Pravara Rural Education Society's

Pravara Rural Engineering College, Loni

Tal: - Rahata, Dist: - Ahmednagar, Loni-413736 (M.S.)

Department of Computer Engineering

<u>Lab Manual</u>

310247: Computer Network and Security Laboratory

Third Year of Computer Engineering (2019 Course)

Department of Computer Engineering

TE (Computer Engineering 2019 Course) Semester –ISubject:

310247: Computer Network and Security Laboratory

Manual Content

11.	Write a program for DNS lookup. Given an IP address as input, it should return URL and viceversa	
12.	Installing and configure DHCP server and write a program to install the software on remote machine.	
13.	Capture packets using Wireshark, write the exact packet capture filter expressions to accomplish the following and save the output in file: 1. Capture all TCP traffic to/from Facebook, during the time when you log in to your Facebook account 2. Capture all HTTP traffic to/from Facebook, when you log in to your Facebook account 3. Write a DISPLAY filter expression to count all TCP packets (captured under item #1) that have the flags SYN, PSH, and RST set. Show the fraction of packets that had each flag set. 4. Count how many TCP packets you received from / sent to Face book, and how many of each were also HTTP packets.	
14.	Study and Analyze the performance of HTTP, HTTPS and FTP protocol using Packet tracer tool.	
15.	To study the SSL protocol by capturing the packets using Wireshark tool while visiting any SSL secured website (banking, e-commerce etc.).	
16.	. Illustrate the steps for implementation of S/MIME email security through Microsoft® Office Outlook.	
17.	To study the IPsec (ESP and AH) protocol by capturing the packets using Wireshark tool.	

Verified by:

Academic Coordinator HOD

About College

Pravara Rural Engineering College, Loni, started on 21“August 1983, is the premier of all the institutions under the technical education complex developed by the Pravara Rural Education Society. The Society was established in 1964 by Late Padmashri Dr.Vitthalrao Vikhe Patil, who was also founder of the Pravara Sahakari Sakhar Karkhana Ltd, Pravaranagar. The Society has been registered under the Societies Registration Act 1960 and the Mumbai Public Trust Act 1950.

Vision of the Institute

Enrich the youth with skills and values to enable them to contribute in the development of society: nationally and globally.

Mission of the Institute

To provide quality technical education through effective teaching-learning and research to foster the youth with skills and values to make them capable of delivering significant contribution in local to global development.

About Department

Computer Engineering Department established in 1985 with the vision to produce world class quality computer engineers. PG in Computer Engineering started in 2014. Since its inception the Department has gained reputation in the S P Pune University as a renowned department for its quality in academics and treating students and alumni as Ambassadors of institute. More than 1800 alumni are working in India and abroad in leading multinational companies, government and various research organizations such as IBM, CAPGEMINI, SYNTEL, TCS and renowned top industries. The department has an environmental friendly infrastructure, well equipped laboratories and qualified, experienced staff.

Vision of the Department

To develop Computer Engineering graduates capable of exhibiting leadership, creativity and skills for improving the quality of life.

Mission of the Department

To provide a platform for self-learning to meet the challenges of changing technology and build leadership qualities to succeed in professional career.

Program Outcomes:

PO1	**Engineering Knowledge:** Apply the knowledge of mathematics, science, engineering fundamentals, and computer engineering to the solution of complex engineering problems.
PO2	**Problem analysis:** Identify, formulate, review research literature, and analyze complex computer engineering problems reaching substantiated conclusions using first principles of mathematics, natural sciences, and engineering sciences.
PO3	**Design/Development of solutions:** Design solutions for complex computer engineering problems and design system components or processes that meet the specified needs with appropriate consideration for the public health and safety, and the cultural, societal, and environmental considerations.
PO4	**Conduct Investigations of complex problems:** Use research based knowledge and research methods including design of experiments, analysis and interpretation of data, and synthesis of the information to provide valid conclusions.
PO5	**Modern Tool usage:** Create, select, and apply appropriate techniques, and modern engineering and IT tools including prediction and modeling to complex computer engineering activities with an understanding of the limitations.
PO6	**The engineer and society:** Apply reasoning informed by the contextual knowledge to assess society, health, safety, legal and cultural issues and the consequent responsibilities relevant to the computer engineering practice.
PO7	**Environment and Sustainability:** Understand the computer engineering solutions in societal and environmental contexts, and demonstrate the knowledge of, and need for sustainable development.
PO8	**Ethics:** Apply ethical principles and commit to computer engineering ethics and responsibilities and norms of the engineering practice.
PO9	**Individual and team work:** Function effectively as an individual, and as a member or leader in diverse teams, and in multidisciplinary settings.
PO10	**Communication:** Communicate effectively on complex computer engineering activities with the engineering community and with society at large, such as, being

	able to comprehend and write effective reports and design documentation, make effective presentations, and give and receive clear instructions.
PO11	**Project Management and Finance:** Demonstrate knowledge and understanding of computer engineering and management principles and apply these to one's own work, as a member and leader in a team, to manage projects and in multidisciplinary environments.
PO12	**Life-long learning:** Recognize the need for, and have the preparation and ability to engage in independent and life-long learning in broadest context of technological change.

Program Specific Outcomes:

PSO1:	The ability to understand, analyze and develop computer programs in the areas related to Algorithms, System Software, Machine Learning, Artificial Intelligence, Web Applications, Big Data Analytics and Networking for efficient design of computer based systems of varying complexity.
PSO2:	The ability to apply standard practices and strategies in software / embedded project development using open source programming tools and environment to deliver a quality product for business success.
PSO3:	The ability to employ modern computer language, environment and platforms in creating innovative career paths to be an entrepreneur and path for higher studies.

Program Educational Outcomes (PEOs):

1. To work in a multidisciplinary environment by providing solutions to real time problems.
2. To develop computer programmers on design and programming techniques, technologies and tools related to computer engineering.
3. To inculcate, in students, professional and ethical attitude, effective communication skills, team work skill and ability to relate engineering issues in broader social context.

CO-PO AND CO-PSO MAPPING MATRIX

Class: TE Computer (2019 Pattern)	Sub: Computer Network and Security Laboratory (310247)

Course Outcomes (CO): Course outcomes are the statements of what a studentshould know, understand and/or be able to demonstrate after completion of a course.

Course Outcomes	Description of COs
C347.1	**CO1:** Analyze**[L4: Analyzing]** the requirements of network types, topology and transmission media
C347.2	**CO2:** Experiment with **[L3: Apply]** error control, flow control techniques and protocols and analyze them
C347.3	**CO3:** Experiment with **[L3: Apply]** the subnet formation with IP allocation mechanism and apply various routing algorithms
C347.4	**CO4**: Develop **[L6:Creating]**Client-Server architectures and prototypes
C347.5	**CO5:** Experiment with **[L3: Apply]** web applications and services using application layer protocols
C347.6	**CO6:** Use **[L3: Apply]** network security services and mechanisms

CO-PO and CO-PSO Mapping Matrix:

Course Outcomes	PO1	PO2	PO3	PO4	PO5	PO6	PO7	PO8	PO9	PO10	PO11	PO12	PSO1	PSO2	PSO3
C454.1	1	2	2		2	1	1			1	1	1	3	2	1
C454.2	1	3		1	1					1			3	2	1
C454.3	3	2	1	1				1		1		1	3	2	1
C454.4	1	1	2	1	1	1		1		1		1	3	2	1
C454.5	2	3			1				1	1			3	2	1
C454.6	1	1	3	1	1		1		1	1		1	3	2	1

Guidelines for Instructor's Manual

The instructor‘s manual is to be developed as a reference and hands-on resource. It should include prologue (about University/program/ institute/ department/foreword/ preface), curriculumof the course,
Conduction and Assessment guidelines, topics under consideration, concept, objectives, outcomes,set of typical applications/assignments/ guidelines, and references.

Guidelines for Student's Laboratory Journal

The laboratory assignments are to be submitted by student in the form of journal. Journal consists of Certificate, table of contents, and handwritten write-up of each assignment (Title, Date of Completion, Objectives, Problem Statement, Software and Hardware requirements, Assessment grade/marks and assessor's sign, Theory- Concept in brief, algorithm, flowchart, test cases, Test Data Set(if applicable), mathematical model (if applicable), conclusion/analysis. Program codes with sample output of all performed assignments are to be submitted as softcopy. As a conscious effort and little contribution towards Green IT and environment awareness, attaching printed papers as part of write-ups and programlisting to journal must be avoided. Use of DVD containing students programs maintained by Laboratory In-charge is highly encouraged. For reference one or two journals may be maintained with program prints in the Laboratory.

Guidelines for Laboratory /Term Work Assessment

Continuous assessment of laboratory work should be based on overall performance of Laboratory assignments by a student. Each Laboratory assignment assessment will assign grade/marks based on parameters, such as timely completion, performance, innovation, efficient codes, punctuality.

Guidelines for Oral Examination

Oral examination should be jointly conducted by the internal examiner and external examiner. Relevant questions may be asked at the time of evaluation to test the student‘s understanding of the fundamentals, effective and efficient implementations in term work. This will encourage, transparent evaluation and fair approach, and hence will not create any uncertainty or doubt in the minds of the students. So, adhering to these principles will consummate our team efforts to the promising start of student's academics.

Guidelines for Laboratory Conduction

The instructor is expected to frame the assignments by understanding the prerequisites, technological aspects, utility and recent trends related to the topic. The assignment framing policyneed to address the average students and inclusive of an element to attract and promote the intelligent students. Use of open source software is encouraged. Use of open source software is encouraged. Based on the concepts learned. Instructor may also set one assignment or mini-project that is suitable to respective branch beyond the scope of syllabus.

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

Programming tools recommended: - Open-Source /C/C++/JAVA Programming tool
like G++/GCC, Wireshark/Ethereal and Packet Tracer

Virtual Laboratory:

- http://vlabs.iitb.ac.in/vlab/

Suggested List of Laboratory Experiments/Assignments

Assignments from all Groups (A,B,C) are compulsory.

Experiment Number: A1

Lab Assignment on Unit I: (Mandatory Assignment):

Title: Study of Existing LAN

OBJECTIVES:

1. To understand the structure and working of various networks including the interconnecting devices used in them.

2. To get hands on experience of making and testing cables

PROBLEM STATEMENT

Part A: Setup a wired LAN using Layer 2 Switch and then IP switch of minimum four computers. It includes preparation of cable, testing of cable using line tester, configuration machine using IP addresses, testing using PING utility and demonstrate the PING packets captured traces using Wireshark Packet Analyzer Tool.

TYPES OF NETWORK

Common examples of area network types are:

LAN - Local Area Network
WLAN - Wireless Local Area Network
WAN - Wide Area Network
MAN - Metropolitan Area Network
SAN - Storage Area Network, System Area Network, Server Area Network, or sometimes Small Area Network
CAN - Campus Area Network, Controller Area Network, or sometimes Cluster Area Network
PAN - Personal Area Network
DAN - Desk Area Network

LAN - Local Area Network

A LAN connects network devices over a relatively short distance. A networked office building, school, or home usually contains a single LAN, though sometimes one building will contain a few small LANs (perhaps one per room), and occasionally a LAN will span a group of nearby buildings.

MAN-Metropolitan Area Network

A network spanning a physical area larger than a LAN but smaller than a WAN, such as a city. A MAN is typically owned and operated by a single entity such as a government body or large corporation.

WAN - Wide Area Network

As the term implies, a WAN spans a large physical distance. The Internet is the largest WAN, spanning the Earth.

A WAN is a geographically-dispersed collection of LANs. A network device called a router connects LANs to a WAN. In IP networking, the router maintains both a LAN address and a WAN address.

TYPES OF CABLES

Cable is the medium through which information usually moves from one network device to another. There are several types of cable which are commonly used with LANs. In some cases, a network will utilize only one type of cable, other networks will use a variety of cable types. The type of cable chosen for a network is related to the network's topology, protocol, and size. Understanding the characteristics of different types of cable and how they relate to other aspects of a network is necessary for the development of a successful network.

The following sections discuss the types of cables used in networks and other related topics.

- Unshielded Twisted Pair (UTP) Cable
- Shielded Twisted Pair (STP) Cable
- Coaxial Cable
- Fiber Optic Cable
- Cable Installation Guides
- Wireless LANs

Unshielded Twisted Pair (UTP) Cable

Twisted pair cabling comes in two varieties: shielded and unshielded. Unshielded twisted pair (UTP) is the most popular and is generally the best option for school networks

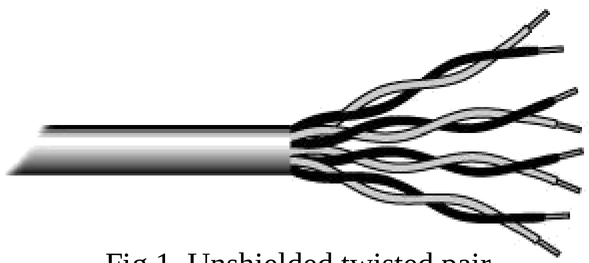


Fig.1. Unshielded twisted pair

The quality of UTP may vary from telephone-grade wire to extremely high-speed cable. The cable has four pairs of wires inside the jacket. Each pair is twisted with a different number of twists per inch to help eliminate interference from adjacent pairs and other electrical devices. The tighter the twisting, the higher the supported transmission rate and the greater the cost per foot. The EIA/TIA (Electronic Industry Association/Telecommunication Industry Association) has established standards of UTP and rated six categories of wire (additional categories are emerging).

Categories of Unshielded Twisted Pair

Category	**Speed**	**Use**
1	1 Mbps	Voice Only (Telephone Wire)
2	4 Mbps	LocalTalk & Telephone (Rarely used)
3	16 Mbps	10BaseT Ethernet
4	20 Mbps	Token Ring (Rarely used)
5	100 Mbps (2 pair) 1000 Mbps (4 pair)	100BaseT Ethernet Gigabit Ethernet
5e	1,000 Mbps	Gigabit Ethernet
6	10,000 Mbps	Gigabit Ethernet

Unshielded Twisted Pair Connector

The standard connector for unshielded twisted pair cabling is an RJ-45 connector. This is a plastic connector that looks like a large telephone-style connector (See fig. 2). A slot allows the RJ-45 to be inserted only one way. RJ stands for Registered Jack, implying that the connector follows a standard borrowed from the telephone industry. This standard designates which wire goes with each pin inside the connector.

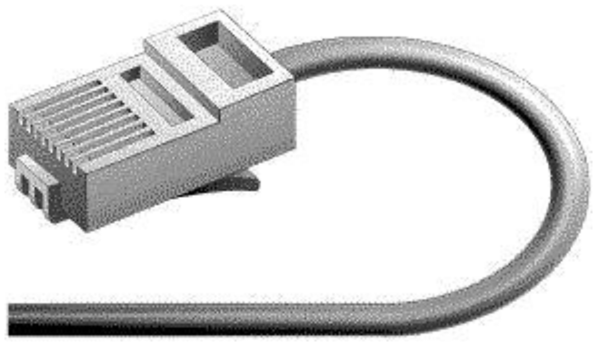


Fig. 2. RJ-45 connector

Shielded Twisted Pair (STP) Cable

Although UTP cable is the least expensive cable, it may be susceptible to radio and electrical frequency interference (it should not be too close to electric motors, fluorescent lights, etc.). If you must place cable in environments with lots of potential interference, or if you must place cable in extremely sensitive environments that may be susceptible to the electrical current in the UTP, shielded twisted pair may be the solution. Shielded cables can also help to extend the maximum distance of the cables.

Shielded twisted pair cable is available in three different configurations:

1. Each pair of wires is individually shielded with foil.
2. There is a foil or braid shield inside the jacket covering all wires (as a group).
3. There is a shield around each individual pair, as well as around the entire group of wires (referred to as double shield twisted pair).

Coaxial Cable

Coaxial cabling has a single copper conductor at its center. A plastic layer provides insulation between the center conductor and a braided metal shield (See fig. 3). The metal shield helps to block any outside interference from

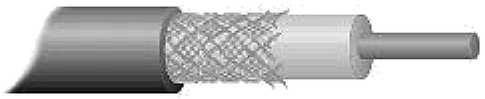


Fig. 3. Coaxial cable

Although coaxial cabling is difficult to install, it is highly resistant to signal interference. In addition, it can support greater cable lengths between network devices than twisted pair cable. The two types of coaxial

Thin coaxial cable is also referred to as thinnet. 10Base2 refers to the specifications for thin coaxial cable carrying Ethernet signals. The 2 refers to the approximate maximum segment length being 200 meters. In actual fact the maximum segment length is 185 meters. Thin coaxial cable has been popular in

Thick coaxial cable is also referred to as thicknet. 10Base5 refers to the specifications for thick coaxial cable carrying Ethernet signals. The 5 refers to the maximum segment length being 500 meters. Thick coaxial cable has an extra protective plastic cover that helps keep moisture away from the center conductor. This makes thick coaxial a great choice when running longer lengths in a linear bus network. One disadvantage of thick coaxial is that it does

Coaxial Cable Connectors

The most common type of connector used with coaxial cables is the Bayone-Neill-Concelman (BNC) connector (See fig. 4). Different types of adapters are available for

BNC connectors, including a T-connector, barrel connector, and terminator. Connectors on the cable are the weakest points in any network. To help avoid problems with your network, always use the BNC connectors that crimp, rather

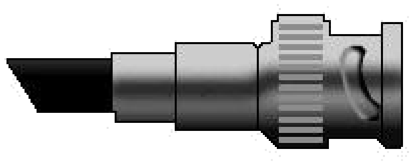


Fig. 4. BNC connector

Fiber Optic Cable

Fiber optic cabling consists of a center glass core surrounded by several layers of protective materials (See fig. 5). It transmits light rather than electronic signals eliminating the problem of electrical interference. This makes it ideal for certain environments that contain a large amount of electrical interference. It has also made it thestandard for connecting networks between

Fiber optic cable has the ability to transmit signals over much longer distances than coaxial and twisted pair. It also has the capability to carry information at vastly greater speeds. This capacity broadens communication possibilities to include services such as video conferencing and interactive services. The cost of fiber optic cabling is comparable to copper cabling; however, it is

The center core of fiber cables is made from glass or plastic fibers (see fig 5). A plastic coating then cushions the fiber center, and kevlar fibers help to strengthen the cables and prevent breakage. The outer insulating jacket made of teflon or PVC.

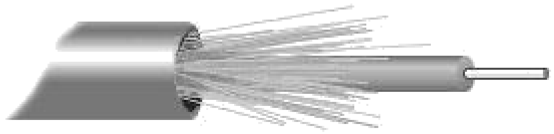


Fig. 5. Fiber optic cable

There are two common types of fiber cables -- single mode and multimode. Multimode cable has a larger diameter; however, both cables provide high bandwidth at high speeds. Single mode can provide more distance, but it is more expensive.

NETWORK DEVICES

Computer networking devices are units that mediate data in a computer network. A lst of computer networking devices follows:

Gateway:

A device sitting at a network node for interfacing with another network that uses different protocols. It may contain devices such as protocol translators, impedance matching devices, rate converters, fault isolators, or signal translators as necessary to provide system interoperability. It also requires the establishment of mutually acceptable administrative procedures between the two networks.

A protocol translation/mapping gateway interconnects networks with different network protocol technologies by performing the required protocol conversions.

Gateways, also called protocol converters, can operate at any layer of the OSI model. The job of a gateway is much more complex than that of a router or switch. Typically, a gateway must convert one protocol stack into another. Routers exemplify special cases of gateways.

Router

A router is a computer networking device that forwards data packets across a network toward their destinations, through a process known as routing. Routing occurs at layer 3 (the Network layer e.g. IP) of the OSI seven-layer protocol stack.

A router acts as a junction between two or more networks to transfer data packets among them. A router is different from a switch. A switch connects devices to form a Local area network (LAN).

One easy illustration for the different functions of routers and switches is to think of switches as neighborhood streets, and the router as the intersections with the street signs. Each house on the street has an address within a range on the block. In the same way, a switch connects various devices each with their own IP address(es) on a LAN.

However, the switch knows nothing about IP addresses except its own management address. Routers connect networks together the way that on-ramps or major intersections connect streets to both highways and freeways, etc. The street signs at the intersection (routing table) show which way the packets need to flow.

So for example, a router at home connects the Internet Service Provider's (ISP) network (usually on an Internet address) together with the LAN in the home (typically using a range of private IP addresses) and a single broadcast domain. The switch connects devices together to form the LAN. Sometimes the switch and the router are combined together in one single package sold as a multiple port router.

In order to route packets, a router communicates with other routers using routing protocols and using this information creates and maintains a routing table. The routing table stores the best routes to certain network destinations, the "routing metrics" associated with those routes, and the path to the next hop router. See the routing article for a more detailed discussion of how this works.

Routing is most commonly associated with the Internet Protocol, although other less-popular routed protocols are in use.

