

```
[root@HostName ~]#  
[root@HostName ~]#  
[root@HostName ~]#  
[root@HostName ~]#  
[root@HostName ~]#  
[root@HostName ~]#  
[root@HostName ~]#  
[root@HostName ~]#  
[root@HostName ~]# echo -e "  Discovering Linux\n - By  
Aniket Kakde(Associate Software Engineer at Datamato)"
```

# “Discovering Linux”

-By Aniket Kakde (Associate Software Engineer at DATAMATO)

(2023)

# Preface

Hello! Having worked in the industry for more than a year and a half, I can honestly say that Linux is phenomenal! There are a number of places where it is used because of its versatility. During my time in the field, I have gained a great deal of knowledge about Linux, and I am very eager to share my knowledge with beginners who are interested in learning Linux. Exactly that is the focus of this book!

# Index

1. Introduction to Linux.
2. Linux File System.
3. Command Line Basics.
4. Package Management.
5. User and permissions.
6. Text Editors.
7. Networking.
8. System Monitoring.
9. Shell Scripting.

# Introduction to Linux.

(Linux was developed by Linus Torvalds, on 25<sup>th</sup> August 1991.)

## ● What is Linux?

Linux is a type of computer operating system, which is basically like the "brain" of your computer that manages all its hardware and software. The difference with Linux is that it's completely free and open-source, meaning anyone can access and modify its code. This makes it very flexible and customizable.

Linux is used in many different types of devices, from servers and supercomputers to smartphones and home appliances. It's known for being stable and secure, which is why it's popular for running websites, databases, and other important applications.

There are many different versions of Linux, each with its own unique features and tools. Some are more user-friendly and similar to other operating systems like Windows or macOS, while others are more geared towards advanced users who want more control and customization options.

Overall, Linux is a powerful and versatile operating system that can be used for many different purposes, and it's a great option for those who value open-source software and want more control over their computing experience.

## ● How to Install Linux.

If you're interested in installing Linux on your Windows system, there are a few methods you can try out. One way is to dual boot your system, allowing both Linux and Windows to coexist. Another option is to install VirtualBox on your Windows machine and use it to boot a Linux ISO image. Alternatively, you can opt for an easier route by installing Git Bash, which will enable you to run basic Linux commands on your Windows system.

(Note:- Please be advised that Git Bash is limited in its support for Linux commands, and only basic ones are available. If you wish to practice using a wider range of Linux commands, you will need to use an actual Linux system.)

**Key Point:-** The Linux philosophy emphasizes on collaboration, openness, and flexibility, with a focus on creating a secure, efficient, and customizable operating system.

**Openness:** Linux is an open-source operating system, which means that the source code is available for anyone to access, modify, and redistribute.

**Collaboration:** The Linux community is built on collaboration, with developers from all over the world working together to improve the operating system.

**Modularity:** Linux is designed to be modular, meaning that it is made up of many different components that can be easily replaced or updated without affecting the overall system.

**Flexibility:** Linux is highly flexible and can be customized to meet the specific needs of users or organizations.

**Security:** Linux is known for its strong security features, including built-in firewalls, encryption, and user account controls.

**Efficiency:** Linux is designed to be efficient and lightweight, with minimal system requirements and low resource usage.

**Freedom:** Linux is free and can be used, modified, and distributed without any restrictions or licensing fees.

#### ● Linux vs Windows --> What' s the Difference?

Linux	Windows
Uses a tree-like hierarchical file system.	Uses different data drives like C: D: E to store files and folders.
There are no drives in Linux.	Has different drives like C: D: E.
Considers peripherals like hard drives, CD-ROMs, printers as files.	Considers hard drives, CD-ROMs, printers as devices.
Has 3 types of user account types: Regular, Root, and Service Account.	Has 4 types of user account types: Administrator, Standard, Child, and Guest.
Root user is the superuser and has all administrative privileges.	Administrator user has all administrative privileges of computers.
File naming convention is case-sensitive in Linux/Unix, thus "sample" and "SAMPLE" are two different files.	Cannot have 2 files with the same name in the same folder.
For every user, the directory	Default home directory is "My

<code>/home/username</code> is created, which is called their home directory.	Documents.”
--	-------------

- **Some of famous Linux distribution.**

- Ubuntu
- CentOS
- RedHat (RHEL)
- Arch Linux

# Linux File System.

(Linux file system is a hierarchical tree-like structure that organizes files and directories on a Linux operating system)

- What is linux file system?

It is set of process which decides how and where is data stored and retrieve from the data device.

- Types of Linux File System.

1. Ext, Ext2, Ext3 and Ext4 (Extended File System)

Type	Description
Ext	It uses unix like file system hierarchy, with separate file for system file, user file, device files etc. but major drawback of it is that it does not support journaling, which means that if a system crash occurs, data can be lost or corrupted.
Ext2	updated version of ext with better performance, faster disk access and support for large files.
Ext3	It includes journaling support, which improves reliability and data consistency in the event of a system crash.
Ext4	It includes many new features such as support for larger file sizes (up to 16 terabytes), faster disk access, and better performance. It also includes enhanced journaling support.

2. JFS (Journaled File System)

JFS is a high-performance, journaling file system that offers advanced features such as fast recovery after system crashes, data integrity, and support for large files and file systems. It is based

on B+ tree structure which allows faster accessing of file and effective file lookups.

### **3. Raiser file system.**

ReiserFS is a type of file system used in Linux. It uses a special way to organize files and folders, called a balanced tree structure or B+ tree. This helps it find files faster and makes it more efficient. One of the unique features of ReiserFS is that it can also record changes to file information like who owns the file, when it was created, and what permissions it has. This helps make sure that everything stays organized and there's less chance of something going wrong and data getting lost.

### **4. XFS file system.**

The XFS file system is a high-performance file system based on the B+ tree data structure, which allows for fast disk access and efficient file lookups. One unique feature of the XFS file system is dynamic allocation of inodes, which are the data structures that store information about files. This can be especially useful in environments with a large number of small files.

### **5. BTRFS (B-tree file system)**

Btrfs (B-tree file system) is a modern, advanced file system for Linux that is designed to be scalable, efficient, and reliable. Btrfs uses a B-tree data structure to organize files and metadata, which allows for fast disk access and efficient file lookups. It includes advanced features such as snapshotting, checksumming, and RAID support, which make it suitable for use in enterprise-level storage systems.

One of the key features of Btrfs is its support for snapshots. A snapshot is a read-only copy of the file system at a specific point in time. Snapshots can be used for backups, system recovery, or to create point-in-time copies of data for testing or analysis.

### **6. swap file system**

In a computer system, the swap file system (or swap space) is an area of the hard drive that is used as a virtual memory extension when the RAM is full. When a computer runs out of physical memory, it uses the swap space to temporarily transfer data from the RAM to the hard drive, allowing the computer to continue functioning. The swap space



is essentially a way to supplement the physical memory with additional virtual memory.

- **Linux file system directories.**

Directory	Description
/	The "/" directory, also known as the root directory, is the base directory where all other directories are stored.
/bin	The "bin/" directory contains binary files of commonly used commands such as "man", "ls", etc.
dev/	The "dev/" directory contains device files, which are generated when the system is booted.
etc/	The "etc/" directory contains system-wide configuration files, including information about users and their passwords, as well as network settings.
lib/	The "lib/" directory contains all the libraries needed by applications to run, as libraries are files that contain code for the application to execute.
home/	The "home/" directory contains personal directories for each user.

Note:- If you execute "ls -lrt /" then you can see all the directories.

```
[root@DIWRND254 lib]# ls -lrt /
total 20
drwxr-xr-x.  2 root root    6 Apr 11  2018 srv
drwxr-xr-x.  2 root root    6 Apr 11  2018 mnt
drwxr-xr-x.  2 root root    6 Apr 11  2018 media
drwxr-xr-x.  2 root root    6 Apr 11  2018 home
lrwxrwxrwx.  1 root root    7 Dec 15 14:14 bin -> usr/bin
lrwxrwxrwx.  1 root root    9 Dec 15 14:14 lib64 -> usr/lib
lrwxrwxrwx.  1 root root    7 Dec 15 14:14 lib -> usr/lib
lrwxrwxrwx.  1 root root    8 Dec 15 14:14 sbin -> usr/sbin
drwxr-xr-x. 13 root root  155 Dec 15 14:14 usr
drwxr-xr-x. 19 root root  267 Dec 15 14:21 var
dr-xr-xr-x. 187 root root    0 Mar 10 15:44 proc
dr-xr-xr-x. 13 root root    0 Mar 10 15:44 sys
drwxr-xr-x. 20 root root 3180 Mar 10 15:44 dev
dr-xr-xr-x.  5 root root 4096 Mar 24 03:35 boot
drwxr-xr-x. 84 root root 8192 Mar 24 07:29 etc
drwxr-xr-x. 26 root root  700 Apr  4 12:54 run
drwxr-xr-x.  8 root root  132 Apr  6 19:25 opt
drwxrwxr-x.  4 root root   60 Apr 28 13:23 credence
dr-xr-x---. 17 root root 4096 May  4 12:10 root
drwxrwxrwt.  9 root root  195 May  4 17:11 tmp
```

## ● File Permissions in Linux: How to Read, Write & Change?

In Linux, file and folder permissions are used to control access to files and directories. There are three types of permissions:

Read (r)(4) - allows a user to view the contents of a file or directory.

