

Vishwakarma Government Engineering College

Computer Engineering

INDEX

Subject: Computer Networks (3150710)

Enrollment No: 180170107030

Name of the Student: Gabu Siddharth Merambhai

Sr. No	Aim	CO	Date	Page No	Marks (out of 10)	Faculty Signature
1	Networking Commands	CO2	10/7	1		
2	Configuration of Wireshark packet analyzer and capturing packets using TCPDUMP/Wireshark.	CO1	17/7	16		
3	Dissection of Packets using Wireshark	CO1	24/7	19		
4	Simulating LAN configuration using the Cisco Packet tracer emulator.	CO4	31/7	22		
5	Design a heterogeneous network that consists of a wired and wireless LAN connection using a router.	CO4	7/8	29		
6	Network Subnetting	CO4	14/8	33		
7	Wireless Router Configuration in Cisco Packet Tracer	CO4	4/9	37		
8	Implementing echo sever using Socket programming.	CO3	11/9	40		
9	Implementing file server using socket programming.	CO3	18/9	43		
10	Implementing Error Detection mechanism using Socket Programming.	CO5	9/10	46		

Vishwakarma Government Engineering College

Computer Engineering

Sr. No	Practical Description	CO
1	Basics and Networking Commands	CO2
	Execution of various Networking Commands. Understand the below commands properly. Execute the commands with different switch options. Note: Refer manual for the each command using man command in linux platform Students are supposed to prepare a write up for the below commands along with the screenshot of their execution with different parameters applied. 1. ifconfig 2. traceroute 3. netstat 4. arp 5. ssh 6. ping 7. nslookup 8. iwconfig 9. ftp 10. wget	
2	Packet Capturing and Dissection	CO1
	Capturing and filtering packets using TCPDUMP/Wireshark Tool. You need to first install Linux version of Wireshark software in your computer which is available for download at http://www.wireshark.org/ . In this assignment, you will be evaluated for your familiarity in using Tcpdump and Wire shark utility for packet capture and analysis. Start packet capture in Wireshark application and then open your web browser and type in an URL of website of your choice (www. Vgecg.ac.in, www.google.com, etc). Apply various filters on it as discussed during the lab session.	
3	Dissection of packets using wire shark. Configure the Webserver in your local machine (e.g., Apache web server) and deploy the server side scripting page which validates your userid and password. Design a client side html page which prompts for user id and password. Trace the step by step transactional flow between client and webserver (i.e., dns, http and tcp stream) when you press submit on client page. Find the user id and password from the TCP stream. Summarize the flow in your own words.	

Vishwakarma Government Engineering College

Computer Engineering

	Network topology Configuration	CO4
4	LAN Configuration Create and Simulate LAN configuration using Cisco Packet tracer emulator. Save the Pkt file of your network configuration.	
5	Heterogeneous Network Configuration Design a heterogeneous network which consists of wired and wireless LAN connected using router. Create a file server which allows user to download and upload the files. Prepare the document which should comprise the screen shot. Save the Pkt file of your network configuration.	
6	Subnetting Network You have sub-netted your class C network 192.168.1.0 with a subnet mask of 255.255.255.252. Please list the following: number of networks, number of hosts per network, the full range of the first three networks, and the usable address range from those first three networks. Additionally, identify the broadcast addresses for each network. Design this network in packet tracer and simulate the working of the network. Save the Pkt file of your network design	
7	Wireless Router Configuration Configuring various network parameter in Wireless Router <ol style="list-style-type: none"> 1) Configure security key 2) Enable DHCP server 3) Apply Application/Network/Mac layer Filter 4) Configure Port Filtering 	
	Socket Programming	CO3/
	Client-Sever Programming in C	CO5
8	Implementing connection oriented echo sever using Socket programming.	
9	Implementing connection oriented file server using socket programming.	
10	Implementing connection less Error Detection mechanism using Socket Programming.	

Practical – 1

Aim: Networking Command

Arp(Address Resolution Protocol):

Displays and modifies entries in the Address Resolution Protocol (ARP) cache. The ARP cache contains one or more tables that are used to store IP addresses and their resolved Ethernet or Token Ring physical addresses. There is a separate table for each Ethernet or Token Ring network adapter installed on your computer. Used without parameters, Arp displays help information.

Syntax: Arp [parameter [<addr>]]

/a - Displays current arp cache tables for all interfaces.

/d [inetaddr] - Deletes an entry with a specific IP address, where inetaddr is the IP address.

/n -To display the arp cache entry for a specific IP address.

/s [<inetaddr> <etheraddr>] - Adds a static entry to the arp cache that resolves the IP address inetaddr to the physical address etheraddr.

```
C:\Windows\system32>arp /a

Interface: 192.168.43.150 --- 0x4
Internet Address      Physical Address      Type
192.168.43.1          2e-fd-ab-d6-8a-dd     dynamic
192.168.43.255        ff-ff-ff-ff-ff-ff     static
224.0.0.22            01-00-5e-00-00-16     static
224.0.0.252          01-00-5e-00-00-fc     static
239.255.255.250       01-00-5e-7f-ff-fa     static
255.255.255.255       ff-ff-ff-ff-ff-ff     static

C:\Windows\system32>arp /d 192.168.43.1

C:\Windows\system32>arp /a

Interface: 192.168.43.150 --- 0x4
Internet Address      Physical Address      Type
192.168.43.255        ff-ff-ff-ff-ff-ff     static
224.0.0.22            01-00-5e-00-00-16     static
224.0.0.252          01-00-5e-00-00-fc     static
239.255.255.250       01-00-5e-7f-ff-fa     static
255.255.255.255       ff-ff-ff-ff-ff-ff     static
```

Ipconfig:

Displays all current TCP/IP network configuration values and refreshes Dynamic Host Configuration Protocol (DHCP) and Domain Name System (DNS) settings. Used without parameters, ipconfig displays Internet Protocol version 4 (IPv4) and IPv6 addresses, subnet mask, and default gateway for all adapters.

Syntax: ipconfig [/allcompartments]

/all - Display full configuration information.

/release [adapter] - Release the IP address for the specified adapter.

/renew [adapter] - Renew the IP address for the specified adapter.

/flushdns - Purge the DNS Resolver cache.

/registerdns - Refresh all DHCP leases and re-register DNS names.

/displaydns - Display the contents of the DNS Resolver Cache.

/showclassid [adapter] - Display all the DHCP class IDs allowed for adapter.

/setclassid adapter [classid] - Modify the dhcp class id.

```
C:\Windows\system32>ipconfig /all

Windows IP Configuration

Host Name . . . . . : DELL
Primary Dns Suffix . . . . . :
Node Type . . . . . : Mixed
IP Routing Enabled. . . . . : No
WINS Proxy Enabled. . . . . : No


Ethernet adapter Bluetooth Network Connection:

Media State . . . . . : Media disconn
Connection-specific DNS Suffix . :
Description . . . . . : Bluetooth Devi
Physical Address. . . . . : EC-0E-C4-06-AF
DHCP Enabled. . . . . : Yes
Autoconfiguration Enabled . . . . : Yes


Wireless LAN adapter Local Area Connection* 15:

Media State . . . . . : Media disconn
Connection-specific DNS Suffix . :
Description . . . . . : Microsoft Host
Physical Address. . . . . : 5E-0E-C4-06-AF
```

Ping:

The ping command verifies IP-level connectivity to another TCP/IP computer by sending Internet Control Message Protocol (ICMP) echo Request messages. The receipt of corresponding echo Reply messages are displayed, along with round-trip times. ping is the primary TCP/IP command used to troubleshoot connectivity, reachability, and name resolution. Used without parameters, ping displays help.

Syntax: ping [/parameters]

/t - Specifies that ping continue sending echo Request messages to the destination until interrupted.

/n <count> - Specifies the number of echo Request messages sent. The default is 4.

/l <size> - Specifies the length, in bytes, of the Data field in the echo Request messages sent. The default is 32. The maximum size is 65,527.

/w <timeout> - Specifies the amount of time, in milliseconds, to wait for the echo Reply message that corresponds to a given echo Request message to be received.

/4 - Specifies that IPv4 is used to ping.

/6 - Specifies that IPv6 is used to ping.

/s - Specifies that the Internet timestamp option in the IP header is used to record the time of arrival for the echo Request message and corresponding echo Reply message for each hop.

```
C:\Windows\system32>ping google.com /n 2

Pinging google.com [2404:6800:4009:803::200e] with 32 bytes of data:
Reply from 2404:6800:4009:803::200e: time=32ms
Reply from 2404:6800:4009:803::200e: time=63ms

Ping statistics for 2404:6800:4009:803::200e:
    Packets: Sent = 2, Received = 2, Lost = 0 (0% loss),
    Approximate round trip times in milli-seconds:
        Minimum = 32ms, Maximum = 63ms, Average = 47ms

C:\Windows\system32>ping google.com /t

Pinging google.com [2404:6800:4009:803::200e] with 32 bytes of data:
Reply from 2404:6800:4009:803::200e: time=65ms
Reply from 2404:6800:4009:803::200e: time=73ms
Reply from 2404:6800:4009:803::200e: time=41ms
Reply from 2404:6800:4009:803::200e: time=63ms
Reply from 2404:6800:4009:803::200e: time=57ms

Ping statistics for 2404:6800:4009:803::200e:
    Packets: Sent = 5, Received = 5, Lost = 0 (0% loss),
    Approximate round trip times in milli-seconds:
        Minimum = 41ms, Maximum = 73ms, Average = 59ms
Control-C
```

Tracert:

Determines the path taken to a destination by sending Internet Control Message Protocol (ICMP) echo Request or ICMPv6 messages to the destination with incrementally increasing time to Live (TTL) field values. The path displayed is the list of near/side router interfaces of the routers in the path between a source host and a destination. The near/side interface is the interface of the router that is closest to the sending host in the path.

Syntax: `tracert [-d] [-h maximum_hops] [-j host-list] [-w timeout] [-R] [-S srcaddr] [-4] [-6] target_name`

<code>-d</code>	Do not resolve addresses to hostnames.
<code>-h maximum_hops</code>	Maximum number of hops to search for target.
<code>-j host-list</code>	Loose source route along host-list (IPv4-only).
<code>-w timeout</code>	Wait timeout milliseconds for each reply.
<code>-R</code>	Trace round-trip path (IPv6-only).
<code>-S srcaddr</code>	Source address to use (IPv6-only).
<code>-4</code>	Force using IPv4.
<code>-6</code>	Force using IPv6.

```
F:\iTextexample>tracert -h 3 -w 6000 google.com

Tracing route to google.com [2404:6800:4009:803::200e]
over a maximum of 3 hops:

  1     2 ms     3 ms     1 ms  2402:8100:3983:540e::7e
  2    99 ms    206 ms    230 ms  2402:8100:39af:ffff:ffff:ffff:ffff:ffff
  3      *       *       *      Request timed out.

Trace complete.

F:\iTextexample>tracert -d google.com

Tracing route to google.com [2404:6800:4009:803::200e]
over a maximum of 30 hops:

  1     36 ms     1 ms     2 ms  2402:8100:3983:540e::7e
  2    299 ms    197 ms    212 ms  2402:8100:39af:ffff:ffff:ffff:ffff:ffff
  3      *       *       *      Request timed out.
  4      *       *       *      Request timed out.
  5      *       *       *      Request timed out.
  6      *       *       *      Request timed out.
^C
F:\iTextexample>
```

Getmac:

This tool enables an administrator to display the MAC address for network adapters on a system.

Parameter List:

/S system Specifies the remote system to connect to.

/U [domain\]user Specifies the user context under which the command should execute.

/P [password] Specifies the password for the given user context. Prompts for input if omitted.

/FO format Specifies the format in which the output is to be displayed.
Valid values: "TABLE", "LIST", "CSV".

/NH Specifies that the "Column Header" should not be displayed in the output.
Valid only for TABLE and CSV formats.

/V Specifies that verbose output is displayed.

/? Displays this help message.

```
Physical Address    Transport Name
=====
-2A-72-CB-A9-06    Media disconnected
-0E-C4-06-AF-55    \Device\Tcpip_{C4F5268B-7281-4D73-B614-33AD48C745B8}
-0E-C4-06-AF-56    Media disconnected
```


Pathping:

Provides information about network latency and network loss at intermediate hops between a source and destination. This command sends multiple echo Request messages to each router between a source and destination, over a period of time, and then computes results based on the packets returned from each router. Because this command displays the degree of packet loss at any given router or link, you can determine which routers or subnets might be having network problems.

Usage: pathping [-g host-list] [-h maximum_hops] [-i address] [-n]
 [-p period] [-q num_queries] [-w timeout]
 [-4] [-6] target_name

Options:

-g host-list	Loose source route along host-list.
-h maximum_hops	Maximum number of hops to search for target.
-i address	Use the specified source address.
-n	Do not resolve addresses to hostnames.
-p period	Wait period milliseconds between pings.
-q num_queries	Number of queries per hop.
-w timeout	Wait timeout milliseconds for each reply.
-4	Force using IPv4.
-6	Force using IPv6.

```
F:\iTextexample>pathping google.com -h 2

Tracing route to google.com [2404:6800:4009:807::200e]
over a maximum of 2 hops:
  0  DELL [2402:8100:3986:e91a:f1a2:953b:4a0f:dd2a]
  1  2402:8100:3986:e91a::87
  2  2402:8100:39af:ffff:ffff:ffff:ffff:fffe

Computing statistics for 50 seconds...
Hop  RTT      Source to Here   This Node/Link   Address
    0                               DELL [2402:8100:3986:e91a:f1a2:953
b:4a0f:dd2a]
    1    8ms      0/ 100 = 0%      0/ 100 = 0%      | 2402:8100:3986:e91a::87
    2   45ms      0/ 100 = 0%      0/ 100 = 0%      | 2402:8100:39af:ffff:ffff:ffff:ffff
:fffe

Trace complete.
```

Netstat:

Displays protocol statistics and current TCP/IP network connections.