Bridge

A network bridge connects multiple network segments at the data link layer. It is sometimes called a network switch, and it works by using bridging. Traffic from one network is forwarded through it to another network. The bridge simply does what its name entails, by connecting two sides from adjacent networks.

A repeater is a similar device that connects network segments at the physical layer. An Ethernet hub is a type of repeater.

Bridging takes place at the data link layer of the OSI model. Therefore a bridge can only read the Ethernet header which provides the MAC address of the source and destination address. When a broadcast packet is transmitted, the bridge floods all the ports with the broadcast packets. Bridges use two methods to resolve the network segment that a MAC address belongs to.

Transparent Bridging – This method uses a forwarding database to send packets across network segments. The forwarding database is initially empty and entries in the database are built as the bridge receives packets. If an address entry is not found in the forwarding table, the packet is flooded to all ports of the bridge which sends the packet to all segments except the source address. This type of bridging is common in Ethernet networks. To avoid frame looping, a spanning tree is created from the network graph and bridges not present in it are kept inactive; they can become active again if another bridge stops working.

Source route bridging – This method is used in Token Ring networks..

In Ethernets, the term "bridge" formally means a device that behaves according to the IEEE 802.1D standard - this is most often referred to as a network switch in marketing literature.

Switch

A switch is a device that allocates traffic from one network segment to certain lines which connect the segment to another network segment. So unlike a hub a switch splits the network traffic and sends it to different destinations rather than to all systems on the network.

It performs transparent bridging (connection of multiple network segments with forwarding based on MAC addresses) at full wire speed in hardware. The use of specially designed hardware also makes it possible to have large numbers of ports (unlike a PC based bridge which is very limited by expansion slot count).

If a network has only switches and no hubs then the collision domains are either reduced to a single link or, if both ends support full duplex, eliminated altogether. The principle of a fast hardware forwarding device with many ports can be extended to higher layers giving the multilayer switch.

A switch can connect Ethernet, Token Ring, Fibre Channel or other types of packet switched network segments together to form a heterogeneous network operating at OSI Layer 2 (though there may be complications caused by the different MTUs of the standards).

As a frame comes into a switch, the switch saves the originating MAC address and the originating (hardware) port in the switch's MAC address table. This table often uses content-addressable memory, so it is sometimes called the "CAM table". The switch then selectively transmits the frame from specific ports based on the frame's destination MAC address and previous entries in the MAC address table. If the destination MAC address is unknown, for instance, a broadcast address or (for simpler switches) a multicast address, the switch simply transmits the frame out of all of the connected interfaces except the incoming port. If the destination MAC address is known, the frame is forwarded only to the corresponding port in the MAC address table. If the destination port is the same as the originating port, the frame is filtered out and not forwarded.

Switches, unlike hubs, use micro segmentation to create collision domains, one per connected segment. This way, only the NICs which are directly connected via a point-to-point link, or directly connected hubs are contending for the medium. If the switch and the equipment (other than a hub) it connects to support full-duplex then the collision domain is eliminated entirely.

The higher level operation also allows some more advanced features that would be impractical with simple hubs. For example Virtual LANs can be used in switches to reduce the size of the broadcast domains and at the same time increase security and switches can also implement spanning tree protocol allowing use of redundant links.

Hub

An Ethernet hub or concentrator connects multiple Ethernet segments together making them act as a single segment. When using a hub, every attached device shares the same broadcast domain and the same collision domain. Therefore, only one computer connected to the hub is able to transmit at a time. Depending on the network topology, the hub provides a basic level 1 OSI model connection among the network objects (workstations, servers, etc). It provides bandwidth which is shared among all the objects, compared to switches, which provide a dedicated connection between individual nodes..

It is a device for connecting multiple twisted pair or fibre optic Ethernet devices together, making them act as a single segment. It works at the physical layer of the OSI model, repeating the signal received at one port out each of the other ports (but not the original one). The device is thus a form of multiport repeater. Ethernet hubs are also responsible for forwarding a jam signal to all ports if it detects a collision.

A hubbed Ethernet network behaves like a shared-medium, that is only one device can successfully transmit at a time and each host remains responsible for collision detection and retransmission. With 10BASE-T and 100BASE-T links (which generally account for most or all of the ports on a hub) there are separate pairs for transmit and receive but they are used in half duplex mode in which they still effectively behave like shared medium links

The need for hosts to be able to detect collisions limits the number of hubs and the total size of the network. For 10 Mbit/s networks, up to 5 segments (4 hubs) are allowed between any two end stations. For 100 Mbit/s networks, the limit is reduced to 3 segments (2 hubs) between any two end stations, and even that is only allowed if the hubsare of the low delay variety. Some hubs have special (and generally manufacturer specific) stack ports allowing them to be combined in a way that allows more hubs than simple chaining through Ethernet cables, but even so a large Fast Ethernet network is likely to require switches to avoid the chaining limits of hubs.

Most hubs detect typical problems, such as excessive collisions on individual ports, and *partition* the port, disconnecting it from the shared medium. Thus, hub-based Ethernet is generally more robust than coaxial cable-based Ethernet, where a misbehaving device can disable the entire segment. Even if not partitioned automatically, a hub makes troubleshooting easier because status lights can indicate the possible problem source or, as a last resort, devices can be disconnected from a hub one at a time much more easily than a coaxial cable. They also remove the need to troubleshoot faults on a huge cable with multiple taps.

Repeater

A repeater is an electronic device that receives a weak or low-level signal and retransmits it at a higher level or higher power, so that the signal can cover longer distances without degradation.

In telecommunication, the term **repeater** has the following standardized meanings:

- An analog device that amplifies an input signal regardless of its nature (analog or digital).
- A digital device that amplifies, reshapes, retimes, or performs a combination of any of these functions on a digital input signal for retransmission.

Repeaters are often used in trans-continental and trans-oceanic cables, because the attenuation (signal loss) over such distances would be completely unacceptable without them. Repeaters are used in both copper-wire cables carrying electrical signals, and in fibre optics carrying light.

In optical communications the term **repeater** is used to describe a piece of equipmentthat receives an optical signal, converts that signal into an electrical one, regenerates it, and then retransmits an optical signal. Since such a device converts the optical signal intoan electrical one, and then back to an optical signal, they are often known as Optical-Electrical-Optical (OEO) repeaters.

NETWORK TOPOLOGY

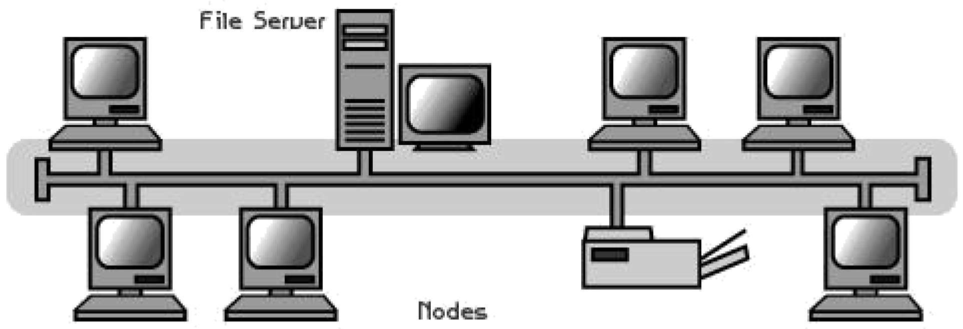
The study of network topology recognizes seven basic topologies: [3]

- Point-to-point topology
- Bus (point-to-multipoint) topology
- Star topology
- Ring topology
- Tree topology
- Mesh topology
- Hybrid topology

This classification is based on the interconnection between computers — be it physical or logical.

The physical topology of a network is determined by the capabilities of the network access devices and media, the level of control or fault tolerance desired, and the cost associated with cabling or telecommunications circuits.

Bus



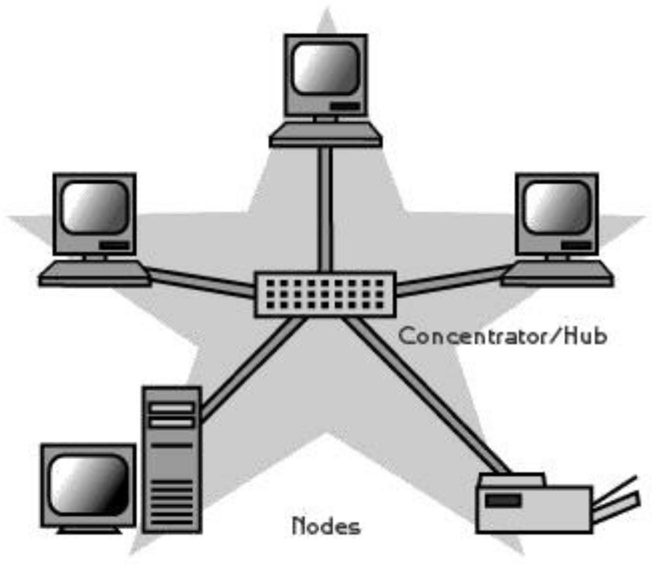
Bus network topology

In local area networks where bus topology is used, each machine is connected to a single cable. Each computer or server is connected to the single bus cable through some kind of connector. A terminator is required at each end of the bus cable to prevent the signal from bouncing back and forth on the bus cable. A signal from the source travels in both directions to all machines connected on the bus cable until it finds the MAC address or IP address on the network that is the intended recipient. If the machine address does not match the intended address for the data, the machine ignores the data. Alternatively, if the data does match the machine address, the data is accepted. Since the bus topology consists of only one wire, it is rather inexpensive to implement when compared to other topologies. However, the low cost of implementing the technology is offset by the high cost of managing the network. Additionally, since only one cable is utilized, it can be the single point of failure. If the network cable breaks, the entire network will be down.

Star

A star topology is designed with each node (file server, workstations, and peripherals) connected directly to a central network hub, switch, or concentrator (See fig. 2).

Data on a star network passes through the hub, switch, or concentrator before continuing to its destination. The hub, switch, or concentrator manages and controls all functions of the network. It also acts as a repeater for the data flow. This configuration is common with twisted pair cable; however, it can also be used with coaxial cable or fiber optic cable.



Advantages of a Star Topology

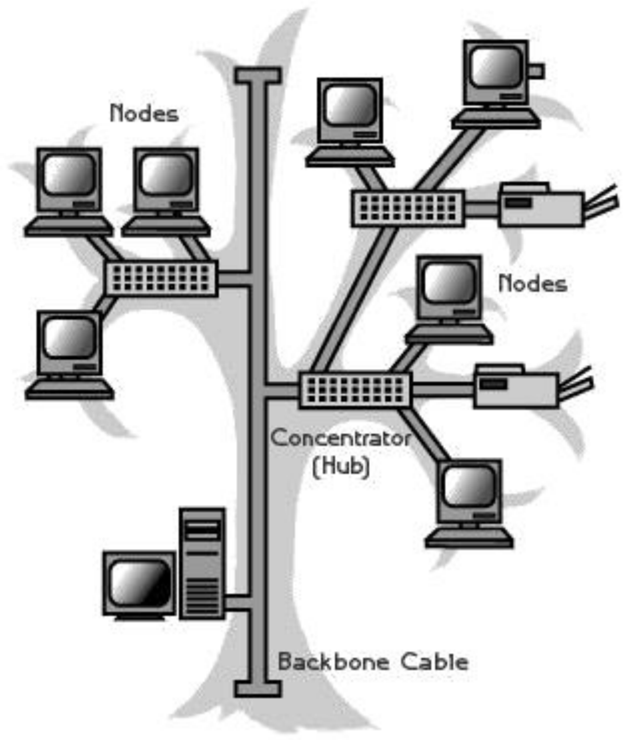
- Easy to install and wire.
- No disruptions to the network when connecting or removing devices.
- Easy to detect faults and to remove parts.

Disadvantages of a Star Topology

- Requires more cable length than a linear topology.
- If the hub, switch, or concentrator fails, nodes attached are disabled.
- More expensive than linear bus topologies because of the cost of the hubs, etc.

Tree or Expanded Star

A tree topology combines characteristics of linear bus and star topologies. It consists of groups of star-configured workstations connected to a linear bus backbone cable (See fig.). Tree topologies allow for the expansion of an existing network, and enable schools to configure a network to meet their needs.



Advantages of a Tree Topology

- Point-to-point wiring for individual segments.
- Supported by several hardware and software venders.

Disadvantages of a Tree Topology

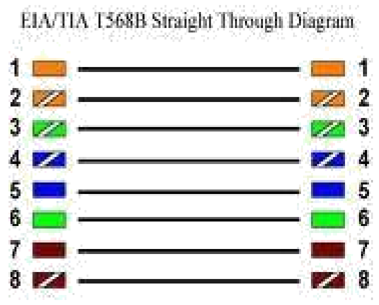
- Overall length of each segment is limited by the type of cabling used.
- If the backbone line breaks, the entire segment goes down.
- More difficult to configure and wire than other topologies.

PROCEDURE

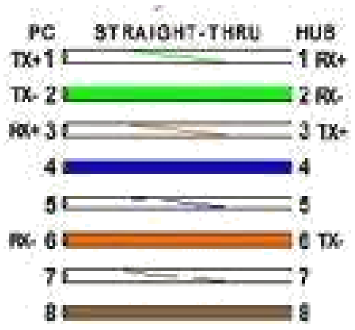
There are generally three main types of networking cables: straight-through, crossover, and rollover cables. Each cable type has a distinct use, and should not be used in place of another. So how do you know which cable to use for what you need?

The Purpose of Straight-Through Cables

Straight-through cables get their name from how they are made. Out of the 8 pins that exist on both ends of an Ethernet cable, each pin connects to the same pin on the opposite side. Review the diagram below for a visual example:



OR



Notice how each wire corresponds to the same pin. This kind of wiring diagram is part of the 568A standard. The 568B standard achieves the same thing, but through different wiring. It is generally accepted to use the 568A standard as pictured, since it allows compatibility with certain telephone hardware- while 568**B doesn't.**

Straight-through cables are primarily used for connecting unlike devices. A straight-through cable is typically used in the following situations:

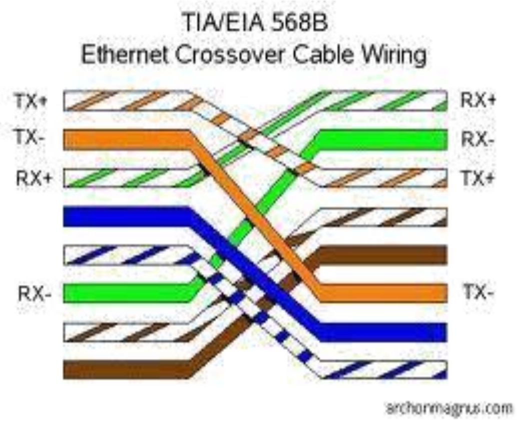
Use a straight-through cable when:

- Connecting a router to a hub
- Connecting a computer to a swtich
- Connecting a LAN port to a switch, hub, or computer

Note that some devices such as routers will have advanced circuitry, which enables them to use both crossover and straight-through cables. In general, however, straight-through cables will not connect a co**mputer and router because they are not "unlike devices."**

The purpose of Crossover Cables

Crossover cables are very similar to straight-through cables, except that they have pairs of wires that crisscross. This allows for two devices to communicate at the same time. Unlike straight-through cables, we use crossover cables to connect like devices. A visual examplecan be seen below:



Notice how all we did was switch the orange-white and green-white wires, and then the orange and green wires. This will enable like devices to communicate. Crossover cables are typically used in the following situations:

Use a crossover cable when:

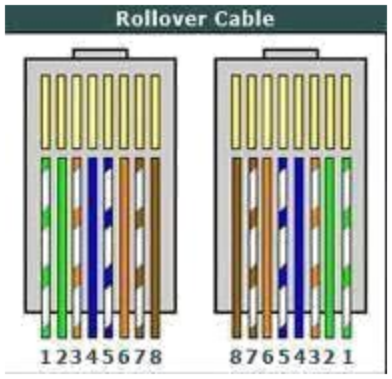
- Connecting a computer to a router
- Connecting a computer to a computer
- Connecting a router to a router
- Connecting a switch to a switch
- Connecting a hub to a hub

While the rule of thumb is to use crossover cables with like devices, some devices do not follow standards. Others provide support for both types of cables. However, there is still something that both crossover and straight-**through cables can't do.**

The Purpose of Rollover Cables

Rollover cables, like other cabling types, got their name from how they are wired. Rollover cables essentially have one end of the cable wired exactly opposite from the other. This

essentially “rolls over” the wires- but why would we need to do such a thing? Rollover **cables, also called Yost cables, usually connect a device to a router or switch’s console port.** This allows a programmer to make a connection to the router or switch, and program it as needed. A visual example can be seen below:



Notice that each wire is simply “rolled over.” These types of cables are generally not used very much, so are usually colored differently from other types of cables.

APPLICATION

1. Most networks in the real world use the above-mentioned devices to create networks.

FAQS

Please refer the above theory for answers

1. What are routers, hubs, switches and bridges?
2. What is the difference between them
3. Explain the learning bridge
4. On which layers do these devices work on?
5. In order to increase bandwidth per node which device will you use? A hub or a switch?

Experiment Number: A2

Title: Study of different types of topology and Transmission Media

OBJECTIVES:

- To understand different types of topology
- To understand Transmission Media

PROBLEM STATEMENT:

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

TYPES OF CABLES

Cable is the medium through which information usually moves from one network device to another. There are several types of cable which are commonly used with LANs. In some cases, a network will utilize only one type of cable, other networks will use a variety of cable types. The type of cable chosen for a network is related to the network's topology, protocol, and size. Understanding the characteristics of different types of cable and how they relate to other aspects of a network is necessary for the development of a successful network.

The following sections discuss the types of cables used in networks and other related topics.

- Unshielded Twisted Pair (UTP) Cable
- Shielded Twisted Pair (STP) Cable
- Coaxial Cable
- Fiber Optic Cable

NETWORK TOPOLOGY

The study of network topology recognizes seven basic topologies: [3]

- Point-to-point topology
- Bus (point-to-multipoint) topology
- Star topology
- Ring topology
- Tree topology
- Mesh topology
- Hybrid topology

APPLICATION

1. Most networks in the real world use the above-mentioned devices to create networks.

FAQS

Please refer the above theory for answers

1. Explain different types of tologolgy?
2. Differentiate between types of topology?
3. Explain Transmission Media?
4. Differentiate between them?

Experiment Number: A3

Lab Assignment on Unit II: (Use C/C++)

Title: Study of CRC and Hamming Code

OBJECTIVES:

- To understand the structure and working of CRC and Hamming Code

PROBLEM STATEMENT

Write a program for error detection and correction for 7/8 bits ASCII codes using Hamming Codes or CRC. Demonstrate the packets captured traces using Wireshark Packet Analyzer Tool for peer to peer mode.(50% students will perform Hamming Code and others will perform CRC)

Theory: -

1. *Introduction*

All communication systems try to make sure that the transmitted messages reach the destination without any problem. So they intend to implement different algorithms in order to satisfy this requirement. One of the options is to encode the message in a way that enables the receiver to check the corrupted messages; example of such coding is the CRC.
Data storage devices must also prevent any corruption of its data. The best choice for this problem is to use data redundancy, which costs a lot. So instead of that, data can be encoded using redundant codes in order to detect the corrupted data and the CRC is the one of the most common codes that is used in such cases.

2. *What is CRC*

CRC stands for Cyclic Redundancy Check. Which means that is based on cyclic algorithm that generates redundant information.
The CRC performs a mathematical calculation on a block of data and returns information (number) about the contents and organization of that data. So the resultant number uniquely identifies that block of data. This unique number can be used to check the validity of data or to compare two blocks. So this approach is used in many communication and computer systems to ensure the validity of the transmitted or stored data.

In general CRC codes are able to detect:

- All single- and double-bit errors.
- All odd numbers of errors.
- All burst errors less than or equal to the degree of the polynomial used.
- Most burst errors greater than the degree of the polynomial used.

3. Check sum concept

One approach in of error checking is to append the sum value of all message bytes to the end of the message. This sum can identify the message and changes in its contents. On the other hand, if there is more than one change one that adds up a value and one subtracts one in a way that the sum remains the same, so it cannot be used to detect errors. The same can happen if the check sum is changed with the same value as the message.

4. CRC idea

The main idea of CRC is to treat the message as binary numbers, and divide it by fixed binary number. The remainder from this division is considered the checksum. The recipient of the message performs the same division and compare the remainder with the "checksum" (transmitted remainder).

5. Theory of operation:

As stated in the previous section, the CRC is a simple binary division and subtraction. The only difference is that these operations are done on modulo arithmetic based on mod 2. For example the addition and subtraction are replaced with XOR operation that do the sum and subtraction without carry.