Write (w)(2) - allows a user to modify the contents of a file or directory.

Execute (x)(1) - allows a user to run a file or access the contents of a directory.

These permissions are assigned to three groups of users:

Owner - the user who created the file or directory.

Group - a group of users who have been granted access to the file or directory.

Others - all other users who are not the owner or part of the group.

The first character indicates whether the item is a file (-) or directory (d), followed by three sets of permissions for the owner, group, and others.

Example:-

### 1. drwxr-xr-x

The first character 'd' indicates that it is a directory. The file permissions are divided into three parts: user, group, and other. The user has read, write, and execute permissions, the group has execute and read permissions, and others have only execute permissions.

Numerical number will be = user(4+2+1) + group(1+4) + other(1) = 7-3-1 = 731.

### 2. -rw-r--r--

The '-' indicates that it is a file. 'rw' indicates the user has read and write permissions. 'r' indicates the group has read permission. 'r' indicates others have read permission.

Numerical number will be = user(4+2) + group(4) + other(4) = 5-4-4 = 544.

### 3. -rwxr-xr-x

The '-' indicates that it is a file. 'rwx' indicates the user has read, write, and execute permissions. 'xr' indicates the group has execute and read permissions. 'x' indicates others have execute permission. 'r' after 'rwx' indicates everyone has read permission.

Numerical number will be = user(4+2+1) + group(1+4) + other(1+4) = 7-5-1 = 755.

### 4. -rw-rw-r--

The '-' indicates that it is a file. 'rw' indicates the user has read and write permissions. 'rw' indicates the group has read and write permissions. 'r' indicates others have read permission, but no write permission.

Numerical number will be = user(4+2) + group(4+2) + other(4) = 664.

### 5. -rwxr--r--

The '-' indicates that it is a file. 'rwx' indicates the user has read, write, and execute permissions. 'r' indicates the group has read permission, but no write or execute permission. 'r' indicates others have read permission, but no write or execute permission.

Numerical number will be = user(4+2+1) + group(4) + other(4) = 754.

### 6. -rw-r-----

The '-' indicates that it is a file. 'rw' indicates the user has read and write permissions. 'r' indicates the group has read

permission, but no write or execute permission. '-' indicates others have no permission.

Numerical number will be = user(4+2) + group(4) + other(0) = 640

#### 7. drwxr-xr-x

The first character 'd' indicates that it is a directory. The file permissions are divided into three parts: user, group, and other. The user has read, write, and execute permissions. The group has execute and read permissions. Others have execute permissions, but no write permission.

Numerical number will be = user(4+2+1) + group(1+4) + other(1) = 7-5-5 = 755.

#### 8. drwxrwxrwx

The first character 'd' indicates that it is a directory. The file permissions are divided into three parts: user, group, and other. All have read, write, and execute permissions.

Numerical number will be = user(4+2+1) + group(4+2+1) + other(4+2+1) = 7-7-7 = 777.

```
[root@DIWRND254 Linux_Tutorial_Test]# ls -lrt
total 8
drwxrwxrwx. 2 root root  6 Apr 17 16:51 Example_of_777
-rw-rw-r--. 1 root root 851 Apr 17 17:46 Example_of_664
-rw-r-----. 1 root root 25 Apr 27 11:18 Example_of_640
-rwxr-xr--. 1 root root  0 May  4 19:43 Example_of_754
drwxr-xr-x. 2 root root 21 May  4 19:45 Example_of_755
-r-xr--r--. 1 root root  0 May  4 19:45 Example_of_544
drwx-wx--x. 2 root root  6 May  4 19:46 Example_of_731
[root@DIWRND254 Linux_Tutorial_Test]#
```

Note:- To change the permission of files and directories by using command mentioned below.

chmod <number> <file or directory>

# Command Line Basics

(Introduction essential command line concepts, such as navigating directories, creating, copying, moving, and deleting files and directories, and using basic command line utilities like ls, cd, cp, mv, rm, mkdir, etc.)

## ● Basics Linux Commands with Examples & Syntax

Understanding the file system in Linux is crucial for working with the operating system efficiently. The correct use of basic commands can make working with Linux much easier. In this lesson, we will discuss some basic commands for working with files and directories in Linux.

### a> Listing the files and directories:

To view a list of files and directories in your current directory, use the 'ls' command. This command will display all files and directories in the current directory, with folders highlighted in blue and files in white.

Command: ls

```
[root@DIWRND254 ~]# ls
10.250.94.196.txt  CloudDeployment  host.xml  jboss-eap-7.2.0.zip  tmp.txt  ucd-agent.zip  Velocity_Server
Agent             GIT             Java     jboss_setup.zip     UCD       Urbancode      WAR
anaconda-ks.cfg  host-salve.xml  JBOSSE   Linux_Tutorial_Test  ucd-agent-install  Velocity       YBL
```

There are various options you can use with the ls command to customize the output to your needs:

# ls -r: This option will sort the output in reverse order.

# ls -t: This option will sort the output by date and time.

# ls -h: This option will print all values (e.g. size of files and directories) in human-readable format, making it easier to understand.

# ls -a: This option will print all hidden files and directories, including those with names that begin with a dot (.) character.

# ls -l --block-size=<KB/MB/GB/TB>: This option will allow you to specify the size in which you want the file size to be displayed, making it easier to understand the size of files and directories.



```
[root@DIWRND254 jdk1.8.0_301]# ls -lrtha
total 26M
-rw-r--r--. 1 10143 10143 190 Jun 9 2021 THIRDPARTYLICENSEREADME-JAVAFX.txt
-rw-r--r--. 1 10143 10143 5.0M Jun 9 2021 javafx-src.zip
-r--r--r--. 1 10143 10143 190 Jun 9 2021 THIRDPARTYLICENSEREADME.txt
-rw-r--r--. 1 10143 10143 21M Jun 9 2021 src.zip
-rw-r--r--. 1 10143 10143 486 Jun 9 2021 release
-r--r--r--. 1 10143 10143 159 Jun 9 2021 README.html
drwxr-xr-x. 4 10143 10143 47 Jun 9 2021 man
-r--r--r--. 1 10143 10143 44 Jun 9 2021 LICENSE
drwxr-xr-x. 3 10143 10143 132 Jun 9 2021 include
-r--r--r--. 1 10143 10143 3.2K Jun 9 2021 COPYRIGHT
drwxr-xr-x. 4 10143 10143 223 Jun 9 2021 lib
drwxr-xr-x. 4 10143 10143 31 Jun 9 2021 legal
drwxr-xr-x. 6 10143 10143 198 Jun 9 2021 jre
drwxr-xr-x. 2 10143 10143 4.0K Jun 9 2021 bin
-rw-r--r--. 1 10143 10143 195 Jun 9 2021 jmc.txt
drwxr-xr-x. 8 10143 10143 273 Jun 9 2021 .
drwxr-xr-x. 3 root root 45 Dec 15 16:26 ..
[root@DIWRND254 jdk1.8.0_301]#
```

### b> Creating Files.

Creating files is a basic operation in Linux. To create an empty file, you can use the "touch" command. For example, if you want to create a file named "Linux.txt," you can execute the following command:

```
# touch Linux.txt
```

This command will create an empty file with the specified name and extension. The "touch" command can be used to create any type of file with any extension.

```
[root@DIWRND254 ~]# cd Linux_Tutorial_Test/
[root@DIWRND254 Linux_Tutorial_Test]# ls -lrt
total 8
drwxrwxrwx. 2 root root 6 Apr 17 16:51 Example_of_777
-rw-rw-r--. 1 root root 851 Apr 17 17:46 Example_of_664
-rw-r-----. 1 root root 25 Apr 27 11:18 Example_of_640
-rwxr-xr--. 1 root root 0 May 4 19:43 Example_of_754
drwxr-xr-x. 2 root root 21 May 4 19:45 Example_of_755
-r-xr--r--. 1 root root 0 May 4 19:45 Example_of_544
drwx-wx--x. 2 root root 6 May 4 19:46 Example_of_731
[root@DIWRND254 Linux_Tutorial_Test]# touch Linux.txt
[root@DIWRND254 Linux_Tutorial_Test]# ls -lrt
total 8
drwxrwxrwx. 2 root root 6 Apr 17 16:51 Example_of_777
-rw-rw-r--. 1 root root 851 Apr 17 17:46 Example_of_664
-rw-r-----. 1 root root 25 Apr 27 11:18 Example_of_640
-rwxr-xr--. 1 root root 0 May 4 19:43 Example_of_754
drwxr-xr-x. 2 root root 21 May 4 19:45 Example_of_755
-r-xr--r--. 1 root root 0 May 4 19:45 Example_of_544
drwx-wx--x. 2 root root 6 May 4 19:46 Example_of_731
-rw-r--r--. 1 root root 0 May 5 12:37 Linux.txt
[root@DIWRND254 Linux_Tutorial_Test]#
```

