# 20MCA136–NETWORKING & SYSTEM ADMINISTRATION LAB

*Submitted in partial fulfilment of the requirements for the award of*

*Masters of Computer Applications*

*at*

## COLLEGE OF ENGINEERING POONJAR

**Managed by I.H.R.D., A Govt. of Kerala undertaking**

*(Affiliated to APJ Abdul Kalam Technological University)*



## SUBMITTED BY
## SUNITHA MOL SASI(PJR24MCA-2010)

## DEPARTMENT OF COMPUTER APPLICATION
## COLLEGE OF ENGINEERING POONJAR

# COLLEGE OF ENGINEERING POONJAR

**Managed by I.H.R.D., A Govt. of Kerala undertaking**
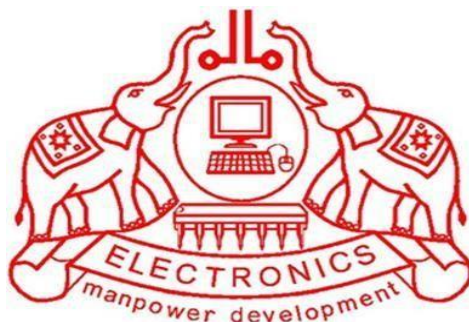
*(Affiliated to APJ Abdul Kalam Technological University)*



## CERTIFICATE

Certified that this is a Bonafide record of practical work done in Networking & System Administration Lab(20MCA136) Laboratory by SUNITHAMOL SASI, Reg No.**PJR24MCA2021**of College of Engineering, Poonjar during the academic year 2024-2026.


Dr. Annie Julie Joseph                                              Anjana Shekhar

Head of the Department                                           Assistant Professor,CSE


Submitted to the University Examination held on:


INTERNAL EXAMINER                                    EXTERNAL EXAMINER

# INDEX

# EXPERIMENT: 01

## AIM:

Introduction to Computer Hardware. This experiment helps in understanding different components of a computer system, their roles and how they interact with each other.

## 1. Physical Identification of Major Components

### a) Motherboard

It acts as the backbone of the computer, connecting all components.

Common motherboard sizes:

• ATX - Standard size for desktops.

Micro-ATX - Smaller version of ATX with fewer expansion slots.

Mini-ITX - Compact form factor used for small PCs.

• Important components:

• Chipset - Manages communication between CPAM, and peripherals.

• VRM (Voltage Regulator Module)

Supplies the right voltage to the CPU and RAM.

### b) RAM (Random Access Memory) Modules

• Temporary storage that holds data for active applications.

• Types of RAM:

SRAM (Static RAM) - Used for cache memory, faster but expensive.

DRAM (Dyna RAM) - Used in main memory, requires constant refreshing.

• Common  RAM Generations:

• DDR3 – 800 to 2133 MHz speed.

  DDR4 – 2133 to 3200 MHz, better power efficiency.

  DDR5 – 3200+ MHz, improved bandwidth.

  ECC (Error – Correcting Code) RAM – Used in servers to detect and correct memory errors.

## c) Daughter Cards (Expansion Cards)

• Additional circuit boards plugged into motherboard slots to enhance functionalities.

• Types:

• Graphics Card (GPU) - Handles video rendering; common brands include NVIDIA and AMD.

Sound Card - Provides high-quality audio processing.

Network Interface Card (NIC) Allows wired wireless networking.

• Capture Card - Used for recording and streaming video content.

## d) Bus Slots (Expansion Slots)

Slots on the motherboard used to install expansion cards.

• Types of Expansion Slots:

• PCIe (Peripheral Component Interconnect Express) - Used for high-speed components like GPUs and SSDs.

• PCI (Older Standard) - Used for sound and network cards.

• AGP (Accelerated Graphics Port, Obsolete) - Used for older graphics cards.

• M.2 Slot - Used for high-speed SSDs and Wi-Fi cards.

### e) SMPS (Switched-Mode Power Supply)

• Converts AC power from the wall socket to DC power.

Power Supply Ratings:

Wattage - Determines the total power output (e.g., 500W, 750W, 1000W).

Efficiency Rating (80 PLUS Certification) - Indicates power efficiency (Bronze, Silver, Gold, Platinum)

• Types:

• ATX Power Supply - Standard for desktops.

• SFX Power Supply - Compact for small form factor PCs.

### f) Internal Storage Devices

• HDD (Hard Disk Drive) • Uses spinning platters to store data.

• Slower but cheaper per GB.

• Common speeds: 5400 RPM, 7200 RPM, 10,000 RPM.

• SSD (Solid-State Drive)

• No moving parts, much faster than HDDs

• Types:

SATA SSD - Uses SATA interface, limited to 600MB/s.

NVMe SSD - Uses PCle interface, speeds up to 7000MB/s.

• RAID (Redundant Array of Independent Disks)

• RAID 0: Performance boost, no redundancy.

• RAID 1: Mirroring for data safety.

• RAID 5/10: Combination of performance and redundancy.

**g) Interfacing Ports**

USB Ports:

• USB 2.0-480 Mbps

  USB 3.05 Gbps  USB

  3.1/3.2- 10-20 Gbps

  USB 4 Up to 40 Gbps

• Display Ports:

• VGA - Analog video signal.

• DVI - Digital video signal.

• HDMI - Supports high-definition video and audio.

• DisplayPort - Higher bandwidth for gaming and professional use.

• Other Ports:

• Ethernet (RJ-45) - Used for wired networking.

• Thunderbolt - High-speed data and display connection.

## 2. Specifications of Desktop and Server-Class Computers

**Desktop Computer Specifications**

- Used for general tasks like web browsing, office work, and gaming.

- Example Configuration:

- Processor: Intel Core i5-13600K /AMD Ryzen 5 7600X

- RAM: 16GB DDR5 5600 MHz

- Storage: 1TB NVMe SSD

- Graphics: NVIDIA RTX 4060 Ti / AMD RX 7700 XT

- Power Supply: 600W 80+ Gold  **Server-Class Computer Specifications**

- Designed for high-performance tasks like database management, cloud hosting, and Al computing.

- Example Configuration:

- Processor: Intel Xeon Gold 6226R /AMD EPYC 7313

•RAM: 128GB DDR5 ECC

- Storage: 2TB NVMe SSD + 4TB HDD (RAID 1)  • Network: Dual 10Gbps Ethernet ports

- OS: Windows Server 2022 / Ubuntu Server 22.04

## 3. Installation of Common Operating Systems

**Windows Installation (For Desktop & Server)**

1. Boot from USB/DVD - Press F12/F2/DEL to enter boot menu.

2. Choose Language & Keyboard Layout.

3. Partition the Disk:

• Select "Custom Installation."

• Create partitions if necessary.

4. Install OS Files - Wait for installation to complete.

5. Create User Account and Set Up Security Settings.

6. Install Drivers & Updates.

**Linux Installation (Ubuntu Server/Desktop)**

1. Boot from USB/DVD.

2.Select Install Ubuntu Option.

3.Choose Keyboard Layout & Language.

4.Disk Partitioning:

• Choose "Erase Disk" for automatic partitioning.

• Manual option: Create root swap, and home partitions.

5. Set Up User and Password.

6. Install Packages and Configure Network.

7. Complete Installation and Reboot.

## RESULT

Familiarized with Hardware components of a computer system, their roles and how they interact with each other.

## EXPERIMENT-02

**AIM:**

Steps for installing windows and linux operating System

## Steps for windows os installation

There are two main methods to install Windows 10:

**Before you start:**

- Make sure you have a stable internet connection.

- Get a blank USB drive with at least 8GB of storage space. You'll need to erase everything on the drive, so use a blank one if you can.

This will erase everything and install a clean version of Windows 10. Follow these steps:

**Step 1:** Download and Open Windows 10 installation media from Microsoft's website.

**Step 2:** Accept license terms by choosing Accept.

**Step 3:** Choose Create installation media and click Next.

**Step 4:** Click Next.

**Step 5:** Choose USB flash drive and click Next to proceed.

**Step 6:** Plug your USB Device into the PC

**Step 7:** Select the drive to install Windows and click Next. This will erase and reformat the drive.

**Step 8:** Windows 10 is now being downloaded into your USB device. **Step 9:** Plug your USB Device (as a bootable device now) with Windows 10 installer into a new PC.

