **sudo** command. Unlocking the root account will allow us to log in as root.

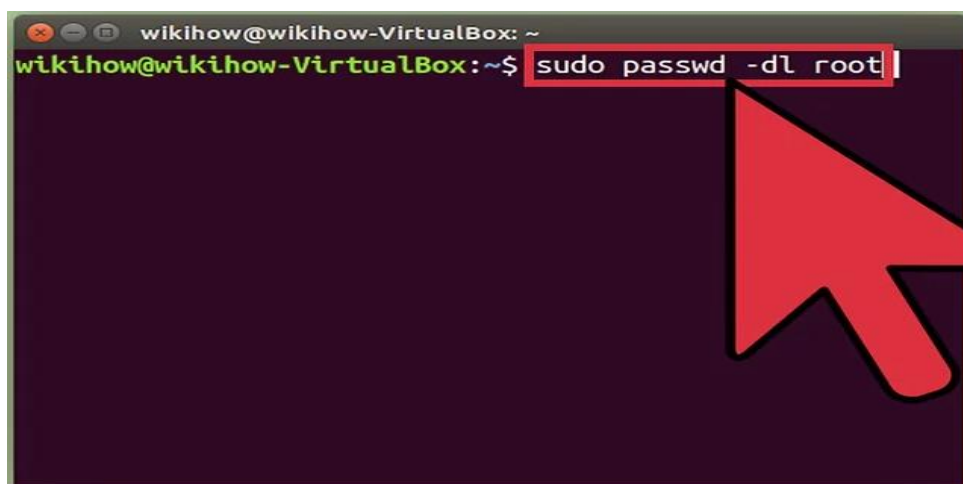
Open The Terminal: If we're in the desktop environment, we can press Ctrl + Alt + T to start the terminal.



Type `sudo passwd root` and press Enter: When prompted for a password, enter *user* password.



Set A New Password: We'll be prompted to create a new password and enter it twice. Once a password has been set, the root account will be active.

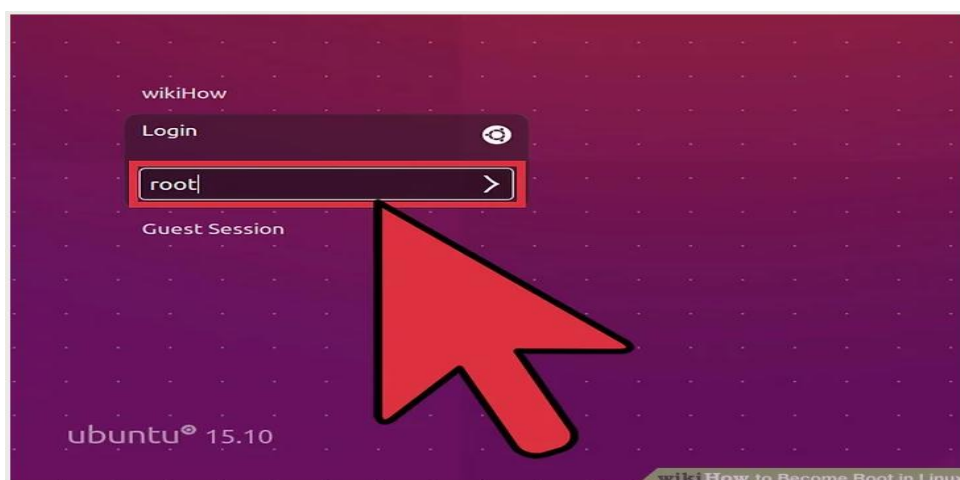


Lock The Root Account Again: If we want to lock the root account, enter the following command to remove the password and lock root: `sudo passwd -dl root`

3. LOGGING IN AS ROOT:

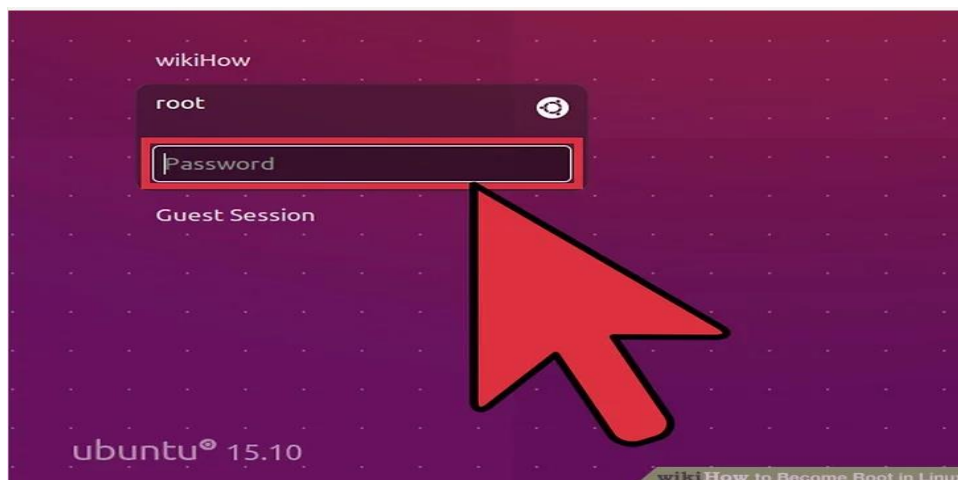
Consider Using Other Methods For Gaining Temporary Root Access: Logging in as root is not recommended for regular use, as it is very easy to perform commands that will render our system inoperable, and it also poses a security risk, especially if we are running an SSH server on our machine. Only log in as root when performing emergency repairs, such as dealing with disk failures or restoring locked accounts.

- Using sudo or su instead of logging in as root will help prevent unintended damage while logged in as root. Using these commands gives the user a chance to think about the command before severe damage is done.
- Some distributions, such as Ubuntu, leave the root account locked until we manually unlock it. Not only does this prevent users from unknowingly doing too much damage using the root account, it also secures the system from potential hackers, as the root account is typically targeted first. With a locked root account, hackers aren't able to gain access with it.



Enter root As the User When Logging into Linux: If the root account is unlocked and we know the password, we can log in as root when we're prompted to log in with a user account.

Enter root as the user when prompted to log in. If we need root access to perform a command, use the method in the previous section.

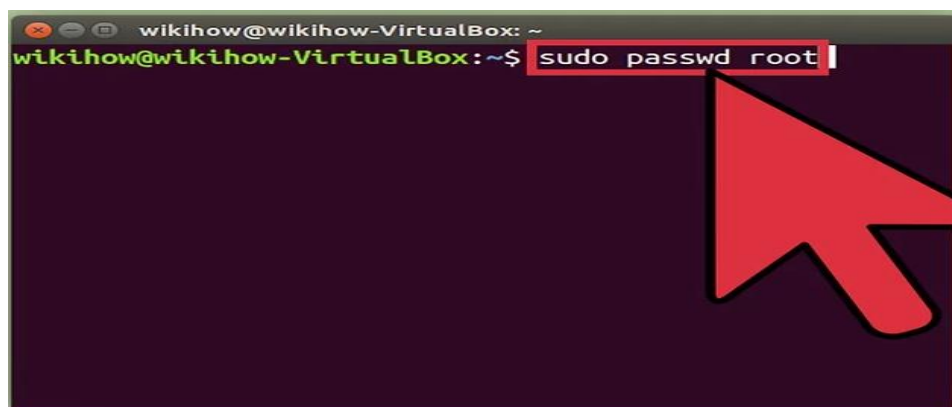


Enter The Root Password As The User Password: After entering root as the username, enter the root password when prompted. In many cases, the root password may be "password." If we don't know the root password, or have forgotten it, see the next section for instructions on resetting it. In Ubuntu, the root account is locked and cannot be used until it has been unlocked.



Avoid Running Complex Programs While Logged In As Root: There's a chance that the program we intend to run will have a negative effect on our system when it has root access. It's highly recommended that we use sudo or su to run programs instead of logging in as root.

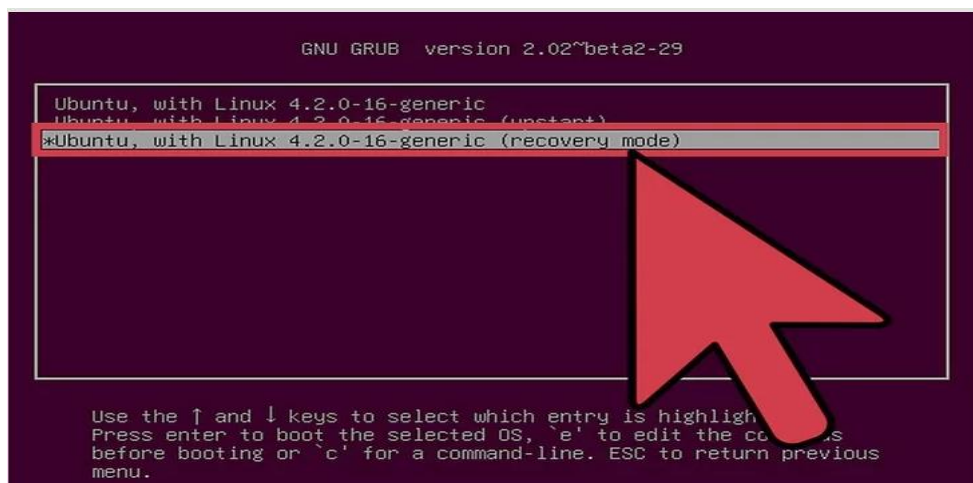
4. RESETTING THE ROOT OR ADMIN PASSWORD:



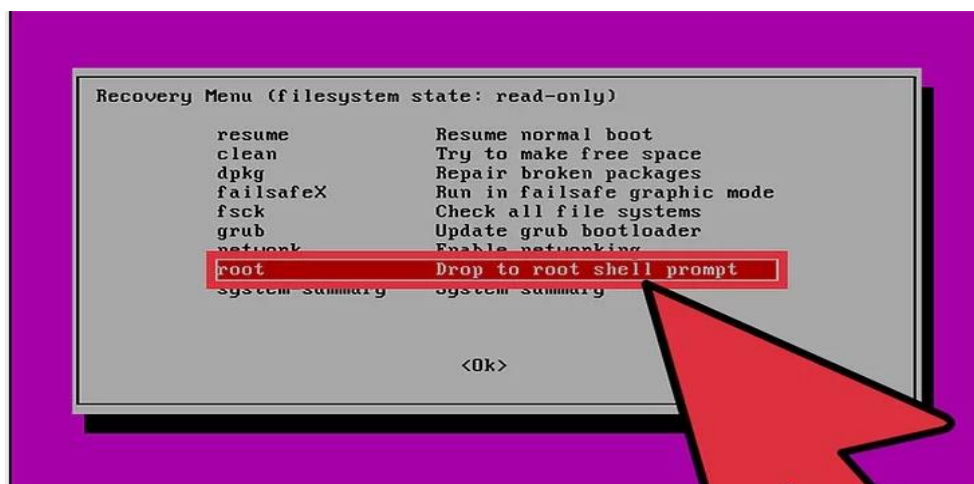
Reset the Root Password If It Has Been Forgotten: If we've forgotten the root password *and* our user password, we'll need to boot into recovery mode in order to change them. If we know our user password and need to change the root password, just type `sudo passwd root`, enter user password, then create a new root password.



