



**Datta Meghe College of Engineering**  
**Airoli, Navi Mumbai**

**DEPARTMENT OF COMPUTER ENGINEERING**  
**ACADEMIC YEAR : 2023 – 24 (TERM – I)**

**List of Experiments**

**Course Name : Computer Network**  
**Course Code : CSL502/CSC503.**

Sr. No	Name of experiment	Cos Covered	Page No.	Date of Performance	Date of Submission	Marks & Signature
1	Use of Crimping Tool for RJ45.	CSC503.1				
2	Use of basic networking Commands in Linux.	CSC503.1				
3	Perform network discovery using discovery tools ( eg. Nmap, mrtg )	CSC503.2, CSC503.3				
4	WAP to implement Socket Programming using TCP & UDP.	CSC503.5				
5	Perform File Transfer and Access using FTP.	CSC503.6				
6	Install and use Telnet in Ubuntu.	CSC503.6				
7	Build simple Topology.	CSC503.1				
8	Study and installation of Network Simulator.	CSC503.1				
9	Use Wireshark to understand the operation of TCPIP layers.	CSC503.4, CSC503.5				
10	Setup multiple addresses on a single LAN.	CSC503.2				
	Assignment 1	CSC503.1, CO2				
	Assignment 2	CSC503.3, CSC503.4, CSC503.5 & CSC503.6				

This is to certify that Mr. / Miss \_\_\_\_\_ of \_\_\_\_\_ Roll No. \_\_\_\_\_ has performed the Experiments / Assignments / Tutorials / Case Study Work mentioned above in the premises of the institution.

\_\_\_\_\_  
**Practical Incharge**



**DATTA MEGHE COLLEGE OF ENGINEERING,  
AIROLI, NAVI MUMBAI**

**DEPARTMENT OF COMPUTER ENGINEERING**

**Institute Vision** : To create value - based technocrats to fit in the world of work and research

**Institute Mission** :

- To adopt the best engineering practices
- To empower students to work in the world of technology and research
- To create competent human beings

**Department Vision** : To provide an intellectually stimulating environment for education, technological excellence in computer engineering field and professional training along with human values.

**Department Mission :**

- M1:** To promote an educational environment that combines academics with intellectual curiosity.
- M2:** To develop human resource with sound knowledge of theory and practical in the discipline of Computer Engineering and the ability to apply the knowledge to the benefit of society at large.
- M3:** To assimilate creative research and new technologies in order to facilitate students to be a lifelong learner who will contribute positively to the economic well-being of the nation.

**Program Educational Objectives (PEO):**

- PEO1:** To explicate optimal solutions through application of innovative computer science techniques that aid towards betterment of society.
- PEO2:** To adapt recent emerging technologies for enhancing their career opportunity prospects.
- PEO3:** To effectively communicate and collaborate as a member or leader in a team to manage multidisciplinary projects
- PEO4:** To prepare graduates to involve in research, higher studies or to become entrepreneurs in long run.

**Program Specific Outcomes (PSO):**

- PSO1:** To apply basic and advanced computational and logical skills to provide solutions to computer engineering problems
- PSO2:** Ability to apply standard practices and strategies in design and development of software and hardware based systems and adapt to evolutionary changes in computing to meet the challenges of the future.
- PSO3:** To develop an approach for lifelong learning and utilize multi-disciplinary knowledge required for satisfying industry or global requirements.

## **Program Outcomes as defined by NBA (PO)**

### **Engineering Graduates will be able to:**

1. **Engineering knowledge:** Apply the knowledge of mathematics, science, engineering fundamentals, and an engineering specialization to the solution of complex engineering problems.
2. **Problem analysis:** Identify, formulate, review research literature, and analyze complex engineering problems reaching substantiated conclusions using first principles of mathematics, natural sciences, and engineering sciences.
3. **Design/development of solutions:** Design solutions for complex engineering problems and design system components or processes that meet the specified needs with appropriate consideration for the public health and safety, and the cultural, societal, and environmental considerations.
4. **Conduct investigations of complex problems:** Use research-based knowledge and research methods including design of experiments, analysis and interpretation of data, and synthesis of the information to provide valid conclusions.
5. **Modern tool usage:** Create, select, and apply appropriate techniques, resources, and modern engineering and IT tools including prediction and modelling to complex engineering activities with an understanding of the limitations.
6. **The engineer and society:** Apply reasoning informed by the contextual knowledge to assess societal, health, safety, legal and cultural issues and the consequent responsibilities relevant to the professional engineering practice.
7. **Environment and sustainability:** Understand the impact of the professional engineering solutions in societal and environmental contexts, and demonstrate the knowledge of, and need for sustainable development.
8. **Ethics:** Apply ethical principles and commit to professional ethics and responsibilities and norms of the engineering practice.
9. **Individual and team work:** Function effectively as an individual, and as a member or leader in diverse teams, and in multidisciplinary settings.
10. **Communication:** Communicate effectively on complex engineering activities with the engineering community and with society at large, such as, being able to comprehend and write effective reports and design documentation, make effective presentations, and give and receive clear instructions.
11. **Project management and finance:** Demonstrate knowledge and understanding of the engineering and management principles and apply these to one's own work, as a member and leader in a team, to manage projects and in multidisciplinary environments.
12. **Life-long learning:** Recognize the need for, and have the preparation and ability to engage in independent and life-long learning in the broadest context of technological change.

# **DATTA MEGHIE COLLEGE OF ENGINEERING**

## **DEPARTMENT OF COMPUTER ENGINEERING**

---

**Course Name: Computer Network Lab (R-19)**

**Course Code: CSC503/CSL502**

**Year of Study: T.E., Semester: V**

### **Course Outcomes**

CSC503.1	Demonstrate the concepts of data communication at physical layer and compare ISO - OSI model with TCP/IP model.
CSC 503.2	Demonstrate the knowledge of networking protocols at data link layer.
CSC 503.3	Design the network using IP addressing and subnetting / supernetting schemes.
CSC 503.4	Analyze various routing algorithms and protocols at network layer.
CSC 503.5	Analyze transport layer protocols and congestion control algorithms.
CSC 503.6	Explore protocols at application layer .

**DATTA MEGHE COLLEGE OF ENGINEERING**  
**DEPARTMENT OF COMPUTER ENGINEERING**  
**ACADEMIC YEAR 2023-24 (TERM I)**  
**SUBJECT: COMPUTER NETWORK**  
**SEM: V**

**RUBRICS FOR GRADING EXPERIMENTS**

<b>Rubric Number</b>	<b>Rubric Title</b>	<b>Criteria</b>	<b>Marks (out of 15)</b>
R1	Punctuality, Completion Time / Timeline	On-time	3
		Delayed by a Week	2
		Delayed more than a Week	1
R2	Knowledge & Concept	Clear understanding	3
		Partially understood	2
		Weak understanding	1
R3	Implementation	Correct Implementation	3
		Partial Implementation	2
		Implementation with error	1
R4	Results	Correct Results	3
		Partial Results	2
		Results with error	1
R5	Documentation	Correct Documentation	3
		Moderate documented	2

		Not properly organized	1
--	--	------------------------	---

**DATTA MEGHE COLLEGE OF ENGINEERING**  
**DEPARTMENT OF COMPUTER ENGINEERING**  
**ACADEMIC YEAR 2023-24 (TERM I)**  
**SUBJECT: Computer Network**  
**SEM: V**  
**RUBRICS FOR GRADING ASSIGNMENTS**

<b>Rubric Number</b>	<b>Rubric Title</b>	<b>Criteria</b>	<b>Marks (out of 5)</b>
<b>R1</b>	<b>Punctuality, Completion Time / Timeline</b>	<b>On-time</b>	<b>2</b>
		<b>Delayed by a Week</b>	<b>1</b>
		<b>Delayed more than a Week</b>	<b>0</b>
<b>R2</b>	<b>Knowledge &amp; Concept</b>	<b>Clear understanding</b>	<b>2</b>
		<b>Partially understood</b>	<b>1</b>
		<b>Weak understanding</b>	<b>0</b>
<b>R3</b>	<b>Documentation</b>	<b>Correct Documentation</b>	<b>1</b>
		<b>Not documented properly</b>	<b>0</b>

## EXPERIMENT NO: 1

**Date of Performance :**

**Date of Submission :**

**AIM:** Use of Crimping Tool for RJ45.

### **THEORY:**

Crimping an RJ45 Connector Correctly Proper Wiring for Ethernet Cat5/Cat5e/Cat 6 Cables



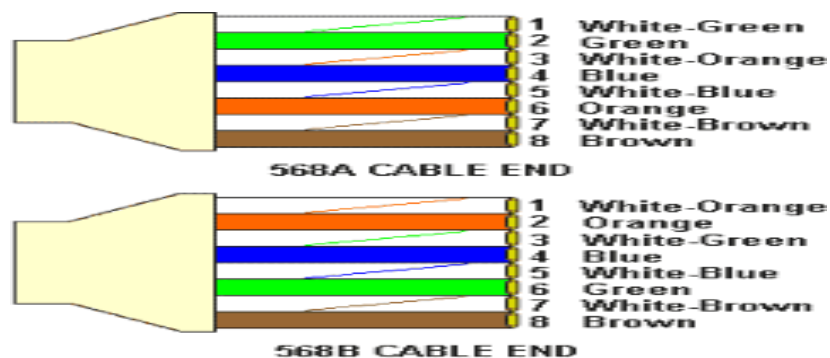
Cables can transmit information along their length. To actually get that information where it needs to go, you need to make the right connections to an RJ45 connector.

Your cable run needs to terminate into a connector, and that connector needs a jack to plug into.

Registered Jack 45 (RJ45) is a standard type of physical connector for network cables. RJ45 connectors are commonly seen with Ethernet cables and networks.

Modern Ethernet cables feature a small plastic plug on each end of the cable. That plug is inserted into RJ45 jacks of Ethernet devices. The term “plug” refers to the cable or “male” end of the connection while the term “jack” refers to the port or “female” end.

T568A or T568B Wiring Standard:



T568A and T568B are the two colour codes used for wiring eight-position modular plugs. Both are allowed under the ANSI/TIA/EIA wiring standards. The only difference between the two color codes is that the orange and green pairs are interchanged.

There is no transmission differences between T568A and T568B cabling schemes. North America's preference is for T568B. Both ends must use the same standard. It makes no difference to the transmission characteristics of data.

**T568B** wiring pattern is recognized as the preferred wiring pattern.

### STEP 1:

Using a Crimping Tool, trim the end of the cable you're terminating, to ensure that the ends of the conducting wires are even.



### STEP 2:

Being careful not to damage the inner conducting wires, strip off approximately 1 inch of the cable's jacket, using a modular crimping tool or a UTP cable stripper.



### STEP 3:

Separate the 4 twisted wire pairs from each other, and then unwind each pair, so that you end up with 8 individual wires. Flatten the wires out as much as possible, since they'll need to be very straight for proper insertion into the connector.



### STEP 4:

Holding the cable with the wire ends facing away from you. Moving from left to right, arrange the wires in a flat, side-by-side ribbon formation, placing them in the following order: white/orange, solid orange, white/green, solid blue, white/blue, solid green, white/brown, solid brown.





### **STEP 5:**

Holding the RJ45 connector so that its pins are facing away from you and the plug-clip side is facing down, carefully insert the flattened, arranged wires into the connector, pushing through until the wire ends emerge from the pins. For strength of connection, also push as much of the cable jacket as possible into the connector.



### **STEP 6:**

Check to make sure that the wire ends coming out of the connector's pin side are in the correct order; if not, remove them from the connector, rearrange into proper formation, and re-insert. Remember, once the connector is crimped onto the cable, it's permanent. If you realize that a mistake has been made in wire order after termination, you'll have to cut the connector off and start all over again!



### STEP 7:

Insert the prepared connector/cable assembly into the RJ45 slot in your crimping tool. Firmly squeeze crimper's handles together until you can't go any further. Release the handles and repeat this step to e proper crimp.



### STEP 8:

If your crimper doesn't automatically trim the wire ends upon termination, carefully cut wire ends to them as flush with the connector's surface as possible. The closer the wire ends are trimmed, the better final plug-in connection will be.



### STEP 9:

After the first termination is complete, repeat process on the opposite end of your cable.



**CONCLUSION:** Thus, we have studied the use of crimping tool for RJ-45.

**Sign and Remark:**

R1 (3 Marks)	R2 (3 Marks)	R3 (3 Marks)	R4 (3 Mark)	R5 (3 Mark)	Total (15 Marks)	Signature

## EXPERIMENT No. 2

Date of Performance :

Date of Submission :

**AIM:** Use basic networking commands in Linux (ping, tracer, nslookup, netstat, ARP, RARP, ip, ifconfig, dig, route )

### THEORY:

#### 1. ifconfig

**ifconfig**(interface configuration) command is used to configure the kernel-resident network interfaces. It is used at the boot time to set up the interfaces as necessary. After that, it is usually used when needed during debugging or when you need system tuning. Also, this command is used to assign the IP address and netmask to an interface or to enable or disable a given interface.

```
student@lenovo804-ThinkCentre-M70e: ~  
student@lenovo804-ThinkCentre-M70e:~$ ifconfig  
docker0    Link encap:Ethernet  HWaddr 02:42:cf:c7:15:71  
            inet addr:172.17.0.1  Bcast:0.0.0.0  Mask:255.255.0.0  
            UP BROADCAST MULTICAST  MTU:1500  Metric:1  
            RX packets:0 errors:0 dropped:0 overruns:0 frame:0  
            TX packets:0 errors:0 dropped:0 overruns:0 carrier:0  
            collisions:0 txqueuelen:0  
            RX bytes:0 (0.0 B)  TX bytes:0 (0.0 B)  
  
eth0       Link encap:Ethernet  HWaddr 44:37:e6:4d:df:1b  
            inet addr:10.1.8.4  Bcast:10.255.255.255  Mask:255.0.0.0  
            inet6 addr: fe80::4637:e6ff:fe4d:df1b/64 Scope:Link  
            UP BROADCAST RUNNING MULTICAST  MTU:1500  Metric:1  
            RX packets:51944 errors:0 dropped:0 overruns:0 frame:0  
            TX packets:18626 errors:0 dropped:0 overruns:0 carrier:0  
            collisions:0 txqueuelen:1000  
            RX bytes:27621649 (27.6 MB)  TX bytes:2682227 (2.6 MB)  
            Interrupt:17  
  
lo         Link encap:Local Loopback  
            inet addr:127.0.0.1  Mask:255.0.0.0  
            inet6 addr: ::1/128 Scope:Host  
            UP LOOPBACK RUNNING  MTU:65536  Metric:1  
            RX packets:2173 errors:0 dropped:0 overruns:0 frame:0  
            TX packets:2173 errors:0 dropped:0 overruns:0 carrier:0  
            collisions:0 txqueuelen:0  
            RX bytes:193433 (193.4 KB)  TX bytes:193433 (193.4 KB)  
  
student@lenovo804-ThinkCentre-M70e:~$
```

#### 2. NSLOOKUP

**Nslookup** (stands for “Name Server Lookup”) is a useful command for getting information from 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.

```
student@lenovo804-ThinkCentre-M70e: ~  
student@lenovo804-ThinkCentre-M70e:~$ nslookup www.atharvacoe.ac.in  
Server:      127.0.1.1  
Address:     127.0.1.1#53  
  
Non-authoritative answer:  
www.atharvacoe.ac.in  canonical name = atharvacoe.ac.in.  
Name:   atharvacoe.ac.in  
Address: 192.185.180.65  
  
student@lenovo804-ThinkCentre-M70e:~$
```

