Dr. D. Y. Patil Pratishthan's

DR. D. Y. PATIL INSTITUTE OF ENGINEERING, MANAGEMENT & RESEARCH

Approved by A.I.C.T.E, New Delhi , Maharashtra State Government, Affiliated to Savitribai Phule Pune University
Sector No. 29, PCNTDA , Nigidi Pradhikaran, Akurdi, Pune 411044. Phone: 020–27654470, Fax: 020-27656566
Website :www.dypiemr.ac.in Email : principal.dypiemr@gmail.com

Department of Artificial Intelligence and Data Science

LAB MANUAL
Computer Network Laboratory (TE)
Semester I

Prepared by:
Mrs. Sabha Dharawadkar

Computer Network Laboratory

Course Code	Course Name	Teaching Scheme (Hrs./ Week)	Credits
317524	Computer Network Laboratory	2	1

Course Objectives:

- To learn computer network hardware and software components
- To learn computer network topologies and types of network
- To develop an understanding of various protocols, modern technologies and applications
- To learn modern tools for network traffic analysis
- To learn network programming

.

Course Outcomes:

On completion of the course, learner will be able to–

- CO1: Analyze the requirements of network types, topology and transmission media
- CO2: Demonstrate error control, flow control techniques and protocols and analyze them
- CO3: Demonstrate the subnet formation with IP allocation mechanism and apply various routing algorithms
- CO4: Develop Client-Server architectures and prototypes
- CO5: Implement web applications and services using application layer protocols

Operating System recommended: 64-bit Open source Linux or its derivative

Table of Contents

Group A

Lab Assignment No.	1
Title	Demonstrate the different types of topologies and types of transmission media by using a packet tracer tool.
Roll No.	
Class	TE
Date of Completion	
Subject	Computer Network Laboratory
Assessment Marks	
Assessor's Sign	

ASSIGNMENT No: 01

Title: Demonstrate the different types of topologies and types of transmission media.

Problem Statement: Demonstrate the different types of topologies and types of transmission media by using a packet tracer tool.

Prerequisite:

a) IP Address

b) OSI & TCP/IP Model

c) Networking Devices

Objective:

- Understand how to Create different topology.

Outcomes:

- Students could run the practical for all topologies in Packet tracer

Theory:

Introduction

What is Topology?

Network topologies describe the methods in which all the elements of a network are mapped. The topology term refers to both the physical and logical layout of a network.

Type of Network Topology

1. Point to Point
2. Bus Topology
3. Ring Topology
4. Star Topology
5. Mesh Topology
6. Tree Topology
7. Hybrid Topology

How to select a Network Topology?

Types of Networking Topologies

Two main types of network topologies in computer networks are

1) Physical topology 2) Logical topology

1) Physical topology:

This type of network is an actual layout of the computer cables and other network devices

2) Logical topology:

Logical topology gives insight's about network's physical design.

Different types of Physical Topologies are:

P2P Topology

Bus Topology

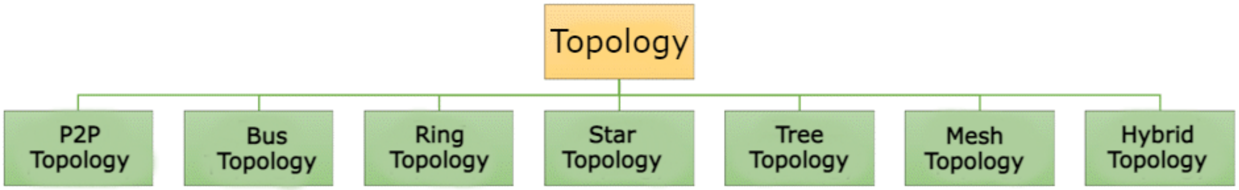
Ring Topology

Star Topology

Tree Topology

Mesh Topology

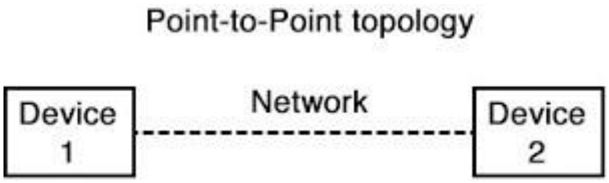
Hybrid Topology



Network Topology Diagram

Point to Point

Point-to-point topology is the easiest of all the network topologies. In this method, the network consists of a direct link between two computers.



P2P Topology Diagram

Advantages:

This is faster and highly reliable than other types of connections since there is a direct connection.

No need for a network operating system

Does not need an expensive server as individual workstations are used to access the files

No need for any dedicated network technicians because each user sets their permissions

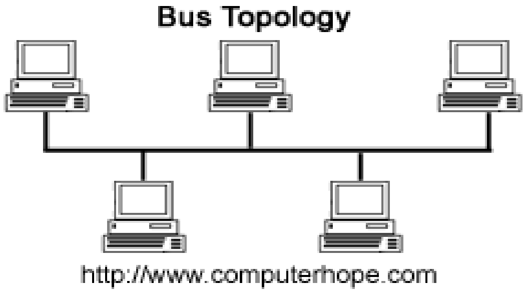
Disadvantages:

The biggest drawback is that it only be used for small areas where computers are in close proximity.

You can’t back up files and folders centrally

There is no security besides the permissions. Users often do not require to log onto their workstations.

Bus Topology



Bus Topology Diagram

Bus topology uses a single cable which connects all the included nodes. The main cable acts as a spine for the entire network. One of the computers in the network acts as the computer server. When it has two endpoints, it is known as a linear bus topology.

Advantages:

Here are pros/benefits of using a bus topology:

Cost of the cable is very less as compared to other topology, so it is widely used to build small networks.

Famous for LAN network because they are inexpensive and easy to install.

It is widely used when a network installation is small, simple, or temporary.

It is one of the passive topologies. So computers on the bus only listen for data being sent, that are not responsible for moving the data from one computer to others.

Disadvantages:

Here are the cons/drawbacks of bus topology:

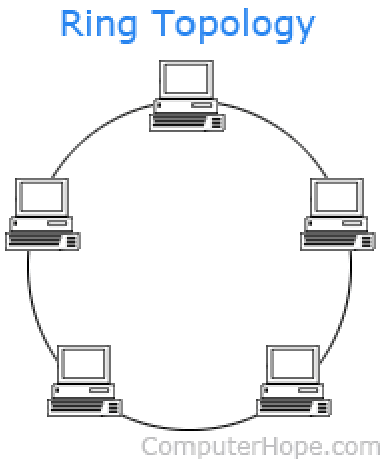
In case if the common cable fails, then the entire system will crash down.

When network traffic is heavy, it develops collisions in the network.

Whenever network traffic is heavy, or nodes are too many, the performance time of the network significantly decreases.

Cables are always of a limited length.

Ring Topology



Ring Topology Diagram

In a ring network, every device has exactly two neighboring devices for communication purpose. It is called a ring topology as its formation is like a ring. In this topology, every computer is connected to another computer. Here, the last node is combined with a first one.

This topology uses token to pass the information from one computer to another. In this topology, all the messages travel through a ring in the same direction.

Advantages:

Here are pros/benefits of ring topology:

Easy to install and reconfigure.

Adding or deleting a device in-ring topology needs you to move only two connections.

The troubleshooting process is difficult in a ring topology.

Failure of one computer can disturb the whole network.

Offers equal access to all the computers of the networks

Faster error checking and acknowledgment.

Disadvantages:

Here are drawbacks/cons of ring topology:

Unidirectional traffic.

Break in a single ring can risk the breaking of the entire network

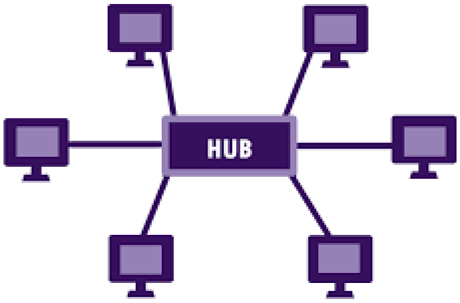
Modern days high-speed LANs made this topology less popular.

In the ring, topology signals are circulating at all times, which develops unwanted power consumption.

It is very difficult to troubleshoot the ring network.

Adding or removing the computers can disturb the network activity.

Star Topology



Star Topology Diagram

In the star topology, all the computers connect with the help of a hub. This cable is called a central node, and all other nodes are connected using this central node. It is most popular on LAN networks as they are inexpensive

and easy to install.

Advantages:

Here are pros/benefits of start topology:

Easy to troubleshoot, set up, and modify.

Only those nodes are affected, that has failed. Other nodes still work.

Fast performance with few nodes and very low network traffic.

In Star topology, addition, deletion, and moving of the devices are easy.

Disadvantages:

Here are cons/drawbacks of using Star:

If the hub or concentrator fails, attached nodes are disabled.

Cost of installation of star topology is costly.

Heavy network traffic can sometimes slow the bus considerably.

Performance depends on the hub’s capacity

A damaged cable or lack of proper termination may bring the network down.

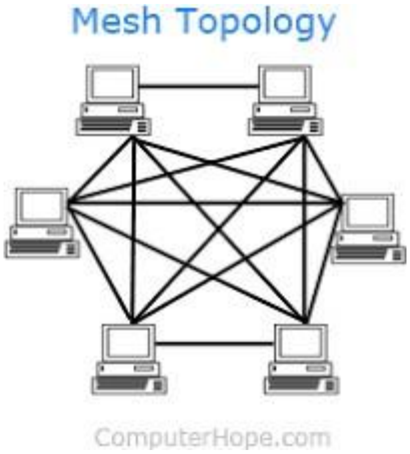
Mesh Topology

The mesh topology has a unique network design in which each computer on the network connects to every other.

It is develops a P2P (point-to-point) connection between all the devices of the network. It offers a high level of redundancy, so even if one network cable fails, still data has an alternative path to reach its destination.

Mesh Topology:

In this topology, every nodes or device are directly connected with each other.



Mesh Topology

Advantages:

Here, are pros/benefits of Mesh topology

The network can be expanded without disrupting current users.

Need extra capable compared with other LAN topologies.

No traffic problem as nodes has dedicated links.

Dedicated links help you to eliminate the traffic problem.

A mesh topology is robust.

It has multiple links, so if any single route is blocked, then other routes should be used for data communication.

P2P links make the fault identification isolation process easy.

It helps you to avoid the chances of network failure by connecting all the systems to a central node.

Every system has its privacy and security.

Disadvantages:

Installation is complex because every node is connected to every node.

It is expensive due to the use of more cables. No proper utilization of systems.

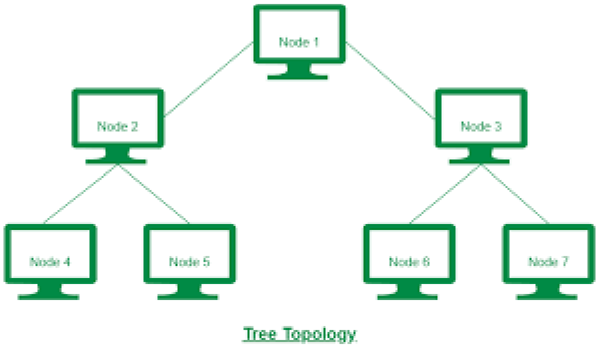
Complicated implementation.

It requires more space for dedicated links.

Because of the amount of cabling and the number of input-outputs, it is expensive to implement.

It requires a large space to run the cables.

Tree Topology



Tree Topology

Tree topologies have a root node, and all other nodes are connected which form a hierarchy. So it is also known

as hierarchical topology. This topology integrates various star topologies together in a single bus, so it is known

as a Star Bus topology. Tree topology is a very common network which is similar to a bus and star topology.

Advantages:

Here are pros/benefits of tree topology:

Failure of one node never affects the rest of the network.

Node expansion is fast and easy.

Detection of error is an easy process

It is easy to manage and maintain

Disadvantages:

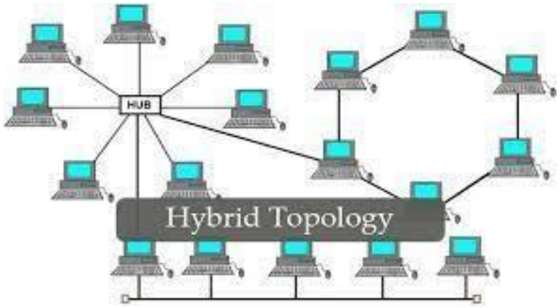
Here are cons/drawback of tree topology:

It is heavily cabled topology

If more nodes are added, then its maintenance is difficult

If the hub or concentrator fails, attached nodes are also disabled.

Hybrid Topology



Hybrid Topology

Hybrid topology combines two or more topologies. You can see in the above architecture in such a manner that the resulting network does not exhibit one of the standard topologies.

For example, as you can see in the above image that in an office in one department, Star and P2P topology is used. A hybrid topology is always produced when two different basic network topologies are connected.

Advantages:

Here, are advantages/pros using Hybrid topology:

Offers the easiest method for error detecting and troubleshooting

Highly effective and flexible networking topology

It is scalable so you can increase your network size

Disadvantages:

The design of hybrid topology is complex

It is one of the costliest processes

How to select a Network Topology?

Here are some important considerations for selecting the best topology to create a network in your organization:

Bus topology is surely least expensive to install a network.

If you want to use a shorter cable or you planning to expand the network is future, then star topology is the best choice for you.

Fully mesh topology is theoretically an ideal choice as every device is connected to every other device.

If you want to use twisted pair cable for networking, then you should build star topologies.

Conclusion: Demonstrated the different types of topologies and types of transmission media by using a packet tracer tool.

Lab Assignment No.	2
Title	Use packet Tracer tool for configuration of 3 router networks using one of the following protocols RIP/OSPF/BGP.
Roll No.	
Class	TE
Date of Completion	
Subject	Computer Network Laboratory
Assessment Marks	
Assessor's Sign	

ASSIGNMENT No: 02

Title: Use packet Tracer tool for configuration of 3 router networks.

Problem Statement: Use packet Tracer tool for configuration of 3 router networks using one of the following protocols RIP/OSPF/BGP.

Prerequisite:

a) IP Address

b) OSI & TCP/IP Model

c) Networking Devices

Objective:

- Understand how to Create LAN, PAN & CAN
- Understand how to Create wireless LAN using Access Point.

Outcomes:

- Students could create packet tracer LAN protocol and understand the concept of Switches, routers etc.

New Concepts:

1. Crimping
2. Access Point Configuration

Theory:

A **computer network** is a system in which multiple computers are connected to each other to share information and resources.

Characteristics of a Computer Network

- Share resources from one computer to another.
- Create files and store them in one computer, access those files from the other computer(s) connected over the network.
- Connect a printer, scanner, or a fax machine to one computer within the network and let other computers of the network use the machines available over the network.

Network Cables

Network cables are used to connect computers. The most commonly used cable is CAT cable & RJ-45.

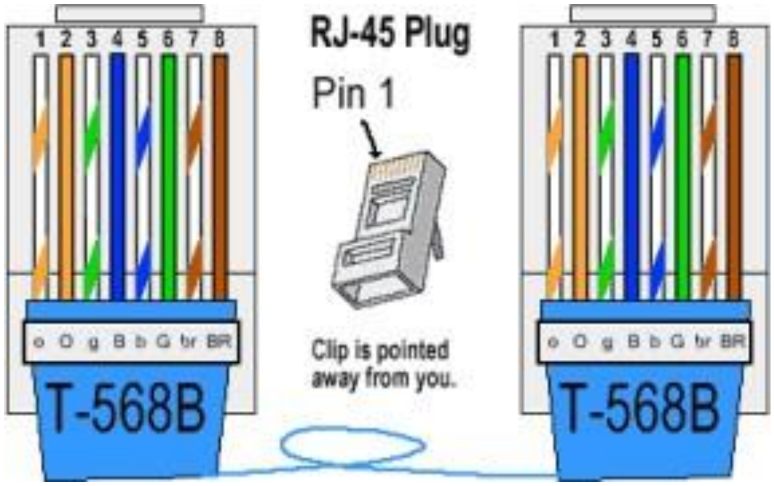
Network Card

Network card is a necessary component of a computer without which a computer cannot be connected over a network. It is also known as the network adapter or Network Interface Card (NIC). Most branded computers have network card pre-installed. Network cards are of two types: Internal and External Network Cards.

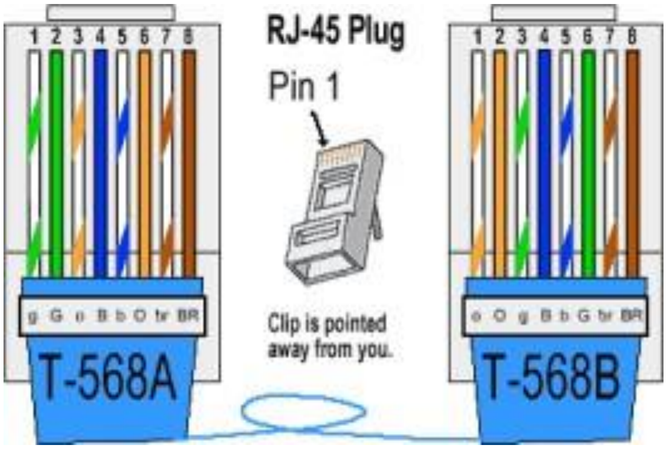
Straight Through & Crossover Cable:

The information listed here is to assist Network Administrators in the color coding of Ethernet cables. Please be aware that modifying Ethernet cables improperly may cause loss of network connectivity.

Straight-Through Ethernet Cable



Straight-Through Ethernet Cable



- **Local Area Network:**

A local area network (LAN) is a computer network that interconnects computers within a limited area such as a residence, school, laboratory, university campus or office building and has its network equipment and interconnects locally managed. By contrast, a wide area network (WAN) not only covers a larger geographic distance, but also generally involves leased telecommunication circuits or Internet links. An even greater contrast is the Internet, which is a system of globally connected business and personal computers. Ethernet and Wi-Fi are the two most common transmission technologies in use for local area networks.