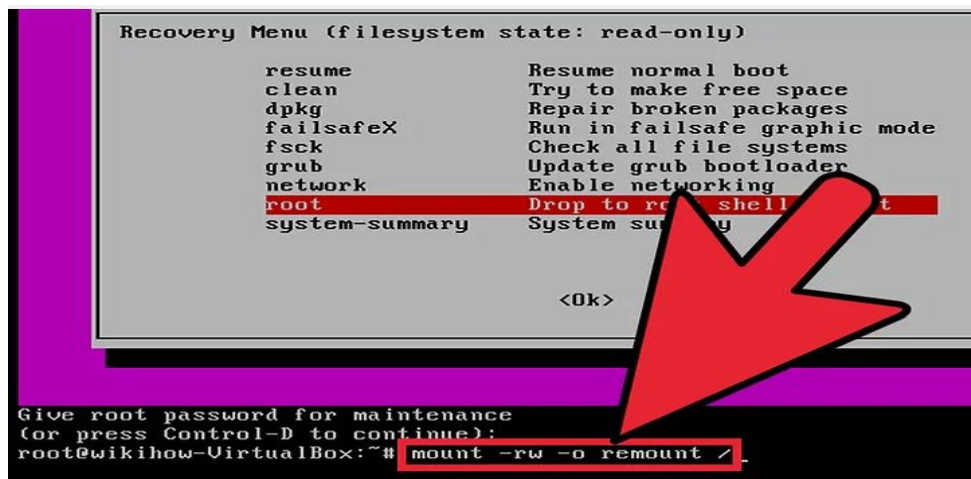
Reboot Computer and Hold Left Shift after the BIOS Screen: This will open the GRUB menu. The timing on this can be tricky, so we may have to try multiple times.



Select The First (Recovery Mode) Entry on the List: This will load recovery mode for our current distribution.



Select the root option from the menu that appears: This will start the terminal with us logged in as the root account.



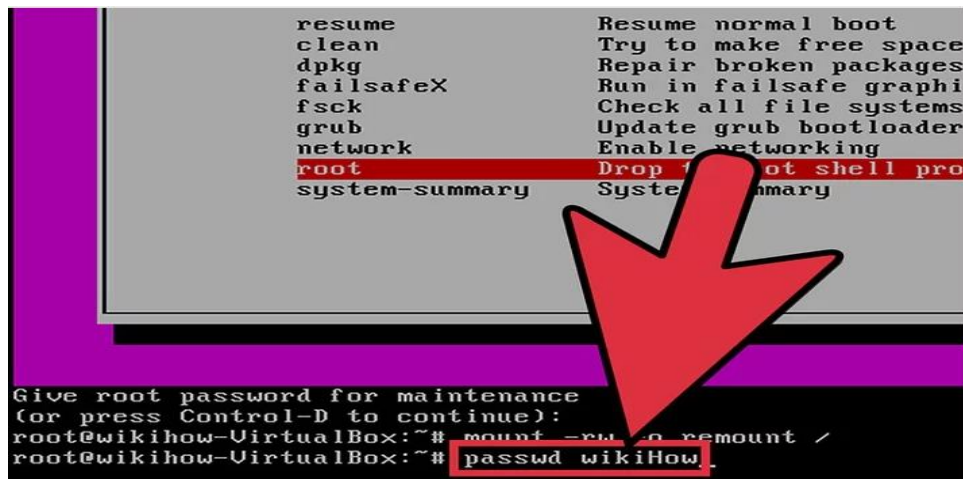
```
Recovery Menu (filesystem state: read-only)

resume          Resume normal boot
clean           Try to make free space
dpkg            Repair broken packages
failsafeX       Run in failsafe graphic mode
fsck            Check all file systems
grub            Update grub bootloader
network         Enable networking
root            Drop to root shell prompt
system-summary System summary

<Ok>

Give root password for maintenance
(or press Control-D to continue):
root@wikihow-VirtualBox:~# mount -rw -o remount /
```

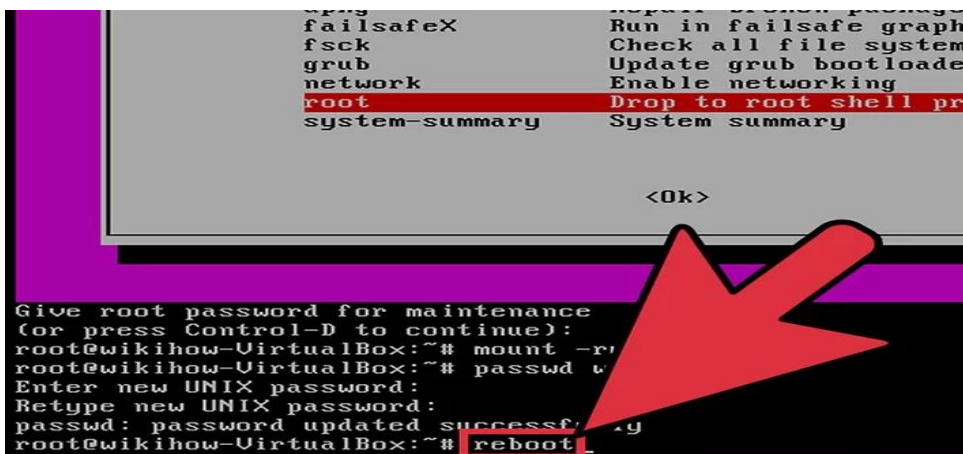
Remount The Drive With Write Permissions: When we boot into recovery mode, we will typically only have read permissions. Enter the following command to enable write access:
`mount -rw -o remount /`



```
resume          Resume normal boot
clean           Try to make free space
dpkg            Repair broken packages
failsafeX       Run in failsafe graphi
fsck            Check all file systems
grub            Update grub bootloader
network         Enable networking
root            Drop to root shell pro
system-summary System summary

Give root password for maintenance
(or press Control-D to continue):
root@wikihow-VirtualBox:~# mount -rw -o remount /
root@wikihow-VirtualBox:~# passwd wikiHow
```

Create a New Password for Any Accounts we're locked Out Of: Once we're logged in as root and have changed the access permissions, we can create a new password for any account: Type `passwd accountName` and press `↵` Enter. If we need to change the root password, type `passwd root`. Enter the new password twice when prompted.



```
failsafeX       Run in failsafe graph
fsck            Check all file system
grub            Update grub bootloade
network         Enable networking
root            Drop to root shell pr
system-summary System summary

<Ok>

Give root password for maintenance
(or press Control-D to continue):
root@wikihow-VirtualBox:~# mount -rw -o remount /
root@wikihow-VirtualBox:~# passwd w
Enter new UNIX password:
Retype new UNIX password:
passwd: password updated successfully
root@wikihow-VirtualBox:~# reboot
```

Reboot Computer After Resetting Passwords: Once we're finished resetting passwords, we can reboot and use our computer as normal. Our new passwords will take effect immediately.

SUPERUSER (SUDO):

There are two ways to run administrative applications in Linux. We can either switch to the super user (root) with the su command, or we can take advantage of sudo. How we do this will depend upon which distribution we use. Some distributions enable the root user (such as Fedora, Red Hat, openSUSE), while some do not (such as Ubuntu and Debian). There are pros and cons for each.

Sudo stands for either "substitute user do" or "super user do" (depending upon how we want to look at it). What sudo does is incredibly important and crucial to many Linux distributions. Effectively, sudo allows a user to run a program as another user (most often the root user). There are many that think sudo is the best way to achieve "best practice security" on Linux. There are some, however, that feel quite the opposite. Regardless of where we stand, and what distribution we are currently using, there will come a time when we will have to take advantage of sudo. Whether we will need to simply use sudo or we will have to configure sudo or we will want to know the ins and outs of this powerful tool.

History of sudo:

Sudo found its beginnings in 1980 at the department of computer science SUNY/Buffalo (created by Bob Coggeshall and Cliff Spencer). Since its first inception, sudo has been re-iterated numerous times (adding new features and changing developers). At one point, around 1994, sudo was being developed by Todd Miller at the Colorado University in Boulder, CO and an unofficial "fork" of sudo was released called "CU sudo". This "fork" added support for more distributions as well as numerous bug fixes. This "CU" prefix was finally dropped in 1999 and what was "CU sudo" is now the version of sudo we use today. The original sudo has not had a release since 1991. So the "fork" won and is still developed by Todd Miller. Now that we have had a bit of a history lesson, let's take a look and see how sudo is used.

Difference Between sudo and su:

If we are accustomed to a more traditional Linux setup, then we are used to use the su command to gain root privileges. We can even issue the command su - to effectively log in as the root (root's home becomes our home). With these types of distributions we can also log in as the root user. To many think this is a bad idea. NEVER log in as the root user. If we are using a distribution that relies on su and allows root user log in, log in as our standard user and su to the root user.

Now with sudo-based distributions we will most likely notice that we cannot log in as a root user. In fact, in distributions such as Ubuntu, the root user account has been "disabled." We cannot log in as root and we cannot su to become the root user. All we can do is issue commands with the help of sudo to gain administrative privileges.

Usage:

Using sudo, in its most basic form, is simple. Say we have to run the *dpkg* to install a piece of software. If, as our standard user, we just issue the command *dpkg -i software.deb* we will receive

an error warning us that the user does not have proper permissions to execute the command. That is because standard users, by default, cannot install applications on a Linux machine. In order to successfully install an application on a Linux machine we have to have super user privileges. So, to change that command so that we can successfully run the installation, we would instead issue the command *sudo dpkg -i software.deb*.

Configuration:

This is a warning: If we mis-configure our */etc/sudoers* file, we can damage our installation (at which point we will have to log boot in rescue mode). Sudo is VERY particular about syntax in the configuration file. So always double check configurations before saving our file. Fortunately there is only one file we need to concern ourselves with and that is */etc/sudoers*. We may notice that, even in order to view the */etc/sudoers* file we have to use the *sudo* command. This file will seem very simple, once we understand the layout and the function.

To make changes to the *sudo* configuration file we need to use a specific command - *sudo visudo*. When we open up this file we will notice that the *sudoers* file is fairly small in size. There really isn't much to it, but what there is to it is key. Let's take a look at how to add a user to the *sudoers* file.

The basic entry for a user looks like this:

user hostlist = (userlist) commandlist

Typically you will find an entry like this:

root ALL = (ALL) ALL

Which indicates that the user *root* on all hosts using any user can run all commands. Fairly straight-forward. But let's say we want to allow a single user access to one administrative command without having to enter a password. Let's use the command *dpkg* (not wise, but an easy means of illustration) and allow the user *Ram* to issue those commands without having to issue a password. To do this we would add a line similar to this:

Ram ALL = NOPASSWD: /usr/sbin/synaptic

to the */etc/sudoers* file. Now the user *Ram* can run *synaptic* by entering *sudo synaptic* but will not be prompted for a password. This is handy on a single-user system but should be used with caution. We do not want to allow just any command to be run sans password or we open ourselves to all sorts of vulnerabilities.

Now, let's say we want to prevent certain users from using *sudo*. We can do this as well. If we have one user that is to be administrator of a machine, say *Kundan*, and all other users should be users without admin privileges, we can do this a couple of ways. The first (and less desirable method) is to do the following:

Add an entry for *Kundan* like so:

Kundan ALL = (ALL) ALL

And now comment out the entry:

%admin ALL = (ALL) ALL

by adding a "#" character at the beginning of the line.