### c> Changing directories.

In Linux, you can use the "cd" command to change directories. To change to a specific directory, you can use the following command:

# cd <Directory name>

For example, if you are in the home directory that contains directories named "Example-1" and "Example-2," and you want to change to "Example-1," you can execute the following command:

# cd Example-1

To move up one level in the directory hierarchy, you can use the following command:

# cd ..

```
[root@DIWRND254 Linux_Tutorial_Test]# ls -lrt | tail -n +7
-r-xr--r--. 1 root root  0 May  4 19:45 Example_of_544
drwx-wx--x. 2 root root  6 May  4 19:46 Example_of_731
-rw-r--r--. 1 root root  0 May  5 12:37 Linux.txt
[root@DIWRND254 Linux_Tutorial_Test]# cd Example_of_731
[root@DIWRND254 Example_of_731]# pwd
/root/Linux_Tutorial_Test/Example_of_731
[root@DIWRND254 Example_of_731]#
```

d> path to the folder we are currently working in(pwd)

In Linux, the "pwd" command is used to display the current working directory. For example, if you are in the directory "/root/Example-1/Linux-lesson-1," executing the following command will display the current directory:

# pwd

This command will print the full path of your current working directory.

```
[root@DIWRND254 Example_of_731]# pwd
/root/Linux_Tutorial_Test/Example_of_731
[root@DIWRND254 Example_of_731]#
```

e> Create Directory

In Linux, you can create a new directory using the "mkdir" command. For example, if you want to create a directory named "LessonFiles," you can use the following command:

mkdir LessonFiles

This command will create a directory with the name "LessonFiles" in your current working directory.

You can also use the "mkdir" command to create multiple directories at once. For example, the following command will create three directories named "Directory-1," "Directory-2," and "Directory-3":

# mkdir Directory-1 Directory-2 Directory-3

If you want to create a directory with a specific path, you can use the "-p" option followed by the path and directory name. For example, the following command will create a directory named "Directory-1" inside the "/root/Example-1/" directory:

# mkdir -p /root/Example-1/Directory-1

```
[root@DIWRND254 Example_of_731]# ls -lrt
total 0
[root@DIWRND254 Example_of_731]# mkdir Linux_Folder
[root@DIWRND254 Example_of_731]# ls -lrt
total 0
drwxr-xr-x. 2 root root 6 May  5 12:43 Linux_Folder
[root@DIWRND254 Example_of_731]#
```

#### f> delete Files and Directories.

In Linux, you can use the "rm" command to delete files and the "rmdir" command to delete empty directories.

To delete a file, use the following syntax:

```
# rm <file_name>
```

For example, to delete a file called "example.txt", you can use the following command:

```
# rm example.txt
```

Be careful when using the "rm" command as there is no confirmation prompt before deletion.

To delete a directory, you can use the "rmdir" command with the following syntax:

```
# rmdir <directory_name>
```

Note that this command only works for empty directories. To delete a non-empty directory and its contents, use the "rm" command with the "-r" flag, which stands for "recursive".

```
# rm -r <directory_name>
```

This will delete the directory and all its contents, including any sub-directories and files within it. As with the "rm" command, be careful when using the "-r" flag, as there is no confirmation prompt before deletion.

```
[root@DIWRND254 Example_of_731]# ls
Linux_Folder
[root@DIWRND254 Example_of_731]# rm -rf Linux_Folder/
[root@DIWRND254 Example_of_731]# ls
[root@DIWRND254 Example_of_731]#
```

```
[root@DIWRND254 Linux_Tutorial_Test]# ls -lrt | tail -n +6
drwxr-xr-x. 2 root root 21 May  4 19:45 Example_of_755
-r-xr--r--. 1 root root  0 May  4 19:45 Example_of_544
-rw-r--r--. 1 root root  0 May  5 12:37 Linux.txt
drwx-wx--x. 2 root root  6 May  5 12:44 Example_of_731
[root@DIWRND254 Linux_Tutorial_Test]# rm -rf Linux.txt
[root@DIWRND254 Linux_Tutorial_Test]# ls -lrt | tail -n +6
drwxr-xr-x. 2 root root 21 May  4 19:45 Example_of_755
-r-xr--r--. 1 root root  0 May  4 19:45 Example_of_544
drwx-wx--x. 2 root root  6 May  5 12:44 Example_of_731
[root@DIWRND254 Linux_Tutorial_Test]#
```



### g> Printing Messages.

To display messages on the terminal, you can use the "echo" command. This command is often used for simple programs like the famous "Hello, World!" program. For instance, you can print the message "Hello, World!" by running the following command:

```
# echo "Hello, World!"
```

You can substitute "Hello, World!" with any message you want to display on the terminal.

```
[root@DIWRND254 Linux_Tutorial_Test]# echo "Hello World"
Hello World
[root@DIWRND254 Linux_Tutorial_Test]#
```

### h> move files and folders

The "mv" command in Linux is used to move files and directories from one location to another. The basic syntax of the command is:

```
mv [options] source destination
```

Here, "source" refers to the file or directory that you want to move, and "destination" refers to the new location where you want to move the file or directory.

For example, to move a file named "Example.txt" to the "/root/opt/Sample" directory, you would use the following command:

```
# mv Example.txt /root/opt/Sample/
```

Note that you need to specify the full path of the destination directory.

Similarly, to move a directory named "Directory1" to the "/root/opt/Sample" directory, you would use the following command:

```
# mv Directory1 /root/opt/Sample/
```

Again, make sure to specify the full path of the destination directory.

```
[root@DIWRND254 Example_of_777]# ls -lrt
total 0
drwxr-xr-x. 2 root root 6 May  5 12:48 Dest
-rw-r--r--. 1 root root 0 May  5 12:49 Example.txt
drwxr-xr-x. 2 root root 6 May  5 12:49 Source
[root@DIWRND254 Example_of_777]# mv Example.txt Source/
[root@DIWRND254 Example_of_777]# ls -lrt
total 0
drwxr-xr-x. 2 root root  6 May  5 12:48 Dest
drwxr-xr-x. 2 root root 25 May  5 12:49 Source
[root@DIWRND254 Example_of_777]# ls -lrt Source/
total 0
-rw-r--r--. 1 root root 0 May  5 12:49 Example.txt
[root@DIWRND254 Example_of_777]#
```

```
[root@DIWRND254 Example_of_777]# ll
total 0
drwxr-xr-x. 2 root root 6 May 5 12:48 Dest
drwxr-xr-x. 2 root root 25 May 5 12:49 Source
[root@DIWRND254 Example_of_777]# mv Source/ Dest/
[root@DIWRND254 Example_of_777]# ls -lrt Dest/
total 0
drwxr-xr-x. 2 root root 25 May 5 12:49 Source
[root@DIWRND254 Example_of_777]#
```

### i> copy files and folders

To copy files or directories from one location to another, you can use the 'cp' command. The general format of the command is:

```
cp [OPTION]... SOURCE DEST
```

Here, SOURCE specifies the file or directory you want to copy, and DEST specifies the destination directory where you want to copy the file or directory.

For example, to copy a file named Application.war to the directory /opt/tomcatHome/webapps/, you can use the following command:

```
# cp Application.war /opt/tomcatHome/webapps/
```

To copy a directory named War along with its contents to the directory /opt/tomcatHome/, you can use the following command:

```
# cp -r War/ /opt/tomcatHome/
```

The -r option specifies that the directory should be copied recursively along with its contents.

```
[root@DIWRND254 Example_of_777]# ls -lrt
total 0
-rw-r--r--. 1 root root 0 May 5 12:49 Example.txt
drwxr-xr-x. 2 root root 6 May 5 12:54 Dest
drwxr-xr-x. 2 root root 6 May 5 12:54 Source
[root@DIWRND254 Example_of_777]# cp Example.txt Source/
[root@DIWRND254 Example_of_777]# ls -lrt Source/
total 0
-rw-r--r--. 1 root root 0 May 5 12:55 Example.txt
[root@DIWRND254 Example_of_777]# cp -R Source/ Dest/
[root@DIWRND254 Example_of_777]# ls -lrt Dest/
total 0
drwxr-xr-x. 2 root root 25 May 5 12:55 Source
[root@DIWRND254 Example_of_777]# ls -lrt Dest/Source/
total 0
-rw-r--r--. 1 root root 0 May 5 12:55 Example.txt
[root@DIWRND254 Example_of_777]#
```

# Users and Permissions

(Cover user and group management, including creating and managing user accounts.)

## ● Creating and Managing User Accounts

1. Creating user accounts is a fundamental task in user and group management. To create a user account in a Unix-based operating system, you can use the `useradd` command followed by the username.

```
sudo useradd <username>
```

2. This command will create a user account with the specified username. By default, this account will not have a password, and its home directory will be created in the `/home` directory.