**Step 10**: Turn on your new computer and access the BIOS/UEFI (typically using F2, F10, or Del).

**Step 11**: In BIOS/UEFI, set the USB flash drive as the first boot option.

**Step 12:** Save changes and exit. Your PC will reboot from the USB drive.

**Step 13:** When the Windows Setup appears, select language, time/currency format, etc. then choose Next.

**Step 14:** After that, Click Install Now and follow the prompts to set up Windows.

## Linux OS installation

Linux is an operating system, similar to Windows, but with many different versions due to the nature of being open source and fully customizable. To install Linux, you must choose an install method and choose a Linux distribution.

To install Linux:

- Choose an install method: Windows Subsystem for Linux (WSL), Bare metal Linux; or create a Virtual Machine (VM) to run Linux locally or in the cloud.

- Choose a Linux distribution: Ubuntu, Debian, Kali Linux, openSUSE, etc.

- Follow the steps for your preferred install method:

- Use the install Linux command with Windows Subsystem for Linux (WSL)

- Create a Linux Virtual Machine (VM) in the cloud

- Create a Linux Virtual Machine (VM) on your local machine

- Create a bootable USB to install bare-metal Linux

After installing Linux: Get familiar with your distribution's package manager, update and upgrade the packages available, and get familiar with the other Linux resources at Microsoft, such as training courses, Linux-versions of popular tools, news, and Open Source events.

## RESULT

Familiarized with windows and Linux installation.

# EXPERIMENT- 03

**AIM**

To install and use VirtualBox to run multiple operating systems on one computer.

## PROCEDURE:

In the world of technology virtualization plays a crucial role in allowing users to run multiple operating systems on a single computer. Oracle VM VirtualBox is one of the most popular and free open-source virtualization software that enables users to create and manage virtual machines. Whether for software testing, running multiple operating systems or experimenting with new technologies VirtualBox provides an efficient and user-friendly platform.

**Step 1: Downloading VirtualBox**

The first step in installing VirtualBox is to download the software from its official website [www.virtualbox.org.](www.virtualbox.org) Oracle provides versions compatible with various operating systems, including
Windows, macOS, and Linux. It is important to download the correct version that matches the user's system. Additionally, users can also download the VirtualBox Extension Pack which enhances the functionality of the software by providing support for USB devices, remote desktops, and other advanced features.

**Step 2: Installing VirtualBox**

Once the setup file is downloaded, the installation process begins.

For Windows users, running the .exe file launches the installer, where users must follow the on-screen instructions, select installation options, and grant administrator permissions.

On macOS, users must open the .dmg file, drag VirtualBox to the Applications folder and allow permissions in the Security & Privacy settings if needed.

Linux users can install VirtualBox using terminal commands such as: sudo

apt update sudo apt install virtualbox

After installation, users can launch VirtualBox from the Start menu (Windows), Applications folder (Mac), or terminal (Linux).

**Step 3: Configuring VirtualBox**

After installation, users can create a Virtual Machine (VM) to run a guest operating system. This process involves:

1.     Clicking "New" to create a virtual machine.

2.     Selecting the operating system (Windows, Linux, macOS, etc.).

3.     Allocating RAM (recommended: at least 2GB for Windows, 4GB+ for better performance).

4.     Creating a virtual hard disk, which determines storage space for the VM.

Additionally, installing the VirtualBox Extension Pack improves the user experience by enabling USB support, clipboard sharing, and other advanced features.

**Step 4: Install the Operating System**

For Windows

1. Select the newly created Windows VM and click "Settings".

2. Go to Storage -> Click Empty under "Controller: IDE".

3. Click the CD icon -> Choose a disk file -> Select the Windows ISO.

4. Click OK and then Start the VM.

5. Follow the Windows installation steps:

•   Click Install Now.

•   Enter a product key (or click "I don't have a product key").

•   Select Windows edition and installation type.

•   Create a partition and install Windows.

6. Wait for installation to complete, then follow on-screen setup steps

For Linux

1. Select the newly created Linux VM and click "Settings".

2.  Go to Storage ->Click Empty under "Controller: IDE".

3.  Click the CD icon -> Choose a disk file ->Select the Linux ISO.

4.  Click OK and then Start the VM.

5.  Follow the Linux installation steps:

Select language and keyboard layout.

-   Click Install.

-   Choose installation type (default settings work fine).

-   Create a username and password.

6.  Wait for installation to complete and restart the VM.

## RESULT

Familiarized with virtual machines and installation of Linux and Windows operating systems on virtual machine.

## EXPERIMENT: 04

### AIM:

Familiarization of basic Linux commands

### COMMANDS:

**1.man**

The man command displays a user manual for any commands or utilities available in the terminal, including their name, description, and options.

**Syntax:** man <command name>

**2. ls**

The ls command is commonly used to identify the files and directories in the working directory.

**Syntax:** ls[options][file/directory]

**3. echo**

Print string or text to the terminal

**Syntax:** echo <Text to print on terminal>

**4. read**

The read command in Linux is a built-in command used to read a line of input from the standard input (stdin) or a file descriptor and store it in a variable. This command is commonly used in shell scripts to capture user input or handle file operations

**Syntax:** read [options] variable_name

**5. more**

A        command line utility that allows users to view text files on the page at a time.It is often used when dealing with large files.

**Syntax:** more[file name]

**6. less**

The less command shows a file's content one screen at a time.

**Syntax:** less[file name]

7. **cat**

   Appends a file's contents to another file.

   **Syntax:** cat[source file]>>[destination file]

8. **cd**

   Used to change the directory.

   **Syntax:** cd[options][directory]

9. **mkdir**

   Creates a new directory.'mkdir' stands for make directory.

   **Syntax:** mkdir[options] directory name

10. **pwd**

    Shows the current working directory.

    **Syntax:** pwd[options]

11. **find**

    Used to search for files and directories within a specified path. It allows users to locate files by name,type,size,permissions etc.

    **Syntax:** find[options][path][expression]

12. **mv**

    Move,rename files or directories

    **Syntax:** mv[options][source file] [destination file] //Rename

file          mv[options] [source directory] [destination

directory]//Move a directory

13. **cp**

    Used to copy files and directories.

**Syntax:** cp[option] [source] [destination]

## 14. rm

Removing files or directories.

**Syntax:** rm[options] [file name]

## 15. tar    tar stands for 'tape archive' and used for archiving and compressing files.

**Syntax:** tar[options] [archive file] [file/directory to be archived]

## 16. wc    wc command is used to count the no.of characters,words,lines and bytes in a file.

**Syntax:** wc[options] [location file]

## 17. cut

Used to extract specific sections of a file.The command cuts parts of a line based on fields,delimeters,byte,position and character position.

**Syntax:** cut[options] file

## 18. paste

Used to merge lines of files horizontally rather than vertically.

**Syntax:** paste[options] file

## 19. head

Shows the first ten lines of a file.

**Syntax:** head[options] file

## 20. tail

Shows the last ten lines of a file.

**Syntax:** tail[options]

file

## 21. grep

Command searches for specific text patterns in files.It is short for 'global regular expression print'.

**Syntax:** grep[options] patterns[file]

## 22. expr

A          command-line utility that evaluates expression and outputs the result.

**Syntax:** $expr expression

## 23. chmod

Modifies the read,write and execute permissions of specified files and the search permissions of specified directories.

**Syntax:** chmod[options] [mode] [file name]

## 24. chown

Changes the owner and group of a file,directory or symbolic links.

**Syntax:** chown[options] user [:group] file

## 25. redirections and piping

In Linux, redirection controls where a command's input and output go, while piping passes the output of one command to another. Both are useful for automating workflow.Use redirection operators like >, <, >>, and &> to direct output to a file.Use > to direct error messages from a command to a file .Use &> to direct both standard output and error to a file.Use the pipe character (|) to pass the output of one command to the input of another.

**26. useradd**          useradd command adds a new user to the directory. The user is given the ID

automatically depending on which category it falls in.

## 27. usermod

This command can change the user ID of a user. The user with the given username will be assigned with the new ID given in the command and the old ID will be removed.

## 28. userdel

Command deletes the user whose username is provided. Make sure that the user is not

part of a group. **29. passwd**

Assign a password to a user.

**Syntax:** passwd [options] [username]

## 30. df

Check the details of the file system

**Syntax:** df [options] [file]

## 31. top

The top command in Linux is a powerful utility that provides a real-time, dynamic view of the system's running processes and resource usage.

**Syntax:** top [options]

## 32. ps          used to check the active processes in the terminal

**Syntax:** ps [options]

## 33. ssh

The ssh command is used to securely log into a remote machine and execute commands on that machine.