At this point the only user on the system that will be able to run administrative commands is Kundan. Now this can cause issues if we have certain applications that must run with administrative privileges and are allowed such privileges by being a member of the admin group. We can avoid this issue by simply opening up the Users administrative tool and removing all users, except for those we want to be allowed to have admin rights from the admin group. Let's stick with our example. We want all users other than Kundan to have restricted access to run administrative commands and tools. To do this, follow these steps:

1. Open up the User administrator.
2. Go to the Groups manager.
3. Select the admin group.
4. Click properties.
5. Uncheck all users but Kundan from the list.
6. Close the Groups manager and the User administrator.

Now only the user Kundan will have administrative rights on the machine.

CONFIGURATION OF HARDWARE WITH KUDZU:

When we add or remove hardware from our computer and reboot Red Hat Linux, a window appears during the reboot process advising that hardware has either been added or removed and asking if we want to reconfigure it. The program that detects and reconfigures our hardware is called kudzu.

The kudzu program is a hardware auto-detection and configuration tool that runs automatically at boot time. If we like, we can also start kudzu while Red Hat Linux is running. In either case, here is what kudzu does:

1. It checks the hardware connected to our computer.
2. It compares the hardware it finds to the database of hardware information stored in the /etc/sysconfig/hwconf file.
3. It prompts us to change our system configuration, based on new or removed hardware that was detected.

The following is a list of hardware that kudzu can detect (according to the kudzu README file), followed by a description of what kudzu does to configure the device. Other devices may be detected as well (such as USB devices).

- ❖ **Network Devices:** Adds an Ethernet interface alias (eth0, eth1, etc.) if necessary and either migrates the old device configuration or creates a new one.
- ❖ **SCSI:** Adds an alias for scsi_hostadapter.
- ❖ **Sound Card:** Runs the sndconfig command to configure and test the sound card.
- ❖ **Mouse:** Links the new mouse device to /dev/mouse and runs the mouseconfig command to configure and test the mouse.
- ❖ **Modem:** Links the new modem device to /dev/modem.
- ❖ **CD-ROM:** Links the CD-ROM device to /dev/cdrom.
- ❖ **Scanner:** Links the new scanner device to /dev/scanner.

- ❖ **Keyboard:** Runs the `kbdconfig` command to reconfigure the keyboard. Also, if we are using a serial console, it makes sure `/etc/inittab` and `/etc/securetty` are configured to be used by a serial console.

The following is a list of actions kudzu takes when a device is removed:

- ❖ **Network:** Removes the alias for the Ethernet interface (`eth0`, `eth1`, etc.).
- ❖ **SCSI:** Removes the alias for the SCSI host adapter (`scsi_hostadapter`).
- ❖ **Mouse:** Removes the link to `/dev/mouse`.
- ❖ **Modem:** Removes the link to `/dev/modem`.
- ❖ **CD-ROM:** Removes the link to `/dev/cdrom`.
- ❖ **Scanner:** Removes the link to `/dev/scanner`.

To run kudzu, either reboot (during the reboot, kudzu is run automatically) or switch to a virtual terminal (`Ctrl+Alt+F2`), log in as root, and run the kudzu command. For any hardware that has been added or removed since the last time kudzu was run, we are asked if we want to configure it, not configure it, or do nothing.

CONFIGURE MODULES:

In a perfect world, after installing and booting Red Hat Linux, all of our hardware should be detected and available for access. While Red Hat Linux is rapidly moving closer to that world, there are times when we must take special steps to get our computer hardware working.

Red Hat Linux comes with tools for configuring the drivers that stand between the programs we run (such as CD players and Web browsers) and the hardware they use (such as CD-ROM drives and network cards). The intention is to have the drivers our system needs most often built into the kernel; these are called *resident drivers*. Other drivers that are added dynamically as needed are referred to as *loadable modules*.

Finding Available Modules:

If we have installed the Linux kernel source code (kernel-source package), source code files for available drivers are stored in subdirectories of the `/usr/src/linux-2.4/drivers` directory. There are several ways of finding information about these drivers:

- ❖ **make xconfig:**

With `/usr/src/linux-2.4` as our current directory, type **make xconfig** from a Terminal window on the desktop. Select the category of module we are interested in and click Help next to the driver that interests us. The help information that appears tells us the module name and a description of the driver.

- ❖ **Documentation:**

The `/usr/src/linux-2.4/Documentation` directory contains lots of plain-text files describing different aspects of the kernel and related drivers. Of particular interest is the `modules.txt` file (which describes how to work with modules) and the `Configure.help` file (which contains all the help files hardware drivers).

- ❖ **kernel-doc:**

The kernel-doc software package (available on CD #3 of the Red Hat Linux distribution) contains a large set of documents describing the kernel and drivers. These documents are stored in the /usr/share/doc/kernel-doc* directory.

After modules have been built, they are installed in the /lib/modules/2.4* directory. The name of the directory is based on the current release number of the kernel. Modules that are in that directory can then be loaded and unloaded as they are needed.

Note: In previous releases, Red Hat Linux stored modules in the /lib/modules directory, rather than the /lib/modules/2.4* directory. This new structure allows us to store modules on our system that relate to different kernel versions we may be running.

Listing Loaded Modules:

To see which modules are currently loaded into the running kernel on our computer, we can use the lsmod command. Here's an example:

```
# lsmod
Module                Size  Used by
sr_mod                15120  0  (autoclean)
es1371                26784  0  (autoclean)
ac97_codec             8704  0  (autoclean) [es1371]
gameport              1920  0  (autoclean) [es1371]
soundcore              4112  4  (autoclean) [es1371]
binfmt_misc           6272  1
nuscsitcp             17200  0  (unused)
autofs                10816  1  (autoclean)
tulip                 46400  1
ipchains              36960  0  (unused)
ide-scsi               8192  0
scsi_mod              93568  3  [sr_mod nuscsitcp ide-scsi]
hid                   18160  0  (unused)
input                  3456  0  [hid]
usb-uhci              21440  0  (unused)
usbcore               50432  1  [hid usb-uhci]
ext3                  50656  2
jbd                   39376  2  [ext3]
```

This output shows a variety of modules that have been loaded on a Linux system. The modules loaded on this system include several to support the Ensoniq 1371 sound card that is installed (es1371, ac97_codec, gameport, and soundcore). There are also modules to support the IDE CD-ROM drive running in SCSI emulation (scsi_mod, sr_mod, nuscsitcp, and ide-scsi).

To find information about any of the loaded modules, we can use the modinfo command. For example, we could type the following:

```
# modinfo -d es1371
```

```
"ES1371 AudioPCI97 Driver"
```

Not all modules have descriptions available. In this case, however, the es1371 module is described as an ES1371 AudioPCI87 Driver. We can also use the -a option to see the author of the module or -n to see the object file representing the module. The author information often has the e-mail address of the driver's creator, so we can contact the author if we have problems or questions about it.

Loading Modules:

We can load any module that has been compiled and installed (to the `/lib/modules` directory) into our running kernel using the `insmod` command. The most common reasons for loading a module are that we want to use a feature temporarily (such as loading a module to support a special file system on a floppy we want to access) or to identify a module that will be used by a particular piece of hardware that could not be auto-detected.

Here is an example of the `insmod` command being used to load the `parport` module. The `parport` module provides the core functions to share parallel ports with multiple devices.

insmod parport

Using `/lib/modules/2.4.20-2.48/kernel/drivers/parport/parport.o`

After `parport` is loaded we can load the `parport_pc` module to define the PC-style ports available through the interface. The `parport_pc` module lets us optionally define the addresses and IRQ numbers associated with each device sharing the parallel port. For example:

insmod parport_pc io=0x3bc irq=auto

In the previous example, a device is identified as having an address of `0x3bc`. The IRQ for the device is auto-detected.

The `insmod` command loads modules temporarily. At the next system reboot, the modules we enter disappear. To permanently add the module to our system, add the `insmod` command line to one of the start-up scripts that are run at boot time.

Removing Modules

We can remove a module from a running kernel using the `rmmod` command. For example, to remove the module `parport_pc` from the current kernel, type the following:

rmmod parport_pc

If the module is not currently busy, the `parport_pc` module is removed from the running kernel.

CHECKING SYSTEM SPACE:

A quick way to get a summary of the available and used disk space on our Linux system is to type in the `df` command in a terminal window. The command `df` stands for "**d**isk **f**ilesystem". With the `-h` option (`df -h`) it shows the disk space in "human readable" form, which in this case means, it gives us the units along with the numbers.

The output of the `df` command is a table with four columns. The first column contains the file system path, which can be a reference to a hard disk or another storage device, or a file system connected to the network.

The second column shows the capacity of that file system. The third column shows the available space, and the last column shows the path on which that file system is mounted. The mount point is the place in the directory tree where we can find and access that file system.

The `du` command, on the other hand, shows the disk space used by the files and directories in the current directory. Again the `-h` option (`df -h`) makes the output easier to comprehend.

By default, the `du` command lists all subdirectories to show how much disk space each has occupied. This can be avoided with the `-s` option (`df -h -s`). This only shows a summary. Namely the combined disk space used by all subdirectories. If we want to show the disk usage of a directory (folder) other than the current directory, we simply put that directory name as the last argument. For example: **`du -h -s images`**, where "images" would be a subdirectory of the current directory.

More about the `df` Command:

By default, we will only need to see the accessible file systems which is the default when using the `df` command. We can, however, return the usage of all file systems including pseudo, duplicate and inaccessible file systems by using either of the following commands:

`df -a`

`df -all`

The above commands won't seem very useful to most people but the next ones will. By default, the used and available disk space is listed in bytes. We can, of course, use the following command:

`df -h`

This displays the output in a more readable format such as size 546G, available 496G. Whilst this is ok the units of measure differ for each filesystem. To standardize the units across all file systems we can simply use the following commands:

`df -BM`

`df --block-size=M`

The M stands for megabytes. We can also use any of the following formats:

- K = Kilobytes
- M = Megabytes
- G = Gigabytes
- T = Terabytes
- P = Petabytes
- E = Exabyte
- Z = Zettabyte
- Y = Yottabyte

A kilobyte is 1024 bytes and a megabyte is 1024 kilobytes. We may wonder why we use 1024 and not 1000. It is all to do with the binary makeup of a computer. We start at 2 and then 4, 8, 16, 32, 64, 128, 256, 512 and then 1024.

Human beings, however, tend to count in decimal and so we are used to thinking in 1, 10, 100, 1000. We can use the following command to display the values in a decimal format as opposed to the binary format. (i.e. it prints values in powers of 1000 instead of 1024).

`df -H`

df --si

We will find that numbers such as 2.9G become 3.1G.

Running out of disk space isn't the only problem we might face when running a Linux system. A Linux system also uses the concept of inodes. Each file we create is given an inode. We can, however, create hard links between files which also use inodes.