- **Wireless LAN:**

A wireless local area network (WLAN) is a wireless computer network that links two or more devices using a wireless distribution method (often spread-spectrum orOFDM radio) within a limited area such as a home, school, computer laboratory, or office building. This gives users the ability to move around within a local coverage area and yet still be connected to the network. A WLAN can also provide a connection to the wider Internet.

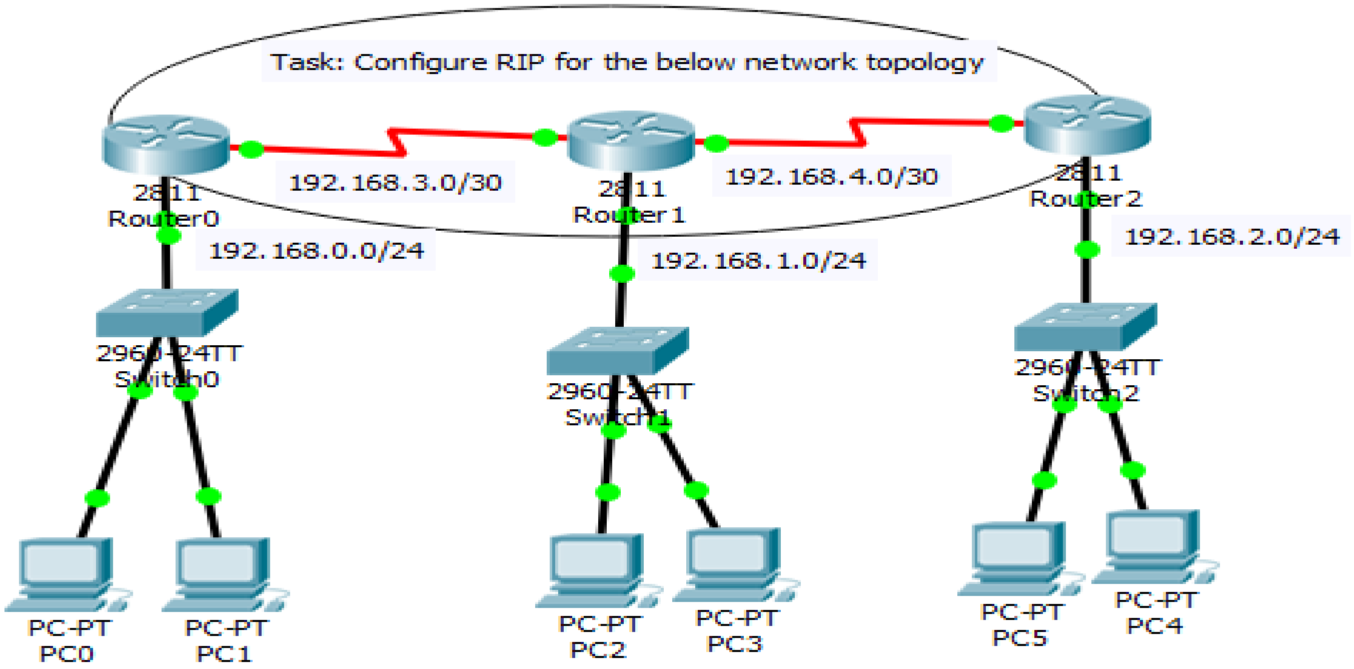
Most modern WLANs are based on IEEE 802.11 standards and are marketed under the Wi Fi brand name.

Benefits of Wireless LAN:

- People can access the network from where they want; they are no longer limited by the length of the cable
- Some places and vehicles have Wireless LANs. This means that people can access the internet even outside their normal work environment, for example when they ride a train.
- Setting up a wireless LAN can be done with one box called wireless access point or wireless router. This box can handle many connections at the same time. Wired networks require cables to be laid. This can be difficult for certain places.
- Access points can serve a varying number of computers.

Using Packet tracer tool :

Cisco Packet Tracer designed to be used as multi-tasking, that's been to examine varied network exercises like application of dissimilar topologies, development of apt servers, subnetting and study of different network setups, configuration and different troubleshooting defined commands. To initialise communication among two networking devices i.e., user networking devices and to organise a network, we intend to demand to pick applicable networking devices like routers hubs, switches or interconnecting devices and build physical change of integrity by connecting cables, quick local area network seaports from the module list of packet trace



Steps:

Step 1: Launch Packet Tracer.

Step 2: Build the topology.

Step 3: Configure the Wireless Router.

Step 4: Configure the Laptop. Or

Step 5: Configure the PC.

Step 6 :Send the packets from one PC to another

Conclusion: Use packet Tracer tool for configuration of 3 router networks using one of the following protocols RIP/OSPF/BGP.

Lab Assignment No.	3
Title	Write a program to demonstrate Sub-netting and find subnet masks.
Roll No.	
Class	TE
Date of Completion	
Subject	Computer Network Laboratory
Assessment Marks	
Assessor's Sign	

ASSIGNMENT No: 03

Title: Program to demonstrate Sub-netting and find subnet masks.

Problem Statement: Write a program to demonstrate Sub-netting and find subnet masks.

Prerequisite:

1. IP Address Classes

2. Classless & Classful IP Addressing

Objective:

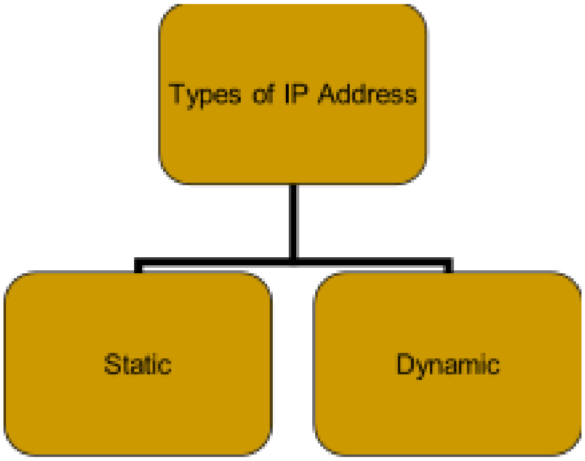
- Understand the concept Subnetting.
- Understand the Concept of Supernet.

Outcomes:

- Students should be able to classify the IP Address

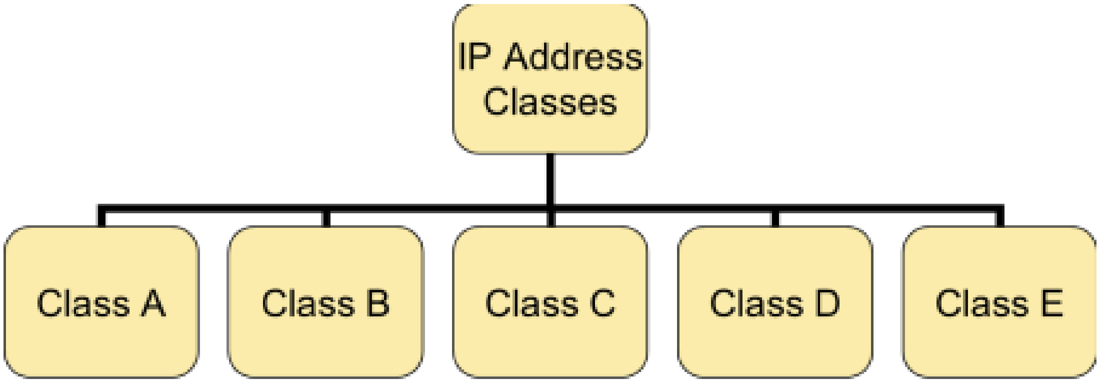
Theory:

- **A Unique, 32-bit address used by computers to communicate over a computer network**



Classes of Address

- IP address structure consists of two addresses, Network and Host
- IP address is divided into five classes



IP Address Classes

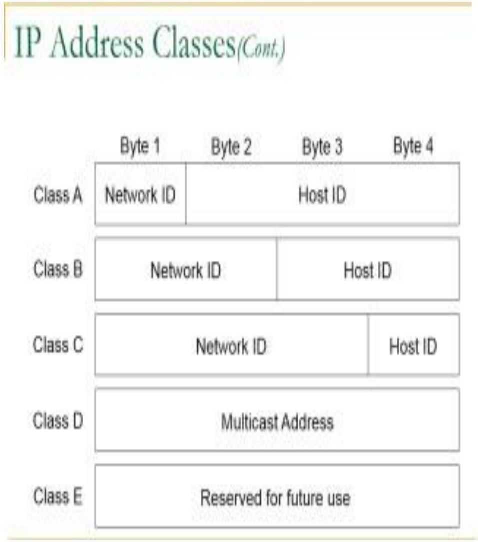
The 5 IP classes are split up based on the value in the 1st octet: Class A: 0-127

Class B: 128-191

Class C: 192-223

Class D: 224-239

Class E: 240-2552



IP Address Classes(Cont.)

	Byte 1	Byte 2	Byte 3	Byte 4
Class A	Network ID	Host ID		
Class B	Network ID		Host ID	
Class C	Network ID			Host ID
Class D	Multicast Address			
Class E	Reserved for future use			

Examples of IP Address

- 14.23.120.8 - The first byte of the address represents 14 which lies between 0 and 127, hence Class A address.
- 134.11.78.56 - The first byte of address is 134 which lies between 128 and 191 hence the address belongs to Class B.
- 193.14.56.22 - As first byte is 193 which is between 192 and 223, hence the address belongs to Class C.

Subnet Mask

An IP address has 2 parts:

- The Network identification.
- The Host identification.

Frequently, the Network & Host portions of the address need to be separately extracted.

- In most cases, if you know the address class, it's easy to separate the 2 portions.
- ■ Specifies part of IP address used to identify a subnetwork.
- ■ Subnet mask when logically ANDed with IP address provides 32-bit network address

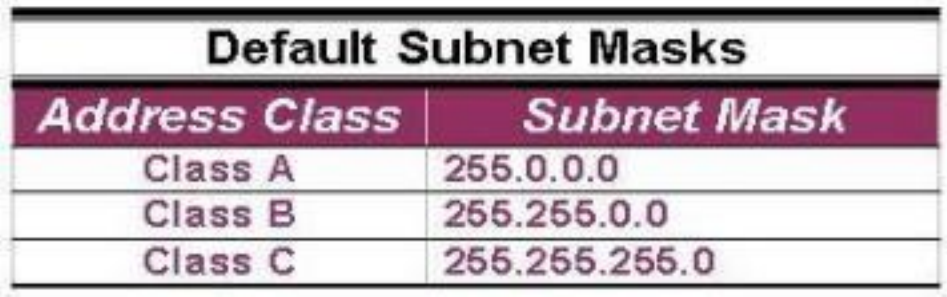
Default Mask:

- Has predetermined number of 1s
- Class A, B and C contains 1s in network ID fields for default subnet mask.

Address Class	**Default Mask** **(in Binary)**
Class A	**11111111.00000000.00000000.00000000**
Class B	**11111111.11111111.00000000.00000000**
Class C	**11111111.11111111.11111111.00000000**

Default Standard Subnet Masks

There are default standard subnet masks for Class A, B and C addresses:



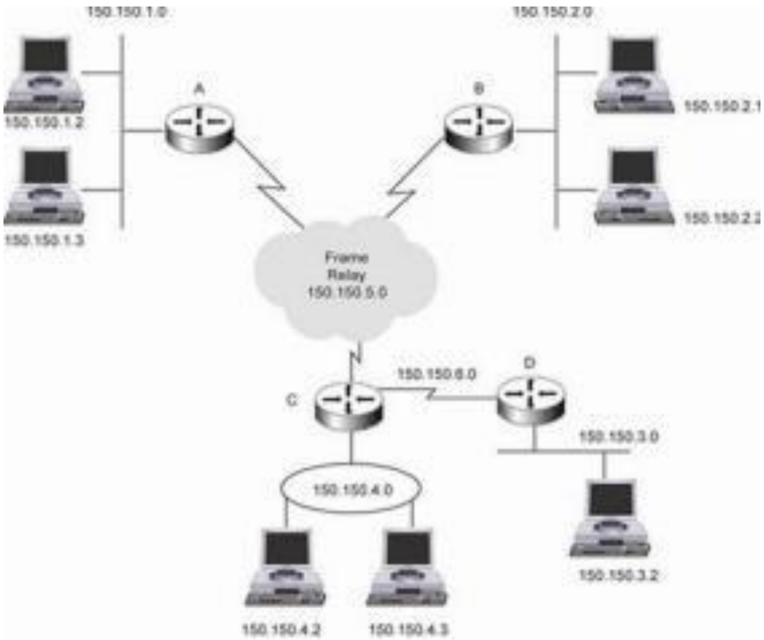
Default Subnet Masks	
Address Class	**Subnet Mask**
Class A	255.0.0.0
Class B	255.255.0.0
Class C	255.255.255.0

IP Subnetting:

■ Allows you to divide a network into smaller sub-networks

■ Each subnet has its own sub-network address

■ Subnet can be created within Class A, B, or C based networks



Subnetting:

■ Division of a network into subnets

- For example, division of a Class B address into several Class C addresses

- Some of the host IDs are used for creating subnet IDs

Need for Subnetting:

- ■ Classes A and B have a large number of hosts corresponding to each network ID

- It may be desirable to subdivide the hosts in Class C subnets
- ■ Often, there is a limitation on the number of hosts that could be hosted on a single network segment
- The limitation may be imposed by concerns related to the management of hardware
- ■ Smaller broadcast domains are more efficient and easy to manage

Subnetting Principle:

- ■ Use parts of the host IDs for subnetting purpose
- A subnet mask is used to facilitate the flow of traffic between the different subnets

  and the outside network (hops)
- A hop is the distance a data packet travels form one node to the other .

How to Calculate Subnets:

- ■ To determine the number of subnets & hosts per subnet available for any of the available subnet masks, 2 simple formulas to calculate these numbers:
- ■ Number of Subnets=(2^{n})
- ■ Number of Host per Subnets=(2^{h-2})
- ■ Although the 2 formulas look identical, the key is to remember the number you’re trying to calculate, hosts or subnets.
- ■ Eg., suppose you are asked to determine the number of subnets available & the number of hosts available on each subnet on the network 192.168.1.0
- ■ Using the subnet & hosts formulas, the answers are easily calculated. Of course, you must know your powers of 2 to calculate the answers

Example:

■ Host IP Address: 138.101.114.250

■ Network Mask: 255.255.0.0 (or /16)

■ Subnet Mask: 255.255.255.192 (or /26)

Given the following Host IP Address, Network Mask and Subnet mask find the Major Network Address, Network Broadcast Address, Range of Host if not subnetted, Subnet Address, range of host (First address and last address) ,Broadcast address,Total no of subnets and number of hosts per subnet.

Major Informations:

■ **Host IP Address**: 138.101.114.250

■ **Network Mask**: 255.255.0.0

■ **Subnet Mask**: 255.255.255.192

■ **Major Network Address**: 138.101.0.0

■ **Major Network Broadcast Address**: 138.101.255.255

■ **Range of Hosts if not Subnetted**: 138.101.0.1 to 138.101.255.254

Step 1: Convert to Binary

138. 101. 114. 250

IP Address 10001010 01100101 01110010 11111010 **Mask** 11111111 11111111 11111111 11000000
255. 255. 255. 192

Step 2: Find the Subnet Address

138. 101. 114. 250

IP Address 10001010 01100101 01110010 11111010 **Mask** 11111111 11111111 11111111 11000000 **Network** 10001010 01100101 01110010 11000000 **138 101 114 192**

Determine the Network (or Subnet) where this Host address lives: 1. Draw a line under the mask

2. Perform a bit-wise AND operation on the IP Address and the Subnet Mask Note: 1 AND 1

results in a 1, 0 AND anything results in a 0

3. Express the result in Dotted Decimal Notation

4. The result is the **Subnet Address** of this Subnet or "Wire" which is 138.101.114.192

138. 101. 114. 250

IP Address 10001010 01100101 01110010 11111010 **Mask** 11111111 11111111 11111111 11000000
Network 10001010 01100101 01110010 11000000 **138 101 114 192**

Quick method:

1. Find the last (right-most) 1 bit in the subnet mask.

2. Copy all of the bits in the IP address to the Network Address

3. Add 0's for the rest of the bits in the Network Address

Step 3: Subnet Range / Host Range

IP Address 10001010 01100101 01110010 11 111010 **Mask** 11111111 11111111 11111111 11 000000
Network 10001010 01100101 01110010 11 000000 **subnet host**

counting range counting

range

Determine which bits in the address contain Network (subnet) information and which contain Host information:

- ■ Use the **Network Mask**: 255.255.0.0 and divide (**Great Divide**) the from the rest of the address.
- ■ Use **Subnet Mask**: 255.255.255.192 and divide (**Small Divide**) the subnet from the hosts between the last "1" and the first "0" in the subnet mask.

Step 4: First Host / Last Host

IP Address 10001010 01100101 01110010 11 111010 **Mask** 11111111 11111111 11111111 11 000000
Network 10001010 01100101 01110010 11 000000 **subnet host**

counting range counting

range

First Host 10001010 01100101 01110010 11 000001 138 101 114 193

Last Host 10001010 01100101 01110010 11 111110 138 101 114 254

Broadcast 10001010 01100101 01110010 11 111111 138 101 114 255

Host Portion

- **Subnet Address:** all 0's
- **First Host**: all 0's and a 1
- **Last Host**: all 1's and a 0
- **Broadcast**: all 1's

Step 5: Total Number of Subnets

- Total number of subnets
    - Number of subnet bits 10
    - 2^{10} = 1,024
    - 1,024 total subnets
        - Subtract one "**if**" all-zeros subnet cannot be used
        - Subtract one "**if**" all-ones subnet cannot be used

Step 6: Total Number of Hosts per Subnet

IP Address 10001010 01100101 01110010 11 111010 **Mask** 11111111 11111111 11111111 11 000000
Network 10001010 01100101 01110010 11 000000 **subnet host**

counting range counting range

- Total number of hosts per subnet
    - Number of host bits 6
    - 2^6 = 64
    - 64 host per subnets
        - Subtract one for the subnet address

        - Subtract one for the broadcast address

    - 62 hosts per subnet

Conclusion:

Hence we have studied Subnetting and the importance of subnetting

Lab Assignment No.	4
Title	Write a program to implement link state /Distance vector routing protocol to find a suitable path for transmission.
Roll No.	
Class	TE
Date of Completion	
Subject	Computer Network Laboratory
Assessment Marks	
Assessor's Sign	

ASSIGNMENT No: 04

Title: Program to implement link state /Distance vector routing protocol to find a suitable path for transmission.