NETSTAT [-a] [-b] [-e] [-f] [-n] [-o] [-p proto] [-r] [-s] [-x] [-t] [interval]

- a Displays all connections and listening ports.
- e Displays Ethernet statistics. This may be combined with the -s option.
- f Displays Fully Qualified Domain Names (FQDN) for foreign addresses.
- n Displays addresses and port numbers in numerical form.
- o Displays the owning process ID associated with each connection.
- p proto Shows connections for the protocol specified by proto;
- r Displays the routing table.
- s Displays per-protocol statistics. By default, statistics are shown for IP, IPv6, ICMP, ICMPv6, TCP, TCPv6, UDP, and UDPv6; the -p option may be used to specify a subset of the default.
- t Displays the current connection offload state.
- x Displays NetworkDirect connections, listeners, and shared endpoints.
- y Displays the TCP connection template for all connections. Cannot be combined with the other options.

```
F:\iTextexample>netstat -e
Interface Statistics
```

	Received	Sent
Bytes	697247392	113897488
Unicast packets	780256	612496
Non-unicast packets	432	41240
Discards	0	0
Errors	0	0
Unknown protocols	0	

```
F:\iTextexample>netstat -p
```

```
Active Connections
```

Proto	Local Address	Foreign Address	State
-------	---------------	-----------------	-------

Hostname:

Displays the host name portion of the full computer name of the computer

Hostname /?

```
F:\iTextexample>hostname
DELL

F:\iTextexample>hostname /?

Prints the name of the current host.

hostname
```

Nslookup:

Displays information that you can use to diagnose Domain Name System (DNS) infrastructure. Before using this tool, you should be familiar with how DNS works. The nslookup command-line tool is available only if you have installed the TCP/IP protocol.

The nslookup command-line tool has two modes: interactive and noninteractive.

If you need to look up only a single piece of data, we recommend using the non-interactive mode. For the first parameter, type the name or IP address of the computer that you want to look up. For the second parameter, type the name or IP address of a DNS name server. If you omit the second argument, **nslookup** uses the default DNS name server.

If you need to look up more than one piece of data, you can use interactive mode. Type a hyphen (-) for the first parameter and the name or IP address of a DNS name server for the second parameter. If you omit both parameters, the tool uses the default DNS name server. While using the interactive mode, you can:

- Interrupt interactive commands at any time, by pressing CTRL+B
- Exit, by typing **exit**.
- Treat a built-in command as a computer name, by preceding it with the escape character (.). An unrecognized command is interpreted as a computer name.

Commands: (identifiers are shown in uppercase, [] means optional)

NAME - print info about the host/domain NAME using default server

NAME1 NAME2 - as above, but use NAME2 as server

help or ? - print info on common commands

set OPTION - set an option

all - print options, current server and host

[no]debug - print debugging information

[no]d2 - print exhaustive debugging information

[no]defname - append domain name to each query

[no]recurse - ask for recursive answer to query

[no]search - use domain search list

[no]vc - always use a virtual circuit

domain=NAME - set default domain name to NAME

srchlist=N1[/N2/.../N6] - set domain to N1 and search list to N1,N2, etc.

root=NAME - set root server to NAME

retry=X - set number of retries to X

timeout=X - set initial time-out interval to X seconds

type=X - set query type (ex.
A,AAAA,A+AAAA,ANY,CNAME,MX,NS,PTR,

SOA,SRV)

querytype=X - same as type

class=X - set query class (ex. IN (Internet), ANY)

[no]mxfr - use MS fast zone transfer

ixfrver=X - current version to use in IXFR transfer request

server NAME - set default server to NAME, using current default server

lserver NAME - set default server to NAME, using initial server

root - set current default server to the root

ls [opt] DOMAIN [> FILE] - list addresses in DOMAIN (optional: output to FILE)

-a - list canonical names and aliases

-d - list all records

-t TYPE - list records of the given RFC record type (ex. A,CNAME,MX,NS,
PTR etc.)

view FILE - sort an 'ls' output file and view it with pg

exit - exit the program

```
C:\Windows\system32>nslookup
Default Server: UnKnown
Address: 192.168.43.1

> ls
Server: UnKnown
Address: 192.168.43.1

*** No internal type for both IPv4 and IPv6 Addresses (A+AAAA) records available
for ls
> search google.com
Server: google.com
Addresses: 2404:6800:4009:80a::200e
          172.217.27.206

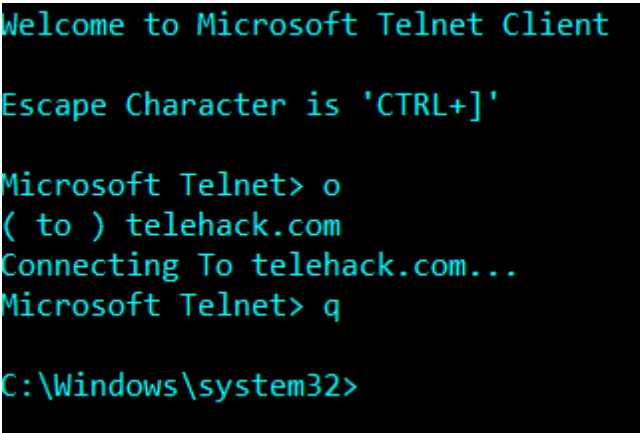
DNS request timed out.
    timeout was 2 seconds.
DNS request timed out.
    timeout was 2 seconds.
*** Request to google.com timed-out
>
```

Telnet:

Communicates with a computer running the telnet Server service.

```
telnet [-a][-e escape char][-f log file][-l user][-t term][host [port]]
```

- a Attempt automatic logon. Same as -l option except uses the currently logged on user's name.
 - e Escape character to enter telnet client prompt.
 - f File name for client side logging
 - l Specifies the user name to log in with on the remote system.
Requires that the remote system support the TELNET ENVIRON option.
 - t Specifies terminal type.
Supported term types are vt100, vt52, ansi and vtnt only.
- host Specifies the hostname or IP address of the remote computer to connect to.
- port Specifies a port number or service name.



```
Welcome to Microsoft Telnet Client

Escape Character is 'CTRL+]'

Microsoft Telnet> o
( to ) telehack.com
Connecting To telehack.com...
Microsoft Telnet> q

C:\Windows\system32>
```

Where:

Displays the location of files that match the search pattern. By default, the search is done along the current directory and in the paths specified by the PATH environment variable.

Parameter List:

- /R Recursively searches and displays the files that match the given pattern starting from the specified directory.
- /Q Returns only the exit code, without displaying the list of matched files. (Quiet mode)
- /F Displays the matched filename in double quotes.
- /T Displays the file size, last modified date and time for all matched files.

pattern Specifies the search pattern for the files to match.

Wildcards * and ? can be used in the pattern. The "\$env:pattern" and "path:pattern" formats can also be specified, where "env" is an environment variable and the search is done in the specified paths of the "env" environment variable. These formats should not be used with /R. The search is also done by appending the extensions of the PATHEXT variable to the pattern.

- /? Displays this help message.

NOTE: The tool returns an error level of 0 if the search is successful, of 1 if the search is unsuccessful and of 2 for failures or errors.

```
F:\iTextexample>where /r E:\javaperformance P9_1.java
E:\javaperformance\collage\9_Practical\P9_1.java

F:\iTextexample>where /r E:\javaperformance Cyc.java
E:\javaperformance\Test\Cyc.java
```

More:

Displays output one screen at a time.

`MORE [/E [/C] [/P] [/S] [/Tn] [+n]] < [drive:][path]filename`
`command-name | MORE [/E [/C] [/P] [/S] [/Tn] [+n]]`
`MORE /E [/C] [/P] [/S] [/Tn] [+n] [files]`

`[drive:][path]filename` Specifies a file to display one screen at a time.

`command-name` Specifies a command whose output will be displayed.

`/E` Enable extended features
`/C` Clear screen before displaying page
`/P` Expand FormFeed characters
`/S` Squeeze multiple blank lines into a single line
`/Tn` Expand tabs to n spaces (default 8)

Switches can be present in the MORE environment variable.

`+n` Start displaying the first file at line n

```
MORE [/E [/C] [/P] [/S] [/Tn] [+n]] < [drive:][path]filename
command-name | MORE [/E [/C] [/P] [/S] [/Tn] [+n]]
MORE /E [/C] [/P] [/S] [/Tn] [+n] [files]

[drive:][path]filename Specifies a file to display one
                        screen at a time.

command-name           Specifies a command whose output
                        will be displayed.

/E      Enable extended features
/C      Clear screen before displaying page
/P      Expand FormFeed characters
/S      Squeeze multiple blank lines into a single line
/Tn     Expand tabs to n spaces (default 8)

Switches can be present in the MORE environment
variable.

+n      Start displaying the first file at line n

files   List of files to be displayed. Files in the list
        are separated by blanks.
-- More (72%) --
```


Color:

Sets the default console foreground and background colors.

COLOR [attr]

attr Specifies color attribute of console output

Color attributes are specified by TWO hex digits -- the first corresponds to the background; the second the foreground. Each digit can be any of the following values:

0 = Black	8 = Gray
1 = Blue	9 = Light Blue
2 = Green	A = Light Green
3 = Aqua	B = Light Aqua
4 = Red	C = Light Red
5 = Purple	D = Light Purple
6 = Yellow	E = Light Yellow
7 = White	F = Bright White

If no argument is given, this command restores the color to what it was when CMD.EXE started. This value either comes from the current console window, the /T command line switch or from the DefaultColor registry value.

The COLOR command sets ERRORLEVEL to 1 if an attempt is made to execute the COLOR command with a foreground and background color that are the same.

Practical - 2

AIM: Configuration of Wireshark packet analyzer and capturing packets using TCPDUMP/Wireshark.

Wireshark can capture traffic from many different network media types, including Ethernet, Wireless LAN, Bluetooth, USB, and more. The specific media types supported may be limited by several factors, including your hardware and operating system.

The following are some of the many features Wireshark provides:

Available for UNIX and Windows.

Capture live packet data from a network interface.

Open files containing packet data captured with tcpdump/WinDump, Wireshark, and many other packet capture programs.

Import packets from text files containing hex dumps of packet data.

Display packets with very detailed protocol information.

Save packet data captured.

Export some or all packets in a number of capture file formats.

Filter packets on many criteria.

Search for packets on many criteria.

Colorize packet display based on filters.

Create various statistics etc.

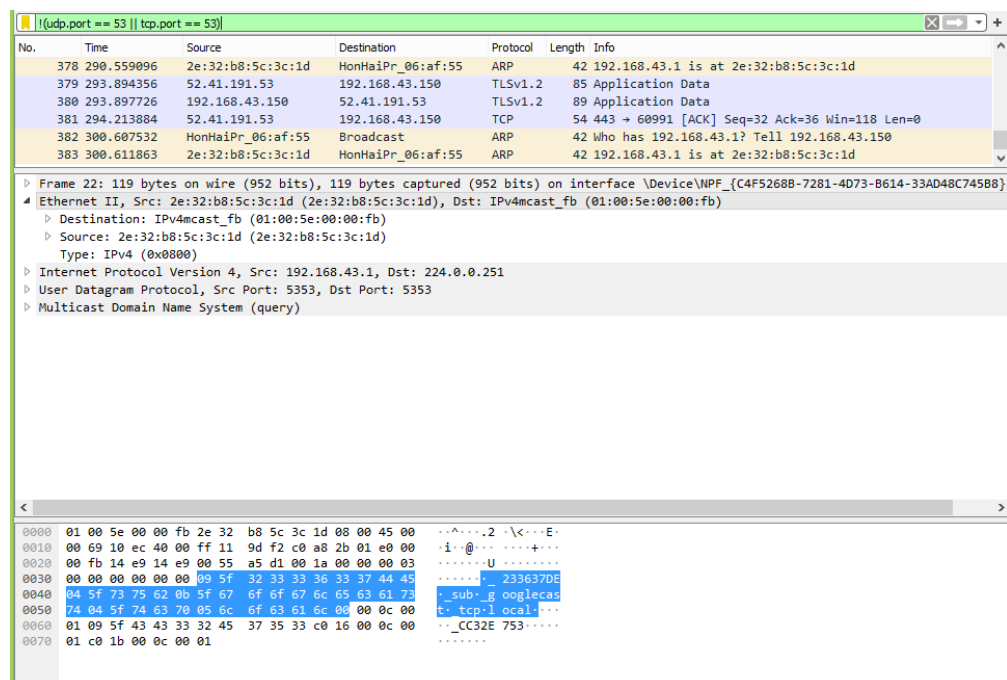


Fig 2.1

In Fig 2.1 as shown above we are applying filter of Non-DNS to capturing.

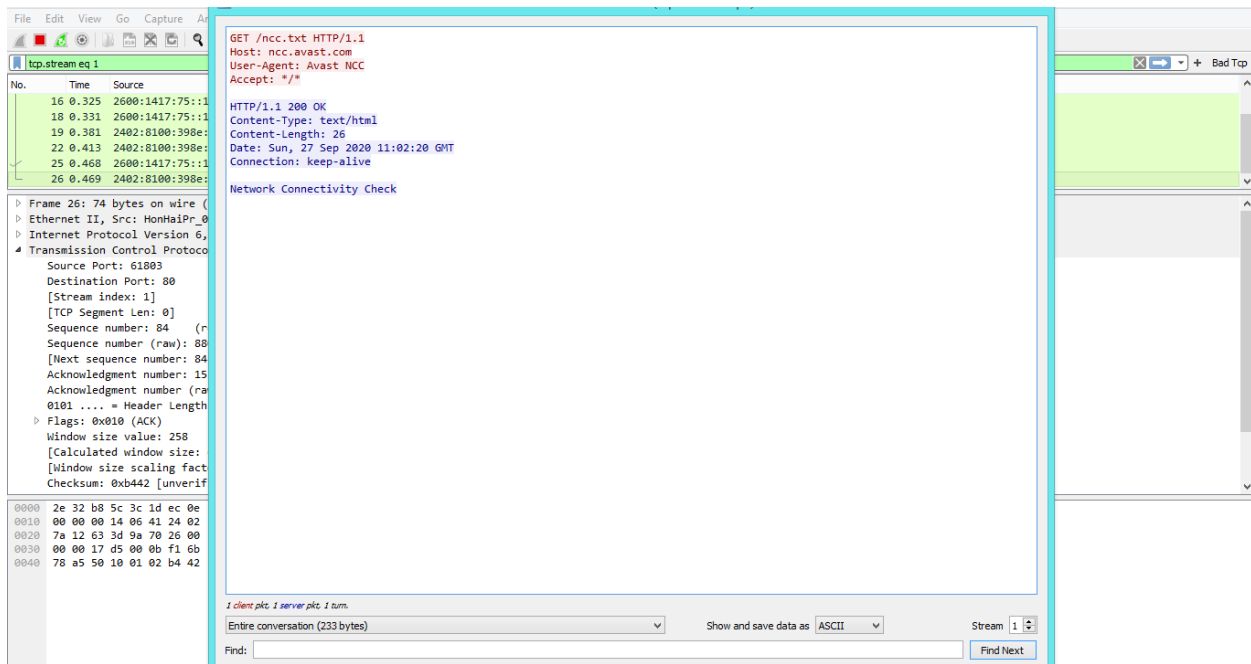


Fig 2.2

Analyzing tcp stream in Fig 2.2 its showing status HTTP 200 connection is ok, content length many other info.