Polynomial concept

The CRC algorithm uses the term polynomial to perform all of its calculations. This polynomial is the same concept as the traditional arithmetic polynomials. The divisor, dividend, quotient, and remainder that are represented by numbers are represented as polynomials with binary coefficients. For example the number 23 (10111b) can be represented in the polynomial form as:

$1*x^4 + 0*x^3 + 1*x^2 + 1*x^1 + 1*x^0$
or
$x^4 + x^2 + x^1 + x^0$

Note the binary representation of the number (10111).
This representation simplifies the traditional arithmetic operations (addition, multiplication, etc…) that are all done on normal algebraic polynomials.

If we can assume that X is 2, then the operations are simplified more and some because some terms can be canceled. For example the term $3*x^3$ is represented as 24 in normal number representation and 24 = 16+8 which is x^4+x^3 in polynomial representation.

Generator polynomial:

In order to do the CRC calculation; a divisor must be selected which can be any one. This divisor is called the **generator polynomial**. Even though, some polynomials became standard for many applications. Polynomial selection is behind the scope of this summary.
One of the most used terms in CRC is the width of the polynomial. This width is represented by the order of the highest power in the polynomial. The width of the polynomial in the previous example is 4, which has 5 bits in its binary representation.
Since CRC is used to detect errors, a suitable generator polynomial must be selected for each application. This is because each polynomial has different error detection capabilities.

CRC algorithms are commonly called after the generator polynomial width, for example CRC-16 uses a generator polynomial of width 15 and 16-bit register and CRC-32 uses polynomial width of 31 and 32-bit register.

Common used polynomials:

No. of bits	**Polynomial**	**Name**
16 bits:	(16,12,5,0)	X25 standard
	(16,15,2,0)	CRC-16/ CCITT
32 bits:	(32,26,23,22,16,12,11,10,8,7,5,4,2,1,0)	Ethernet, ATM, CRC-32

To Summarize:
The data D is multiplied by $\mathbf{X^n}$ and divided by the generator polynomial **G**, the quotient **Q** is discarded and the Remainder **R** is considered the check sum. On the other side, the data stream **D** is multiplied again by $\mathbf{X^n}$ and the check sum (CRC) **R** is added to it (normally it come with the stream) and the whole result is divided by **G** again. The result now should be zero for valid data. This operation can be described by the following equation:

$$\mathbf{(X^n*D)+R = (Q*G)+0}$$

6. How Do CRC works

Transmitter calculation
The transmitter can append zeros to the end of the message (LSB), perform the division and find the remainder and appended it to the original message.

Receiver calculation
The message receiver can do one of the followings:
Separate the message and checksum. Calculate the checksum for the message (after appending zeros) and compare the two checksums.
Checksum the whole message including the CRC (without appending zeros) and check if the new CRC comes out as Zero.

7. Implementation:

CRC has two main implementation techniques:

- Straightforward
- Look-up table based

Straightforward:
This approach is a direct mapping for the CRC algorithm.
In fact this approach does not use standard microprocessor divide instruction because 1. We need xor based division (no carry in addition or subtraction) 2. The dividend (the message) can be very large and behind the processor support.
This approach is relatively low speed and consumes very small resources

This implementation is described in PAINLESS GUIDE TO CRC as:

```
To perform the division, do the following:

  Load the register with zero bits.
  Augment the message by appending W zero bits to the end of it.
  While (more message bits)
    Begin
    Shift the register left by one bit, reading the next bit of the
      augmented message into register bit position 0.
    If (a 1 bit popped out of the register during step 3)
      Register = Register XOR Poly.
    End
  The register now contains the remainder.
```

Note: The register holds the remainder only after the last bit of the message gets out of it.

This algorithm is done by the long division

Dout	Register Bit	$\oplus$	Comment
0	0	0	Pass Register bit
0	1	1	
1	0	1	Xor with Poly
1	1	0	

This approach can be implemented directly using Linear-feedback shift registers (LFSR) where the division is performed by left shifting and subtraction by XOR.

Parallel Implementation:

In real world the serial approach (bit-by-bit) calculation is not acceptable for many applications that requires high performance or the smallest processing word size is more than a bit For that reason parallel implementation is needed.

Parallel implementation can be done by entering the data word to the CRC and perform the serial shift and xoring the bits in parallel. Simply it calculates all xor combinations as if a single bit shifting is done on the data and CRC register.

This implementation is best described in the cypress CRC document

Look-up table based:

This approach is based on pre-calculating the CRC for all input combinations.

This approach is best described in “A PAINLESS GUIDE TO CRC ERROR DETECTION ALGORITHMS”

FAQS

Please refer the above theory for answers:

1. What is CRC code?
2. What is the difference between CRC and Hamming Code?
3. How do CRC works?
4. Polynomial concept

Conclusion:

Hence we studied and implement program for error detection and correction for 7/8 bits ASCII codes using CRC.

Experiment Number: A4

Lab Assignment on Unit II: (Use Java/Python)

Title: Study of Go Back-N and Selective Repeat mode of Sliding window protocol.

OBJECTIVES:

1. To Understand Go back n and selective repeat mode of sliding window protocol.
2. Designing simple client or server applications for stream.

PROBLEM STATEMENT

Write a program to simulate Go back N and Selective Repeat Modes of Sliding Window Protocol in peer to peer mode and demonstrate the packets captured traces using Wireshark Packet Analyzer Tool for peer to peer mode.

Theory: -

Data-link layer is responsible for implementation of point-to-point flow and error control mechanism.

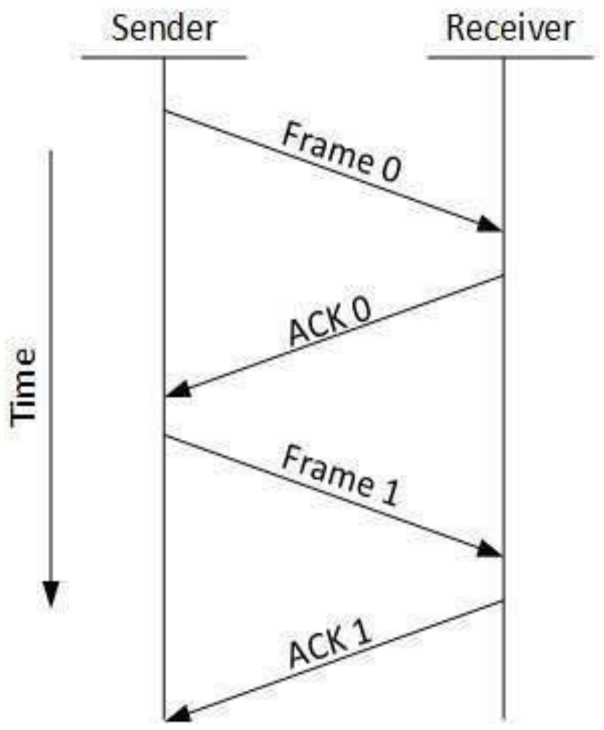
Flow Control :-

When a data frame (Layer-2 data) is sent from one host to another over a single medium, it is required that the sender and receiver should work at the same speed. That is, sender sends at a speed on which the receiver can process and accept the data. What if the speed (hardware/software) of the sender or receiver differs? If sender is sending too fast the receiver may be overloaded, (swamped) and data may be lost.

Two types of mechanisms can be deployed to control the flow:

- **Stop and Wait**

  This flow control mechanism forces the sender after transmitting a data frame to stop and wait until the acknowledgement of the data-frame sent is received.



-

- **Sliding Window**

  In this flow control mechanism, both sender and receiver agree on the number of data-frames after which the acknowledgement should be sent. As we learnt, stop and wait flow control mechanism wastes resources, this protocol tries to make use of underlying resources as much as possible.

Error Control : -

When data-frame is transmitted, there is a probability that data-frame may be lost in the transit or it is received corrupted. In both cases, the receiver does not receive the correct data-frame and sender does not know anything about any loss.In such case, both sender and receiver are equipped with some protocols which helps them to detect transit errors such as loss of data-frame. Hence, either the sender retransmits the data-frame or the receiver may request to resend the previous data-frame.

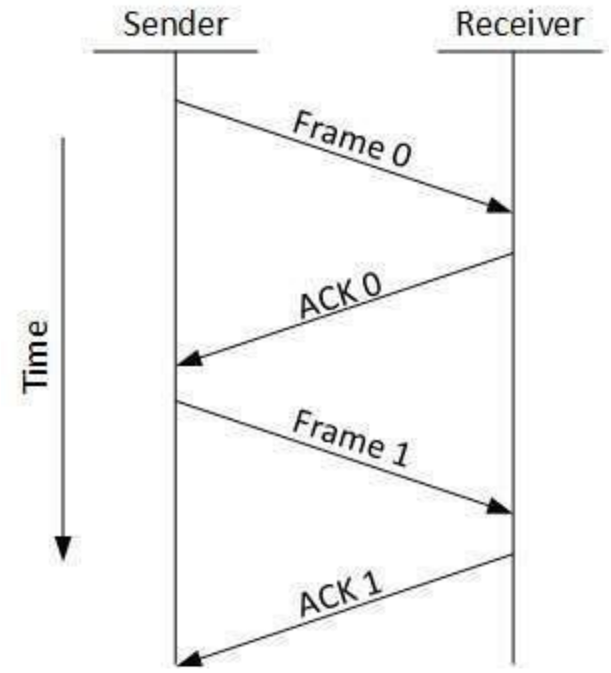
Requirements for error control mechanism:

- **Error detection** - The sender and receiver, either both or any, must ascertain that there is some error in the transit.
- **Positive ACK** - When the receiver receives a correct frame, it should acknowledge it.
- **Negative ACK** - When the receiver receives a damaged frame or a duplicate frame, it sends a NACK back to the sender and the sender must retransmit the correct frame.

- **Retransmission:** The sender maintains a clock and sets a timeout period. If an acknowledgement of a data-frame previously transmitted does not arrive before the timeout the sender retransmits the frame, thinking that the frame or it’s acknowledgement is lost in transit.

There are three types of techniques available which Data-link layer may deploy to control the errors by Automatic Repeat Requests (ARQ):

- **Stop-and-wait ARQ**

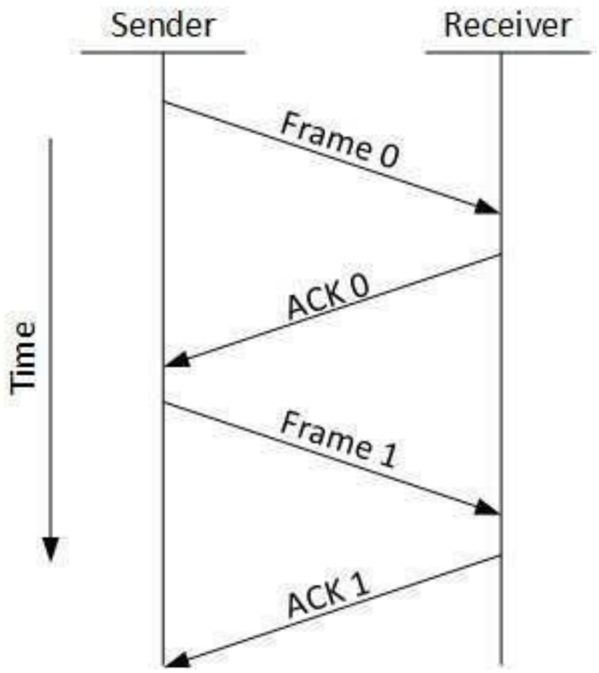


The following transition may occur in Stop-and-Wait ARQ:

  - The sender maintains a timeout counter.
  - When a frame is sent, the sender starts the timeout counter.
  - If acknowledgement of frame comes in time, the sender transmits the next frame in queue.
  - If acknowledgement does not come in time, the sender assumes that either the frame or its acknowledgement is lost in transit. Sender retransmits the frame and starts the timeout counter.
  - If a negative acknowledgement is received, the sender retransmits the frame.

- **Go-Back-N ARQ**

  Stop and wait ARQ mechanism does not utilize the resources at their best.When the acknowledgement is received, the sender sits idle and does nothing. In Go-Back-N ARQ method, both sender and receiver maintain a window.

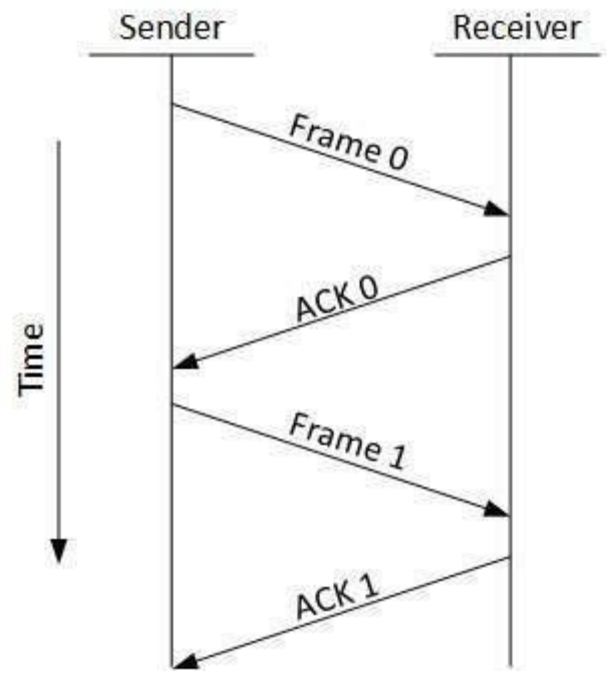


  The sending-window size enables the sender to send multiple frames without receiving the acknowledgement of the previous ones. The receiving-window enables the receiver to receive multiple frames and acknowledge them. The receiver keeps track of incoming frame's sequence number.

  When the sender sends all the frames in window, it checks up to what sequence number it has received positive acknowledgement. If all frames are positively acknowledged, the sender sends next set of frames. If sender finds that it has received NACK or has not receive any ACK for a particular frame, it retransmits all the frames after which it does not receive any positive ACK.

- **Selective Repeat ARQ**

  In Go-back-N ARQ, it is assumed that the receiver does not have any buffer space for its window size and has to process each frame as it comes. This enforces the sender to retransmit all the frames which are not acknowledged.



In Selective-Repeat ARQ, the receiver while keeping track of sequence numbers, buffers the frames in memory and sends NACK for only frame which is missing or damaged.

The sender in this case, sends only packet for which NACK is received.

FAQS

Please refer the above theory for answers:

1. What is Error and flow control?
2. Explain Stop and wait protocol?
3. Explain go back n and Selective repeat protocol in details?
4. Difference between Go-back n and Selective repeat.

Conclusion:

Hence we studied and implement program Go-back n and Selective repeat modes of sliding window protocol

Experiment Number: B1

Lab Assignment on Unit III: (Use JAVA/PYTHON)

Title: Study of IP addresses and Subnetting.

OBJECTIVES:

1) To understand the structure of IP addresses and subnet mask.
2) To understand the concept of Subnetting and create subnet of given IP address.

PROBLEM STATEMENT

Write a program to demonstrate subletting and find the subnet masks.

Theory: -

Introduction: -

If definitions are helpful to you, use these vocabulary terms in order to get you started:

- Address - The unique number ID assigned to one host or interface in a network.
- Subnet - A portion of a network that shares a particular subnet address.
- Subnet mask - A 32-bit combination used to describe which portion of an address refers to the subnet and which part refers to the host.
- Interface - A network connection.

If you have already received your legitimate address (es) from the Internet Network Information Center (InterNIC), you are ready to begin. If you do not plan to connect to the Internet, Cisco strongly suggests that you use reserved addresses from RFC 1918

Understand IP Addresses

An IP address is an address used in order to uniquely identify a device on an IP network. The address is made up of 32 binary bits, which can be divisible into a network portion and host portion with the help of a subnet mask. The 32 binary bits are broken into four octets (1 octet = 8 bits).

Each octet is converted to decimal and separated by a period (dot). For this reason, an IP address is said to be expressed in dotted decimal format (for example, 172.16.81.100). The value in each octet ranges from 0 to 255 decimal, or 00000000 - 11111111 binary.

Here is how binary octets convert to decimal: The right most bit, or least significant bit, of an octet holds a value of 20. The bit just to the left of that holds a value of 21. This continues until the left-most bit, or most significant bit, which holds a value of 27. So if all binary bits are a one, the decimal equivalent would be 255 as shown here:

```
  1    1  1  1   1 1 1 1
128 64 32 16 8 4 2 1 (128+64+32+16+8+4+2+1=255)
```

Here is a sample octet conversion when not all of the bits are set to 1.

```
0 1 0 0 0 0 0 1
0 64 0 0 0 0 0 1 (0+64+0+0+0+0+0+1=65)
```

And this sample shows an IP address represented in both binary and decimal.

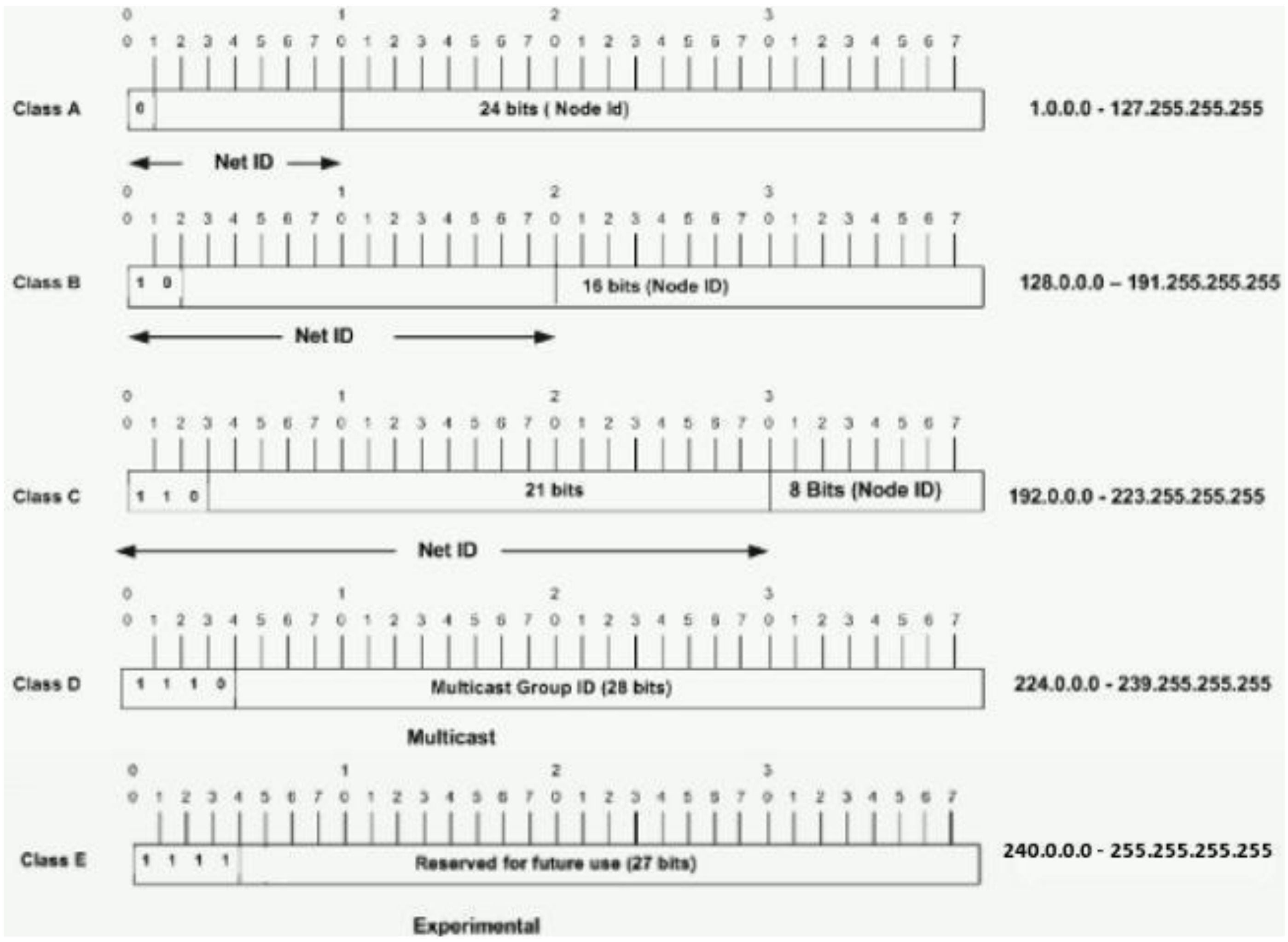
```
10.                  1.         23.         19 (decimal)
00001010.00000001.00010111.00010011 (binary)
```

These octets are broken down to provide an addressing scheme that can accommodate large and small networks. There are five different classes of networks, A to E. This document focuses on classes A to C, since classes D and E are reserved and discussion of them is beyond the scope of this document.

Note: Also note that the terms "Class A, Class B" and so on are used in this document in order to help facilitate the understanding of IP addressing and subnetting. These terms are rarely used in the industry anymore because of the introduction of classless interdomain routing (CIDR).

Given an IP address, its class can be determined from the three high-order bits (the three left-most bits in the first octet). Figure 1 shows the significance in the three high order bits and the range of addresses that fall into each class. For informational purposes, Class D and Class E addresses
are also shown.