Problem Statement: Write a program to implement link state /Distance vector routing protocol to find a suitable path for transmission.

Prerequisite:

- Shortest path finding
- Classification of routing Algorithm

Objective:

- Understand the concept Distance vector routing
- Understand the Concept of Routing Algorithms

Outcomes:

- Students should be able to identify the route

Theory:

A distance-vector routing (DVR) protocol requires that a router inform its neighbors of topology changes periodically. Historically known as the old ARPANET routing algorithm (or known as Bellman-Ford algorithm).

Bellman Ford Basics – Each router maintains a Distance Vector table containing the distance between itself and ALL possible destination nodes. Distances,based on a chosen metric, are computed using information from the neighbors' distance vectors.

Information kept by DV router -

- Each router has an ID

Associated with each link connected to a router,

- there is a link cost (static or dynamic).
- Intermediate hops

Distance Vector Table Initialization -

- Distance to itself = 0
- Distance to ALL other routers = infinity number

Distance Vector Algorithm –

1. A router transmits its distance vector to each of its neighbors in a routing packet.
2. Each router receives and saves the most recently received distance vector from each of its neighbors.
3. A router recalculates its distance vector when:
   - It receives a distance vector from a neighbor containing different information than before.
   - It discovers that a link to a neighbor has gone down.

The DV calculation is based on minimizing the cost to each destination

Dx(y) = Estimate of least cost from x to y

C(x,v) = Node x knows cost to each neighbor v

Dx = [Dx(y): y ∈ N] = Node x maintains distance vector

Node x also maintains its neighbors' distance vectors

– For each neighbor v, x maintains Dv = [Dv(y): y ∈ N]

Introduction

Distance Vector Routing –

- It is a dynamic routing algorithm in which each router computes distance between itself and each possible destination i.e. its immediate neighbors.
- The router share its knowledge about the whole network to its neighbors and accordingly updates table based on its neighbors.
- The sharing of information with the neighbors takes place at regular intervals.
- It makes use of Bellman Ford Algorithm for making routing tables.
- Problems – Count to infinity problem which can be solved by splitting horizon.

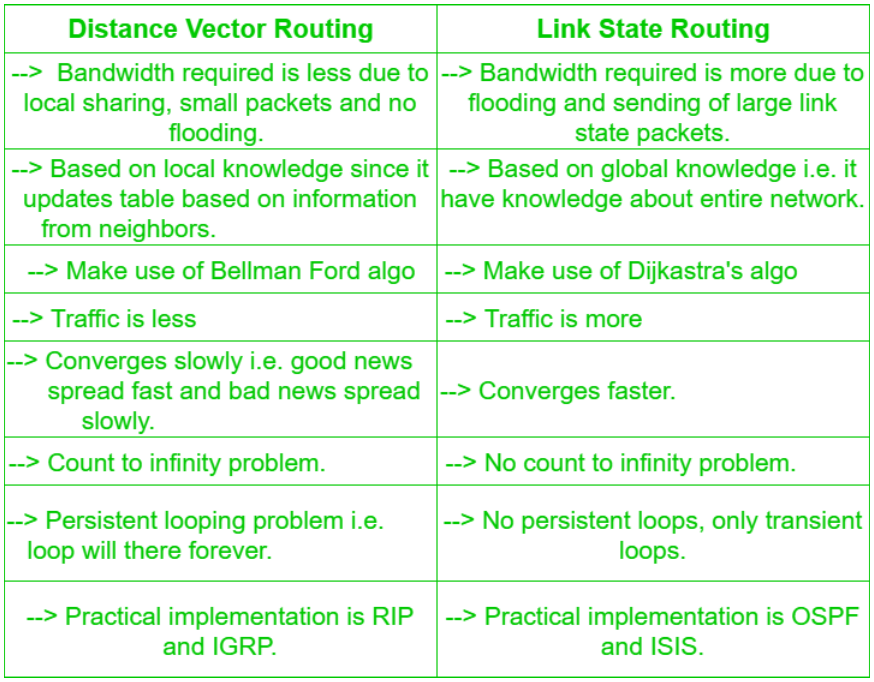
  – Good news spread fast and bad news spread slowly.

  – Persistent looping problem i.e. loop will be there forever.

Link State Routing –

- It is a dynamic routing algorithm in which each router shares knowledge of its neighbors with every other router in the network.

- A router sends its information about its neighbors only to all the routers through flooding.
- Information sharing takes place only whenever there is a change.
- It makes use of Dijkastra's Algorithm for making routing tables.
- Problems – Heavy traffic due to flooding of packets.
  – Flooding can result in infinite looping which can be solved by using Time to live (TTL) field.



Distance Vector Routing	Link State Routing
--> Bandwidth required is less due to local sharing, small packets and no flooding.	--> Bandwidth required is more due to flooding and sending of large link state packets.
--> Based on local knowledge since it updates table based on information from neighbors.	--> Based on global knowledge i.e. it have knowledge about entire network.
--> Make use of Bellman Ford algo	--> Make use of Dijkastra's algo
--> Traffic is less	--> Traffic is more
--> Converges slowly i.e. good news spread fast and bad news spread slowly.	--> Converges faster.
--> Count to infinity problem.	--> No count to infinity problem.
--> Persistent looping problem i.e. loop will there forever.	--> No persistent loops, only transient loops.
--> Practical implementation is RIP and IGRP.	--> Practical implementation is OSPF and ISIS.

Conclusion: Hence we have studied distance vector algorithm to find suitable path for transmission

Group B

Lab Assignment No.	5
Title	Socket Programming using C/C++/Java. a. TCP Client, TCP Server.
Roll No.	
Class	TE
Date of Completion	
Subject	Computer Network Laboratory
Assessment Marks	
Assessor's Sign	

ASSIGNMENT No: 05

Title: Implement Socket Programming using C/C++/Java.

Problem Statement: Implement Socket Programming using C/C++/Java. a. TCP Client, TCP Server.

Prerequisite:

a) Socket Header b) Network Programming c) Ports

Objective:

- To understand Work of Socket
- Different methods associated with Client & Server Socket

New Concepts:

- Client Server Communication
- Port Address

Outcomes:

- Students could be able to know different methods associated with client and server

Theory:

Socket Programming:

The Berkeley socket interface, an API, allows communications between hosts or between processes on one computer, using the concept of a socket. It can work with many different I/O devices and drivers, although support for these depends on the operating system implementation. This interface implementation is implicit for TCP/IP, and it is therefore one of the fundamental technologies underlying the Internet. It was first developed at the University of California, Berkeley for use on Unix systems. All modern operating systems now have some implementation of the Berkeley socket interface, as it has become the standard interface for connecting to the Internet. Programmers can make the socket interfaces accessible at three different levels, most powerfully and fundamentally at the RAW socket level. Very few applications need the degree of control over outgoing communications that this provides, so RAW sockets support was intended to be available only on computers used for developing Internet related technologies. TCP TCP provides the concept of a connection. A process creates a TCP socket by calling the socket() function with the parameters PF_INET or PF_INET6 and SOCK_STREAM. Server Setting up a simple TCP server involves the following steps: Creating a TCP socket, with a call to socket(). Computer Networks Lab 2 Binding the socket to the listen port, with a call to bind(). Before calling bind(), aprogrammer must declare a sockaddr_in structure, clear it (with bzero() ormemset()), and the sin_family (AF_INET or AF_INET6), and fill its sin_port (the listening port, in network byte order) fields. Converting a short int to networkbyte order can be done by calling the function htons() (host to network short). Preparing the socket to listen for connections (making it a listening socket), with acall to listen(). Accepting incoming connections, via a call to accept(). This blocks until anincoming connection is received, and then returns a socket descriptor for theaccepted connection. The initial descriptor remains a listening descriptor, and

accept() can be called again at any time with this socket, until it is closed. Communicating with the remote host, which can be done through send() andrecv(). Eventually closing each socket that was opened, once it is no longer needed, using close(). Note that if there were any calls to fork(), each process must close the sockets it knew about (the kernel keeps track of how many processes have a descriptor open), and two processes should not use the same socket

at once.

Client: Setting up a TCP client involves the following steps:

1. Creating a TCP socket, with a call to socket().

2. Connecting to the server with the use of connect, passing a sockaddr_in structurewith the sin_family set to AF_INET or AF_INET6, sin_port set to the port theendpoint is listening (in network byte order), and

sin_addr set to the IPv4 or IPv6address of the listening server (also in network byte order.) 1. Communicating with the server by send()ing and recv()ing.Terminating the connection and cleaning up with a call to close(). Again, if therewere any calls to fork(), each process must close() the socket. Functions: 1. socket(): socket() creates an endpoint for communication and returns a descriptor. socket() takes three arguments: domain, which specifies the protocol family of the created socket. For example: PF_INET for network protocol IPv4 or PF_INET6 for IPv6). type, one of: Computer Networks Lab 3 SOCK_STREAM (reliable stream-oriented service) SOCK_DGRAM (datagram service) SOCK_SEQPACKET (reliable sequenced packet service), or SOCK_RAW (raw protocols atop the network layer).protocol usually set to 0 to represent the default transport protocol for the specified domain and type values (TCP for PF_INET or PF_INET6 andSOCK_STREAM, UDP for those PF_ values and SOCK_DGRAM), but whichcan also explicitly specify a protocol. The function returns -1 if an error occurred. Otherwise, it returns an integer representing the newly-assigned descriptor. Prototype: int socket(int domain, int type, int protocol); connect(): connect() returns an integer representing the error code: 0 represents success, while -1 represents an error. Certain types of sockets are connectionless, most commonly user datagram protocol sockets. For these sockets, connect takes on a special meaning: the default target for sending and receiving data gets set to the given address, allowing the use of functions such as send() and recv() on connectionless sockets. Prototype: int connect(intsockfd, conststructsockaddr *serv_addr, socklen_taddrlen); bind(): bind() assigns a socket an address. When a socket is created using socket(), it is given an address family, but not assigned an address. Before a socket may accept incoming connections, it must be bound. bind() takes three arguments: sockfd, a descriptor representing the socket to perform the bind onmy_addr, a pointer to a sockaddr structure representing the address to bind to. addrlen, a socklen_t field representing the length of the sockaddr structure. It returns 0 on success and -1 if an error occurs. Prototype: int bind(intsockfd, structsockaddr *my_addr, socklen_taddrlen); listen() listen() prepares a bound socket to accept incoming connections. This function is only applicable to the SOCK_STREAM and SOCK_SEQPACKET socket types. It takes two arguments: sockfd, a valid socket descriptor. Computer Networks Lab 4 backlog, an integer representing the number of pending connections that can be queued up at any one time.

The operating system usually places a cap on this value. Once a connection is accepted, it is dequeued. On success, 0 is returned. If an error occurs, -1 is returned. Prototype: int listen(intsockfd, int backlog); accept() Programmers use accept() to accept a connection request from a remote host. It takes the following arguments: sockfd, the descriptor of the listening socket to accept the connection from. cliaddr, a pointer to the sockaddr structure that accept() should put the client'saddress information into. addrlen, a pointer to the socklen_t integer that will indicate to accept() how largethe sockaddr structure pointed to by cliaddr is. When accept() returns, the ocklen_t integer then indicates how many bytes of the cliaddr structure wereactually used. The function returns a socket corresponding to the accepted connection, or -1 if an error occurs. Prototype: int accept(intsockfd, structsockaddr *cliaddr, socklen_t *addrlen); Blocking vs. nonblocking Berkeley sockets can operate in one of two modes: blocking or non-blocking. A blocking socket will not "return" until it has sent (or received) all the data specified for the operation. This may cause problems if a socket continues to listen: a program may hang as the socket waits for data that may never arrive. A socket is typically set to blocking or nonblocking mode using the fcntl() or ioctl() functions. Cleaning up The system will not release the resources allocated by the socket() call until a close() call occurs. This is especially important if the connect() call fails and may be retried. Each call to socket() must have a matching call to close() in all possible execution paths.

Algorithm:

Server Program

1. Open the Server Socket: ServerSocket server = new ServerSocket(PORT);

2. Wait for the Client Request: Socket client = server.accept();

3. Create I/O streams for communicating to the client DataInputStream is = new DataInputStream(client.getInputStream()); DataOutputStreamos = new DataOutputStream(client.getOutputStream());

4. Perform communication with client Receive from client: String line = is.readLine(); Send to client: os.writeBytes(“Hello\n”)

5. Close socket: client.close();

Client Program

1) Create a Socket Object: Socket client = new Socket(server, port_id);

2) Create I/O streams for communicating with the server. is = new DataInputStream(client.getInputStream()); os = new DataOutputStream(client.getOutputStream());

3) Perform I/O or communication with the server: Receive data from the server: String line = is.readLine(); Send data to the server: os.writeBytes(“Hello\n”);

4) Close the socket when done:

5) client.close();

TYPES OF SOCKETS

Socket Types

There are four types of sockets available to the users. The first two are most commonly used and the last two are rarely used.

Processes are presumed to communicate only between sockets of the same type but there is no restriction that prevents communication between sockets of different types.

- Stream Sockets − Delivery in a networked environment is guaranteed. If you send through the stream socket three items "A, B, C", they will arrive in the same order − "A, B, C". These sockets use TCP (Transmission Control Protocol) for data transmission. If delivery is impossible, the sender receives an error indicator. Data records do not have any boundaries.
- Datagram Sockets − Delivery in a networked environment is not guaranteed. They're connectionless because you don't need to have an open connection as in Stream Sockets − you build a packet with the destination information and send it out. They use UDP (User Datagram Protocol).
- Raw Sockets − These provide users access to the underlying communication protocols, which support socket abstractions. These sockets are normally datagram oriented, though their exact characteristics are dependent on the interface provided by the protocol. Raw sockets are not intended for the general user; they have been provided mainly for those interested in developing new communication protocols, or for gaining access to some of the more cryptic facilities of an existing protocol.

Here is the description of the parameters −

- **socket_family −** This is either AF_UNIX or AF_INET, as explained earlier.
- **socket_type −** This is either SOCK_STREAM or SOCK_DGRAM.
- **protocol −** This is usually left out, defaulting to 0.

Once you have socket object, then you can use required functions to create your client or server program. Following is the list of functions required –

SERVER SOCKET METHODS

Sr.No.	Method & Description
1	**s.bind()**This method binds address (hostname, port number pair) to socket.
2	**s.listen()**This method sets up and start TCP listener.
3	**s.accept()**This passively accept TCP client connection, waiting until connection arrives (blocking).

CLIENT SOCKET METHODS

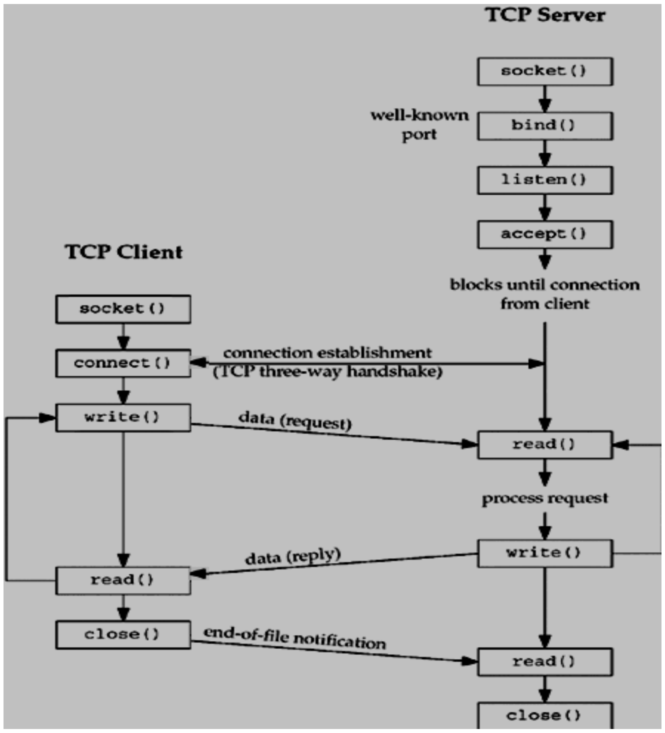
Sr.No.	Method & Description
1	**s.connect()**This method actively initiates TCP server connection.

GENERAL SOCKET METHODS

Sr.No.	Method & Description
1	**s.recv()**This method receives TCP message
2	**s.send()**This method transmits TCP message
3	**s.recvfrom()**This method receives UDP message
4	**s.sendto()**This method transmits UDP message
5	**s.close()**This method closes socket

6	**socket.gethostname()**Returns the hostname.

Methods Associated with Socket:The following diagram shows the complete Client and Server interaction −



Conclusion: Thus we have successfully implemented the socket programming for TCP

Lab Assignment No.	6
Title	Write a program using TCP socket for wired network for following a. Say Hello to Each other. b. File transfer.
Roll No.	
Class	TE
Date of Completion	
Subject	Computer Network Laboratory
Assessment Marks	
Assessor's Sign	

ASSIGNMENT No: 06

Title: Write a program using TCP socket for wired network for following
a. Say Hello to Each other.
b. File transfer.