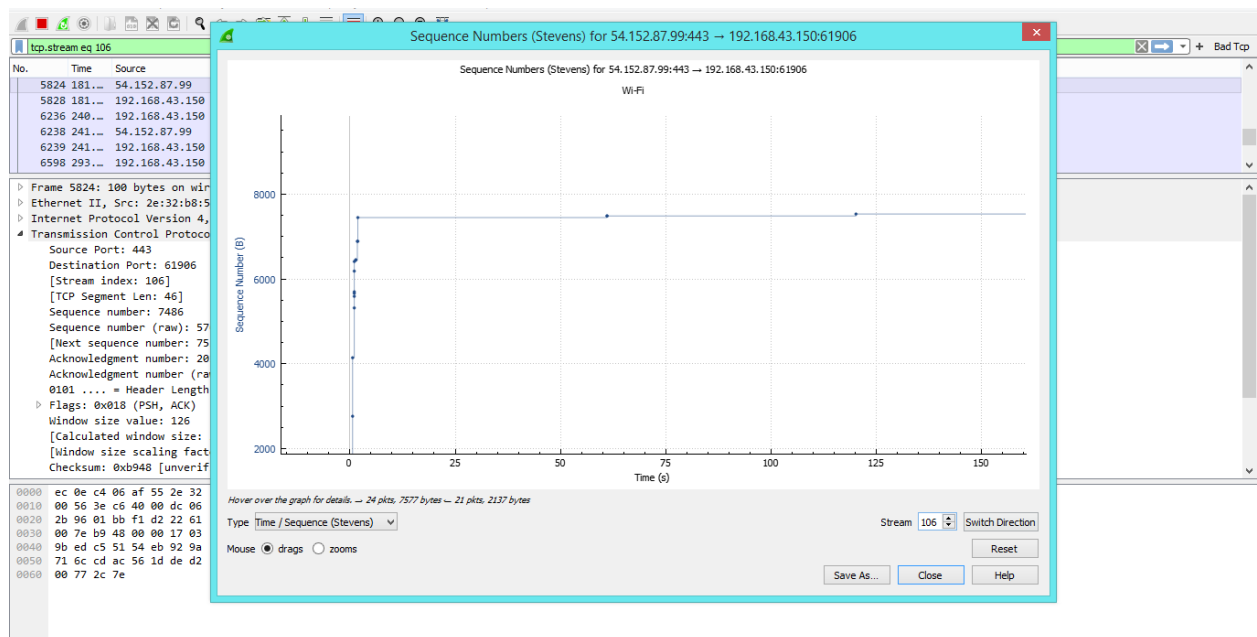


Fig 2.3

Fig 2.3 is tcp stream graph to analyze source to destination tcp stream.

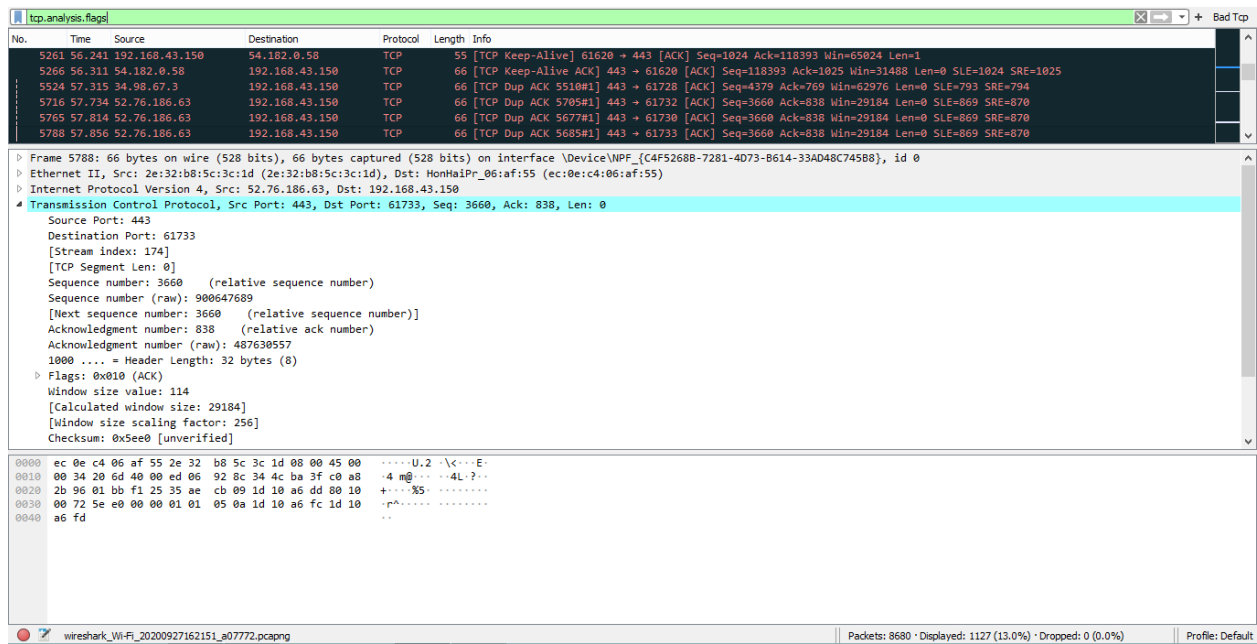


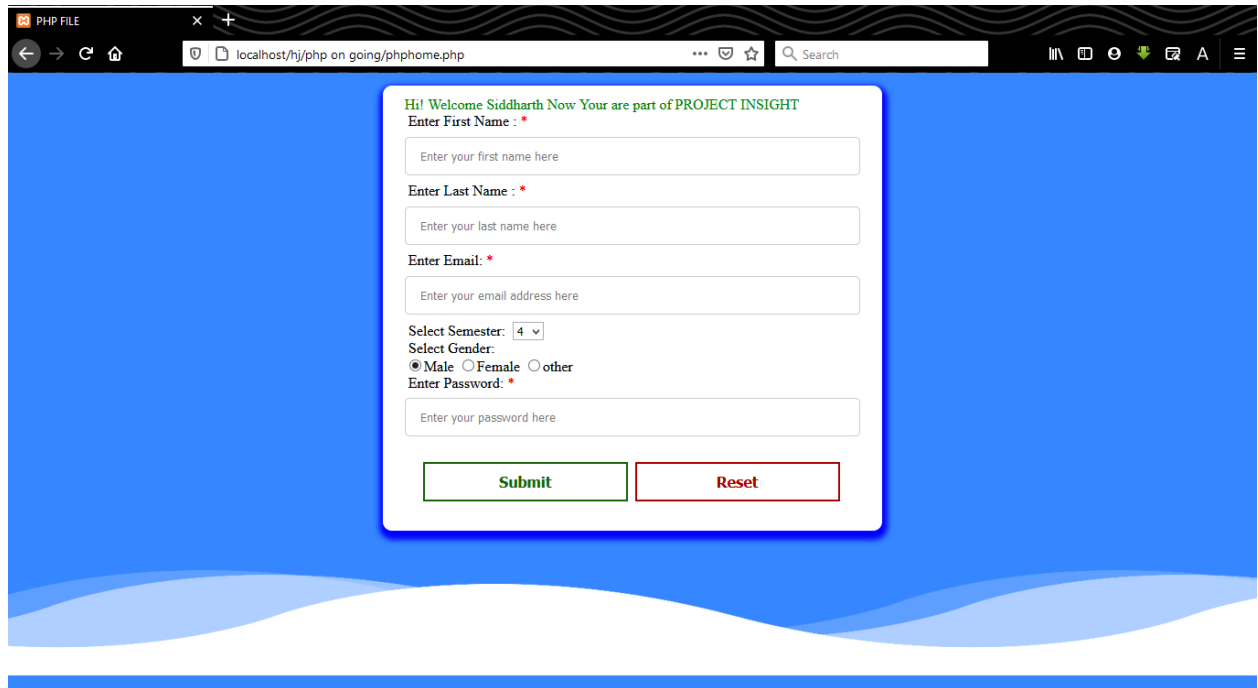
Fig 2.4

In Fig2.4 applying filter to analysis a tcp flags like retransmission, timeout, etc.

Practical – 3

Aim: Dissection of Packets using Wireshark.

Fig 3.1 show registration page for user. We are going to use this page for user taking details.

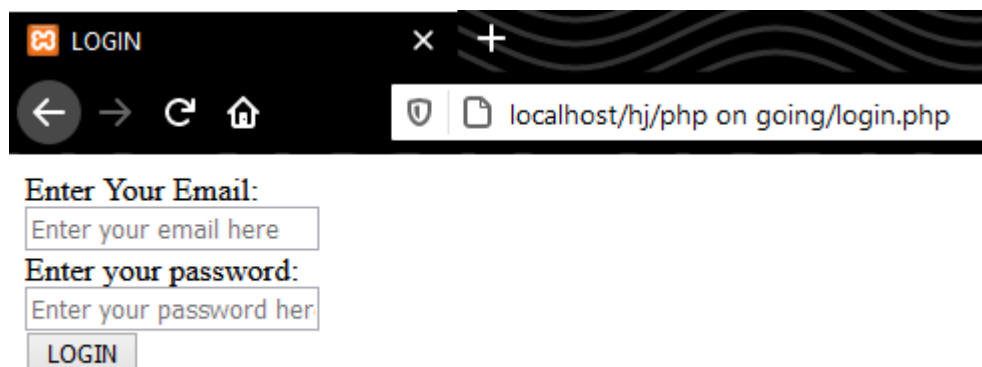


The screenshot shows a web browser window with the address bar displaying 'localhost/hj/php on going/phphome.php'. The page has a blue background with a white registration form in the center. The form contains the following fields and controls:

- Greeting: 'Hi! Welcome Siddharth Now Your are part of PROJECT INSIGHT'
- Enter First Name : * (text input)
- Enter Last Name : * (text input)
- Enter Email: * (text input)
- Select Semester: 4 (dropdown menu)
- Select Gender: ☒ Male ☐ Female ☐ other
- Enter Password: * (text input)
- Submit button (green border)
- Reset button (red border)

Fig 3.1

Fig 3.2 shows login page that we are going to dissection in wireshark



The screenshot shows a web browser window with the address bar displaying 'localhost/hj/php on going/login.php'. The page has a black header with the word 'LOGIN' in white. The main content area is white and contains the following fields and controls:

- Enter Your Email: (text input)
- Enter your password: (text input)
- LOGIN button (grey)

Fig 3.2

In Fig 3.4 & Fig 3.5 we capturing registration page and during dissection using filter “tcp.port == 80” of this we following tcp stream that show username, user email, password, etc.

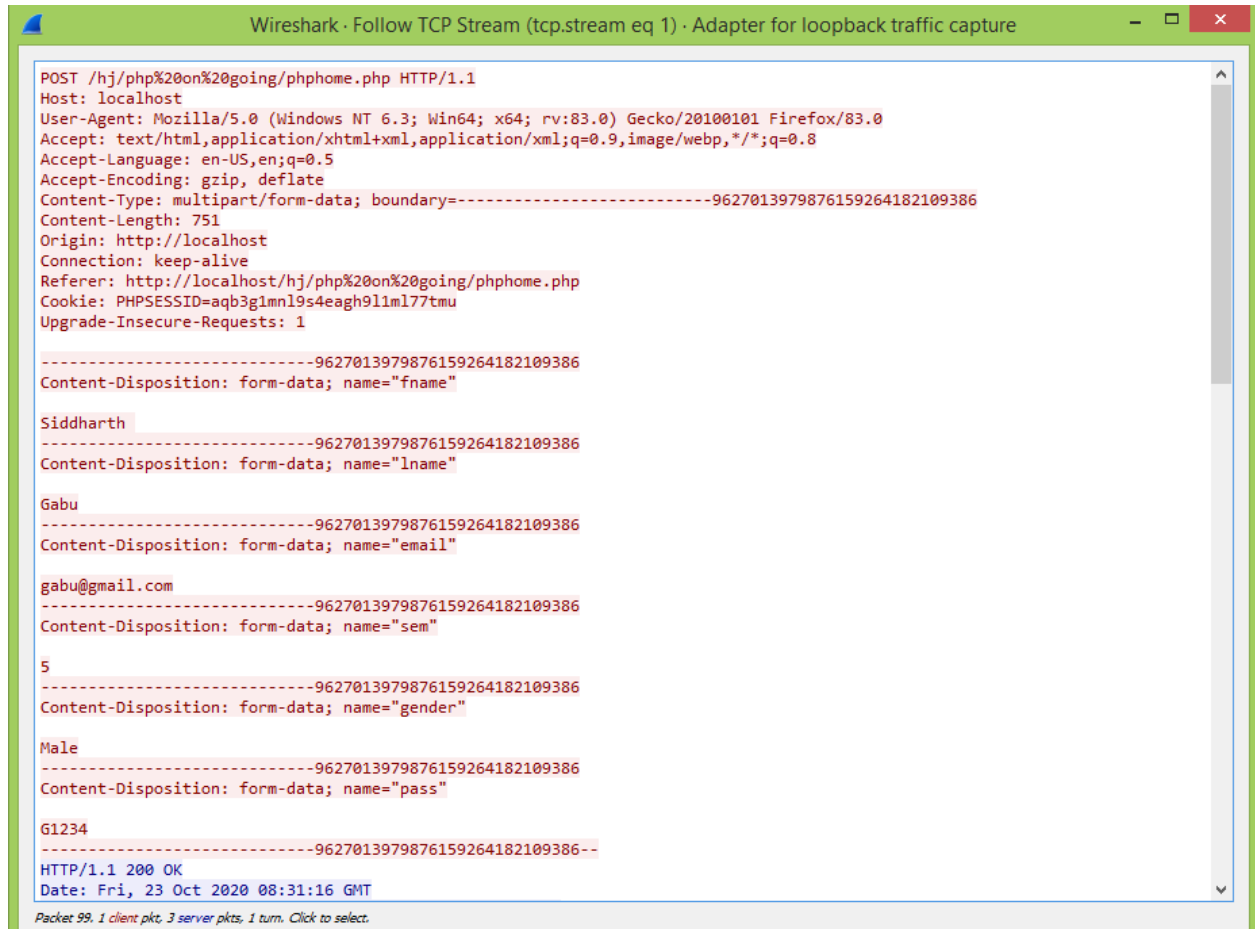


Fig 3.4

```

-----9627013979876159264182109386
Content-Disposition: form-data; name="fname"

Siddharth
-----9627013979876159264182109386
Content-Disposition: form-data; name="lname"

Gabu
-----9627013979876159264182109386
Content-Disposition: form-data; name="email"

gabu@gmail.com
-----9627013979876159264182109386
Content-Disposition: form-data; name="sem"

5

```

Fig 3.5

In Fig 3.6 & 3.7 we are using login page during dissection using only filter “http” of this we following that show username, password.