To set a password for the user account, you can use the `passwd` command:

```
sudo passwd <username>
```

3. To delete a user account, you can use the `userdel` command:

```
sudo userdel <username>
```

```
[root@DIWRND254 ~]# su - LinuxTest
su: user LinuxTest does not exist
[root@DIWRND254 ~]# sudo useradd LinuxTest
[root@DIWRND254 ~]# sudo passwd LinuxTest
Changing password for user LinuxTest.
New password:
BAD PASSWORD: The password fails the dictionary check - it is based on a dictionary word
Retype new password:
passwd: all authentication tokens updated successfully.
[root@DIWRND254 ~]# sudo su - LinuxTest
[LinuxTest@DIWRND254 ~]$ whoami
LinuxTest
[LinuxTest@DIWRND254 ~]$ █

[LinuxTest@DIWRND254 ~]# sudo userdel -f LinuxTest
userdel: user LinuxTest is currently used by process 17313
[root@DIWRND254 ~]# █
```

<There is much more topics related user and permission you can explore them your-self>

# Package Management

(Package management systems, such as apt or yum, for installing, updating, and managing software packages on Linux systems.)

Package management systems are essential for installing, updating, and managing software packages on Linux systems. These systems allow you to easily install and manage software packages, dependencies, and updates, ensuring that your system is up to date and running smoothly.

Two popular package management systems used on Linux systems are apt and yum.

## 1. apt

apt (Advanced Packaging Tool) is a package management system used primarily on Debian-based systems such as Ubuntu, Debian, and Linux Mint. It allows you to install and manage packages from a central repository or from locally downloaded package files.

Commands:

apt-get update: Refreshes the list of available packages from the repositories.

apt-get upgrade: Upgrades all packages to their latest versions.

apt-get install packagename: Installs the specified package and its dependencies.

apt-get remove packagename: Removes the specified package but leaves any configuration files intact.

apt-get purge packagename: Removes the specified package and its configuration files.

## 2. yum

yum (Yellowdog Updater Modified) is a package management system used primarily on Red Hat-based systems such as CentOS, Fedora, and Red Hat Enterprise Linux. It allows you to install and manage packages from a central repository or from locally downloaded package files.

Commands:

yum update: Upgrades all packages to their latest versions.

yum install packagename: Installs the specified package and its dependencies.

yum remove packagename: Removes the specified package but leaves any configuration files intact.

`yum erase packagename:` Removes the specified package and its configuration files.

Both `apt` and `yum` have many other options and commands that allow you to manage packages and their dependencies in a variety of ways. It's important to regularly update your system's packages to ensure that any security vulnerabilities or bugs are addressed, and to use caution when installing and removing packages to avoid breaking system dependencies or causing other issues.

```
[root@DIWRND254 ~]# sudo userdel -f LinuxTest
userdel: user LinuxTest is currently used by process 17313
[root@DIWRND254 ~]# clear
[root@DIWRND254 ~]# sudo yum update
Loaded plugins: fastestmirror
Determining fastest mirrors
 * base: centos.excellmedia.net
 * extras: centos.excellmedia.net
 * updates: centos.excellmedia.net
base                                     | 3.6 kB  00:00:00
docker-ce-stable                       | 3.5 kB  00:00:00
extras                                 | 2.9 kB  00:00:00
updates                               | 2.9 kB  00:00:00
zabbix                                 | 2.9 kB  00:00:00
zabbix-non-supported                  | 2.9 kB  00:00:00
(1/2): docker-ce-stable/7/x86_64/primary_db | 106 kB  00:00:00
(2/2): updates/7/x86_64/primary_db      | 21 MB  00:00:03
Resolving Dependencies
--> Running transaction check
---> Package containerd.io.x86_64 0:1.6.18-3.1.el7 will be updated
---> Package containerd.io.x86_64 0:1.6.21-3.1.el7 will be an update
---> Package docker-buildx-plugin.x86_64 0:0.10.2-1.el7 will be updated
---> Package docker-buildx-plugin.x86_64 0:0.10.4-1.el7 will be an update
```

# Text Editors

(Familiarize with popular text editors, such as vi or nano, for editing files from the command line.)

- VI

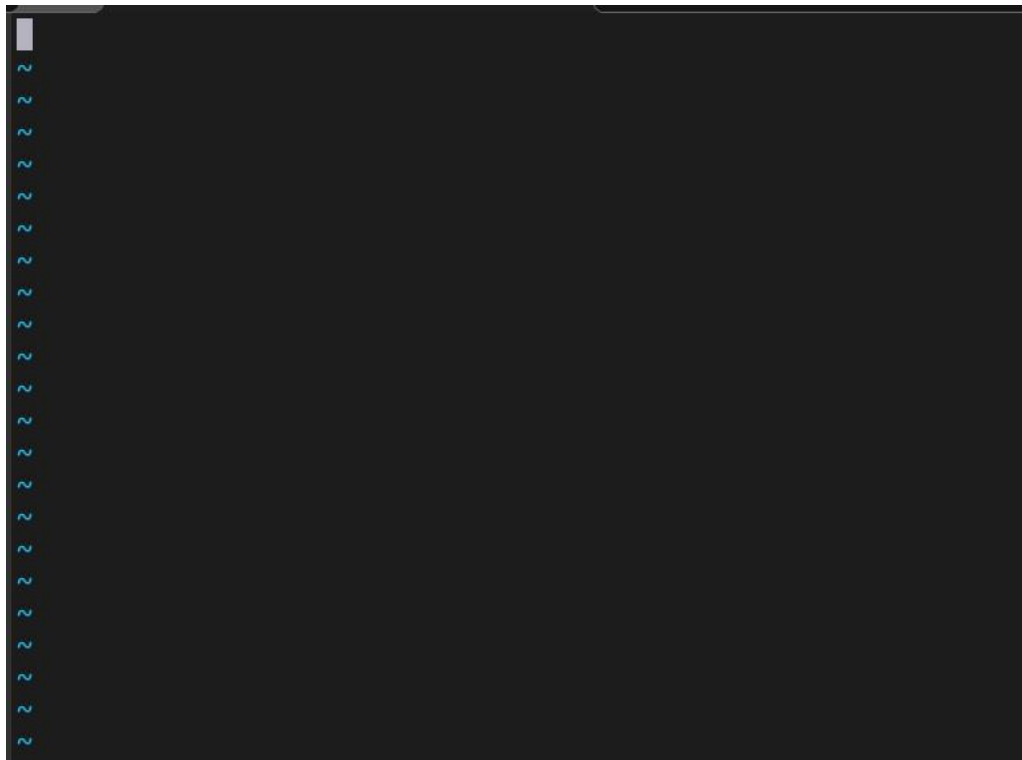
It is a powerful and highly customizable editor that can be used to create and edit text files of any size.

Example:-

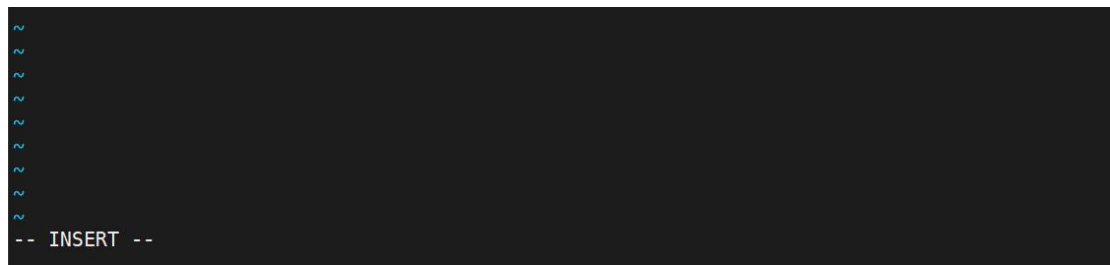
To create a new file in Vi, simply enter the command "vi filename" in the terminal. For instance, to create a new file named Linux.txt,

```
[root@DIWRND254 Linux_Tutorial_Test]# vi Linux.txt
```