Problem Statement :- Write a program using TCP socket for wired network forfollowing

a) Say hello to each other

b) File transfer

c) Calculator

Prerequisite:

a) Socket Header b) Network Programming c) Ports

Objective:

- To understand Work of Socket
- Different methods associated with Client & Server Socket

New Concepts:

- Client Server Communication
- Port Address

Outcomes:

- Students could be able to know different methods associated with client and server

□ **Theory**:

Socket:

Sockets allow communication between two different processes on the same or different machines. To be more precise, it's a way to talk to other computers using standardUnix file descriptors. In Unix, every I/O action is done by writing or reading a file descriptor. A file descriptor is just an integer associated with an open file and it can be a network connection, a text file, a terminal, or something else.

Socket Types :-

There are four types sockets available to the users. The first two are mostcommonly used and the last two are rarely used.

Processes are presumed to communicate only between sockets of the same type but there is no restriction that prevents communication between sockets of different types.

1) Stream Sockets – Delivery in a networked environment is guaranteed. If you send through the stream socket three items “A, B, C”, they will arrive in the same order – “A, B, C”. These sockets use TCP (Transmission Control Protocol) for data transmission. If delivery is impossible, the sender receives an error indicator. Data records do not have any boundaries.

2) Datagram Sockets – Delivery in a networked environment is not guaranteed. They’reconnectionless because you don’t need to have an open connection as in Stream Sockets – you build a packet with the destination information and send it out. They

use UDP (User Datagram Protocol).

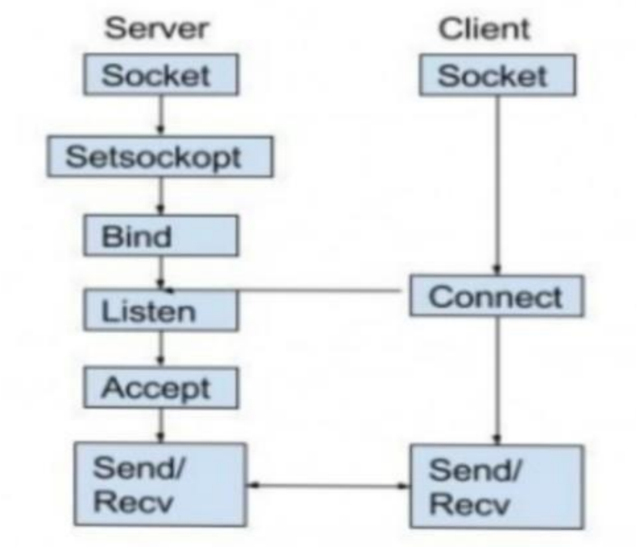
3) Raw Sockets – These provide users access to the underlying communication protocols, which support socket abstractions. These sockets are normally datagram oriented, though their exact characteristics are dependent on the interface provided bythe protocol. Raw sockets are not intended for the general user; they have been provided mainly for those interested in developing new communication protocols, or for gaining access to some of the more cryptic facilities of an existing protocol.

4) Sequenced Packet Sockets – They are similar to a stream socket, with the exception that record boundaries are preserved. This interface is provided only as a part of the Network Systems (NS) socket abstraction, and is very important in most serious NS applications. Sequenced-packet sockets allow the user to manipulate the Sequence Packet Protocol (SPP) or Internet Datagram Protocol (IDP) headers on a packet or a group of packets, either by writing a prototype header along with whatever data is to be sent, or by specifying a default header to be used with all outgoing data, and allowsthe user to receive the headers on incoming packets.

- What is socket programming?

Socket programming is a way of connecting two nodes on a network to communicate with each other. One socket(node) listens on a particular port at an IP, whileother socket reaches out to the other to form a connection. Server forms the listener socketwhile client reaches out to the server.

State diagram for server and client detail



Header Files :

Netinet/in.h Defines Internet constants and structures. Arpa/nameser.h Contains Internet

name server information.Netdb.h Contains data definitions for socket subroutines.

Resolv.h Contains resolver global definitions and variables. Sys/socket.h Contains data

definitions and socket structures.

Sys/socketvar.h Defines the kernel structure per socket and contains buffer queues.

Sys/types.h Contains data type definitions.

Sys/un.h Defines structures for the UNIX interprocess communication domain.

Sys/ndd_var.h Defines structures for the AIX Network Device Driver (NDD) domain.

Sys/atmsock.h Contains constants and structures for the Asynchronous Transfer Mode(ATM) protocol in the AIX NDD domain.

Stages for server

1. Socket creation:

Int sockfd = socket(domain, type, protocol)

Sockfd: socket descriptor, an integer (like a file-handle)

Domain: integer, communication domain e.g., AF_INET (IPv4 protocol) ,AF_INET6 (IPv6 protocol)

Type: communication type

SOCK_STREAM: TCP(reliable, connection oriented) SOCK_DGRAM: UDP(unreliable,

connectionless)

Protocol: Protocol value for Internet Protocol(IP), which is 0. This is the same numberwhich appears on protocol field in the IP header of a packet.(man protocols for more details) .

2. Bind:

Int bind(int sockfd, const struct sockaddr *addr,

Socklen_t addrlen);

After creation of the socket, bind function binds the socket to the address and port number specified in addr(custom data structure). In the example code, we bind the server to the localhost, hence we use INADDR_ANY to specify the IP address.

3. Listen:

Int listen(int sockfd, int backlog);

It puts the server socket in a passive mode, where it waits for the client to approachthe server to make a connection. The backlog, defines the maximum length to which the queue of pending connections for sockfd may grow. If a connection request arrives when the queue is full, the client may receive an error with an indication of ECONNREFUSED.

4. Accept:

Int new_socket= accept(int sockfd, struct sockaddr *addr, socklen_t *addrlen);

It extracts the first connection request on the queue of pending connections for thelistening socket, sockfd, creates a new connected socket, and returns a new file descriptorreferring to that socket. At this point, connection is established between client and server,and they are ready to transfer data.

Stages for Client

1. Socket connection: Exactly same as that of server's socket creation

2. Connect:

Int connect(int sockfd, const struct sockaddr *addr, socklen_t addrlen);

The connect() system call connects the socket referred to by the file descriptor sockfd to the address specified by addr. Server's address and port is specified in addr.

Memset(void *str, int c, size_t n)Parameters

- str – This is a pointer to the block of memory to fill.
- c – This is the value to be set. The value is passed as an int, but the function fillsthe block of memory using the unsigned char conversion of this value.
- n – This is the number of bytes to be set to the value.

Fopen:

The fopen() function shall open the file whose pathname is the string pointedto by filename, and associates a stream with it.

The mode argument points to a string. If the string is one of the following, thefile shall be opened in the indicated mode. Otherwise, the behavior is undefined.

1. R or rb : Open file for reading.
2. W or wb : Truncate to zero length or create file for writing.
3. A or ab : Append; open or create file for writing at end-of-file.
4. R+ or rb+ or r+b : Open file for update (reading and writing).
5. W+ or wb+ or w+b: Truncate to zero length or create file for update.

6. A+ or ab+ or a+b: Append; open or create file for update, writing at end-of-file.

The htonl() function converts the unsigned integer hostlong from host byteorder to network byte order.

The htons() function converts the unsigned short integer hostshort from hostbyte order to network byte order.

The ntohl() function converts the unsigned integer netlong from network byteorder to host byte order.

The ntohs() function converts the unsigned short integer netshort from networkbyte order to host byte order.

Conclusion: - Successfully implemented the TCP socket programming.

Lab Assignment No.	7
Title	Write a program using UDP Sockets to enable file transfer (Script, Text, Audio and Video one file each) between two machines.
Roll No.	
Class	TE
Date of Completion	
Subject	Computer Network Laboratory
Assessment Marks	
Assessor's Sign	

ASSIGNMENT No: 07

Title: Program using UDP Sockets to enable file transfer

Problem Statement: Write a program using UDP Sockets to enable file transfer (Script, Text, Audio and Video one file each) between two machines.

Prerequisite:

a) Socket Header b) Network Programming c) Ports

Objective:

- To understand Work of Socket
- Different methods associated with Client & Server Socket

New Concepts:

- Client Server Communication
- Port Address

Outcomes:

- Students could be able to know different methods associated with client and server

Theory:

User Datagram Protocol:-

UDP is a connectionless and unreliable transport protocol. The two ports serve to identify the end points within the source and destination machines. User Datagram Protocol is used, in place of TCP, when a reliable delivery is not required.However, UDP is never used to send important data such as web-pages, database information, etc. Streaming media such as video,audio and others use UDP because it offers speed.

Why UDP is faster than TCP?

The reason UDP is faster than TCP is because there is no form of flow control. No error checking,error correction, or acknowledgment is done by UDP.UDP is only concerned with speed. So when, the data sent over the Internet is affected by collisions, and errors will be present. UDP packet's called as user datagrams with 8 bytes header. A format of user datagrams is shown in figur 3. In the user datagrams first 8 bytes contains header information and the remaining bytes contains data.

LINUX SOCKET PROGRAMMING:

The Berkeley socket interface, an API, allows communications between hosts or between processes on one computer, using the concept of a socket. It can work with many different I/O devices and drivers, although support for these depends on the operating-system implementation. This interface implementation is implicit for TCP/IP, and it is therefore one of the fundamental technologies underlying the Internet. It was first developed at the University of California, Berkeley for use on Unix systems. All modern operating systems now have some implementation of the Berkeley socket interface, as it has become the standard interface for connecting to the Internet. Programmers can make the socket interfaces accessible at three different levels, most powerfully and fundamentally at the RAW socket level. Very few applications need the degree of control over outgoing communications that this provides, so RAW sockets support was intended to be available only on computers used for developing Internet-related technologies. In recent years, most operating systems have implemented support for it anyway, including Windows XP. The header files: The Berkeley socket development library has many associated header files. They include: Definitions for the most basic of socket structures with the BSD socket API Basic data types associated with structures within the BSD socket API Definitions for the socketaddr_in{} and other base data structures.

The header files:

The Berkeley socket development library has many associated header files.

They include: ***<sys/socket.h>***
Definitions for the most basic of socket structures with the BSD

socket API <*sys/socket.h*>

Basic data types associated with structures within the BSD socket API <sys/types.h>

Socket API<sys/types.h>

Definitions for the socketaddr_in{} and other base data structures

<sys/un.h>

Definitions and data type declarations for SOCK_UNIX streams

UDP: UDP consists of a connectionless protocol with no guarantee of delivery. UDP packets may arrive out of order, become duplicated and arrive more than once, or even not arrive at all. Due to the minimal guarantees involved, UDP has considerably less overhead than TCP. Being connectionless means that there is no concept of a stream or connection between two hosts, instead, data arrives in datagrams. UDP address space, the space of UDP port numbers (in ISO terminology, the TSAPs), is completely disjoint from that of TCP ports. Server: Code may set up a UDP server on port 7654 as follows:

```
sock = socket(PF_INET,SOCK_DGRAM,0);

sa.sin_addr.s_addr = INADDR_ANY;

sa.sin_port = htons(7654);

bound = bind(sock,(struct sockaddr *)&sa, sizeof(struct sockaddr));

 if (bound < 0) fprintf(stderr, "bind(): %s\n",strerror(errno)); listen(sock,3);
```

bind() binds the socket to an address/port pair. listen() sets the length of the new connections queue.

```
 while (1)
 {
 printf ("recv test. . \n");
recsize = recvfrom(sock, (void *)hz, 100, 0, (struct sockaddr *)&sa, fromlen);
printf ("recsize: %d\n ",recsize);
 if (recsize < 0)
fprintf(stderr, "%s\n", strerror(errno));
sleep(1);
printf("datagram: %s\n",hz);
 }
```

This infinite loop receives any UDP datagrams to port 7654 using recvfrom(). It uses the parameters: l socket l pointer to buffer for data l size of buffer l flags (same as in read or other receive socket function)

Client: A simple demo to send an UDP packet containing "Hello World!" to address 127.0.0.1, port 7654 might look like this:

```
#include #include #include #include #include

#include int main(int argc, char *argv[])
{
int sock; struct sockaddr_in sa;
int bytes_sent, buffer_length;
char buffer[200];
sprintf(buffer, "Hello World!");
buffer_length = strlen(buffer) + 1;
 sock = socket(PF_INET, SOCK_DGRAM, 0);
 sa.sin_family = AF_INET;
 sa.sin_addr.s_addr = htonl(0x7F000001);
sa.sin_port = htons(7654); bytes_sent = sendto(sock, buffer, buffer_length, 0, &sa, sizeof(struct
sockaddr_in) );
 if(bytes_sent < 0) printf("Error sending packet: %s\n", strerror(errno) );
 return 0;
 }
```

In this code, buffer provides a pointer to the data to send, and buffer_length specifies the size of the buffer contents. Typical UDP client code

- Create UDP socket to contact server (with a given hostname and service port number)
- Create UDP packet.
- Call send(packet), sending request to the server.
- Possibly call receive(packet) (if we need a reply).

Typical UDP Server code

- Create UDP socket listening to a well known port number.
- Create UDP packet buffer Call receive(packet) to get a request, noting the address of the client.
- Process request and send reply back with send(packet).

APPLICATION :

Socket programming is essential in developing any application over a network.

Conclusion: Thus we have studied Working of UDP Socket

Lab Assignment No.	8
Title	Study and Analyze the performance of HTTP, HTTPS and FTP protocol using Packet tracer tool.
Roll No.	
Class	TE
Date of Completion	
Subject	Computer Network Laboratory
Assessment Marks	
Assessor's Sign	

ASSIGNMENT No: 08

Title: Study the performance of HTTP, HTTPS and FTP protocol using Packet tracer tool.

Problem Statement: Study and Analyze the performance of HTTP, HTTPS and FTP protocol using Packet tracer tool.

Prerequisite:

a) HTTP Protocols b) Network Programming c) Ports

Objective:

- To understand different types of protocols
- Analyzing performance of protocols

Outcomes:

- Students could be able to analyze the performance of HTTP,HTTPS and FTP protocol

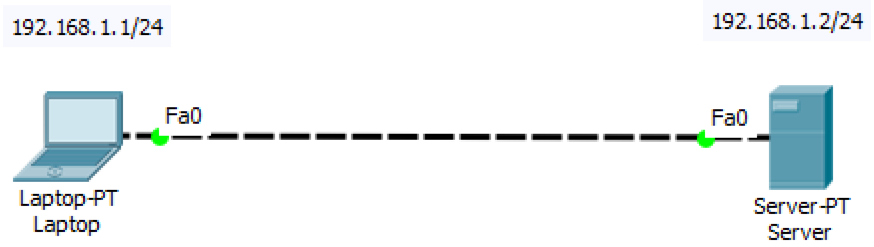
Theory: The File Transfer Protocol (FTP) is a standard network protocol used for the transfer of computer files between a client and server on a computer network.

FTP employs a client-server architecture whereby the client machine has an FTP client installed and establishes a connection to an FTP server running on a remote machine. After the connection has been established and the user is successfully authenticated, the data transfer phase can begin.

Worth noting: Although FTP does support user authentication, all data is sent in clear text, including usernames and passwords. For secure transmission that protects the username and password, and encrypts the content, FTP is often secured with SSL/TLS (FTPS) or replaced with SSH File Transfer Protocol (SFTP).

Let’s now do FTP configuration in Packet Tracer:

1.Build the network topology.



FTP topology.PNG

2. Configure static IP addresses on the Laptop and the server.

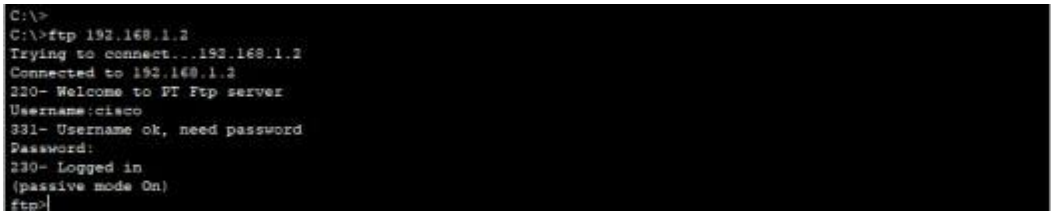
Laptop: IP address: 192.168.1.1 Subnet Mask: 255.255.255.0

Server: IP address: 192.168.1.2 Subnet Mask: 255.255.255.0

3. Now try using an FTP client built in the Laptop to send files to an FTP server configured in the Server.

From the Laptop's command prompt, FTP the server using the server IP address by typing: ftp 192.168.1.2

Provide the username(cisco) and password(cisco) [which are the defaults] for ftp login.



```
C:\>
C:\>ftp 192.168.1.2
Trying to connect...192.168.1.2
Connected to 192.168.1.2
220- Welcome to PT Ftp server
Username:cisco
331- Username ok, need password
Password:
230- Logged in
(passive mode On)
ftp>
```

ftp from laptop.PNG

You are now in the FTP prompt .

PC0 has an FTP client which can be used to read, write, delete and rename files present in the FTP server.

The FTP server can be used to read and write configuration files as well as IOS images. Additionally, the FTP server also supports file operations such rename, delete and listing directory.

With that in mind, we can do something extra. So let’s do this:

4. Create a file in the Laptop then upload it to the server using FTP.

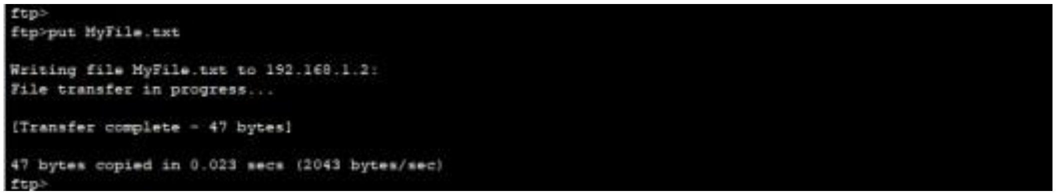
To do this, open the Text Editor in the Laptop, create a file and give it your name of choice.

Type any text in the editor then save your file. e.g. myFile.txt.