There is a limit on the number of inodes a file system can use. To see whether our file systems are close to hit their limit run the following commands:

df -i

df --inodes

We can customize the output of the df command as follows:

df --output=FIELD_LIST

The available options for the FIELD_LIST are as follows:

- source
- fstype
- itotal
- iused
- iavail
- ipcent
- size
- used
- avail
- pcent
- file
- target

We can combine any or all of the fields. For example:

df --output=source, size, used

We may also wish to see totals for the values on the screen such as the total available space across all file systems. To do this use the following command:

df --total

By default, the df listing doesn't show the file system type. We can output the file system type by using the following commands:

df -T

df --print-type

The file system type will be something like ext4, vfat, tmpfs

If we just want to see information for a certain type we can use the following commands:

df -t ext4

```
dt --type=ext4
```

Alternatively, we can use the following commands to exclude file systems.

```
df -x ext4
```

```
df --exclude-type=ext4
```

More about the du Command:

The du command as we have already read lists details about the file space usage for each directory. Here are a few more switches which we may or may not find useful. By default after each item is listed a carriage return is shown which lists each new item on a new line. We can omit the carriage return by using the following commands:

```
du -0
```

```
du --null
```

This isn't particularly useful unless we want to see the total usage quickly. A more useful command is the ability to list the space taken by all files and not just the directories. To do this use the following commands:

```
du -a
```

```
du --all
```

We will probably want to output this information to a file using the following command:

```
du -a > filename
```

As with the df command, we can specify the way the output is presented. By default, it is in bytes but we can choose kilobytes, megabytes etc. using the following commands:

```
du -BM
```

```
du --block-size=M
```

We can also go for the human readable for such as 2.5G using the following commands:

```
du -h
```

```
du --human-readable
```

To get a total at the end use the following commands:

```
du -c
```

```
du --total
```

MONITORING SYSTEM PERFORMANCE:

Monitoring our Linux system is essential in order to be able to improve its performance, locate the source of a problem and take more targeted corrective actions. As it is always the case with

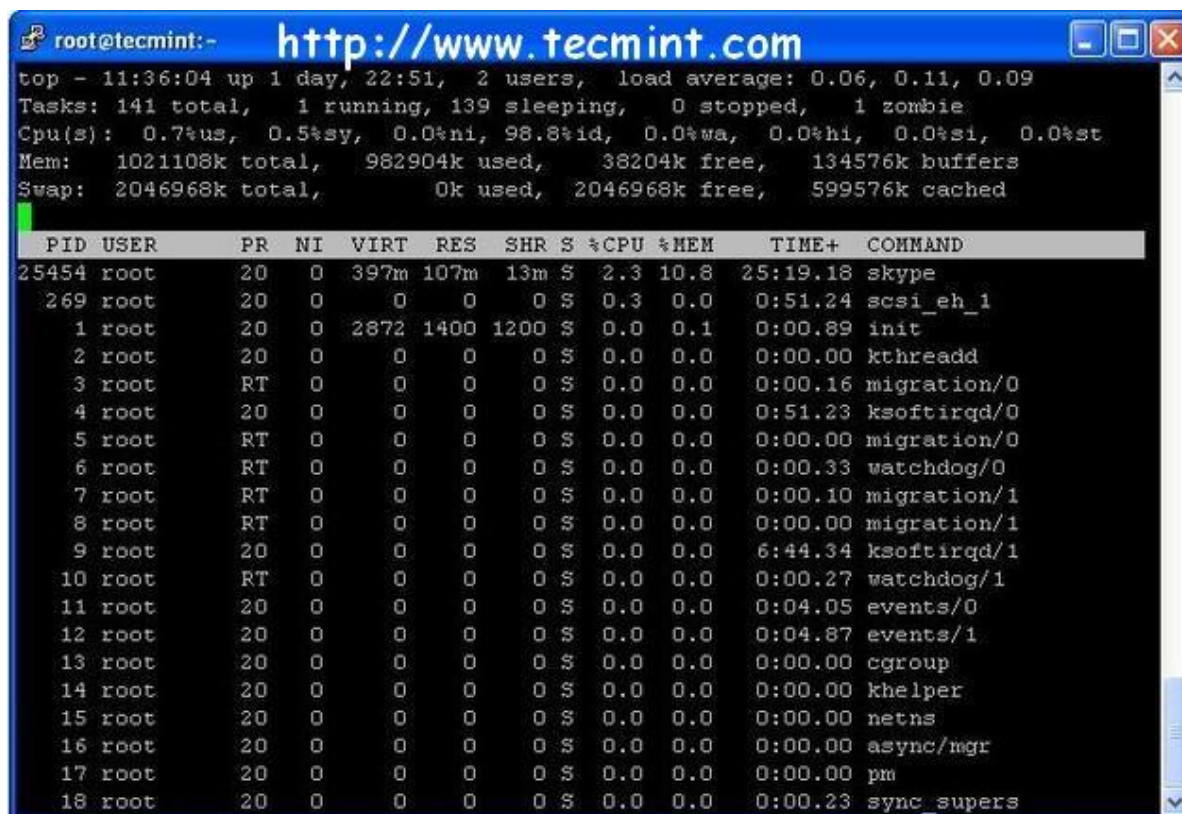
Linux, there are quite a few tools and many different ways we can utilize to monitor different aspects of our system's performance.

Top 20 frequently used command line monitoring tools that might be useful for every Linux System Administrator. These commands are available under all flavors of Linux and can be useful to monitor and find the actual causes of performance problem. This list of commands shown here are very enough for us to pick the one that is suitable for our monitoring scenario.

1. Top-Linux Process Monitoring:

Linux Top command is a performance monitoring program which is used frequently by many system administrators to monitor Linux performance and it is available under many Linux/Unix like operating systems. The top command used to display all the running and active real-time processes in ordered list and updates it regularly. It display CPU usage, Memory usage, Swap Memory, Cache Size, Buffer Size, Process PID, User, Commands and much more. It also shows high memory and cpu utilization of a running processes. The top command is much useful for system administrator to monitor and take correct action when required. Let's see top command in action.

top



```
root@tecmint:~ http://www.tecmint.com
top - 11:36:04 up 1 day, 22:51, 2 users, load average: 0.06, 0.11, 0.09
Tasks: 141 total, 1 running, 139 sleeping, 0 stopped, 1 zombie
Cpu(s): 0.7%us, 0.5%sy, 0.0%ni, 98.8%id, 0.0%wa, 0.0%hi, 0.0%si, 0.0%st
Mem: 1021108k total, 982904k used, 38204k free, 134576k buffers
Swap: 2046968k total, 0k used, 2046968k free, 599576k cached

  PID USER      PR  NI  VIRT  RES  SHR S %CPU  %MEM    TIME+  COMMAND
25454 root        20   0 397m 107m 13m S  2.3 10.8 25:19.18 skype
  269 root        20   0    0    0    0 S   0.3  0.0  0:51.24 scsi_ah_1
    1 root        20   0 2872 1400 1200 S   0.0  0.1  0:00.89 init
    2 root        20   0    0    0    0 S   0.0  0.0  0:00.00 kthreadd
    3 root        20   0    0    0    0 S   0.0  0.0  0:00.16 migration/0
    4 root        20   0    0    0    0 S   0.0  0.0  0:51.23 ksoftirqd/0
    5 root        20   0    0    0    0 S   0.0  0.0  0:00.00 migration/0
    6 root        20   0    0    0    0 S   0.0  0.0  0:00.33 watchdog/0
    7 root        20   0    0    0    0 S   0.0  0.0  0:00.10 migration/1
    8 root        20   0    0    0    0 S   0.0  0.0  0:00.00 migration/1
    9 root        20   0    0    0    0 S   0.0  0.0  6:44.34 ksoftirqd/1
   10 root        20   0    0    0    0 S   0.0  0.0  0:00.27 watchdog/1
   11 root        20   0    0    0    0 S   0.0  0.0  0:04.05 events/0
   12 root        20   0    0    0    0 S   0.0  0.0  0:04.87 events/1
   13 root        20   0    0    0    0 S   0.0  0.0  0:00.00 cgroup
   14 root        20   0    0    0    0 S   0.0  0.0  0:00.00 khelper
   15 root        20   0    0    0    0 S   0.0  0.0  0:00.00 netns
   16 root        20   0    0    0    0 S   0.0  0.0  0:00.00 async/mgr
   17 root        20   0    0    0    0 S   0.0  0.0  0:00.00 pm
   18 root        20   0    0    0    0 S   0.0  0.0  0:00.23 sync_supers
```

2. VmStat-Virtual Memory Statistics:

Linux VmStat command used to display statistics of virtual memory, kernel threads, disks, system processes, I/O blocks, interrupts, CPU activity and much more. By default vmstat command is not available under Linux systems we need to install a package called sysstat that includes a vmstat program. The common usage of command format is.

vmstat

```
# vmstat
procs -----memory----- --swap-- -----io----- --system-- -----cpu-----
r  b   swpd   free   inact active    si   so    bi    bo    in   cs  us  sy  id  wa  st
1  0       0 810420  97380  70628     0    0   115    4   89   79  1  6  90  3  0
```

3. Lsof – List Open Files:

Lsof command used in many Linux/Unix like system that is used to display list of all the open files and the processes. The open files included are disk files, network sockets, pipes, devices and processes. One of the main reason for using this command is when a disk cannot be unmounted and displays the error that files are being used or opened. With this command we can easily identify which files are in use. The most common format for this command is.

```
# lsof
COMMAND      PID      USER   FD   TYPE    DEVICE  SIZE      NODE NAME
init          1      root    cwd   DIR     104,2    4096        2 /
init          1      root   rtd   DIR     104,2    4096        2 /
init          1      root   txt   REG     104,2   38652   17710339 /sbin/init
init          1      root   mem   REG     104,2  129900   196453 /lib/ld-2.5.so
init          1      root   mem   REG     104,2 1693812   196454 /lib/libc-2.5.so
init          1      root   mem   REG     104,2   20668   196479 /lib/libdl-2.5.so
init          1      root   mem   REG     104,2  245376   196419 /lib/libsepol.so.
init          1      root   mem   REG     104,2   93508   196431 /lib/libselinux.s
init          1      root   10u    FIFO     0,17        953 /dev/initctl
```

4. Tcpdump – Network Packet Analyzer:

Tcpdump one of the most widely used command-line network packet analyzer or packets sniffer program that is used capture or filter TCP/IP packets that received or transferred on a specific interface over a network. It also provides an option to save captured packages in a file for later analysis. tcpdump is almost available in all major Linux distributions.

```
# tcpdump -i eth0
tcpdump: verbose output suppressed, use -v or -vv for full protocol decode
listening on eth0, link-type EN10MB (Ethernet), capture size 96 bytes
22:08:59.617628 IP tecmint.com.ssh > 115.113.134.3.static-mumbai.vsnl.net.in.28472: P
22:09:07.653466 IP tecmint.com.ssh > 115.113.134.3.static-mumbai.vsnl.net.in.28472: P
22:08:59.617916 IP 115.113.134.3.static-mumbai.vsnl.net.in.28472 > tecmint.com.ssh: .
```

5. Netstat – Network Statistics:

Netstat is a command line tool for monitoring incoming and outgoing network packets statistics as well as interface statistics. It is very useful tool for every system administrator to monitor network performance and troubleshoot network related problems.

```
# netstat -a | more
Active Internet connections (servers and established)
Proto Recv-Q Send-Q Local Address           Foreign Address         State
tcp        0      0 *:mysql                 *:*                     LISTEN
tcp        0      0 *:sunrpc                 *:*                     LISTEN
tcp        0      0 *:realm-rusd            *:*                     LISTEN
tcp        0      0 *:ftp                   *:*                     LISTEN
tcp        0      0 localhost.localdomain:ipp *:*                     LISTEN
tcp        0      0 localhost.localdomain:smtp *:*                     LISTEN
tcp        0      0 localhost.localdomain:smtp localhost.localdomain:42709 TIME_WAIT
tcp        0      0 localhost.localdomain:smtp localhost.localdomain:42710 TIME_WAIT
tcp        0      0 *:http                  *:*                     LISTEN
tcp        0      0 *:ssh                   *:*                     LISTEN
tcp        0      0 *:https                 *:*                     LISTEN
```

6. Htop – Linux Process Monitoring:

Htop is a much advanced interactive and real time Linux process monitoring tool. This is much similar to Linux top command but it has some rich features like user friendly interface to manage process, shortcut keys, vertical and horizontal view of the processes and much more. Htop is a third party tool and doesn't included in Linux systems, we need to install it using YUM package manager tool. For more information on installation read our article below.