Wireshark packet capture showing an HTTP POST request for a login page. The packet list shows a POST request to /hj/php%20on%20going/Login.php. The packet details show the HTML Form URL Encoded data with fields for 'username' (gabugmail.com) and 'password' (12345). The packet bytes show the raw data of the request.

No.	Time	Source	Destination	Protocol	Length	Info
26	13.070	::1	::1	HTTP	661	POST /hj/php%20on%20going/Login.php HTTP/1.1 (application/x-www-form-urlencoded)
32	13.074	::1	::1	HTTP	1321	HTTP/1.1 200 OK (text/html)
76	13.186	::1	::1	HTTP	524	GET /hj/php%20on%20going/pphp.php HTTP/1.1
82	13.192	::1	::1	HTTP	995	HTTP/1.1 200 OK (text/html)
160	18.699	::1	::1	HTTP	632	POST /hj/php%20on%20going/pphp.php HTTP/1.1 (application/x-www-form-urlencoded)
166	18.703	::1	::1	HTTP	1037	HTTP/1.1 200 OK (text/html)
196	18.758	::1	::1	HTTP	524	GET /hj/php%20on%20going/login.php HTTP/1.1
200	18.764	::1	::1	HTTP	1279	HTTP/1.1 200 OK (text/html)
331	21.377	::1	::1	HTTP	640	POST /hj/php%20on%20going/login.php HTTP/1.1 (application/x-www-form-urlencoded)
333	21.382	::1	::1	HTTP	1320	HTTP/1.1 200 OK (text/html)
428	21.470	::1	::1	HTTP	524	GET /hj/php%20on%20going/pphp.php HTTP/1.1
437	21.478	::1	::1	HTTP	995	HTTP/1.1 200 OK (text/html)

Frame 26: 661 bytes on wire (5288 bits), 661 bytes captured (5288 bits) on interface \Device\NPF_{Loopback, id 0

Null/Loopback

Internet Protocol Version 6, Src: ::1, Dst: ::1

Transmission Control Protocol, Src Port: 50487, Dst Port: 80, Seq: 1, Ack: 1, Len: 597

Hypertext Transfer Protocol

HTML Form URL Encoded: application/x-www-form-urlencoded

- Form item: "username" = "gabugmail.com"
 - Key: username
 - Value: gabugmail.com
- Form item: "password" = "12345"
 - Key: password
 - Value: 12345

0000 18 00 00 00 60 00 00 00 02 69 06 80 00 00 00 00i.....
 0010 00 00 00 00 00 00 00 00 00 00 00 01 00 00 00 00
 0020 00 00 00 00 00 00 00 00 00 00 00 01 c5 37 00 507.P
 0030 12 d9 5d ff a4 d0 5f 30 50 18 01 02 8b 39 00 00 ...]...P...9..
 0040 50 4f 53 54 20 2f 68 6a 2f 70 68 70 25 32 30 6f POST /hj /php%20o
 0050 6e 25 32 30 67 6f 69 6e 67 2f 4c 6f 67 69 6e 2e n%20goi g/Login.
 0060 70 68 70 20 48 54 54 50 2f 31 2e 31 0d 0a 48 6f php HTTP /1.1 Ho
 0070 73 74 3a 20 6c 6f 63 61 6c 68 6f 73 74 0d 0a 55 st: loca lhost:l
 0080 73 65 72 2d 41 67 65 6e 74 3a 20 4d 6f 7a 69 6c ser-Agen t: Mozil
 0090 6c 61 2f 35 2e 30 20 28 57 69 6e 64 6f 77 73 20 la/5.0 (Windows
 00a0 4e 54 20 36 2e 33 3b 20 57 69 6e 36 34 3b 20 78 NT 6.3; Win64; x
 00b0 36 34 3b 20 72 76 3a 38 33 2e 30 29 20 47 65 63 64; rv:8.3.0) Ge

Fig 3.6

Wireshark packet capture showing an HTTP POST request for a login page. The packet details show the HTML Form URL Encoded data with fields for 'username' (gabugmail.com) and 'password' (12345).

Hypertext Transfer Protocol

HTML Form URL Encoded: application/x-www-form-urlencoded

- Form item: "username" = "gabugmail.com"
 - Key: username
 - Value: gabugmail.com
- Form item: "password" = "12345"
 - Key: password
 - Value: 12345

Fig 3.7

Practical-4

Aim: Simulating LAN configuration using the Cisco Packet tracer emulator.

Packet Tracer is a cross-platform visual simulation tool designed by Cisco Systems that allows users to create network topology and imitate modern computer network. 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. Previously students enrolled in a CCNA Academy program could freely download and use the tool free of charge for educational use.

Feature and Advantages of Cisco Packet Tracer :

- Provides a visual demonstration of complex technologies and configurations
- Allows instructors to author customized, guided activities that provide immediate feedback using the Activity Wizard
- Facilitates numerous learning activities such as lectures, individual and group lab activities, homework, assessments, games, network design, troubleshooting, modeling tasks, case studies, and competitions
- Enables visualization, animation, and detailed modeling for exploration, experimentation, and explanation
- Supports self-paced learning outside the classroom
- Supports social learning processes by enabling collaboration and competition
- Supports the majority of protocols and technologies taught in the following Networking Academy curricula: Cisco CCNA® Discovery, CCNA Exploration, and CCNA Security, and can be used to teach concepts from IT Essentials and Cisco CCNP® courses.

Cisco packet tracer supports following Protocols

Layer	Cisco Packet Tracer Supported Protocols
Application	FTP , SMTP, POP3, HTTP, TFTP, Telnet, SSH, DNS, DHCP, NTP, SNMP, AAA, ISR VOIP, SCCP config and calls ISR command support, Call Manager Express
Transport	TCP and UDP, TCP Nagle Algorithm & IP Fragmentation, RTP
Network	BGP, IPv4, ICMP, ARP, IPv6, ICMPv6, IPsec, RIPv1/ v2/ng, Multi-Area OSPF, EIGRP, Static Routing, Route Redistribution, Multilayer Switching, L3 QoS, NAT, CBAL , Zone-based policy firewall and Intrusion Protection System on the ISR, GRE VPN, IPsec VPN
Network Access / Interface	Ethernet (802.3), 802.11, HDLC, Frame Relay, PPP, PPPoE, STP, RSTP, VTP, DTP, CDP, 802.1q, PAgP, L2 QoS, SLARP, Simple WEP, WPA, EAP

WorkSpace :

Cisco Packet Tracer has two workspaces—logical and physical.

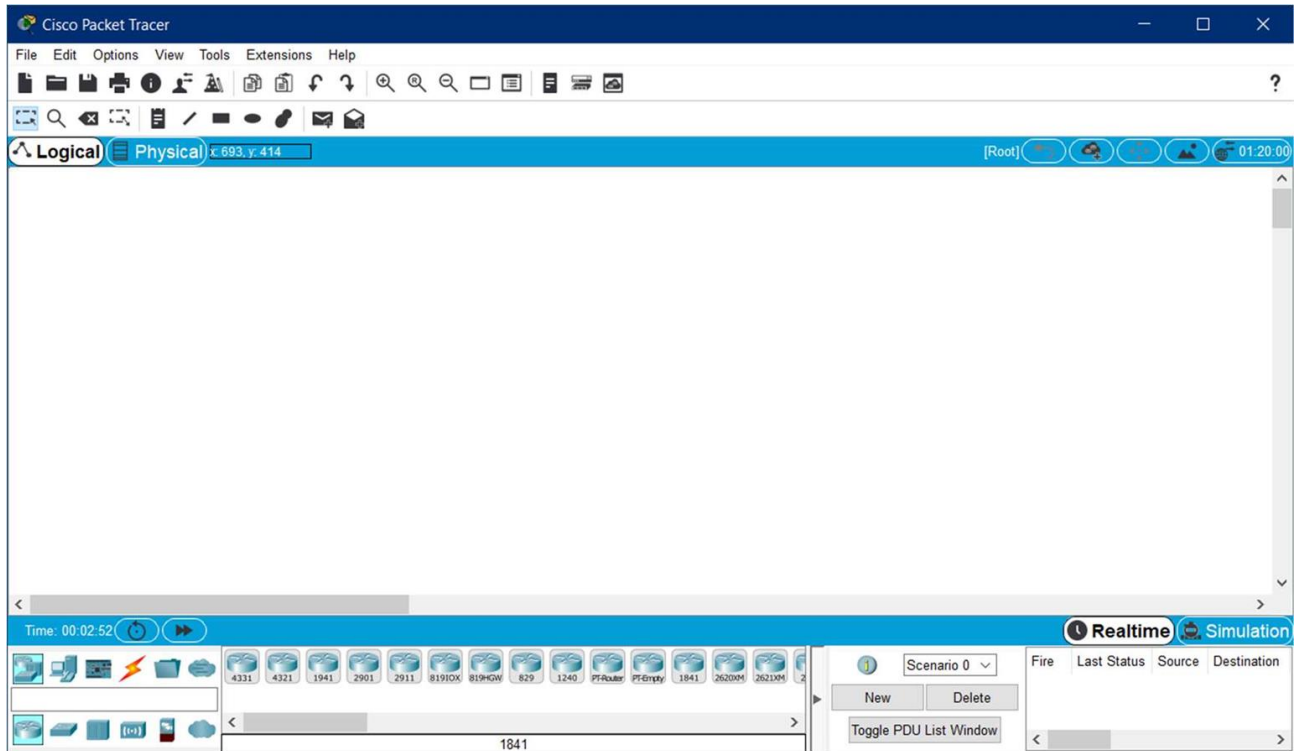
1) Logical Workspace :

The logical workspace allows users to build logical network topologies by placing, connecting, and clustering virtual network devices.

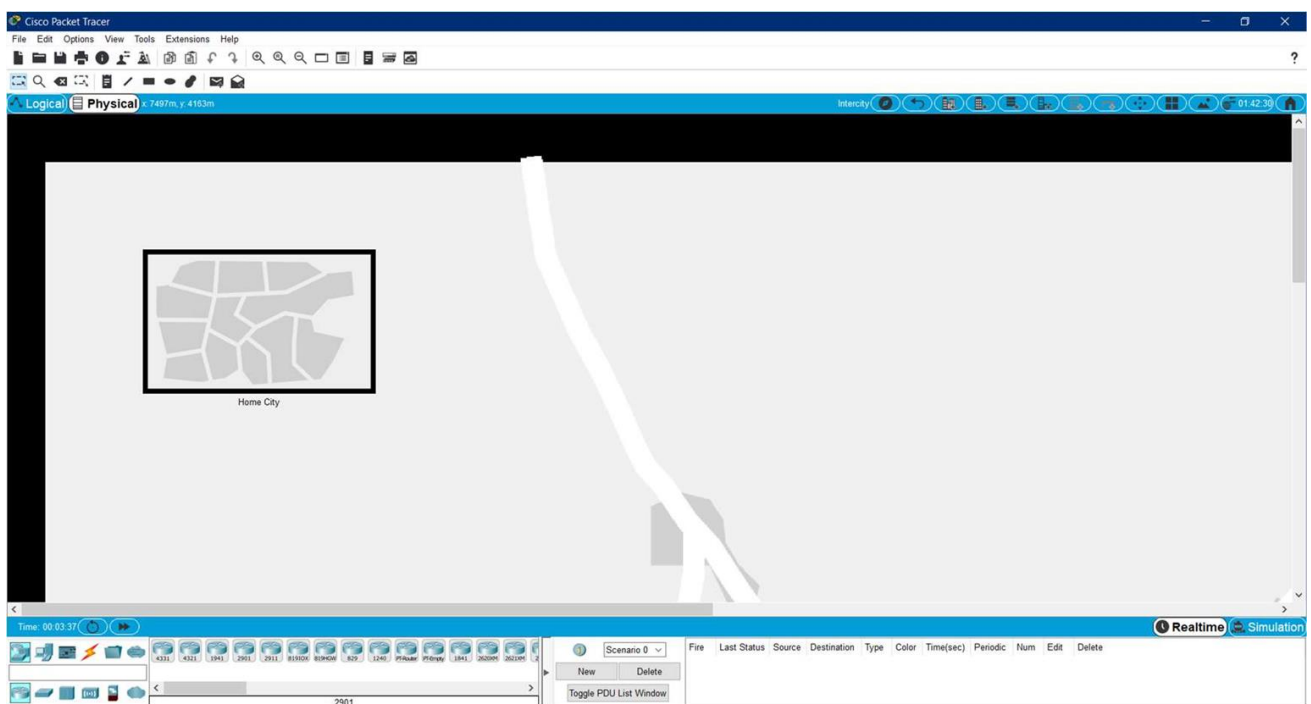
2) Physical Workspace :

The physical workspace provides a graphical physical dimension of the logical network, giving a sense of scale and placement in how network devices such as routers, switches, and hosts would look in a real environment. The physical view also provides geographic representations of networks, including multiple cities, buildings, and wiring closets.

Logical Workspace :



Physical Workspace :



Additional Features of Cisco Packet Tracer :

- Lab grading function

- International language support
- Compatible with the following platforms: Windows, Windows XP; Vista (Vista Basic, Vista Premium); Windows 7; and Linux (Ubuntu, Fedora)
- Available to registered Networking Academy instructors, students, and alumni

Star Topology :

A star topology is a topology for a Local Area Network (LAN) in which all nodes are individually connected to a central connection point, like a hub or a switch. A star takes more cable than e.g. a bus, but the benefit is that if a cable fails, only one node will be brought down.

All traffic emanates from the hub of the star. The central site is in control of all the nodes attached to it. The central hub is usually a fast, self contained computer and is responsible for routing all traffic to other nodes. The main advantages of a star network is that one malfunctioning node does not affect the rest of the network. However this type of network can be prone to bottleneck and failure problems at the central site.

A star network is often combined with a bus topology. The central hub is then connected to the backbone of the bus. This combination is called a tree.