5. Now upload the file from the Laptop to the server using FTP. (An FTP connection has to be started first. But this is what we’ve done in step 3)

So to do an FTP upload, we’ll type:

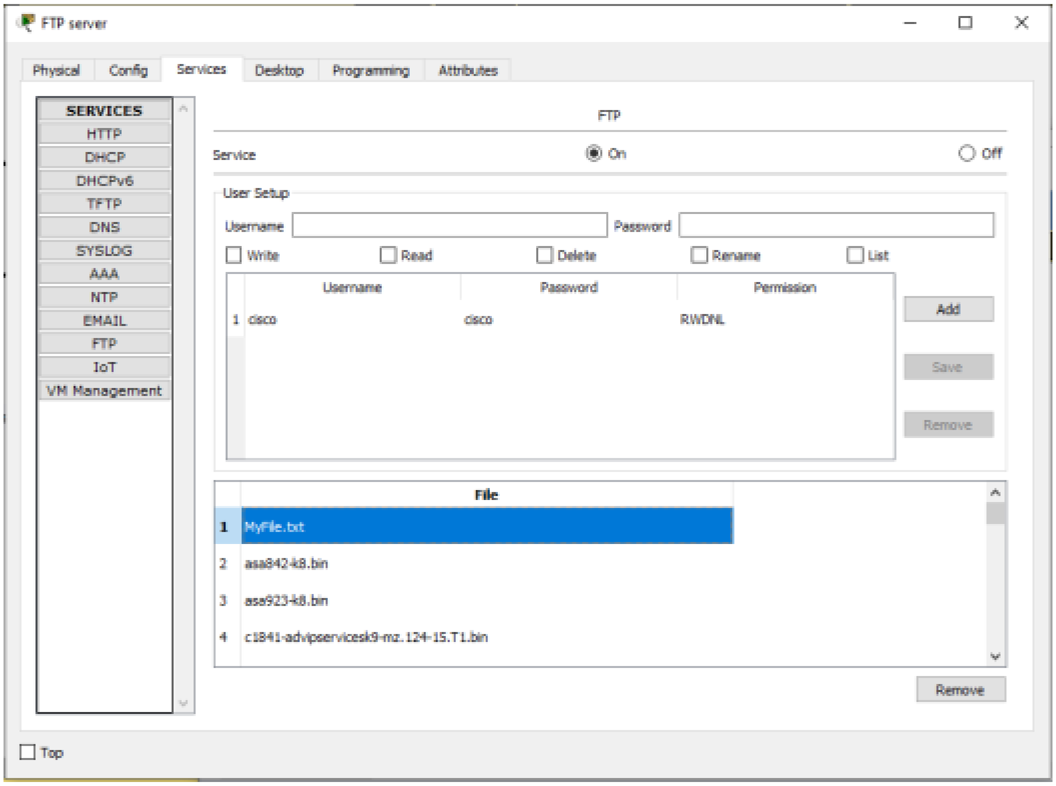
put MyFile.txt



```
ftp>
ftp>put MyFile.txt

Writing file MyFile.txt to 192.168.1.2:
File transfer in progress...

[Transfer complete - 47 bytes]

47 bytes copied in 0.023 secs (2043 bytes/sec)
ftp>
```

put MyFile to FTP directory.PNG

6. Once file upload is successful, go to the Server FTP directory to verify if the file sent has been received . To do this, go to Server-> Services->FTP. Here look for MyFile.txt sent from the laptop.



FTP

Service ● On ○ Off

User Setup

Username | Password

□ Write □ Read □ Delete □ Rename □ List

	Username	Password	Permission
1	cisco	cisco	RWDNL

	File
1	MyFile.txt
2	asa842-k8.bin
3	asa923-k8.bin
4	c1841-advipservicesk9-mz.124-15.T1.bin

MyFile.txt really send to sever.PNG

Something extra: To check other FTP commands supported by the FTP client running on the Laptop(or PC), you can use a question mark (?) on the Laptop’s command prompt as shown below:

All FTP commands supported

You can see the put command that we used to upload our file to the FTP server. Other commands listed include:

get-used to get(download) a file from the server.

For example: get MyFile.txt

delete– to delete a file in the FTP directory with the server

For example: delete MyFile.txt

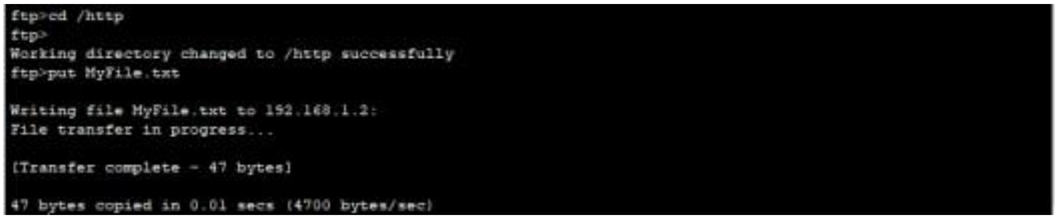
Rename– used to Rename a file

cd – used to change directory.

For example, we can open an HTTP directory in the server by typing: cd /http. This will change the current directory from FTP directory to HTTP directory

Once the http directory is open, you can upload a file to the HTTP server. You’re now uploading a file to an HTTP folder(directory) using FTP.

For example: put MyFile.txt

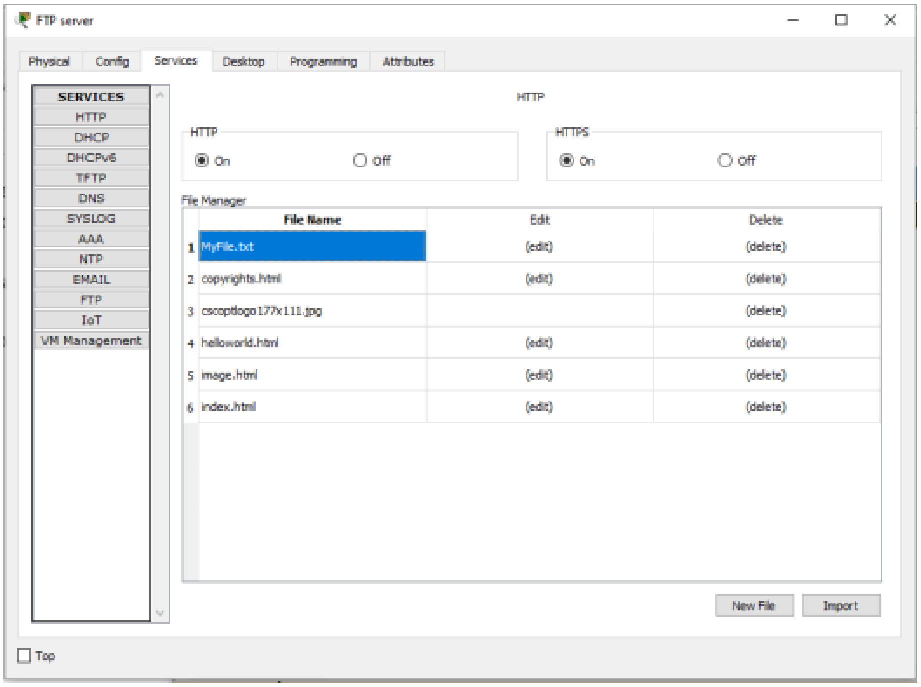


```
ftp>cd /http
ftp>
Working directory changed to /http successfully
ftp>put MyFile.txt

Writing file MyFile.txt to 192.168.1.2:
File transfer in progress...

[Transfer complete - 47 bytes]

47 bytes copied in 0.01 secs (4700 bytes/sec)
```

To see this working, let’s open an HTTP directory and upload(put) a file to it using FTP:

changing directory then put files to HTTP directory using FTP

You can now check up in the HTTP directory in the server and verify that the file uploaded from the Laptop(MyFile.txt) is well received:



HTTP

HTTP: ● On ○ Off

HTTPS: ● On ○ Off

File Manager

	File Name	Edit	Delete
1	MyFile.txt	(edit)	(delete)
2	copyrights.html	(edit)	(delete)
3	cscoptlogo177x111.jpg		(delete)
4	helloworld.html	(edit)	(delete)
5	image.html	(edit)	(delete)
6	index.html	(edit)	(delete)

□ Top

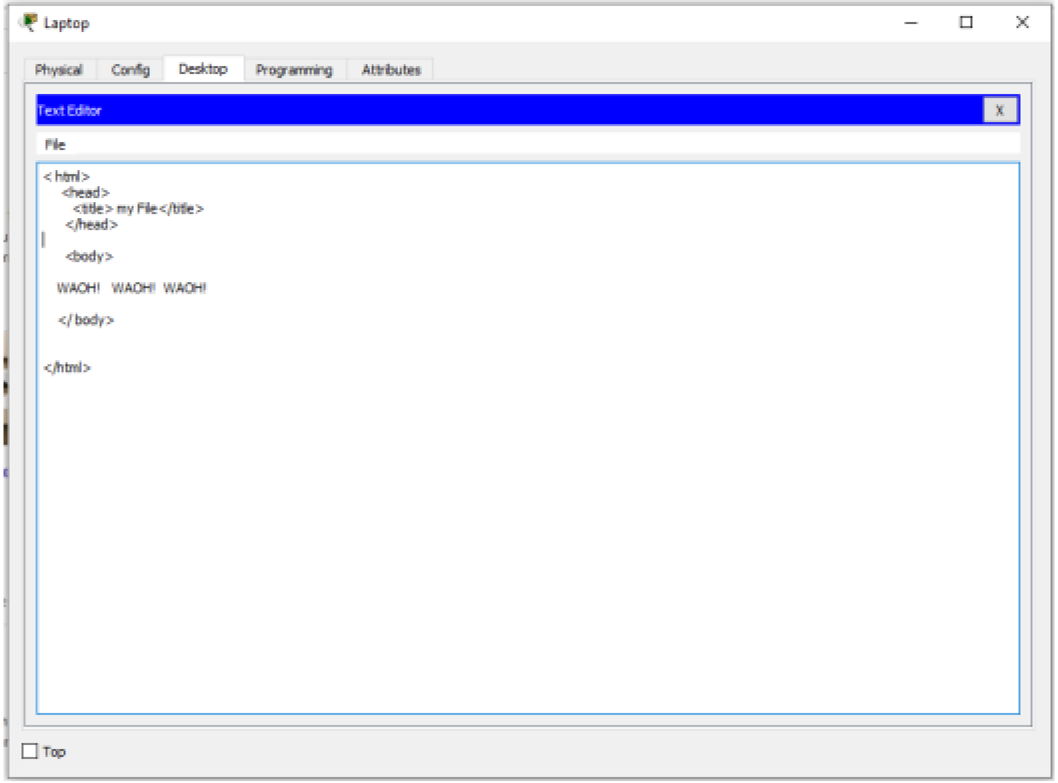
MyFile.txt really send to HTTP server

Notice that we are uploading files to an HTTP Server directory using File Transfer Protocol.(FTP). This is what actually happens when you use an FTP client such as FileZilla client to upload files to a website. In our case here, we are using an FTP client built-in the Laptop.

This may interest you: The first FTP client applications were command-line programs developed before operating systems had graphical user interfaces, and are still shipped with most Windows and Linux operating systems. (Actually this is what we have been using this far). Many FTP clients(e.g. FileZilla) and automation utilities have since been developed for desktops, servers, mobile devices, and hardware. FTP has also been incorporated into productivity applications, such as HTML editors.

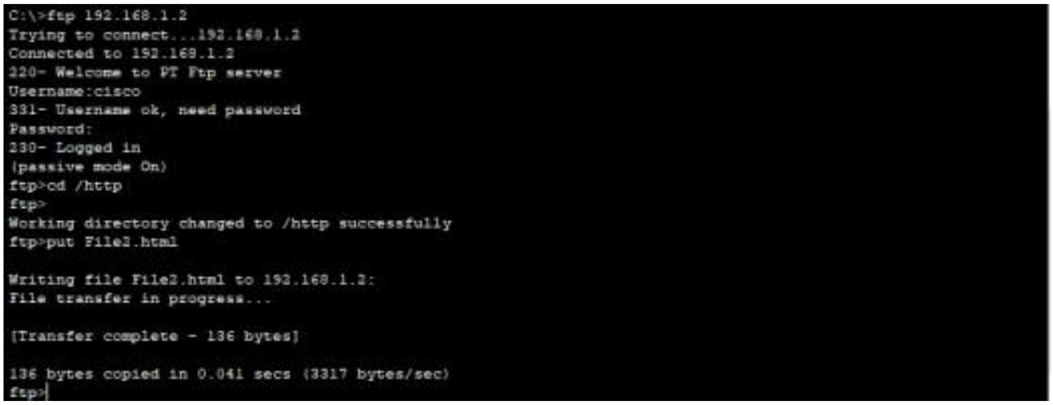
We'll create an html file in our Laptop, upload it to HTTP server directory using FTP, then try to access the file from the Laptop's browser.

On the Laptop, open the text editor, then type some markup(html) and save the file with the extension .html. See all this below:



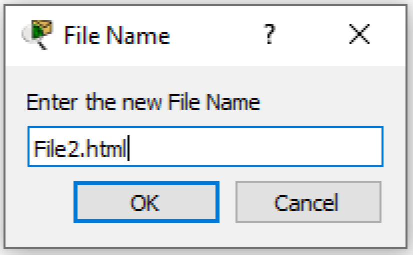
```
< html>
   <head>
     <title> my File</title>
    </head>

    <body>

  WAOH!   WAOH!  WAOH!

  </ body>


</html>
```

File2 HTML code



```
C:\>ftp 192.168.1.2
Trying to connect...192.168.1.2
Connected to 192.168.1.2
220- Welcome to PT Ftp server
Username:cisco
331- Username ok, need password
Password:
230- Logged in
(passive mode On)
ftp>cd /http
ftp>
Working directory changed to /http successfully
ftp>put File2.html

Writing file File2.html to 192.168.1.2:
File transfer in progress...

[Transfer complete - 136 bytes]

136 bytes copied in 0.041 secs (3317 bytes/sec)
ftp>
```

Save your file as an html file like this:

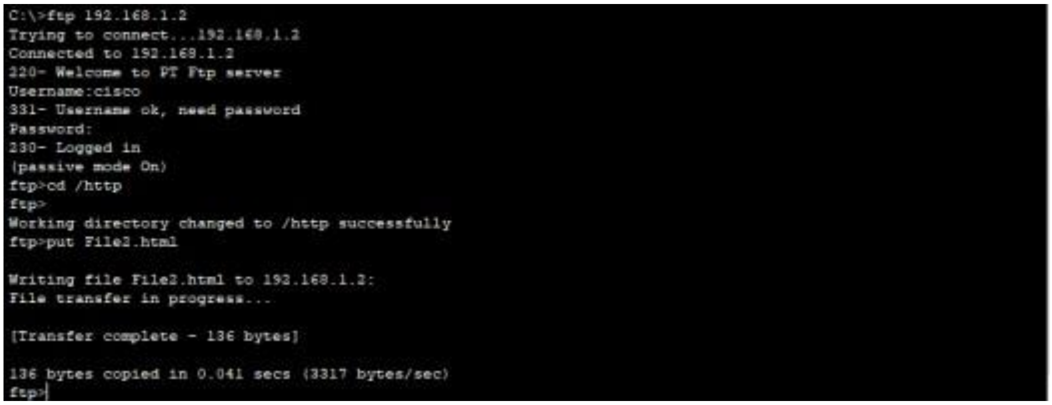


File2.html

File2 html.PNG

Now upload the file(File2.html) to the HTTP server using FTP. This is easy. We’ve already done it previously!

If you’re already in the HTTP directory, you just need to type: put File2.html. If no, first ftp the server(ftp 192.168.1.2), provide the login username(cisco) and password(cisco); change the current directory to HTTP(cd /http) , and finally upload the html file onto the HTTP directory(put File2.html)

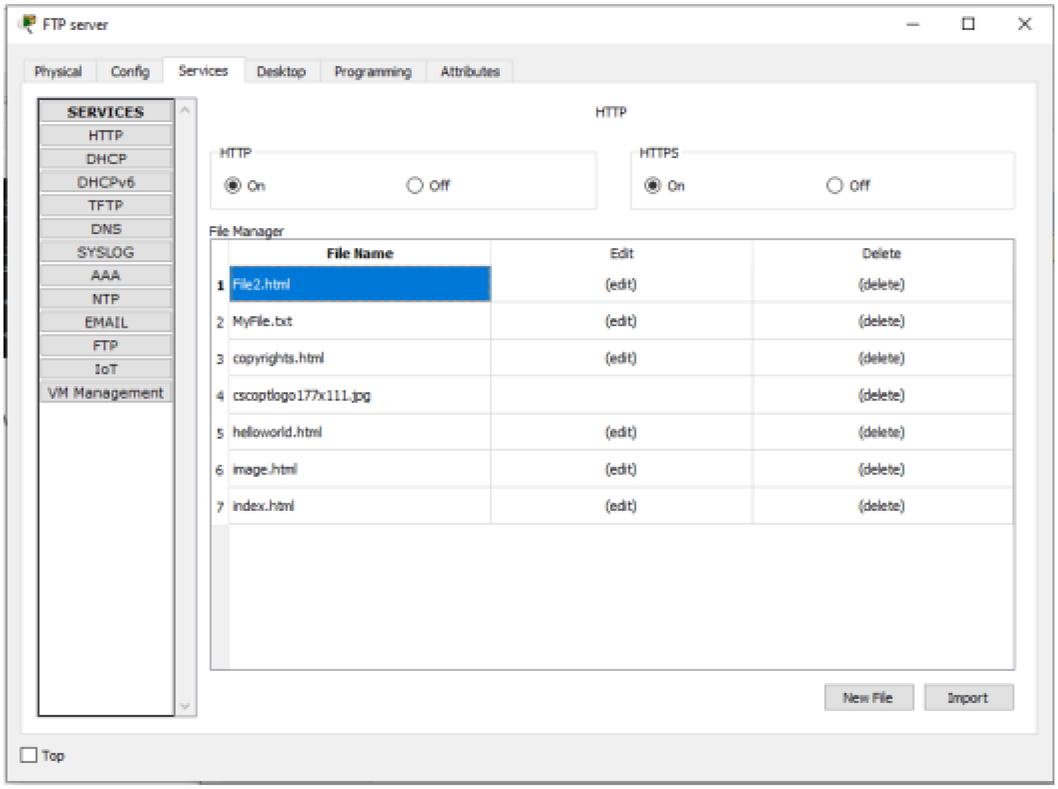


```
C:\>ftp 192.168.1.2
Trying to connect...192.168.1.2
Connected to 192.168.1.2
220- Welcome to PT Ftp server
Username:cisco
331- Username ok, need password
Password:
230- Logged in
(passive mode On)
ftp>cd /http
ftp>
Working directory changed to /http successfully
ftp>put File2.html

Writing file File2.html to 192.168.1.2:
File transfer in progress...

[Transfer complete - 136 bytes]

136 bytes copied in 0.041 secs (3317 bytes/sec)
ftp>
```

Sending File2. html to HTTP directory.PNG

Check whether the html file uploaded has been received in the HTTP directory:

Go to Server->Services-> HTTP. Then look up for the file in the File Manager.