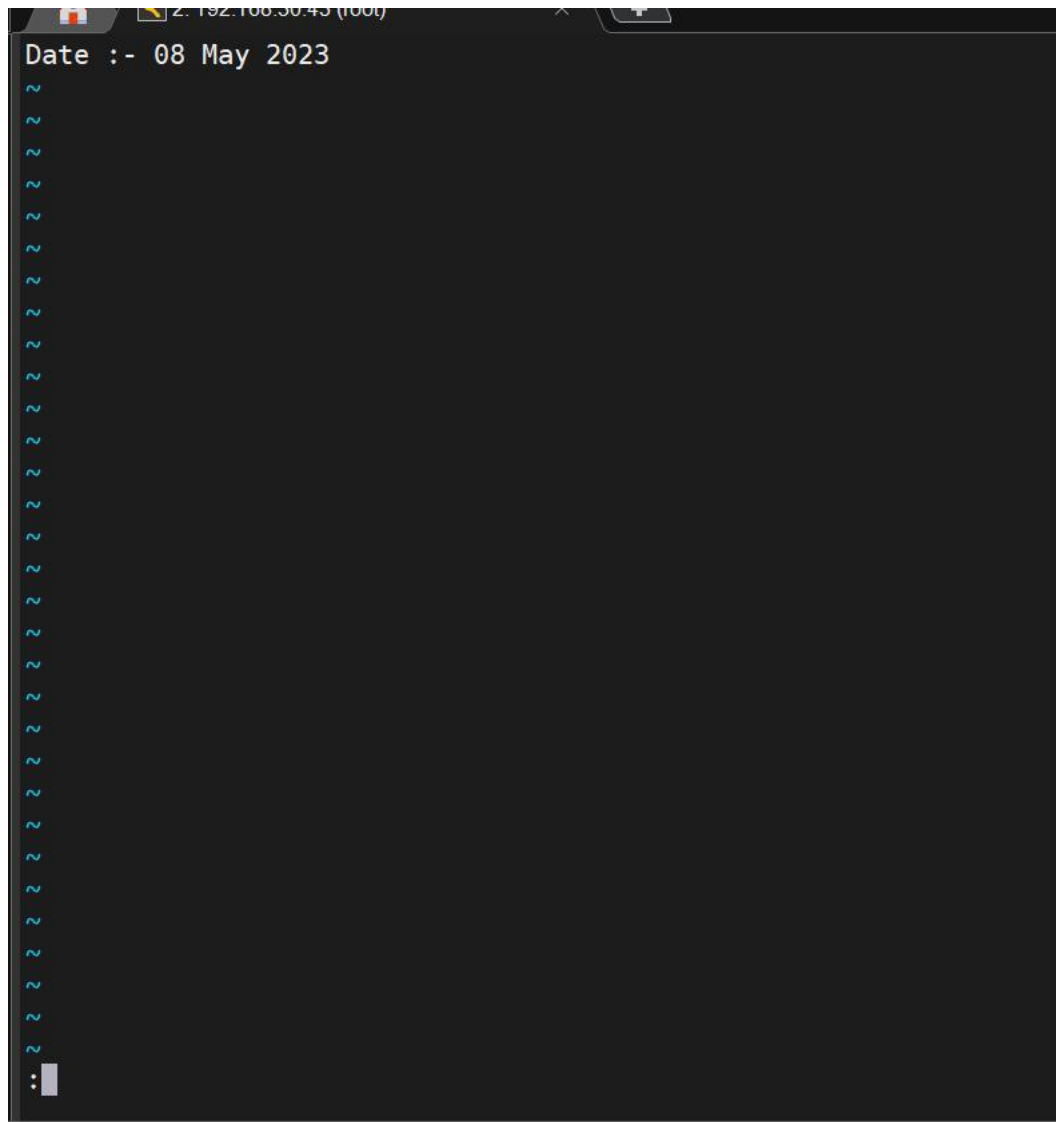
After executing command you will get output similar to below mentoined image.



Once the file opens in Vi, enter the insert mode by pressing Enter followed by the "I" key. This will allow you to enter the text you want to write. After you have typed in your content, exit insert mode by pressing the ESC key.



You will get output similar to above image.



To save the file, press Shift + colon to enter the command mode, and type "wq" followed by the Enter key. This will save the changes you made to the file and exit Vi.

```
[root@DIWRND254 Linux_Tutorial_Test]# vi Linux.txt
[root@DIWRND254 Linux_Tutorial_Test]# cat Linux.txt
Date :- 08 May 2023
[root@DIWRND254 Linux_Tutorial_Test]#
```

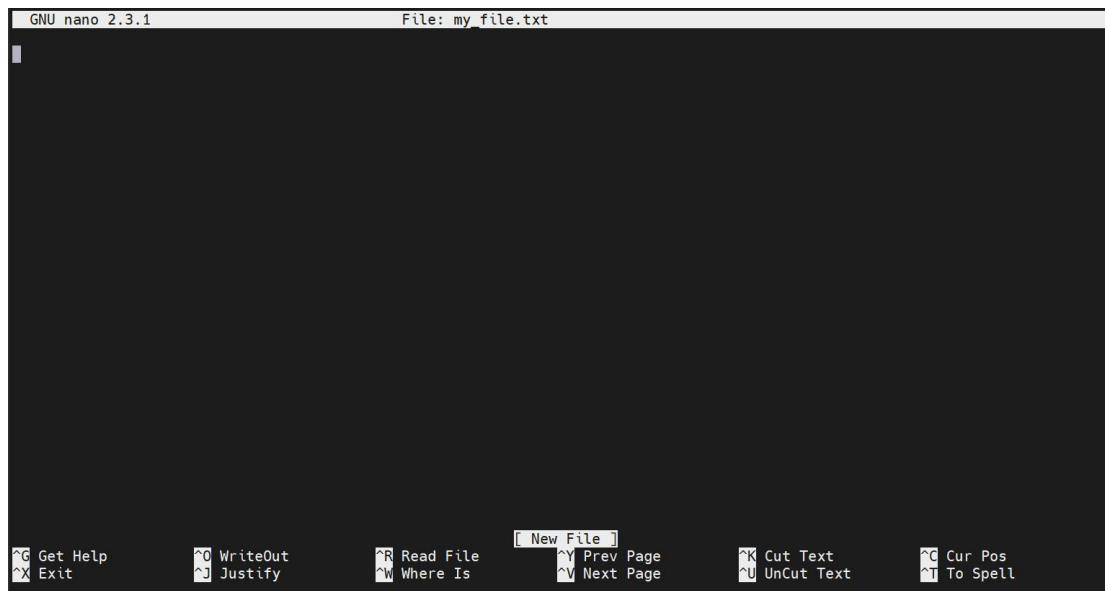
In summary, Vi is a powerful text editor that provides a wide range of features to create and modify text files efficiently. Learning Vi commands can be highly beneficial for developers and system administrators who work with text files regularly.

## ● NANO

Nano is a command-line text editor that comes pre-installed on many Unix-based systems. In case you don't have it in your system, you can install it with your system's package manager. It is a simple and user-friendly text editor that is well-suited for editing configuration files, scripts, and other plain text files.

Example:-

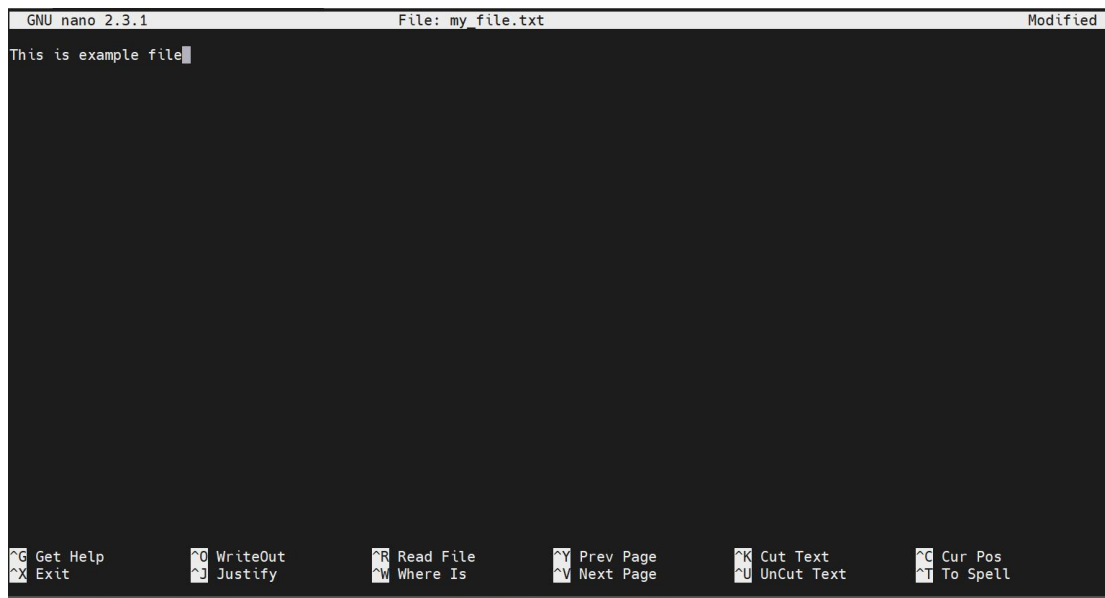
Opening a file: To open a file in Nano, simply type nano followed by the filename. For example, to open a file called my\_file.txt, type:  
nano my\_file.txt



Navigation: You can move the cursor around the file using the arrow keys or the Page Up and Page Down keys.