### 3. Ping

PING (Packet Internet Groper) command is used to check the network connectivity between host and server/host. This command takes as input the IP address or the URL and sends a data packet to the specified address with the message “PING” and get a response from the server/host this time is recorded which is called latency. Fast ping low latency means faster connection. Ping uses [ICMP\(Internet Control Message Protocol\)](#) to send an ICMP echo message to the specified host if that host is available then it sends ICMP reply message. Ping is generally measured in millisecond every modern operating system has this ping pre-installed.

```
student@lenovo804-ThinkCentre-M70e: ~  
student@lenovo804-ThinkCentre-M70e:~$ ping -c 4 10.1.8.3  
PING 10.1.8.3 (10.1.8.3) 56(84) bytes of data.  
64 bytes from 10.1.8.3: icmp_seq=1 ttl=64 time=0.324 ms  
64 bytes from 10.1.8.3: icmp_seq=2 ttl=64 time=0.333 ms  
64 bytes from 10.1.8.3: icmp_seq=3 ttl=64 time=0.316 ms  
64 bytes from 10.1.8.3: icmp_seq=4 ttl=64 time=0.302 ms  
  
--- 10.1.8.3 ping statistics ---  
4 packets transmitted, 4 received, 0% packet loss, time 3000ms  
rtt min/avg/max/mdev = 0.302/0.318/0.333/0.024 ms  
student@lenovo804-ThinkCentre-M70e:~$
```

### 4. TRACEROUTEac

**traceroute** command in Linux prints the route that a packet takes to reach the host. This command is useful when you want to know about the route and about all the hops that a packet takes. Below image depicts how traceroute command is used to reach the Google(172.217.26.206) host from the local machine and it also prints detail about all the hops that it visits in between.



```

student@lenovo804-ThinkCentre-M70e:~$ traceroute
Usage:
  traceroute [ -46dFITnreAUV ] [ -f first_ttl ] [ -g gate,... ] [ -i device ] [ -m max_ttl ] [ -N queries ] [ -p port ] [ -t tos ] [ -l flow_label ] [ -w waittime ] [ -q queries ] [ -s src_addr ] [ -z sendwait ] [
  --fwmark=num ] host [ packetlen ]
Options:
  -4                               Use IPv4
  -6                               Use IPv6
  -d --debug                       Enable socket level debugging
  -F --dont-fragment              Do not fragment packets
  -f first_ttl --first=first_ttl   Start from the first ttl hop (instead from 1)
  -g gate,... --gateway=gate,...   Route packets through the specified gateway
                                   (maximum 8 for IPv4 and 127 for IPv6)
  -I --icmp                       Use ICMP ECHO for tracerouting
  -T --tcp                        Use TCP SYN for tracerouting (default port is 80)
  -i device --interface=device     Specify a network interface to operate with
  -m max_ttl --max-hops=max_ttl   Set the max number of hops (max TTL to be
                                   reached). Default is 30
  -N queries --sim-queries=squeries Set the number of probes to be tried
                                   simultaneously (default is 16)
  -n                               Do not resolve IP addresses to their domain names
  -p port --port=port             Set the destination port to use. It is either
                                   initial udp port value for "default" method
                                   (incremented by each probe, default is 33434), or
                                   initial seq for "icmp" (incremented as well,
                                   default from 1), or some constant destination
                                   port for other methods (with default of 80 for
                                   "tcp", 53 for "udp", etc.)
  -t tos --tos=tos               Set the TOS (IPv4 type of service) or TC (IPv6
                                   traffic class) value for outgoing packets
  -l flow_label --flowlabel=flow_label Use specified flow_label for IPv6 packets
  -w waittime --wait=waittime     Set the number of seconds to wait for response to
                                   a probe (default is 5.0). Non-integer (float
                                   point) values allowed too
  -q queries --queries=nqueries   Set the number of probes per each hop. Default is
                                   3
  -r                               Bypass the normal routing and send directly to a
                                   host on an attached network
  -s src_addr --source=src_addr   Use source src_addr for outgoing packets
  -z sendwait --sendwait=sendwait Minimal time interval between probes (default 0).
                                   If the value is more than 10, then it specifies a
                                   number in milliseconds, else it is a number of
                                   seconds (float point values allowed too)
  -e --extensions                Show ICMP extensions (if present), including MPLS
  -A --as-path-lookups            Perform AS path lookups in routing registries and
                                   print results directly after the corresponding
                                   addresses
  -M name --module=name          Use specified module (either builtin or external)

```

## 5. Netstat

Netstat command displays various network related information such as network connections, routing tables, interface statistics, masquerade connections, multicast memberships etc.

```

student@lenovo804-ThinkCentre-M70e:~$ netstat -a
Active Internet connections (servers and established)
Proto Recv-Q Send-Q Local Address           Foreign Address         State
tcp        0      0 lenovo804-ThinkC:domain *:                        LISTEN
tcp        0      0 localhost:ipp           *:                        LISTEN
tcp        0      0 10.1.8.4:40190          bom05s11-in-f2.1e:https TIME_WAIT
tcp        0      0 10.1.8.4:52797          151.101.2.114:https    TIME_WAIT
tcp        0      0 10.1.8.4:38575          bom05s15-in-f14.1:https ESTABLISHED
tcp        0      0 10.1.8.4:38576          bom05s15-in-f14.1:https ESTABLISHED
tcp        0      0 10.1.8.4:52065          bom05s15-in-f4.1e:https TIME_WAIT
tcp        0      0 10.1.8.4:52796          151.101.2.114:https    TIME_WAIT
tcp        0      0 10.1.8.4:40191          bom05s11-in-f2.1e:https TIME_WAIT
tcp        0      0 10.1.8.4:38634          bom05s15-in-f14.1:https ESTABLISHED
tcp        0      0 10.1.8.4:38637          bom05s15-in-f14.1:https TIME_WAIT
tcp        0      0 10.1.8.4:38573          bom05s15-in-f14.1:https ESTABLISHED
tcp        0      0 10.1.8.4:37409          server-52-222-135:https TIME_WAIT
tcp        0      0 10.1.8.4:41299          a184-30-54-102.de:https TIME_WAIT

```

## 6. ARP

```
student@lenovo804-ThinkCentre-M70e: ~
student@lenovo804-ThinkCentre-M70e:~$ arp -v
Address              HWtype  HWaddress          Flags Mask          Iface
10.8.1.3              (incomplete)
10.0.0.3              ether    08:35:71:f0:35:c0   C                  eth0
10.1.8.3              ether    44:37:e6:4d:e0:f7   C                  eth0
Entries: 3           Skipped: 0           Found: 3
student@lenovo804-ThinkCentre-M70e:~$
```

**ARP command** manipulates the System's ARP cache. It also allows a complete dump of the ARP cache. ARP stands for Address Resolution Protocol. The primary function of this protocol is to resolve the IP address of a system to its mac address, and hence it works between level 2(Data link layer) and level 3(Network layer).

## 7. IP

**ip** command in Linux is present in the net-tools which is used for performing several network administration tasks. IP stands for Internet Protocol. This command is used to show or manipulate routing, devices, and tunnels. It is similar to [ifconfig](#) command but it is much more powerful with more functions and facilities attached to it. *ifconfig* is one of the deprecated commands in the net-tools of Linux that has not been maintained for many years. **ip** command is used to perform several tasks like assigning an address to a network interface or configuring network interface parameters. It can perform several other tasks like configuring and modifying the default and static routing, setting up tunnel over IP, listing IP addresses and property information, modifying the status of the interface, assigning, deleting and setting up IP addresses and routes.

## 8. Dig

**dig** command stands for *Domain Information Groper*. It is used for retrieving

```
student@lenovo804-ThinkCentre-M70e: ~
student@lenovo804-ThinkCentre-M70e:~$ ip addr show
1: lo: <LOOPBACK,UP,LOWER_UP> mtu 65536 qdisc noqueue state UNKNOWN group default
   link/loopback 00:00:00:00:00:00 brd 00:00:00:00:00:00
   inet 127.0.0.1/8 scope host lo
       valid_lft forever preferred_lft forever
   inet6 ::1/128 scope host
       valid_lft forever preferred_lft forever
2: eth0: <BROADCAST,MULTICAST,UP,LOWER_UP> mtu 1500 qdisc pfifo_fast state UP group default qlen 1000
   link/ether 44:37:e6:4d:df:1b brd ff:ff:ff:ff:ff:ff
   inet 10.1.8.4/8 brd 10.255.255.255 scope global eth0
       valid_lft forever preferred_lft forever
   inet6 fe80::4637:e6ff:fe4d:df1b/64 scope link
       valid_lft forever preferred_lft forever
3: docker0: <NO-CARRIER,BROADCAST,MULTICAST,UP> mtu 1500 qdisc noqueue state DOWN group default
   link/ether 02:42:cf:c7:15:71 brd ff:ff:ff:ff:ff:ff
   inet 172.17.0.1/16 scope global docker0
       valid_lft forever preferred_lft forever
student@lenovo804-ThinkCentre-M70e:~$
```

information about DNS name servers. It is basically used by network administrators. It is used for verifying and troubleshooting DNS problems and to perform DNS lookups. Dig command replaces older tools such as [nslookup](#) and the [host](#).

```
student@lenovo804-ThinkCentre-M70e: ~
student@lenovo804-ThinkCentre-M70e:~$ dig atharvacoe.ac.in
; <<>> DiG 9.9.5-4.3-Ubuntu <<>> atharvacoe.ac.in
;; global options: +cmd
;; Got answer:
;; ->>HEADER<<- opcode: QUERY, status: NOERROR, id: 44951
;; flags: qr rd ra; QUERY: 1, ANSWER: 1, AUTHORITY: 0, ADDITIONAL: 0

;; QUESTION SECTION:
;atharvacoe.ac.in.                IN      A

;; ANSWER SECTION:
atharvacoe.ac.in.                14399   IN      A      192.185.180.65

;; Query time: 479 msec
;; SERVER: 127.0.1.1#53(127.0.1.1)
;; WHEN: Thu Aug 30 13:58:05 IST 2018
;; MSG SIZE rcvd: 50

student@lenovo804-ThinkCentre-M70e:~$ █
```

**CONCLUSION:** Hence, in this experiment, we have successfully studied some important networking command and also implemented them in Linux

**SIGN AND REMARK**

R1 (3 Marks)	R2 (3 Marks)	R3 (3 Marks)	R4 (3 Mark)	R5 (3 Mark)	Total (15 Marks)	Signature

## **EXPERIMENT No. 3**

**Date of Performance :**

**Date of Submission :**

**AIM:** Perform network discovery using discovery tools (eg. Nmap, mrtg)

Nmap (Network Mapper) is a security scanner originally written by Gordon Lyon (also known by his pseudonym Fyodor Vaskovich) used to discover hosts and services on a computer network, thus creating a "map" of the network. To accomplish its goal, Nmap sends specially crafted packets to the target host and then analyzes the responses. Unlike many simple port scanners that just send packets at some predefined constant rate, Nmap accounts for the network conditions (latency fluctuations, network congestion, the target interference with the scan) during the run. Also, owing to the large and active user community providing feedback and contributing to its features, Nmap has been able to extend its discovery capabilities beyond simply figuring out whether a host is up or down and which ports are open and closed; it can determine the operating system of the target, names and versions of the listening services, estimated uptime, type of device, and presence of a firewall.

### **Nmap features include:**

- Host Discovery – Identifying hosts on a network. For example, listing the hosts which respond to pings or have a particular port open.
- Port Scanning – Enumerating the open ports on one or more target hosts.
- Version Detection – Interrogating listening network services listening on remote devices to determine the application name and version number.
- OS Detection – Remotely determining the operating system and some hardware characteristics of network devices.

### **Basic commands working in Nmap:**

- For target specifications: `nmap <target's URL or IP with spaces between them>`
- For OS detection: `nmap -O <target-host's URL or IP>`
- For version detection: `nmap -sV <target-host's URL or IP>`

SYN scan is the default and most popular scan option for good reasons. It can be performed quickly, scanning thousands of ports per second on a fast network not hampered by restrictive firewalls. It is also relatively unobtrusive and stealthy since it never completes TCP connections

### **Algorithm\Implementation Steps\Installation Steps:**

1. Download Nmap from [www.nmap.org](http://www.nmap.org) and install the Nmap Software with WinPcap Driver utility.
2. Execute the Nmap-Zenmap GUI tool from Program Menu or Desktop Icon
3. Type the Target Machine IP Address(ie.Guest OS or any website Address)
4. Perform the profiles shown in the utility.



Zenmap

Scan Tools Profile Help

Target: 192.168.56.101 Profile: Intense scan [Scan] [Cancel]

Command: nmap -T4 -A -v 192.168.56.101

Hosts Services Nmap Output Ports / Hosts Topology Host Details Scans

OS Host

192.168.56.101

```

Starting Nmap 7.80 ( https://nmap.org ) at 2019-09-24 09:56 India Standard Time
NSE: Loaded 151 scripts for scanning.
NSE: Script Pre-scanning.
Initiating NSE at 09:56
Completed NSE at 09:56, 0.00s elapsed
Initiating NSE at 09:56
Completed NSE at 09:56, 0.00s elapsed
Initiating NSE at 09:56
Completed NSE at 09:56, 0.00s elapsed
Initiating Ping Scan at 09:56
Scanning 192.168.56.101 [4 ports]
Completed Ping Scan at 09:56, 0.28s elapsed (1 total hosts)
Initiating Parallel DNS resolution of 1 host. at 09:56
Completed Parallel DNS resolution of 1 host. at 09:56, 0.02s elapsed
Initiating SYN Stealth Scan at 09:56
Scanning 192.168.56.101 [1000 ports]
Discovered open port 80/tcp on 192.168.56.101
Discovered open port 587/tcp on 192.168.56.101
Discovered open port 21/tcp on 192.168.56.101
Discovered open port 110/tcp on 192.168.56.101
Discovered open port 143/tcp on 192.168.56.101
Discovered open port 443/tcp on 192.168.56.101
Discovered open port 587/tcp on 192.168.56.101
Discovered open port 1863/tcp on 192.168.56.101
Discovered open port 465/tcp on 192.168.56.101
Discovered open port 5050/tcp on 192.168.56.101
Completed SYN Stealth Scan at 09:56, 6.05s elapsed (1000 total ports)
Initiating Service scan at 09:56
Scanning 10 services on 192.168.56.101
Service scan timing: About 40.00% done; ETC: 09:58 (0:01:08 remaining)
Service scan timing: About 60.00% done; ETC: 10:00 (0:01:44 remaining)
Completed Service scan at 09:59, 166.31s elapsed (10 services on 1 host)
Initiating OS detection (try #1) against 192.168.56.101
Initiating Traceroute at 09:59
Initiating Parallel DNS resolution of 1 host. at 09:59
Completed Parallel DNS resolution of 1 host. at 09:59, 0.00s elapsed
NSE: Script scanning 192.168.56.101.
Initiating NSE at 09:59
Completed NSE at 10:02, 169.07s elapsed
Initiating NSE at 10:02
Completed NSE at 10:04, 156.88s elapsed
Initiating NSE at 10:04
Completed NSE at 10:04, 0.00s elapsed
Nmap scan report for 192.168.56.101
Host is up (0.0016s latency).
Not shown: 990 filtered ports
PORT      STATE SERVICE VERSION
21/tcp    open  ftp
25/tcp    open  smtp
|_setp-commands: Couldn't establish connection on port 25
80/tcp    open  http
110/tcp   open  pop3
143/tcp   open  imap
443/tcp   open  https
465/tcp   open  smtps
|_setp-commands: Couldn't establish connection on port 465
587/tcp   open  tcpwrapped
|_setp-commands: Couldn't establish connection on port 587
1863/tcp  open  msnnp
5050/tcp  open  mmcc
Warning: OSScan results may be unreliable because we could not find at least 1 open and 1 closed port
Device type: phone|general purpose
Running: Google Android 7.X, Linux 3.X|2.6.X
OS CPE: cpe:/o:google:android:7.1.2 cpe:/o:linux:linux_kernel:3.10 cpe:/o:linux:linux_kernel:2.6
OS details: Android 7.1.2 (Linux 3.10), Linux 3.6.18 - 2.6.22
Network Distance: 1 hop
TCP Sequence Prediction: Difficulty=259 (Good luck!)
IP ID Sequence Generation: All zeros