**Syntax:** ssh user@host ,where user is the username on the remote machine and host is the address or hostname of the remote machine.

## 34. scp

The SCP (Secure Copy Protocol) command is a powerful tool in Unix systems used to securely copy files and directories between hosts on a network.

**Syntax:** scp [options] [user@source_host:path/to/source/file] [user@target_host:target/path]

## 35. ssh-keygen

A          command-line utility used for generating, configuring, and managing SSH keys. These keys are primarily used for secure shell protocol (SSH) connections, providing a means of encrypting communications while allowing for password-less logins between systems.

## 36. ssh-copyid

A          convenient utility for copying a public SSH key to a remote machine, enabling passwordless authentication.

**Syntax:** ssh-copy-id [options] user@host-ip   user is the username on the remote host, and host-ip is the IP address of the remote server[1].

**RESULT**

Familiarized with basic linux commands

# EXPERIMENT NO:05

**AIM:**

Shell scripting: study bash syntax, environment variables, variables, control constructs such as if, for and while, aliases and functions, accessing command line arguments passed to shell scripts.

**Program No: 1**
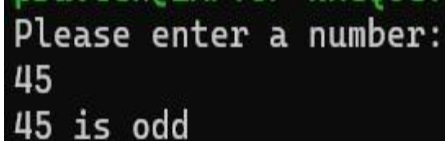
Shell program to find if a number is odd or even.

**Program:** check_odd_even()

```
{      if [
$((number % 2)) -
eq 0 ];      then
echo "$number is even"
    else
echo
"$number is odd"
fi      }
    echo   "Please   enter   a
number: "          read  number
check_odd_even "$number"
```

**Output:**

```
Please enter a number:
45
45 is odd
```

**Program No: 2**

Write a shell program to find the average exam marks and generate a grade according to the average mark.

**Program:**

```
#!/bin/bash    echo

"Enter the marks of the

subjects"    read -p

"Maths: " m1    read -p

"English: " m2    read -p

"Java: " m3    read -p

"Python: " m4    read -p

"PHP: " m5    t=$((m1 + m2

+ m3 + m4 + m5))

avg=$((t / 5))    if

((avg >= 90)); then

grade="S"    elif ((avg

>= 80)); then

grade="A"

elif ((avg >= 70)); then

grade="B"

 elif ((avg >= 60)); then

grade="C"

elif ((avg >= 50)); then

grade="D"

elif ((avg >= 40)); then

grade="E"

else
```
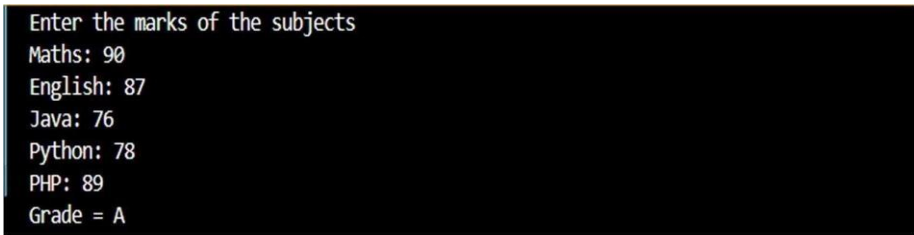
grade="Failed"

Failed

fi

echo "Grade = $grade"  **Output:**

```
Enter the marks of the subjects
Maths: 90
English: 87
Java: 76
Python: 78
PHP: 89
Grade = A
```

**Program No: 3**

Write a shell program to find Fibonacci series of any number.

 **Program:**  #!/bin/bash  read -p

"Enter the limit: " n a=0 b=1  echo

"Fibonacci

Series    is:"

for((i=0;i<n

;i++ )) do
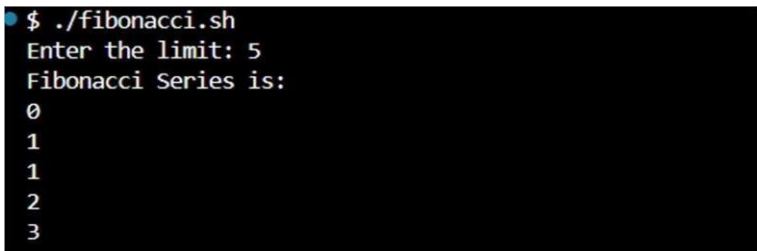
echo        $a

f=$((a + b))

a=$b    b=$f

done

**Output:**

```
$ ./fibonacci.sh
Enter the limit: 5
Fibonacci Series is:
0
1
1
2
3
```

**Program No: 4**

Write a Shell program to check the given number and its reverse are same.

**Program:**  echo "Enter

a number:"

 read num

reverse=$(echo

"$num"

| rev) if [ "$num" -eq

"$reverse" ];

then

   echo "$num is same

when reversed."
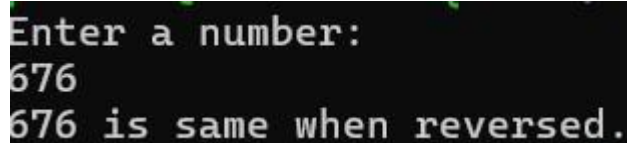
 else

echo "$num is not same

when reversed."

 fi

**Output:**
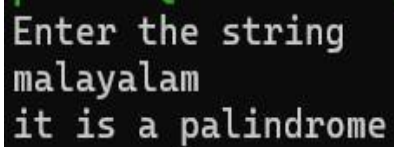
```
Enter a number:
676
676 is same when reversed.
```

**Program No: 5**

Write a shell program to check whether a string is palindrome or not   **Program:**   echo Enter the string

 read s

echo $s>temp

rvs="$(rev temp)"

if [ $s = $rvs ]

 then

   echo "it is a

palindrome"

else   echo " it is

not a

Palindrome"

fi

**Output:**



**Program No: 6**

Write a shell program to find sum of digits using function.

**Program:**
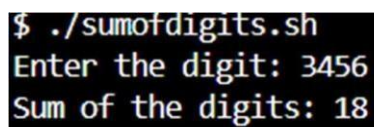
   read -p "Enter

the digit: " num

sum() {

n=$num        s=0

while (( n > 0 ));

do

```
   digit=$((n % 10))
```

```
s=$((s + digit))
```

```
n=$((n / 10))
```

```
done
```

```
echo $s
```

```
   }
```

```
   result=$(sum
```

```
"$num")     echo
```

```
"Sum of the digits:
```

```
$result"
```

**Output:**



```
$ ./sumofdigits.sh
Enter the digit: 3456
Sum of the digits: 18
```

### Program No: 7

Write a shell to check whether a number is an Armstrong number or not.

**Program:**

```
#!/bin/bash
```

```
read -p "Enter a
```

```
number: " num
```

```
original=$num
```

```
sum=0
```

```
n=${#num}
```

```
while [ $num -gt 0 ];
```

```
do
```
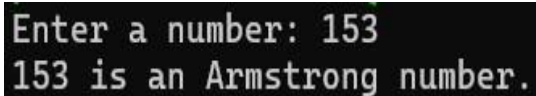
```
  digit=$((num % 10))

  power=$((digit ** n))

   sum=$((sum + power))

     snum=$((num / 10))

done

 if [ $sum -eq $original ];

then

echo "$original is an Armstrong number."

else

 echo "$original is not anArmstrongnumber."

 fi
```

**Output:**

```
Enter a number: 153
153 is an Armstrong number.
```

**Program No: 8**

Write a Shell program to find the sum of odd and even numbers from a set of numbers

**Program:**

```
#!/bin/bash

 echo -n "Enter a set of numbers

separated by spaces: "

 read -a numbers

sum_even=0 sum_odd=0

for num in

"${numbers[@]}";

do
```

```
if ((num % 2 =0));

then

((sum_even +=num))

 else

 ((sum_odd += num))

fi

done  echo "Sum of even

numbers is:$sum_even"

 echo "Sum of odd numbersis:

$sum_odd"
```

**Output:**

```
Enter a set of numbers separated by spaces: 1 2 3 4 5 6 7 8 9
Sum of even numbers is: 20
Sum of odd numbers is: 25
```

**Program No: 9**

Write a menu driven shell script that lists current directory, prints working directory, displays date and displays users logged in.

