Linux Fundamentals & Networking in Linux (TCSCCS0204T)

~ by Asst. Prof. Shivkumar R. Chandey,
Department of Computer Science,
Thakur College of Science & Commerce (Autonomous College)

Objectives

- This course introduces various tools and techniques commonly used by Linux programmers, system administrators and end users to achieve their day to day work in Linux environment. It is designed for computer learners who have limited or no previous exposure to Linux.

Unit I Syllabus

Unit I	**Introduction** History of Linux, Philosophy, Community, Terminology, Distributions, Linux kernel vs distribution. Importance of Linux in software ecosystem: web servers, supercomputers, mobile, servers. Linux Structure, Linux Architecture, Filesystem basics. The boot process, init scripts, runlevels, shutdown process, Very basic introductions to Linux processes **Packaging methods** rpm/deb, Graphical Vs Command line.

Text Books and Reference Books

- Text Books
  - Unix Concepts and Applications by Sumitabha Das, 4th Ed, 2017
  - Official Ubuntu Book, 8th Edition, by Matthew Helmke & Elizabeth K. Joseph with Jose Antonio Rey and Philips Ballew, Prentice Hall
- Additional References
  - Linux kernel Home: http://kernel.org
  - Open Source Initiative: https://opensource.org/
  - The Linux Foundation: http://www.linuxfoundation.org/

Expected Learning Outcomes

1. Upon completion of this course, Learners should have a good working knowledge of Linux, from both a graphical and command line perspective, allowing them to easily use any Linux distribution.

2. This course shall help student to learn advanced subjects in computer science practically.

3. Student shall be able to progress as a Developer or Linux System Administrator using the acquired skill set.

History of Linux

Evolution of Computer

- In earlier days, computers were as big as houses or parks. So you can imagine how difficult it was to operate them.
- Moreover, every computer has a different operating system which made it completely worse to operate on them.
- Every software was designed for a specific purpose and was unable to operate on other computer.
- It was extremely costly and normal people neither can afford it nor can understand it.

Evolution of Unix

- In 1969, a team of developers of Bell Labs started a project to make a common software for all the computers and named it as 'Unix'.
- It was simple and elegant, used 'C' language instead of assembly language and its code was recyclable.
- As it was recyclable, a part of its code now commonly called 'kernel' was used to develop the operating system and other functions and could be used on different systems. Also its source code was open source.
- Initially, Unix was only found in large organizations like government, university, or larger financial corporations with mainframes and minicomputers (PC is a microcomputer).

Unix Expansion

- In eighties, many organizations like IBM, HP and dozen other companies started creating their own Unix. It result in a mess of Unix dialects.
- Then in 1983, Richard Stallman developed GNU project with the goal to make it freely available Unix like operating system and to be used by everyone. But his project failed in gaining popularity.
- Many other Unix like operating system came into existence but none of them was able to gain popularity.

Evolution of Linux

- In 1991, Linus Torvalds a student at the university of Helsinki, Finland, thought to have a freely available academic version of Unix started writing its own code.
- Later this project became the Linux kernel. He wrote this program specially for his own PC as he wanted to use Unix 386 Intel computer but couldn't afford it. He did it on MINIX using GNU C compiler.
- GNU C compiler is still the main choice to compile Linux code but other compilers are also used like Intel C compiler.
- He started it just for fun but ended up with such a large project. Firstly he wanted to name it as 'Freax' but later it became 'Linux'.
- He published the Linux kernel under his own license and was restricted to use as commercially. Linux uses most of its tools from GNU software and are under GNU copyright. In 1992, he released the kernel under GNU General Public License.

Linux Today

- Today, supercomputers, smart phones, desktop, web servers, tablet, laptops and home appliances like washing machines, DVD players, routers, modems, cars, refrigerators, etc use Linux OS.

Philosophy

- All operating systems have a philosophy. And, the philosophy of an operating system matters.
- Linux treats everyone equally and allows everyone the maximum amount of power. That is egalitarian.
- Other operating systems are elitist and exclusive because they withhold or hide their power behind an inflexible Graphical User Interface that allows one to do only what the developers think we should be allowed to do.
- The Linux philosophy is epitomized by the ease with which one can open a terminal emulator to access the CLI and its concomitant power.

Linux Distributions (Distros)

- Other operating systems like Microsoft combine each bit of codes internally and release it as a single package. You have to choose from one of the version they offer.
- But Linux is different from them. Different parts of Linux are developed by different organizations.
- Different parts include kernel, shell utilities, X server, system environment, graphical programs, etc. If you want you can access the codes of all these parts and assemble them yourself.
- From here on distribution (also called as distros) comes into the picture. They assemble all these parts for us and give us a compiled operating system of Linux to install and use.

Linux Distributions List

- There are on an average six hundred Linux distributors providing different features. Here, we'll discuss about some of the popular Linux distros today.

1) Ubuntu

- It came into existence in 2004 by Canonical and quickly became popular. Canonical wants Ubuntu to be used as easy graphical Linux desktop without the use of command line.
- It is the most well known Linux distribution. Ubuntu is a next version of Debian and easy to use for newbies.
- It comes with a lots of pre-installed apps and easy to use repositories libraries.
- Earlier, Ubuntu uses GNOME2 desktop environment but now it has developed its own unity desktop environment. It releases every six months and currently working to expand to run on tablets and smartphones.

Choosing a Linux Distro

Distribution	Why To Use
UBuntu	It works like Mac OS and easy to use.
Linux mint	It works like windows and should be use by new comers.
Debian	It provides stability but not recommended to a new user.
Fedora	If you want to use red hat and latest software.
Red hat enterprise	To be used commercially.
CentOS	If you want to use red hat but without its trademark.
OpenSUSE	It works same as Fedora but slightly older and more stable.
Arch Linux	It is not for the beginners because every package has to be installed by yourself.

Linux Licensing

- Linus Torvalds has given linux kernel license to GNU General Public License (GPL) version 2.
- GNU make sure that any software source code licensed under it have to make originating source code open and freely available to all its users.
- Here, freely doesn't mean by cost but it means that it is freely available to users to distribute and modify the code.
- There is the third version of GNU, GNU Lesser General Public License (LGPL) version 3. But it imposes some more permissions on the license.