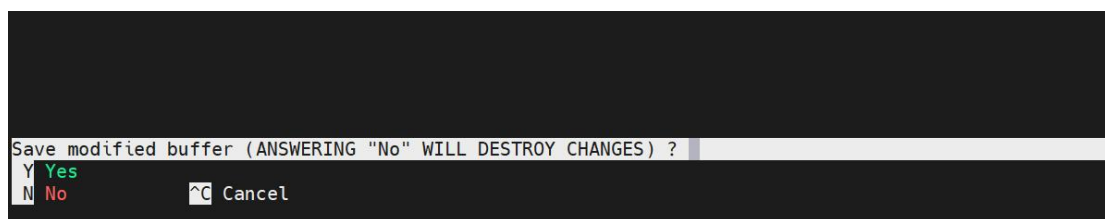


Editing text: You can insert, delete, or modify text just like in any other text editor. To insert text, simply start typing at the cursor. To delete text, use the Backspace or Delete key.



```
GNU nano 2.3.1      File: my_file.txt      Modified
This is example file
^G Get Help  ^O WriteOut  ^R Read File  ^Y Prev Page  ^K Cut Text  ^C Cur Pos
^X Exit      ^J Justify   ^W Where Is   ^V Next Page  ^U UnCut Text ^T To Spell
```

Saving and exiting: To save your changes and exit Nano, press Ctrl+X. You will be prompted to save the file if you have made any changes. Press Y to save the changes or N to discard them. To save the file without exiting, press Ctrl+O.



```
Save modified buffer (ANSWERING "No" WILL DESTROY CHANGES) ?
Y Yes
N No      ^O Cancel
```

Search and replace: To search for a specific word or phrase, press Ctrl+W and enter the search term. To replace a word or phrase, press Ctrl+\ and enter the search and replacement terms.

```
GNU nano 2.3.1      File: my_file.txt
This is example file

Search [example]: example
^G Get Help      ^Y First Line    ^T Go To Line    ^W Beg of Par    M-J FullJstify    M-B Backwards
^C Cancel        ^V Last Line     ^R Replace       ^O End of Par    M-C Case Sens     M-R Regexp
```

Press Enter

```
GNU nano 2.3.1      File: my_file.txt      Modif
This is example file

Search (to replace) [example]: example
^G Get Help      ^Y First Line    M-C Case Sens    M-R Regexp      ^N NextHistory
^C Cancel        ^V Last Line     M-B Backwards    ^P PrevHistory   ^R No Replace
```

Hit Enter and Y to replace word.

```
GNU nano 2.3.1      File: my_file.txt
This is example file

Replace with:
^G Get Help          ^Y First Line       ^P PrevHistory
^C Cancel            ^V Last Line        ^N NextHistory
```

Output:-

```
GNU nano 2.3.1      File: my_file.txt
This is my_filefile

Replaced 1 occurrence
^G Get Help          ^O WriteOut         ^R Read File        ^Y Prev Page        ^K Cut Text         ^C Cur Pos
^X Exit              ^J Justify          ^W Where Is         ^V Next Page        ^U UnCut Text       ^T To Spell
```

These are just a few basic commands and features of Nano. For more advanced usage, you can refer to the Nano manual by typing `man nano` in the terminal.

# Networking

(Essential networking concepts, including configuring network interfaces, managing network services, and basic troubleshooting using commands like `ifconfig`, `ping`, and `netstat`.)

`ping`: This command is used to test network connectivity by sending an ICMP echo request to a specified host or IP address. For example, to ping Google's DNS server at 8.8.8.8, type: `ping 8.8.8.8`

```
Linux_Tutorial_Test]# ping 8.8.8.8
PING to Google's DNS server
[root@DIWRND254 Linux_Tutorial_Test]# ping 8.8.8.8
PING 8.8.8.8 (8.8.8.8) 56(84) bytes of data:
64 bytes from 8.8.8.8: icmp_seq=1 ttl=117 time=4.11 ms
64 bytes from 8.8.8.8: icmp_seq=2 ttl=117 time=4.19 ms
64 bytes from 8.8.8.8: icmp_seq=3 ttl=117 time=4.13 ms
64 bytes from 8.8.8.8: icmp_seq=4 ttl=117 time=7.06 ms
64 bytes from 8.8.8.8: icmp_seq=5 ttl=117 time=4.20 ms
64 bytes from 8.8.8.8: icmp_seq=6 ttl=117 time=4.11 ms
64 bytes from 8.8.8.8: icmp_seq=7 ttl=117 time=4.10 ms
64 bytes from 8.8.8.8: icmp_seq=8 ttl=117 time=7.16 ms
64 bytes from 8.8.8.8: icmp_seq=9 ttl=117 time=4.10 ms
64 bytes from 8.8.8.8: icmp_seq=10 ttl=117 time=7.31 ms
64 bytes from 8.8.8.8: icmp_seq=11 ttl=117 time=4.10 ms
64 bytes from 8.8.8.8: icmp_seq=12 ttl=117 time=7.13 ms
64 bytes from 8.8.8.8: icmp_seq=13 ttl=117 time=4.20 ms
^C
--- 8.8.8.8 ping statistics ---
13 packets transmitted, 13 received, 0% packet loss, time 12015ms
rtt min/avg/max/mdev = 4.100/5.073/7.318/1.400 ms
[root@DIWRND254 Linux_Tutorial_Test]#
```

Note:- Instead of google DNS server you can use your target system's IP.

`nslookup`: This command is used to look up DNS information, including IP addresses and hostnames. For example, to look up the IP address of `www.example.com`, type: `nslookup <hostname>`

```
nslookup example
[root@DIWRND254 Linux_Tutorial_Test]# nslookup DIWRND140
Server:         192.168.30.2
Address:        192.168.30.2#53

** server can't find DIWRND140: NXDOMAIN

[root@DIWRND254 Linux_Tutorial_Test]#
```

telnet: This command is used to connect to a remote server over a network using the Telnet protocol. For example, to connect to a web server on port 80, type: telnet <hostname/ip> <portnumber>

```
Example of Telnet
[root@DIWRND254 Linux_Tutorial_Test]# telnet google.com 80
Trying 142.250.66.14...
Connected to google.com.
Escape character is '^['.
^C^C
Connection closed by foreign host.
[root@DIWRND254 Linux_Tutorial_Test]# telnet 192.168.30.18 7919
Trying 192.168.30.18...
telnet: connect to address 192.168.30.18: Connection refused
[root@DIWRND254 Linux_Tutorial_Test]#
```

curl: This command is used to transfer data over a network, typically used to download files or web pages from a server. For example, to download the home page of www.example.com, type: curl <URL>

```
Example of curl
[root@DIWRND254 Linux_Tutorial_Test]# curl https://www.datamato.com/
<!DOCTYPE html><!-- Last Published: Thu May 04 2023 07:18:45 GMT+0000 (Coordinated Universal Time) --><html data-wf-domain="www.datamato.com" data-wf-page="6440cc745081909f8679c9c7" data-wf-site="613ef01dd83966c5530ca721"><head><meta charset="utf-8"/><title>Datamato - DevOps, IoT, Security, Cloud - Software Consultant & Services</title><meta content="Datamato - DevOps, IoT, Security, Cloud - Software Consultant & Services" property="og:title"/><meta content="Datamato - DevOps, IoT, Security, Cloud - Software Consultant & Services" property="twitter:title"/><meta content="width=device-width, initial-scale=1" name="viewport"/><link href="https://uploads-ssl.webflow.com/613ef01dd83966c5530ca721/css/datamato.webflow.1f353e9c2.css" rel="stylesheet" type="text/css"/><link href="https://fonts.googleapis.com" rel="preconnect"/><link href="https://fonts.gstatic.com" rel="preconnect" crossorigin="anonymous"/><script src="https://ajax.googleapis.com/ajax/libs/webfont/1.6.26/webfont.js" type="text/javascript"></script><script type="text/javascript">WebFont.load({ google: { families: ["Montserrat:100,100italic,200,200italic,300,300italic,400,400italic,500,500italic,600,600italic,700,700italic,800,800italic,900,900italic", "Merriweather:300,300italic,400,400italic,700,700italic,900,900italic", "Poppins:regular,600"] } });</script><!--[if lt IE 9]><script src="https://cdnjs.cloudflare.com/ajax/libs/html5shiv/3.7.3/html5shiv.min.js" type="text/javascript"></script><![endif]><script type="text/javascript">!function(o,c){var n=c.documentElement,t=" w-mod-";n.className+=t+"js",("ontouchstart"in o||o.DocumentTouch&&instanceof DocumentTouch)&&(n.className+=t+"touch");(window,document);</script><link href="https://uploads-ssl.webflow.com/613ef01dd83966c5530ca721/615759f1513aa5a752f493_Datamato-new-logo-32.png" rel="shortcut icon" type="image/x-icon"/><link href="https://uploads-ssl.webflow.com/613ef01dd83966c5530ca721/6261575f3826036e3ffedeb8_Datamato-new-logo.png" rel="apple-touch-icon"/><link rel="stylesheet" type="text/css" href="https://cdn.jsdelivr.net/npm/slick-carousel@1.8.1/slick/slick.css"/><!--<link href="https://apps.uniteddesigns.co/datamato.css" rel="stylesheet" type="text/css"/>-->

<style>
/*
.header a:not(.dropdown-link){
  color:currentColor;
}*/
.btn-icon{
  transition:ease 0.25s;
}

```

This will send a GET request to <https://www.google.com> and display the response body in the terminal. If the request is successful, you should see the HTML code of the Google homepage printed in the terminal.

traceroute: This command is used to trace the path that network packets take from your computer to a specified host or IP address. For example, to trace the route to Google's DNS server at 8.8.8.8, type: traceroute 8.8.8.8



```

Example of traceroute
[root@DIWRND254 Linux_Tutorial_Test]# traceroute 8.8.8.8
traceroute to 8.8.8.8 (8.8.8.8), 30 hops max, 60 byte packets
 1  secure.dtpldns.com (192.168.30.1)  0.201 ms  0.157 ms  0.148 ms
 2  103.224.245.1 (103.224.245.1)  2.312 ms  2.277 ms  2.230 ms
 3  115.247.156.145 (115.247.156.145)  4.022 ms  3.911 ms  4.085 ms
 4  142.250.165.42 (142.250.165.42)  5.094 ms  4.853 ms  4.810 ms
 5  * * *
 6  dns.google (8.8.8.8)  3.997 ms  4.434 ms  3.934 ms
[root@DIWRND254 Linux_Tutorial_Test]#