```

Filter Hosts

Zenmap

Scan Tools Profile Help

Target: 192.168.56.101 Profile: Intense scan [Scan] [Cancel]

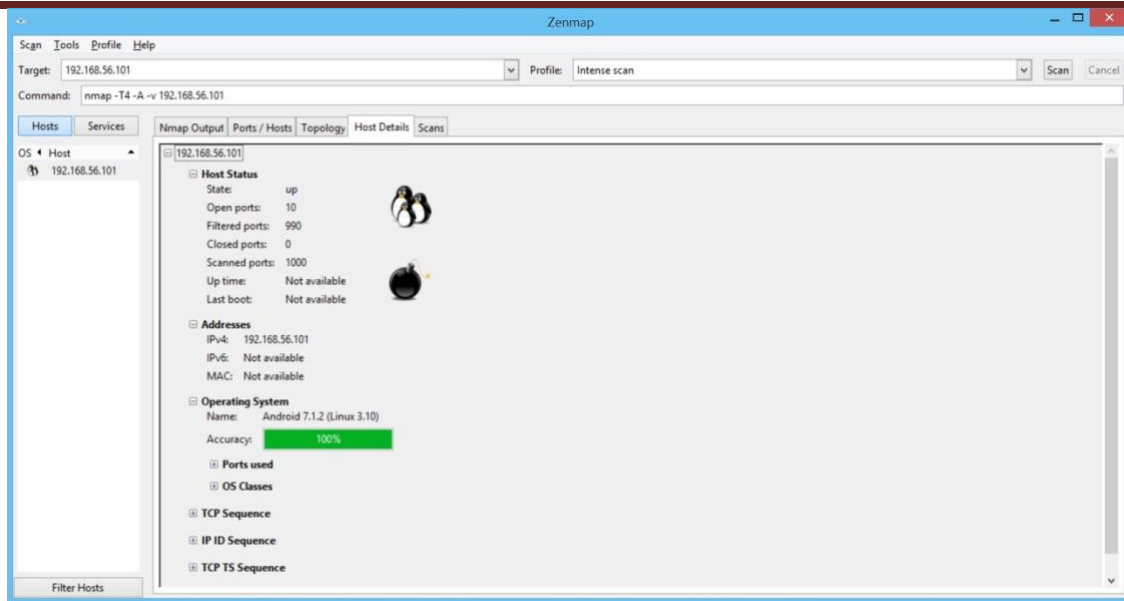
Command: nmap -T4 -A -v 192.168.56.101

Hosts Services Nmap Output Ports / Hosts Topology Host Details Scans

OS Host

192.168.56.101

Port	Protocol	State	Service	Version
21	tcp	open	ftp	
25	tcp	open	smtp	
80	tcp	open	http	
110	tcp	open	pop3	
143	tcp	open	imap	
443	tcp	open	https	
465	tcp	open	smtps	
587	tcp	open	tcpwrapped	
1863	tcp	open	msnnp	
5050	tcp	open	mmcc	



**CONCLUSION:** Thus, we have studied different options to scan ports in Nmap

## SIGN AND REMARK

R1 (3 Marks)	R2 (3 Marks)	R3 (3 Marks)	R4 (3 Mark)	R5 (3 Mark)	Total (15 Marks)	Signature

## EXPERIMENT No. 4

Date of Performance :

Date of Submission :

**Aim:**WAP to implement Socket Programming using TCP & UDP

### Theory:

**Socket** : the communication object.

A socket connection is a 4-tuple -- (HostA, PortA, HostB, PortB) -- uniquely defining the connection.

### Transmission Control Protocol (TCP)

TCP provides a *connection oriented service*, since it is based on connections between clients and servers.

TCP provides reliability. When a TCP client send data to the server, it requires an acknowledgement in return. If an acknowledgement is not received, TCP automatically retransmit the data and waits for a longer period of time.

**TCP properties:** reliable, connection-oriented, byte-stream, connection established before application-level protocols exchange information, two-way communication

**The client-server model:**The client-server model is one of the most used communication paradigms in networked systems. Clients normally communicates with one server at a time. From a server's perspective, at any point in time, it is not unusual for a server to be communicating with multiple clients. Client need to know of the existence of and the address of the server, but the server does not need to know the address of (or even the existence of) the client prior to the connection being established

### TCP Socket API

The sequence of function calls for the client and a server participating in a TCP connection is presented in below..

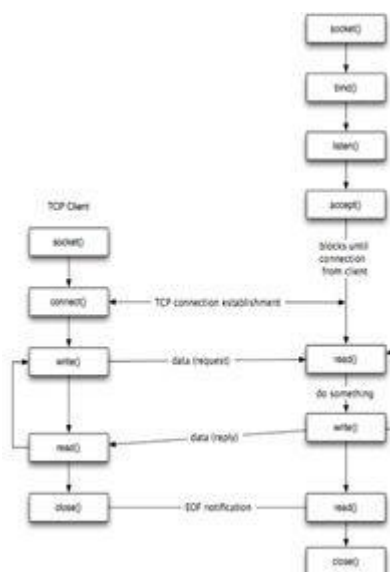


Fig: TCP Client Server

As shown in the figure, the steps for establishing a TCP socket on the client side are the following:

- Create a socket using the `socket()` function;

- Connect the socket to the address of the server using the `connect()` function;
- Send and receive data by means of the `read()` and `write()` functions.

The steps involved in establishing a TCP socket on the server side are as follows:

- Create a socket with the `socket()` function;
- Bind the socket to an address using the `bind()` function;
- Listen for connections with the `listen()` function;
- Accept a connection with the `accept()` function system call. This call typically blocks until a client connects with the server.
- Send and receive data by means of `send()` and `receive()`.

## B) WAP to implement socket programming using UDP. Theory:

Datagram sockets, also known as connectionless sockets, which use User Datagram Protocol (UDP). Stream sockets, also known as connection-oriented sockets, which use Transmission Control Protocol (TCP), Stream Control Transmission Protocol (SCTP) or Datagram Congestion Control Protocol (DCCP).

**UDP properties:** unreliable, packet-switched, packet data, no connection overhead, application-level protocols exchange information immediately, two-way communication.

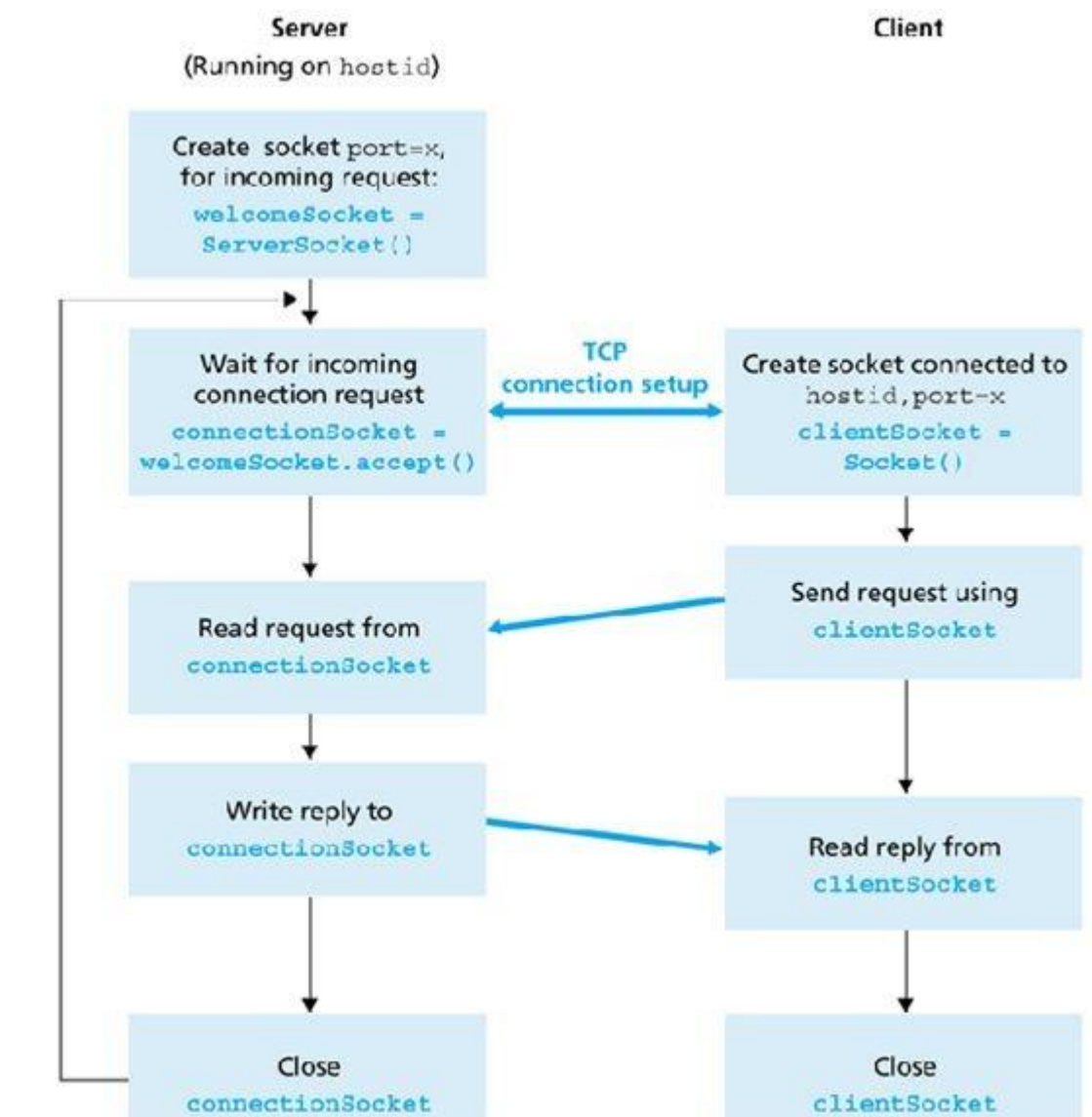


Fig. TCP/IP client/server communication flow

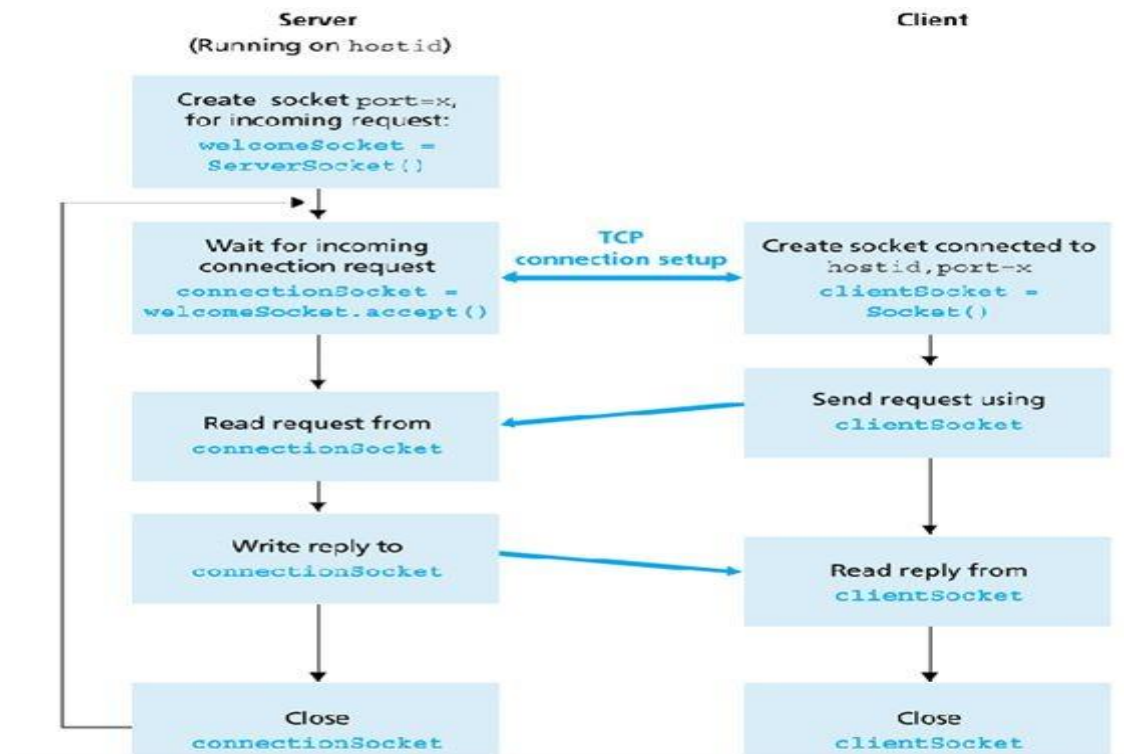


Fig: UDP client/server communication flow:

### TCP/IP Vs UDP :

1. Connection oriented (bidirectional communication sockets which keeps track of connection state, established connection , and also create receiver buffer on both sides of the end to end connection) & other is Connection less (does not keep track of the connection state and it does not have receiver buffer on either side sender or receiver)
2. Will have Acknowledgment & other one don't
3. Performance slow & other one fast
4. More secure & other is not much
5. TCP/IP offer guaranteed delivery while UDP does not
6. TCP/IP consumes high bandwidth. UDP is good guy and shares band with everyone.
7. TCP/IP guarantees sequencing of packets (Packet sent first will reach destination first). In UDP you may get last packet first or not at all.
8. TCP does not have message block boundaries (User has to define its own)
9. TCP can Transmit large amount of data as compared to udp
10. Sequencing of packet is guaranteed in TCP. Means the packets that are sent is delivered in time where in UDP it is not guaranteed that the packets will reach in time to the destination.