Figure 1



In a Class A address, the first octet is the network portion, so the Class A example in Figure 1 has a major network address of 1.0.0.0 - 127.255.255.255. Octets 2, 3, and 4 (the next 24 bits) are for the network manager to divide into subnets and hosts as he/she sees fit. Class A addresses are used for networks that have more than 65,536 hosts (actually, up to 16777214 hosts!).

In a Class B address, the first two octets are the network portion, so the Class B example in Figure 1 has a major network address of 128.0.0.0 - 191.255.255.255. Octets 3 and 4 (16 bits) are for local subnets and hosts. Class B addresses are used for networks that have between 256 and 65534 hosts.

In a Class C address, the first three octets are the network portion. The Class C example in Figure 1 has a major network address of 192.0.0.0 - 223.255.255.255. Octet 4 (8 bits) is for local subnets and hosts - perfect for networks with less than 254 hosts.

Network Masks:

A network mask helps you know which portion of the address identifies the network and which portion of the address identifies the node. Class A, B, and C networks have default masks, also known as natural masks, as shown here:

Class A:
255.0.0.0

Class B: 255.255.0.0
Class C: 255.255.255.0

An IP address on a Class A network that has not been subnetted would have an address/mask pair similar to: 8.20.15.1 255.0.0.0. In order to see how the mask helps you identify the network and node parts of the address, convert the address and mask to binary numbers.

8.20.15.1 = 00001000.00010100.00001111.00000001
255.0.0.0 = 11111111.00000000.00000000.00000000

Once you have the address and the mask represented in binary, then identification of the network and host ID is easier. Any address bits which have corresponding mask bits set to 1 represent the network ID. Any address bits that have corresponding mask bits set to 0 represent the node ID.

```
8.20.15.1= 00001000.00010100.00001111.00000001

255.0.0.0 = 11111111.00000000.00000000.00000000
-----------------------------
  net id                         host id
```

netid = 00001000 = 8
hostid = 00010100.00001111.00000001 = 20.15.1

Understand Subnetting:-

Subnetting allows you to create multiple logical networks that exist within a single Class A, B, or C network. If you do not subnet, you are only able to use one network from your Class A, B, or C network, which is unrealistic.

Each data link on a network must have a unique network ID, with every node on that link being a member of the same network. If you break a major network (Class A, B, or C) into smaller subnetworks, it allows you to create a network of interconnecting subnetworks. Each data link on this network would then have a unique network/subnetwork ID. Any device, or gateway, that connects n networks/subnetworks has n distinct IP addresses, one for each network / subnetwork that it interconnects.

In order to subnet a network, extend the natural mask with some of the bits from the host ID portion of the address in order to create a subnetwork ID. For example, given a Class C network of which has a natural mask of 255.255.255.0, you can create subnets in this manner:

```
 11001100.00010001.00000101.00000000
255.255.255.224 - 11111111.11111111.11111111.11100000
.................................|sub|......
```

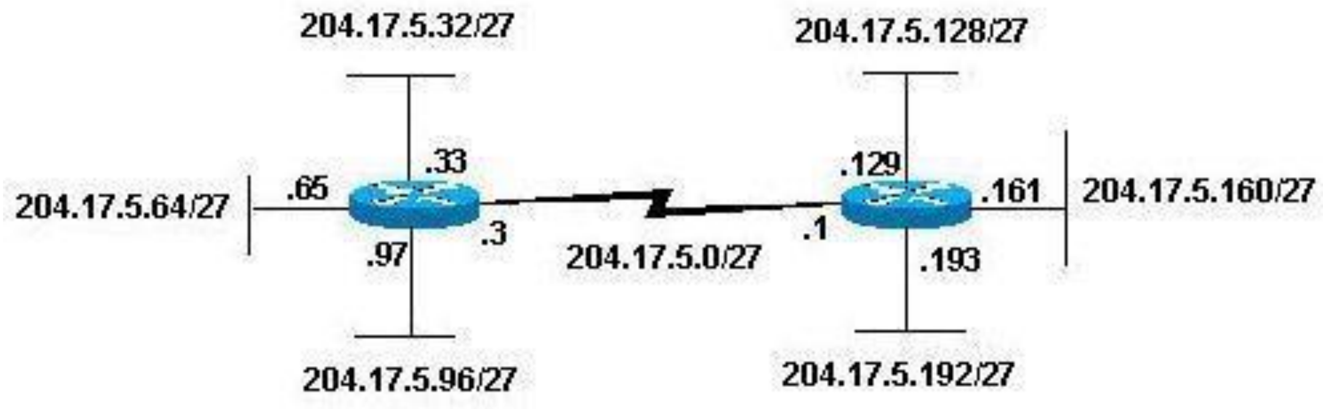
By extending the mask to be 255.255.255.224, you have taken three bits (indicated by "sub") from the original host portion of the address and used them to make subnets. With these three bits, it is possible to create eight subnets. With the remaining five host ID bits, each subnet can have up to 32 host addresses, 30 of which can actually be assigned to a device since host ids of all zeros or all ones are not allowed (it is very important to remember this). So, with this in mind, these subnets have been created.

204.17.5.0 255.255.255.224	host address range 1 to 30	
204.17.5.32 255.255.255.224	host address range 33 to 62	
204.17.5.64 255.255.255.224	host address range 65	to 94
204.17.5.96 255.255.255.224	host address range 97	to 126
204.17.5.128 255.255.255.224	host address range 129	to 158
204.17.5.160 255.255.255.224	host address range 161	to 190
204.17.5.192 255.255.255.224	host address range 193	to 222
204.17.5.224 255.255.255.224	host address range 225	to 254

Note: There are two ways to denote these masks. First, since you use three bits more than the "natural" Class C mask, you can denote these addresses as having a 3-bit subnet mask.
Or, secondly, the mask of 255.255.255.224 can also be denoted as /27 as there are 27 bits that are set in the mask. This second method is used with CIDR. With this method, one of these networks can be described with the notation prefix/length. For example, 204.17.5.32/27 denotes the network 204.17.5.32 255.255.255.224. When appropriate, the prefix/length notation is used to denote the mask throughout the rest of this document.

The network subnetting scheme in this section allows for eight subnets, and the network might appear as:

Figure 2



Notice that each of the routers in Figure 2 is attached to four subnetworks, one subnetwork is common to both routers. Also, each router has an IP address for each subnetwork to which it is attached. Each subnetwork could potentially support up to 30 host addresses.

This brings up an interesting point. The more host bits you use for a subnet mask, the more subnets you have available. However, the more subnets available, the less host addresses available per subnet. For example, a Class C network of 204.17.5.0 and a mask of 255.255.255.224 (/27) allows you to have eight subnets, each with 32 host addresses (30 of which could be assigned to devices). If you use a mask of 255.255.255.240 (/28), the break down is:

```
204.17.5.0 -          11001100.00010001.00000101.00000000
255.255.255.240 - 11111111.11111111.11111111.11110000
.....................................|sub |.....
```

Since you now have four bits to make subnets with, you only have four bits left for host addresses. So in this case you can have up to 16 subnets, each of which can have up to 16 host addresses (14 of which can be assigned to devices).

Take a look at how a Class B network might be subnetted. If you have network 172.16.0.0 ,then you know that its natural mask is 255.255.0.0 or 172.16.0.0/16. Extending the mask to anything beyond 255.255.0.0 means you are subnetting. You can quickly see that you have the ability to create a lot more subnets than with the Class C network. If you use a mask of 255.255.248.0 (/21), how many subnets and hosts per subnet does this allow for?

```
172.16.0.0    -  10101100.00010000.00000000.00000000
255.255.248.0 - 11111111.11111111.11111000.00000000
.........................| sub |..................
```

You use five bits from the original host bits for subnets. This allows you to have 32 subnets (25). After using the five bits for subnetting, you are left with 11 bits for host addresses. This allows each subnet so have 2048 host addresses (211), 2046 of which could be assigned to devices.

Note: In the past, there were limitations to the use of a subnet 0 (all subnet bits are set to zero) and all ones subnet (all subnet bits set to one). Some devices would not allow the use of these subnets. Cisco Systems devices allow the use of these subnets when the ip subnet zero command is configured.

Examples

Sample Exercise 1

Now that you have an understanding of subnetting, put this knowledge to use. In this example, you are given two address / mask combinations, written with the prefix/length notation, which have been assigned to two devices. Your task is to determine if these devices are on the same subnet or different subnets. You can use the address and mask of each device in order to determine to which subnet each address belongs.

Device A: 172.16.17.30/20
Device B: 172.16.28.15/20

Determine the Subnet for Device A:

```
172.16.17.30    -  10101100.00010000.00010001.00011110
255.255.240.0 -  11111111.11111111.11110000.00000000
.........................| sub|....................
subnet =                 10101100.00010000.00010000.00000000 = 172.16.16.0
```

Looking at the address bits that have a corresponding mask bit set to one, and setting all the other address bits to zero (this is equivalent to performing a logical "AND" between the mask and address), shows you to which subnet this address belongs. In this case, DeviceA belongs to subnet 172.16.16.0.

Determine the Subnet for Device B:

```
172.16.28.15    -  10101100.00010000.00011100.00001111
255.255.240.0 -  11111111.11111111.11110000.00000000
.........................| sub|....................
subnet =                 10101100.00010000.00010000.00000000 = 172.16.16.0
```

From these determinations, Device A and Device B have addresses that are part of the same subnet.

Sample Exercise 2

Given the Class C network of 204.15.5.0/24, subnet the network in order to create the network in Figure 3 with the host requirements shown.

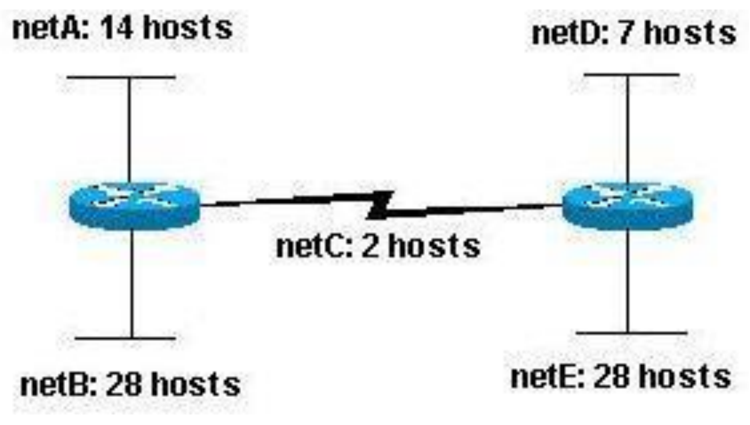
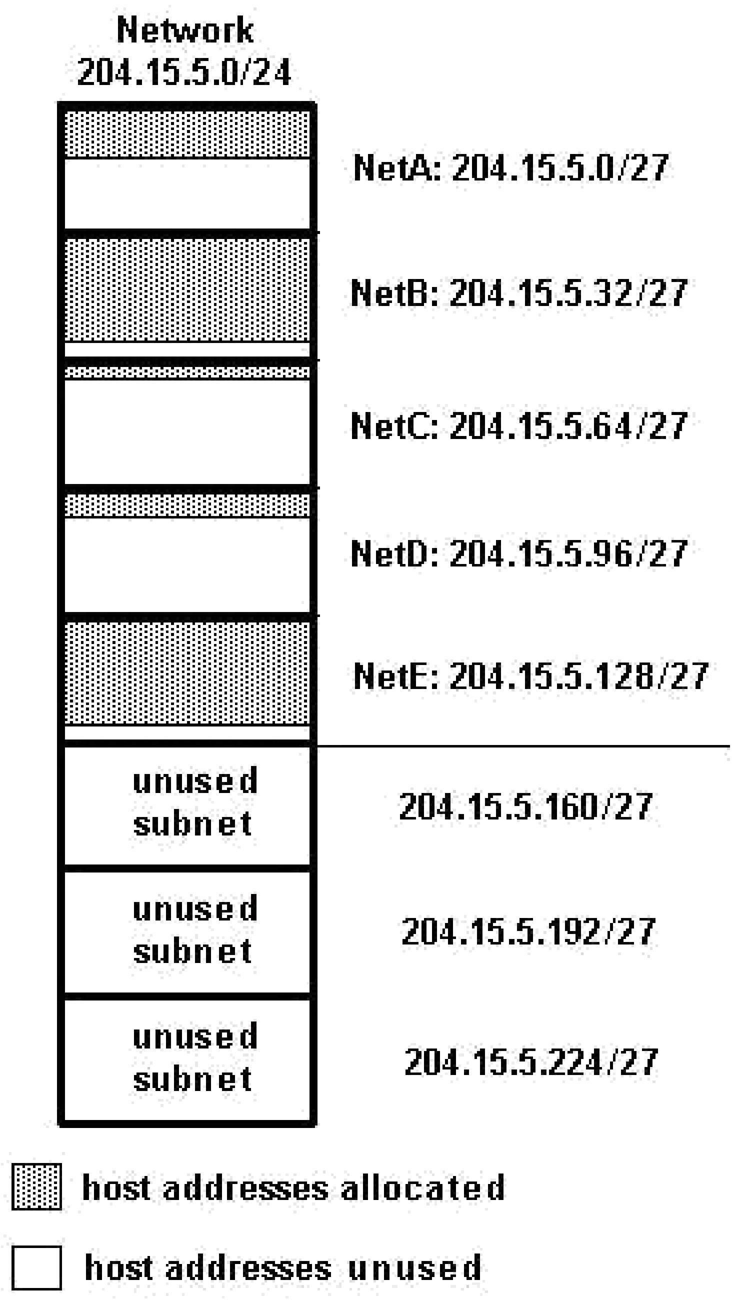


Figure 3



The subnets that are being used, Net A, Net C, and Net D have a lot of unused host address space. It is possible that this was a deliberate design accounting for future growth, but in many cases this is just wasted address space due to the fact that the same subnet mask is used for all the subnets.

Variable Length Subnet Masks (VLSM) allows you to use different masks for each subnet, thereby using address space efficiently.

VLSM Example

Given the same network and requirements as in Sample Exercise 2 develop a subnetting scheme with the use of VLSM, given:

Net A: must support 14 hosts
net B: must support 28 hosts
net C: must support 2 hosts net
D: must support 7 hosts net E:
must support 28 host

Determine what mask allows the required number of hosts.

netA: requires a /28 (255.255.255.240) mask to support 14 hosts netB: requires a /27 (255.255.255.224) mask to support 28 hosts netC: requires a /30 (255.255.255.252) mask to support 2 hosts netD*: requires a /28 (255.255.255.240) mask to support 7 hosts netE: requires a /27 (255.255.255.224) mask to support 28 hosts

* a /29 (255.255.255.248) would only allow 6 usable host addresses therefore netD requires a /28 mask.

The easiest way to assign the subnets is to assign the largest first. For example, you can assign in this manner:

netB: 204.15.5.0/27 host address range 1 to 30 netE: 204.15.5.32/27 host address range 33 to 62 netA: 204.15.5.64/28 host address range 65 to 78 netD: 204.15.5.80/28 host address range 81 to 94 netC: 204.15.5.96/30 host address range 97 to 98

This can be graphically represented as shown in Figure 5:

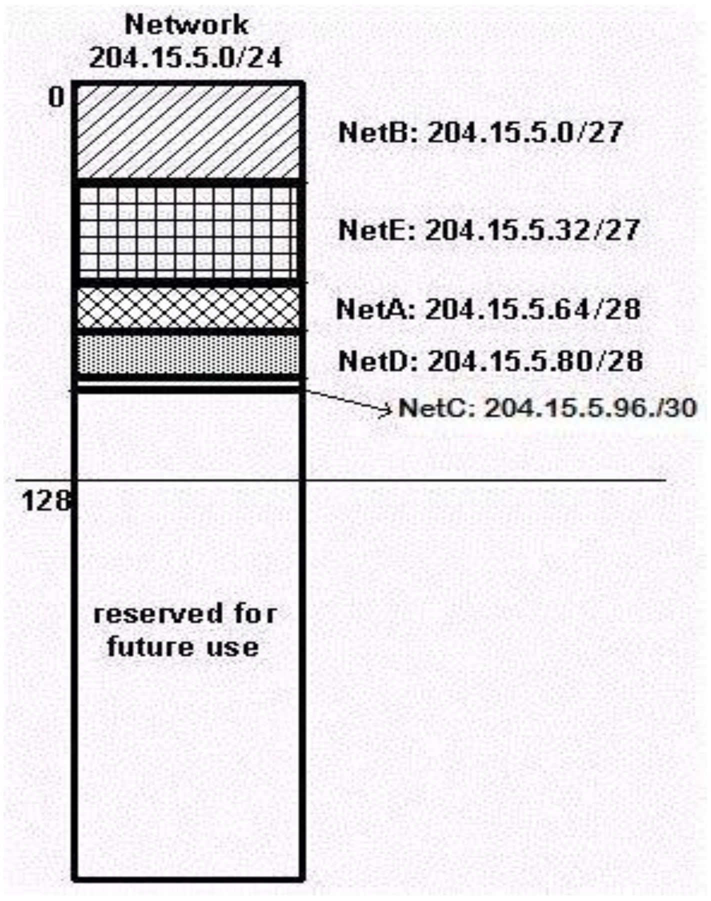


Figure 5

CIDR:

Classless Inter domain Routing (CIDR) was introduced in order to improve both address space utilization and routing scalability in the Internet. It was needed because of the rapid growth of the Internet andgrowth of the IP routing tables held in the Internet routers.

CIDR moves way from the traditional IP classes (Class A, Class B, Class C, and so on). In CIDR , an IP network is represented by a prefix, which is an IP address and some indication of the length of the mask. Length means the number of left-most contiguous mask bits that are set to one. So network 172.16.0.0 255.255.0.0 can be represented as 172.16.0.0/16. CIDR also depicts a more hierarchical Internet architecture, where each domain takes its IP addresses from a higher level. This allows for the summarization of the domains to be done at the higher level. For example, if an ISP owns network 172.16.0.0/16, then the ISP can offer 172.16.1.0/24, 172.16.2.0/24, and so on to customers. Yet, when advertising to other providers, the ISP only needs to advertise 172.16.0.0/16.

For more information on CIDR, see RFC 1518 and RFC 1519 .

Please refer the above theory for answers:

1. What is Subnet mask?
2. Explain concept of subnetting?
3. What is CIDI notation?
4. What is netid and host id in networking?

Conclusion:

Hence we studied and implement program to demonstrate subletting and find the subnet masks.

Experiment Number: B2

Lab Assignment on Unit III: (Mandatory Assignment) (Use C/C++)

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

OBJECTIVES:

- Students will able to understand working of link state /Distance vector routing protocol

PROBLEM STATEMENT

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

✓ **Distance vector :**

Routing algorithm is a part of network layer software which is responsible for deciding which output line an incoming packet should be transmitted on. If the subnet uses datagram internally, this decision must be made anew for every arriving data packet since the best route may have changed since last time. If the subnet uses virtual circuits internally, routing decisions are made only when a new established route is being set up. The latter case is sometimes called session routing, because a rout remains in force for an entire user session (e.g., login session at a terminal or a file).

Routing algorithms can be grouped into two major classes: adaptive and nonadaptive. Nonadaptive algorithms do not base their routing decisions on measurement or estimates of current traffic and topology. Instead, the choice of route to use to get from I to J (for all I and J) is compute in advance, offline, and downloaded to the routers when the network ids booted. This procedure is sometime called static routing.

Adaptive algorithms, in contrast, change their routing decisions to reflect changes in the topology, and usually the traffic as well. Adaptive algorithms differ in where they get information (e.g., locally, from adjacent routers, or from all routers), when they change the routes (e.g., every ΔT sec, when the load changes, or when the topology changes), and what metric is used for optimization (e.g., distance, number of hops, or estimated transit time).

Two algorithms in particular, distance vector routing and link state routing are the most popular. Distance vector routing algorithms operate by having each router maintain a table (i.e., vector) giving the best known distance to each destination and which line to get there. These tables are updated by exchanging information with the neighbors.

The distance vector routing algorithm is sometimes called by other names, including the distributed Bellman-Ford routing algorithm and the Ford-Fulkerson algorithm, after the researchers who developed it (Bellman, 1957; and Ford and Fulkerson, 1962). It was the original ARPANET routing algorithm and was also used in the Internet under the RIP and in early versions of DECnet and Novell’s IPX. AppleTalk and Cisco routers use improved distance vector protocols.

In distance vector routing, each router maintains a routing table indexed by, and containing one entry for, each router in subnet. This entry contains two parts: the preferred out going line to use for that destination, and an estimate of the time or distance to that destination. The metric used might be number of hops, time delay in milliseconds, total number of packets queued along the path, or something similar.

The router is assumed to know the “distance” to each of its neighbor. If the metric is hops, the distance is just one hop. If the metric is queue length, the router simply examines each queue. If the metric is delay, the router can measure it directly with special ECHO packetshat the receiver just time stamps and sends back as fast as possible.