Terminology

1. Command

- ‘Commands’ are prompts that you type into a program called the ‘command line’, which enable you to give instructions to your OS.
- You can use multiple applications to access the command line, so the way it looks can vary, but it’s usually a simple window:

2. Distribution

- As we mentioned, different versions of Linux are called ‘distros’. Anyone can create their own Linux distro, so you have a lot of options to choose from.
- If you’re thinking about switching OSs, it’s important to read up on some of the most popular distros available, and consider using a bootable USB to try out whichever one strikes your fancy.

3. GNOME

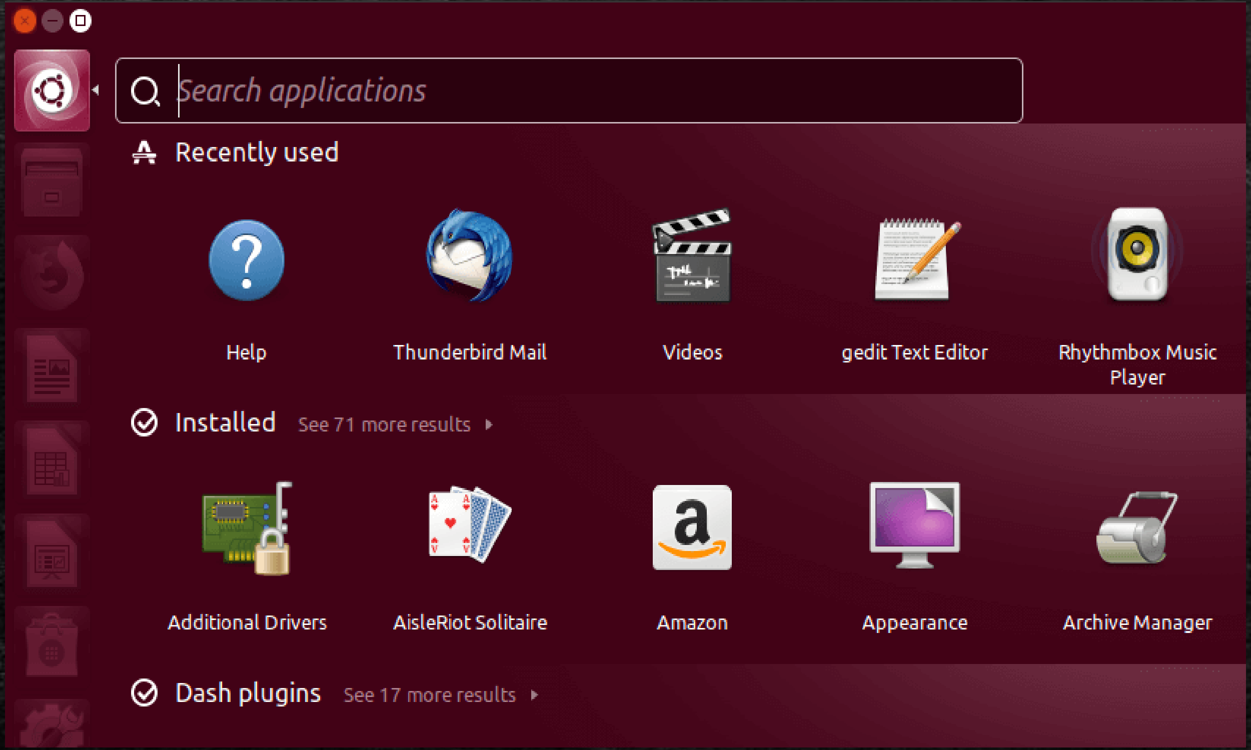
- ‘GNU Network Object Model Environment (GNOME)’ is a visual desktop interface used by several Linux distros. In fact, it’s very similar to the Windows desktop.
- Therefore, distros that use GNOME can be a great starting point if you’re jumping over from that OS.

4. GNU

- The ‘GNU’ project is a collection of free software that includes some of Linux’s most popular applications.
- It’s developed by the same people behind the General Public License (GPL), which is a big hit in the open source world.
- Simply put, you can do almost anything you want with software published under the GPL license, including modifying and sharing it (under the same license).

5. Unity

- As you already know, GNOME is a desktop environment built for Linux systems. Unity is an interface built for GNOME, which offers one of the most user-friendly experiences available for Linux users:



6. Root

- Linux OSs have a built-in system of user roles. Each user has a designated role, with varying levels of permissions. For example, if you're a guest, you won't be able to modify the OS's core files.
- A 'root' account, on the other hand, has full access to every command and file in the system. That is to say, if you're a root user, you can do just about anything you want.

7. Terminal

- Your ‘terminal’ is the primary way you’ll interact with Linux. This is where you enter all of your commands, and the interface tends to be very straightforward.
- However, you can also opt to use terminal emulators, which are software options that provide you with a more user-friendly interface.

8. Package Manager

- When it comes to Linux, you install ‘packages’ rather than programs. Typically, you’ll do this through the terminal.
- A ‘package manager’ is a tool that provides you with a graphical interface to help you find new packages, then install, update, and even configure them.

9. Binaries

- A ‘binary’ file isn’t composed of regular text, but rather is made up of computer code.
- In many cases, binary files on Linux are executable, much like Windows .exe files. In other words, they can be run in order to perform some task or functionality.