## ***Code:***

### **A] TCP**

#### **Server Side:**

```
import java.net.*;
import java.io.*;
class ServerSide
{
public static void main(String[] args) throws Exception
{
int choice,a,b,c=0;
ServerSocket ss = new ServerSocket(1024);
Socket s = ss.accept();
BufferedReader br = new BufferedReader(new InputStreamReader
(s.getInputStream() ) );
choice =Integer.parseInt(br.readLine());
a =Integer.parseInt(br.readLine());
b = Integer.parseInt(br.readLine());
switch(choice)
{
case 1 : c = a+b; break;
case 2 : c = a-b; break;
case 3 : c = a*b; break;
case 4 : c = a/b; break;
case 5 : c = (a%b); break;
}
PrintStream pr = new PrintStream(s.getOutputStream()) ;
pr.println(c);
ss.close();
s.close();
}
}
```

#### **Client Side:**

```
import java.net.*;
import java.io.*;
class ClientSide
{
public static void main(String[] args) throws Exception
{
int ch=0,a,b,c;
Socket s = new Socket("localhost",1024); BufferedReader br =
newBufferedReader(newInputStreamReader(System.in));
PrintStream ps=new PrintStream(s.getOutputStream());
System.out.println("Please Enter Number 1:");
a = Integer.parseInt(br.readLine());
System.out.println("Please Enter Number 2:"); b =
Integer.parseInt(br.readLine());
System.out.println("Please Enter The Operation to Be
Performed\n");
System.out.println("1.Addition 2.Subtraction 3.Multiplication 4.Divison
5.Modulo 0.Exit");
ch = Integer.parseInt(br.readLine());
```

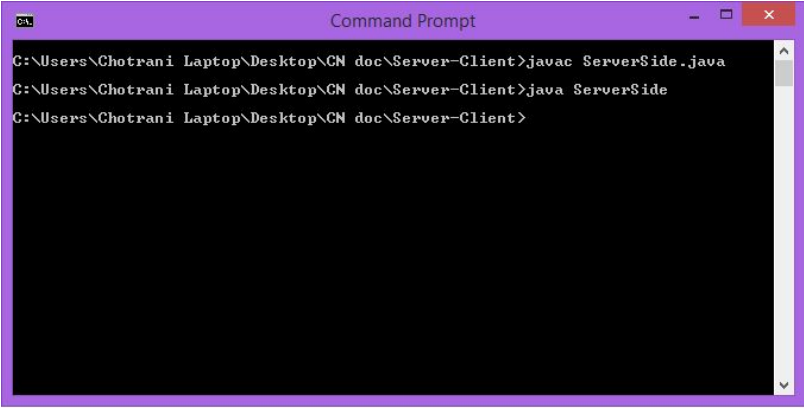
```

ps.println(ch);
ps.println(a);
ps.println(b);
BufferedReader br1 = new BufferedReader(new
InputStreamReader(s.getInputStream()));
c=Integer.parseInt(br1.readLine());
System.out.println("Answer: "+c); s.close();
}
}

```

OUTPUT:

Server Side:

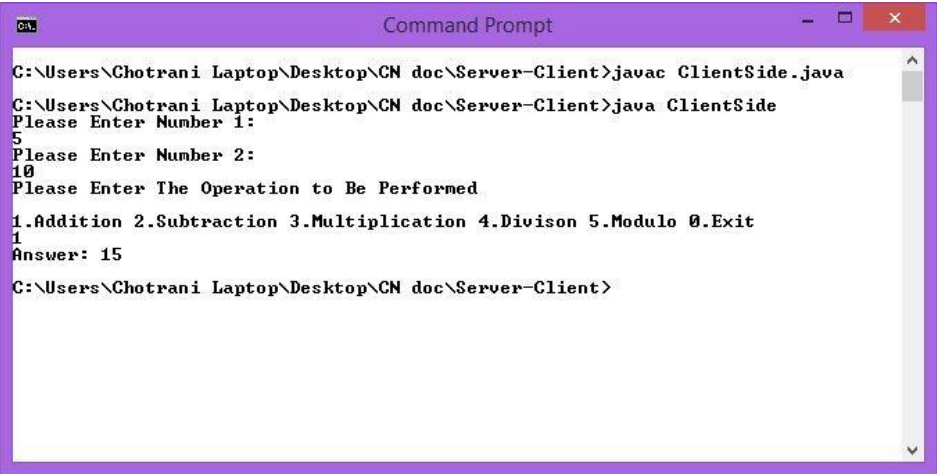


```

C:\Users\Chotrani Laptop\Desktop\CN doc\Server-Client>javac ServerSide.java
C:\Users\Chotrani Laptop\Desktop\CN doc\Server-Client>java ServerSide
C:\Users\Chotrani Laptop\Desktop\CN doc\Server-Client>

```

Client Side:



```

C:\Users\Chotrani Laptop\Desktop\CN doc\Server-Client>javac ClientSide.java
C:\Users\Chotrani Laptop\Desktop\CN doc\Server-Client>java ClientSide
Please Enter Number 1:
5
Please Enter Number 2:
10
Please Enter The Operation to Be Performed
1.Addition 2.Subtraction 3.Multiplication 4.Division 5.Modulo 0.Exit
1
Answer: 15
C:\Users\Chotrani Laptop\Desktop\CN doc\Server-Client>

```

## B] UDP

### Server Socket

#### Program:

```
import java.io.*;
import java.util.*;
import java.net.*;
class ServerUDP
{
    public static void main(String args[])throws IOException
    {
        DatagramSocket ss=new DatagramSocket(2100);
        byte[] sendData=new byte[1024];
        byte[] recData=new byte[1024];
        while(true)
        {
            DatagramPacket dp=new
                DatagramPacket(recData,recData.length);
            ss.receive(dp);
            String input=new String(dp.getData());
            if(input=="end")
                break;
            InetAddress ip=dp.getAddress();
            System.out.println("Received input : "+input);
            String output="Hello ";
            int port=dp.getPort();
            output=output.concat(input);
            sendData=output.getBytes();
            DatagramPacket dp1=new
                DatagramPacket(sendData,sendData.length,ip,port);
            ss.send(dp1);
        }
        ss.close();
    }
}
```

### Client Socket

#### Program:

```
import java.io.*;
import java.util.*;
import java.net.*;
class ClientUDP
{public static void main(String args[])throws IOException
    {
        BufferedReader br=new BufferedReader(new
            InputStreamReader(System.in));
        DatagramSocket ds=new DatagramSocket();
        InetAddress ip=InetAddress.getByName("localhost");
        byte[] sendData=new byte[1024];
        byte[] recData=new byte[1024];
        System.out.println("Enter your name :");
        String input=br.readLine();
        sendData=input.getBytes();
        DatagramPacket dp=new
```



```

        DatagramPacket (sendData, sendData.length, ip, 2100);
        ds.send(dp);
        DatagramPacket dp1=new
        DatagramPacket (recData, recData.length);
        ds.receive(dp1);
        String output=new String(dp1.getData());
        System.out.println(output);
        ds.close();
    }
}

```

## OUTPUT:

**Server Side:**

```

Administrator: C:\Windows\system32\cmd.exe
C:\Users\AGP\Desktop\UDP>javac ServerUDP.java
C:\Users\AGP\Desktop\UDP>java ServerUDP
Received input : Ajay

C:\Users\AGP\Desktop\UDP>_

```

**Client Side:**

```

Administrator: C:\Windows\system32\cmd.exe
C:\Users\AGP\Desktop\UDP>javac ClientUDP.java
C:\Users\AGP\Desktop\UDP>java ClientUDP
Enter your name :
Ajay
Hello Ajay

C:\Users\AGP\Desktop\UDP>_

```

**Conclusion:** Hence we successfully studied and implemented the program of TCP and UDP.

## SIGN AND REMARK

R1 (3 Marks)	R2 (3 Marks)	R3 (3 Marks)	R4 (3 Mark)	R5 (3 Mark)	Total (15 Marks)	Signature

## EXPERIMENT No. 5

**Date of Performance :**

**Date of Submission :**

**Aim:** Perform File Transfer and Access using FTP

**Theory:**

### *Configuration of ftp server*

File Transfer Protocol (FTP) is a TCP protocol for uploading and downloading files between computers. FTP works on a client/server model. The server component is called an *FTP daemon*. It continuously listens for FTP requests from remote clients. When a request is received, it manages the login and sets up the connection. For the duration of the session it executes any of commands sent by the FTP client.

Access to an FTP server can be managed in two ways:

- Anonymous
- Authenticated

In the Anonymous mode, remote clients can access the FTP server by using the default user account called "anonymous" or "ftp" and sending an email address as the password.

In the Authenticated mode a user must have an account and a password. User access to the FTP server directories and files is dependent on the permissions defined for the account used at login. As a general rule, the FTP daemon will hide the root directory of the FTP server and change it to the FTP Home directory. This hides the rest of the file system from remote sessions.

### *Steps*

1. vsftpd is an FTP daemon available in Ubuntu. To install **vsftpd** we can run the following command in root mode:

```
student@lab307-02:~$ sudo su
[sudo] password for student:
root@lab307-02:/home/student# sudo apt-get install vsftpd
```

2. User authenticated FTP Configuration

To configure **vsftpd** to authenticate system users and allow them to upload files. To make a copy of the original config file, we use cp command. Configure vsftpd using gedit.

```

root@lab307-02:~# cp /etc/vsftpd.conf /etc/vsftpd.conf.original
root@lab307-02:~# gedit /etc/vsftpd.conf

(gedit:4067): Gtk-WARNING **: Calling Inhibit failed: GDBus.Error:org.freedesktop.DBus.Error.ServiceUnknown: The name org.gnome.SessionManager was not provided by any .service files

(gedit:4067): Gtk-WARNING **: Calling Inhibit failed: GDBus.Error:org.freedesktop.DBus.Error.ServiceUnknown: The name org.gnome.SessionManager was not provided by any .service files

```

Uncomment the following lines in “vsftpd.conf” configuration file

```

# Uncomment this to allow local users to log in.
local_enable=YES

#

# Uncomment this to enable any form of FTP write command.
write_enable=YES

#

```

3. Now restart vsftpd

```

root@lab307-02:~# /etc/init.d restart
-bash: /etc/init.d: Is a directory
root@lab307-02:~# █

```

To start or stop the service use command:

```
#service vsftpd start
```

```
#service vsftpd stop
```

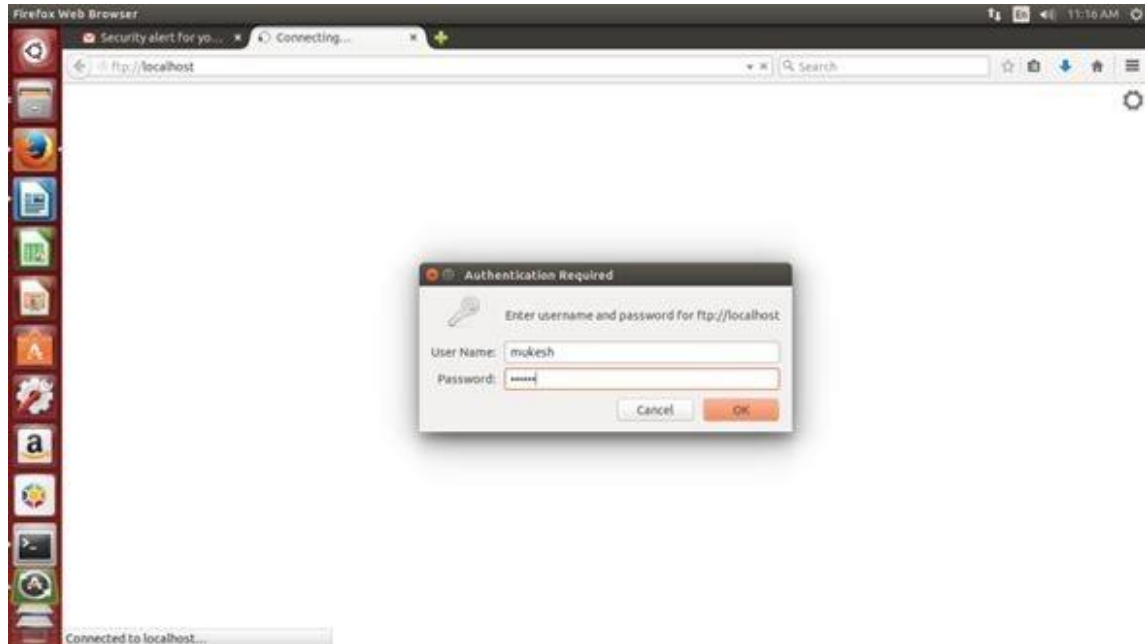
4. add new user

```

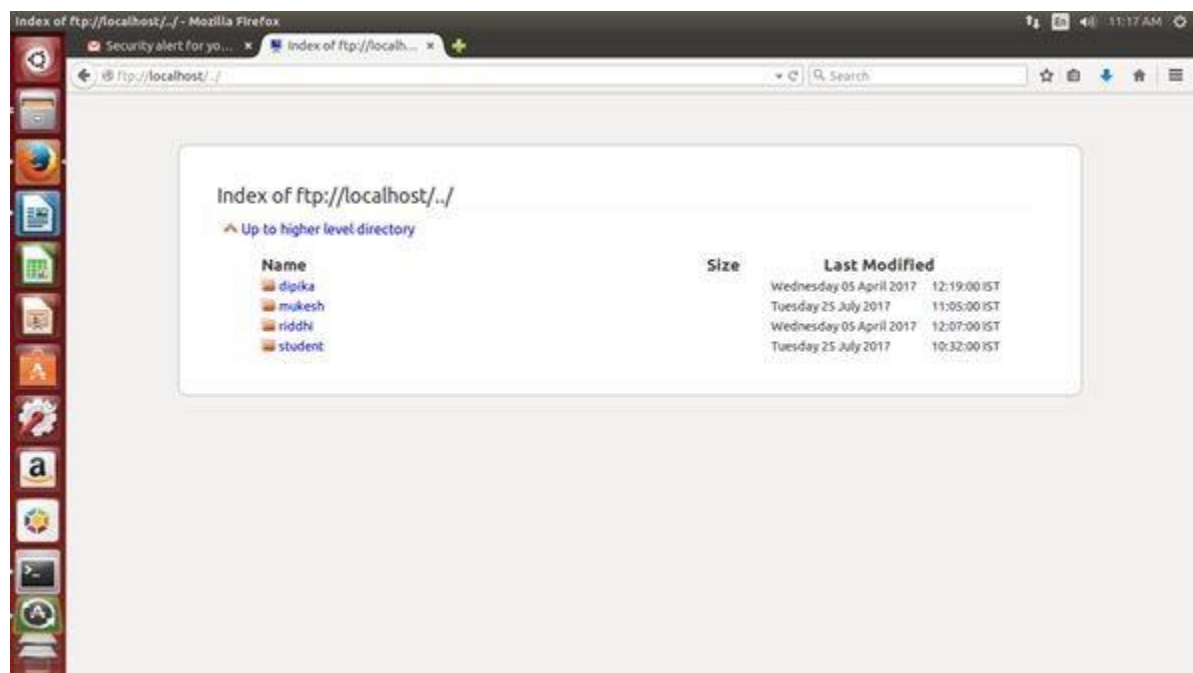
root@lab307-02:~# adduser mukesh
Adding user `mukesh' ...
Adding new group `mukesh' (1004) ...
Adding new user `mukesh' (1004) with group `mukesh' ...
Creating home directory `/home/mukesh' ...
Copying files from `/etc/skel' ...
Enter new UNIX password:
Retype new UNIX password:
passwd: password updated successfully
Changing the user information for mukesh
Enter the new value, or press ENTER for the default
    Full Name []: Mukesh Yadav
    Room Number []: 301
    Work Phone []:
    Home Phone []:
    Other []:
Is the information correct? [Y/n] y
root@lab307-02:~#

```

5. In your browser type: <ftp://localhost> and enter login username and password.



6. After authentication, we can transfer and access files using ftp



Some common ftp commands:

?	to request help or information about the FTP commands
ascii	to set the mode of file transfer to ASCII

binary	to set the mode of file transfer to binary
bye	to exit the FTP environment (same as quit)
cd	to change directory on the remote machine
close	to terminate a connection with another computer
delete	to delete (remove) a file in the current remote directory (same as rm in UNIX)
get	<p>to copy one file from the remote machine to the local machine</p> <p><b>get ABC DEF</b></p> <p>This copies file ABC in the current remote directory to (or on top of) a file named DEF in your current local directory.</p> <p><b>get ABC</b></p> <p>This copies file ABC in the current remote directory to (or on top of) a file with the same name, ABC, in your current local directory.</p>
help	to request a list of all available FTP commands
lcd	to change directory on your local machine (same as UNIX cd)
ls	to list the names of the files in the current remote directory
mkdir	to make a new directory within the current remote directory
mget	<p>to copy multiple files from the remote machine to the local machine;</p> <p>you are prompted for a y/n answer before transferring each file</p> <p><b>mget *</b></p> <p>This copies all the files in the current remote directory to your current local directory, using the same filenames. Notice the use of the wild card character, *.</p>
mput	<p>to copy multiple files from the local machine to the remote machine;</p> <p>you are prompted for a y/n answer before transferring each file</p>
open	to open a connection with another computer
put	to copy one file from the local machine to the remote machine



pwd	to find out the pathname of the current directory on the remote machine
quit	to exit the FTP environment (same as bye)
rmdir	to to remove (delete) a directory in the current remote directory

```

administrator@ubuntu:~$ ftp
ftp> exit
administrator@ubuntu:~$ ftp metalab.unc.edu
Connected to www.ibiblio.org.
220 ProFTPD Server
Name (metalab.unc.edu:administrator): anonymous
331 Anonymous login ok, send your complete email address as your password
Password:
230-
    Welcome to ftp.ibiblio.org, the public ftp server of ibiblio.org. We
    hope you find what you're looking for.

    If you have any problems or questions, please see

    http://www.ibiblio.org/help/

    Thanks!

230 Anonymous access granted, restrictions apply
Remote system type is UNIX.
Using binary mode to transfer files.
ftp> ls
200 PORT command successful
150 Opening BINARY mode data connection for file list
-rw-r--r--  1 root    root      2897 Jan 26  2006 HEADER.html
drwxr-xr-x  2 root    root      4096 Jan 26  2006 HEADER.images
drwxr-xr-x  4 root    root      4096 Jan 18  2013 incoming
drwxr-xr-x 14 root    root      4096 Apr 14  2011 pub
-rw-r--r--  1 root    root       212 Apr  6  2011 README
drwxr-xr-x  4 root    root      4096 Dec  1  2004 unc
226 Transfer complete
ftp> cd pub
250 CWD command successful
ftp> ls
200 PORT command successful
150 Opening BINARY mode data connection for file list
drwxr-xr-x 31 root    root      4096 Feb  6  2004 academic
drwxr-xr-x  8 root    bin       4096 Feb 23  2012 archives
drwxr-xr-x 29 root    root      4096 May 24  2011 docs
drwxr-xr-x 12 root    bin       4096 Mar 23  2011 electronic-publications
drwxr-xr-x  5 root    root      4096 Apr 30  2012 historic-linux
drwxr-xr-x  7 root    root      4096 Feb 13  2006 languages