```

traceroute 8.8.8.8 will show you the path that network packets take from your computer to Google's DNS server, by displaying the IP addresses of the routers that the packets pass through on their way to the destination.

```

[root@DIWRND254 Linux_Tutorial_Test]# traceroute 8.8.8.8
traceroute to 8.8.8.8 (8.8.8.8), 30 hops max, 60 byte packets
 1  secure.dtpldns.com (192.168.30.1)  0.165 ms  0.218 ms  0.166 ms
 2  103.224.245.1 (103.224.245.1)  1.264 ms  0.987 ms  1.298 ms
 3  115.247.156.145 (115.247.156.145)  3.758 ms  3.596 ms  3.975 ms
 4  142.250.165.42 (142.250.165.42)  4.622 ms  4.590 ms  4.407 ms
 5  * * *
 6  dns.google (8.8.8.8)  4.007 ms  3.907 ms  3.905 ms
[root@DIWRND254 Linux_Tutorial_Test]#

```

Traceroute to Datamato.com

```

[root@DIWRND254 Linux_Tutorial_Test]# traceroute pdns03.domaincontrol.com
traceroute to pdns03.domaincontrol.com (97.74.111.51), 30 hops max, 60 byte packets
 1  secure.dtpldns.com (192.168.30.1)  0.201 ms  0.147 ms  0.166 ms
 2  103.224.245.1 (103.224.245.1)  1.901 ms  2.271 ms  3.422 ms
 3  203.199.138.193 (203.199.138.193)  5.378 ms  5.341 ms  5.392 ms
 4  * * *
 5  14.142.18.70.static-mumbai.vsnl.net.in (14.142.18.70)  7.316 ms  4.411 ms  4.384 ms
 6  243.154.109.208.host.secureserver.net (208.109.154.243)  10.323 ms  5.481 ms  5.236 ms
 7  pdns03.domaincontrol.com (97.74.111.51)  7.404 ms  9.447 ms  4.461 ms
[root@DIWRND254 Linux_Tutorial_Test]#

```

ifconfig: This command is used to display information about the network interfaces on your system, including IP addresses, netmasks, and MAC addresses. For example, to display information about the eth0 interface, type: ifconfig

```
[root@DIWRND254 Linux_Tutorial_Test]# echo "Example of ifconfig"
Example of ifconfig
[root@DIWRND254 Linux_Tutorial_Test]# ifconfig
ens192: flags=4163<UP,BROADCAST,RUNNING,MULTICAST> mtu 1500
    inet 192.168.30.43 netmask 255.255.254.0 broadcast 192.168.31.255
    inet6 fe80::9156:e247:dd46:52a7 prefixlen 64 scopeid 0x20<link>
    ether 00:50:56:8a:85:cb txqueuelen 1000 (Ethernet)
    RX packets 3564603 bytes 2006514424 (1.8 GiB)
    RX errors 0 dropped 5094 overruns 0 frame 0
    TX packets 1477075 bytes 2595488244 (2.4 GiB)
    TX errors 0 dropped 0 overruns 0 carrier 0 collisions 0

lo: flags=73<UP,LOOPBACK,RUNNING> mtu 65536
    inet 127.0.0.1 netmask 255.0.0.0
    inet6 ::1 prefixlen 128 scopeid 0x10<host>
    loop txqueuelen 1000 (Local Loopback)
    RX packets 139057443 bytes 59687142581 (55.5 GiB)
    RX errors 0 dropped 0 overruns 0 frame 0
    TX packets 139057443 bytes 59687142581 (55.5 GiB)
    TX errors 0 dropped 0 overruns 0 carrier 0 collisions 0

[root@DIWRND254 Linux_Tutorial_Test]#
```

netstat: This command is used to display information about active network connections, routing tables, and other networking statistics. For example, to display a list of active TCP connections, type: netstat -tn

```
Example of netstat
[root@DIWRND254 Linux_Tutorial_Test]# netstat -tn
Active Internet connections (w/o servers)
Proto Recv-Q Send-Q Local Address           Foreign Address         State
tcp        0      0 192.168.30.43:22       10.0.8.5:57410         ESTABLISHED
tcp        0      0 192.168.30.43:22       10.0.8.5:57409         ESTABLISHED
tcp6       0      0 127.0.0.1:48908        127.0.0.1:11377        ESTABLISHED
tcp6       0      0 127.0.0.1:49312        127.0.0.1:11377        ESTABLISHED
tcp6       0      0 192.168.30.43:39188    192.168.30.43:7919     ESTABLISHED
tcp6       0      0 127.0.0.1:37348        127.0.0.1:43843        ESTABLISHED
tcp6       0      0 127.0.0.1:11377        127.0.0.1:49312        ESTABLISHED
tcp6       0      0 127.0.0.1:48840        127.0.0.1:11377        ESTABLISHED
tcp6       0      0 127.0.0.1:49124        127.0.0.1:11377        TIME_WAIT
tcp6       0      0 127.0.0.1:11377        127.0.0.1:48908        ESTABLISHED
tcp6       0      0 127.0.0.1:11377        127.0.0.1:48840        ESTABLISHED
tcp6       0      0 192.168.30.43:7919     192.168.30.43:39188    ESTABLISHED
tcp6       0      0 127.0.0.1:43843        127.0.0.1:37348        ESTABLISHED
[root@DIWRND254 Linux_Tutorial_Test]#
```

The netstat -tn command displays a list of all TCP (Transmission Control Protocol) connections that are currently established on your system, including the local address and port number, the foreign address and port number, and the status of each connection.



# System Monitoring.

(Introduction to commands for monitoring system resources, such as top, htop, and df.)

## System Monitoring:-

It is process of observing various components of system such as CPU usage, Memory usage, disk space, network activity. To identify performance issues, detect anomalies, and ensure optimal system performance.

### 1. top command:-

The top command is used for real-time monitoring of system processes and resource utilization. It provides a dynamic view of the current state of the system, allowing you to see the CPU usage, memory usage, running processes, and other important system statistics.

Example:-

Type "top" and hit enter.

```
top - 17:45:22 up 1 day, 5:00, 2 users, load average: 0.00, 0.01, 0.05
Tasks: 168 total, 1 running, 167 sleeping, 0 stopped, 0 zombie
%Cpu(s): 0.2 us, 0.2 sy, 0.0 ni, 99.7 id, 0.0 wa, 0.0 hi, 0.0 si, 0.0 st
KiB Mem : 3861288 total, 2581208 free, 407372 used, 872708 buff/cache
KiB Swap: 2621436 total, 2621436 free, 0 used. 3206008 avail Mem
```

PID	USER	PR	NI	VIRT	RES	SHR	S	%CPU	%MEM	TIME+	COMMAND
2648	root	20	0	162096	2288	1572	R	0.7	0.1	0:00.07	top
2420	root	20	0	1026716	35448	13124	S	0.3	0.9	0:03.96	containerd
1	root	20	0	128148	6768	4176	S	0.0	0.2	0:02.67	systemd
2	root	20	0	0	0	0	S	0.0	0.0	0:00.01	kthreadd
4	root	0	-20	0	0	0	S	0.0	0.0	0:00.00	kworker/0:0H
6	root	20	0	0	0	0	S	0.0	0.0	0:00.12	ksoftirqd/0
7	root	rt	0	0	0	0	S	0.0	0.0	0:00.14	migration/0
8	root	20	0	0	0	0	S	0.0	0.0	0:00.00	rcu_bh
9	root	20	0	0	0	0	S	0.0	0.0	0:07.34	rcu_sched
10	root	0	-20	0	0	0	S	0.0	0.0	0:00.00	lru-add-drain
11	root	rt	0	0	0	0	S	0.0	0.0	0:00.53	watchdog/0
12	root	rt	0	0	0	0	S	0.0	0.0	0:00.36	watchdog/1
13	root	rt	0	0	0	0	S	0.0	0.0	0:00.06	migration/1
14	root	20	0	0	0	0	S	0.0	0.0	0:00.14	ksoftirqd/1
16	root	0	-20	0	0	0	S	0.0	0.0	0:00.00	kworker/1:0H
18	root	20	0	0	0	0	S	0.0	0.0	0:00.00	kdevtmpfs
19	root	0	-20	0	0	0	S	0.0	0.0	0:00.00	netns
20	root	20	0	0	0	0	S	0.0	0.0	0:00.07	khungtaskd
21	root	0	-20	0	0	0	S	0.0	0.0	0:00.00	writeback
22	root	0	-20	0	0	0	S	0.0	0.0	0:00.00	kintegrityd
23	root	0	-20	0	0	0	S	0.0	0.0	0:00.00	bioaset
24	root	0	-20	0	0	0	S	0.0	0.0	0:00.00	bioaset
25	root	0	-20	0	0	0	S	0.0	0.0	0:00.00	bioaset
26	root	0	-20	0	0	0	S	0.0	0.0	0:00.00	kblockd
27	root	0	-20	0	0	0	S	0.0	0.0	0:00.00	md
28	root	0	-20	0	0	0	S	0.0	0.0	0:00.00	edac-poller

Header: The header displays general system information, including the current time, system uptime, number of users, and the load average of the system over the past 1, 5, and 15 minutes.