Advantages and disadvantages of a star network :

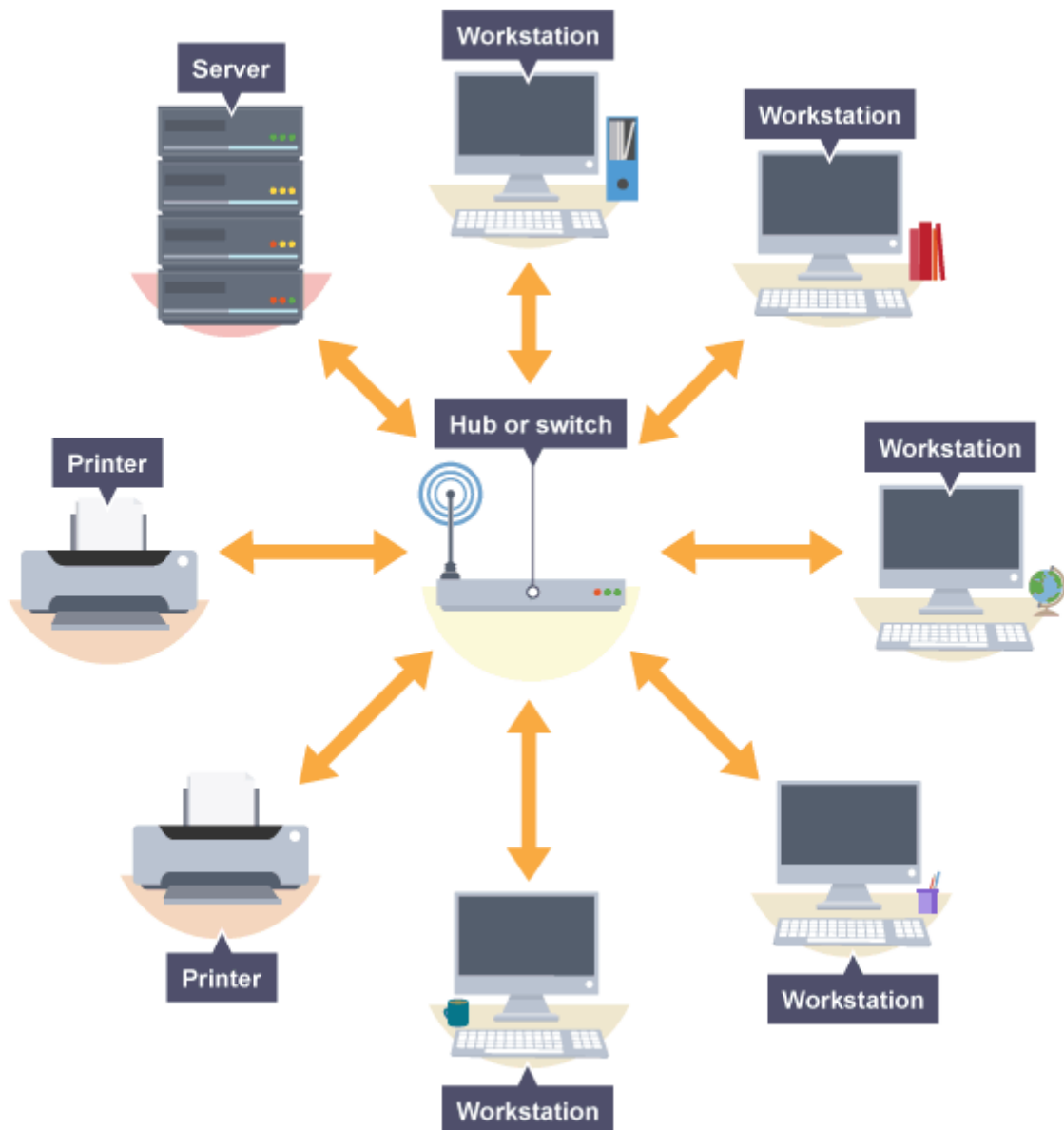
-->The advantages of a star network are:

- it is very reliable – if one cable or device fails then all the others will continue to work
 - it is high-performing as no data collisions can occur

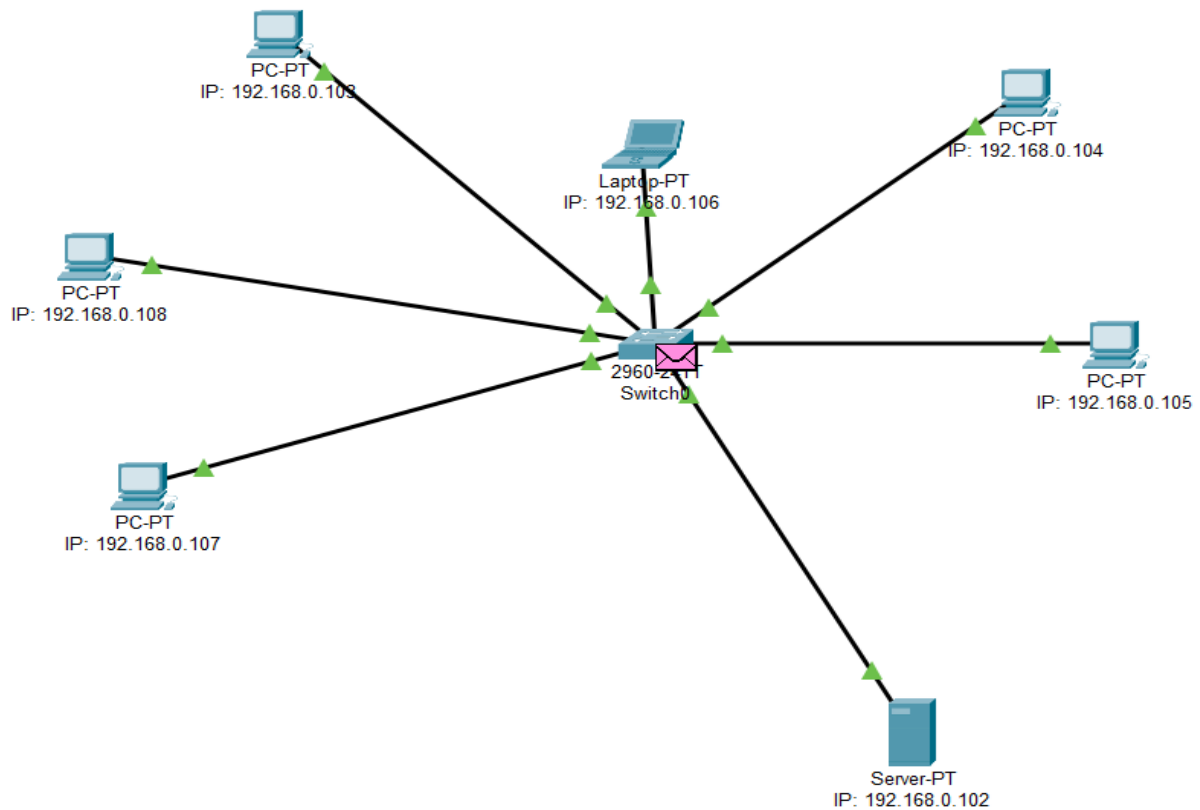
-->The disadvantages of a star network are:

- it is expensive to install as this type of network uses the most cable (network cable is expensive)
 - extra hardware is required (hubs or switches) which adds to cost
 - if a hub or switch fails, all the devices connected to it will have no network connection

Here is the picture of Star Topology in General Type .



Simulation of Star Topology in Cisco Packet Tracer



Simulation Panel				
Event List				
Vis.	Time(sec)	Last Device	At Device	Type
	0.000	--	IP: 192.168.0.106	ICMP
	0.000	--	IP: 192.168.0.108	ICMP
	0.001	IP: 192.168.0.106	Switch0	ICMP
	0.001	IP: 192.168.0.108	Switch0	ICMP
	0.003	Switch0	IP: 192.168.0.102	ICMP
	0.003	--	Switch0	ICMP
	0.004	IP: 192.168.0.102	Switch0	ICMP
	0.005	Switch0	IP: 192.168.0.102	ICMP
	0.006	Switch0	IP: 192.168.0.106	ICMP
	0.007	IP: 192.168.0.102	Switch0	ICMP
	0.009	Switch0	IP: 192.168.0.108	ICMP
Reset Simulation <input type="checkbox"/> Constant Delay Captured to: 0.009 s				

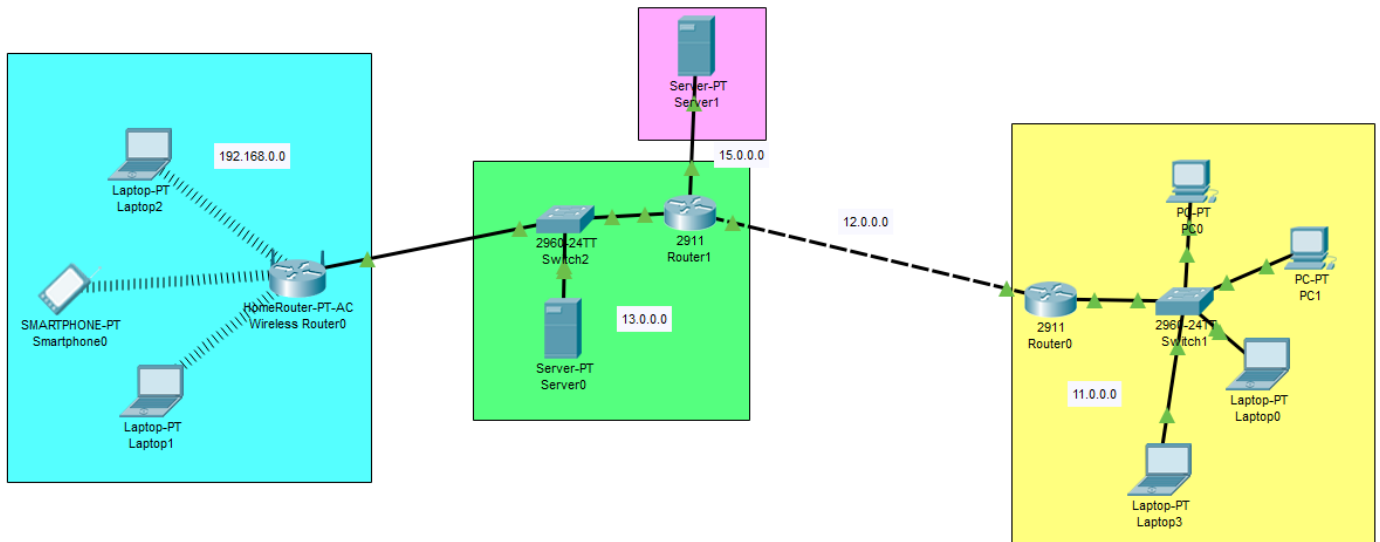
- Here in the Simulation Laptop with IP address of 192.168.0.106 wants to communicate with Server with IP address of 192.168.0.102.

- Because laptop and server both are connected with a common device i.e switch, the message will be passed via switch.
- Hence , message will leave laptop and then will be received by switch,which then will be passed to the server.Server will follow same method for giving acknowledgement back to laptop.

Hence , Cisco Packet Tracer allows us to do rea time networking in simulation mode so we can get the idea if we put this type of method or network in real world what will happen.

Practical - 5

AIM: Design a heterogeneous network which consist of wired and wireless LAN connected using router. Create a file server which allows user to download and upload the files.



Wireless LAN

Fig5.1

Wired LAN

File SERVER:



Fig5.2

Establish connection from wired LAN's PC to server, list out directories on server and upload the file 'testfile.txt':

```

Packet Tracer PC Command Line 1.0
C:\>ftp 15.0.0.1
Trying to connect...15.0.0.1
Connected to 15.0.0.1
220- Welcome to PT Ftp server
Username:sid
331- Username ok, need password
Password:
230- Logged in
(passive mode On)
ftp>dir

Listing /ftp directory from 15.0.0.1:
0   : asa842-k8.bin                      5571584
1   : asa923-k8.bin                      30468096
2   : c1841-advipservicesk9-mz.124-15.T1.bin 33591768
3   : c1841-ipbase-mz.123-14.T7.bin        13832032
4   : c1841-ipbasek9-mz.124-12.bin        16599160
5   : c1900-universalk9-mz.SPA.155-3.M4a.bin 33591768
6   : c2600-advipservicesk9-mz.124-15.T1.bin 33591768
7   : c2600-i-mz.122-28.bin              5571584
8   : c2600-ipbasek9-mz.124-8.bin         13169700
9   : c2800nm-advipservicesk9-mz.124-15.T1.bin 50938004
10  : c2800nm-advipservicesk9-mz.151-4.M4.bin 33591768
11  : c2800nm-ipbase-mz.123-14.T7.bin     5571584
12  : c2800nm-ipbasek9-mz.124-8.bin       15522644
13  : c2900-universalk9-mz.SPA.155-3.M4a.bin 33591768
14  : c2950-i6q4l2-mz.121-22.EA4.bin     3058048
15  : c2950-i6q4l2-mz.121-22.EA8.bin     3117390
16  : c2960-lanbase-mz.122-25.FX.bin     4414921
17  : c2960-lanbase-mz.122-25.SEE1.bin   4670455
18  : c2960-lanbasek9-mz.150-2.SE4.bin    4670455
19  : c3560-advipservicesk9-mz.122-37.SE1.bin 8662192
20  : c3560-advipservicesk9-mz.122-46.SE.bin 10713279
21  : c800-universalk9-mz.SPA.152-4.M4.bin 33591768
22  : c800-universalk9-mz.SPA.154-3.M6a.bin 83029236
23  : cat3k_caa-universalk9.16.03.02.SPA.bin 505532849
24  : cgr1000-universalk9-mz.SPA.154-2.CG 159487552
25  : cgr1000-universalk9-mz.SPA.156-3.CG 184530138
26  : ir800-universalk9-bundle.SPA.156-3.M.bin 160968869
27  : ir800-universalk9-mz.SPA.155-3.M    61750062
28  : ir800-universalk9-mz.SPA.156-3.M    63753767
29  : ir800_yocto-1.7.2.tar              2877440
30  : ir800_yocto-1.7.2_python-2.7.3.tar 6912000
31  : pt1000-i-mz.122-28.bin              5571584
32  : pt3000-i6q4l2-mz.121-22.EA4.bin     3117390
ftp>put testfile.txt

Writing file testfile.txt to 15.0.0.1:
File transfer in progress...

[Transfer complete - 49 bytes]