```

**Output:**

```

ftp> cd docs
250 CWD command successful
ftp> ls
200 PORT command successful
150 Opening BINARY mode data connection for file list
drwxr-xr-x 13 root root 4096 Jan 26 2005 about-the-net
drwxr-xr-x 2 root bin 4096 Dec 9 1993 ACCESS-bus
drwxr-xr-x 8 root bin 4096 Jan 31 2005 books
drwxr-xr-x 2 root root 4096 Jan 31 2005 concert
drwxrwxr-x 2 523 bin 4096 Aug 21 1998 exploris
drwxr-xr-x 2 root bin 4096 Mar 13 1996 faqs
drwxr-xr-x 14 893 bin 4096 Dec 10 1999 filesystems
drwxr-xr-x 2 942 bin 4096 Dec 9 1999 freefilm
drwxr-xr-x 6 root bin 4096 Jan 31 2005 humor
drwxr-xr-x 2 root bin 4096 Jan 31 2005 lafa
drwxr-xr-x 2 root root 4096 Jan 31 2005 ietf
-rw-r--r-- 1 root bin 1208 Mar 25 1994 INDEX
drwxr-xr-x 2 root root 4096 Jan 31 2005 lib-catalogs.online
drwxr-xr-x 2 root bin 4096 Jan 31 2005 local-area-networks
drwxr-xr-x 2 root bin 4096 Jan 31 2005 mail-directories
drwxr-xr-x 3 root root 4096 Jan 31 2005 misc
drwxr-xr-x 41 root root 4096 Jan 30 1995 nc-supreme-court
drwxr-xr-x 2 root root 20480 Jan 31 2005 nih-nsf
-rw-r--r-- 1 root root 595 Jan 8 1993 README
drwxr-xr-x 2 237 bin 4096 Aug 14 2000 recipes
drwxr-xr-x 4 2358 users 4096 Nov 15 2007 resource.org
drwxr-xr-x 66 root bin 45056 Jan 26 1996 rfc
drwxr-xr-x 2 root root 4096 Jan 31 2005 security
drwxr-xr-x 2 root bin 4096 Jan 31 2005 services
drwxr-xr-x 2 704 root 4096 May 13 2003 typing-injury
drwxr-xr-x 2 root root 4096 Jan 31 2005 UNC-bbs
drwxr-xr-x 3 root root 4096 Jan 31 2005 unix-tutorials
drwxr-xr-x 2 root bin 4096 Jan 31 2005 vms-tutorials
226 Transfer complete
ftp> cd security
250 CWD command successful
ftp> ls
200 PORT command successful
150 Opening BINARY mode data connection for file list
-rw-r--r-- 1 root root 104328 Apr 21 1992 gao-report
-rw-r--r-- 1 root root 16307 Apr 21 1992 inet.worm
-rw-r--r-- 1 root bin 52202 Feb 22 1993 pkt_filtering.ps.Z
-rw-r--r-- 1 root root 58441 Apr 21 1992 security-doc.tar.Z
-rw-r--r-- 1 root root 249218 Apr 21 1992 security-draft-26nov.txt
226 Transfer complete
ftp> cd security

```

**Conclusion:** Hence we successfully studied the program of FTP.

## SIGN AND REMARK

R1 (3 Marks)	R2 (3 Marks)	R3 (3 Marks)	R4 (3 Mark)	R5 (3 Mark)	Total (15 Marks)	Signature

## EXPERIMENT No. 6

Date of Performance :

Date of Submission :

**Aim :** Install and Use Telnet in Ubuntu.

### Theory:

The telnet command is used for interactive communication with another host using the TELNET protocol. It begins in command mode, where it prints a telnet prompt ("telnet> "). If telnet is invoked with a host argument, it performs an open command implicitly; see the description below.

Options:

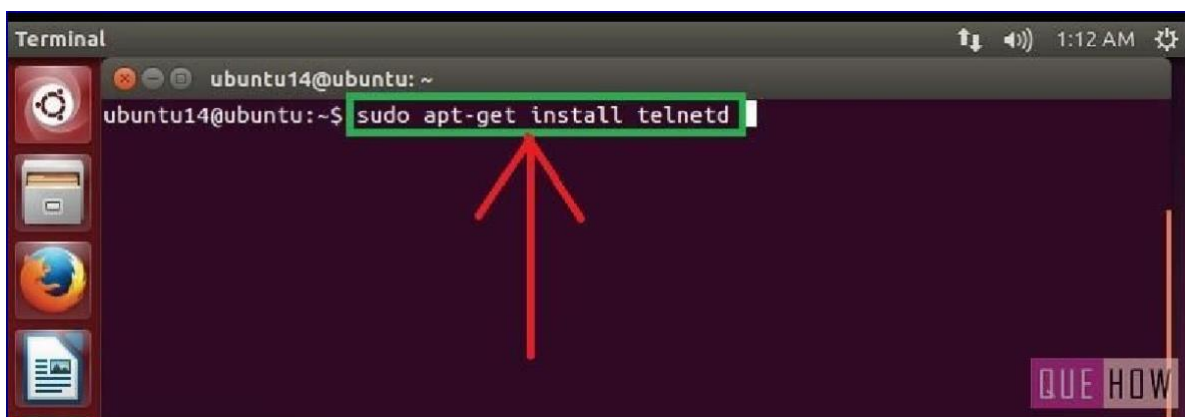
- 4 Force IPv4 address resolution.
- 6 Force IPv6 address resolution.
- 8 Request 8-bit operation. This causes an attempt to negotiate the TELNET BINARY option for both input and output. By default telnet is not 8-bit clean.
- E Disables the escape character functionality; that is, sets the escape character to ``no character".
- K Specifies no automatic login to the remote system.
- L Specifies an 8-bit data path on output. This causes the TELNET BINARY option to negotiated on just output.

Once a connection has been opened, **telnet** will attempt to enable the TELNET LINEMODE option. If this fails, then **telnet** will revert to one of two input modes: either "character at a time" or "old line by line" depending on what the remote system supports

### Steps to Install and Use Telnet in Ubuntu:

**Step 1:** Firstly, open the "Terminal" window by pressing "Ctrl + Alt + T". In the figure, you may see "\$" that signifies that you are not logged in as a root user.

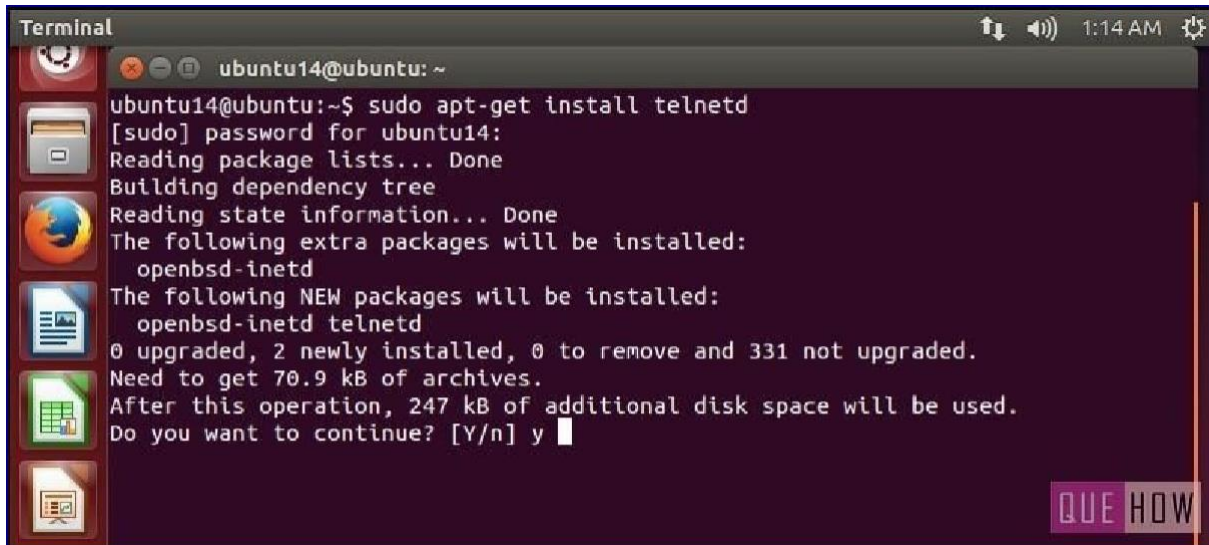
So, I'll write "**sudo apt-get install telnetd**" and press enter. If you are a root user, then you don't need to write sudo in Ubuntu. "**telnetd**" is a daemon that gets invoked by "*inetd*" or its extension "*xinetd*", both are the internet servers.





**Step 2:** Then you are asked to enter the user password and then press enter. Processing will start as soon as you press enter. After this, I have noticed a line “**274 KB additional disk space will be used**” on the terminal screen.

You may also observe some sort of a message like this and then you’ll be asked to continue or not. Just write “y” and then press enter to continue.

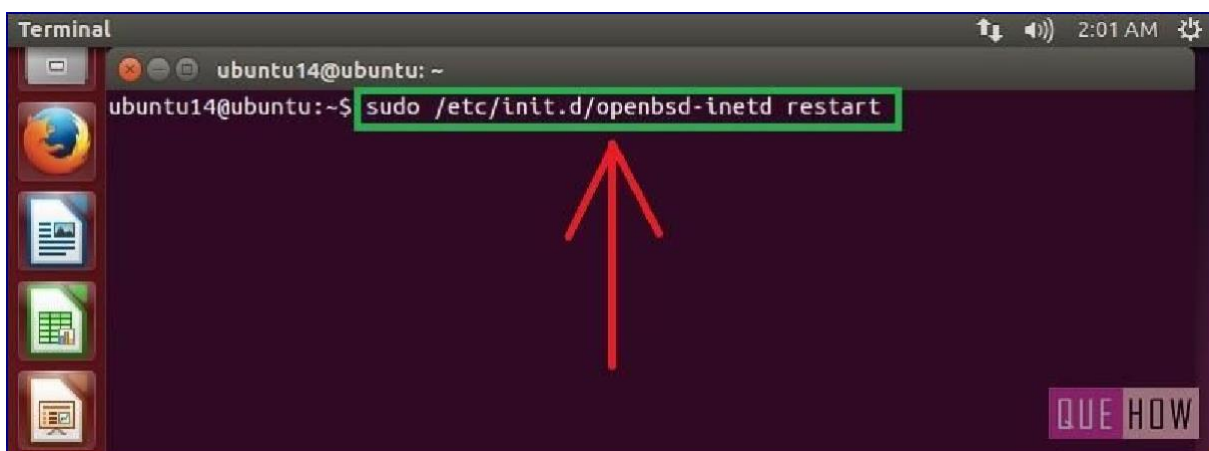
A terminal window titled 'Terminal' showing the command 'sudo apt-get install telnetd'. The output shows the password prompt, package lists being read, and the dependency tree being built. It lists 'openbsd-inetd' as an extra package to be installed along with 'telnetd'. It states that 0 packages will be upgraded, 2 newly installed, and 331 not upgraded, requiring 70.9 kB of archives. The final line indicates that 247 kB of additional disk space will be used and asks for confirmation to continue, with 'y' entered.

```
Terminal
ubuntu14@ubuntu: ~
ubuntu14@ubuntu:~$ sudo apt-get install telnetd
[sudo] password for ubuntu14:
Reading package lists... Done
Building dependency tree
Reading state information... Done
The following extra packages will be installed:
  openbsd-inetd
The following NEW packages will be installed:
  openbsd-inetd telnetd
0 upgraded, 2 newly installed, 0 to remove and 331 not upgraded.
Need to get 70.9 kB of archives.
After this operation, 247 kB of additional disk space will be used.
Do you want to continue? [Y/n] y
```

**Step 3:** Now when you are done with it, **restart “inetd”**.

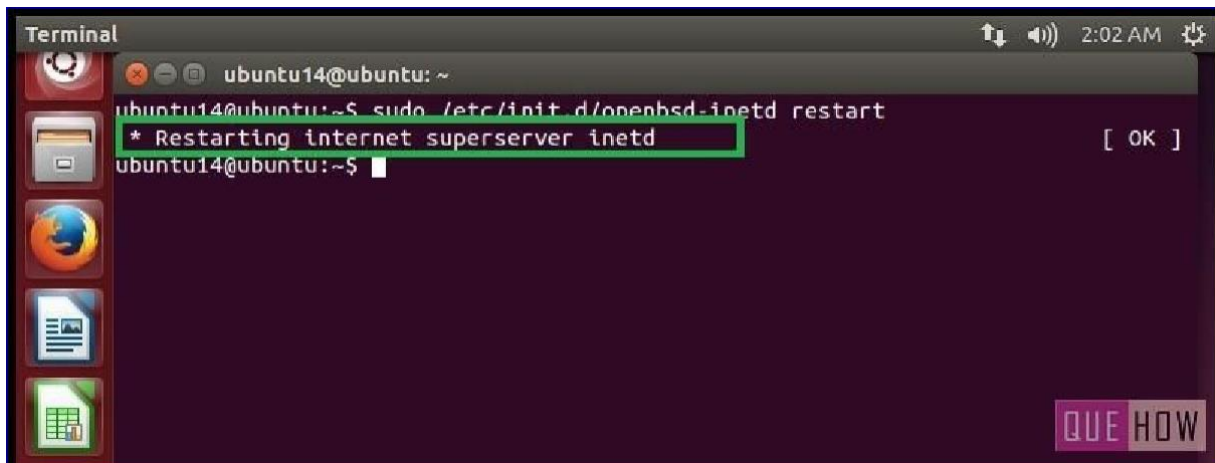
Type “**sudo /etc/init.d.open-bsd-inetd restart**”.

“*inetd*” is daemon used for *dealing with incoming network* and it is responsible for deciding which program to run when a request comes.