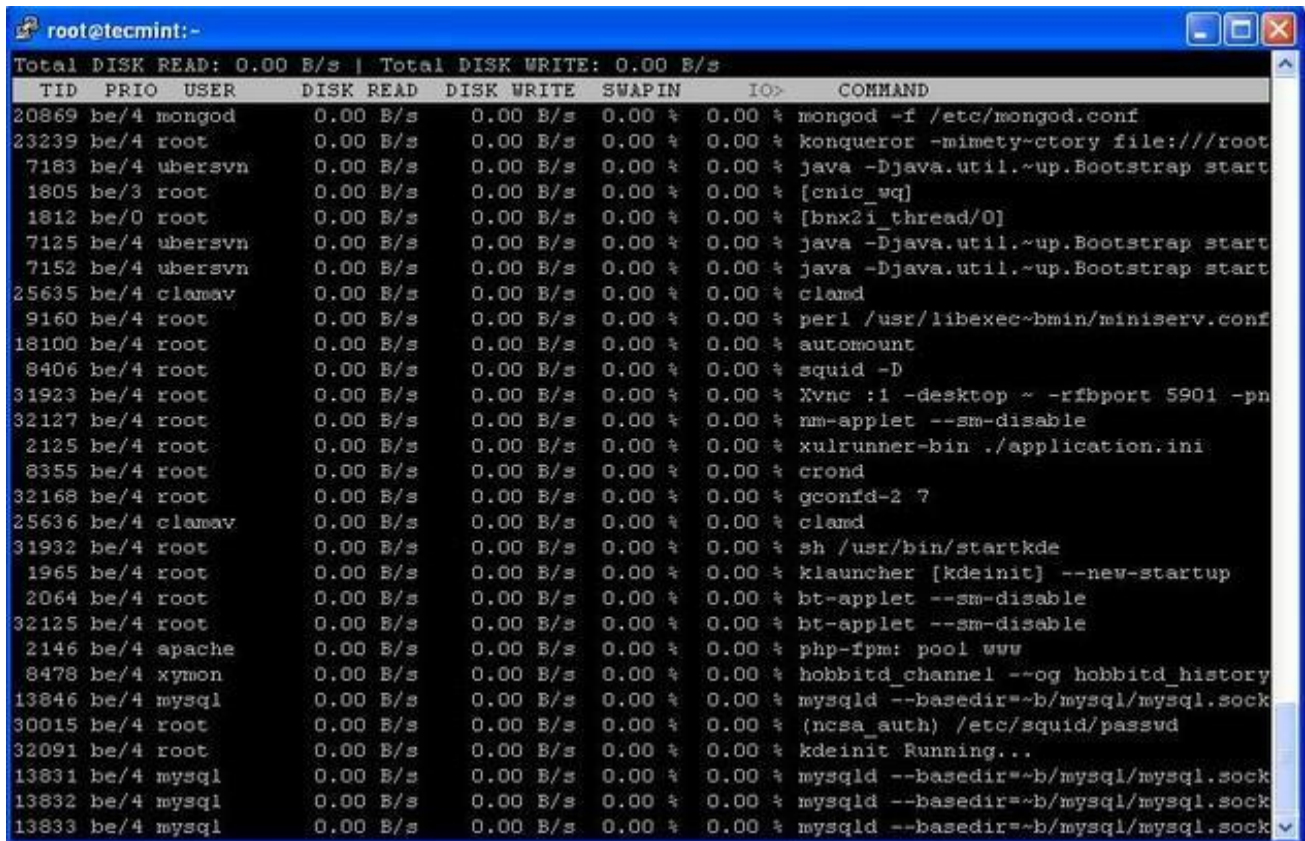
#htop



7. Iotop – Monitor Linux Disk I/O:

Iotop is also much similar to top command and Htop program, but it has accounting function to monitor and display real time Disk I/O and processes. This tool is much useful for finding the exact process and high used disk read/writes of the processes.

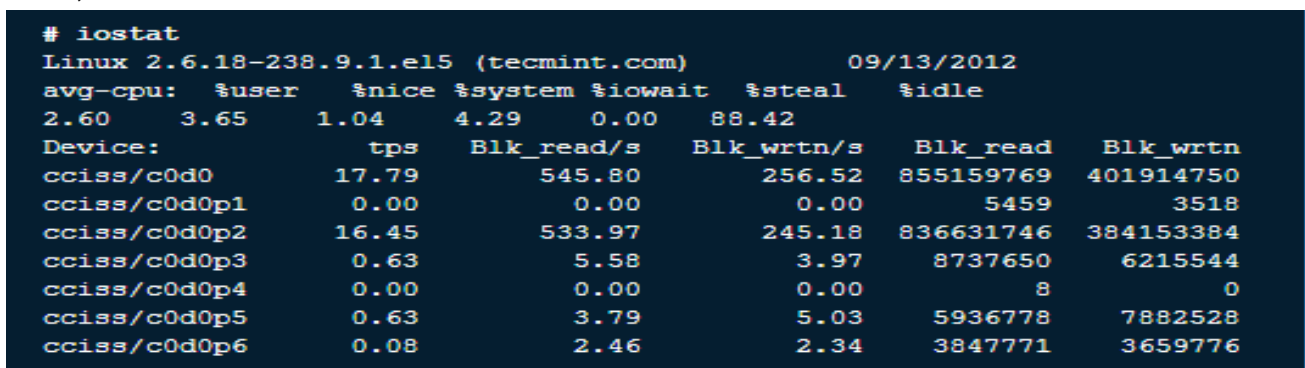
iotop



TID	PRIO	USER	DISK READ	DISK WRITE	SWAPIN	IO>	COMMAND
20869	be/4	mongod	0.00 B/s	0.00 B/s	0.00 %	0.00 %	mongod -f /etc/mongod.conf
23239	be/4	root	0.00 B/s	0.00 B/s	0.00 %	0.00 %	konqueror -mimety~ctory file:///root
7183	be/4	ubersvn	0.00 B/s	0.00 B/s	0.00 %	0.00 %	java -Djava.util.~up.Bootstrap start
1805	be/3	root	0.00 B/s	0.00 B/s	0.00 %	0.00 %	[cnic_wq]
1812	be/0	root	0.00 B/s	0.00 B/s	0.00 %	0.00 %	[bnx2i_thread/0]
7125	be/4	ubersvn	0.00 B/s	0.00 B/s	0.00 %	0.00 %	java -Djava.util.~up.Bootstrap start
7152	be/4	ubersvn	0.00 B/s	0.00 B/s	0.00 %	0.00 %	java -Djava.util.~up.Bootstrap start
25635	be/4	clamav	0.00 B/s	0.00 B/s	0.00 %	0.00 %	clamd
9160	be/4	root	0.00 B/s	0.00 B/s	0.00 %	0.00 %	perl /usr/libexec~bmin/miniserv.conf
18100	be/4	root	0.00 B/s	0.00 B/s	0.00 %	0.00 %	automount
8406	be/4	root	0.00 B/s	0.00 B/s	0.00 %	0.00 %	squid -D
31923	be/4	root	0.00 B/s	0.00 B/s	0.00 %	0.00 %	Xvnc :1 -desktop ~ -rfbport 5901 -pn
32127	be/4	root	0.00 B/s	0.00 B/s	0.00 %	0.00 %	nm-applet --sm-disable
2125	be/4	root	0.00 B/s	0.00 B/s	0.00 %	0.00 %	xulrunner-bin ./application.ini
8355	be/4	root	0.00 B/s	0.00 B/s	0.00 %	0.00 %	crond
32168	be/4	root	0.00 B/s	0.00 B/s	0.00 %	0.00 %	gconfd-2 7
25636	be/4	clamav	0.00 B/s	0.00 B/s	0.00 %	0.00 %	clamd
31932	be/4	root	0.00 B/s	0.00 B/s	0.00 %	0.00 %	sh /usr/bin/startkde
1965	be/4	root	0.00 B/s	0.00 B/s	0.00 %	0.00 %	klauncher [kdeinit] --new-startup
2064	be/4	root	0.00 B/s	0.00 B/s	0.00 %	0.00 %	bt-applet --sm-disable
32125	be/4	root	0.00 B/s	0.00 B/s	0.00 %	0.00 %	bt-applet --sm-disable
2146	be/4	apache	0.00 B/s	0.00 B/s	0.00 %	0.00 %	php-fpm: pool www
8478	be/4	xymon	0.00 B/s	0.00 B/s	0.00 %	0.00 %	hobbitd_channel --og hobbitd_history
13846	be/4	mysql	0.00 B/s	0.00 B/s	0.00 %	0.00 %	mysqld --basedir=~b/mysql/mysql.sock
30015	be/4	root	0.00 B/s	0.00 B/s	0.00 %	0.00 %	(nscsa_auth) /etc/squid/passwd
32091	be/4	root	0.00 B/s	0.00 B/s	0.00 %	0.00 %	kdeinit Running...
13831	be/4	mysql	0.00 B/s	0.00 B/s	0.00 %	0.00 %	mysqld --basedir=~b/mysql/mysql.sock
13832	be/4	mysql	0.00 B/s	0.00 B/s	0.00 %	0.00 %	mysqld --basedir=~b/mysql/mysql.sock
13833	be/4	mysql	0.00 B/s	0.00 B/s	0.00 %	0.00 %	mysqld --basedir=~b/mysql/mysql.sock

8. Iostat – Input/Output Statistics:

IoStat is simple tool that will collect and show system input and output storage device statistics. This tool is often used to trace storage device performance issues including devices, local disks, remote disks such as NFS.



```
# iostat
Linux 2.6.18-238.9.1.el5 (tecmin.com)      09/13/2012
avg-cpu:  %user   %nice %system %iowait  %steal   %idle
2.60      3.65     1.04    4.29     0.00    88.42

Device:            tps    Blk_read/s    Blk_wrtn/s    Blk_read    Blk_wrtn
cciss/c0d0          17.79       545.80        256.52    855159769    401914750
cciss/c0d0p1         0.00         0.00         0.00         5459         3518
cciss/c0d0p2        16.45       533.97        245.18   836631746   384153384
cciss/c0d0p3         0.63         5.58         3.97    8737650    6215544
cciss/c0d0p4         0.00         0.00         0.00         8           0
cciss/c0d0p5         0.63         3.79         5.03   5936778    7882528
cciss/c0d0p6         0.08         2.46         2.34   3847771    3659776
```