**Program:**

```
!#/bin/bash

echo "1. List  current  directory  contents"

echo "2. Print working directory"

   echo "3. Display current date and time"

echo "4. Display users currently logged in"

echo "5. Exit"
```

```
 read -p "Enter your choice [15]: " choice
case $choice in
    1)
        echo "Listing contents of current directory:" ls;;
    2)
        echo "Current workingdirectory:"
pwd ;;

3) echo"Current date and time:"
date  ;;


 4) echo "Users currentlylogged
in:"  who

   ;;
  5)


echo"Exiting...Goodbye!
"   break
    s  ;;
      *)
          echo "Invalid choice, please select from 1 to 5."
          ;;esac

  echo ""

 read -p "Press Enter to continue..."
```

```
 clear


   done
```

**Output:**

```
$ ./work.sh
1. List current directory contents
2. Print working directory
3. Display current date and time
4. Display users currently logged in
5. Exit
Enter your choice [1-5]: █
```

**Program No: 10**

Write a shell program to check executable rights for all files in the current directory, if file does not have the execute permission then make it executable.

**Program:**

```
  for file in *; do

 if [ -f "$file" ]; then

    if [ ! -x "$file" ];

then

   echo"Adding execute permissionto:

$file"

 chmod +x "$file"

   else

  echo

"Already Executable: $file"

 fi

fi

done
```

**Output:**

```
Made 10.sh executable
Made 11.sh executable
Made 12.sh executable
Made 13.sh executable
Made 14.sh executable
```

## Program No :11

Write a Shell program to find the area and circumference of a circle.

### Program:

```
echo "Enter the radius:"
  read r
area =`echo 3.14 \* $r \* $r| bc`
cir=`echo 2 \* 3.14 \*
$r| bc `
echo "Area : $area"
echo "Circumference :
$cir"
```

### Output:

```
Enter the radius:
34
Area : 3629.84
Circumference : 213.52
```

## Program No :12

Write a shell program to find the roots of a quadratic equation.

**Program:**

```bash
#!/bin/bash

echo "Enter the coefficients of the quadratic

equation (a, b, c): " read a b c discriminant=$((b *

b - 4 * a * c))

if [ $discriminant -lt 0 ]; then

  real_part=$(echo "scale=2; $b / (2 * $a)" | bc)
imaginary_part=$(echo "scale=2; sqrt(1 * $discriminant) / (2

* $a)" | bc)

  echo "The quadratic equation has imaginary roots."

  echo "Root 1: $real_part + ${imaginary_part}i"

 echo "Root 2: $real_part - ${imaginary_part}i"

else

   root1=$(echo "scale=2; (-$b + sqrt($discriminant)) / (2 *

$a)" | bc)

  root2=$(echo "scale=2; (-$b - sqrt($discriminant)) / (2 *

$a)" | bc)

 echo "The roots of the quadratic equation are real."    echo
"Root

1: $root1"    echo "Root 2: $root2"

 fi
```

**Output:**

```
Enter the coefficients of the quadratic equation (a, b, c):
1 2 5
The quadratic equation has imaginary roots.
Root 1: -1.00 + 2.00i
Root 2: -1.00 - 2.00i
```

```
Enter the coefficients of the quadratic equation (a, b, c):
2 5 -6
The roots of the quadratic equation are real.
Root 1: .88
Root 2: -3.38
```

**Program No:13**

Write a Shell program to check the given integer is prime or not.

**Program:**

" echo "Enter an integer number: "

 read number

flag=1

for ((i = 2; i

<=number / 2;

i++));

 do

if [ $((number i))eq

];

then

flag=0

```
break

 fi

done if [ $number -eq 1 ];

 then

 echo "1 is neither prime nor

composite."

 elif [ $flag -eq 1 ];

then

echo "$number is a prime

number."

 else

 echo "$number is not a

prime number."

 fi
```

**Output:**

```
Enter an integer:
13
13 is a prime number.
```

**Program No:14**

Write a shell program to generate prime numbers between 1 and 50.

**Program:**

```
#!/bin/bash

echo "Prime numbers between 1and

50 are:"
```

```
for ((number = 2; number <= 50;

number++)); do

        flag=1

for ((i = 2; i <= number / 2; i++)); do

 if

[ $((number % i)) -eq 0 ]; then

 flag=0

 break

     fi

   done

  if [ $flag -eq 1 ]; then

echo $number      fi

done
```
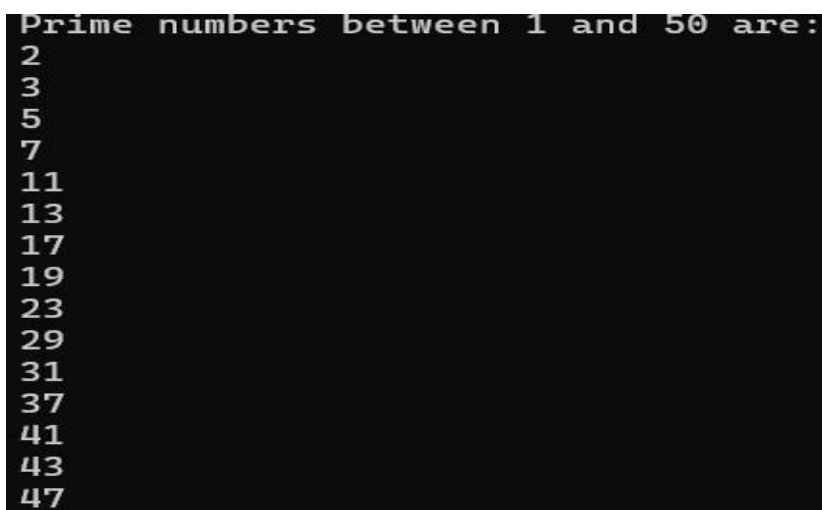
**Output:**

```
Prime numbers between 1 and 50 are:
2
3
5
7
11
13
17
19
23
29
31
37
41
43
47
```

**Program No:15**

Write a Shell program to find the sum of all numbers between 50 and 100, which are divisible by 3 and not divisible by 5.

**Program:**

```
sum=0
for ((i = 50; i <= 100; i++)); do        if ((i %

3 == 0)) && ((i % 5 != 0));

then

sum=$((sum + i))

  fi

done  echo "Sum of numbers between 50 and 100, which are divisible by 3 and not

divisible by 5: $sum"
```

**Output:**



```
Sum of numbers between 50 and 100, which are divisible by 3 and not divisible by 5:
1050
```

**Program No:16**

Write a Shell program to find the sum of square of individual digits of a number.

**Program:**

```
echo "Enter a  number: "

Read num

 sum=0  while [ $number -ne 0 ]; do

digit=$((number % 10))

sum=$((sum + digit * digit))

number=$((number / 10)) done echo
```

"The sum of the squares of the

digits is $sum."

**Output:**



```
Enter a number:
56
The sum of the squares of the digits is 61.
```

**Program No:17**

Write a Shell program to print the reverse of a number using function.

**Program:**

```
function reverse_number

{ local number=$1 local

reverse=0

while ((number

> 0)); do reverse=$((reverse

* 10 + number % 10))

number=$((number / 10))

done echo "$reverse"

}

 echo "Enter a

number:"

read number

result=$(reverse_nu
```
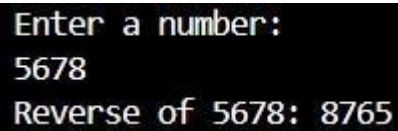
mber $number) echo

"Reverse of

$number: $result"

**Output:**

```
Enter a number:
5678
Reverse of 5678: 8765
```

**Program No:18**

Write a shell program to count the number of vowels in a line of text.

**Program:**

echo "Enter a line of text:"

read line

vowel_co

unt=0  for ((i = 0; i < ${#line};

i++)); do


 char=${line:i:1}

        case $char in


 [aeiouAEIOU])

vowel_count=$((vowel_count + 1))

;;      esac done echo "The number of vowels in

the line of text is:

$vowel_count"

**Output:**

```
Enter a line of text:
hello welcome to Bash
The number of vowels in the line of text is: 7
```

**Program No:19**

Write a Shell program to find the factorial of a number using for loop.

**Program:**

echo "Enter a decimal number: "

 read number

factorial =1

 for ((i = 1; i <= number; i++)); do

factorial=$((factor ial * i))

done

 echo "Factorial of $number: $factorial"

**Output:**

```
Enter a number:
5
Factorial of 5: 120
```

### Program No:20

Write a shell program to find smallest and largest number from a set of numbers.