The Count to Infinity Problem.

Distance vector routing algorithm reacts rapidly to good news, but leisurely to bad news. Consider a router whose best route to destination *X* is large. If on the next exchange neighbor *A* suddenly reports a short delay to *X*, the router just switches over to using the line to *A* to send traffic to *X*. In one vector exchange, the good news is processed.

To see how fast good news propagates, consider the five node (linear) subnet of following figure, where the delay metric is the number of hops. Suppose A is down initially and all the other routers know this. In other words, they have all recorded the delay to A as infinity.

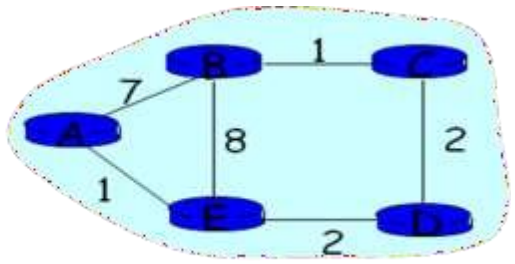
A B C D E A B C D E

____ ____ ____ ____ ____ ____ ____ ____

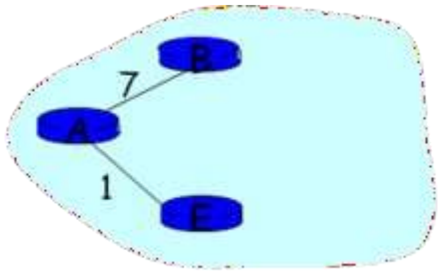
∞	∞	∞	∞	Initially	1	2	3	4	Initially
1	∞	∞	∞	After 1 exchange	3	2	3	4	After 1 exchange
1	2	∞	∞	After 2 exchange	3	3	3	4	After 2 exchange
1	2	3	∞	After 3 exchange	5	3	5	4	After 3 exchange
1	2	3	4	After 4 exchange	5	6	5	6	After 4 exchange
					7	6	7	6	After 5 exchange
					7	8	7	8	After 6 exchange
						:			
					∞	∞	∞	∞	

Many ad hoc solutions to the count to infinity problem have been proposed in the literature, each one more complicated and less useful than the one before it. The **split horizon** algorithm works the same way as distance vector routing, except that the distance to *X* is not reported on line that packets for *X* are sent on (actually, it is reported as infinity). In the initial state of right figure, for example, *C* tells *D* the truth about distance to *A* but *C* tells *B* that its distance to *A* is infinite. Similarly, *D* tells the truth to *E* but lies to *C*.

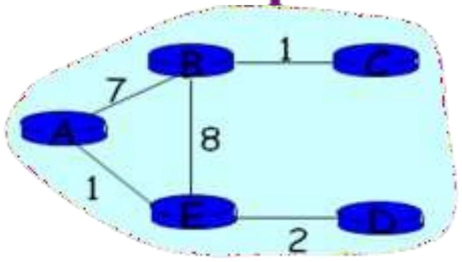
- A's distance vector D(A,*):
  - After Iteration 1 is: [0, 7, INFINITY, INFINITY, 1]
  - After Iteration 2 is: [0, 7, 8, 3, 1]
  - After Iteration 3 is: [0, 7, 5, 3, 1]
  - After Iteration 4 is: [0, 6, 5, 3, 1]



Example network



A's 1-hop view (After 1st iteration)



A's 2-hop view (After 2nd Iteration)

Algorithm:

• c(x,v) = cost for direct link from x to v

–Node x maintains costs of direct links c(x,v)

• Dx(y) = estimate of least cost from x to y

–Node x maintains distance vector **D**x = [Dx(y): y _ N]

• Node x maintains its neighbors’ distance vectors

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

• Each node v periodically sends Dv to its neighbors

– And neighbors update their own distance vectors

–Dx(y) _ minv{c(x,v) + Dv(y)} for each node y ε N

• Over time, the distance vector Dx converges

Output:

A sample run of the program works

```
as:-Enter the number of nodes :
3
Enter the cost matrix :
0 2 7
2 0 1
7 1 0
For router 1
node 1 via 1 Distance 0
node 2 via 2 Distance 2
node 3 via 3 Distance 3
For router 2
node 1 via 1 Distance 2
node 2 via 2 Distance 0
node 3 via 3 Distance 1
For router 3
node 1 via 1 Distance 3
node 2 via 2 Distance 1
node 3 via 3 Distance 0
```

Conclusion:

The least-cost route between any two nodes is the route with minimum distance. Each node maintains a vector of minimum distances to every node.

Experiment Number: B3

Lab Assignment on Unit IV: (Mandatory Assignment)

Title: Study of RIP, OSPF and BGP using packet tracer.

OBJECTIVES:

- Student should be able to understand and design network using different network devices such machine, router, switches using packet tracer.

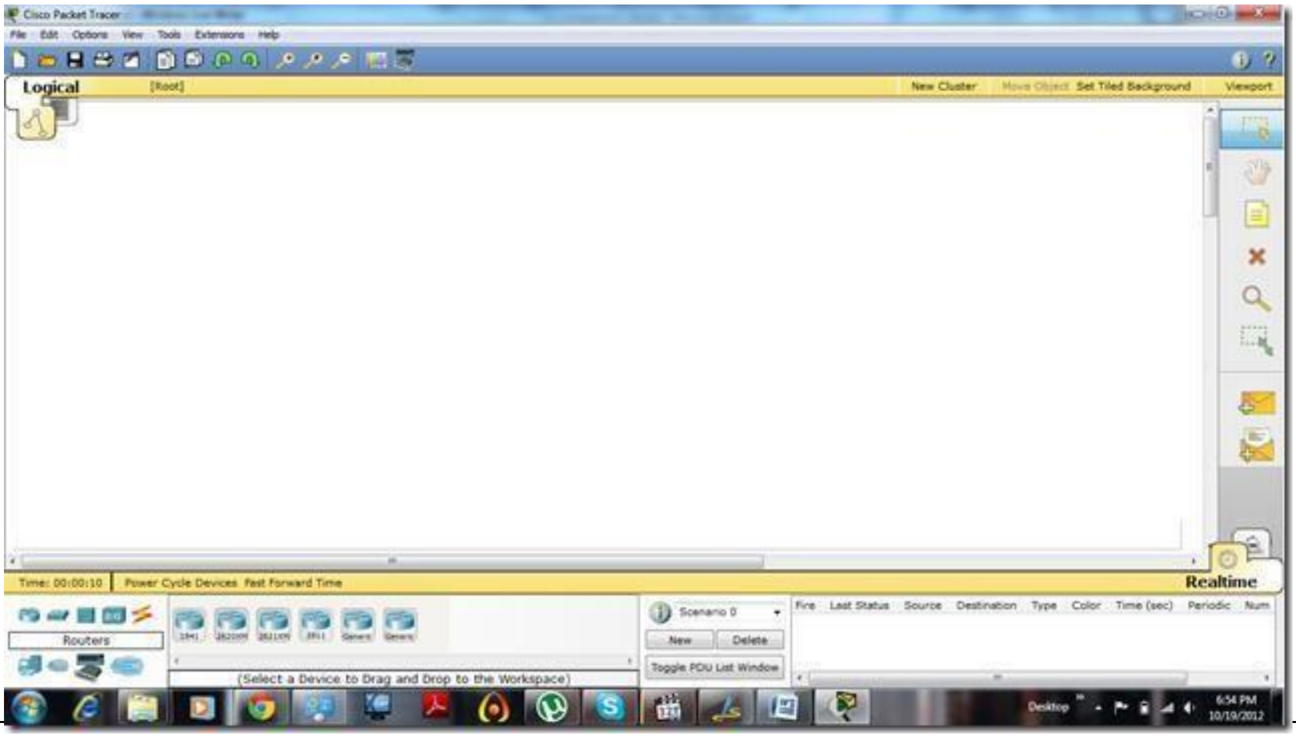
PROBLEM STATEMENT

Configure RIP/OSPF/BGP using packet Tracer.

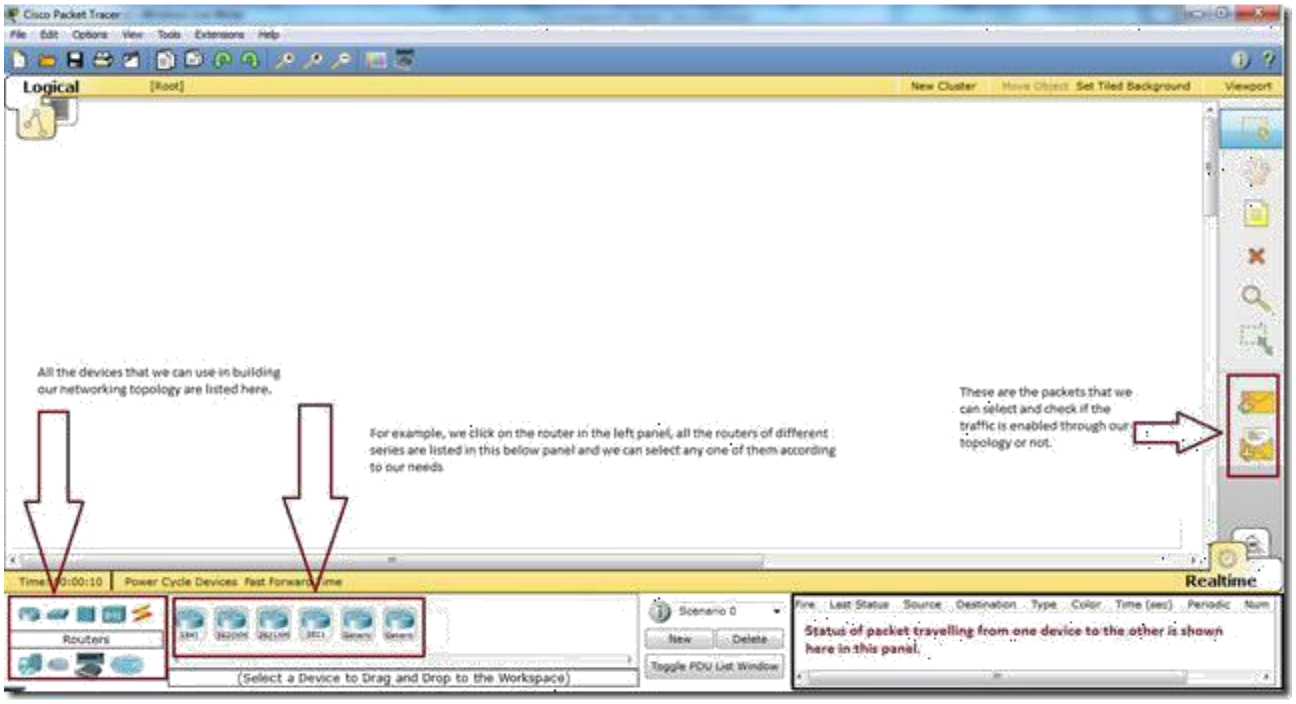
Theory: -

Packet Tracer is a powerful network simulator that can be utilized in training for CCNA and CCNP certification exam by allowing students to create networks with an almost unlimited number of devices and to experience troubleshooting without having to buy real Cisco routers or switches. The tool is created by Cisco Systems. The purpose of Packet Tracer is to offer students a tool to learn the principles of networking as well as develop Cisco technology specific skills. However, it is not be used as a replacement for Routers or Switches.

Here how it looks like after we start it.



We are different modules and panels available in the packet tracer. Some important modules, which are important to understand for the working in Packet Tracer, are mentioned in the following diagram.

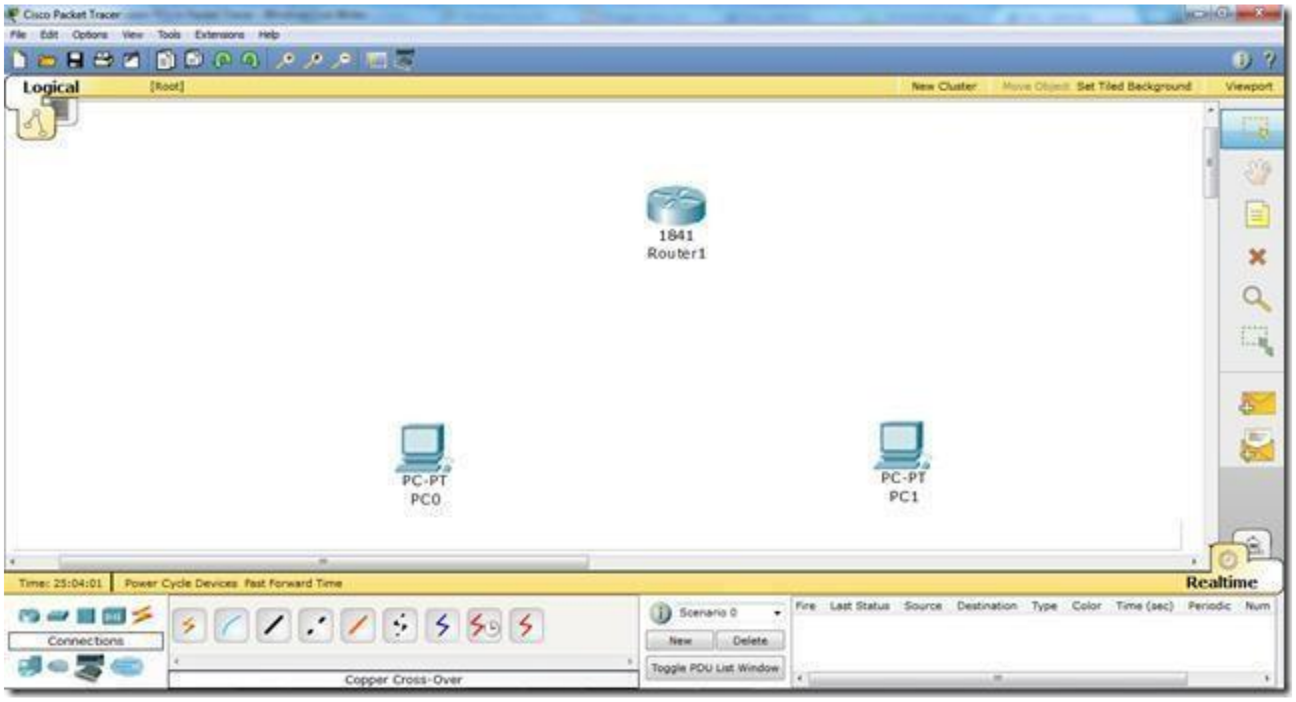

All the devices that we can use in building our networking topology are listed here.

For example, we click on the router in the left panel, all the routers of different series are listed in this below panel and we can select any one of them according to our needs

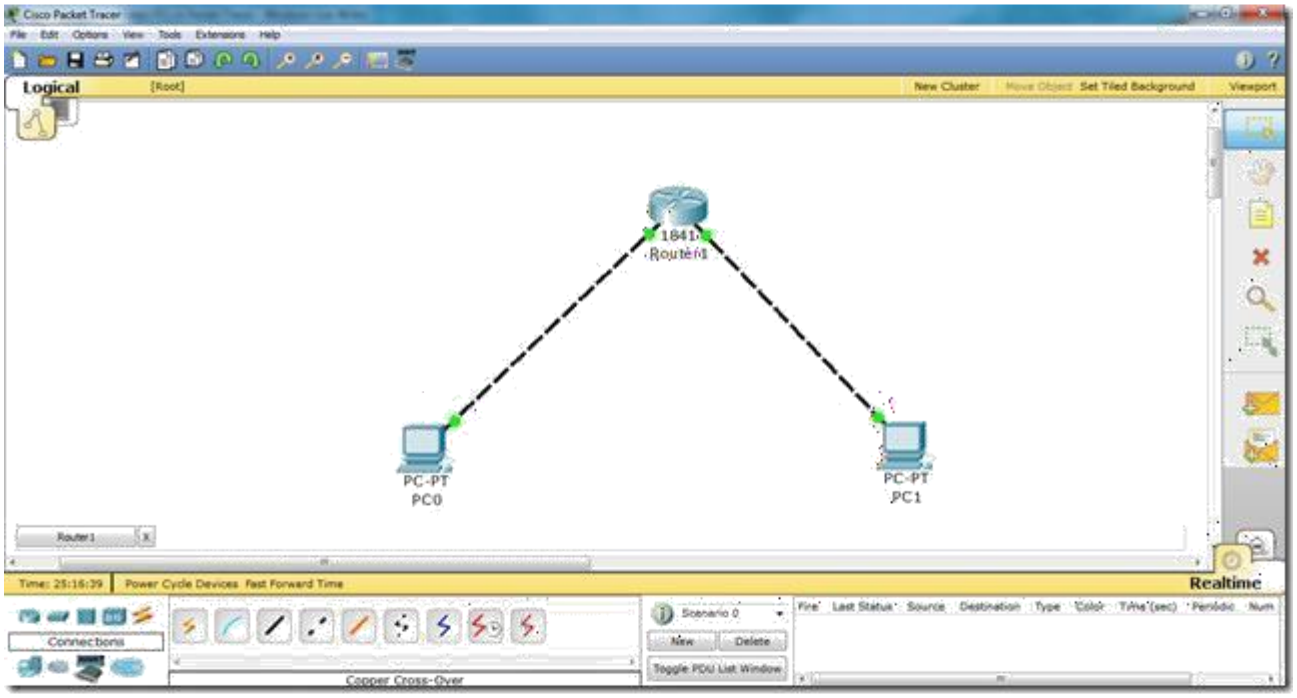
These are the packets that we can select and check if the traffic is enabled through our topology or not.

Status of packet travelling from one device to the other is shown here in this panel.

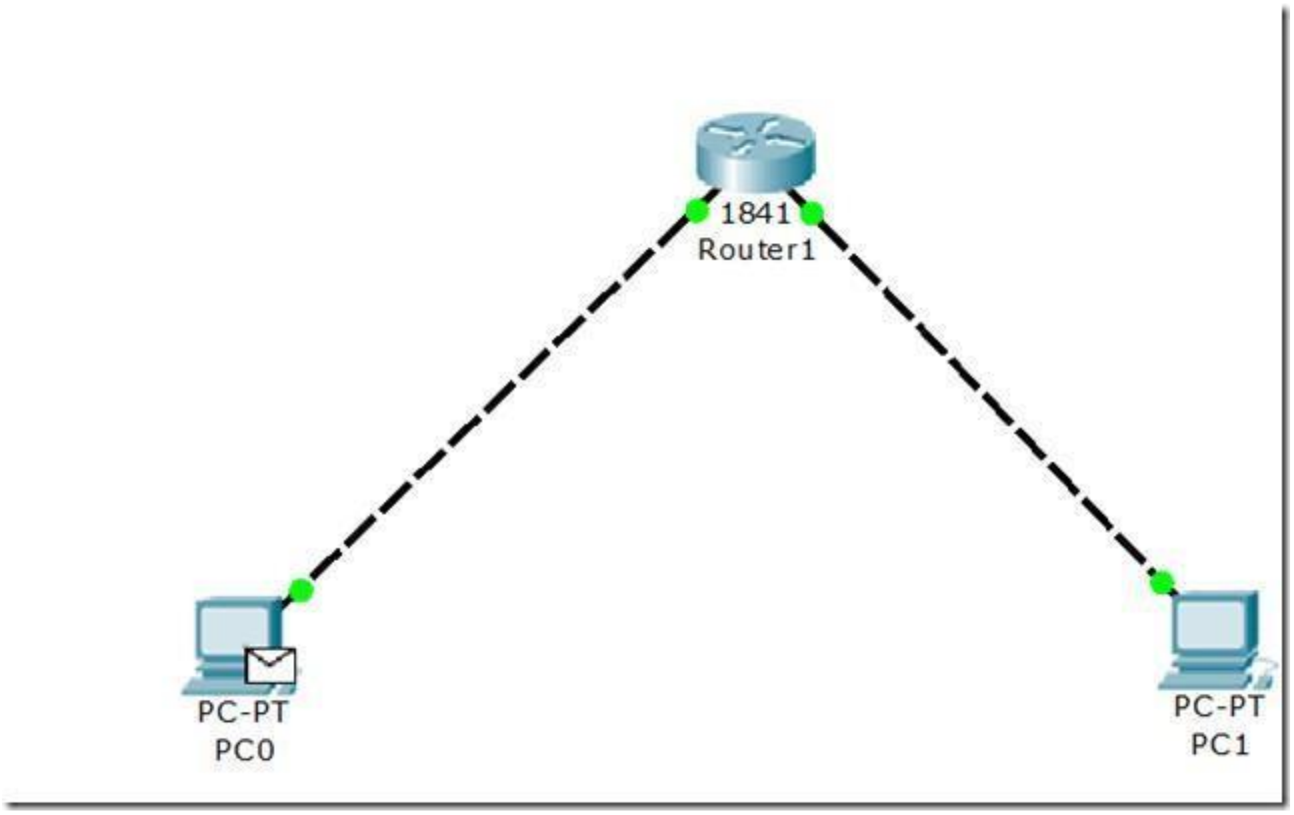
Now, in order to create a topology, we will have to select some of the devices and put them in our main window i.e. the white portion of packet tracer. and here how it looks after we add the devices.