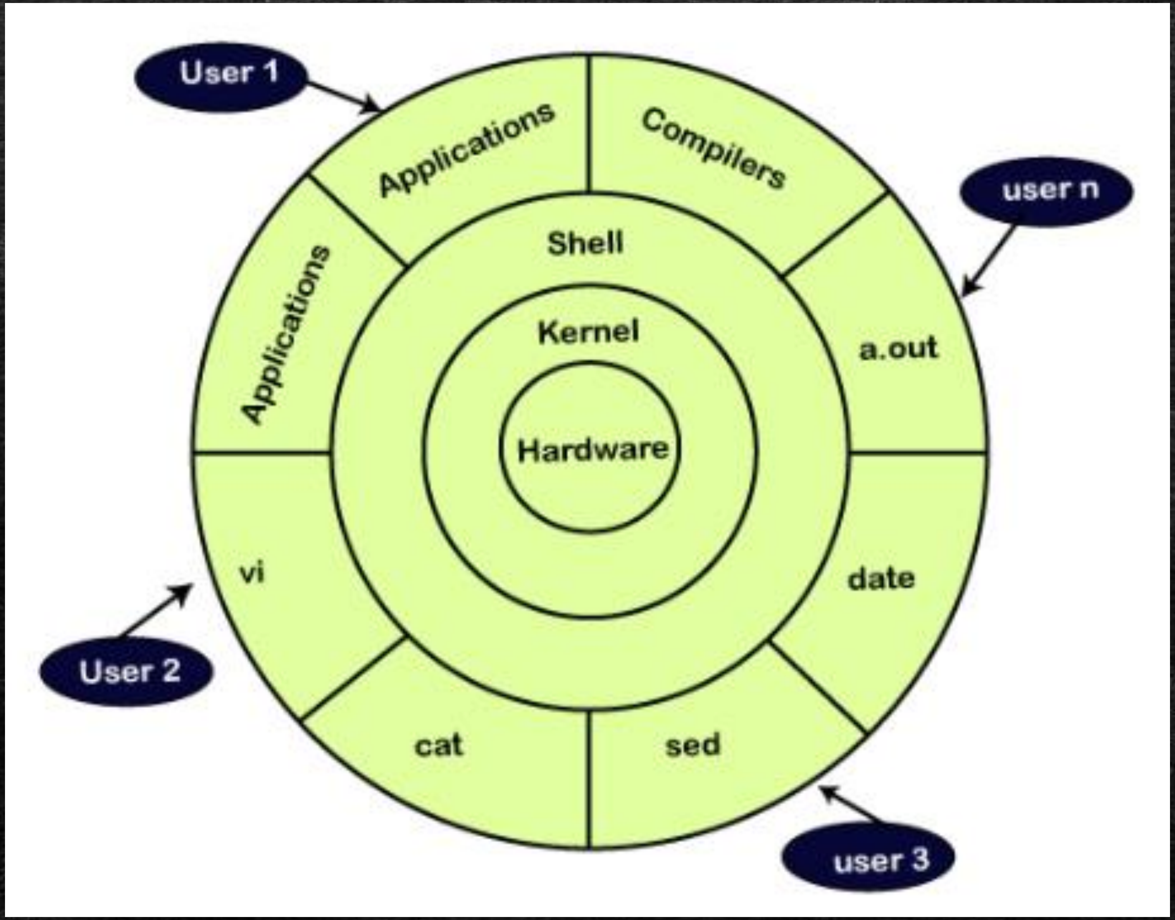
10. Kernel

- ‘Kernels’ are at the core of every OS. What a kernel does is manage your system’s hardware, along with all the programs on your computer.
- The base Linux kernel is open-source, and it provides the core for a lot of distributions, some of which modify that code to better suit their particular goals.

Kernel

- Linux kernel is the core part of the operating system.
- It establishes communication between devices and software. Moreover, it manages system resources.
- It has four responsibilities:
  - Device Management
  - Memory Management
  - Process Management
  - Handling System Calls

Kernel



User 1
Applications
Compilers
user n
Applications
Shell
Kernel
a.out
Hardware
vi
date
User 2
cat
sed
user 3

Device Management

- A system has many devices connected to it like CPU, a memory device, sound cards, graphic cards, etc.
- A kernel stores all the data related to all the devices in the device driver (without this kernel won't be able to control the devices).
- Thus kernel knows what a device can do and how to manipulate it to bring out the best performance.
- It also manages communication between all the devices.
- The kernel has certain rules that have to be followed by all the devices.

Memory Management

- Another function that kernel has to manage is the memory management.
- The kernel keeps track of used and unused memory and makes sure that processes shouldn't manipulate data of each other using virtual memory addresses.

Process Management

- In the process, management kernel assigns enough time and gives priorities to processes before handling CPU to other processes.
- It also deals with security and ownership information.

Handling System Calls

- Handling system calls means a programmer can write a query or ask the kernel to perform a task.

Linux kernel vs distribution

- A kernel is the most low-level piece of the rather complicated architecture that makes up an operating system, and even within the kernel there are tons of layers.
- But in a nutshell, the kernel interacts with the hardware, and most of what builds on top of that, is stuff that Ubuntu includes in its distribution.
- The kernel is an abstraction layer between the hardware and the apps.
- Saying that the kernel does "all the heavy lifting" may be over-simplifying things, but a kernel does some extremely heavy lifting nonetheless.
- Everything that doesn't use a terminal interface, but a graphical one, is Ubuntu stuff.

Linux kernel vs distribution

- What All Distros Have in Common: the Linux Kernel
- All distros have at least one thing in common: the Linux kernel. This piece of software is the core of the operating system, bridging the software that you interact with (e.g. the browser) with the underlying hardware that does all the work.
- It also includes many device drivers to provide support for whatever hardware you may be sporting.
- That's why it's important to keep the kernel updated or to compile the kernel yourself if you have special requirements.
- Developers around the world contribute to the kernel, along with its creator, Linus Torvalds.

Importance of Linux in software ecosystem: web servers, supercomputers, mobile, servers.

Importance of Linux in software ecosystem

- Linux has many variations and distributions because of its modular design.
- A kernel is at the base or core of the Linux system. It schedules applications or processes, manages basic peripheral devices, handles network access.
- Linux provides many advantages over other operating systems and that is why it is used almost in every field nowadays, from smartphones to supercomputers, cars to home appliances and many more.

Generic Nature of Linux Kernel

- Linux kernel is generic, as much as possible. This implies that single source code can be written to run on large supercomputers and also on small even hand-held gadgets; this is entirely up to user how one uses Linux, either on giant systems or smaller systems.
- There is no need to add fundamental and large changes to the kernel in order to run on larger or smaller systems.
- Typically Linux kernel can be configured to be as small as 2Mb or as large as 1G or 1T without impending time and effort.

Scalability

- Scalability can be defined as the ability of the server to adapt to larger loads. Scalability can be directly thought of as a measure of efficiency, performance. System must be such that addition of new server should be painless.
- Linux has tremendous scalability as it can accommodate the new and higher loads rather easily. This why you can find Linux run supercomputers and Android (using Linux kernel) on mobile phones, refrigerators and even microwave ovens!

Open Source Nature

- One of the main advantages of Linux is that it is an open source operating system i.e. its source code is easily available for everyone. Anyone capable of coding can contribute, modify, enhance and distribute the code to anyone and for any purpose.
- This implies that supercomputer administrators can customize the OS to any level.

Community Support

- Linux being Open source has immense community support that is unparalleled on any other operating system.