HTTP

HTTP: ● On ○ Off

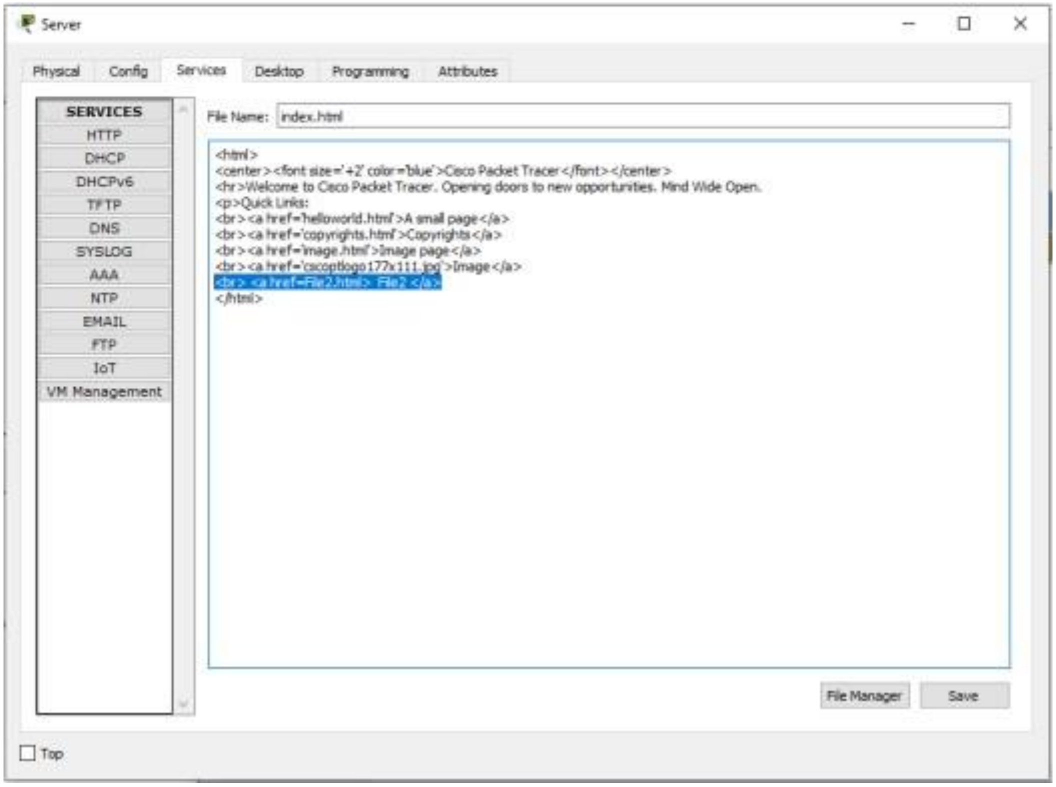
HTTPS: ● On ○ Off

File Manager

	File Name	Edit	Delete
1	File2.html	(edit)	(delete)
2	MyFile.txt	(edit)	(delete)
3	copyrights.html	(edit)	(delete)
4	cscoptlogo177x111.jpg		(delete)
5	helloworld.html	(edit)	(delete)
6	image.html	(edit)	(delete)
7	index.html	(edit)	(delete)

File2 HTML really uploaded into HTTP directory.PNG

Now edit index.html file in the HTTP directory so as to include a link to File2 that we've just uploaded. This will make File2 accessible from the Laptop's browser. To do this, locate index.html then click edit. Proceed to edit it as shown below. Then save and accept overwrite.Index.html editing to include File2 html.PNG



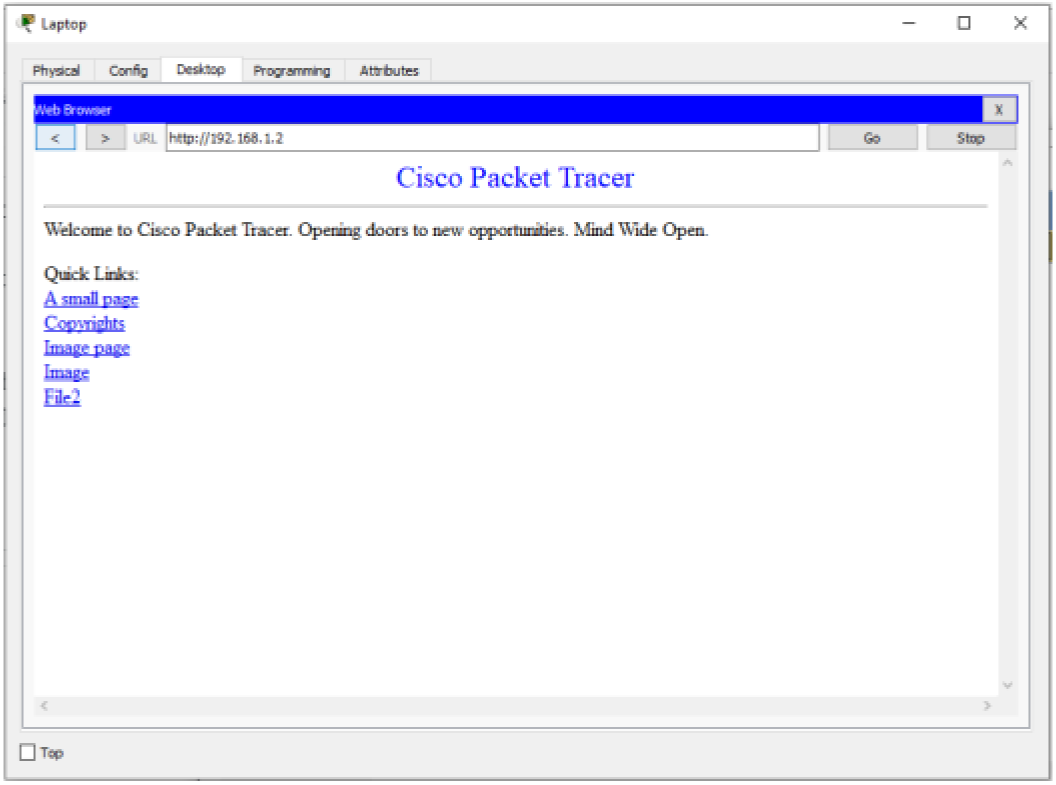
File Name: index.html

```
<html>
<center><font size='+2' color='blue'>Cisco Packet Tracer</font></center>
<hr>Welcome to Cisco Packet Tracer. Opening doors to new opportunities. Mind Wide Open.
<p>Quick Links:
<br><a href='helloworld.html'>A small page</a>
<br><a href='copyrights.html'>Copyrights</a>
<br><a href='image.html'>Image page</a>
<br><a href='cscoptlogo177x111.jpg'>Image</a>
<br> <a href=File2.html> File2 </a>
</html>
```

Finally, try to access the newly uploaded file from the Laptop's browser.

So go to the Laptop's browser and access the server using the server's IP address. By doing this, the browser is making an http request to the server. The server will respond to the Laptop with the index.html file containing a link to File2 which we've uploaded from the Laptop using FTP.

Http response with File2.PNG



http://192.168.1.2

Cisco Packet Tracer

Welcome to Cisco Packet Tracer. Opening doors to new opportunities. Mind Wide Open.

Quick Links:
A small page
Copyrights
Image page
Image
File2

Click File2 link to view the contents of the file in the browser.

Conclusion:- Studied and Analyzed the performance of HTTP, HTTPS and FTP protocol using Packet tracer tool.

Lab Assignment No.	9
Title	Illustrate the steps for implementation of S/MIME email security, POP3 through Microsoft® Office Outlook.
Roll No.	
Class	TE
Date of Completion	
Subject	Computer Network Laboratory
Assessment Marks	
Assessor's Sign	

ASSIGNMENT No: 09

Title: Illustrate the steps for implementation of S/MIME email security.

Problem Statement: Illustrate the steps for implementation of S/MIME email security, POP3 through Microsoft® Office Outlook.

Objective:

- Understand the concept and working of Encrypted mails

Outcomes:

- Students should understand regarding encryption in mail

Theory:

MIME (Secure/Multipurpose Internet Mail Extensions)

S/MIME allows users to send encrypted and digitally signed emails . This protocol allows recipients of the email to be certain the email they receive is the exact message that began with the sender. It also helps ensure that a message going to an outbound recipient is from a specific sender and not someone assuming a false identity.

How does S/MIME work?

S/MIME provides cryptographic-based security services like authentication, message integrity, and digital signatures. All these elements work together to enhance privacy and security for both the sender and recipient of an email.

S/MIME also works with other technologies such as Transport Layer Security (TLS) which encrypts the path between two email servers. The protocol is also compatible with Secure Sockets Layer (SSL) which masks the connection between email messages and Office 365 (a common email service) servers.

In addition, BitLocker works in conjunction with S/MIME protocol, which encrypts data on a hard drive in a data center so if a hacker gets access, he or she won't be able to interpret the information.

Benefits of encrypted email

1. Safeguards sensitive data

If you're sending information like your Social Security number over email, it's important that it's not easily stolen by hackers.

2. Economical

Instead of purchasing security equipment, you can simply rely on email encryption that's integrated directly on the server.

3. Timesaving

Instead of wasting time using several programs to make sure a connection is secure, you can rely on email encryption to do most of the work for you.

4. Regulation compliance

If you work in the healthcare industry, for example, and you haven't taken the right steps to secure medical data, you could be in violation of HIPAA laws [6]. Encryption helps you avoid those missteps.

5. Protects against malware

Malicious emails sometimes contain viruses masked as innocent email attachments. If you or someone else send an attachment using encrypted email, the email has a digital signature to prove its authenticity.

How does email encryption work?

If you don’t want anyone but the receiver to see the contents of a message, encryption is vital. To the outsider, an encrypted email will have a bunch of random letters, digits, or symbols instead of readable text. The person with the private key to decrypt it, typically the receiver, will be able to read the email as usual.

There are generally three encryption types available:

- S/MIME encryption works as long as both the sender and recipient have mailboxes that support it. Windows Outlook is the most popular version that works with this method. Gmail uses it as well.
- Office 365 Message Encryption is best for users with valid Microsoft Office licenses who can use this tool to encrypt the information and files sent via email. It’s also a top choice for Outlook users
- PGP/MIME is a more affordable and popular option that other email clients may prefer to use. It’s reliable and integrated into many of the apps we use today

Other email products may have their own brand of encryption, but the science behind it is the same. Only senders and recipients who have exchanged keys or digital signatures can communicate within the encrypted network.

How to send encrypted email in Outlook

Encrypting email may sound complicated, but it's not. Microsoft has a reputation for providing its users with simple ways to encrypt data, from files to folders to emails, too. It makes sense that they would include built-in tools for Outlook, their proprietary email system. You don't need a separate software tool or plug-in to start sending secure messages. Just follow these steps to begin.

1. Create a digital certificate

For Outlook users, encrypting a single email is simple. First, you must have a digital signature. To create a digital signature:

1. Start in your Outlook window and click on the File tab
2. Select Options, then Trust Center, then Trust Center Settings
3. Select Email Security, Get a Digital ID
4. You'll be asked to choose a certification authority. This is entirely up to you as most are rated the same
5. You'll receive an email with your digital certificate/ID included
6. Go back into Outlook and select Options and the Security tab
7. In the Security Settings Name field, type in a name of your choosing
8. Ensure that S/MIME is selected from the Secure Message Format box and that Default Security Settings is checked as well
9. Go to Certificates and Algorithms, select Signing Certificate, and click Choose
10. Make sure the box is checked next to Secure Email Certificate, and check the box next to"Send These Certificates with Signed Messages"
11. Click OK to save your settings and start using Outlook

2. Use your digital signature

Now that you have a digital ID, you need to start using it:

1. Open a new message to access the Tools tab
2. Click that, then Customize, and finally the Commands tab
3. From Categories, select Standard
4. From Categories, select Digitally Sign Message

3. Encrypt Outlook messages

You can now send encrypted messages to a recipient with the next steps.

1. Open the window to compose a new message and select the Options tab, then More Options
2. Click the dialog box (triangle with arrow pointing down) in the lower-right corner
3. Choose Security Settings and check the box next to Encrypt message contents and attachments
4. Write your message as normal and send

After you've sent and received a message that you've both signed and encrypted, you don't have to sign it again. Outlook will remember your signature.

4. Encrypt all Outlook messages

You can encrypt each one, or you can use the steps below to encrypt all outgoing messages in Outlook:

Open the File tab in Outlook

Select Options, then Trust Center, and Trust Center Settings

From the Email Security tab, select Encrypted email

Check the box next to Encrypt content and attachments for outgoing messages

Use Settings to customize additional options, including certificates

How to Send Encrypted Email

Have you ever wondered about the security of your private email conversations? Whether at work, school, or home, sending emails comes with a bit of a risk. There's one thing you can do to discourage data breaches and attacks on your sensitive data, however. Use encrypted email. Learn how to practice this common-sense method for communicating in our step-by-step guide. But first, let's look at why you should embrace encryption for your email correspondence.

How to Encrypt Email and Send Secure Messages

Emails sent over an open network can be intercepted and malicious actors can see email contents, attachments, or even take over your account.

To drive home the importance of email security, take a look at some alarming statistics that show the widespread cybersecurity issues that may have affected you in the past and still pose a threat today:

In 2016, 3 billion Yahoo accounts were hacked

According to research by cybersecurity company, Symantec, emails with a malicious URL make up a total of 12.3% of all emails

As these numbers illustrate, emails are a point of vulnerability for many unsuspecting users. However, it's not all doom and gloom, there are ways to protect yourself and your information.

To help safeguard against hackers and ensure your privacy is maintained, you can use encryption.

Encryption ensures that your emails remain unreadable, even if they fall into the wrong hands.

Conclusion: Thus we have studied the steps for implementation of S/MIME email security through Microsoft® Office Outlook.ets received from / sent to Face book, and how many of each were also HTTP packets successfully.

Lab Assignment No.	10
Title	To study the IPsec (ESP and AH) protocol by capturing the packets using Wireshark tool.
Roll No.	
Class	TE
Date of Completion	
Subject	Computer Network Laboratory
Assessment Marks	
Assessor's Sign	

ASSIGNMENT No: 10

Title: To study the IPsec (ESP and AH) protocol by capturing the packets using Wireshark tool.

Problem Statement: To study the IPsec (ESP and AH) protocol by capturing the packets using Wireshark tool.

Prerequisite:

- Protocols

Objective:

- To understand packets in networking
- Different protocols

Outcomes:

- Students should be able to know about the packets which are send and received while communication

Theory:

Wireshark is a free and open source packet analyzer used for network troubleshooting and analysis. These activities will show you how to use Wireshark to capture and analyze IPSEC traffic.

IPSEC (IP SECURITY PROTOCOL):

IPsec (Internet Protocol Security) is a framework that helps us to protect IP traffic on the network layer. Why? because the IP protocol itself doesn’t have any security features at all. IPsec can protect our traffic with the following features:

- **Confidentiality**: by encrypting our data, nobody except the sender and receiver will be able to read our data.
- **Integrity**: we want to make sure that nobody changes the data in our packets. By calculating a hash value, the sender and receiver will be able to check if changes have been made to the packet.
- **Authentication**: the sender and receiver will authenticate each other to make sure that we are really talking with the device we intend to.
- **Anti-replay**: even if a packet is encrypted and authenticated, an attacker could try to capture these packets and send them again. By using sequence numbers, IPsec will not transmit any duplicate packets.

IPsec can be used on many different devices, it's used on routers, firewalls, hosts and servers. Here are some examples how you can use it:

- Between two routers to create a site-to-site VPN that "bridges" two LANs together.
- Between a firewall and windows host for remote access VPN.
- Between two linux servers to protect an insecure protocol like telnet.

IKE builds the tunnels for but it doesn't authenticate or encrypt user data. We use two other protocols for this:

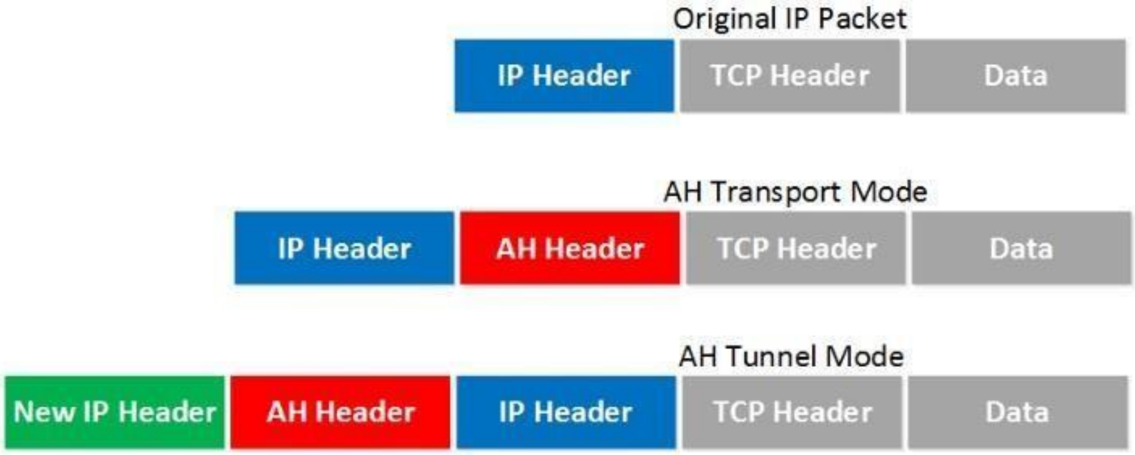
- **AH (Authentication Header)**
- **ESP (Encapsulating Security Payload)**

AH and ESP both offer authentication and integrity but only **ESP supports encryption**. Because of this, ESP is the most popular choice nowadays.