**Program:**

```
 NUMBERS=(5 3 8 1 9 4 7 2)

smallest=${NUMBERS[

0]}

largest=${NUMBERS[0

]}

 for number in

"${NUMBERS[@]}";

do

  if ((number < smallest)); then

smallest=$number

   fi

   if ((number > largest)); then

largest=$number

   fi

done  echo "Smallest

number: $smallest

" echo "Largest number:

$largest"
```

**Output:**

```
Smallest number: 1
Largest number: 9
```

**Program No:21**

Write a shell program to find the smallest digit from a number.

**Program**:

```
echo "Enter a decimal number: "

read number

smallest=${number:0:

1}

 for ((i = 1; i < ${#number};

i++));

do

digit=${number:i:1}

  if ((digit < smallest));

then

smallest=$digit

   fi

done

 echo "Smallest digit:

$smallest"
```

**Output:**

```
Enter a number:
6589
Smallest digit: 5
```

**Program No:22**

Write a Decimal to Binary Conversion Shell Script
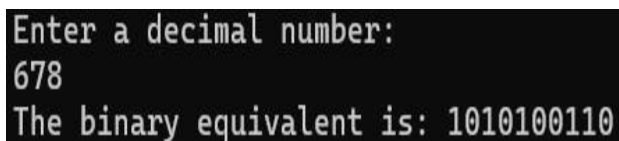
```
#!/bin/bash

echo "Enter a decimal number: "

 read decimal binary=""

 while [ $decimal -gt 0 ]; do

remainder=$((decimal % 2))

binary="$remainder$binary"

decimal=$((decimal / 2)) done

 echo "The binary equivalent is:

$binary"
```

**Output:**

```
Enter a decimal number:
678
The binary equivalent is: 1010100110
```

**Program No:23**

Write a shell script to validate password strength. Here are a few assumptions for the password string.: Length – minimum of 8 characters and contain both alphabet and number.

**Program:**

```
 #!/bin/bash

 read -p "Enter your password: "

password if [[ ${#password} -lt 8

]]; then

   echo "Password length must be at

least 8 characters."
```

```
exit 1 fi  if ! [[ "$password" =~ [A-Za-z]+[0-

9]+ ]]; then     echo "Password must contain

both alphabet and number."

 exit 1

 fi

if ! [[ "$password" =~ [a-z]+ ]] || ! [[ "$password" =~

[A-Z]+ ]]; then

echo "Password must include both small and capital

case letters."

 exit 1

fi

 echo "Password is valid."
```
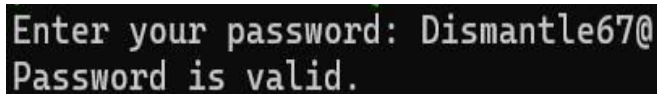
**Output:**



```
Enter your password: Dismantle67@
Password is valid.
```

**Program No: 24**

Write a shell script to print the count of files and subdirectories in the specified

**Program:**

```bash
#!/bin/bash

echo -n "Enter directory path: "

 read directory

 num_files=$(find "$directory" -type f 2>/dev/null | wc -l)

 num_directories=$(find "$directory" -type d 2>/dev/null | wc -l)
```
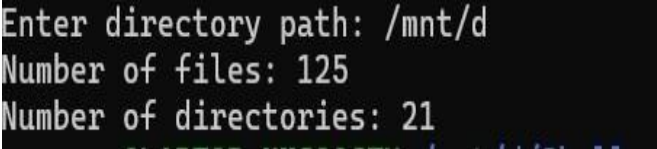
```
echo "Number of files: $num_files"

s echo "Number of directories: $num_directories"
```

**Output:**

```
Enter directory path: /mnt/d
Number of files: 125
Number of directories: 21
```

**RESULT:**

Familiarized with Shell Script

# EXPERIMENT:06

### AIM:

Introduction to command line tools for networking IPv4 networking, network commands: ping,traceroute, nslookup,ifconfig.

## 1. Ping

The command used to check the availability of a host. The response shows the URL you are pinging, the ip address associated with the URL and the size of packets being sent on the first line . The next four lines shows the replies from each individual packets including the time(in milliseconds) for the response and the time to live(TLL) of the packet, that is the amount of time that must pass before the packet discarded.  Syntax of Ping Command in Linux:  ping [options] host_or_IP_address

```
Windows PowerShell
Copyright (C) Microsoft Corporation. All rights reserved.

Install the latest PowerShell for new features and improvements! https://aka.ms/PSWindows

PS C:\Users\User Pc 1\Desktop> ping

Usage: ping [-t] [-a] [-n count] [-l size] [-f] [-i TTL] [-v TOS]
            [-r count] [-s count] [[-j host-list] | [-k host-list]]
            [-w timeout] [-R] [-S srcaddr] [-c compartment] [-p]
            [-4] [-6] target_name

Options:
    -t             Ping the specified host until stopped.
                   To see statistics and continue - type Control-Break;
                   To stop - type Control-C.
    -a             Resolve addresses to hostnames.
    -n count       Number of echo requests to send.
    -l size        Send buffer size.
    -f             Set Don't Fragment flag in packet (IPv4-only).
    -i TTL         Time To Live.
    -v TOS         Type Of Service (IPv4-only. This setting has been deprecated
                   and has no effect on the type of service field in the IP
                   Header).
    -r count       Record route for count hops (IPv4-only).
    -s count       Timestamp for count hops (IPv4-only).
    -j host-list   Loose source route along host-list (IPv4-only).
    -k host-list   Strict source route along host-list (IPv4-only).
    -w timeout     Timeout in milliseconds to wait for each reply.
    -R             Use routing header to test reverse route also (IPv6-only).
                   Per RFC 5095 the use of this routing header has been
                   deprecated. Some systems may drop echo requests if
                   this header is used.
    -S srcaddr     Source address to use.
    -c compartment Routing compartment identifier.
    -p             Ping a Hyper-V Network Virtualization provider address.
    -4             Force using IPv4.
```

## 2. Traceroute

traceroute is a command-line utility in Linux and other Unix-like operating systems that

allows you to track the path that packets take from your computer to a destination host on a network. It's used for troubleshooting network connectivity issues and identifying network delays.  Syntax of Traceroute command in Linux:   traceroute [options] destination



## 3. ifconfig

This commands in windows allows you to see a summarized information of your network such as ip address, subnet mask , server address etc.The command is a part of **net- tools**, a legacy Linux tool for configuring a network interface. Modern distributions use the IP command, which works in a similar manner.Even though  has limited capabilities compared to IP, the command is still commonly used to configure a network interface in Linux.

**Syntax of ifconfigCommand in**

**Linux:**  ifconfig [interface]

[options]

linux ifconfig command

## 4.Nslookup

Nslookup (stands for "Name Server Lookup") is a useful command for getting information from the DNS server. It is a network administration tool for querying the Domain Name System (DNS) to obtain domain name or IP address mapping or any other specific DNS record. It is also used to troubleshoot DNS-related problems.

Syntax of the nslookup Command in Linux:

nslookup [option] [domain]



**RESULT:**

Familiarized with Hardware components of a computer system.

# EXPERIMENT NO:07

**AIM:**

Analyzing  wireshark packet stream using whileshark.

## PROCEDURE

### What is Wireshark?

 Wireshark is a network packet analyzer. A network packet analyzer presents captured packet data in as much detail as possible.

You could think of a network packet analyzer as a measuring device for examining what's happening inside a network cable.

Wireshark is available for free, is open source, and is one of the best packet analyzers available today.

**Installing Wireshark on Ubuntu 22.04.5 LTS**

The Wireshark utility is available on all major desktop platforms, i.e., Linux, Microsoft Windows, FreeBSD,
MacOS, Solaris, and many more. Follow the steps below to install Wireshark on Ubuntu 22.04. 5LTS

**Step 1 : Update APT**

First, as always, update and upgrade your APT through the following command.

**Syntax:**

$ sudo apt update

**Step 2: Download and Install Wireshark**

Now that Wireshark's latest version has been added to the APT, you can download and install it with the following command.

**Syntax:**

$ sudo apt install wireshark

### Step 3: Enable Root Privileges

When Wireshark installs on your system, you will be prompted by the following   window.

As Wireshark requires superuser/root privileges to operate, this option  asks to

enable or disable permissions for all every user on the system. Press the

 "Yes" button to allow other users, or press the "No" button to restrict other users from using Wireshark.

### Step 4:

You must add a username to the Wireshark group so that this user can use Wireshark. To do this, execute the following command, adding your required username after "wireshark" in the command.