Free to use (Low Cost)

- Cost can be of major concern when it comes to huge devices, one like supercomputers.
- Deploying Linux on supercomputers is cost effective as Linux is completely royalty free.
- Linux is freely available on the web to download and use. You do not need to buy the license for it as Linux and many of its software come with GNU General Public License.
- This proved to be one of the major advantages Linux faces over Windows and other operating systems. You need to spend a huge amount to buy the license of Windows which is not the case with Linux.

Security

- Linux is more secure in comparison to other operating systems such as Windows.
- Linux is not completely secure as there is some malware for it also but it is less vulnerable than others.
- Every program in Linux whether an application or a virus needs authorization from the administrator in the form of a password. Unless the password is typed virus won't execute. There is no requirement of any anti-virus program in Linux.

Revive older computer systems

- Linux helps you to use or utilize your old and outdated computer systems as a firewall, router, backup server or file server and many more.
- There are many distributions available to use according to your system capability.
- As you can use Puppy Linux for low- end systems.

Software Updates

- In Linux you encounter a larger number of software updates. These software updates are much faster than updates in any other operating system. Updates in Linux can be done easily without facing any major issue or concern.

Customization

- A feature that gives a major advantage over other operating systems is customization.
- You can customize any feature, add or delete any feature according to your need as it is an open source operating system.
- Not only this, various wallpapers and attractive icon themes can be installed to give an amazing look to your system.

Various Distributions

- There are many distributions available also called distros of Linux. It provides various choices or flavors to the users.
- You can select any distros according to your needs. Some distros of Linux are Fedora, Ubuntu, Arch Linux, Debian, Linux Mint and many more. If you are a beginner you can use Ubuntu or Linux Mint. If you are a good programmer you may use Debian or Fedora.

Stability (Reliability)

- Linux provides high stability also this is good advantage i.e. it does not need to be rebooted after a short period of time.
- Your Linux system rarely slows down or freezes. As in windows, you need to reboot your system after installing or uninstalling an application or updating your software but this is not the case with Linux.
- You can work without any disturbance on your Linux systems.

Privacy

- Linux ensures the privacy of user's data as it never collects much data from the user while using its distributions or software but this is not true for many other operating systems.

Performance

- Linux provides high performance on various networks and workstations. It allows a large number of users to work simultaneously and handles them efficiently.

Network Support

- Linux gives support for network functionality as it was written by programmers over the internet. Linux helps you to set up client and server systems on your computer systems easily and in a fast manner.

Flexibility

- Linux provides a high range of flexibility as you can install only required components. There is no need to install a full or complete suite. You can also keep Linux file under multiple partitions so if one of them corrupts then there is no major loss. You only need to repair that particular partition, not the complete file which is not the case with other operating systems.

Compatibility

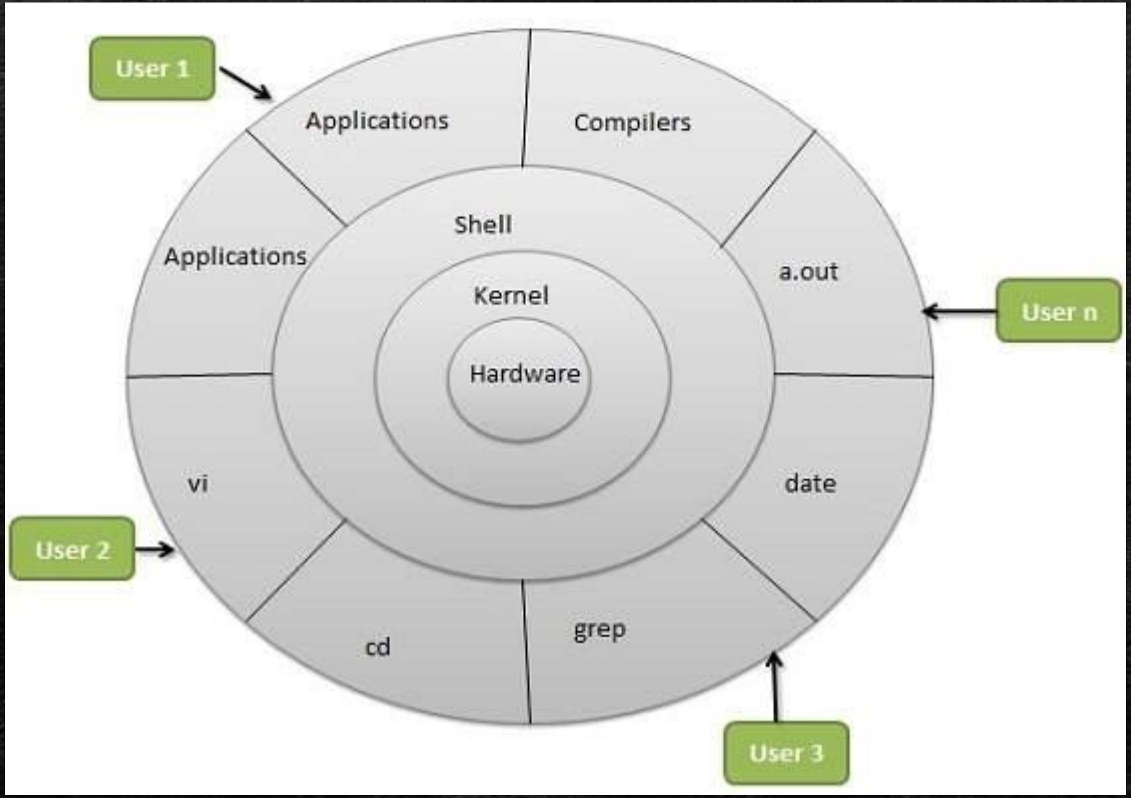
- Linux runs or executes all possible file formats and is compatible with a large number of file formats.

Multitasking

- Linux is a multitasking operating system as it can perform many tasks simultaneously without any decrease in its speed such as downloading a large file would not slow down the system.

Linux Architecture

- The following illustration shows the architecture of a Linux system:



Linux Architecture

The architecture of a Linux System consists of the following layers:

- Hardware layer – Hardware consists of all peripheral devices (RAM/ HDD/ CPU etc).
- Kernel – It is the core component of Operating System, interacts directly with hardware, provides low level services to upper layer components.
- Shell – An interface to kernel, hiding complexity of kernel's functions from users. The shell takes commands from the user and executes kernel's functions.
- Utilities – Utility programs that provide the user most of the functionalities of an operating systems.