```

Fig5.3

Now check uploaded file is there:

```
ftp>dir

Listing /ftp directory from 15.0.0.1:
0   : asa842-k8.bin                5571584
1   : asa923-k8.bin                30468096
2   : c1841-advipservicesk9-mz.124-15.T1.bin 33591768
3   : c1841-ipbase-mz.123-14.T7.bin 13832032
4   : c1841-ipbasek9-mz.124-12.bin 16599160
5   : c1900-universalk9-mz.SPA.155-3.M4a.bin 33591768
6   : c2600-advipservicesk9-mz.124-15.T1.bin 33591768
7   : c2600-i-mz.122-28.bin        5571584
8   : c2600-ipbasek9-mz.124-8.bin  13169700
9   : c2800nm-advipservicesk9-mz.124-15.T1.bin 50938004
10  : c2800nm-advipservicesk9-mz.151-4.M4.bin 33591768
11  : c2800nm-ipbase-mz.123-14.T7.bin 5571584
12  : c2800nm-ipbasek9-mz.124-8.bin 15522644
13  : c2900-universalk9-mz.SPA.155-3.M4a.bin 33591768
14  : c2950-i6q4l2-mz.121-22.EA4.bin 3058048
15  : c2950-i6q4l2-mz.121-22.EA8.bin 3117390
16  : c2960-lanbase-mz.122-25.FX.bin 4414921
17  : c2960-lanbase-mz.122-25.SEE1.bin 4670455
18  : c2960-lanbasek9-mz.150-2.SE4.bin 4670455
19  : c3560-advipservicesk9-mz.122-37.SE1.bin 8662192
20  : c3560-advipservicesk9-mz.122-46.SE.bin 10713279
21  : c800-universalk9-mz.SPA.152-4.M4.bin 33591768
22  : c800-universalk9-mz.SPA.154-3.M6a.bin 83029236
23  : cat3k_caa-universalk9.16.03.02.SPA.bin 505532849
24  : cgr1000-universalk9-mz.SPA.154-2.CG 159487552
25  : cgr1000-universalk9-mz.SPA.156-3.CG 184530138
26  : ir800-universalk9-bundle.SPA.156-3.M.bin 160968869
27  : ir800-universalk9-mz.SPA.155-3.M 61750062
28  : ir800-universalk9-mz.SPA.156-3.M 63753767
29  : ir800_yocto-1.7.2.tar        2877440
30  : ir800_yocto-1.7.2_python-2.7.3.tar 6912000
31  : pt1000-i-mz.122-28.bin        5571584
32  : pt3000-i6q4l2-mz.121-22.EA4.bin 3117390
33  : testfile.txt                 49
ftp>
```

Fig5.4

Establish connection from wireless LAN's PC to server and download file 'testfile.txt':

```
Packet Tracer PC Command Line 1.0
C:\>
dir

Volume in drive C has no label.
Volume Serial Number is 5E12-4AF3
Directory of C:\

1/1/1970    5:30 PM                26      sampleFile.txt
                26 bytes                1 File(s)
C:\>ftp 15.0.0.1
Trying to connect...15.0.0.1
Connected to 15.0.0.1
220- Welcome to PT Ftp server
Username:sid
331- Username ok, need password
Password:
230- Logged in
(passive mode On)
ftp>get testfile.txt

Reading file testfile.txt from 15.0.0.1:
File transfer in progress...

[Transfer complete - 49 bytes]

221- Service closing control connection.
C:\>dir

Volume in drive C has no label.
Volume Serial Number is 5E12-4AF3
Directory of C:\

1/1/1970    5:30 PM                26      sampleFile.txt
1/1/1970    5:30 PM                49      testfile.txt
                75 bytes                2 File(s)
C:\>|
```

Fig5.5

Practical - 6

Aim: You have sub-netted your class C network 192.168.1.0 with a subnetmask of 255.255.255.252. Please list the following:

- 1) number of networks
- 2) number of hosts per network
- 3) the full range of the
- 4) first three networks, and the usable address range from those first three networks.

Additionally, identify the broadcast addresses for each network. Design this network in a packet tracer and simulate the working of the network. Save the Pkt file of your network design

Ip : 192.168.1.0 -> 192.168.1.00000000

Now subnet mask is 255.255.255.252 -> 255.255.255.11111100

192.168.1.00000000

-> subnet network id (6 bits)

-> host id (2 bits)

Number of Sub- Networks: $2^6 = 64$

Number Host per subnet : $2^2 - 2 = 2$

first three networks:

Subnet id	Usable Addresses	Broadcast id
192.168.1.0	192.168.1.1 192.168.1.2	192.168.1.3
192.168.1.4	192.168.1.5 192.168.1.6	192.168.1.7
192.168.1.8	192.168.1.9 192.168.1.10	192.168.1.11

Design Network in Cisco Packet Tracer for Email Server and Configure Devices for same. Show in Fig6.1

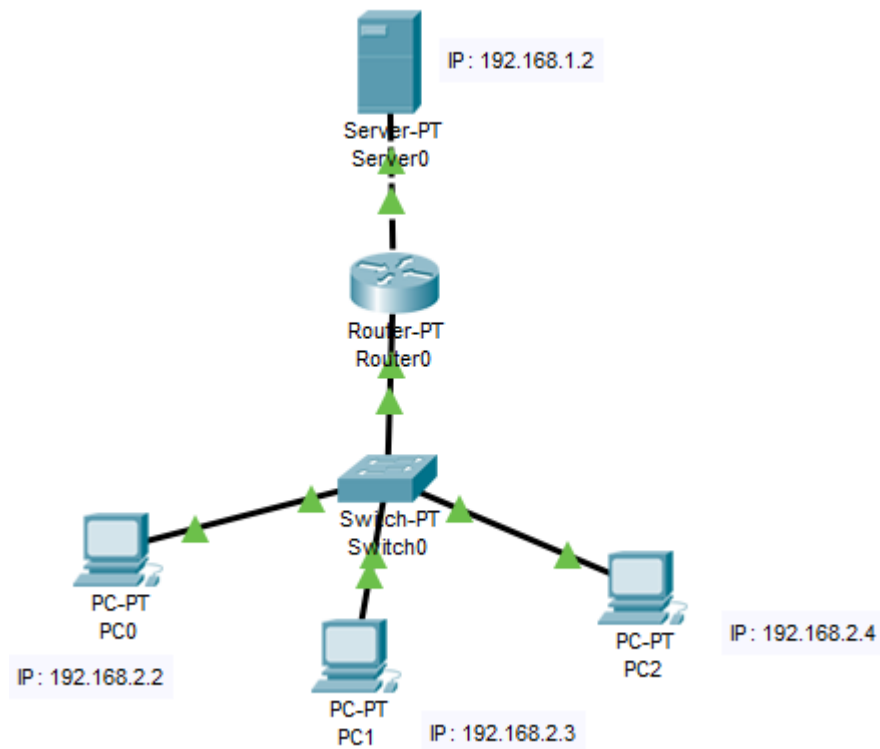


Fig6.1

Setting up Email configuration in on of network devices. Shown in Fig7.2

Physical Config **Desktop** Programming Attributes

Configure Mail [X]

User Information

Your Name:

Email Address:

Server Information

Incoming Mail Server:

Outgoing Mail Server:

Logon Information

User Name:

Password:

Save Clear Reset

Fig6.2

Composing mail from pc0 to pc1. Show in Fig6.3

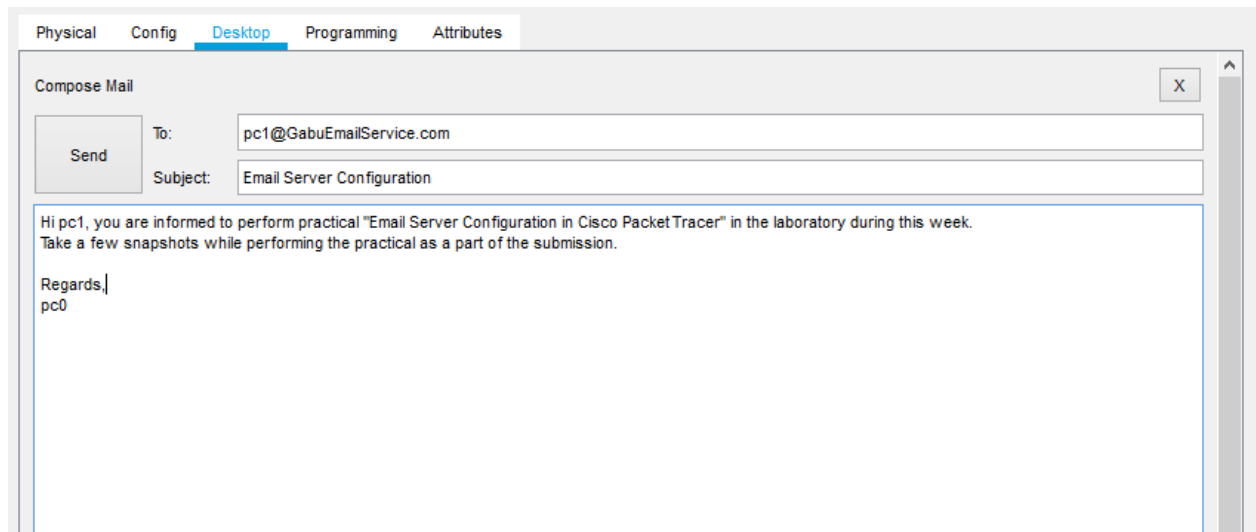


Fig6.3

Checking mail in pc1 that successfully send from pc0. Shown in Fig6.4

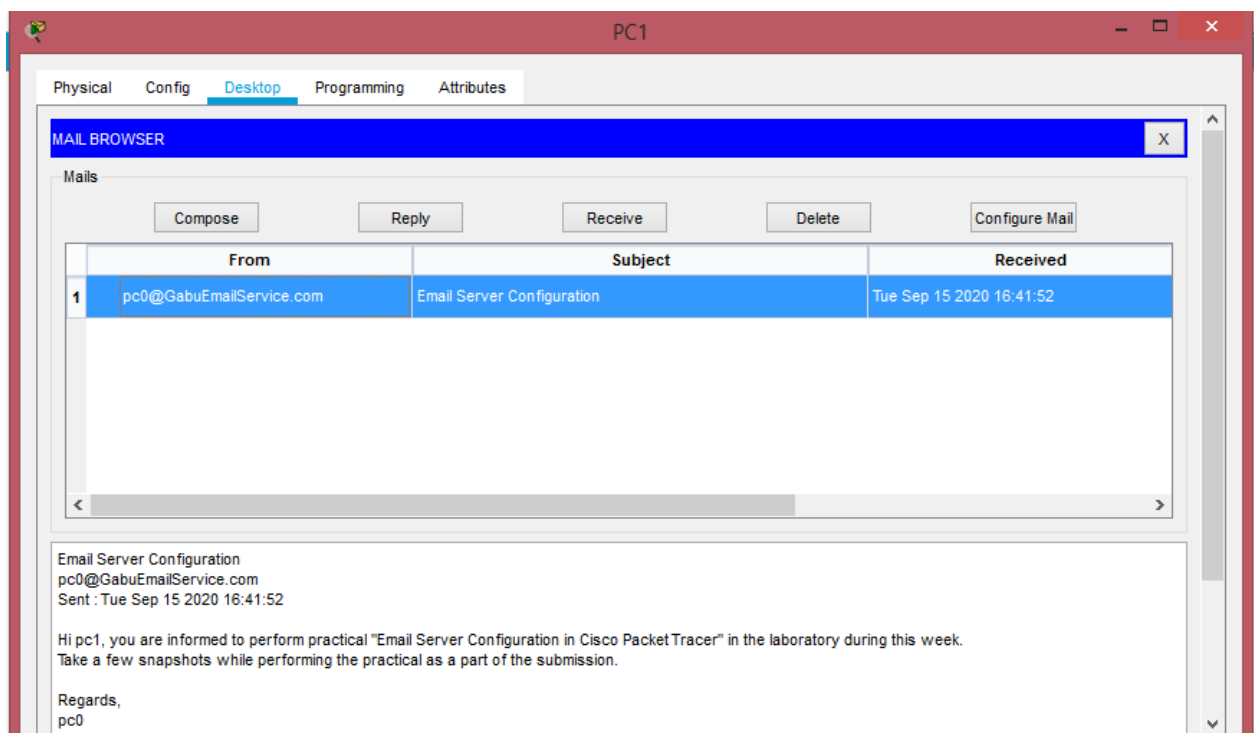


Fig6.4

Checking mail in pc2 that successfully send from pc1. Shown in Fig6.5

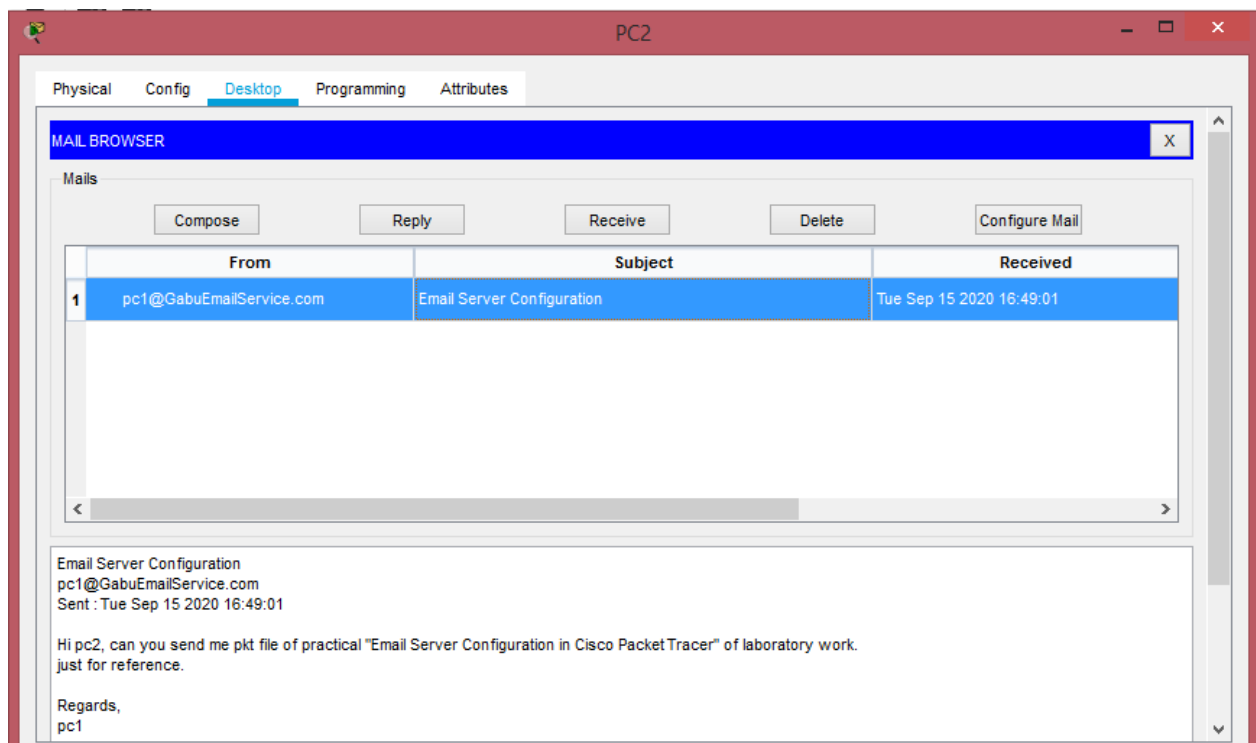


Fig6.5

Practical - 7

AIM: Router Configuration in Cisco Packet Tracer

Design Network in Cisco Packet Tracer for Router Configuration and Configure Devices for same.
Show in Fig7.1