Both protocols support two different modes:

- **Transport mode**
- **Tunnel mode**

The main difference between the two is that with transport mode we will use the **original IP header**while in tunnel mode, we use a new **IP header**.



Transport mode is often between two devices that want to protect some insecure traffic (example: telnet traffic). **Tunnel mode** is typically used for site-to-site

IPsec Protocols

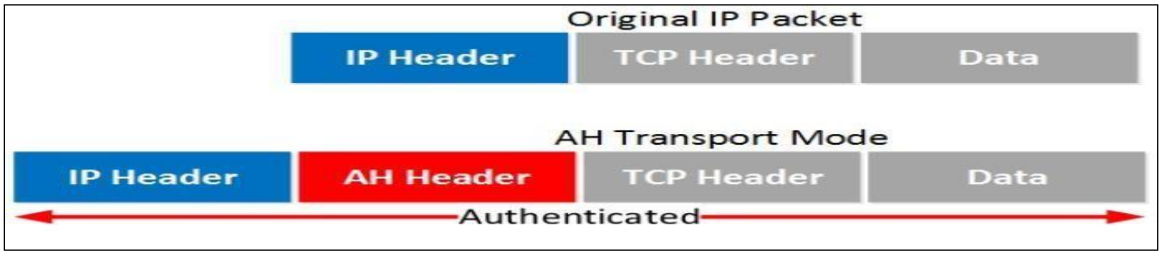
AH and/or ESP are the two protocols that we use to protect user data. Both of them can be used in transport or tunnel mode..

Authentication Header Protocol

AH offers authentication and integrity but it doesn't offer any encryption. It protects the IP packet by calculating a hash value over almost all fields in the IP header. The fields it excludes are the ones that can be changed in transit (TTL and header checksum). Let's start with transport mode…

AH Transport Mode

Transport mode is simple, it just adds an AH header after the IP header. Here's an example of an IP packet that carries some TCP traffic:



We can see here the AH header in between the IP header and ICMP header. This is a capture I took of a ping between two routers. You can see that AH uses 5 fields:

Next Header: this identifies the next protocol, ICMP in our example.

Length: this is the length of the AH header.

SPI (Security Parameters Index): this is an 32-bit identifier so the receiver knows to which flow this packet belongs.

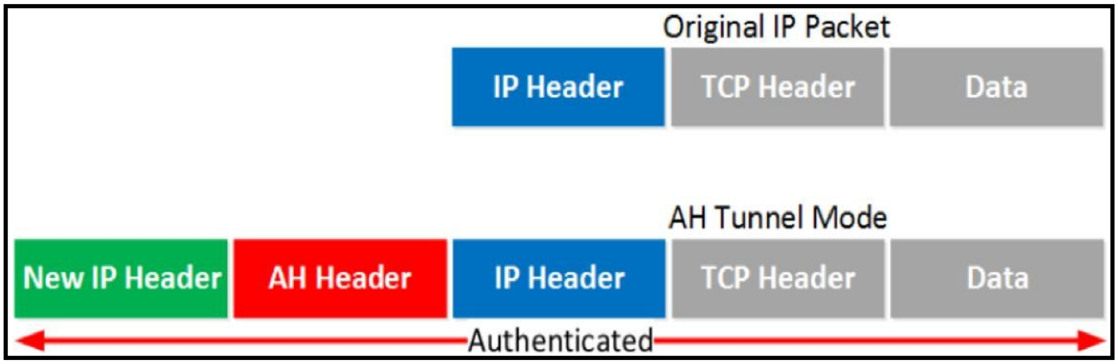
Sequence: this is the sequence number that helps against replay attacks.

ICV (Integrity Check Value): this is the calculated hash for the entire packet. The receiver also calculates a hash, when it's not the same you know something is wrong.

Let's continue with tunnel mode.

AH Tunnel Mode

With tunnel mode we add a new IP header on top of the original IP packet. This could be useful when you are using private IP addresses and you need to tunnel your traffic over the Internet. It's possible with AH but it doesn't offer encryption:



The entire IP packet will be authenticated. Here's what it looks like in wireshark:

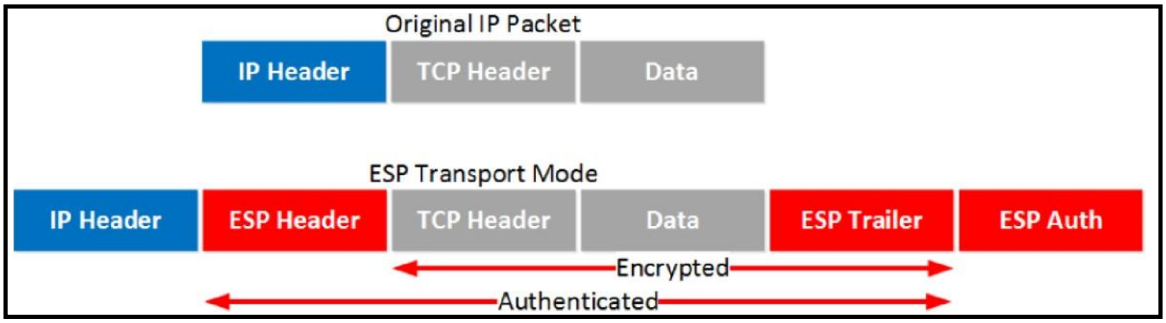
Here new IP header, then the AH header and finally the original IP packet that carries some ICMP traffic. One problem with AH is that it doesn't play well with NAT / PAT. Fields in the IP header like TTL and the checksum are excluded by AH because it knows these will change. The IP addresses and port numbers however are included. If you change these with NAT, the ICV of AH fails

ESP (Encapsulating Security Payload) Protocol

ESP is the more popular choice of the two since it allows you to encrypt IP traffic. We can use it in transport or tunnel mode, let's look at both.

ESP Transport Mode

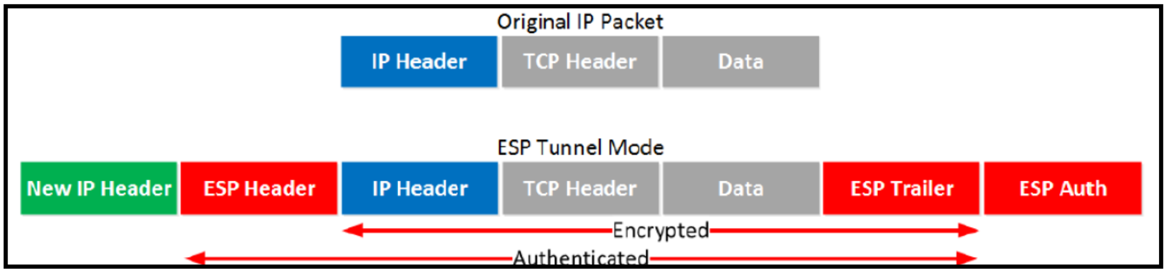
When we use transport mode, we use the original IP header and insert an ESP header. Here's what it looks like:



Transport layer (TCP for example) and payload will be encrypted. It also offers authentication but unlike AH, it's not for the entire IP packet. Here's what it looks like in wireshark:

ESP Tunnel Mode

How about ESP in tunnel mode? This is where we use a new IP header which is useful for site-to-site VPNs:



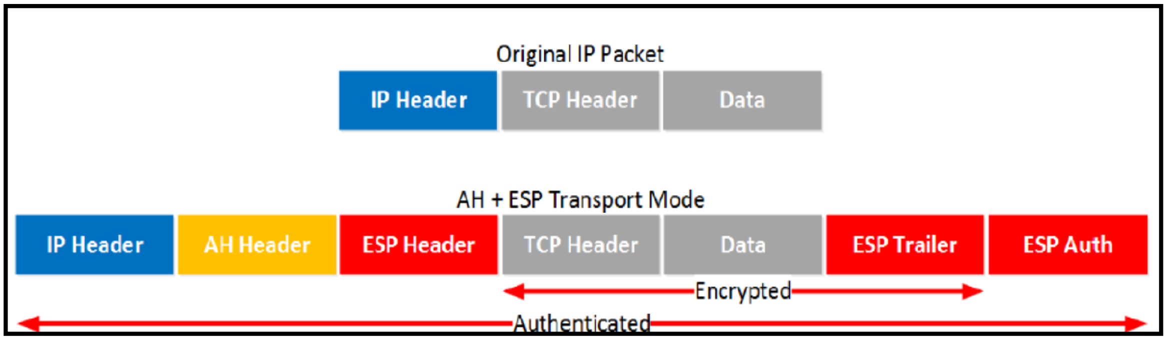
It's similar to transport mode but we add a new header. The original IP header is now also encrypted. Here's what it looks like in wireshark:

AH and ESP

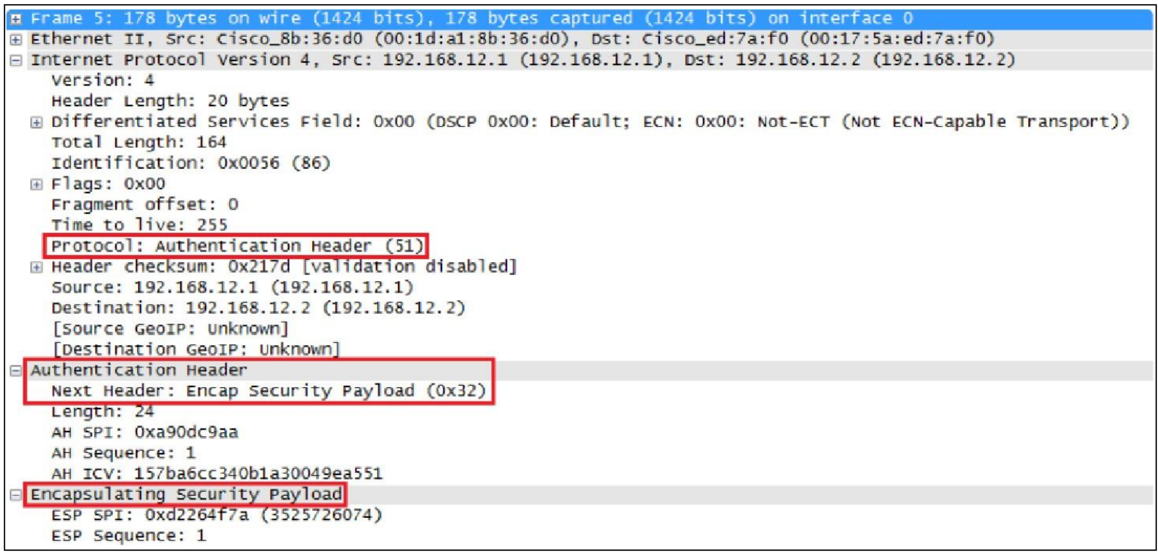
It is possible to use AH and ESP at the same time.

Transport Mode

Let’s start with transport mode, here’s what the IP packet will look like:



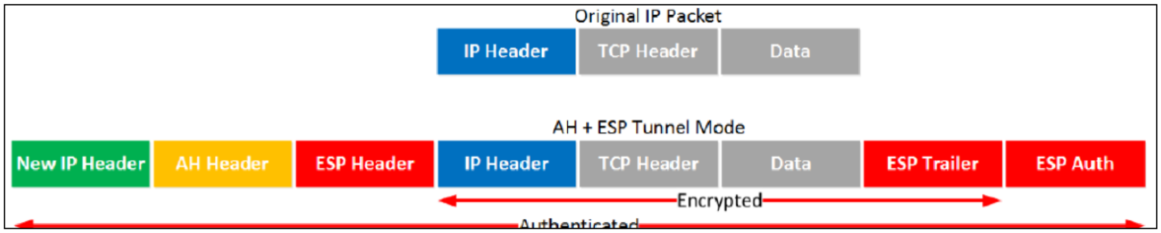
With transport mode we will use the original IP header, followed by an AH and ESP header. The transport layer, payload and ESP trailer will be encrypted.Because we also use AH, the entire IP packet is authenticated. Here’s what it looks like in wireshark:



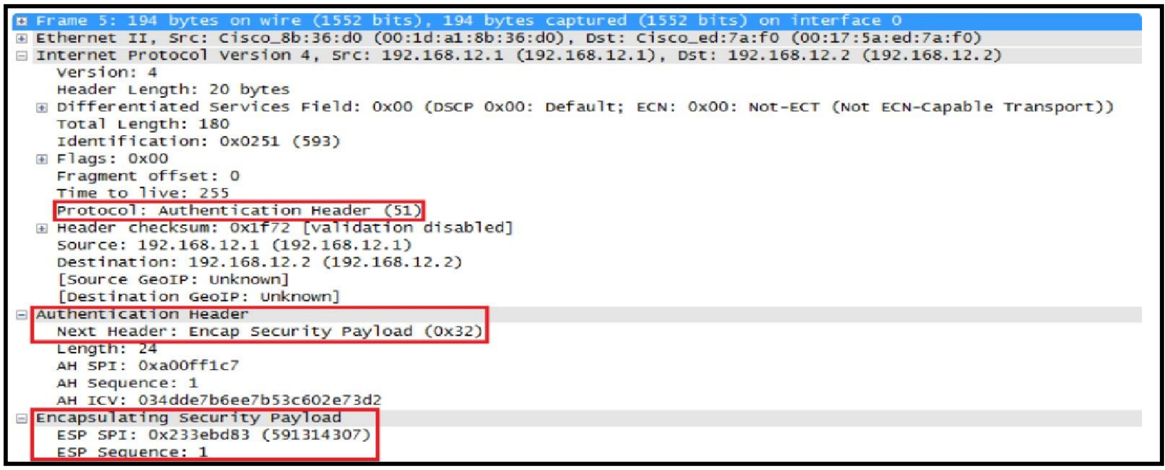
```
⊞ Frame 5: 178 bytes on wire (1424 bits), 178 bytes captured (1424 bits) on interface 0
⊞ Ethernet II, Src: Cisco_8b:36:d0 (00:1d:a1:8b:36:d0), Dst: Cisco_ed:7a:f0 (00:17:5a:ed:7a:f0)
⊟ Internet Protocol Version 4, Src: 192.168.12.1 (192.168.12.1), Dst: 192.168.12.2 (192.168.12.2)
    Version: 4
    Header Length: 20 bytes
  ⊞ Differentiated Services Field: 0x00 (DSCP 0x00: Default; ECN: 0x00: Not-ECT (Not ECN-Capable Transport))
    Total Length: 164
    Identification: 0x0056 (86)
  ⊞ Flags: 0x00
    Fragment offset: 0
    Time to live: 255
    Protocol: Authentication Header (51)
  ⊞ Header checksum: 0x217d [validation disabled]
    Source: 192.168.12.1 (192.168.12.1)
    Destination: 192.168.12.2 (192.168.12.2)
    [Source GeoIP: Unknown]
    [Destination GeoIP: Unknown]
⊟ Authentication Header
    Next Header: Encap Security Payload (0x32)
    Length: 24
    AH SPI: 0xa90dc9aa
    AH Sequence: 1
    AH ICV: 157ba6cc340b1a30049ea551
⊟ Encapsulating Security Payload
    ESP SPI: 0xd2264f7a (3525726074)
    ESP Sequence: 1
```

Above you can see the original IP packet, the AH header and the ESP header.

Tunnel Mode



First we will have a new IP header followed by the AH and ESP header. The original IP packet will be completely encrypted and everything will be authenticated thanks to AH. Here's what it looks like in wireshark:



```
⊞ Frame 5: 194 bytes on wire (1552 bits), 194 bytes captured (1552 bits) on interface 0
⊞ Ethernet II, Src: Cisco_8b:36:d0 (00:1d:a1:8b:36:d0), Dst: Cisco_ed:7a:f0 (00:17:5a:ed:7a:f0)
⊟ Internet Protocol Version 4, Src: 192.168.12.1 (192.168.12.1), Dst: 192.168.12.2 (192.168.12.2)
    Version: 4
    Header Length: 20 bytes
  ⊞ Differentiated Services Field: 0x00 (DSCP 0x00: Default; ECN: 0x00: Not-ECT (Not ECN-Capable Transport))
    Total Length: 180
    Identification: 0x0251 (593)
  ⊞ Flags: 0x00
    Fragment offset: 0
    Time to live: 255
    Protocol: Authentication Header (51)
  ⊞ Header checksum: 0x1f72 [validation disabled]
    Source: 192.168.12.1 (192.168.12.1)
    Destination: 192.168.12.2 (192.168.12.2)
    [Source GeoIP: Unknown]
    [Destination GeoIP: Unknown]
⊟ Authentication Header
    Next Header: Encap Security Payload (0x32)
    Length: 24
    AH SPI: 0xa00ff1c7
    AH Sequence: 1
    AH ICV: 034dde7b6ee7b53c602e73d2
⊟ Encapsulating Security Payload
    ESP SPI: 0x233ebd83 (591314307)
    ESP Sequence: 1
```

Above you can see the new IP header followed by the AH and ESP header.

Conclusion : Thus we analyzed the IPSec protocol with the help of Wireshark tool.