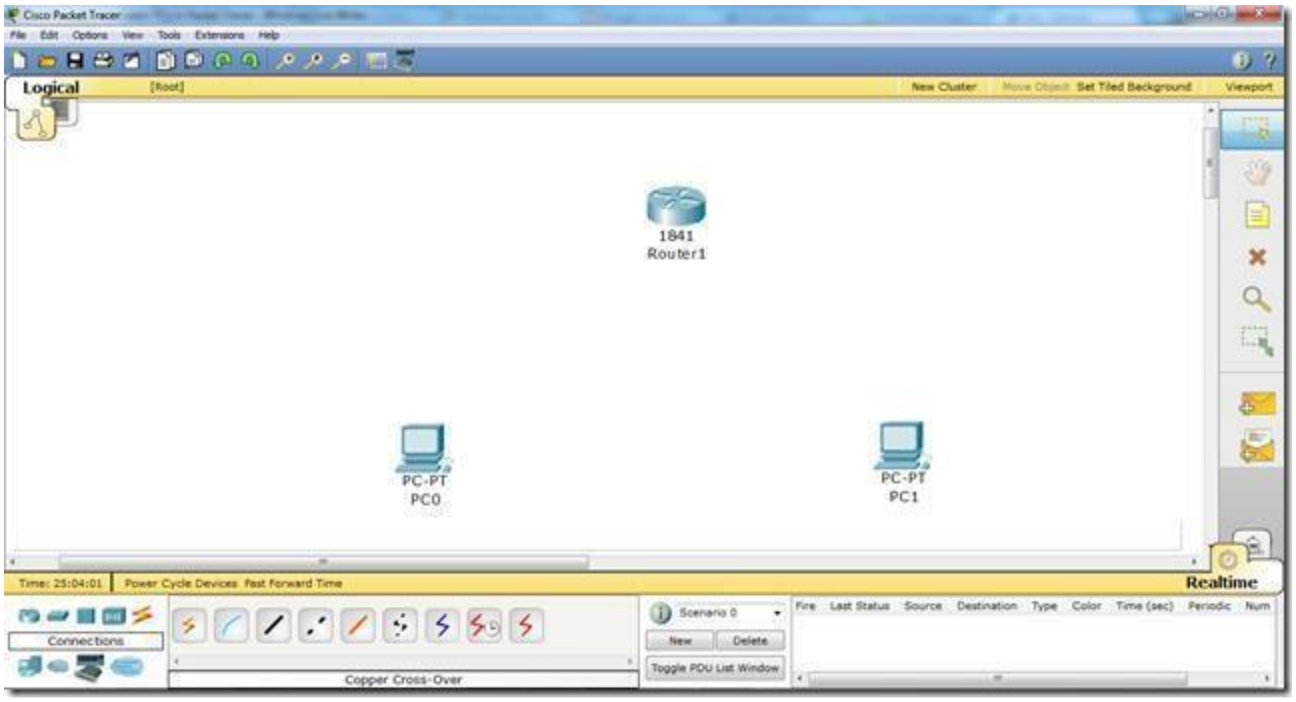
Now, we will have to connect these devices and for that we use cables. To understand the cables, please refer to my following article.



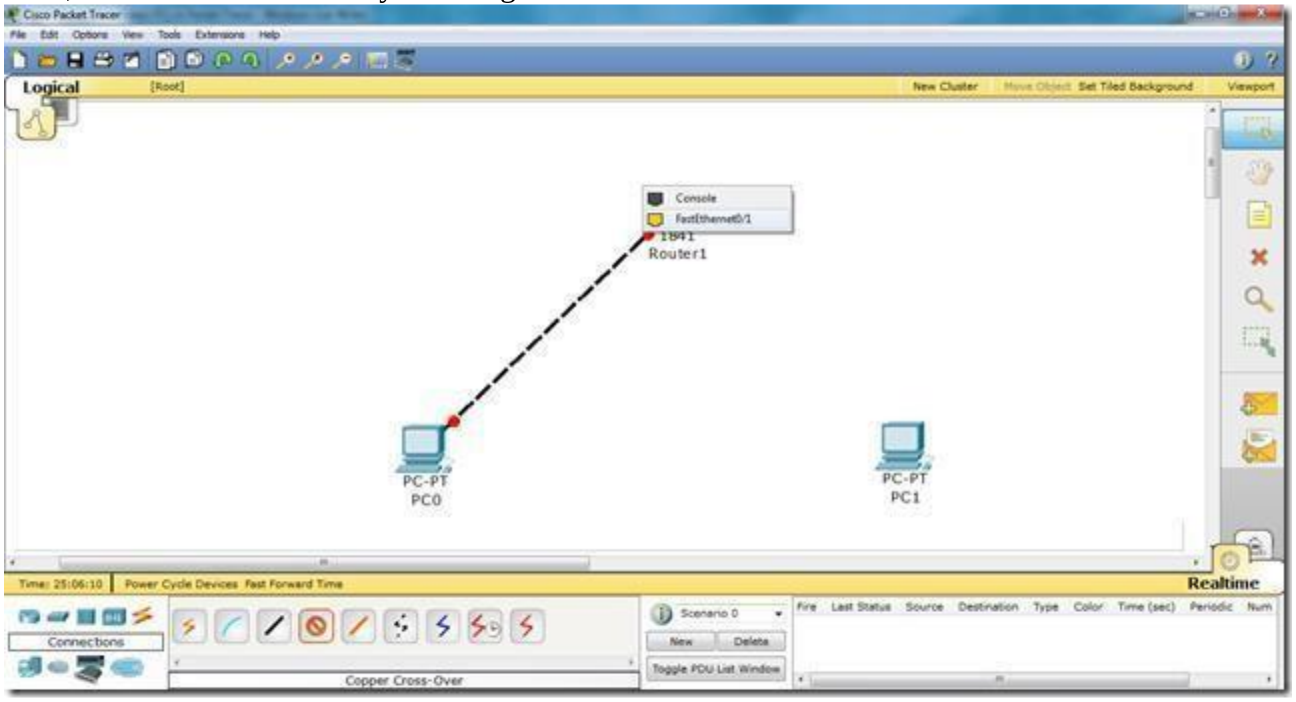
And after you successfully create the topology, you can check either the traffic is flowing or not by selecting the packet from right panel and putting it on both PCs as follows.



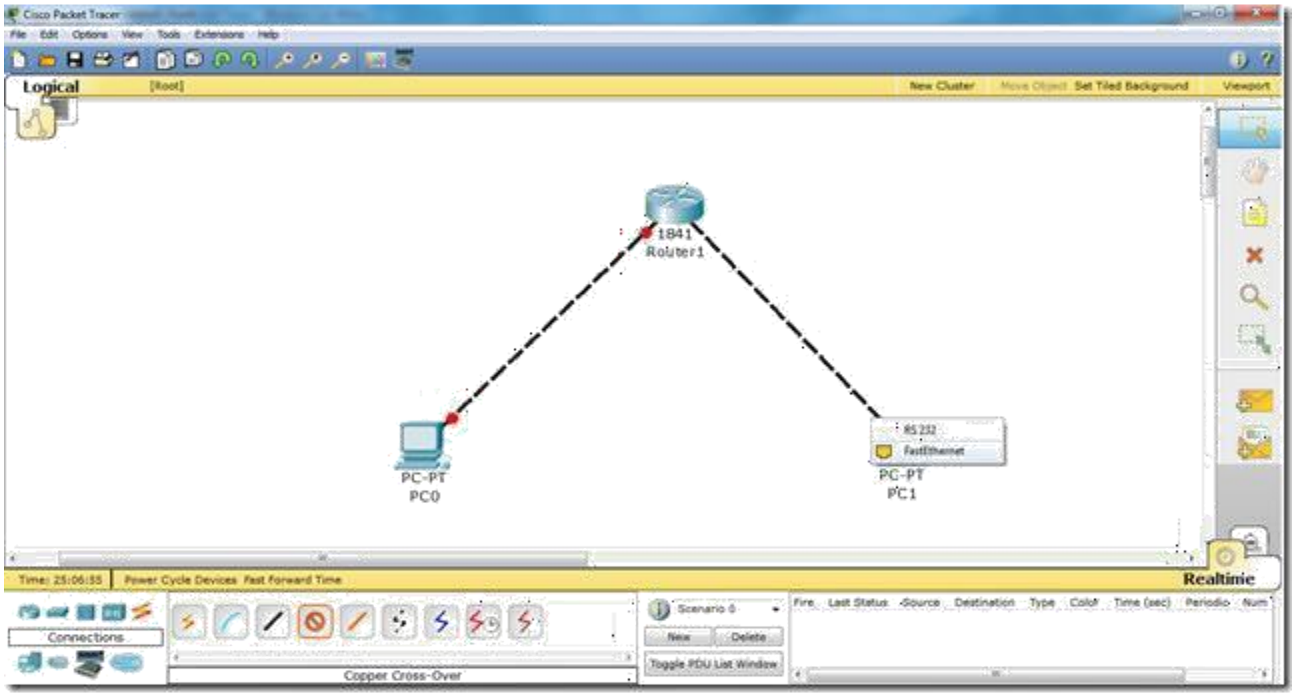
For detailed article on making the topology and successfully enabling the communication, Here, we will see communication enabled between PCs via Router in Packet Tracer. So, for this we need two PCs, a router ,and two cross over cables to connect them.
Important point is that we use cross over cable to connect PC to a router because they both use the same pins for transmission and receiving of data.



Now, we will connect them by selecting fast ethenet interfaces on both ends.

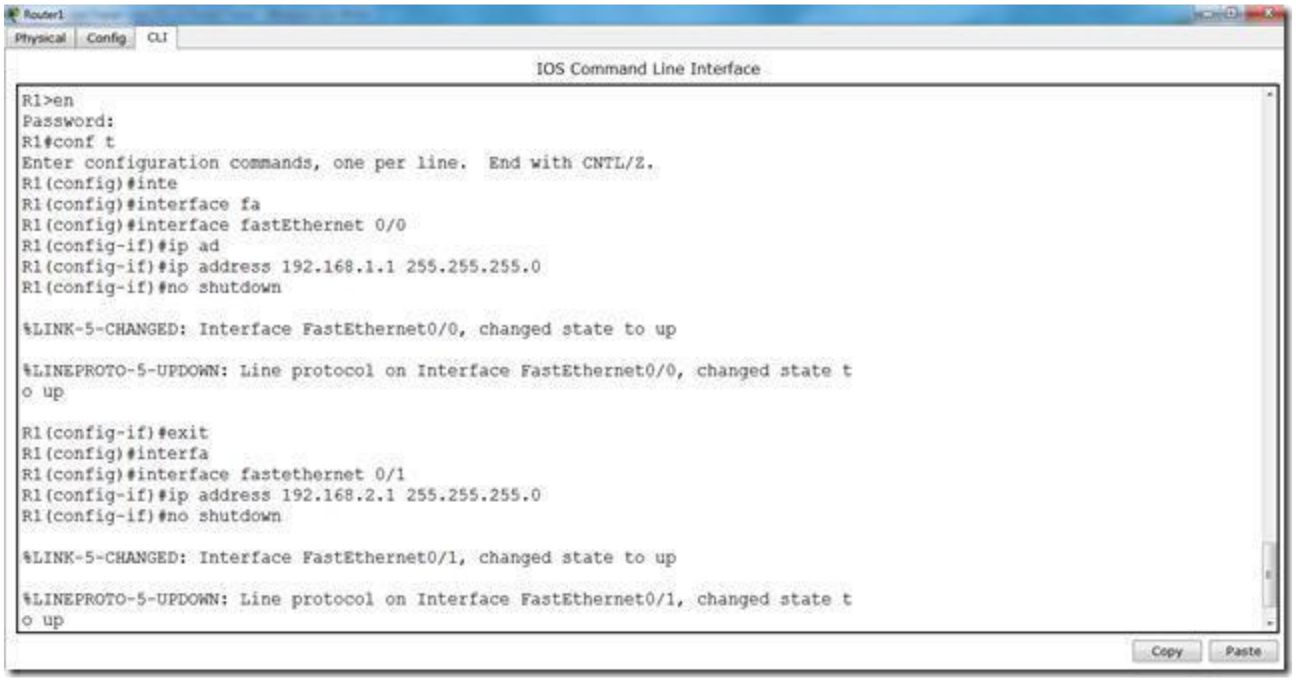


Similarly, on the PC side we will select fast Ethernet interface.



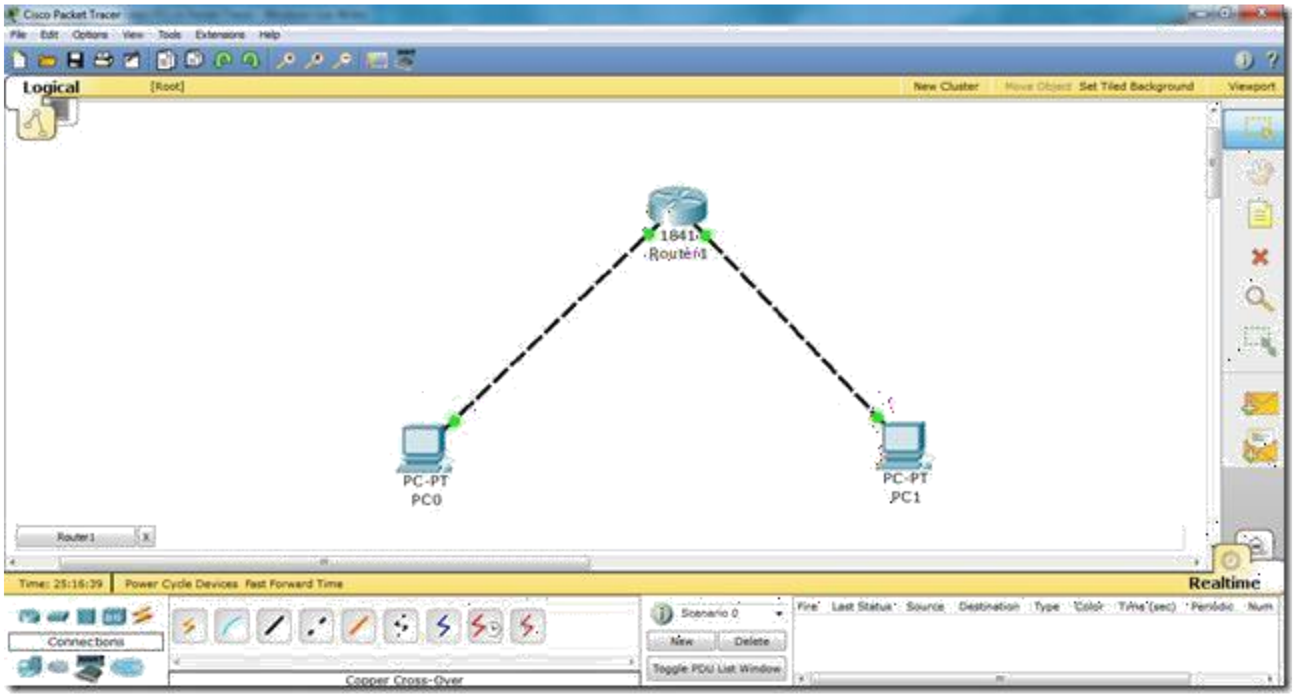
Now, we have connect the devices. Further, we will go to the router CLI mode and enter the following commands.Step by step ,we will have to do the following things.

i. Access the interfaces one by one
ii. Assign IP addresses to interfaces
iii. Change the status of the interfaces i.e. from Down to Up.
iv. Assign IP addresses to PCs.
v. Assign Default GateWay to PCs. FYI fast ethernetip address is the gateway address to the PC. Now, commands of the Router CLI mode are as follows.



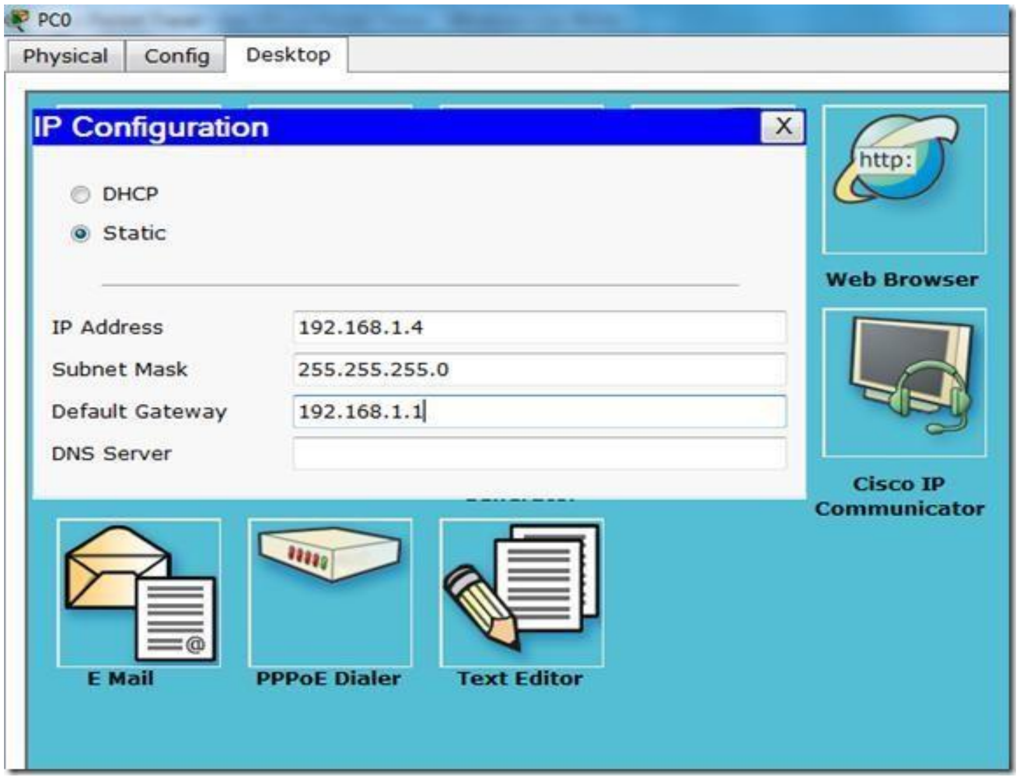
```
R1>en
Password:
R1#conf t
Enter configuration commands, one per line.  End with CNTL/Z.
R1(config)#inte
R1(config)#interface fa
R1(config)#interface fastEthernet 0/0
R1(config-if)#ip ad
R1(config-if)#ip address 192.168.1.1 255.255.255.0
R1(config-if)#no shutdown

%LINK-5-CHANGED: Interface FastEthernet0/0, changed state to up

%LINEPROTO-5-UPDOWN: Line protocol on Interface FastEthernet0/0, changed state t
o up

R1(config-if)#exit
R1(config)#interfa
R1(config)#interface fastethernet 0/1
R1(config-if)#ip address 192.168.2.1 255.255.255.0
R1(config-if)#no shutdown

%LINK-5-CHANGED: Interface FastEthernet0/1, changed state to up

%LINEPROTO-5-UPDOWN: Line protocol on Interface FastEthernet0/1, changed state t
o up
```

Now, we have accessed both interfaces one by one and we have assigned IP addresses respectively.



See the difference the lights have changed the color from Red to Green :)
Now, lets assign IP addresses to the PCs.
Click on PC1, go to Desktop, then click IP Configuration.