9. IPTraf – Real Time IP LAN Monitoring:

IPTraf is an open source console-based real time network (IP LAN) monitoring utility for Linux. It collects a variety of information such as IP traffic monitor that passes over the network, including TCP flag information, ICMP details, TCP/UDP traffic breakdowns, TCP connection packet and byte counts. It also gathers information of general and detailed interface statistics of TCP, UDP, IP, ICMP, non-IP, IP checksum errors, interface activity etc.


```

root@tecmin:~
IPTraf
TCP Connections (Source Host:Port) ----- Packets ----- Bytes Flags Iface
172.16.25.126:22 > 178 39800 -PA- eth0
172.16.25.125:1352 > 94 4696 --A- eth0

TCP: 1 entries ----- Active

UDP (279 bytes) from 172.16.24.33:5353 to 224.0.0.251:5353 on eth0
UDP (229 bytes) from 172.16.16.52:138 to 172.16.31.255:138 on eth0
UDP (229 bytes) from 172.16.16.87:138 to 172.16.31.255:138 on eth0
UDP (279 bytes) from 172.16.24.33:5353 to 224.0.0.251:5353 on eth0
UDP (229 bytes) from 172.16.19.124:138 to 172.16.31.255:138 on eth0
Bottom ----- Elapsed time: 0:00 -----
Pkts captured (all interfaces): 325 | TCP flow rate: 30.00 kbits/s
Up/Dn/PgUp/PgDn-scroll M-more TCP info W-chg actv win S-sort TCP X-exit

```

10. Monit – Linux Process and Services Monitoring:

Monit is a free open source and web based process supervision utility that automatically monitors and manages system processes, programs, files, directories, permissions, checksums and filesystems. It monitors services like Apache, MySQL, Mail, FTP, ProFTP, Nginx, SSH and so on. The system status can be viewed from the command line or using its own web interface.

tecmin.com Monit - Mozilla Firefox

File Edit View History Bookmarks Tools Help

tecmin.com Monit

localhost:2812

Most Visited Red Hat Customer Portal Documentation Red Hat Network

Home> Running Monit on more than one server? Use [M/Monit](#) to manage all your Monit instances monit 5.1.1

http://www.tecmint.com

Monit Service Manager

Monit is running on tecmint.com with uptime, 5m and monitoring:

System	Status	Load	CPU	Memory
tecmin.com	running	[0.10] [0.07] [0.04]	5.0%us, 0.9%sy, 0.9%wa	34.6% [354012 kB]

Process	Status	Uptime	CPU	Memory
sshd	running	1d 2h 42m	0.0%	0.0% [536 kB]
mysqld	running	1d 2h 42m	0.4%	2.8% [29068 kB]
proftpd	running	23m	0.0%	0.1% [1748 kB]

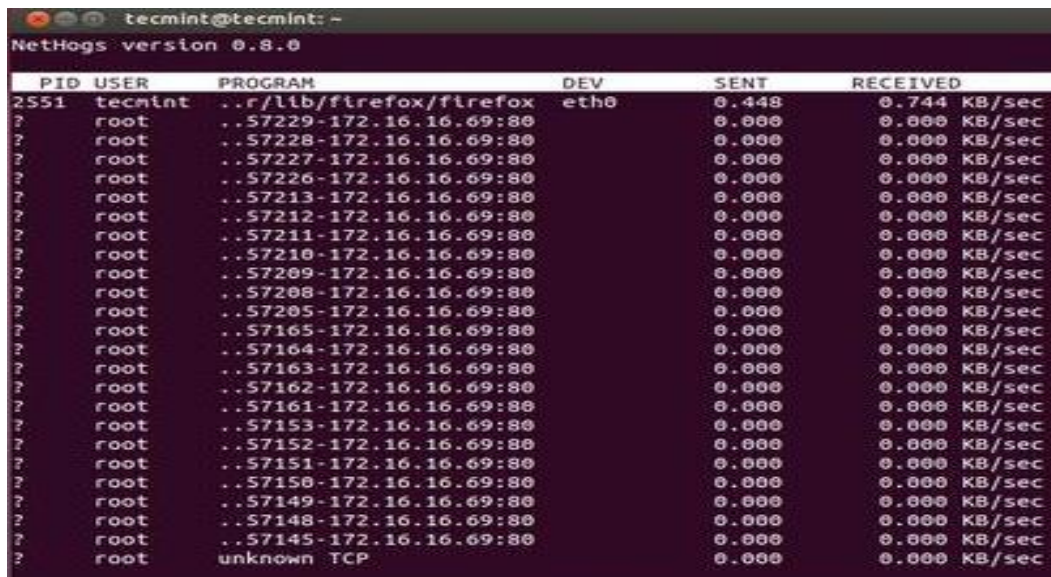
11.Psacct or Acct – Monitor User Activity:

psacct or acct tools are very useful for monitoring each users activity on the system. Both daemons runs in the background and keeps a close watch on the overall activity of each user on the system and also what resources are being consumed by them.

These tools are very useful for system administrators to track each users activity like what they are doing, what commands they issued, how much resources are used by them, how long they are active on the system etc.

12.NetHogs – Monitor Per Process Network Bandwidth:

NetHogs is an open source nice small program (similar to Linux top command) that keeps a tab on each process network activity on your system. It also keeps a track of real time network traffic bandwidth used by each program or application.



NetHogs version 0.8.0

PID	USER	PROGRAM	DEV	SENT	RECEIVED
2551	tecmin	./r/lib/firefox/firefox	eth0	0.448	0.744 KB/sec
?	root	..57229-172.16.16.69:80		0.000	0.000 KB/sec
?	root	..57228-172.16.16.69:80		0.000	0.000 KB/sec
?	root	..57227-172.16.16.69:80		0.000	0.000 KB/sec
?	root	..57226-172.16.16.69:80		0.000	0.000 KB/sec
?	root	..57213-172.16.16.69:80		0.000	0.000 KB/sec
?	root	..57212-172.16.16.69:80		0.000	0.000 KB/sec
?	root	..57211-172.16.16.69:80		0.000	0.000 KB/sec
?	root	..57210-172.16.16.69:80		0.000	0.000 KB/sec
?	root	..57209-172.16.16.69:80		0.000	0.000 KB/sec
?	root	..57208-172.16.16.69:80		0.000	0.000 KB/sec
?	root	..57205-172.16.16.69:80		0.000	0.000 KB/sec
?	root	..57165-172.16.16.69:80		0.000	0.000 KB/sec
?	root	..57164-172.16.16.69:80		0.000	0.000 KB/sec
?	root	..57163-172.16.16.69:80		0.000	0.000 KB/sec
?	root	..57162-172.16.16.69:80		0.000	0.000 KB/sec
?	root	..57161-172.16.16.69:80		0.000	0.000 KB/sec
?	root	..57153-172.16.16.69:80		0.000	0.000 KB/sec
?	root	..57152-172.16.16.69:80		0.000	0.000 KB/sec
?	root	..57151-172.16.16.69:80		0.000	0.000 KB/sec
?	root	..57150-172.16.16.69:80		0.000	0.000 KB/sec
?	root	..57149-172.16.16.69:80		0.000	0.000 KB/sec
?	root	..57148-172.16.16.69:80		0.000	0.000 KB/sec
?	root	..57145-172.16.16.69:80		0.000	0.000 KB/sec
?	root	unknown TCP		0.000	0.000 KB/sec

13.iftop – Network Bandwidth Monitoring:

iftop is another terminal-based free open source system monitoring utility that displays a frequently updated list of network bandwidth utilization (source and destination hosts) that passing through the network interface on our system. iftop is considered for network usage, what 'top' does for CPU usage. iftop is a 'top' family tool that monitor a selected interface and displays a current bandwidth usage between two hosts.

```

root@tecmin: - iftop - 0.17
12.5Kb 25.0Kb http://www.tecmint.com 62.5Kb
172.16.25.126 ==> 172.16.25.125 2.67Kb 2.84Kb 2.84Kb
<== 160b 240b 240b
172.16.25.126 ==> mddc-01.midcorp.mid-d 284b 861b 861b
<== 592b 1.50Kb 1.50Kb
172.16.31.255 ==> 172.16.23.185 0b 0b 0b
<== 0b 240b 240b
172.16.31.255 ==> 172.16.22.152 0b 0b 0b
<== 0b 229b 229b
172.16.31.255 ==> wsus.midcorp.mid-day. 0b 0b 0b
<== 0b 229b 229b
172.16.31.255 ==> 172.16.21.79 0b 0b 0b
<== 916b 229b 229b
172.16.31.255 ==> 172.16.18.159 0b 0b 0b
<== 0b 156b 156b
255.255.255.255 ==> 172.16.18.9 0b 0b 0b
<== 0b 68b 68b

TX:  curm:  3.68KB  peak:  rates:  2.95Kb  3.68Kb  3.68Kb
RX:  2.93KB  4.84Kb  1.63Kb  2.93Kb  2.93Kb
TOTAL:  6.60KB  9.34Kb  4.58Kb  6.60Kb  6.60Kb

```

14. Monitorix – System and Network Monitoring:

Monitorix is a free lightweight utility that is designed to run and monitor system and network resources as many as possible in Linux/Unix servers. It has a built in HTTP web server that regularly collects system and network information and display them in graphs. It Monitors system load average and usage, memory allocation, disk driver health, system services, network ports, mail statistics(Sendmail, Postfix, Dovecot, etc.), MySQL statistics and many more. It designed to monitor overall system performance and helps in detecting failures, bottlenecks, abnormal activities etc.



15. Arpwatch – Ethernet Activity Monitor:

Arpwatch is a kind of program that is designed to monitor Address Resolution (MAC and IP address changes) of Ethernet network traffic on a Linux network. It continuously keeps watch on Ethernet traffic and produces a log of IP and MAC address pair changes along with a timestamps on a network. It also has a feature to send an email alerts to administrator, when a pairing added or changes. It is very useful in detecting ARP spoofing on a network.

16. Suricata – Network Security Monitoring:

Suricata is a high performance open source Network Security and Intrusion Detection and Prevention Monitoring System for Linux, FreeBSD and Windows. It was designed and owned by a non-profit foundation OISF (Open Information Security Foundation).