### Syntax:

$ sudo adduser $user wireshark

### Step 5: Launch Wireshark

In the terminal window, type the following command to start the Wireshark application.

### Syntax:

**$ wireshark**

You can also open Wireshark through the Graphical User Interface (GUI) by opening the activities on the Ubuntu desktop, and in the search bar, type "Wireshark," and click on the application result.

### Step6: Verify Installation

**$ wireshark –version**

You now have **Wireshark** installed on **Ubuntu 22.04.5 LTS** and configured for use as regular user.
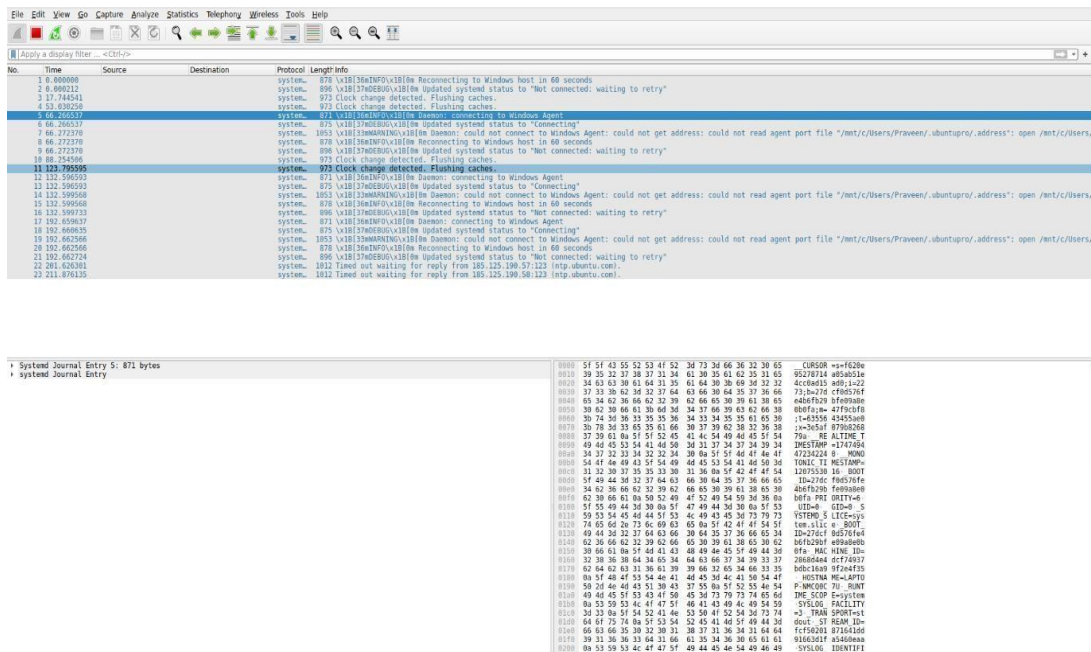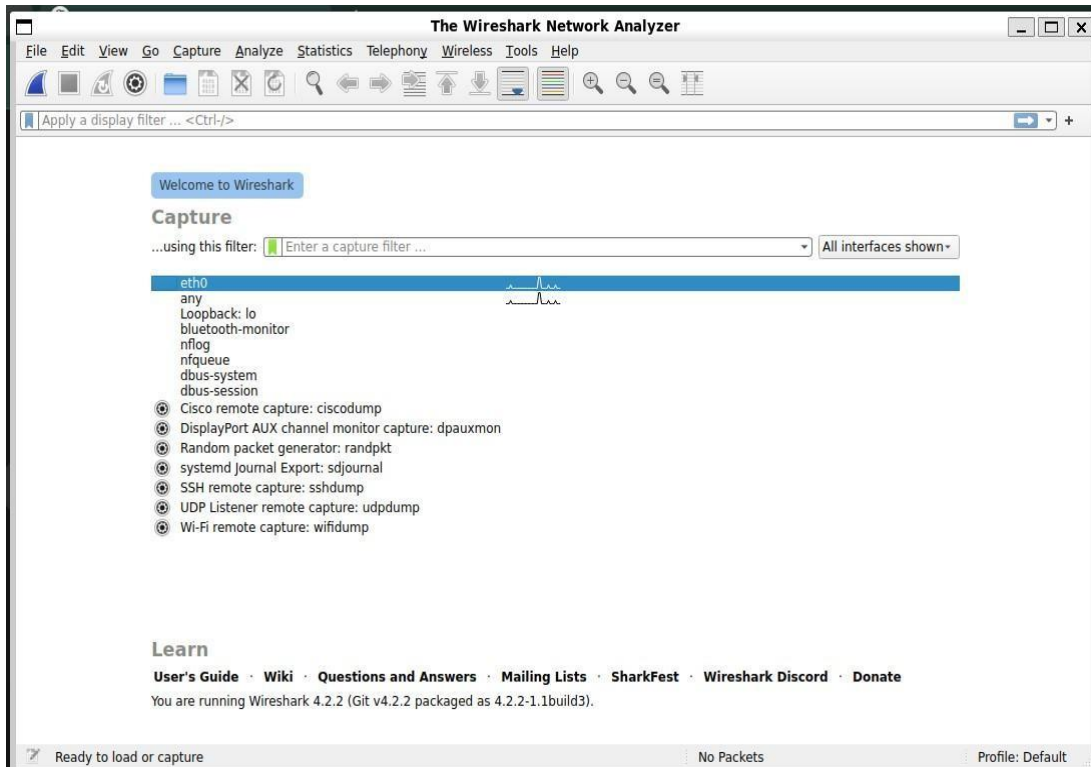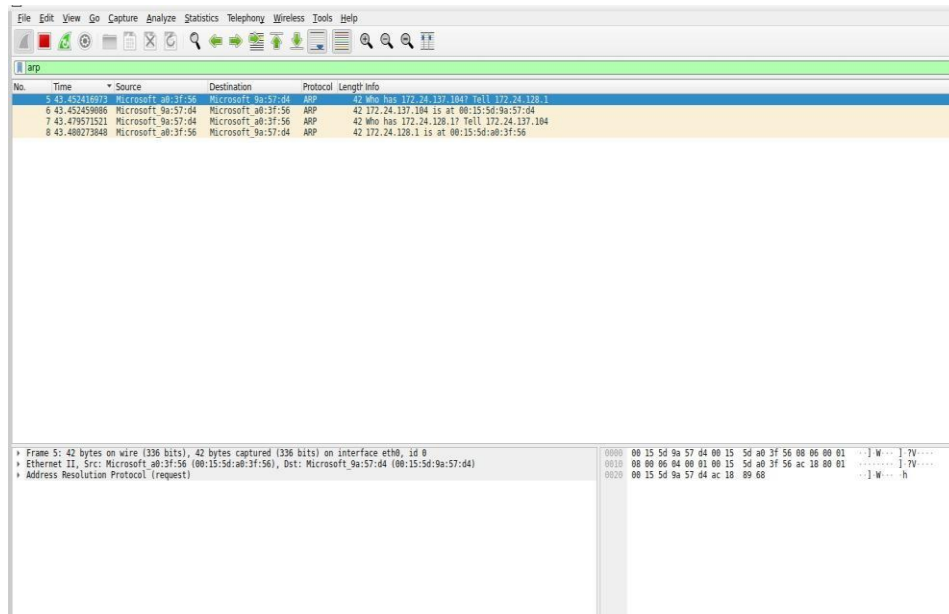
## 1.Frame Information (Physical Layer / Data Link Layer)

These fields give details about the physical characteristics of the captured frame.

- **Frame Number:** The sequential number of the captured frame in the capture session.

- **Arrival Time:** The timestamp indicating when the packet was captured.

- **Frame Length:** The total length of the frame (in bytes).

- **Captured Length:** The length of the packet captured (it may be less than the full frame if packet slicing is enabled).

- **Protocol:** The highest-level protocol identified in the packet (e.g., Ethernet, IPv4, TCP, HTTP).

- **Data:** This shows the raw byte data of the frame, typically in hexadecimal and ASCII format.

## 2. Ethernet Layer (Data Link Layer)

Ethernet frames are the most common type of data link layer packet.

- **Destination MAC Address:** The MAC address of the recipient.

- **Source MAC Address:** The MAC address of the sender.

● **EtherType:** Indicates which protocol is being encapsulated inside the Ethernet frame (e.g., IPv4, IPv6, ARP).

## 3. IP Layer (Network Layer)

The IP layer provides information about how data is being routed across networks.