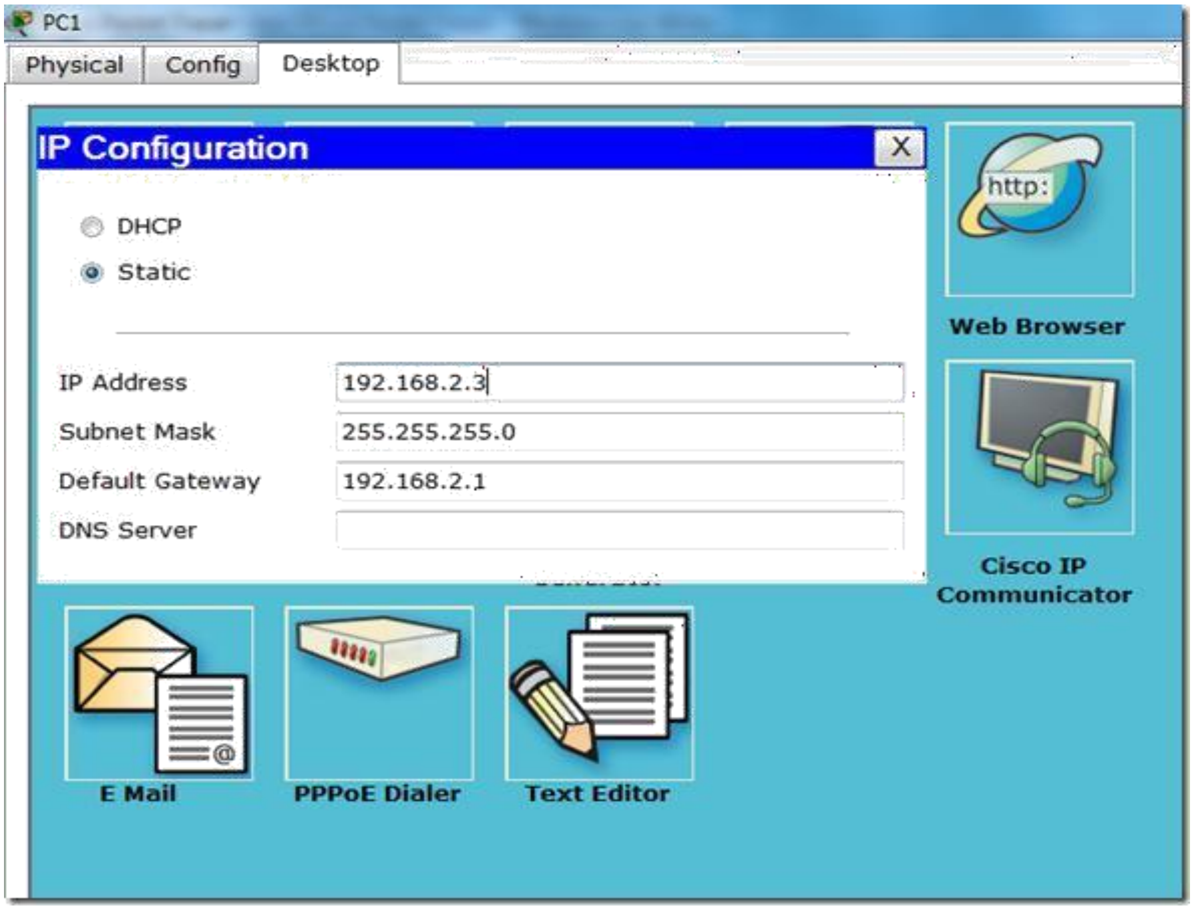
PC1:



DHCP [✓ Static]

IP Address 192.168.1.4
Subnet Mask 255.255.255.0
Default Gateway 192.168.1.1
DNS Server

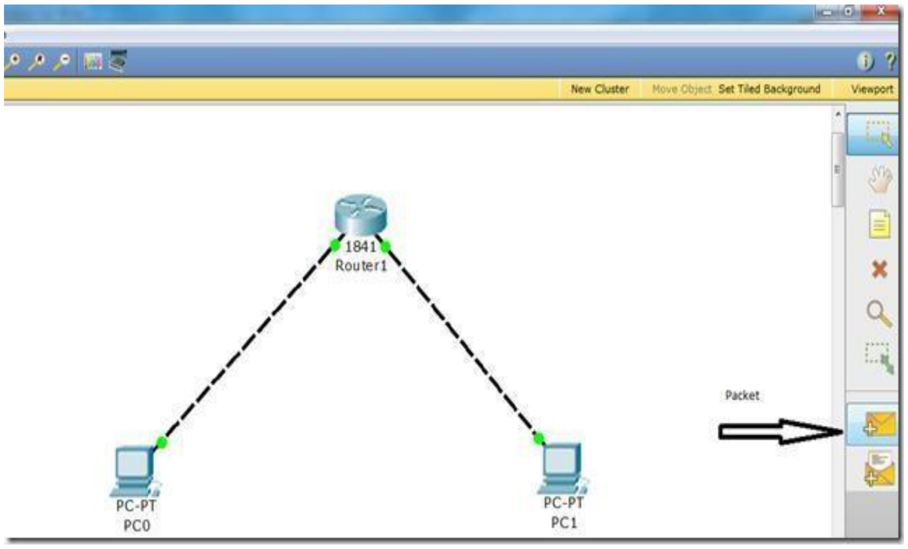
PC2:



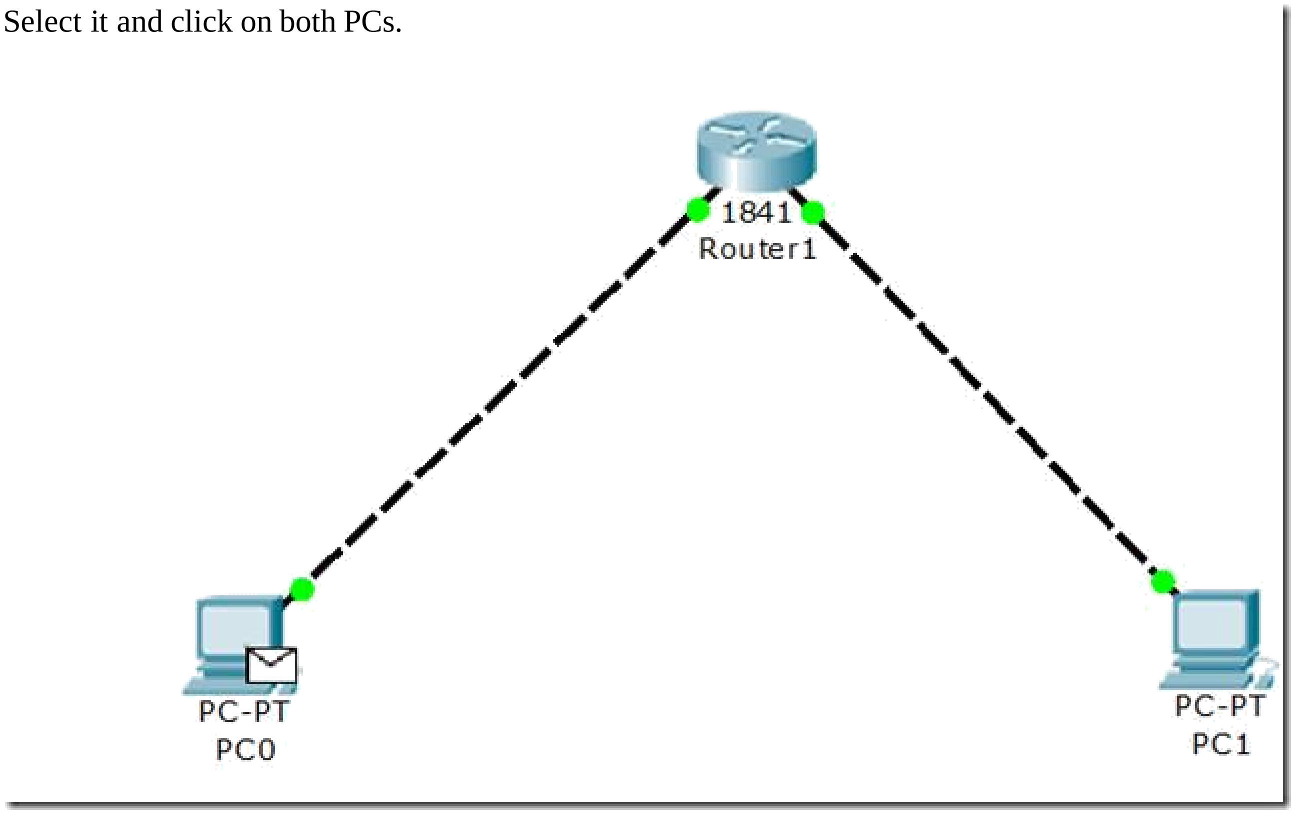
DHCP
[✓ Static]

IP Address: 192.168.2.3
Subnet Mask: 255.255.255.0
Default Gateway: 192.168.2.1
DNS Server:

Now, our communication is enabled and we are able to communicate from PC1 to PC2 via Router.
Click on the packet in the right panel on the packet tracer, then click on PC1 and then click on PC2. You will see the successful packet tracer (status is shown in the bottom right corner)



Select it and click on both PCs.



Please refer the above theory for answers:

1. What is Packet Tracer?
2. What are different routing protocols?

Conclusion:

Hence we studied and configure RIP, OSPF and BGP using packet tracer.

Experiment Number: B4

Lab Assignment on Unit IV: (Use java/Python)

Title: Study of TCP socket programming for wired

OBJECTIVES:

1. Getting familiar with the client-server communication model.
2. Learning the most important library functions (the UNIX and Internet sockets) used for the design of the client-server applications.
3. Designing simple client or server applications for stream.

PROBLEM STATEMENT

Write a program using TCP socket for wired network for following

a. Say Hello to Each other

b. File transfer

c. Calculator

Theory: -

Server-client communication uses TCP/IP. Three main things are needed to establish connection between server-client models:

1) Transport protocol (TCP or UDP)
2) Socket
3) IP address and port

Transport protocol (TCP)

TCP is a connection oriented protocol which stands for transmission control protocol. It is a reliable and secure protocol. In TCP, the receiver can generate the acknowledgement of received packet so sender (client) does need to wait for the acknowledgement and if the back response doesn't come or any packet error is generated, it will resend to the client.

Sockets programming Concepts: -

There are different types of sockets. What makes the difference between them is the topology of the communication and the types of addresses used. There are three types of addresses: the first one is represented by the Internet addresses (the corresponding sockets are called Internetsockets), paths to the local nodes for communicating between processes from the same station (Unix sockets) or addresses of the X.25 type (X.25 sockets, least used).

Socket – it represents an end of the communication, defining a form of the Inter Process Communication, for processes found on the same station, or on different stations of the same network. A pair of connected sockets represents a communication interface between two processes, an interface similar to the pipe interface in Unix.

Association – it defines in a unique way the connection established between two ends of communication, as being a set of parameters like (*local_protocol*, *local_address*, *local_ port*, *remote_address, remote_port*). The *source_port* and *destination_port* notions allow a unique definition of the communication between pairs of sockets and make a unique identification of a SAP (Service access point) of the transport level.

Socket descriptor – it is a file descriptor; the argument used in the library functions.
Binding – it is the operation that associates a socket address to a socket, so the socket can be accessed.

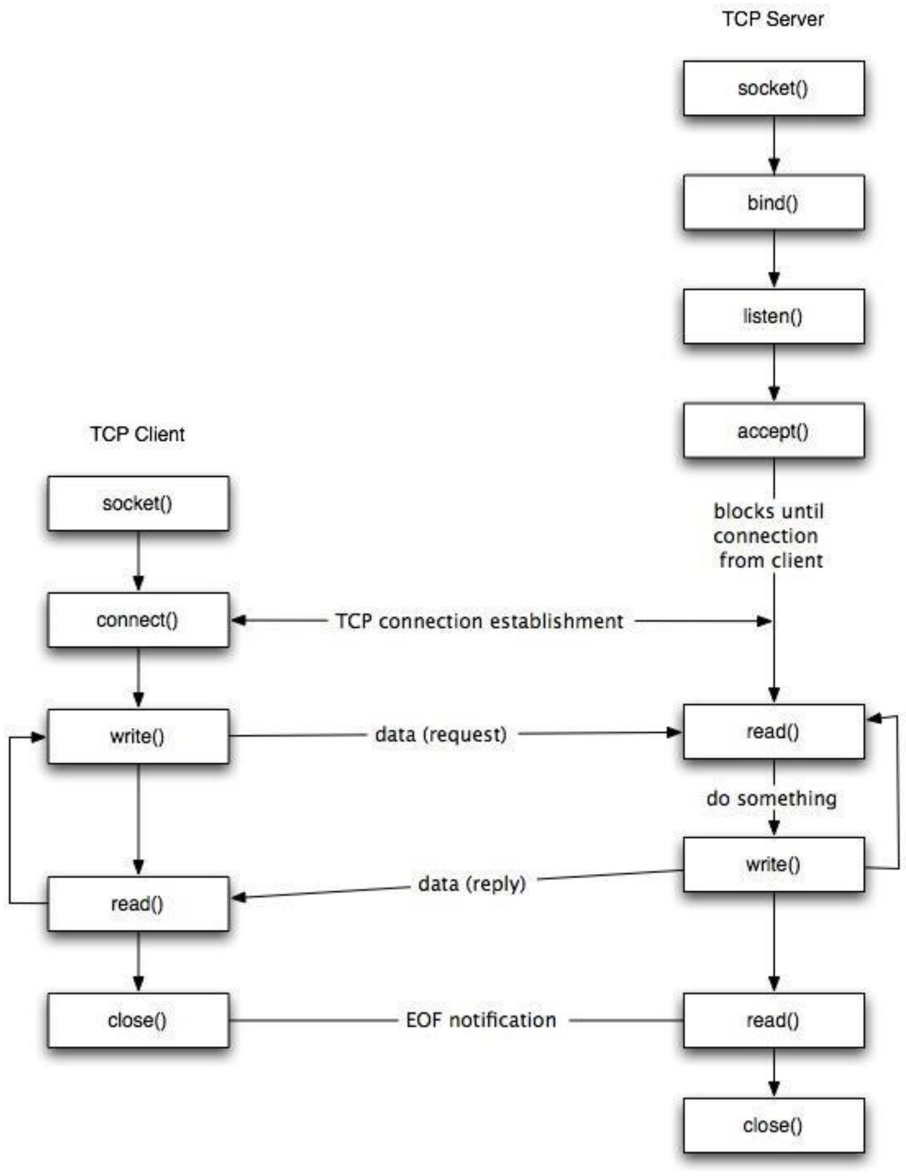
Port – it is an identifier used for making the difference between sockets found at the same address. There can be used addresses between 1024 – 49151 (registered ports) and addresses between 49152 – 65535, named also dynamic or ephemeral ports; the addresses between 1 – 1023 are reserved for the system (see the file etc/services in Unix).

The family of addresses – it represents the format of the addresses used for identifying the addresses used in sockets (usually the Internet and Unix addresses).

The socket address – it depends on the family of addresses used (for the domain of the family of Internet addresses, the socket address is made up by the Internet address and by the port address used at the host).

The Internet address – it is a 4-byte-long address, which identifies a node (host or router) inside a network.

The client – server model: -



Any operation inside a network can be seen as a client process that communicates with a server process. The server process creates a socket, associates to the socket an address and launches a mechanism for listening for the connection requests of the clients. The client process creates a socket and asks for a connection to the server. After accepting the request by the server and after the connection has been established, a communication between the sockets can be established.

This classical model can vary with the nature of the designed application; it can be symmetric or asymmetric, based on simplex or duplex communication. Using this client – server model, different applications have been made, such as: telnet (for the connection of a remote host using the port 23) for which there is a program at that host, program called telnetd, that will respond, ftp/ftpd or bootp/bootpd.

The usual functions from the socket library are:

- socket() – creates a new socket descriptor;
- bind() – makes the binding of an address and of the corresponding port to the socket;
- connect() – allows the establishment of a connection with a remote server;
- listen() – listens for the connection requests; this function is used in a *passive socket;*
- accept() – allows the creation of a new socket, corresponding to a connection request;
- send(), recv(), sendto(), recvfrom() – transmits/ receives *streams* or *datagrams;*
- close(), shutdown() – closes a connection;
- getpeername() – determines the name of the peer host from the connection;
- getsocketname() – determines the name of the socket used;
- gethostbyname(), gethostbyaddr() – determines the name pr the address from the specified host.

The communication between the client processes and the server is based on the call of the socket() function, call that returns a socket descriptor. This descriptor is used in the calls of different functions specialized in the data transmission (as send() and recv()).

Stream sockets vs. datagram sockets:-

The communication between client and server programs is made using the level protocols TCP ("Transmission Control Protocols"), which is a connection-oriented transport protocol or UDP ("User Datagram Protocol"), which is a connectionless-oriented transport protocol.

The *stream sockets* are secure communication flows (with no errors), full duplex, based on the TCP protocol, which provides a sequential, errorless data transmission.

The *datagram sockets*, also known under the name of connectionless sockets, use the IP addresses for routing, but the transport level protocol is the UDP. These sockets do not maintain an open connection during the communication, but they make the transfer packet by packet (the tftp - Trivial File Transfer Protocol, the bootp applications use the datagram sockets).

Data types used by the socket interface: -

The next structure stores the socket address for different types of sockets:

```
struct sockaddr
{
unsigned short sa_family;  /* the family of addresses, FA_xxx */
char     sa_data[14];        /* 14 bytes - the protocol address */
};
```

In order to be able to use in programs the structure struct sockaddr, a parallel structure struct sockaddr_in has been designed:

```
struct sockaddr_in{
short int sin_family;               /* the family of addresses */
unsigned short int sin_port; /* the port number */
struct in_addr sin_addr;          /* the Internet address */
char sin_zero[8];                              /*unsed*/
};
and
```

```
/* the Internet address */
struct in_addr{
unsigned long s_addr;
};
```

This structure allows a simple reference to the elements of a socket address. Sin_zero will be set to 0, using bzero() or memset() functions. The pointer to the struct sockaddr_in maps the pointer to the structure struct sockaddr. We must highlight that sin_family corresponds to sa_family in the structure struct sockaddr, and it will be set to “AF_INET”, value used for the TCP/IP protocol family.

The conversion functions: -

Due to the fact that in data transmission, in order to design portable applications it is necessary to convert the transmitted data between the hosts that use the little_endian or the big_endian representation, and also due to the fact that the data representation at the network level is unique (the protocols of the TCP/IP family use the representation at the network level called *NetWork Byte Order*), some functions are needed to make these conversions.
Some of the most-used conversion functions in the socket programming are:

- htons() - “Host to Network Short”, host conversion to a short network;
- htonl() – “Host to Long Network”, host conversion at a long network;
- ntohs() – “Network to Host Short”, network conversion to a short host;
- ntohl() – “Network to Host Long”, network conversion to a long host.

Being known the fact that sin_addr and sin_port are put in the IP packet, respectively in the UDP-TCP, they have to be in the Network_byte_order, and the sin_family field, used by the kernel of the operating system (to find out what type of address the structure contains), in the Host Byte Order, it is necessary to use the conversion functions.

Supposing that we analyze the structure struct sockaddr_in ina, and an IP address “xxx.yyy.z.uu” that we want to store in the structure. The function that we use, inet_addr(), converts an IP address from the known format (the numbers-and-dots notation) in an unsigned integer:

```
Ina.sin_addr.s_addr  =  inet_addr(“xxx.yyy.z.uu”)
```

Ine_addr() returns the address in the Network Byte Order. For printing an address in the dot-notation, it is necessary to use the inet_ntoa() function (“ntoa” represents “network to ascii”), as we can see in the next example:

```
printf(“%s”, inet_ntoa(ina.sin_addr));
char *a1, *a2;

a1 = inet_ntoa(ina1.sin_addr);
a2 = inet_ntoa(ina2.sin_addr);
printf(“address 1: %s\n”, a1);
printf(“address 2: %s\n”, a2);
```

FAQS

Please refer the above theory for answers:

1. What is Socket?
2. What are different types of Socket?
3. What are different Socket functions or calls used in TCP Client Server communication?
4. Explain TCP protocol in details?

Conclusion:

Hence we studied and implement program to demonstrate TCP Socket programming for wired network for peer to peer chat and multi user chat.

Experiment Number: B5

Lab Assignment on Unit IV: (Use java/Python)

Title: Study of UCP socket programming for wired

OBJECTIVES:

1. Getting familiar with the client-server communication model.
2. Learning the most important library functions (the UNIX and Internetsockets) used for the design of the client-server applications.
3. Designing simple client or server applications for stream.

PROBLEM STATEMENT

Write a program using UDP Sockets to enable file transfer (Script, Text, Audio and Video one file each) between two machines. Demonstrate the packets captured traces using Wireshark Packet Analyzer Tool for peer to peer mode.

Theory: -

Program A program is an executable file residing on a disk in a directory. A program is read into memory and is executed by the kernel as ad result of an exec () function. The exec () has six variants, but we only consider the simplest one (exec ()) in this course.

Process An executing instance of a program is called a *process*. Sometimes, *task* is used instead of process with the same meaning. UNIX guarantees that every process has a unique identifier called the *process ID*. The process ID is always a non-negative integer.

File descriptors File descriptors are normally small non-negative integers that the kernel uses to identify the files being accessed by a particular process. Whenever it opens an existing file or creates a new file, the kernel returns a file descriptor that is used to read or write the file. As we will see in this course, sockets are based on a very similar mechanism (socket descriptors).

The client-server model

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

Client and servers communicate by means of multiple layers of network protocols. In this course we will focus on the TCP/IP protocol suite.