Linux File Structure

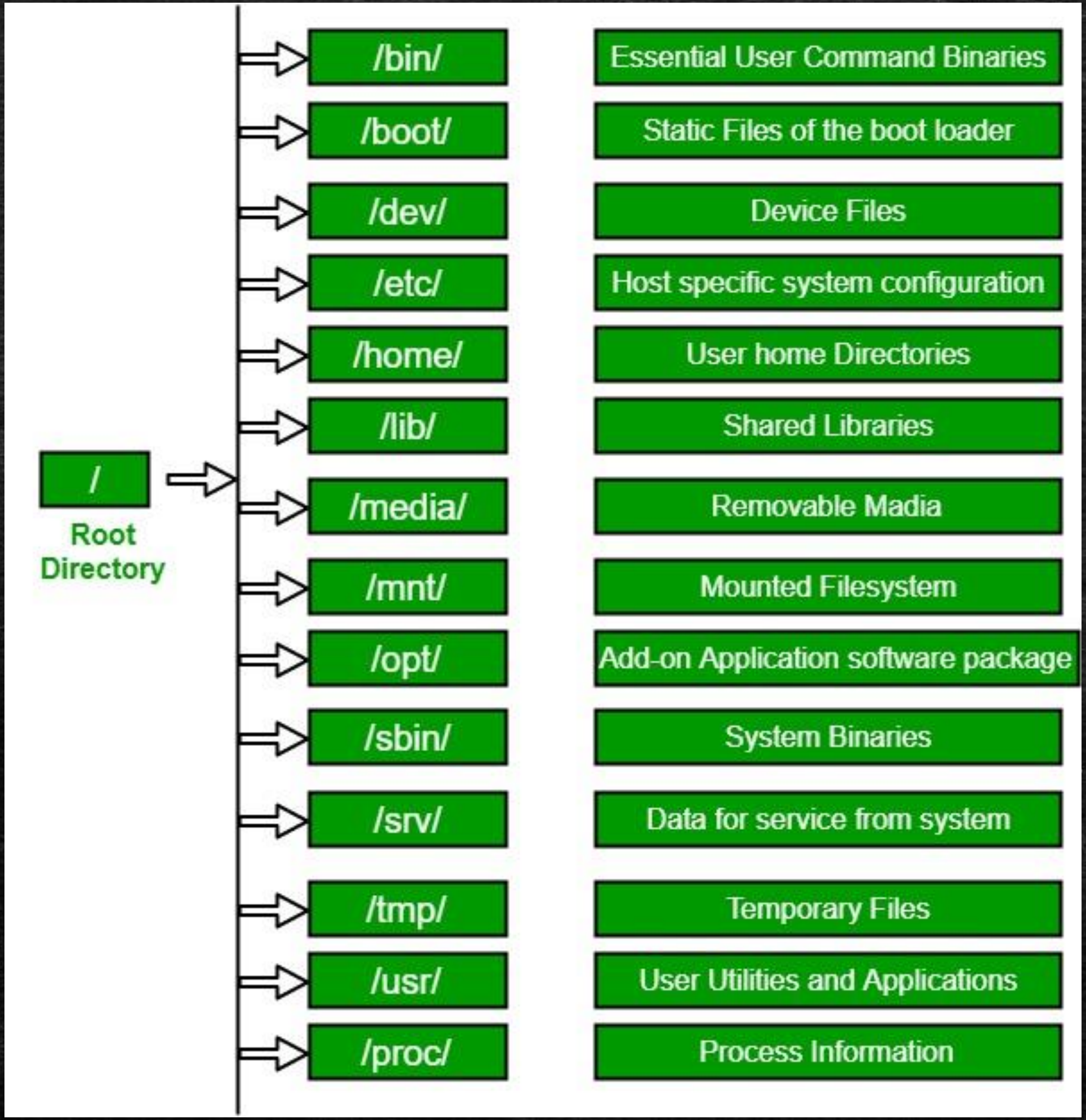
- In the Linux file structure files are grouped according to purpose.
- For example: commands, data files, documentation. Parts of a Unix directory tree are listed below.
- All directories are grouped under the root entry "/".
- That part of the directory tree is left out of the below diagram. See the FSSTND standard (Filesystem standard).

Linux File Structure

- root - The home directory for the root user
- home - Contains the user's home directories along with directories for services
- bin - Commands needed during boot up that might be needed by normal users
- sbin - Like bin but commands are not intended for normal users. Commands run by LINUX.
- proc - This file system is not on a disk. It is a virtual file system that exists in the kernels imagination which is memory.

Linux File Structure

- 1 - A directory with info about process number 1. Each process has a directory below proc.
- usr - Contains all commands, libraries, man pages, games and static files for normal operation.
- bin - Almost all user commands. some commands are in /bin or /usr/local/bin.
- sbin - System admin commands not needed on the root file system. e.g., most server programs.
- include - Header files for the C programming language. Should be below /user/lib for consistency.