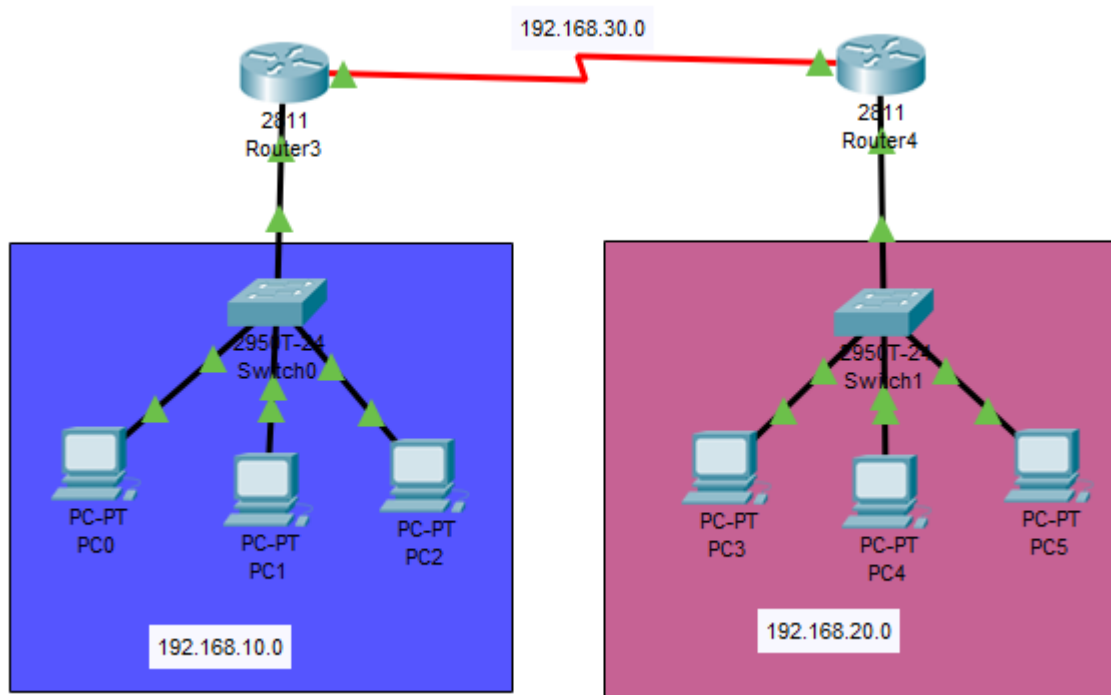


Fig7.1

Configuring Router to connect two different networks. Shown in Fig7.2

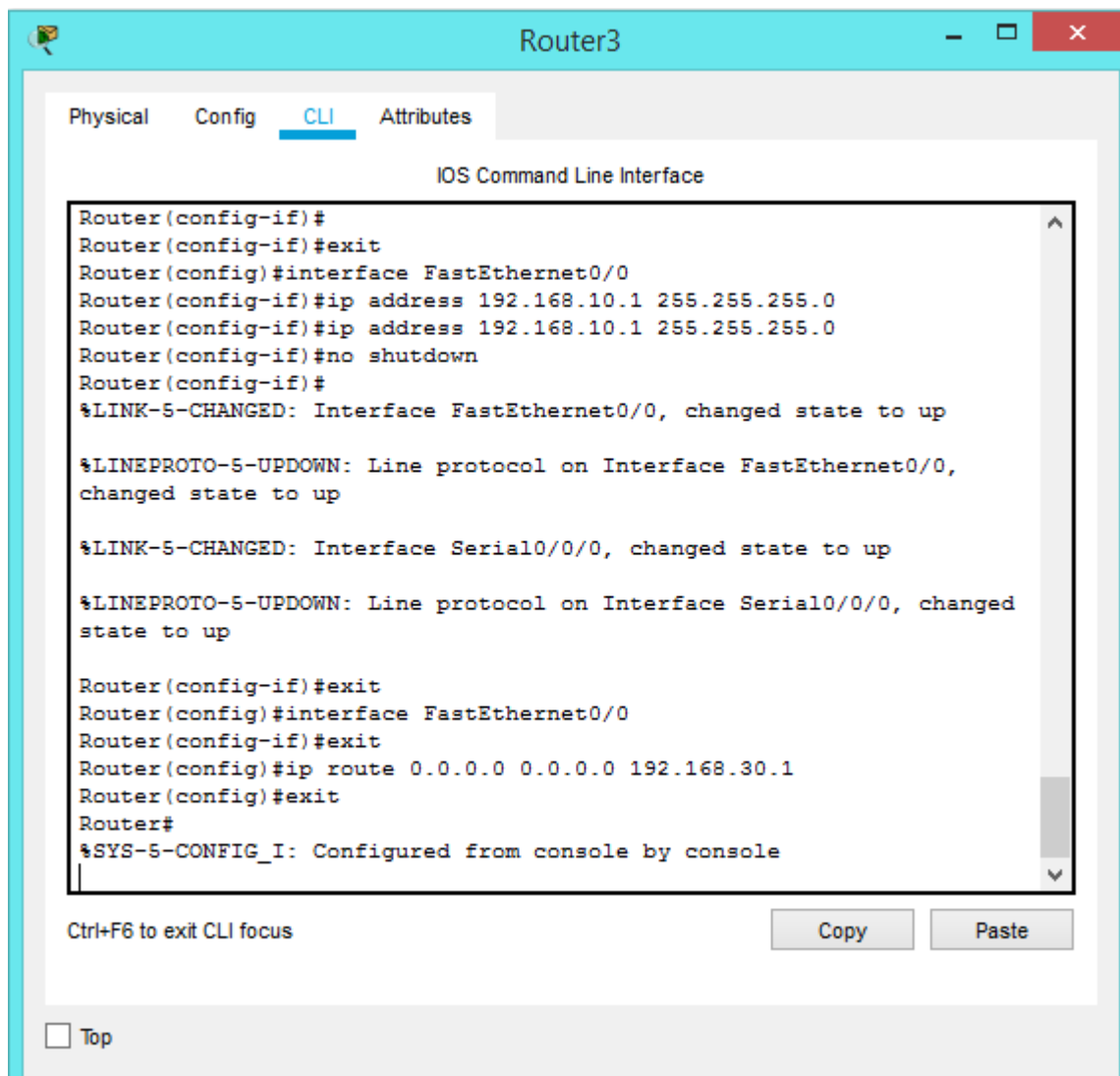


Fig7.2

After configuring Router checking ping command to ping another devices on network. Show in Fig7.3

```

Packet Tracer PC Command Line 1.0
C:\>ping 192.168.20.2

Pinging 192.168.20.2 with 32 bytes of data:

Reply from 192.168.20.2: bytes=32 time=2ms TTL=126
Reply from 192.168.20.2: bytes=32 time=12ms TTL=126
Reply from 192.168.20.2: bytes=32 time=12ms TTL=126
Reply from 192.168.20.2: bytes=32 time=12ms TTL=126

Ping statistics for 192.168.20.2:
    Packets: Sent = 4, Received = 4, Lost = 0 (0% loss),
Approximate round trip times in milli-seconds:
    Minimum = 2ms, Maximum = 12ms, Average = 9ms

C:\>ipconfig

FastEthernet0 Connection:(default port)

    Link-local IPv6 Address . . . . . : FE80::290:2BFF:FE7B:142B
    IP Address. . . . . : 192.168.10.2
    Subnet Mask . . . . . : 255.255.255.0
    Default Gateway . . . . . : 192.168.10.1

Bluetooth Connection:

```

Fig7.3

Sending Packet from one device to another device. Show in Fig7.4

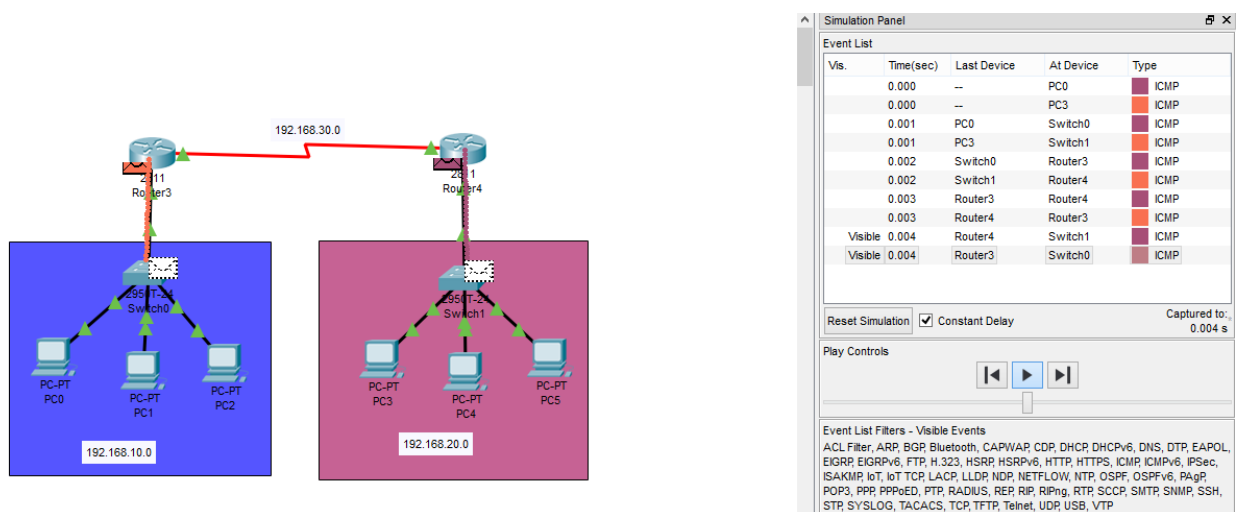


Fig7.4

Practical - 8

AIM: Implementing echo sever using Socket programming.

SERVER:

```
#include <stdio.h>
#include <stdlib.h>
#include <string.h>
#include <unistd.h>
#include <sys/types.h>
#include <sys/socket.h>
#include <netinet/in.h>

void error(const char *msg){
    perror(msg);
    exit(1);
}

int main(int argc,char *argv[]){
    if(argc<2){
        fprintf(stderr,"port not provided. program terminated\n");
        exit(1);
    }
    int sockfd, newsockfd, portno, n, clilen;
    char buffer[256];
    struct sockaddr_in serv_addr,cli_addr;
    sockfd = socket(AF_INET,SOCK_STREAM,0);
    if(sockfd<0){
        error("Error Openinn Socket.");
    }
    memset(&serv_addr,0,sizeof(struct sockaddr_in));
    portno = atoi(argv[1]);
    serv_addr.sin_family = AF_INET;
    serv_addr.sin_addr.s_addr = INADDR_ANY;
    serv_addr.sin_port = htons(portno);

    if(bind(sockfd,(struct sockaddr *)&serv_addr,sizeof(serv_addr))<0)
        error("Binding Faild.");

    listen(sockfd, 5);
    clilen = sizeof(cli_addr);

    newsockfd = accept(sockfd,(struct sockaddr *)&cli_addr,&clilen);
    if(newsockfd<0)error("Error on Accept\n");

    /* *****above code is for connection, only do pretty stuff below of this***** */

    while(1){
        memset(buffer,0,256);
        n = read(newsockfd,buffer,255);
        if(n<0)error("Error on reading");
```

```

        printf("Client: %s",buffer);
        memset(buffer,0,256);
        printf("Server: ");
        fgets(buffer,255,stdin);
        n = write(newsockfd,buffer,strlen(buffer));
        if(n<0)error("Erroe on writing");
        int i = strncmp("Bye",buffer,3);
        if(i==0)break;
    }

//Done
    close(newsockfd);
    close(sockfd);
    return 0;
}

```

CLIENT:

```

#include <stdio.h>
#include <stdlib.h>
#include <string.h>
#include <unistd.h>
#include <sys/types.h>
#include <sys/socket.h>
#include <netinet/in.h>
#include <netdb.h>

void error(const char *msg){
    perror(msg);
    exit(1);
}

int main(int argc,char *argv[]){

    if(argc<3){
        fprintf(stderr,"usage %s hostname port\n",argv[0]);
        exit(1);
    }

    int sockfd, portno, n;
    char buffer[256];

    struct sockaddr_in serv_addr;
    struct hostent *server;
    sockfd = socket(AF_INET,SOCK_STREAM,0);
    if(sockfd<0){
        error("Error Openinn Socket.");
    }
    memset(&serv_addr,0,sizeof(serv_addr));

    portno = atoi(argv[2]);

    server = gethostbyname(argv[1]);

```

```

if(server == NULL){
    fprintf(stderr,"Error , no such host");
    exit(1);
}

serv_addr.sin_family = AF_INET;
bcopy((char *)server->h_addr,(char *)&serv_addr.sin_addr.s_addr,server->h_length);
serv_addr.sin_port = htons(portno);

if(connect(sockfd,(struct sockaddr *)&serv_addr,sizeof(serv_addr))<0)error("connection fail");
while(1){
    memset(buffer,0,256);
    printf("Client: ");
    fgets(buffer,255,stdin);
    n = write(sockfd,buffer,strlen(buffer));
    if(n<0)error("Error on writing");

    memset(buffer,0,256);
    n = read(sockfd,buffer,255);
    if(n<0)error("Error on reading");
    printf("Server: %s",buffer);

    int i = strncmp("Bye",buffer,3);
    if(i==0)break;
}

//Done
close(sockfd);
return 0;
}

```

The image shows two terminal windows side-by-side, demonstrating a client-server interaction. Both windows are titled 'sidpro@sidpro-Inspiron-3542: ~/Desktop'.