A terminal window titled 'Terminal' showing the command 'sudo /etc/init.d/openbsd-inetd restart'. The command is highlighted with a green box, and a red arrow points to it from below.

```
Terminal
ubuntu14@ubuntu: ~
ubuntu14@ubuntu:~$ sudo /etc/init.d/openbsd-inetd restart
```

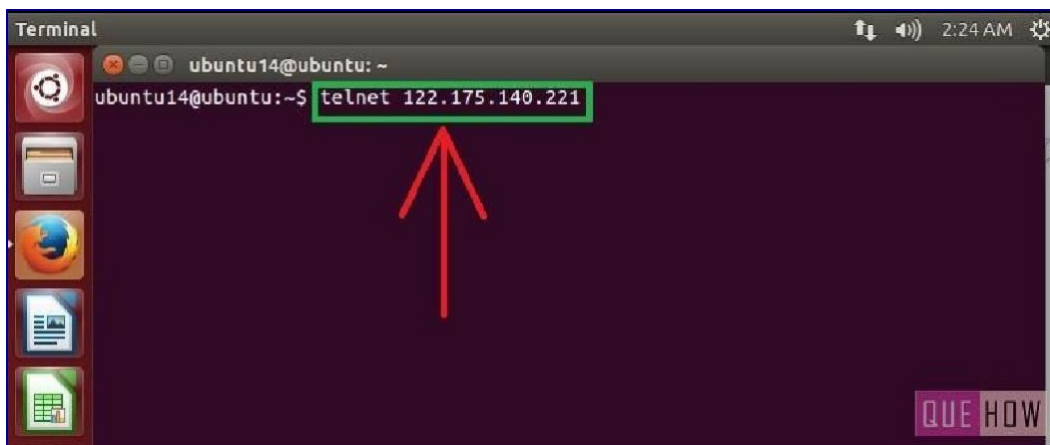
**Step 4:** To ensure “inetd” is started, press enter after writing the above command.

A terminal window titled 'Terminal' with a dark purple background. The prompt is 'ubuntu14@ubuntu: ~'. The command 'sudo /etc/init.d/openbsd-inetd restart' has been entered. The output shows '\* Restarting internet superserver inetd' followed by a green box around the asterisk and the text. A '[ OK ]' message is visible on the right. The system clock in the top right corner shows '2:02 AM'.

```
Terminal
ubuntu14@ubuntu: ~
ubuntu14@ubuntu:~$ sudo /etc/init.d/openbsd-inetd restart
* Restarting internet superserver inetd
ubuntu14@ubuntu:~$
```

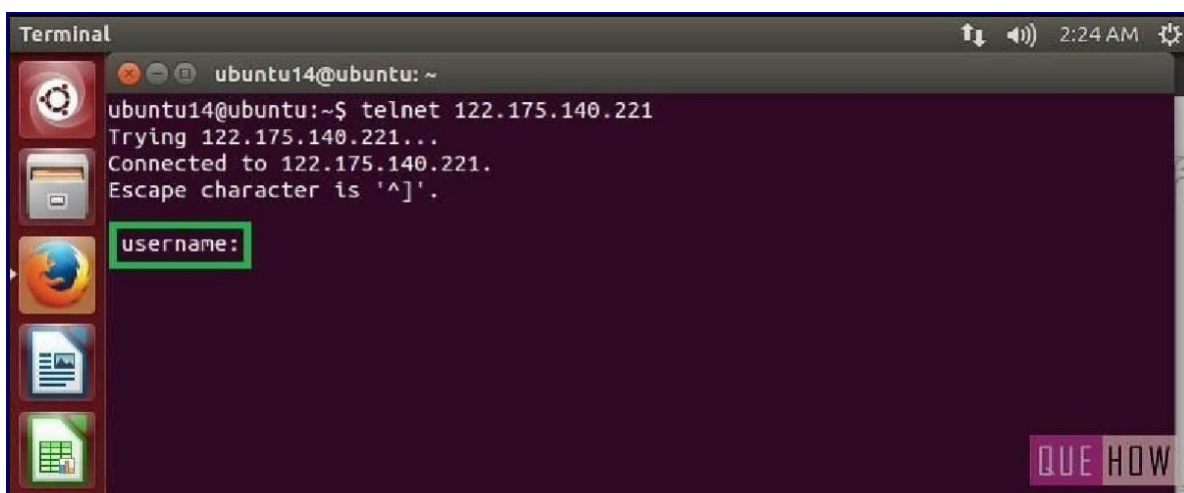
*To connect with any remote client:*

**Step 5:** Just type: “**telnet hostipaddress**”. For an example: “telnet 122.175.140.221” and press enter.

A terminal window titled 'Terminal' with a dark purple background. The prompt is 'ubuntu14@ubuntu: ~'. The command 'telnet 122.175.140.221' has been entered. A red arrow points to the IP address '122.175.140.221'. The system clock in the top right corner shows '2:24 AM'.

```
Terminal
ubuntu14@ubuntu: ~
ubuntu14@ubuntu:~$ telnet 122.175.140.221
```

**Step 6:** Then you’ll see, it is connected to “**host ip address**”. For security reasons, you are required to provide “username” and “password” as well.

A terminal window titled 'Terminal' with a dark purple background. The prompt is 'ubuntu14@ubuntu: ~'. The command 'telnet 122.175.140.221' has been entered. The output shows 'Trying 122.175.140.221...', 'Connected to 122.175.140.221.', and 'Escape character is '^]'. The prompt 'username:' is displayed with a green box around it. The system clock in the top right corner shows '2:24 AM'.

```
Terminal
ubuntu14@ubuntu: ~
ubuntu14@ubuntu:~$ telnet 122.175.140.221
Trying 122.175.140.221...
Connected to 122.175.140.221.
Escape character is '^]'.
username:
```

**Conclusion:** Hence we successfully studied the program of telnet.

**SIGN AND REMARK**

<b>R1 (3 Marks)</b>	<b>R2 (3 Marks)</b>	<b>R3 (3 Marks)</b>	<b>R4 (3 Mark)</b>	<b>R5 (3 Mark)</b>	<b>Total (15 Marks)</b>	<b>Signature</b>

## EXPERIMENT No. 7

**Date of Performance :**

**Date of Submission :**

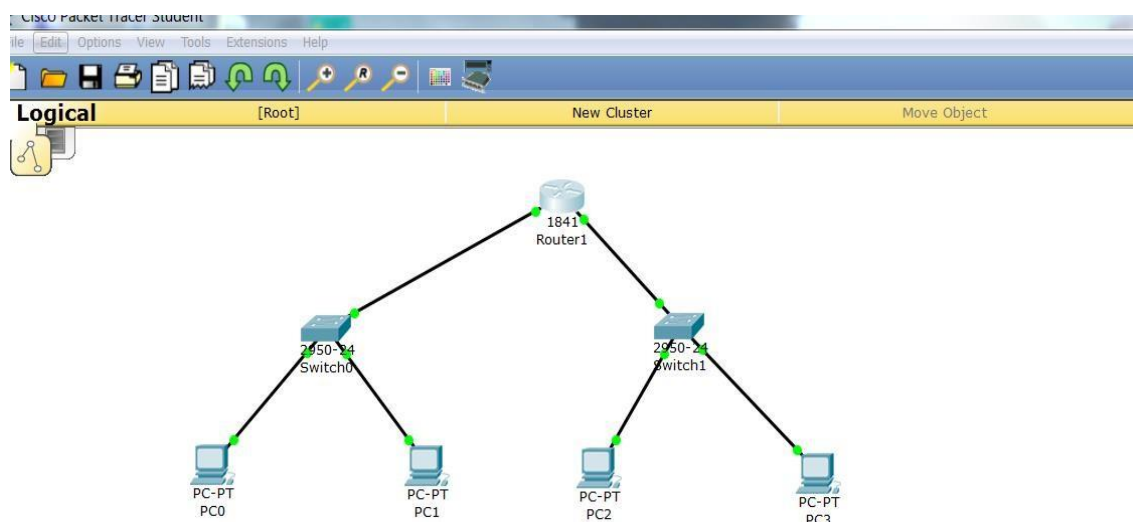
**AIM:** Build a simple network topology and configure it for static routing protocol using packet tracer. Setup a network and configure IP addressing, subnetting, masking.

**THEORY:** Cisco Packet Tracer is a cross-platform visual simulation tool designed by Cisco Systems that allows users to create network topologies and imitate modern computer networks. The software allows users to simulate the configuration of Cisco routers and switches using a simulated command line interface. Packet Tracer makes use of a drag and drop user interface, allowing users to add and remove simulated network devices as they see fit. The software is mainly focused towards Certified Cisco Network Associate Academy students as an educational tool for helping them learn fundamental CCNA concepts.

### Steps:

1. Pick a total of 4 pcs in the packet tracer application.
2. We need 2 routers.
3. We need a single router.

**Connect the devices as shown below:**



4. Give the appropriate IP addresses to the pcs accordingly.
5. Test the network with the help of packets.

**CONCLUSION:** Hence we have successfully created simple network using

**SIGN AND REMARK**

<b>R1</b> <b>(3 Marks)</b>	<b>R2</b> <b>(3 Marks)</b>	<b>R3</b> <b>(3 Marks)</b>	<b>R4</b> <b>(3 Mark)</b>	<b>R5</b> <b>(3 Mark)</b>	<b>Total</b> <b>(15 Marks)</b>	<b>Signature</b>

## **EXPERIMENT No. 8**

**Date of Performance :**

**Date of Submission :**

**AIM:** Study and Installation of Network Simulator (NS3)

**Theory:**

The *ns-3* simulator is a discrete-event network simulator targeted primarily for research and educational use. The [ns-3 project](#), started in 2006, is an open-source project developing *ns-3*.

The purpose of this tutorial is to introduce new *ns-3* users to the system in a structured way. It is sometimes difficult for new users to glean essential information from detailed manuals and to convert this information into working simulations. In this tutorial, we will build several example simulations, introducing and explaining key concepts and features as we go.

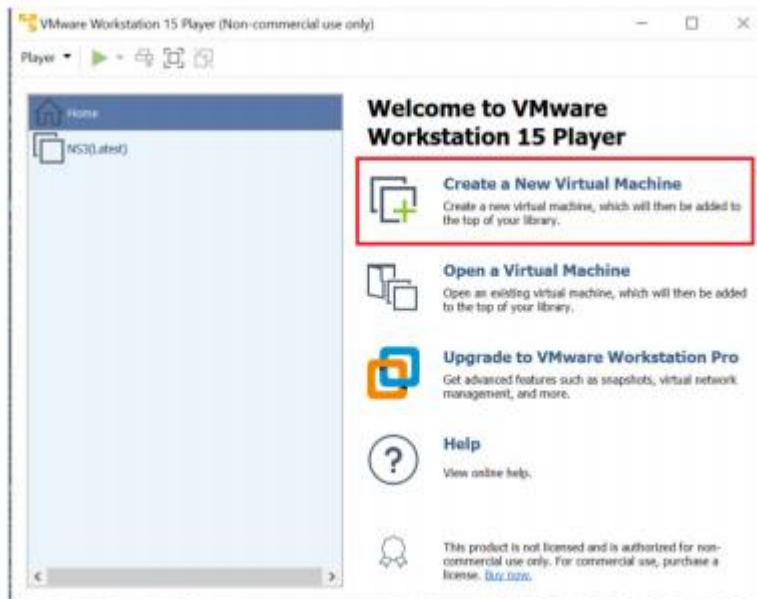
As the tutorial unfolds, we will introduce the full *ns-3* documentation and provide pointers to source code for those interested in delving deeper into the workings of the system.

A few key points are worth noting at the onset:

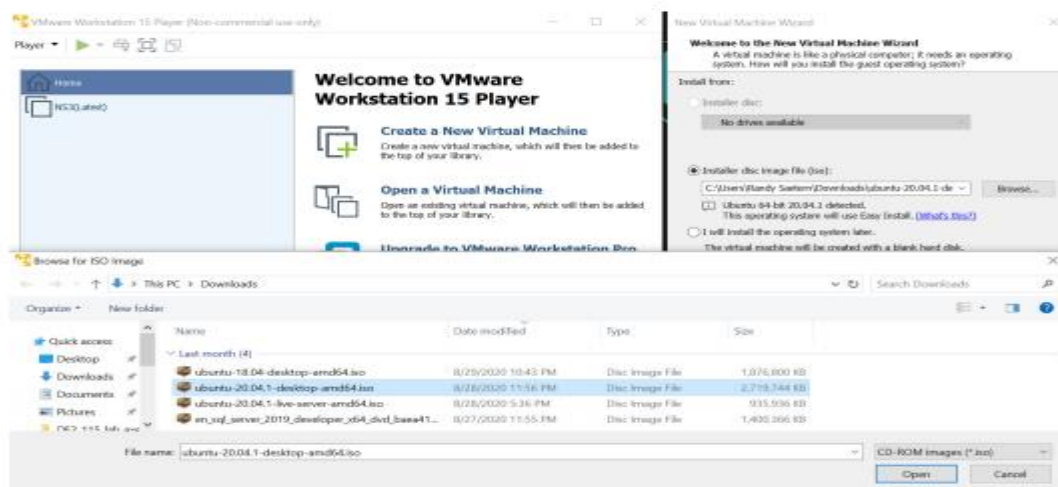
- *ns-3* is open-source, and the project strives to maintain an open environment for researchers to contribute and share their software.
- *ns-3* is not a backwards-compatible extension of [ns-2](#); it is a new simulator. The two simulators are both written in C++ but *ns-3* is a new simulator that does not support the *ns-2* APIs.

**For the installation of NS3, VMware workstation is required to be installed, along with an Ubuntu system.**

1. Download VMWare workstation from the website:  
[https://my.vmware.com/en/web/vmware/downloads/info/slug/desktop\\_end\\_user\\_computing/vmware\\_workstation\\_player/15\\_0](https://my.vmware.com/en/web/vmware/downloads/info/slug/desktop_end_user_computing/vmware_workstation_player/15_0)
2. Download Ubuntu 20.04.01 Desktop AMD 64 from the website:  
<https://ubuntu.com/download/desktop>
3. Install VMWare workstation onto the computer system and open it
4. Set up the VMWare workstation: a. Create a new virtual machine by selecting “Create New Virtual Machine.”



- a. In the installer wizard, select installer disc image file(iso) and select the downloaded Ubuntu 20.04.01 AMD 64 iso file by browsing through the computer download files.

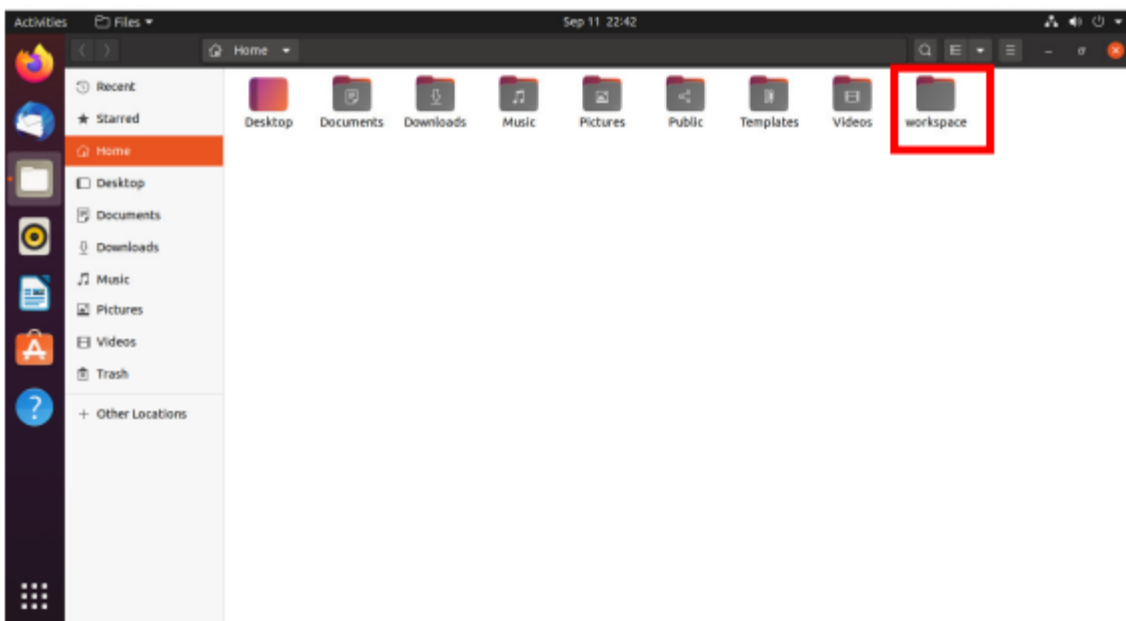


- c. Name the machine and set the password.
- d. Configure the Hardware:
  - i. For memory: set the value to 4600 MB or above.
  - ii. For faster VMware, set processors to 2.
5. Power on the virtual machine and let the machine update.
6. Within the Virtual machine, download NS3 on the VM by opening Mozilla firefox and downloading from the NS3 website.
7. Install prereq packages on Ubuntu using terminal:
  - a. Open the terminal by right clicking the desktop and select “open in terminal.”
  - b. Paste in this code and then press enter: `sudo apt-get install g++ python3 python3-dev pkg-config sqlite3 python3- setuptools git qt5-default mercurial gir1.2-gocanvas-2.0 python-gi`

python-gi-cairo python3-gi python3-gi-cairo python3-pygraphviz gir1.2-gtk-3.0 ipython3  
 openmpi-bin openmpi-common openmpi-doc libopenmpi-dev autoconf cvs bzip2 unrar gdb  
 valgrind uncrustify doxygen graphviz imagemagick texlive texlive-extra-utils texlive-latex-extra  
 texlive-font-utils dvipng latexmk python3-sphinx dia gsl-bin libgsl-dev libgsl23 libgslcblas0  
 tcpdump sqlite sqlite3 libsqlite3-dev libxml2 libxml2-dev cmake libc6-dev libc6-dev-i386  
 libclang-6.0-dev llvm-6.0-dev automake python3-pip libgtk-3-dev synaptic vtun lxc uml-utilities

c. After the packages have finished downloading, paste in this code and press enter: `sudo pip3 install cxxfilt`

8. After installing the required packages, create a folder named workspace in the home directory and then put the NS3 tar package into the workspace. See example figure below.