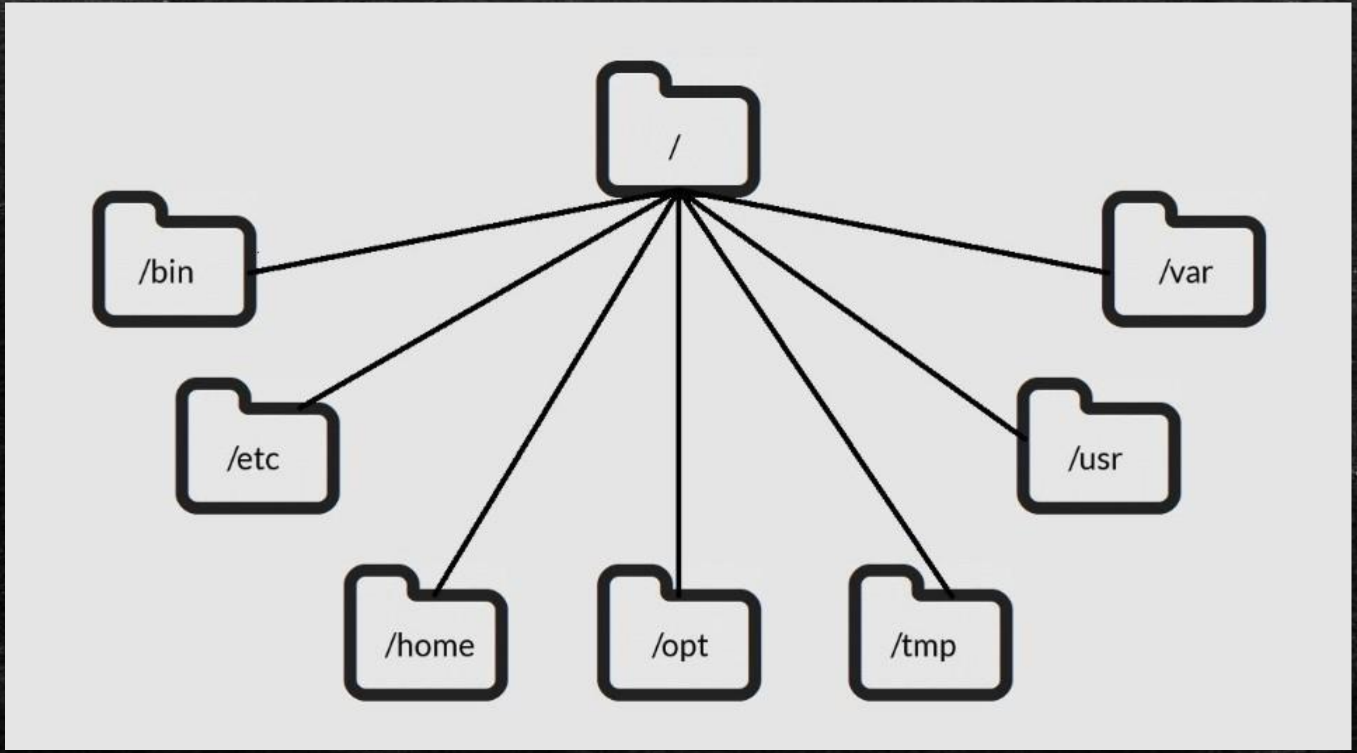

/
Root Directory
/bin/
Essential User Command Binaries
/boot/
Static Files of the boot loader
/dev/
Device Files
/etc/
Host specific system configuration
/home/
User home Directories
/lib/
Shared Libraries
/media/
Removable Madia
/mnt/
Mounted Filesystem
/opt/
Add-on Application software package
/sbin/
System Binaries
/srv/
Data for service from system
/tmp/
Temporary Files
/usr/
User Utilities and Applications
/proc/
Process Information

Directory Structure

- Unix uses a hierarchical file system structure, much like an upside-down tree, with root (/) at the base of the file system and all other directories spreading from there.
- A Unix filesystem is a collection of files and directories that has the following properties –
- It has a root directory (/) that contains other files and directories.
- Each file or directory is uniquely identified by its name, the directory in which it resides, and a unique identifier, typically called an inode.
- By convention, the root directory has an inode number of 2 and the lost+found directory has an inode number of 3. Inode numbers 0 and 1 are not used. File inode numbers can be seen by specifying the -i option to ls command.
- It is self-contained. There are no dependencies between one filesystem and another.

Directory Structure

- The directories have specific purposes and generally hold the same types of information for easily locating files. Following are the directories that exist on the major versions of Unix



Linux Directory Commands

Directory Command	Description
pwd	The pwd command stands for (print working directory). It displays the current working location or directory of the user. It displays the whole working path starting with /. It is a built-in command.
ls	The ls command is used to show the list of a folder. It will list out all the files in the directed folder.
cd	The cd command stands for (change directory). It is used to change to the directory you want to work from the present directory.
mkdir	With mkdir command you can create your own directory.
rmdir	The rmdir command is used to remove a directory from your system.

Difference between Root and Home Directory

- Some key differences between root and home directory are as following:

Root Directory	Home Directory
The root directory is the topmost level of the system drive.	The home directory is a subdirectory of the root directory.
It is denoted by a slash '/'.	It is denoted by '~' and has path "**/users/username**".
The admin has access to make any changes in the files and settings.	No user other than the root user can change the settings of the entire system.
The admin can create a user.	Any user having a home directory cannot create a user.
In the Linux file system, everything comes under the root directory.	The home directory contains a particular user's data.

Shell

- A Shell provides you with an interface to the Unix system. It gathers input from you and executes programs based on that input. When a program finishes executing, it displays that program's output.
- Shell is an environment in which we can run our commands, programs, and shell scripts. There are different flavors of a shell, just as there are different flavors of operating systems. Each flavor of shell has its own set of recognized commands and functions.

Shell Types

- In Unix, there are two major types of shells –
- Bourne shell – If you are using a Bourne-type shell, the $ character is the default prompt.
- C shell – If you are using a C-type shell, the % character is the default prompt.

The Bourne Shell has the following subcategories

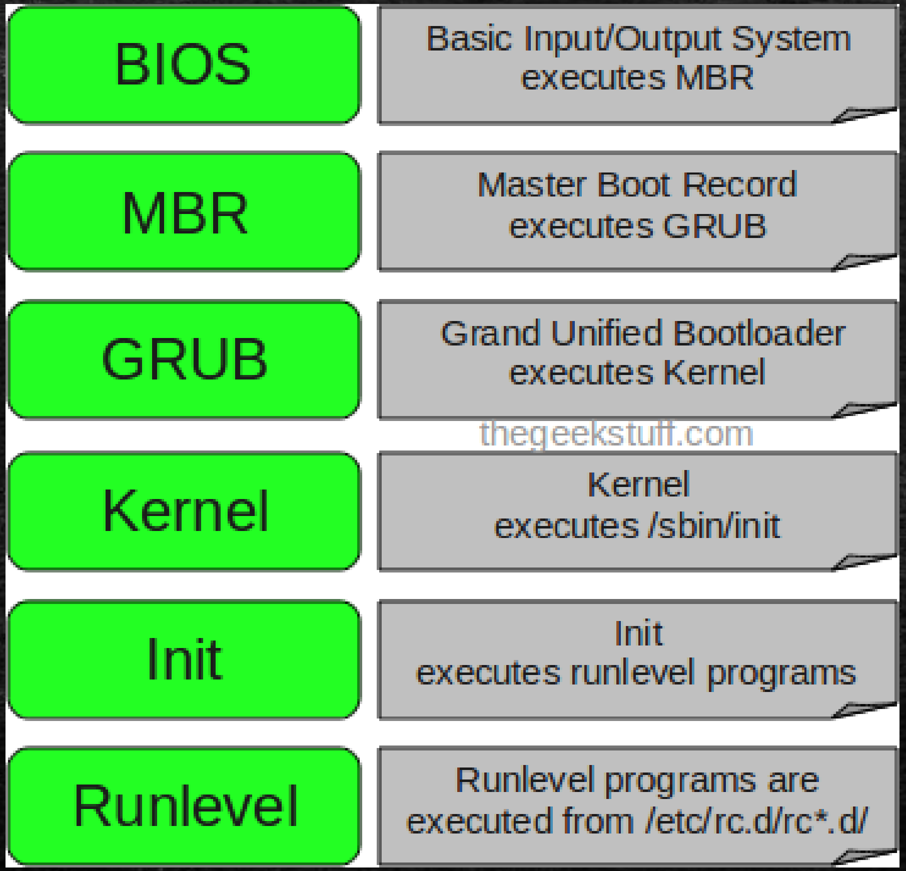
1. Bourne shell (sh)
2. Korn shell (ksh)
3. Bourne Again shell (bash)
4. POSIX shell (sh)