Tasks: This line shows the total number of tasks/processes on the system, categorized by their state (running, sleeping, stopped, or zombie).

%Cpu(s): This line provides the CPU usage breakdown. It shows the percentage of CPU usage in different states such as user space (us), system/kernel space (sy), nice (ni) processes, idle (id), waiting for I/O (wa), hardware interrupts (hi), software interrupts (si), and stolen time (st).

MiB Mem: This line displays information about system memory (RAM) usage. It shows the total memory, free memory, used memory, and memory used for buffers/cache.

MiB Swap: This line provides information about the swap space usage. It shows the total swap space, free swap space, used swap space, and the amount of available memory.

Process List: The process list displays information about individual processes. Each row represents a process and provides details such as the process ID (PID), the user running the process, the process priority (PR), the virtual memory size (VIRT), the resident set size (RES), the shared memory size (SHR), the process state (S), the percentage of CPU usage (%CPU), the percentage of memory usage (%MEM), the total CPU time consumed (TIME+), and the command name (COMMAND).

## 2. Vmstat

The vmstat command is used to report virtual memory statistics and provide information about system processes, memory, swap, and CPU usage. It offers valuable insights into system performance and resource utilization.

```
[root@DIWRND254 ~]# vmstat
procs -----memory----- --swap-- ----io---- -system-- -----cpu-----
r  b   swpd   free   buff  cache   si   so    bi    bo    in   cs us sy id wa st
1  0       0 2583100   2108 870668    0    0     1     4   21   27  0  0 100  0  0
[root@DIWRND254 ~]#
```

procs: This section provides information about processes and their status. It includes the number of processes in the running state (r) and the number of processes in the uninterruptible sleep state (b).

memory: The memory section displays memory-related statistics. It includes the amount of virtual memory used (swpd), the amount of free

memory (free), the amount of memory used as buffers (buff), and the amount of memory used as cache (cache).

swap: The swap section provides information about swap space usage. It includes the amount of swap space used (si) and the amount of swap space swapped out to disk (so).

io: The io section displays input/output (I/O) statistics. It includes the number of blocks received from block devices (bi) and the number of blocks sent to block devices (bo).

system: The system section provides information about system-related activities. It includes the number of interrupts per second (in) and the number of context switches per second (cs).

cpu: The cpu section displays CPU usage statistics. It shows the percentage of CPU time spent in user space (us), system/kernel space (sy), idle state (id), and waiting for I/O (wa). Additionally, it may show the percentage of CPU time spent on handling hardware interrupts (hi), software interrupts (si), and virtualization (st).

### 3. Free

The free command is used to display information about system memory usage, including the total, used, and free memory. It provides a quick overview of the memory allocation on the system.

```
[root@DIWRND254 ~]# free
              total        used         free       shared  buff/cache   available
Mem:          3861288      405512      2583000         11856        872776       3207852
Swap:          2621436           0       2621436
```

total: The total field represents the total amount of physical memory (RAM) available on the system. In the example, the system has a total of 2,048,000 KB (2 GB) of memory.

used: The used field displays the total amount of memory that is currently in use by processes and the operating system. In the example, 1,724,000 KB (1.7 GB) of memory is being used.

free: The free field indicates the amount of memory that is currently not being used and is available for allocation. In the example, there are 26,352 KB (26.3 MB) of free memory.

shared: The shared field represents the amount of memory that is used for shared memory segments in the system.

buff/cache: The buff/cache field displays the amount of memory used for buffers and cache. Buffers are used to temporarily store data being read from or written to a disk, while cache represents the memory used for caching file system data.

available: The available field represents an estimate of the memory available for starting new applications without swapping. It takes into account the current memory usage, as well as the memory that can be reclaimed from buffers and cache.

Swap: The Swap section provides information about the system's swap space, which is a portion of the hard disk used as virtual memory. It shows the total swap space available, the amount of swap space in use, and the amount of swap space free.

#### 4. Iostat -d

The iostat -d command is used to display disk input/output (I/O) statistics for block devices on your system. It provides information about the performance and utilization of your disks.

```
[root@DIWRND254 ~]# iostat -d
Linux 3.10.0-1160.el7.x86_64 (DIWRND254)      Thursday 08 June 2023    _x86_64_      (2 CPU)

Device:            tps    kB_read/s    kB_wrtn/s    kB_read    kB_wrtn
sda                 0.15         2.20         7.11      239005      772336
dm-0                0.13         1.91         7.09      207131      770247
dm-1                0.00         0.02         0.00         2204         0
[root@DIWRND254 ~]#
```

Device: The Device column shows the name of the block device (e.g., hard disk, solid-state drive) being monitored.

tps: The tps (transactions per second) field represents the number of I/O transactions issued to the device per second. It indicates the rate of I/O operations performed on the device.

kB\_read/s: The kB\_read/s field shows the amount of data read from the device per second, measured in kilobytes.

kB\_wrtn/s: The kB\_wrtn/s field displays the amount of data written to the device per second, measured in kilobytes.

kB\_read: The kB\_read field indicates the total amount of data read from the device since the system was booted, measured in kilobytes.

kB\_wrtn: The kB\_wrtn field represents the total amount of data written to the device since the system was booted, measured in kilobytes.

# Shell Scripting.

(Introduce basic shell scripting concepts, including variables, loops, conditionals, and basic scripting syntax using bash, the default Linux shell.)

## 1. Variables:

Variables in bash are used to store data and are represented by a name followed by an equal sign and the value assigned to it.

Example:-

```
[root@DIWRND254 ~]# Name=Aniket
[root@DIWRND254 ~]# Age=24
[root@DIWRND254 ~]# echo "Name: $Name Age: $Age"
Name: Aniket Age: 24
[root@DIWRND254 ~]#
```

## 2. Conditionals:

Conditionals in bash are used to make decisions based on certain conditions. The if statement is commonly used for conditional branching.

Example:-

```
[root@DIWRND254 ~]# cat Sample.sh
Name=Aniket
Age=24
echo "Name: $Name Age: $Age"
if [ $Age -ge 18 ]; then
    echo "Age is greater than 18"
else
    echo "Age is less than 18"
fi
[root@DIWRND254 ~]# ./Sample.sh
Name: Aniket Age: 24
Age is greater than 18
[root@DIWRND254 ~]#
```

Name=Aniket

Age=24

echo "Name: \$Name Age: \$Age"

In these lines, two variables, Name and Age, are defined and assigned values. Name is set to "Aniket" and Age is set to 24. The echo statement is then used to print the values of these variables, along

with some additional text. The variables are referenced using the \$ symbol and their respective names.

```
if [ $Age -ge 18 ]; then
    echo "Age is greater than 18"
else
    echo "Age is less than 18"
fi
```

This section uses an if statement to check whether the value of the Age variable is greater than or equal to 18. The condition is specified within square brackets [ ]. The -ge operator is used for numeric comparison, where -ge stands for "greater than or equal to". If the condition is true, meaning the age is greater than or equal to 18, it prints the message "Age is greater than 18". Otherwise, it prints the message "Age is less than 18".

When the script is executed (./Sample.sh), the output will be:

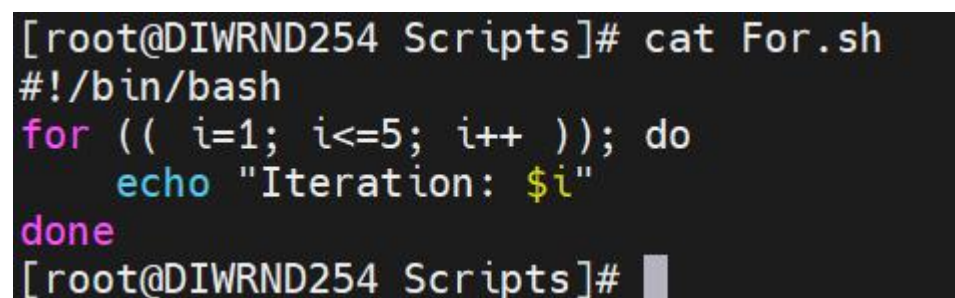
```
Name: Aniket Age: 24
Age is greater than 18
```

This indicates that the script successfully prints the name and age, and determines that the age is greater than 18 based on the condition.

### 3. Loops

A loop is a block of code that allows you to repeat a specific task or a set of tasks multiple times.

#### For Loop



```
[root@DIWRND254 Scripts]# cat For.sh
#!/bin/bash
for (( i=1; i<=5; i++ )); do
    echo "Iteration: $i"
done
[root@DIWRND254 Scripts]#
```

In this script, the for loop iterates from i=1 to i<=5 with an increment of 1 in each iteration. Within the loop, the echo command is used to print the current value of i, along with the text "Iteration:".

```
[root@DIWRND254 Scripts]# ./For.sh  
Iteration: 1  
Iteration: 2  
Iteration: 3  
Iteration: 4  
Iteration: 5  
[root@DIWRND254 Scripts]# █
```