9. Go to terminal and input these commands consecutively after each command finishes executing:

```
cd
cd workspace
tar xjf <Name of Ns3 downloaded file name>
cd <Name of extracted Ns3>
./build.py --enable-examples --enable-tests
```

10. Test the NS3 build and installation success by running test.py in the ns directory using the following commands: `cd ns- ./test.py`



```
randyns3@ubuntu: ~/workspace/ns-allinone-3.31/ns-3.31
randyns3@ubuntu:~/workspace/ns-allinone-3.31$ ls
bake      constants.py  ns-3.31      __pycache__  util.py
build.py  netanim-3.108 pybindgen-0.21.0 README
randyns3@ubuntu:~/workspace/ns-allinone-3.31$ cd ns-3.31/
randyns3@ubuntu:~/workspace/ns-allinone-3.31/ns-3.31$ ./test.py
```

11. If all of the tests were passed, Congratulations! NS3 has now been installed successfully.

**CONCLUSION:** Thus, we have studied and successfully install NS3.

**SIGN AND REMARK**

R1 (3 Marks)	R2 (3 Marks)	R3 (3 Marks)	R4 (3 Mark)	R5 (3 Mark)	Total (15 Marks)	Signature

# EXPERIMENT No. 9

**Date of Performance :**

**Date of Submission :**

AIM: Use Wire shark to understand the operation of TCP/IP layers:

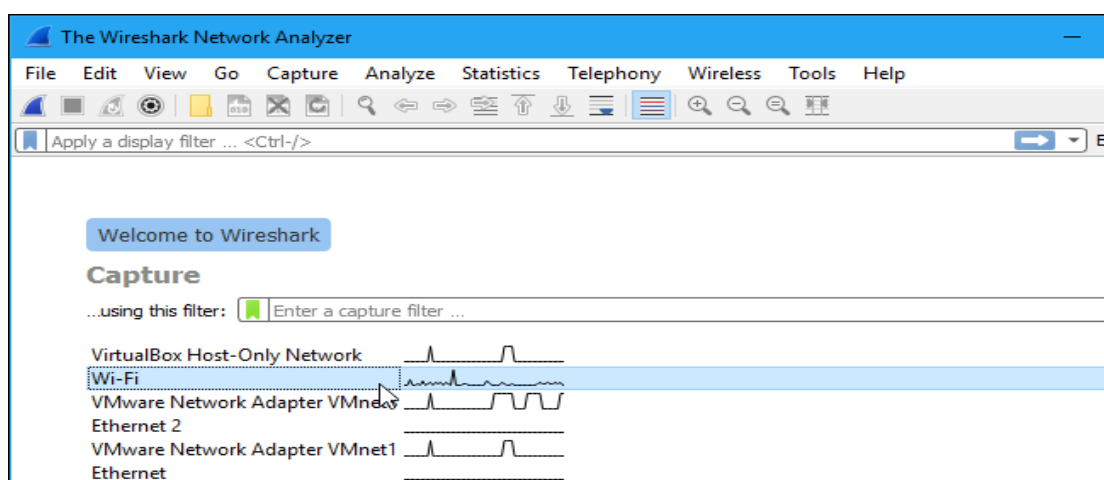
- Ethernet Layer: Frame header, Frame size etc.
- Data Link Layer: MAC address, ARP (IP and MAC address binding)
- Network Layer: IP Packet (header, fragmentation), ICMP (Query and Echo)
- Transport Layer: TCP Ports, TCP handshake segments etc.
- Application Layer: DHCP, FTP, HTTP header formats

## THEORY:

Wireshark, a network analysis tool formerly known as Ethereal, captures packets in real time and display them in human-readable format. Wireshark includes filters, color coding, and other features that let you dig deep into network traffic and inspect individual packets.

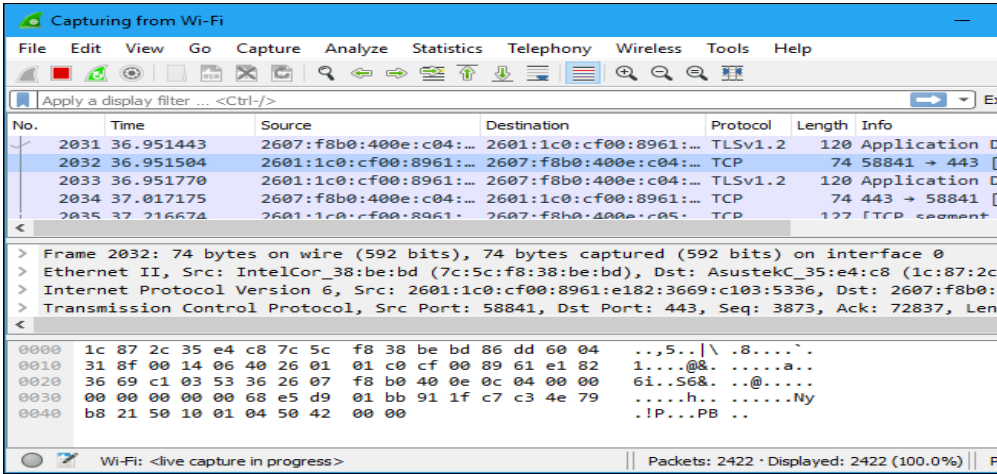
### *Capturing Packets*

After downloading and installing Wireshark, you can launch it and double-click the name of a network interface under Capture to start capturing packets on that interface. For example, if you want to capture traffic on your wireless network, click your wireless interface. You can configure advanced features by clicking Capture > Options, but this isn't necessary for now.

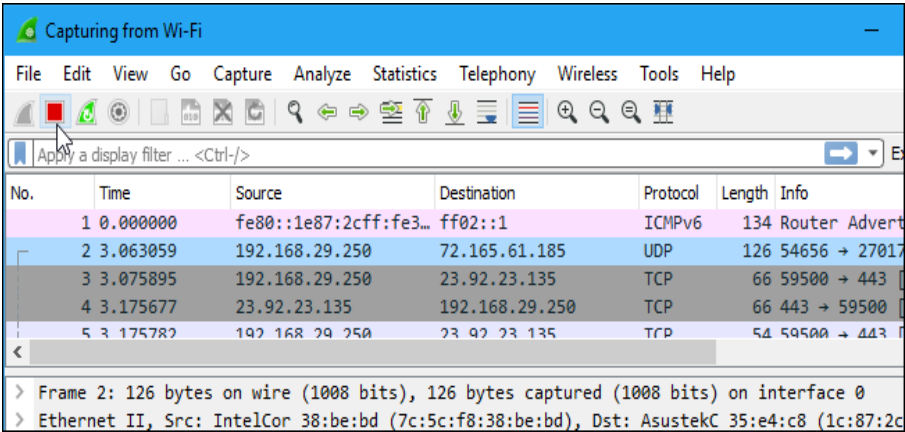


As soon as you click the interface's name, you'll see the packets start to appear in real time. Wireshark captures each packet sent to or from your system.

If you have promiscuous mode enabled—it’s enabled by default—you’ll also see all the other packets on the network instead of only packets addressed to your network adapter. To check if promiscuous mode is enabled, click Capture > Options and verify the “Enable promiscuous mode on all interfaces” checkbox is activated at the bottom of this window.



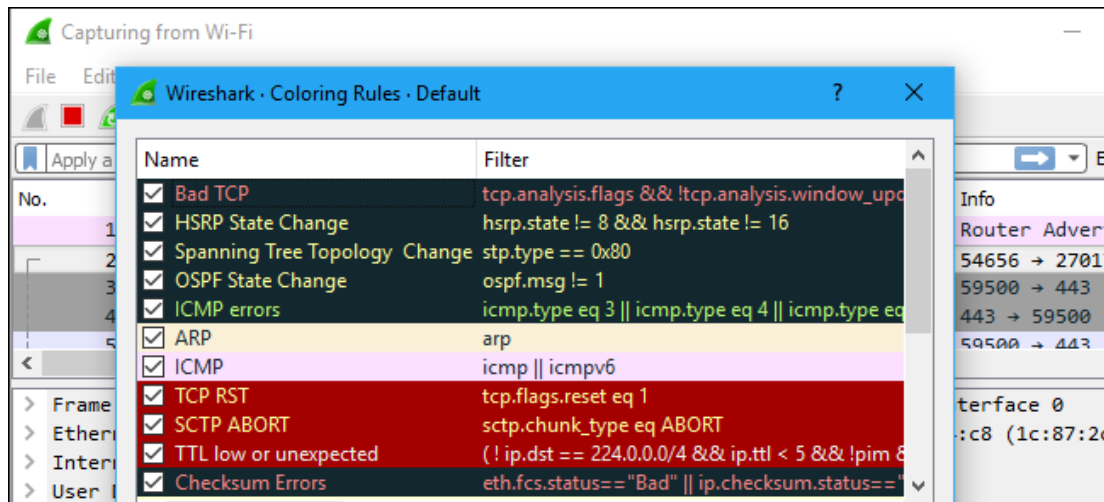
Click the red “Stop” button near the top left corner of the window when you want to stop capturing traffic.



*Color Coding*

You’ll probably see packets highlighted in a variety of different colors. Wireshark uses colors to help you identify the types of traffic at a glance. By default, light purple is TCP traffic, light blue is UDP traffic, and black identifies packets with errors—for example, they could have been delivered out of order.

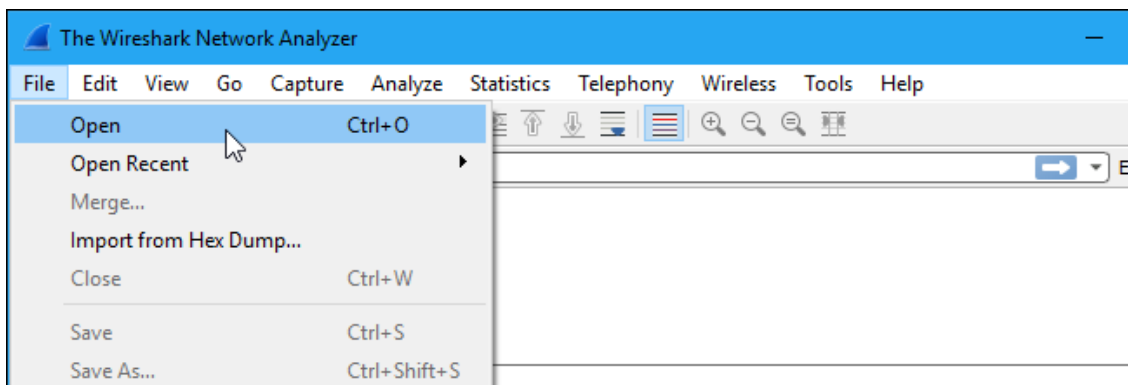
To view exactly what the color codes mean, click View > Coloring Rules. You can also customize and modify the coloring rules from here, if you like.



## Sample Captures

If there's nothing interesting on your own network to inspect, Wireshark's wiki has you covered. The wiki contains a [page of sample capture files](#) that you can load and inspect. Click File > Open in Wireshark and browse for your downloaded file to open one.

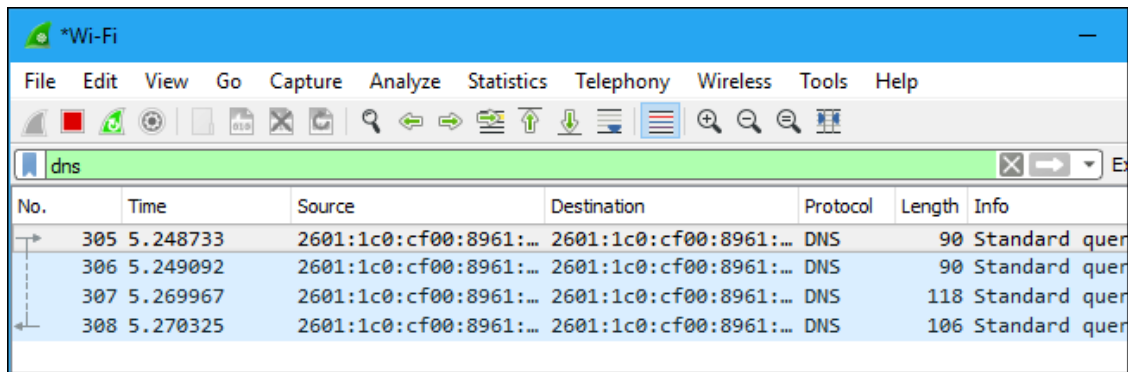
You can also save your own captures in Wireshark and open them later. Click File > Save to save your captured packets.



## Filtering Packets

If you're trying to inspect something specific, such as the traffic a program sends when phoning home, it helps to close down all other applications using the network so you can narrow down the traffic. Still, you'll likely have a large amount of packets to sift through. That's where Wireshark's filters come in.

The most basic way to apply a filter is by typing it into the filter box at the top of the window and clicking Apply (or pressing Enter). For example, type "dns" and you'll see only DNS packets. When you start typing, Wireshark will help you autocomplete your filter.

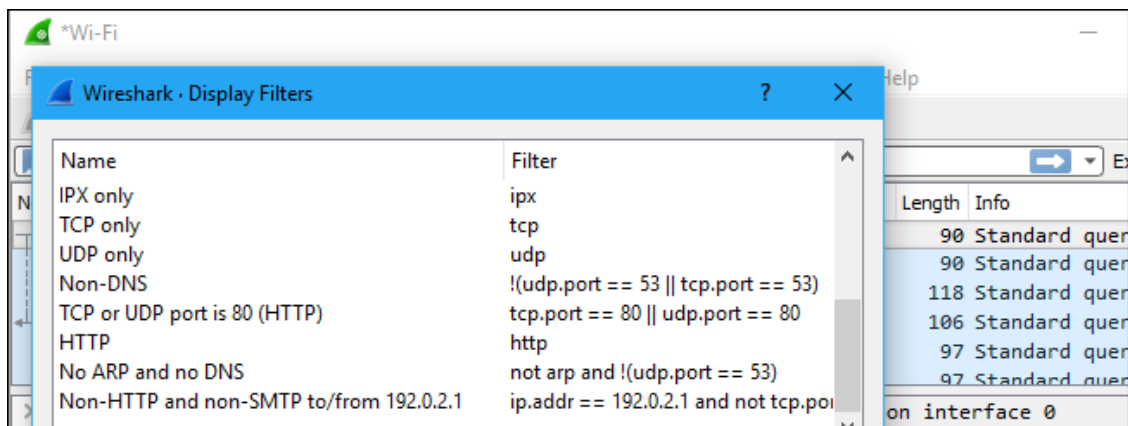


The image shows a Wireshark packet capture window titled '\*Wi-Fi'. The filter bar is set to 'dns'. The packet list shows four DNS packets (No. 305, 306, 307, 308) all with source and destination IP addresses 2601:1c0:cf00:8961:... and protocol DNS. The packet details pane shows the structure of a Standard query.