Linux Bash

- The Linux Bash is also known as 'Bourne-again Shell.' It is a command language interpreter for the Linux based system. It is a replacement of Bourne shell (sh). It was developed under the GNU Project and written by Brian Fox. Nowadays, Bash is the default user shell of most of the Linux distributions.
- The Linux/Unix shell allows us to interact with the Linux system through the commands. It let us invoke an executable file to create a running process. Moreover, it also allows us to interact with the Linux file system. It is designed in such a way that we can perform all the Linux operations through Bash.
- The Bash is a command language interpreter as well as a programming language. It supports variables, functions, and flow control, like other programming languages. It can also read and execute the commands from a file, which is called a shell script.

Linux BIOS

- The following are the 6 high level stages of a typical Linux boot process.



BIOS	Basic Input/Output System executes MBR
MBR	Master Boot Record executes GRUB
GRUB	Grand Unified Bootloader executes Kernel
Kernel	Kernel executes /sbin/init
Init	Init executes runlevel programs
Runlevel	Runlevel programs are executed from /etc/rc.d/rc*.d/

1. BIOS

- BIOS stands for Basic Input/Output System
- Performs some system integrity checks
- Searches, loads, and executes the boot loader program.
- It looks for boot loader in floppy, cd-rom, or hard drive. You can press a key (typically F12 of F2, but it depends on your system) during the BIOS startup to change the boot sequence.
- Once the boot loader program is detected and loaded into the memory, BIOS gives the control to it.
- So, in simple terms BIOS loads and executes the MBR boot loader.

2. MBR

- MBR stands for Master Boot Record.
- It is located in the 1st sector of the bootable disk. Typically /dev/hda, or /dev/sda
- MBR is less than 512 bytes in size. This has three components 1) primary boot loader info in 1st 446 bytes 2) partition table info in next 64 bytes 3) mbr validation check in last 2 bytes.
- It contains information about GRUB (or LILO in old systems).
- So, in simple terms MBR loads and executes the GRUB boot loader.

3. GRUB

- GRUB stands for Grand Unified Bootloader.
- If you have multiple kernel images installed on your system, you can choose which one to be executed.
- GRUB displays a splash screen, waits for few seconds, if you don't enter anything, it loads the default kernel image as specified in the grub configuration file.
- GRUB has the knowledge of the filesystem (the older Linux loader LILO didn't understand filesystem).
- Grub configuration file is /boot/grub/grub.conf (/etc/grub.conf is a link to this). The following is sample grub.conf of CentOS.

GRUB

```
#boot=/dev/sda
default=0
timeout=5
splashimage=(hd0,0)/boot/grub/splash.xpm.gz
hiddenmenu
title CentOS (2.6.18-194.el5PAE)
        root (hd0,0)
        kernel /boot/vmlinuz-2.6.18-194.el5PAE ro root=LABEL=/
        initrd /boot/initrd-2.6.18-194.el5PAE.img
```

- As you notice from the above info, it contains kernel and initrd image.
- So, in simple terms GRUB just loads and executes Kernel and initrd images.

4. Kernel

- Mounts the root file system as specified in the “root=” in grub.conf
- Kernel executes the /sbin/init program
- Since init was the 1st program to be executed by Linux Kernel, it has the process id (PID) of 1. Do a ‘ps -ef | grep init’ and check the pid.
- initrd stands for Initial RAM Disk.
- initrd is used by kernel as temporary root file system until kernel is booted and the real root file system is mounted. It also contains necessary drivers compiled inside, which helps it to access the hard drive partitions, and other hardware.

5. Init

- Looks at the /etc/inittab file to decide the Linux run level.
- Following are the available run levels
  - 0 – halt
  - 1 – Single user mode
  - 2 – Multiuser, without NFS
  - 3 – Full multiuser mode
  - 4 – unused
  - 5 – X11
  - 6 – reboot

Init

- Init identifies the default initlevel from /etc/inittab and uses that to load all appropriate program.
- Execute ‘grep initdefault /etc/inittab’ on your system to identify the default run level
- If you want to get into trouble, you can set the default run level to 0 or 6. Since you know what 0 and 6 means, probably you might not do that.
- Typically you would set the default run level to either 3 or 5.

6. Runlevel programs

- When the Linux system is booting up, you might see various services getting started. For example, it might say "starting sendmail .... OK". Those are the runlevel programs, executed from the run level directory as defined by your run level.

- Depending on your default init level setting, the system will execute the programs from one of the following directories.
  - Run level 0 – /etc/rc.d/rc0.d/
  - Run level 1 – /etc/rc.d/rc1.d/
  - Run level 2 – /etc/rc.d/rc2.d/
  - Run level 3 – /etc/rc.d/rc3.d/
  - Run level 4 – /etc/rc.d/rc4.d/
  - Run level 5 – /etc/rc.d/rc5.d/
  - Run level 6 – /etc/rc.d/rc6.d/

Runlevel programs

- Please note that there are also symbolic links available for these directory under /etc directly. So, /etc/rc0.d is linked to /etc/rc.d/rc0.d.
- Under the /etc/rc.d/rc*.d/ directories, you would see programs that start with S and K.
- Programs starts with S are used during startup. S for startup.
- Programs starts with K are used during shutdown. K for kill.
- There are numbers right next to S and K in the program names. Those are the sequence number in which the programs should be started or killed.
- For example, S12syslog is to start the syslog deamon, which has the sequence number of 12. S80sendmail is to start the sendmail daemon, which has the sequence number of 80. So, syslog program will be started before sendmail.