● **Version:** The version of IP (IPv4 or IPv6).

● **Header Length:** The length of the IP header.

● **Type of Service (TOS):** Defines the quality of service (QoS) for the packet.

● **Total Length:** The total length of the entire IP packet, including the header and data.

● **Identification:** A unique identifier used for fragmenting the packet if necessary.

● **Flags:** Controls fragmenting behavior, including "Don't Fragment" (DF) and "More Fragments" (MF) flags.

● **Fragment Offset:** The position of this fragment in relation to the original data.

● **Time to Live (TTL):** The number of hops the packet can make before being discarded.

● **Protocol:** The next layer protocol, such as TCP, UDP, ICMP.

● **Header Checksum:** Used to detect errors in the IP header.

● **Source IP Address:** The IP address of the sender.

● **Destination IP Address:** The IP address of the receiver.

## 4. Transport Layer (TCP/UDP)

This layer deals with end-to-end communication between devices.

● **Source Port:** The port number on the sender's side.

● **Destination Port:** The port number on the receiver's side.

● **Sequence Number:** In TCP, this is the number of the first byte of data in the segment.

● **Acknowledgment Number:** In TCP, this indicates the next sequence number the sender expects.

● **Data Offset:** The length of the TCP header.

● **Flags:** Various control flags (for TCP), including:

○ **SYN:** Synchronize the connection.

○ **ACK:** Acknowledge received data.

○ **FIN:** Finish a connection. ○ **RST:** Reset a connection.

○ **PSH:** Push function; request immediate delivery.

○ **URG:** Urgent data.

● **Window Size:** Defines the buffer space available to the receiver.

● **Checksum:** Error checking for the TCP/UDP header.

● **Urgent Pointer:** Indicates the end of urgent data (only if URG flag is set).

● **Options:** Any additional options in the TCP/UDP header, such as Maximum Segment Size (MSS) in TCP.

## 5. Application Layer (e.g., HTTP, DNS, etc.)

This layer represents the actual application-level protocol data being transferred.

● **HTTP:**

○ **Request Method:** The type of HTTP request (e.g., GET, POST).

○ **Status Code:** The HTTP response code (e.g., 200 OK).

○ **Host:** The domain name of the server.

○ **User-Agent:** The client software making the request.

○ **Content-Type:** The type of data being sent (e.g., text/html, application/json).

○ **Cookies:** If present, cookies sent between client and server.

○ **Content-Length:** The length of the body data (in bytes).

○ **Headers:** Other headers, such as Authorization, Accept, etc.

○ **Data:** The actual content (e.g., HTML, JSON) sent in the body of the HTTP request or response.

● **DNS:**

○ **Transaction ID:** A unique identifier for the DNS query/response.

○ **Flags:** Indicates if the DNS message is a query or a response.

○ **Questions:** The DNS query being made.

○ **Answers:** The DNS responses (e.g., resolved IP address).

○ **Authority Records:** Information about authoritative name servers.

 ○ **Additional Records:** Additional data returned by the DNS server.

● **SMTP:**

○ **Sender:** The email address of the sender.

○ **Recipient:** The email address of the recipient.

○ **Subject:** The subject line of the email.

○ **Message-ID:** A unique identifier for the email message.

○ **Data:** The actual body content of the email.

● **Other Application Protocols:** For other protocols like FTP, Telnet, or DHCP, similar breakdowns will occur. For example, in FTP, you will see commands like USER, PASS, LIST, etc., along with the corresponding response codes.

### 6. ICMP Layer (Internet Control Message Protocol)

ICMP is used for network diagnostic and error reporting.

● **Type:** Type of ICMP message (e.g., Echo Request, Echo Reply).

● **Code:** The specific type of message (e.g., network unreachable, destination unreachable).

● **Checksum:** Error checking for the ICMP message.

● **Identifier:** Identifier used to match Echo Requests and Echo Replies.

● **Sequence Number:** The sequence number used to match Echo Requests and Echo Replies.
  ● **Data:** The payload data of the ICMP message.

## 7. ARP (Address Resolution Protocol)

ARP is used to resolve IP addresses to MAC addresses.

● **Hardware Type:** Specifies the type of hardware (e.g., Ethernet).

● **Protocol Type:** The protocol being mapped (e.g., IPv4).

● **Sender MAC Address:** The MAC address of the sender.

● **Sender IP Address:** The IP address of the sender.

● **Target MAC Address:** The MAC address of the target (blank in ARP requests).

● **Target IP Address:** The IP address of the target. 8. TLS/SSL (Transport Layer Security/Secure Sockets Layer).

● **Record Layer:** Includes details about the TLS/SSL handshake, encryption, and session keys.

● **Handshake Protocol:** Details about the negotiation of encryption settings, including:

○ **Client Hello:** The client's request to establish a secure connection.

○ **Server Hello:** The server's response confirming the session.

○ **Certificate:** The server's certificate used for authentication.

○ **Key Exchange:** Information on how the symmetric encryption key is exchanged.

## 9. Miscellaneous Fields

● **Time Delta:** Time difference between consecutive packets.

● **Protocol Hierarchy:** Provides an overview of the protocols seen in the capture, summarizing their distribution and usage.

## RESULT

Packets are captured and analyzing sucessfully.

# EXPERIMENT: 08
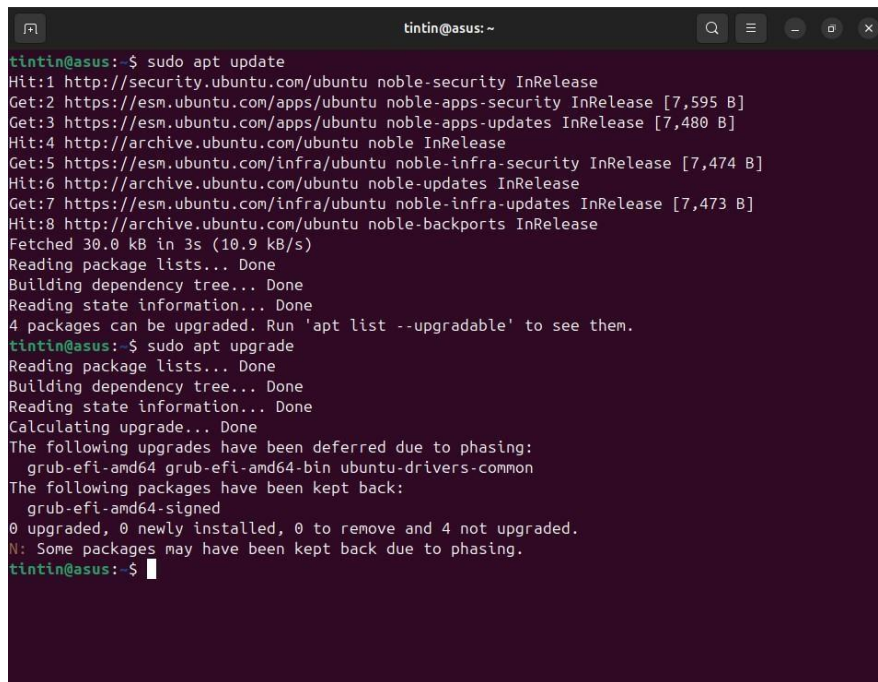
**AIM**

Installation and configuration of LAMP stack.