17.VnStat PHP – Monitoring Network Bandwidth:

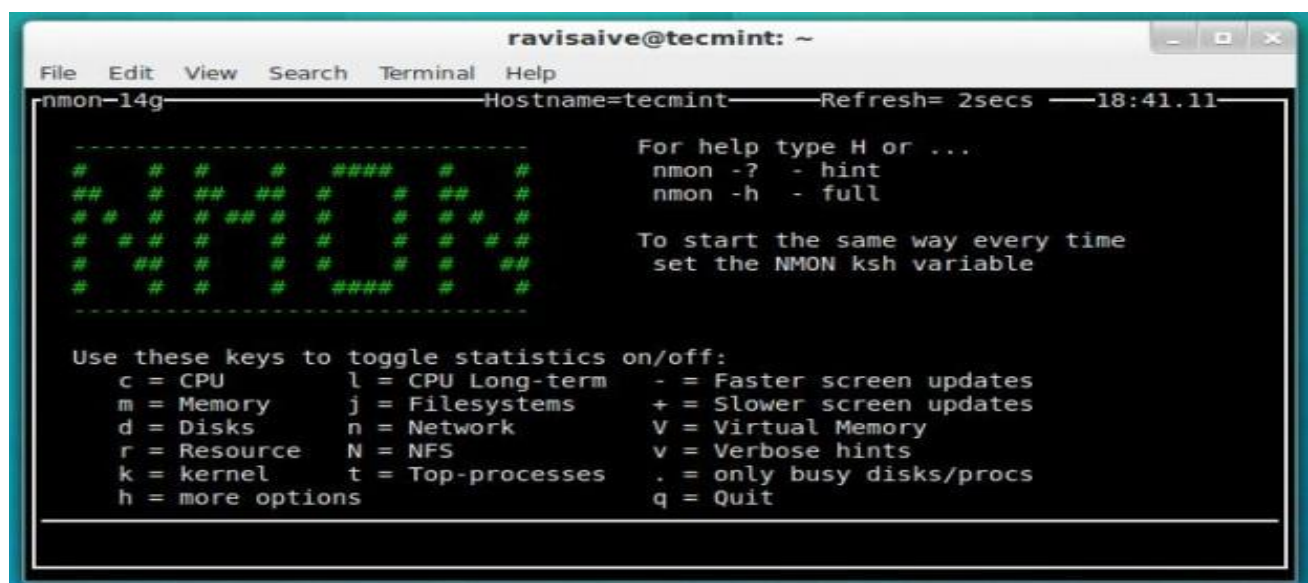
VnStat PHP a web based frontend application for most popular networking tool called “vnstat“. VnStat PHP monitors a network traffic usage in nicely graphical mode. It displays a total IN and OUT network traffic usage in hourly, daily, monthly and full summary report.

18.Nagios – Network/Server Monitoring:

Nagios is a leading open source powerful monitoring system that enables network/system administrators to identify and resolve server related problems before they affect major business processes. With the Nagios system, administrators can able to monitor remote Linux, Windows, Switches, Routers and Printers on a single window. It shows critical warnings and indicates if something went wrong in our network/server which indirectly helps us to begin remediation processes before they occur.

19.Nmon-Monitor Linux Performance:

Nmon (stands for Nigel’s performance Monitor) tool, which is used to monitor all Linux resources such as CPU, Memory, Disk Usage, Network, Top processes, NFS, Kernel and much more. This tool comes in two modes: Online Mode and Capture Mode. The Online Mode, is used for real-time monitoring and Capture Mode, is used to store the output in CSV format for later processing.



```
ravisaive@tecmint: ~  
File Edit View Search Terminal Help  
nmon-14g-----Hostname=tecmint-----Refresh= 2secs -----18:41.11  
  
# # # # ##### # #  
## # ## ## # # ## #  
## # ## ## # # ## #  
# # # # # # # # # #  
# ## # # # # # ##  
# # # # # ##### # #  
-----  
  
For help type H or ...  
nmon -? - hint  
nmon -h - full  
  
To start the same way every time  
set the NMON ksh variable  
  
Use these keys to toggle statistics on/off:  
c = CPU          l = CPU Long-term      - = Faster screen updates  
m = Memory       j = Filesystems      + = Slower screen updates  
d = Disks        n = Network          V = Virtual Memory  
r = Resource     N = NFS              v = Verbose hints  
k = kernel       t = Top-processes    . = only busy disks/procs  
h = more options q = Quit
```

20.w – Find out who is logged on and what they are doing

w command displays information about the users currently on the machine, and their processes.

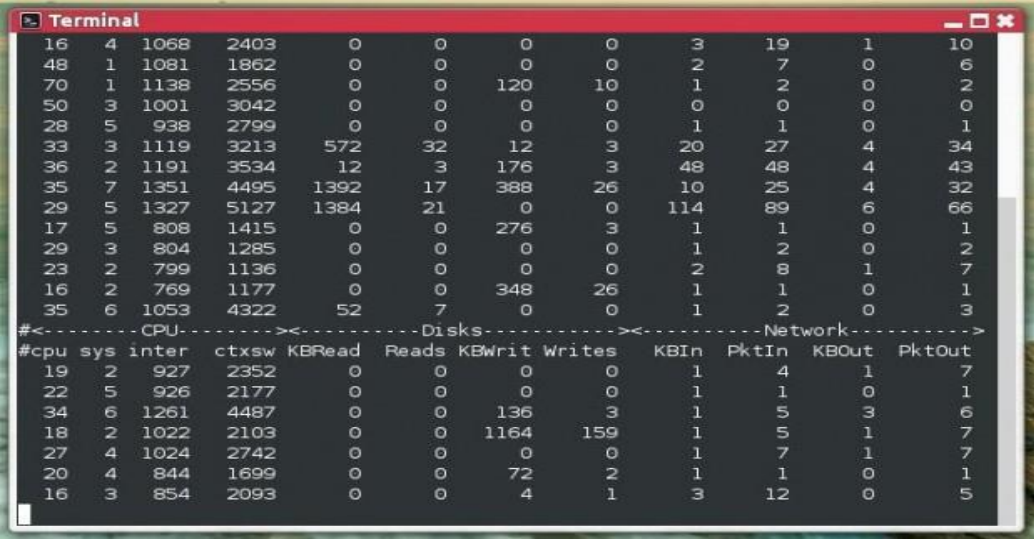
w username

w vivek

17:58:47 up 5 days, 20:28, 2 users, load average: 0.36, 0.26, 0.24						
USER	TTY	FROM	LOGIN@	IDLE	JCPU	PCPU WHAT
root	pts/0	10.1.3.145	14:55	5.00s	0.04s	0.02s vim /et
root	pts/1	10.1.3.145	17:43	0.00s	0.03s	0.00s w

21. Collectl: All-in-One Performance Monitoring Tool

Collectl is a yet another powerful and feature rich command line based utility, that can be used to gather information about Linux system resources such as CPU usage, memory, network, inodes, processes, nfs, tcp, sockets and much more.



CPU											
#	cpu	sys	inter	ctxsw	KBRead	Reads	KBWrit	Writes	KBIn	PktIn	PktOut
16	4	1068	2403	0	0	0	0	3	19	1	10
48	1	1081	1862	0	0	0	0	2	7	0	6
70	1	1138	2556	0	0	120	10	1	2	0	2
50	3	1001	3042	0	0	0	0	0	0	0	0
28	5	938	2799	0	0	0	0	1	1	0	1
33	3	1119	3213	572	32	12	3	20	27	4	34
36	2	1191	3534	12	3	176	3	48	48	4	43
35	7	1351	4495	1392	17	388	26	10	25	4	32
29	5	1327	5127	1384	21	0	0	114	89	6	66
17	5	808	1415	0	0	276	3	1	1	0	1
29	3	804	1285	0	0	0	0	1	2	0	2
23	2	799	1136	0	0	0	0	2	8	1	7
16	2	769	1177	0	0	348	26	1	1	0	1
35	6	1053	4322	52	7	0	0	1	2	0	3

Disks											
#	cpu	sys	inter	ctxsw	KBRead	Reads	KBWrit	Writes	KBIn	PktIn	PktOut
19	2	927	2352	0	0	0	0	0	1	4	7
22	5	926	2177	0	0	0	0	0	1	1	1
34	6	1261	4487	0	0	136	3	1	5	3	6
18	2	1022	2103	0	0	1164	159	1	5	1	7
27	4	1024	2742	0	0	0	0	1	7	1	7
20	4	844	1699	0	0	72	2	1	1	0	1
16	3	854	2093	0	0	4	1	3	12	0	5

Network											
#	cpu	sys	inter	ctxsw	KBRead	Reads	KBWrit	Writes	KBIn	PktIn	PktOut
19	2	927	2352	0	0	0	0	0	1	4	7
22	5	926	2177	0	0	0	0	0	1	1	1
34	6	1261	4487	0	0	136	3	1	5	3	6
18	2	1022	2103	0	0	1164	159	1	5	1	7
27	4	1024	2742	0	0	0	0	1	7	1	7
20	4	844	1699	0	0	72	2	1	1	0	1
16	3	854	2093	0	0	4	1	3	12	0	5

22. uptime – Tell How Long The Linux System Has Been Running:

uptime command can be used to see how long the server has been running. The current time, how long the system has been running, how many users are currently logged on, and the system load averages for the past 1, 5, and 15 minutes.

```
# uptime
```

```
18:02:41 up 41 days, 23:42, 1 user, load average: 0.00, 0.00, 0.00
```

WORKING WITH FILE SYSTEM:

Linux file structure is a tree like structure. It starts from the root directory, represented by '/', and then expands into sub-directories. All the partitions are under the root directory. If a partition is mounted (The mount point defines the place of a particular data set in the file system) anywhere apart from a "device", the system is not aware of the existence of that partition or device. Directories that are only one level below the root directory are often preceded by a slash, to indicate their position.

Root "/" File System:

The kernel needs a root file system to mount at start up. The root file system is generally small and should not be changed often as it may interrupt in booting. The root directory usually does not have the critical files. Instead sub directories are created. E.g. /bin (commands needed during boot up), /etc (config files) , /lib(shared libraries).

/usr File System:

This file system is generally large as it contains the executable files to be shared amongst different machines. Files are usually the ones installed while installing Linux. This makes it possible to update the system from a new version of the distribution, or even a completely new distribution, without having to install all programs again. Sub directories include /bin, /include, /lib, /local (for local executable).

/var File System:

This file system is specific to local systems. It is called as var because the data keeps changing. The sub directories include /cache/man (A cache for man pages), /games (any variable data belong to games), /lib (files that change), /log (log from different programs), /tmp (for temporary files).

/home File System:

This file system differs from host to host. User specific configuration files for applications are stored in the user's home directory in a file. UNIX creates directories for all users directory. E.g. /home/my_name. Once the user is logged in; he is placed in his home directory.

/proc File System:

This file system does not exist on the hard disk. It is created by the kernel in its memory to provide information about the system. This information is usually about the processes. Contains a hierarchy of special files which represent the current state of the kernel. Few of the Directories include /1 (directory with information about process num 1, where 1 is the identification number), /cpuinfo (information about cpu), /devices (information about devices installed), /filesystem (file systems configured), /net (information about network protocols), /mem (memory usage).

At the time of installation of Linux, a file system is assigned and persists in the hard disk. This file system structure resembles a tree. A file can be a list of names and numbers or executable programs. Linux treats every program as a file. Linux treats directories and computer components also as files.

A file could be a list of names and numbers, a cheesecake recipe, or an executable program. But under Linux, everything is a file. In addition to data and executable files, Linux treats directories and even the various components of our computer as files. It could be a keyboard, console, and printer, RAM or ROM. These are referred as special files known as devices. These files are available in /dev directory. Linux performs the communication with these devices by simply reading from or writing to these special files.

WORKING WITH FILES AND FOLDERS:

Determining The Type Of A File:

Before working with a file, we need first to determine the type of the file: whether it is a regular text file, binary executable, directory, soft link, block special file, character special file, etc. For this purpose, the file command was written.


```
#
# file /etc/passwd
/etc/passwd: ASCII text
#
# file /etc
/etc: directory
#
# file /dev/sda
/dev/sda: block special
#
# file /dev/tty
/dev/tty: character special
#
# file /bin/view
/bin/view: symbolic link to `vi'
#
```

🌈 Listing the Contents of a Directory:

After determining where we are (the current directory) using the `pwd` command, and changing to a directory (if we need to do) using the `cd` command, we need to display the contents of a directory. The `ls` command does this for us.