Shutdown process

- The shutdown command brings down system in a secure way. All the logged-in users are notified about the system shutdown.
- Signal SIGTERM notifies all the processes that the system is going down, so that processes can be saved and exit properly.
- Command shutdown signals the init process to change the runlevel.

Shutdown process

- Runlevel 0 halts the system
- Runlevel 6 reboots the system
- Runlevel 1 is default state.
- Five minutes before shutdown sequence starts, file /etc/nologin is created when shutdown is scheduled for future which does not allow new user logins.
- If by any reason, command shutdown is stopped before signalling init, this file is removed. It is also removed to change runlevel before signalling init.
- To run shutdown command root user access is required.

Shutting down system

- You can shutdown a system by passing a definite time (in minutes). System will automatically shutdown after specified minute giving a message and time to save all work.
- Syntax:

```
shutdown <time>
```

- Look at the above snapshot, message is displayed on the terminal.
- To immediately shutdown the system, use now option,

Processes Management

- When you execute a program on your Unix system, the system creates a special environment for that program.
- This environment contains everything needed for the system to run the program as if no other program were running on the system.
- Whenever you issue a command in Unix, it creates, or starts, a new process. When you tried out the ls command to list the directory contents, you started a process.
- A process, in simple terms, is an instance of a running program.

Processes Management

- The operating system tracks processes through a five-digit ID number known as the pid or the process ID. Each process in the system has a unique pid.
- Pids eventually repeat because all the possible numbers are used up and the next pid rolls or starts over.
- At any point of time, no two processes with the same pid exist in the system because it is the pid that Unix uses to track each process.

Starting a Process

- When you start a process (run a command), there are two ways you can run it –
  - Foreground Processes
  - Background Processes

Foreground Processes

- By default, every process that you start runs in the foreground. It gets its input from the keyboard and sends its output to the screen.
- You can see this happen with the ls command. If you wish to list all the files in your current directory, you can use the following command –
- $ls ch*.doc

Background Processes

- A background process runs without being connected to your keyboard. If the background process requires any keyboard input, it waits.
- The advantage of running a process in the background is that you can run other commands; you do not have to wait until it completes to start another!
- The simplest way to start a background process is to add an ampersand (&) at the end of the command.
- $ls ch*.doc &

Listing Running Processes

- It is easy to see your own processes by running the ps (process status) command.
- One of the most commonly used flags for ps is the -f (f for full) option, which provides more information as shown in the following example

Here is the description of all the fields displayed by ps -f command

Sr.No.	Column & Description
1	**UID** User ID that this process belongs to (the person running it)
2	**PID** Process ID
3	**PPID** Parent process ID (the ID of the process that started it)
4	**C** CPU utilization of process
5	**STIME** Process start time
6	**TTY** Terminal type associated with the process
7	**TIME** CPU time taken by the process
8	**CMD** The command that started this process

Stopping Processes

- Ending a process can be done in several different ways. Often, from a console-based command, sending a CTRL + C keystroke (the default interrupt character) will exit the command. This works when the process is running in the foreground mode.

- If a process is running in the background, you should get its Job ID using the ps command. After that, you can use the kill command to kill the process as follows –

Locating the processes

- To kill a process, we have to access the process information. There are various commands to track a process such as top, ps, pgrep, and pidof.

S.NO	CLI	GUI
1.	CLI is difficult to use.	Whereas it is easy to use.
2.	It consumes low memory.	While consumes more memory.
3.	CLI do not use any pointing devices.	While it uses pointing devices for selecting and choosing items.
4.	CLI is faster than GUI.	The speed of GUI is slower than CLI.
5.	CLI operating system needs only keyboard.	While GUI operating system need both mouse and keyboard.
6.	CLI's appearance can not be modified or changed.	While it's appearance can be modified or changed.
7.	In CLI, input is entered only at command prompt.	While in GUI, input can be entered anywhere on the screen.
8.	In CLI, the information is shown or presented to the user in plain text and files.	While in GUI, the information is shown or presented to the user in any form such as: plain text, videos, images, etc.
9.	In CLI, there are no menus provided.	While in GUI, menus are provided.
10.	There are no graphics in CLI.	While in GUI, graphics are used.
11.	CLI do not use any pointing devices.	While it uses pointing devices for selecting and choosing items.

Google Classroom code for “Linux Fundamentals & Networking in Linux”

- Join the Google Classroom by Using following Code:

yrtitzh

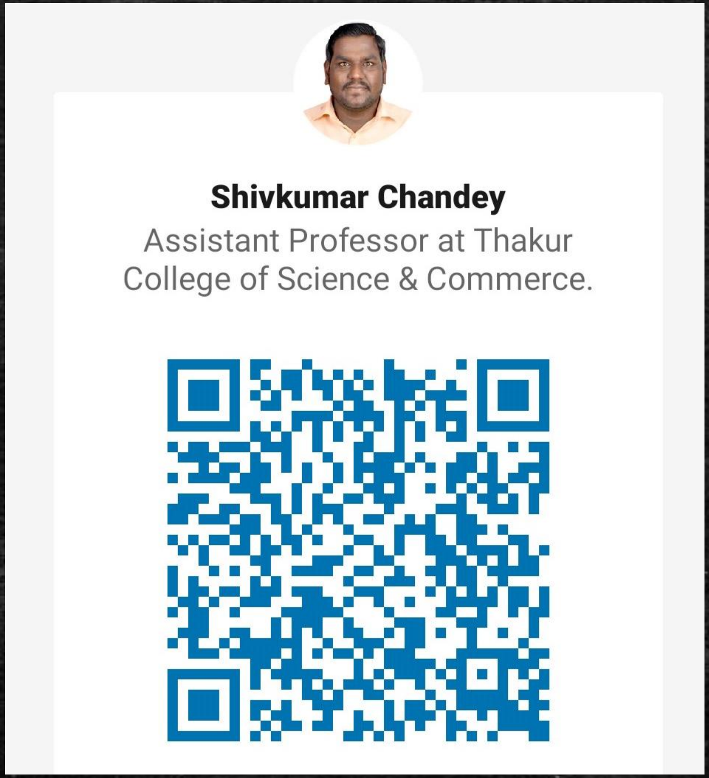
Thank You!!!Any Query?

asktoshivsir@gmail.com

Shivkumar Chandey:

+91 9987389441

Scan QR Code to connect on LinkedIn



Shivkumar Chandey
Assistant Professor at Thakur College of Science & Commerce.