**PROCEDURE:**

**Update your system repositories to the latest version:**

sudo apt update



**Install Apache using apt:** sudo apt install apache2

```
tintin@asus:~$ sudo apt install apache2
Reading package lists... Done
Building dependency tree... Done
Reading state information... Done
The following additional packages will be installed:
  apache2-bin apache2-data apache2-utils libapr1t64 libaprutil1-dbd-sqlite3 libaprutil1-ldap
  libaprutil1t64
Suggested packages:
  apache2-doc apache2-suexec-pristine | apache2-suexec-custom
The following NEW packages will be installed:
  apache2 apache2-bin apache2-data apache2-utils libapr1t64 libaprutil1-dbd-sqlite3
  libaprutil1-ldap libaprutil1t64
0 upgraded, 8 newly installed, 0 to remove and 4 not upgraded.
Need to get 1,900 kB of archives.
After this operation, 7,455 kB of additional disk space will be used.
```

**Confirm that Apache is now running with the following command:**

sudo systemctl status apache2

```
tintin@asus:~$ sudo systemctl start apache2
tintin@asus:~$ sudo systemctl status apache2
● apache2.service - The Apache HTTP Server
     Loaded: loaded (/usr/lib/systemd/system/apache2.service; enabled; preset: enabled)
     Active: active (running) since Sat 2025-05-17 18:26:38 IST; 1min 47s ago
       Docs: https://httpd.apache.org/docs/2.4/
   Main PID: 6248 (apache2)
      Tasks: 55 (limit: 9040)
     Memory: 5.5M (peak: 6.8M)
        CPU: 48ms
     CGroup: /system.slice/apache2.service
             ├─6248 /usr/sbin/apache2 -k start
             ├─6250 /usr/sbin/apache2 -k start
             └─6251 /usr/sbin/apache2 -k start

May 17 18:26:38 asus systemd[1]: Starting apache2.service - The Apache HTTP Server...
May 17 18:26:38 asus apachectl[6247]: AH00558: apache2: Could not reliably determine the serv
May 17 18:26:38 asus systemd[1]: Started apache2.service - The Apache HTTP Server.
```

**If it is not working!** sudo systemctl stop

apache2  # to stop if running sudo systemctl

start apache2  # to start if not running

**Once installed, test by accessing your server's IP in your browser:**

http://127.0.0.1/

### Install MariaDB

sudo apt install mariadb-server mariadb-client



### Confirm that MariaDB is now running with the following command:

sudo systemctl status mysql # to check status sudo systemctl start mysq #

if not running

### Install PHP and commonly used modules:

sudo apt install php libapache2-mod-php php-opcache php-cli php-gd php-curl php-mysql



### Restart the apache2 service: sudo
systemctl restart apache2

### Test PHP Processing on Web Server by creating a php file as follows:

sudo nano /var/www/html/phpinfo.php



### Inside the file, type in the valid PHP code:

<?php

phpinfo ();

?>

**Press CTRL + X to save and close the file. Press y and ENTER to confirm.**

**Now open a browser and type in your IP address as:**

http://127.0.0.1/phpinfo.php



**RESULT**

Familiarized with LAMP installation

# EXPERIMENT-09

**AIM:**

Installing and configuring Laravel

**PROCEDURE**

To install and configure the Laravel PHP framework on a local system by setting up the necessary prerequisites, creating a new Laravel project, and testing the application using Laravel's built-in development server.

Laravel is a popular PHP framework used to build web applications. On Linux, Laravel runs smoothly because Linux supports the tools it needs, like PHP, Composer, Apache/Nginx, and MySQL.Laravel uses the MVC (Model-View-Controller) structure, making code cleaner and easier to manage. It comes with built-in features like routing, authentication, and database migration.In Linux, Laravel is installed using Composer.

**Steps to install Laravel:**

Step 1: Prerequisites

Install Required Software: Make sure you have a web server (e.g., Apache or Nginx), PHP, Composer (dependency manager), and a database server (e.g., MySQL) installed on your system.

Install Composer: Download and install Composer.

•        Install curl tool:

Laravel is installed using Composer, and the standard way to install Composer is:

•        curl -sS https://getcomposer.org/installer | php This command uses curl to:

•        Fetch the Composer installer script from the official Composer website.

•        Pipe it to PHP to run the installer.

```
cep@cep-OptiPlex-5055-Ryzen-CPU:~/sreelakshmi$ sudo apt install -y curl
Reading package lists... Done
Building dependency tree... Done
Reading state information... Done
curl is already the newest version (8.5.0-2ubuntu10.6).
The following packages were automatically installed and are no longer required:
  libevent-core-2.1-7t64 libevent-pthreads-2.1-7t64 libmecab2 libprotobuf-lite32t64 mecab-ipadic mecab-ipadic-utf8 mecab-utils
Use 'sudo apt autoremove' to remove them.
0 upgraded, 0 newly installed, 0 to remove and 326 not upgraded.
```

```
cep@cep-OptiPlex-5055-Ryzen-CPU:~/sreelakshmi$ curl -s s https://getcomposer.org/installer | sudo php -- ---dir=/usr/bin-filename=composer
All settings correct for using Composer
Downloading...

Composer (version 2.8.9) successfully installed to: /home/cep/sreelakshmi/composer.phar
Use it: php composer.phar

cep@cep-OptiPlex-5055-Ryzen-CPU:~/sreelakshmi$ sudo apt install composer
Reading package lists... Done
Building dependency tree... Done
Reading state information... Done
The following packages were automatically installed and are no longer required:
```

Step 2: Install Laravel

Create a New Laravel Project: Open a terminal and navigate to the directory where you want to create your Laravel project. Run the following command:   composer create-project laravel/laravel test1

This will create a new Laravel project named "test1"

```
cep@cep-OptiPlex-5055-Ryzen-CPU:~/sreelakshmi$ composer create-project laravel/laravel test1
Creating a "laravel/laravel" project at "./test1"
Installing laravel/laravel (v12.0.8)
  - Downloading laravel/laravel (v12.0.8)
  - Installing laravel/laravel (v12.0.8): Extracting archive
Created project in /home/cep/sreelakshmi/test1
> @php -r "file_exists('.env') || copy('.env.example', '.env');"
Loading composer repositories with package information
Updating dependencies
```

```
   INFO  Discovering packages.

   laravel/sail .................................................................
   laravel/sanctum ..............................................................
   laravel/tinker ...............................................................
   nesbot/carbon ................................................................
   nunomaduro/collision .........................................................
   nunomaduro/termwind ..........................................................
   spatie/laravel-ignition ......................................................

81 packages you are using are looking for funding.
Use the `composer fund` command to find out more!
> @php artisan vendor:publish --tag=laravel-assets --ansi --force

   INFO  No publishable resources for tag [laravel-assets].

> @php artisan key:generate --ansi

   INFO  Application key set successfully.

cep@cep-OptiPlex-5055-Ryzen-CPU:~/Desktop$ php artisan server
```

Step 3:Run Laravel Development Server

   php artisan serve

 The php artisian serve command is a built-in Laravel development server.It allow you to quickly run a local server without configuring Apache or Nginx manually.

Step 4: Testing the Setup

Access the application: Open a web browser and visit http://127.0.0.1:8000 in the URL bar after starting the Laravel server.

You should see the Laravel welcome page. This means your Laravel project is installed and running correctly.
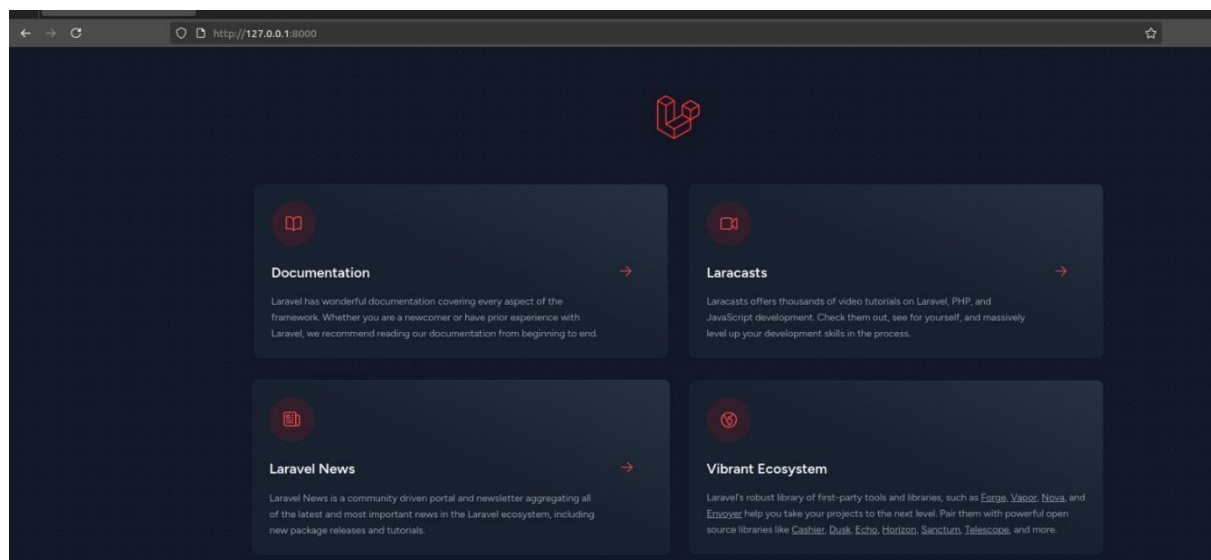


**RESULT**

Familiarized with Laravel and it's installation process.