Left Terminal (Client):

```

sidpro@sidpro-Inspiron-3542:~/Desktop$ ./C localhost 6988
Client: Hello!
Server: hello, how are you?
Client: the best i can be. assuming you're at your best too
Server: yeah
Client: bye
Server: Bye
sidpro@sidpro-Inspiron-3542:~/Desktop$ 

```

Right Terminal (Server):

```

sidpro@sidpro-Inspiron-3542:~/Desktop$ ./S 6988
Client: Hello!
Server: hello, how are you?
Client: the best i can be. assuming you're at your best too
Server: yeah
Client: bye
Server: Bye
sidpro@sidpro-Inspiron-3542:~/Desktop$ 

```

Practical - 9

AIM: Implementing file server using socket programming.

SERVER:

```
#include <stdio.h>
#include <stdlib.h>
#include <string.h>
#include <unistd.h>
#include <sys/types.h>
#include <sys/socket.h>
#include <netinet/in.h>
#include <ctype.h>
void error(const char *msg){
    perror(msg);
    exit(1);
}
int main(int argc,char *argv[]){
    if(argc<2){
        fprintf(stderr,"port not provided. program terminated\n");
        exit(1);
    }
    int sockfd, newsockfd, portno, n, clilen;
    char buffer[256];
    struct sockaddr_in serv_addr,cli_addr;
    sockfd = socket(AF_INET,SOCK_STREAM,0);
    if(sockfd<0){
        error("Error Openinn Socket.");
    }
    memset(&serv_addr,0,sizeof(struct sockaddr_in));
    portno = atoi(argv[1]);
    serv_addr.sin_family = AF_INET;
    serv_addr.sin_addr.s_addr = INADDR_ANY;
    serv_addr.sin_port = htons(portno);
    if(bind(sockfd,(struct sockaddr *)&serv_addr,sizeof(serv_addr))<0)
        error("Binding Faild.");
    listen(sockfd, 5);
    clilen = sizeof(cli_addr);

    newsockfd = accept(sockfd,(struct sockaddr *)&cli_addr,&clilen);
    if(newsockfd<0)error("Error on Accept\n");

    /* *****above code is for connection only, do pretty stuff below of this***** */
    FILE *fp;
    int words;
    int word = 0;
    fp = fopen("testrecivied.txt","a");
    char ch;
    read(newsockfd,&words,sizeof(int));
    while(word!=words){
```

```

        read(newsockfd,&ch,sizeof(char));
        fprintf(fp,"%c",ch);
        word++;
    }
    printf("\nThe file has been successfully received. It saved by name \" testreceived.txt\" \n");
//Done
    close(newsockfd);
    close(sockfd);
    return 0;
}

```

CLIENT:

```

#include <stdio.h>
#include <stdlib.h>
#include <string.h>
#include <unistd.h>
#include <sys/types.h>
#include <sys/socket.h>
#include <netinet/in.h>
#include <netdb.h>
#include <ctype.h>
void error(const char *msg){
    perror(msg);
    exit(1);
}
int main(int argc,char *argv[]){
    if(argc<3){
        fprintf(stderr,"usage %s hostname port\n",argv[0]);
        exit(1);
    }
    int sockfd, portno, n;
    char buffer[256];

    struct sockaddr_in serv_addr;
    struct hostent *server;
    sockfd = socket(AF_INET,SOCK_STREAM,0);
    if(sockfd<0){
        error("Error Opening Socket.");
    }
    memset(&serv_addr,0,sizeof(serv_addr));

    portno = atoi(argv[2]);

    server = gethostbyname(argv[1]);
    if(server == NULL){
        fprintf(stderr,"Error , no such host");
        exit(1);
    }
    serv_addr.sin_family = AF_INET;

```

```
bcopy((char *)server->h_addr,(char *)&serv_addr.sin_addr.s_addr,server->h_length);
serv_addr.sin_port = htons(portno);
```

```
if(connect(sockfd,(struct sockaddr *)&serv_addr,sizeof(serv_addr))<0)error("connection
failed");
```

```
/* *****above code is for connection, only do pretty stuff below of this***** */
```

```
FILE *fp;
int words = 0;
fp = fopen("test.txt","r");
char ch;
while((ch=fgetc(fp))!=EOF){
    words++;
}
printf("Words: %d\n",words);
write(sockfd,&words,sizeof(int));
rewind(fp);
char cp;
while((cp=fgetc(fp))!=EOF){
    write(sockfd,&cp,sizeof(char));
}
printf("The file has been successfully sent. Thank you\n");
//Done
close(sockfd);
return 0;
}
```

The image shows two terminal windows side-by-side, demonstrating a file transfer process between a server and a client.

Left Terminal (Server Side):

```

sidpro@sidpro-Inspiron-3542: ~/Desktop
File Edit View Search Terminal Help
sidpro@sidpro-Inspiron-3542:~/Desktop$ ls
C      FileServer.c  quine      std
Client FS         quine.c    stdoutandstderr.c
ClientbySid.c HammingC     S          temp.txt
client.c HammingClient.c Server      test.txt
FC      HammingS     ServerbySid.c
FileClient.c HammingServer.c server.c
sidpro@sidpro-Inspiron-3542:~/Desktop$ cat test.txt
hello buddy!
gabru siddharth is here,
for IT support.
glad to help.
sidpro@sidpro-Inspiron-3542:~/Desktop$ ./FC 127.0.0.1 6988
Words: 67
The file has been successfully sent. Thank you
sidpro@sidpro-Inspiron-3542:~/Desktop$

```

Right Terminal (Client Side):

```

sidpro@sidpro-Inspiron-3542: ~/Desktop
File Edit View Search Terminal Help
sidpro@sidpro-Inspiron-3542:~/Desktop$ cat testrecivied.txt
cat: testrecivied.txt: No such file or directory
sidpro@sidpro-Inspiron-3542:~/Desktop$ ./FS 6988
The file has been successfully relvied. It saved by name " testrecivied.txt"
sidpro@sidpro-Inspiron-3542:~/Desktop$ ls
C      FileServer.c  quine      std
Client FS         quine.c    stdoutandstderr.c
ClientbySid.c HammingC     S          temp.txt
client.c HammingClient.c Server      testrecivied.txt
FC      HammingS     ServerbySid.c test.txt
FileClient.c HammingServer.c server.c
sidpro@sidpro-Inspiron-3542:~/Desktop$ cat testrecivied.txt
hello buddy!
gabru siddharth is here,
for IT support.
glad to help.
sidpro@sidpro-Inspiron-3542:~/Desktop$

```

Practical - 10

AIM: Implementing Error Detection mechanism using Socket Programming.

SERVER:

```
#include <stdio.h>
#include <stdlib.h>
#include <string.h>
#include <unistd.h>
#include <sys/types.h>
#include <sys/socket.h>
#include <netinet/in.h>

void error(const char *msg){
    perror(msg);
    exit(1);
}

int main(int argc,char *argv[]){

    if(argc<2){
        fprintf(stderr,"port not provided. program terminated\n");
        exit(1);
    }
    int sockfd, portno, n;
    socklen_t clilen;
    char buffer[256];
    memset(buffer,0,256);
    struct sockaddr_in serv_addr,cli_addr;
    sockfd = socket(AF_INET,SOCK_DGRAM,0);
    if(sockfd<0){
        error("Error Openinn Socket.");
    }
    memset(&serv_addr,0,sizeof(struct sockaddr_in));
    portno = atoi(argv[1]);
    serv_addr.sin_family = AF_INET;
    serv_addr.sin_addr.s_addr = INADDR_ANY;
    serv_addr.sin_port = htons(portno);
    if(bind(sockfd,(struct sockaddr *)&serv_addr,sizeof(serv_addr))<0)
        error("Binding Faild.");

    clilen = sizeof(cli_addr);

    //recvfrom(sockfd,buffer,255,0,(struct sockaddr *)&cli_addr,&clilen);
    //printf("\nClient : %s",buffer);

    /* *****above code is for connection, only do pretty stuff below of this***** */

    int data[10];
    int test[10];
    int p,p1,p2,p4;
    printf("The data received is: ");
```



```

for(int i=0;i<7;i++){
    n = recvfrom(sockfd,&data[i],sizeof(data[i]),0,(struct sockaddr *)&cli_addr,&clilen);
    if(n<0)printf("Error while receiving");
    printf("%d",data[i]);
}
printf("\nEnter data to tested: \n");
for(int i=0;i<7;i++){
    scanf("%d",&test[i]);
}
p1= test[6]^test[4]^test[2]^test[0];
p2= test[5]^test[4]^test[1]^test[0];
p4= test[3]^test[2]^test[1]^test[0];

p = (4*p4)+(2*p2)+p1;
printf("The data for testing is: ");
    for(int i=0;i<7;i++){
        printf("%d",test[i]);
    }
if(p==0){printf("No Error");}
else{
    printf("\nThe Error is at position %d",p);
    printf("\nThe correct data is:");
    if(test[7-p]==0)
        test[7-p]=1;
    else
        test[7-p]=0;

    for(int i=0;i<7;i++){
        printf("%d",test[i]);
    }
}
printf("\n");

//Done
    //close(newsockfd);
    close(sockfd);
    return 0;
}

```

CLIENT:

```

#include <stdio.h>
#include <stdlib.h>
#include <string.h>
#include <unistd.h>
#include <sys/types.h>
#include <sys/socket.h>
#include <netinet/in.h>
#include <netdb.h>

void error(const char *msg){
    perror(msg);
    exit(1);
}

```

```

}
int main(int argc,char *argv[]){

    if(argc<3){
        fprintf(stderr,"usage %s hostname port\n",argv[0]);
        exit(1);
    }
    int sockfd, portno, n;
    char buffer[256];
    socklen_t len;
    struct sockaddr_in serv_addr;
    struct hostent *server;
    sockfd = socket(AF_INET,SOCK_DGRAM,0);
    if(sockfd<0){
        error("Error Openinn Socket.");
    }
    memset(&serv_addr,0,sizeof(serv_addr));

    portno = atoi(argv[2]);

    server = gethostbyname(argv[1]);
    if(server == NULL){
        fprintf(stderr,"Error , no such host");
        exit(1);
    }

    serv_addr.sin_family = AF_INET;
    bcopy((char *)server->h_addr,(char *)&serv_addr.sin_addr.s_addr,server->h_length);
    serv_addr.sin_port = htons(portno);
    len = sizeof(serv_addr);
    //sendto(sockfd,buffer,,0,(struct sockaddr *)&serv_addr,sizeof(serv_addr));

/* *****above code is for connection, only do pretty stuff below of this***** */

    int data[10];
    printf("Plese input 4 bits of data\n");
    scanf("%d",&data[0]);
    scanf("%d",&data[1]);
    scanf("%d",&data[2]);
    scanf("%d",&data[4]);

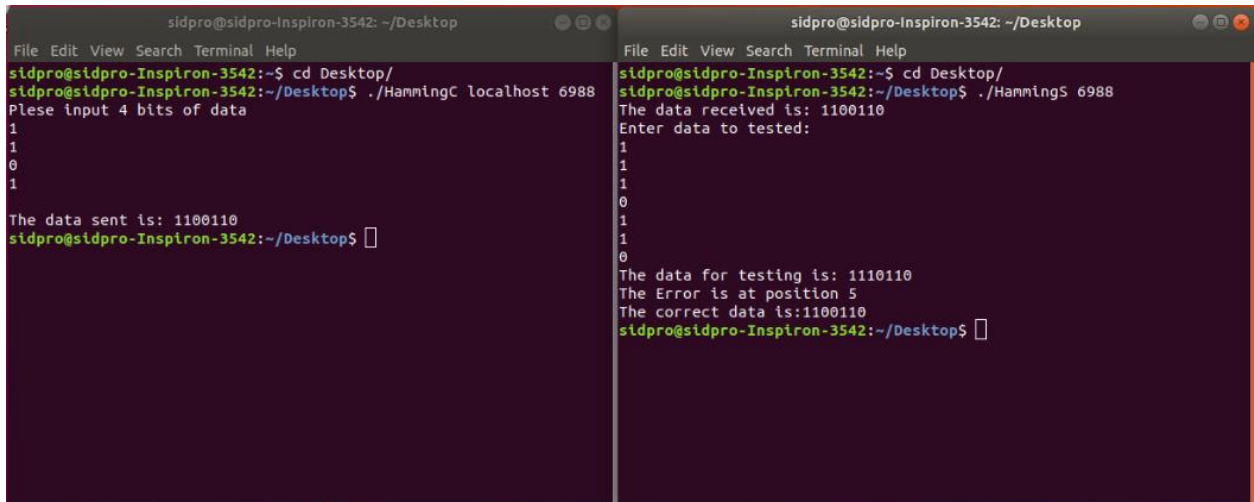
    // calculation fro even parity encoding
    data[6]= data[4]^data[2]^data[0];
    data[5]= data[4]^data[1]^data[0];
    data[3]= data[2]^data[1]^data[0];

    //sending the encoded data to clients
    for(int i=0; i<7; i++){
        n = sendto(sockfd,&data[i],sizeof(data[i]),0,(struct sockaddr *)&serv_addr,len);
        if(n<0)printf("Error while sending");
    }

    printf("\nThe data sent is: ");
    for(int i=0; i<7; i++){
        printf("%d",data[i]);
    }

```

```
}  
printf("\n");  
//Done  
    close(sockfd);  
    return 0;  
}
```



```
sidpro@sidpro-Inspiron-3542: ~/Desktop  
File Edit View Search Terminal Help  
sidpro@sidpro-Inspiron-3542:~$ cd Desktop/  
sidpro@sidpro-Inspiron-3542:~/Desktop$ ./HammingC localhost 6988  
Plese input 4 bits of data  
1  
1  
0  
1  
  
The data sent is: 1100110  
sidpro@sidpro-Inspiron-3542:~/Desktop$  
  
sidpro@sidpro-Inspiron-3542: ~/Desktop  
File Edit View Search Terminal Help  
sidpro@sidpro-Inspiron-3542:~$ cd Desktop/  
sidpro@sidpro-Inspiron-3542:~/Desktop$ ./HammingS 6988  
The data received is: 1100110  
Enter data to tested:  
1  
1  
1  
0  
1  
1  
0  
The data for testing is: 1110110  
The Error is at position 5  
The correct data is:1100110  
sidpro@sidpro-Inspiron-3542:~/Desktop$
```