Group C

Lab Assignment No.	11
Title	Installing and configuring DHCP server and assign IP addresses to client machines using DHCP server.
Roll No.	
Class	TE
Date of Completion	
Subject	Computer Network Laboratory
Assessment Marks	
Assessor's Sign	

ASSIGNMENT No: 11

Title: Installing and configuring DHCP server and assign IP addresses to client machines using DHCP server.

Problem Statement: Installing and configuring DHCP server and assign IP addresses to client machines using DHCP server.

Prerequisite:

1. Knowledge about IP and Subnets.
2. Linux basic commands.

Learning Objectives:

1. Understand the concept of DHCP.
2. Configuring DHCP and installation of software.

New Concepts:

1. Crimping
2. Access Point Configuration

Outcomes:

- Students could be able to know different methods associated with client and server

Theory:

- DHCP (Dynamic Host Configuration Protocol) is a protocol that lets network administrators manage centrally and automate the assignment of IP (Internet Protocol) configurations on a computer network.
- When using the Internet's set of protocols (TCP/IP), in order for a computer system to communicate to another computer system it needs a unique IP address.
- Without DHCP, the IP address must be entered manually at each computer system. DHCP lets a network administrator supervise and distribute IP addresses from a central point.
- The purpose of DHCP is to provide the automatic (dynamic) allocation of IP client configurations for a specific time period (called a lease period) and to eliminate the work

necessary to administer a large IP network.

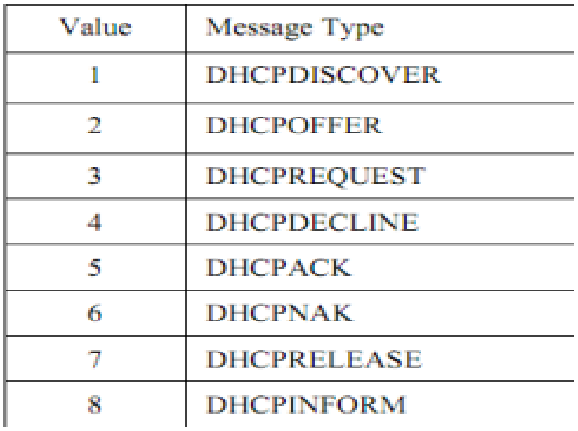
- When connected to a network, every computer must be assigned a unique address.
- However, when adding a machine to a network, the assignment and configuration of network (IP) addresses has required human action.
- The computer user had to request an address, and then the administrator would manually configure the machine. Mistakes in the configuration process are easy for novices to make, and can cause difficulties for both the administrator making the error as well as neighbors on the network. Also, when mobile computer users travel between sites, they have had to relive this process for each different site from which they connected to a network.
- In order to simplify the process of adding machines to a network and assigning unique IP addresses manually, there is a need to automate the task.
- The introduction of DHCP alleviated the problems associated with manually assigning TCP/IP client addresses. Network administrators have quickly appreciated the importance, flexibility and ease-of-use offered in DHCP.

Advantages of DHCP:-

DHCP has several major advantages over manual configurations.

- Each computer gets its configuration from a "pool" of available numbers automatically for a specific time period (called a leasing period), meaning no wasted numbers.
- When a computer has finished with the address, it is released for another computer to use. Configuration information can be administered from a single point.
- Major network resource changes (e.g. a router changing address), requires only the DHCP server be updated with the new information, rather than every system.

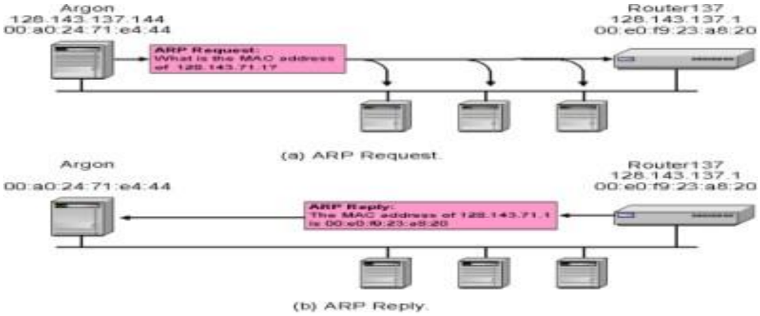
DHCP message types:



Value	Message Type
1	DHCPDISCOVER
2	DHCPOFFER
3	DHCPREQUEST
4	DHCPDECLINE
5	DHCPACK
6	DHCPNAK
7	DHCPRELEASE
8	DHCPINFORM

DHCP Operations:-

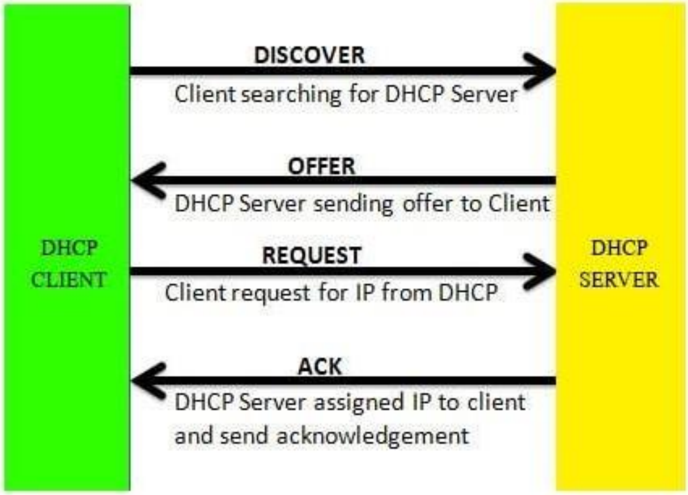
1. DHCP Discover



(a) ARP Request.

(b) ARP Reply.

2. DHCP Offer


DHCP
CLIENT
DISCOVER
Client searching for DHCP Server
OFFER
DHCP Server sending offer to Client
REQUEST
Client request for IP from DHCP
ACK
DHCP Server assigned IP to client
and send acknowledgement
DHCP
SERVER

3. DHCP Discover: At this time, the DHCP client can start to use the IP address

4. DHCP Release: At this time, the DHCP client has released the IP address

Installing DHCP in Ubuntu:

Open terminal and type following commands:-

1. sudo apt-get install isc-dhcp-server
2. sudo gedit /etc/dhcp/dhcpd.conf then make changes in file....

```
default-lease-time 600; max-lease-time 7200;

option subnet-mask 255.255.255.0;

option broadcast-address 10.1.32.255;

subnet 192.168 1.0 netmask 255.255.255.0

range 10.1.32.10 10.1.32.20; }
```

3. save file and close
4. again on terminal give following commands....

```
sudo service isc-dhcp-server restart

sudo service isc-dhcp-server start
```

5. On another PC in Internet properties change to Obtain IP address automatically and then check the IP address.

Conclusion:

Hence we Installed and Configured DHCP and studied Installation of Software on remote Machine.

Lab Assignment No.	12
Title	Write a program for DNS lookup. Given an IP address input, it should return URL and vice versa.
Roll No.	
Class	TE
Date of Completion	
Subject	Computer Network Laboratory
Assessment Marks	
Assessor's Sign	

ASSIGNMENT No: 12

Title: Write a program for DNS lookup.

Problem Statement: Write a program for DNS lookup. Given an IP address input, it should return URL and vice versa.

Prerequisite:

- IP Address and OSI & TCP/IP Model.
- Role of different servers.
- Working of internet.

Objectives:

- Understand what is Domain Name System and DNS lookup working.
- Understand what is DNS Structure and Hierarchy.

New Concepts:

- Name Server and Domain Name System.
- DNS lookup, Zone

Outcomes:

- Students should be able to know about the role of different servers.
- Students should be able to know about DNS and its working.

Theory:

Computers, services, or other resources connected to the Internet or a private network. It associates various information with domain names assigned to each of the participating entities.

it translates more readily memorized domain names to the numerical IP addresses needed for locating and identifying computer services and devices with the underlying network protocols. By providing a worldwide, distributed directory service, the Domain Name System is an essential component of the functionality on the Internet that has been in use since 1985.

HOST.TXT files:

The ARPANET, the predecessor of the Internet, had no distributed host name database. Each network node maintained its own map of the network nodes as needed and assigned those names that were memorable to the users of the system.

The hosts file contains lines of text consisting of an IP address in the first text field followed by one or more host names. Each field is separated by white space – tabs are often preferred for historical reasons, but spaces are also used. Comment lines may be included; they are indicated by an octothorpe (#) in the first position of such lines. Entirely blank lines in the file are ignored. For example, a typical hosts file may contain the following:

127.0.0.1 localhost loopback

::1 localhost

Domain Name Space

The domain name space refers a hierarchy in the internet naming structure. This hierarchy has multiple levels (from 0 to 127), with a root at the top. The following diagram shows the domain name space hierarchy:

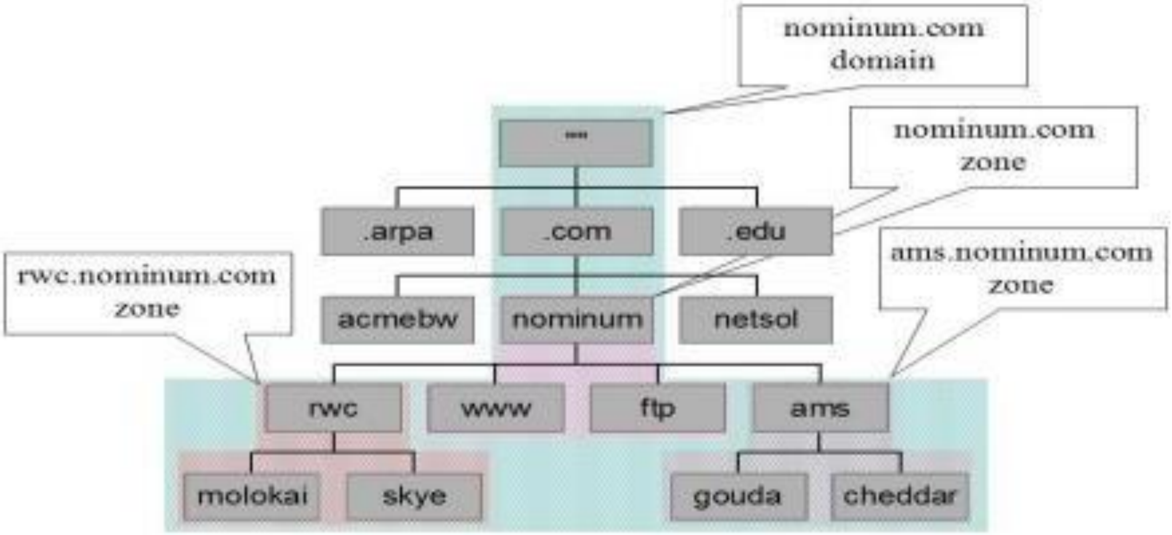
Name Server

Name server contains the DNS database. This database comprises of various names and their corresponding IP addresses. Since it is not possible for a single server to maintain entire DNS database, therefore, the information is distributed among many DNS servers.

- Hierarchy of server is same as hierarchy of names.
- The entire name space is divided into the zones

Zones

Zone is collection of nodes (sub domains) under the main domain. The server maintains a database called zone file for every zone.



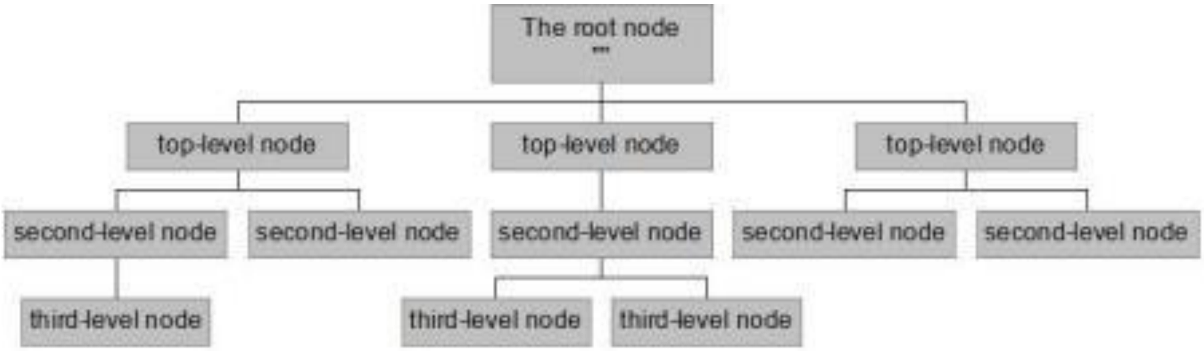
If the domain is not further divided into sub domains then domain and zone refers to the same thing.

The information about the nodes in the sub domain is stored in the servers at the lower levels however; the original server keeps reference to these lower levels of servers.

Types of Name Servers

Following are the three categories of Name Servers that manages the entire Domain Name System:

1. Root Server
2. Primary Server
3. Secondary Server



Root Server

Root Server is the top level server which consists of the entire DNS tree. It does not contain the information about domains but delegates the authority to the other server

Primary Servers

Primary Server stores a file about its zone. It has authority to create, maintain, and update the zone file.

Secondary Server

Secondary Server transfers complete information about a zone from another server which may be primary or secondary server. The secondary server does not have authority to create or update a zone file.

How does DNS work?

DNS servers answer questions from both inside and outside their own domains. When a server receives a request from outside the domain for information about a name or address inside the domain, it provides the authoritative answer. When a server receives a request from inside its own domain for information about a name or address outside that domain, it passes the request out to another server -- usually one managed by its internet service provider. If that server does not know the answer or the authoritative source for the answer, it will reach out to the DNS servers for the top-level domain -- e.g., for all of .com or .edu. Then, it will pass the request down to the authoritative server for the specific domain -- e.g., techtarget.com or stkate.edu; the answer flows back along the same path.

How DNS Lookup Works

By now, you know that there are different servers hosting databases that contain the IP addresses of different domains and their sub-domains. You also know that there are Root Servers that hold the IP address of servers hosting Top Level Domains. These Root Servers help in reaching the servers containing databases that hold IP address of the main domain name. If there are sub-domains, their address can be on the same servers as of the main domain name or on a different server. All these servers are accessible for finding out the IP address of the exact URL that you need to use.

- **Forward Lookup:** When a name query is send to the DNS server against to IP address, it is generally said a forward lookup.

- **Reverse Lookup:** DNS also provides a reverse lookup process, enabling clients to use a known IP address during a name query and look up a computer name based on its address.

The process of finding out the IP address of any URL on the Internet is known as DNS lookup.

To find out how DNS Lookup works, take the following example.

Example: Consider a network of ten computers. Each computer has its own address so that data packets travelling in the network know where to go. There is a 11th computer that hosts a database containing the alias names of each of these ten computers and their IP addresses. While the computer users can refer to the computers using their names, the data packets need the IP addresses of the computers so that they can reach the intended recipient. If computer A needs to use the printer attached to computer B, A will check the database on 11th computer to know the IP address of B and then find out the address of printer attached to B. Only after obtaining the address of the printer, A will route the print command to printer attached to B.

In this case, the following iterations happen:

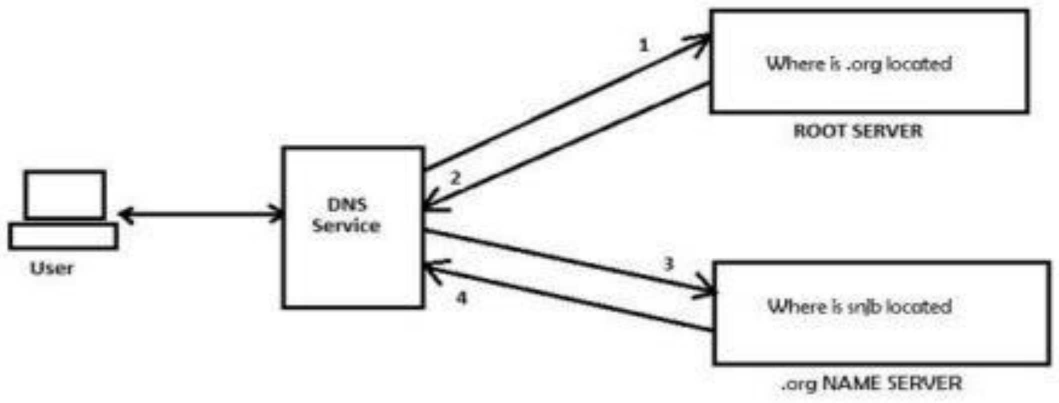
A contacts Computer **11**

A contacts **B**

A contacts printer attached to **B**

A similar method is used to lookup DNS records. For example, when you click on http://snjb.org, your router will contact your default DNS Service for DNS resolution. The DNS service will contact Root Servers and ask for the IP address of server containing **.org** records. This address is sent back to your DNS service. The DNS service again reaches the Name Server containing addresses of **.org** domains and asks it for the address of http://snjb.org. Upon obtaining the IP address of the servers that host snjb.org, your DNS service will return the IP address to your computer which then fires up your browser to download the main webpage. This means your DNS service is sending at least two requests to receive the IP address of a simple domain name.

Following is an image that explains how DNS lookup works:



Understanding How DNS Lookup Works

In the above case, if you were to look for http:// http://snjb.academiaerp.com/snjb/Login ,
your DNS service had to run a request extra to know its IP address.

Since resolving DNS from scratch every time takes up time, many ISPs and DNS Service Providers create local caches that contain already resolved addresses. These are primarily the addresses they already fetched from Root Servers and other Name Servers at some point of time. In this case, when you send a request for a URL, instead of contacting the Root server directly, the DNS service would look up for the resolved address of the URL in its local DNS cache. If found, it would send the resolution back to your computer instantly else would go ahead and resolve the DNS using the above method of contacting Root Servers and other Name Servers.

Some operating systems too contain a local cached copy of addresses that you commonly use on your computer. This too, helps in saving time while using the Internet. We will talk about DNS caches in a different article at some later point of time.

Conclusion:

Hence we conclude that we have lookup the URL which we want to visit the request is travels to local router to DNS server and it resolve the query as possible otherwise it forwards the query to next DNS hop.