Syntax:

`ls [OPTIONS] [ARGUMENTS]`

Examples:

- To list the contents of the current directory, use `ls` without arguments.

```
# ls
authconfig      firewallld      kernel          network-scripts  sshd
cbq             grub           man-db         rdisc            wpa_supplicant
console        init           modules        readonly-root
crond          ip6tables-config netconsole      run-parts
ebtables-config iptables-config network         selinux
#
```

- To list the contents of the directory called `network-scripts`:

```
# ls network-scripts/
ifcfg-lo      ifdown-ipv6    ifup           ifup-isdn      ifup-tunnel
ifcfg-p2p1    ifdown-isdn    ifup-aliases   ifup-plip      ifup-wireless
ifcfg-p7p1    ifdown-post    ifup-bnep      ifup-plusb     init.ipv6-global
ifdown        ifdown-ppp     ifup-eth       ifup-post      network-functions
ifdown-bnep   ifdown-routes  ifup-ippv      ifup-ppp       network-functions-ipv6
ifdown-eth    ifdown-sit     ifup-ipv6      ifup-routes
ifdown-ippv   ifdown-tunnel  ifup-ipx       ifup-sit
#
```

- To see a long listing of the `/etc/ssh` directory, use the `-l` option (small L for long)

```
#
# ls -l /etc/ssh/
total 260
-rw-----. 1 root root    242153 Nov  1  2013 moduli
-rw-r--r--. 1 root root     2123 Nov  1  2013 ssh_config
-rw-----. 1 root root     4425 Mar 30  2015 sshd_config
-rw-r-----. 1 root ssh_keys 1679 Mar 30  2015 ssh_host_rsa_key
-rw-r--r--. 1 root root      382 Mar 30  2015 ssh_host_rsa_key.pub
#
#
```

- To list the files with human-readable file sizes, use the `-h` option.

```
#
# ls -lh /etc/ssh
total 260K
-rw-----. 1 root root      237K Nov  1  2013 moduli
-rw-r--r--. 1 root root      2.1K Nov  1  2013 ssh_config
-rw-----. 1 root root      4.4K Mar 30  2015 sshd_config
-rw-r-----. 1 root ssh_keys 1.7K Mar 30  2015 ssh_host_rsa_key
-rw-r--r--. 1 root root       382 Mar 30  2015 ssh_host_rsa_key.pub
#
```

- To list the files, sorted by their last modification date/time (from oldest to newest), use the `-ltr` option.

```
# ls -ltr /var/log
total 72
drwx-----. 2 root root      4096 Aug  1  2013 ppp
-rw-r--r--. 1 root root      1040 Dec  5  2013 README
-rw-----. 1 root root         0 Mar 30  2015 tallylog
drwxr-xr-x. 2 root root      4096 Mar 30  2015 anaconda
drwxr-sr-x+ 3 root systemd-journal 4096 Mar 30  2015 journal
drwxr-x---. 2 root root      4096 Mar 30  2015 audit
-rw-----. 1 root utmp        384 Mar 30  2015 btmp
-rw-r--r--. 1 root root      6341 Apr 12 05:46 boot.log
-rw-rw-r--. 1 root utmp     17280 Apr 12 05:47 wtmp
-rw-r--r--. 1 root root    292000 Apr 12 05:47 lastlog
#
```

- To list all files, use the `-a` option:

```
# ls
anaconda-ks.cfg
#
# ls -a
.   anaconda-ks.cfg  .bash_logout  .bashrc  .local
..  .bash_history      .bash_profile .cshrc    .tcshrc
#
#
```

Note: Files and directories whose names start with a dot (.) are considered hidden; they are not listed by default.

- To list directories recursively, use the `-R` option.

```
# ls -R /etc/sysconfig/
/etc/sysconfig/:
authconfig      firewallld      kernel          network-scripts  sshd
cbq             grub            man-db          rdisc            wpa_supplicant
console         init            modules         readonly-root    run-parts
cron            iptables-config netconsole      selinux
ebtables-config iptables-config network

/etc/sysconfig/cbq:
avpkt  cbq-0000.example

/etc/sysconfig/console:

/etc/sysconfig/modules:

/etc/sysconfig/network-scripts:
ifcfg-lo      ifdown-ipv6      ifup           ifup-isdn      ifup-tunnel
ifcfg-p2p1    ifdown-isdn      ifup-aliases  ifup-plip      ifup-wireless
ifcfg-p7p1    ifdown-post      ifup-bnep     ifup-plusb     init.ipv6-global
ifdown        ifdown-ppp       ifup-eth      ifup-post      network-functions
ifdown-bnep   ifdown-routes   ifup-ipppp    ifup-ppp       network-functions-ipv6
ifdown-eth    ifdown-sit       ifup-ipv6     ifup-routes
ifdown-ippp   ifdown-tunnel   ifup-ipvx     ifup-sit
#
```

- To display info about the directory itself (not its contents), use the `-ld` options.

```
# ls -ld /etc/sysconfig/
drwxr-xr-x. 6 root root 4096 Mar 30 2015 /etc/sysconfig/
#
```

➤ Creating a Directory:

To create a directory, use the command `mkdir` (make directory).

Syntax:

`mkdir [DIRNAME] ...`

Examples:

- To create a directory named `newdata` under `/opt`:

```
# mkdir /opt/newdata
#
# ls -ld /opt/newdata/
drwxr-xr-x. 2 root root 4096 Apr 12 08:23 /opt/newdata/
#
```

- To create multiple directories with names Jan, Feb, Mar, ..., Dec

```
# mkdir Jan Feb Mar Apr May Jun Jul Aug Sep Oct Nov Dec
#
# ls
anaconda-ks.cfg Apr Aug Dec Feb Jan Jul Jun Mar May Nov Oct Sep
#
```

- To create a directory and create its parent(s) if they don't exist, use the `-p` option.

```
# mkdir -p 2016/April/10
#
# ls -ld 2016/April/10/
drwxr-xr-x. 2 root root 4096 Apr 12 08:29 2016/April/10/
#
# ls -R 2016/
2016/:
April

2016/April:
10

2016/April/10:
#
```

- If we try to execute this command without `-p`, the shell will complain, firing the following error:

```
# mkdir 2016/April/10
mkdir: cannot create directory '2016/April/10': No such file or directory
#
```

➤ Removing an Empty Directory:

To remove one or more empty directories (that has no files or sub-directories inside), use the `rmdir` command.

```
# rmdir Sep Oct Nov
#
# ls
2016  anaconda-ks.cfg  Apr  Aug  Dec  Feb  Jan  Jul  Jun  Mar  May
#
```

➤ If we try this command with a non-empty directory, we will get an error:

```
# rmdir 2016
rmdir: failed to remove '2016': Directory not empty
#
```

🌈 Removing Any Directory:

➤ To remove non-empty (and empty) directories, use the `rm -r` command.

```
# rm -r 2016/
rm: descend into directory '2016/'? y
rm: descend into directory '2016/April'? y
rm: remove directory '2016/April/10'? y
rm: remove directory '2016/April'? y
rm: remove directory '2016/'? y
#
```

Note: To enforce deletion without prompting, use the `-f` option with the `rm` command.

🌈 Creating Files:

There are several ways to create files. The one we are going to discuss in this section is using the `touch` command.

Syntax:

`touch [FILENAME] ...`

Examples:

➤ The following will create an empty file `/opt/test1.txt`

```
# touch /opt/test1.txt
#
# ls -l /opt/test1.txt
-rw-r--r--. 1 root root 0 Apr 12 08:53 /opt/test1.txt
#
```

➤ To create files `day00`, `day01`, ..., `day31`:

```
# touch day{00..31}
# ls
day00  day03  day06  day09  day12  day15  day18  day21  day24  day27  day30
day01  day04  day07  day10  day13  day16  day19  day22  day25  day28  day31
day02  day05  day08  day11  day14  day17  day20  day23  day26  day29
#
```

Copying Files and Directories:

To copy one or more files from a given location (source) to another (destination), the `cp` command is used.

- `cp FILE1 FILE2`
- `cp FILE1 DESTINATION/`
- `cp FILE1 FILE2 ... FILEn DESTINATION/`

The first form of the command copies a file from one location to another using a new name for the resulting copy. The second form copies the file to a destination directory. The new file at the destination directory will have the same name as the original one. The third form copies several files to a destination directory.

Note: *If multiple source files are to be copied at a time, the destination MUST be a directory.*

Coping a Directory

- To copy a directory, use the same command `cp` with `-r` (recursive) option

Moving and Renaming Files and Directories:

When copying files and directories, the source file(s) and/or directories remain unaffected. On the other hand, moving files and directories will remove the source(s). We can think of it like copying then removing.

- The `mv` command moves a file or directory from its current location to another. If the `mv FILE1 FILE2 ... FILEn DESTINATION/` This will move one or more files from its/their current location to (another) destination directory.
- `mv OLDNAME NEWNAME` This renames a file or directory.

Removing Files:

- The `rm` command is used to remove files.

ABSOLUTE AND RELATIVE PATH:

Absolute paths specify a location (file or directory) in relation to the root directory. We can identify them easily as they always begin with a forward slash (/). Relative paths specify a location (file or directory) in relation to where we currently are in the system. They will not begin with a slash.

```
Terminal
1. user@bash: pwd
2. /home/ryan
3. user@bash:
4. user@bash: ls Documents
5. file1.txt file2.txt file3.txt
6. ...
7. user@bash: ls /home/ryan/Documents
8. file1.txt file2.txt file3.txt
9. ...
10. user@bash:
```

Line 1: We ran `pwd` just to verify where we currently are.

Line 4: We ran `ls` providing it with a relative path. `Documents` is a directory in our current location. This command could produce different results depending on where we are. If we had another user on the system, `bob`, and we ran the command when in their home directory then we would list the contents of their `Documents` directory instead.

Line 7: We ran `ls` providing it with an absolute path. This command will provide the same output regardless of our current location when we run it.

More on Paths:

We'll find that a lot of stuff in Linux can be achieved in several different ways. Paths are no different. Here are some more building blocks we may use to help build our paths.

~ (tilde): This is a shortcut for our home directory. E.g. if our home directory is `/home/ryan` then we could refer to the directory `Documents` with the path `/home/ryan/Documents` or `~/Documents`

.(dot): This is a reference to our current directory. E.g. in the example above we referred to `Documents` on line 4 with a relative path. It could also be written as `./Documents`.

.. (dotdot): This is a reference to the parent directory. We can use this several times in a path to keep going up the hierarchy. E.g. if we were in the path `/home/ryan` we could run the command `ls ../../` and this would do a listing of the root directory.

```
Terminal
1. user@bash: pwd
2. /home/ryan
3. user@bash:
4. user@bash: ls ~/Documents
5. file1.txt file2.txt file3.txt
6. ...
7. user@bash: ls ./Documents
8. file1.txt file2.txt file3.txt
9. ...
10. user@bash: ls /home/ryan/Documents
11. file1.txt file2.txt file3.txt
12. ...
13. user@bash:
14. user@bash: ls ../../
15. bin boot dev etc home lib var
16. ...
17. user@bash:
18. user@bash: ls /
19. bin boot dev etc home lib var
20. ...
```