No.	Time	Source	Destination	Protocol	Length	Info
305	5.248733	2601:1c0:cf00:8961:...	2601:1c0:cf00:8961:...	DNS	90	Standard query
306	5.249092	2601:1c0:cf00:8961:...	2601:1c0:cf00:8961:...	DNS	90	Standard query
307	5.269967	2601:1c0:cf00:8961:...	2601:1c0:cf00:8961:...	DNS	118	Standard query
308	5.270325	2601:1c0:cf00:8961:...	2601:1c0:cf00:8961:...	DNS	106	Standard query

You can also click Analyze > Display Filters to choose a filter from among the default filters included in Wireshark. From here, you can add your own custom filters and save them to easily access them in the future.

For more information on Wireshark's display filtering language, read the [Building display filter expressions](#) page in the official Wireshark documentation.

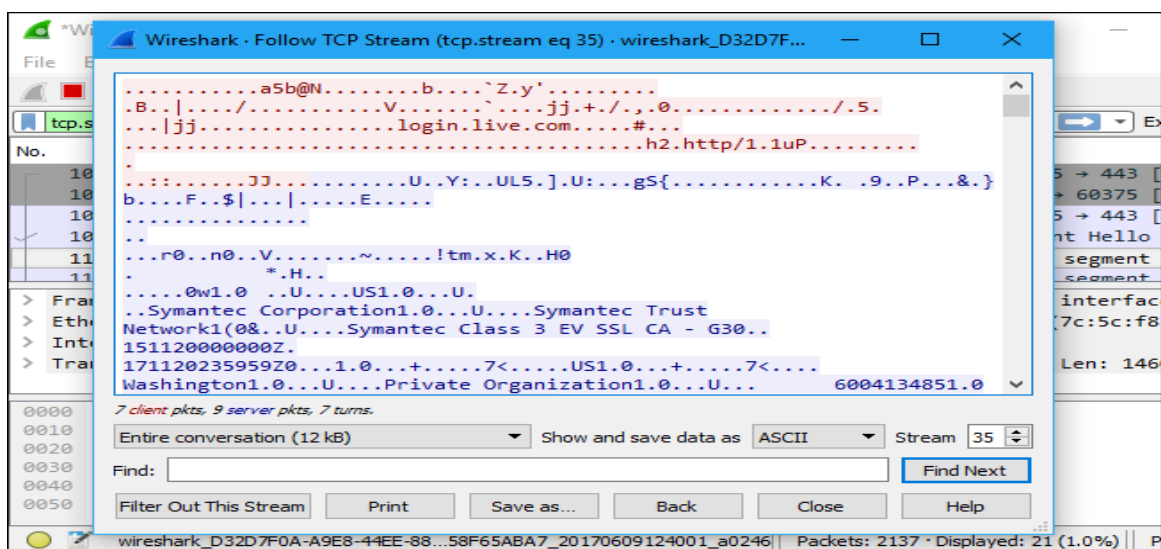


The image shows the 'Wireshark · Display Filters' dialog box. It contains a list of default filters with their corresponding filter expressions.

Name	Filter
IPX only	ipx
TCP only	tcp
UDP only	udp
Non-DNS	!(udp.port == 53    tcp.port == 53)
TCP or UDP port is 80 (HTTP)	tcp.port == 80    udp.port == 80
HTTP	http
No ARP and no DNS	not arp and !(udp.port == 53)
Non-HTTP and non-SMTP to/from 192.0.2.1	ip.addr == 192.0.2.1 and not tcp.port == 80

Another interesting thing you can do is right-click a packet and select Follow > TCP Stream.

You'll see the full TCP conversation between the client and the server. You can also click other protocols in the Follow menu to see the full conversations for other protocols, if applicable.



The image shows the 'Wireshark · Follow TCP Stream (tcp.stream eq 35) · wireshark\_D32D7F...' window. It displays a hex dump and ASCII representation of the TCP stream data. The ASCII part shows a login attempt for 'a5b@N...' on 'login.live.com'.

7 client pkts, 9 server pkts, 7 turns.

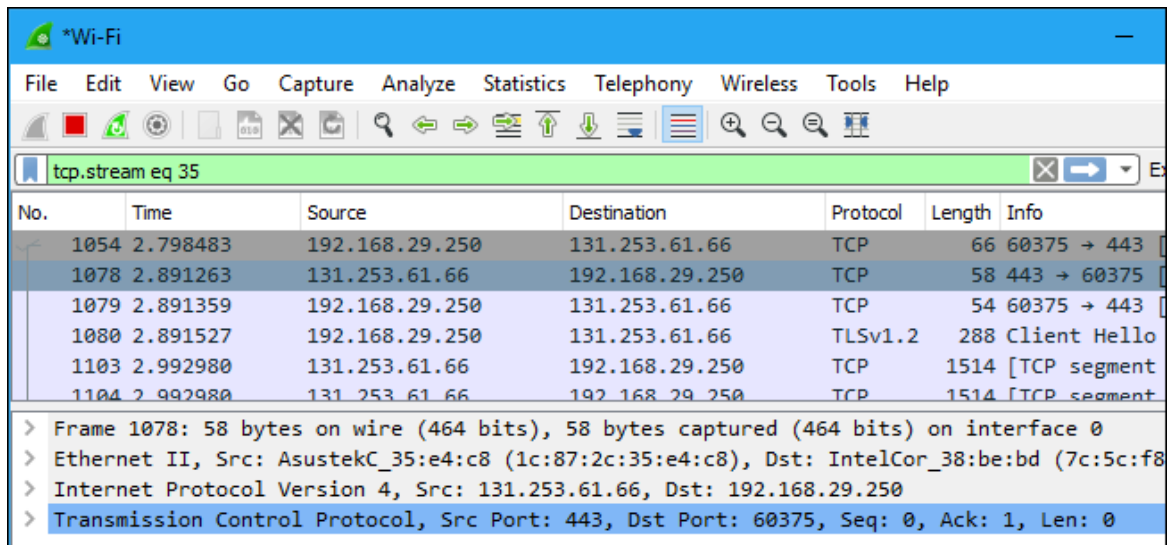
Entire conversation (12 kB) Show and save data as ASCII Stream 35

Find: Find Next

Filter Out This Stream Print Save as... Back Close Help

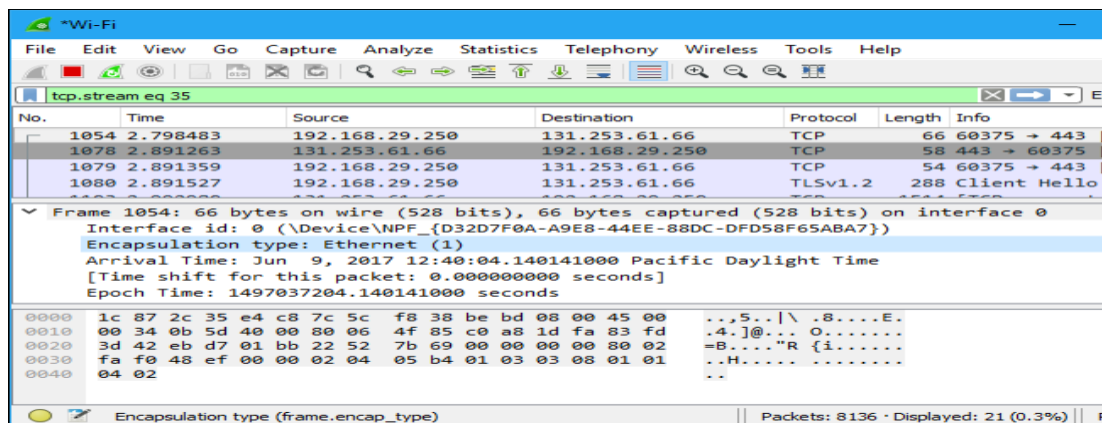
wireshark\_D32D7F0A-A9E8-44EE-88...58F65ABA7\_20170609124001\_a0246 Packets: 2137 · Displayed: 21 (1.0%)

Close the window and you'll find a filter has been applied automatically. Wireshark is showing you the packets that make up the conversation.

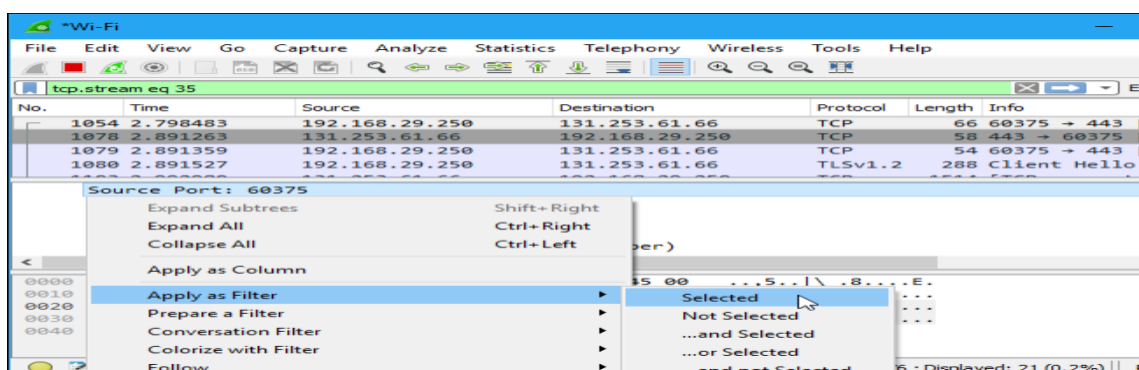


## Inspecting Packets

Click a packet to select it and you can dig down to view its details.



You can also create filters from here — just right-click one of the details and use the Apply as Filter submenu to create a filter based on it.



Wireshark is an extremely powerful tool, and this tutorial is just scratching the surface of what you can do with it. Professionals use it to debug network protocol implementations, examine security problems and inspect network protocol internals.

**CONCLUSION:** Thus, we have studied the working of Wire Shark.

**SIGN AND REMARK**

<b>R1 (3 Marks)</b>	<b>R2 (3 Marks)</b>	<b>R3 (3 Marks)</b>	<b>R4 (3 Mark)</b>	<b>R5 (3 Mark)</b>	<b>Total (15 Marks)</b>	<b>Signature</b>



## EXPERIMENT No. 10

**Date of Performance :**

**Date of Submission :**

**AIM:** Set up multiple IP addresses on a single LAN Using nestat and route commands of Linux, do the following:

- 1] View current routing table
- 2] Add and delete routes
- 3] Change default gateway
- 4] Perform packet filtering by enabling IP forwarding using IPtables in Linux.

### Theory:

First, let us find the IP address of the network card. In my Ubuntu 15.10 server, I use only one network card.

Run the following command to find out the IP address:

```
sudo ip addr
```

### **Sample output:**

```
1: lo: <LOOPBACK,UP,LOWER_UP> mtu 65536 qdisc noqueue state UNKNOWN group default
link/loopback 00:00:00:00:00:00 brd 00:00:00:00:00:00
inet 127.0.0.1/8 scope host lo
valid_lft forever preferred_lft forever
inet6 ::1/128 scope host
valid_lft forever preferred_lft forever
2: enp0s3: <BROADCAST,MULTICAST,UP,LOWER_UP> mtu 1500 qdisc pfifo_fast state UP
group default qlen 1000
link/ether 08:00:27:2a:03:4b brd ff:ff:ff:ff:ff:ff
inet 192.168.1.103/24 brd 192.168.1.255 scope global enp0s3
valid_lft forever preferred_lft forever
inet6 fe80::a00:27ff:fe2a:34e/64 scope link
valid_lft forever preferred_lft forever
```

Or

```
sudo ifconfig
```

### **Sample output:**

```
enp0s3 Link encap:Ethernet HWaddr 08:00:27:2a:03:4b
inet addr:192.168.1.103 Bcast:192.168.1.255 Mask:255.255.255.0
```

```
inet6 addr: fe80::a00:27ff:fe2a:34e/64 Scope:Link
UP BROADCAST RUNNING MULTICAST MTU:1500 Metric:1
RX packets:186 errors:0 dropped:0 overruns:0 frame:0
TX packets:70 errors:0 dropped:0 overruns:0 carrier:0
collisions:0 txqueuelen:1000
RX bytes:21872 (21.8 KB) TX bytes:9666 (9.6 KB)
```

```
lo Link encap:Local Loopback
inet addr:127.0.0.1 Mask:255.0.0.0
inet6 addr: ::1/128 Scope:Host
UP LOOPBACK RUNNING MTU:65536 Metric:1
RX packets:217 errors:0 dropped:0 overruns:0 frame:0
TX packets:217 errors:0 dropped:0 overruns:0 carrier:0
collisions:0 txqueuelen:0
RX bytes:38793 (38.7 KB) TX bytes:38793 (38.7 KB)
```

As you see in the above output, my network card name is **enp0s3**, and its IP address is **192.168.1.103**. Now let us add an additional IP address, for example **192.168.1.104**, to the Interface card. Open your Terminal and run the following command to add additional IP.

```
sudo ip addr add 192.168.1.104/24 dev enp0s3
```

Now, let us check if the IP is added using command:

```
sudo ip address show enp0s3
```

### Sample output:

```
2: enp0s3: <BROADCAST,MULTICAST,UP,LOWER_UP> mtu 1500 qdisc pfifo_fast state UP
group default qlen 1000
link/ether 08:00:27:2a:03:4e brd ff:ff:ff:ff:ff:ff
inet 192.168.1.103/24 brd 192.168.1.255 scope global enp0s3
valid_lft forever preferred_lft forever
inet 192.168.1.104/24 scope global secondary enp0s3
valid_lft forever preferred_lft forever
inet6 fe80::a00:27ff:fe2a:34e/64 scope link
valid_lft forever preferred_lft forever
```

Similarly, you can add as many IP addresses as you want.

Let us ping the IP address to verify it.

```
sudo ping 192.168.1.104
```

### Sample output:

```
PING 192.168.1.104 (192.168.1.104) 56(84) bytes of data.  
64 bytes from 192.168.1.104: icmp_seq=1 ttl=64 time=0.901 ms  
64 bytes from 192.168.1.104: icmp_seq=2 ttl=64 time=0.571 ms  
64 bytes from 192.168.1.104: icmp_seq=3 ttl=64 time=0.521 ms  
64 bytes from 192.168.1.104: icmp_seq=4 ttl=64 time=0.524 ms
```

### To check the routing table

**Command:** `netstat -rn`

```
$ netstat -rn
```

Kernel IP routing table

Destination	Gateway	Genmask	Flags	MSS	Window	irtt	Iface
0.0.0.0	192.168.0.1	0.0.0.0	UG	0	0	0	wlan0
192.168.0.0	0.0.0.0	255.255.255.0	U	0	0	0	wlan0

### Adding route

```
sudo route add -net 192.168.3.0 gw 192.168.1.1 netmask 255.255.255.0 dev eth0
```

### Deleting route

```
sudo route del -net 192.168.3.0 gw 192.168.1.1 netmask 255.255.255.0 dev eth0
```

### A quick way to add default route

```
route add default gw 192.168.1.1
```

### A quick way to delete default route

```
route del default gw 192.168.1.1
```

**CONCLUSION:** Thus, we have studied and successfully add the multiple IP address and also perform actions in Linux

### SIGN AND REMARK

R1 (3 Marks)	R2 (3 Marks)	R3 (3 Marks)	R4 (3 Mark)	R5 (3 Mark)	Total (15 Marks)	Signature