The scenario of the client and the server on the same local network (usually called LAN, Local Area Network) is shown in Figure 1

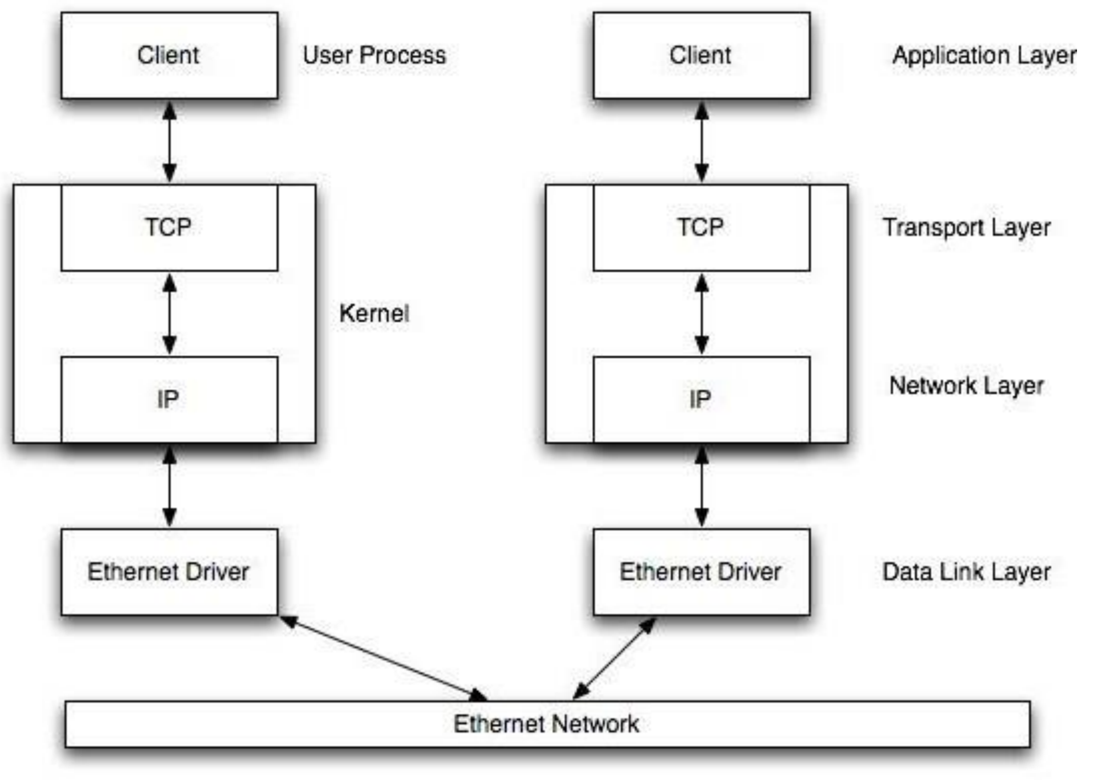


Figure 1: Client and server on the same Ethernet communicating using TCP/IP.

The client and the server may be in different LANs, with both LANs connected to a Wide Area Network (WAN) by means of *routers*. The largest WAN is the Internet, but companies may have their own WANs. This scenario is depicted in Figure 2.

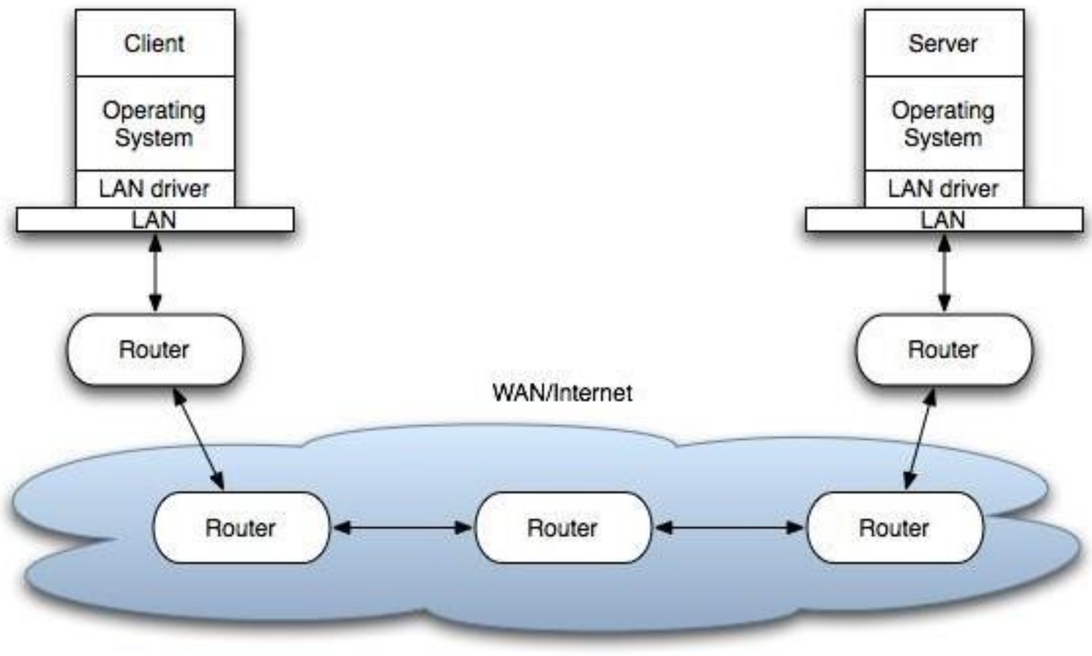


Figure 2: Client and server on different LANs connected through WAN/Internet.

The flow of information between the client and the server goes down the protocol stack on one side, then across the network and then up the protocol stack on the other side.

User Datagram Protocol (UDP):-

UDP is a simple transport-layer protocol. The application writes a message to a UDP socket, which is then encapsulated in a UDP datagram, which is further encapsulated in an IP datagram, which is sent to the destination.

There is no guarantee that a UDP will reach the destination that the order of the datagrams willbe preserved across the network or that datagrams arrive only once.

The problem of UDP is its lack of reliability: if a datagram reaches its final destination but the checksum detects an error, or if the datagram is dropped in the network, it is not automatically retransmitted.

Each UDP datagram is characterized by a length. The length of a datagram is passed to the receiving application along with the data.

No connection is established between the client and the server and, for this reason, we say that UDP provides a *connection-less service*.

Socket addresses

IPv4 socket address structure is named sockaddr_in and is defined by including the <netinet/in.h> header.

The POSIX definition is the following:

```
struct in_addr{
in_addr_t s_addr;                                    /*32 bit IPv4 network byte ordered address*/
};


struct sockaddr_in {
  uint8_t sin_len; /* length of structure (16)*/
  sa_family_t sin_family; /* AF_INET*/
  in_port_t sin_port; /* 16 bit TCP or UDP port number */
  struct in_addr sin_addr; /* 32 bit IPv4 address*/
  char sin_zero[8]; /* not used but always set to zero */
};
```

The uint8_t datatype is unsigned 8-bit integer.

Generic Socket Address Structure: -

A socket address structure is always passed by reference as an argument to any socket functions. But any socket function that takes one of these pointers as an argument must deal with socket address structures from any of the supported protocol families.

A problem arises in declaring the type of pointer that is passed. With ANSI C, the solution is to use void * (the generic pointer type). But the socket functions predate the definition of ANSI C and the solution chosen was to define a generic socket address as follows:

```
struct sockaddr {
  uint8_t sa_len;
  sa_family_t sa_family; /* address family: AD_xxx value */
  char sa_data[14];
};
```

UDP Socket API: -

There are some fundamental differences between TCP and UDP sockets. UDP is a connection-less, unreliable, datagram protocol (TCP is instead connection-oriented, reliable and stream based). There are some instances when it makes to use UDP instead of TCP. Some popular applications built around UDP are DNS, NFS, SNMP and for example, some Skype services and streaming media.

Figure 3 shows the the interaction between a UDP client and server. First of all, the client does not establish a connection with the server. Instead, the client just sends a datagram to the server using the sendto function which requires the address of the destination as a parameter. Similarly, the server does not accept a connection from a client. Instead, the server just callsthe recvfromfunction, which waits until data arrives from some client. recvfrom returns the IP address of the client, along with the datagram, so the server can send a response to the client.

As shown in the Figure, the steps of establishing a UDP socket communication on the client side are as follows:

- Create a socket using the socket() function;
- Send and receive data by means of the recvfrom() and sendto() functions.

The steps of establishing a UDP socket communication on the server side are as follows:

- Create a socket with the socket() function;
- Bind the socket to an address using the bind() function;
- Send and receive data by means of recvfrom() and sendto().

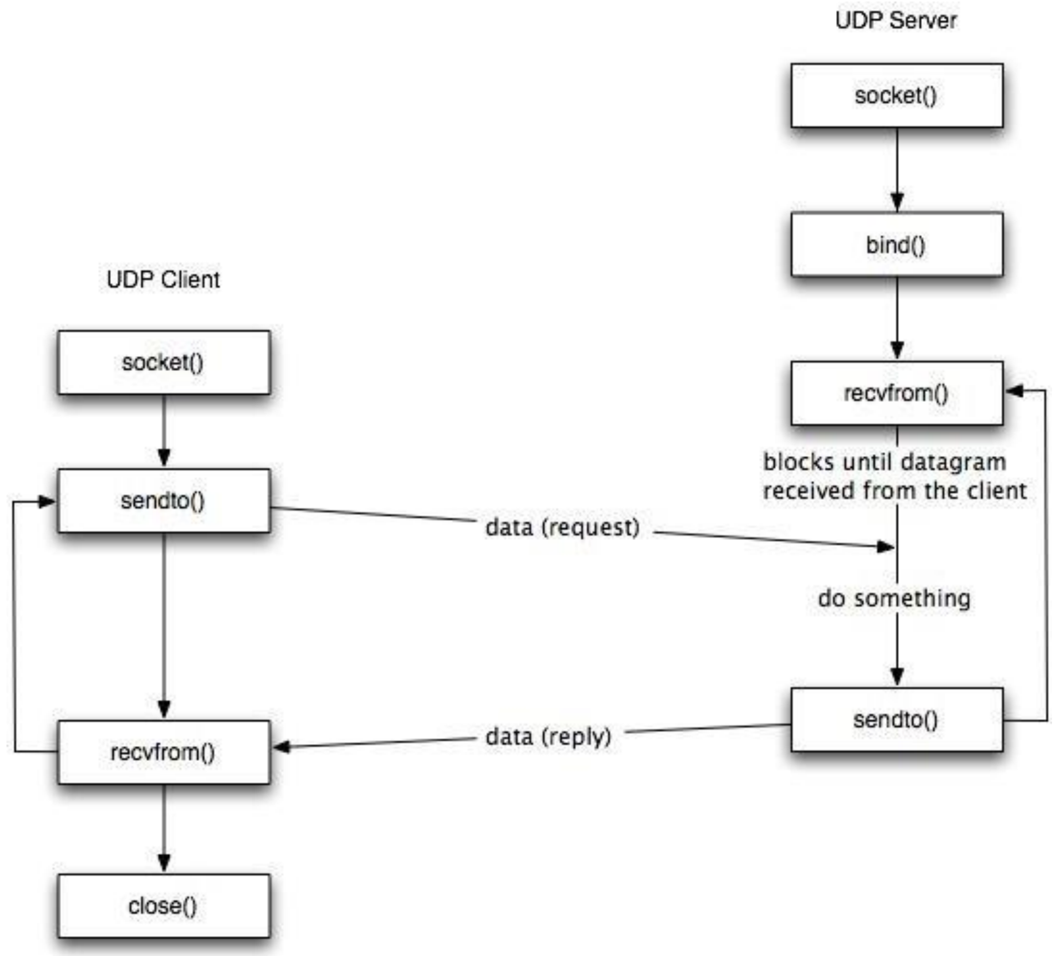


Figure 3: UDP client-server.

The socket () Function

The first step is to call the socket function, specifying the type of communication protocol (TCP based on IPv4, TCP based on IPv6, UDP).

The function is defined as follows:

```
#include <sys/socket.h>

int socket (int family, int type, int protocol);
```

where family specifies the protocol family (AF_INET for the IPv4 protocols), type is a constant described the type of socket (SOCK_STREAM for stream sockets and SOCK_DGRAM for datagram sockets.

The function returns a non-negative integer number, similar to a file descriptor, that we define *socket descriptor* or -1 on error.

The bind() Function

The bind() assigns a local protocol address to a socket. With the Internet protocols, the address is the combination of an IPv4 or IPv6 address (32-bit or 128-bit) address along with a 16 bit TCP port number.

The function is defined as follows:

```
#include <sys/socket.h>

int bind(int sockfd, const struct sockaddr *servaddr, socklen_t addrlen);
```

where sockfd is the socket descriptor, servaddr is a pointer to a protocol-specific address and addrlen is the size of the address structure.

bind() returns 0 if it succeeds, -1 on error.

This use of the generic socket address sockaddr requires that any calls to these functions must cast the pointer to the protocol-specific address structure. For example for and IPv4 socket structure:

```
struct sockaddr_in serv; /* IPv4 socket address structure */

bind(sockfd, (struct sockaddr*) &serv, sizeof(serv))
```

A process can bind a specific IP address to its socket: for a TCP client, this assigns the source IP address that will be used for IP datagrams sent on the sockets. For a TCP server, this restricts the socket to receive incoming client connections destined only to that IP address.

Normally, a TCP client does not bind an IP address to its socket. The kernel chooses the source IP socket is connected, based on the outgoing interface that is used. If a TCP server does not bindan IP address to its socket, the kernel uses the destination IP address of the incoming packets as the server's source address.

bind() allows to specify the IP address, the port, both or neither.

The table below summarizes the combinations for IPv4.

IP Address	**IP Port**	**Result**
INADDR_ANY	0	Kernel chooses IP address and port
INADDR_ANY	non zero	Kernel chooses IP address, process specifies port
Local IP address	0	Process specifies IP address, kernel chooses port
Local IP address	non zero	Process specifies IP address and port

Note, the local host address is 127.0.0.1; for example, if you wanted to run your echoServer (see later) on your local machine the your client would connect to 127.0.0.1 with the suitable port.

The recvfrom() Function

This function is similar to the read() function, but three additional arguments are required. The recvfrom() function is defined as follows:

```
#include <sys/socket.h>

ssize_t recvfrom(int sockfd, void* buff, size_t nbytes,
                 int flags, struct sockaddr* from,
                 socklen_t *addrlen);
```

The first three arguments sockfd, buff, and nbytes, are identical to the first three arguments of read and write. sockfd is the socket descriptor, buff is the pointer to read into, and nbytes is number of bytes to read. In our examples we will set all the values of the flags argument to 0. The recvfrom function fills in the socket address structure pointed to by from with the protocol address of who sent the datagram. The number of bytes stored in the socket address structure is returned in the integer pointed by addrlen.

The function returns the number of bytes read if it succeeds, -1 on error.

The sendto() Function

This function is similar to the send() function, but three additional arguments are required.The sendto() function is defined as follows:

```
#include <sys/socket.h>
ssize_t sendto(int sockfd, const void *buff, size_t nbytes,
                    int flags, const struct sockaddr *to,
                    socklen_t addrlen);
```

The first three arguments sockfd, buff, and nbytes, are identical to the first three arguments of recv. sockfd is the socket descriptor, buff is the pointer to write from, and nbytes is number of bytes to write. In our examples we will set all the values of the flags argument to 0. The to argument is a socket address structure containing the protocol address (e.g., IP address and port number) of where the data is sent. addlen specified the size of this socket.

The function returns the number of bytes written if it succeeds, -1 on error.

The close() Function

The normal close() function is used to close a socket and terminate a TCP socket. It returns 0 if itsucceeds, -1 on error. It is defined as follows:

```
#include <unistd.h>

int close(int sockfd);
```

FAQS

Please refer the above theory for answers:

1. What is Socket?
2. What are different types of Socket?
3. What are different Socket functions or calls used in UDP Client Servercommunication?
4. Explain UCP protocol in details?

Conclusion:

Hence we studied and implement program to demonstrate UCP Socket programming forwired network.

Experiment Number: C1

Lab Assignment on Unit VI: (Use JAVA/PYTHON)

Title: Study of DNS Lookup

OBJECTIVES:

1. To learn and understand DNS lookup.
2. To learn and Understand concept of IP protocol

PROBLEM STATEMENT

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

Theory: -

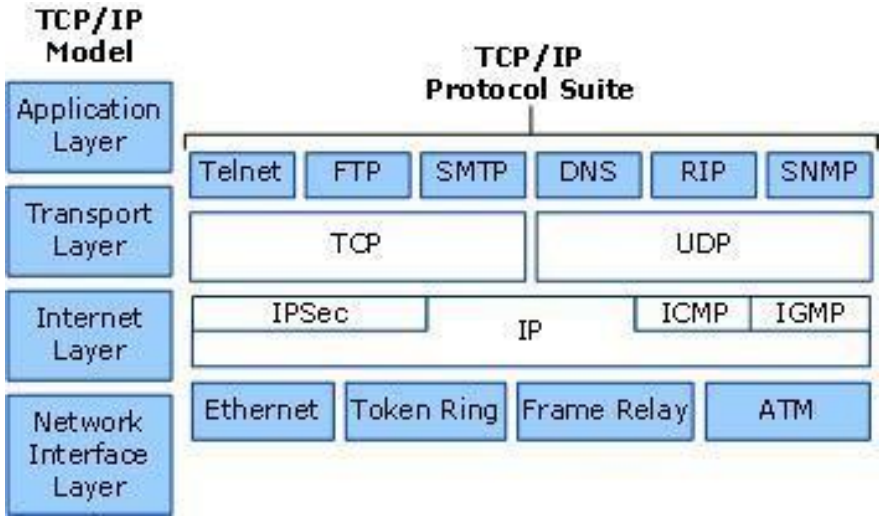
What is DNS?

The Domain Name System (DNS) is the phonebook of the Internet. Humans access information online through domain names, like nytimes.com or espn.com. Webbrowsers interact through Internet Protocol (IP) addresses. DNS translates domain names to IP addresses so browsers can load Internet resources.

Each device connected to the Internet has a unique IP address which other machines use to find the device. DNS servers eliminate the need for humans to memorize IP addresses such as 192.168.1.1 (in IPv4), or more complex newer alphanumeric IP addresses such as 2400:cb00:2048:1::c629:d7a2 (in IPv6).

The Windows Server 2003 DNS Server and Client services use the DNS protocol that is included in the TCP/IP protocol suite. DNS is part of the application layer of the TCP/IP reference model.

DNS in TCP/IP



Technologies That Use DNS

DNS and Active Directory

Windows Server 2003 Active Directory directory service uses DNS as its domain controller location mechanism. When any of the principal Active Directory operations is performed, such as authentication, updating, or searching, Windows Server 2003 computers use DNS to locate Active Directory domain controllers and these domain controllers use DNS to locate each other. For example, when a networkuser with an Active Directory user account logs in to an Active Directory domain, the user's computer uses DNS to locate a domain controller for the Active Directory domain to which the user wants to log in.

DNS and WINS

The earlier method of name resolution for a Windows network was Windows Internet Name Service (WINS). DNS is different than WINS in that DNS is a hierarchical namespace and WINS is a flat namespace. Down-level clients and applications that rely on NetBIOS names continue to use WINS for name resolution. Since Windows Server 2003 DNS is WINS-aware, a combination of both DNS and WINS can be used in a mixed environment to achieve maximum efficiency in locating various network services and resources.

DNS and DHCP

For Windows Server 2003 DNS, the DHCP service provides default support to register and update information for legacy DHCP clients in DNS zones. Legacy clients typically include other Microsoft TCP/IP client computers that were released prior to Windows 2000. The Windows Server 2003 DNS-DHCP integration enablesa DHCP client that is unable to dynamically update DNS resource records directly to have this information updated in DNS forward and reverse lookup zones by the DHCP server.

Domain Name System (DNS) is the default name resolution service used in a Microsoft Windows Server 2003 network. DNS is part of the Windows Server 2003 TCP/IP protocol suite and all TCP/IP network connections are, by default, configured with the IP address of at least one DNS server in order to perform name resolution on the network. Windows Server 2003 components that require name resolution will attempt to use this DNS server before attempting to use the previous default Windows name resolution service, Windows Internet Name Service (WINS).

Typically, Windows Server 2003 DNS is deployed in support of Active Directory directory service. In this environment, DNS namespaces mirror the Active Directory forests and domains used by an organization. Network hosts and services are configured with DNS names so that they can be located in the network, and they are also configured with DNS servers that resolve the names of Active Directory domain controllers.

Windows Server 2003 DNS is also commonly deployed as a non-Active Directory,or standard, Domain Name System solution, for the purposes of hosting the Internet presence of an organization, for example.

DNS Architecture

DNS architecture is a hierarchical distributed database and an associated set of protocols that define:

- A mechanism for querying and updating the database.
- A mechanism for replicating the information in the database among servers.
- A schema of the database.

DNS originated in the early days of the Internet when the Internet was a small network established by the United States Department of Defense for research purposes. The host names of the computers in this network were managed through the use of a single HOSTS file located on a centrally administered server. Each site that needed to resolve host names on the network downloaded this file. As the number of hosts on the Internet grew, the traffic generated by the update process increased, as well as the size of the HOSTS file. The need for a new system, which would offer features such as scalability, decentralized administration, support for various data types, became more and more obvious.

DNS Domain Names

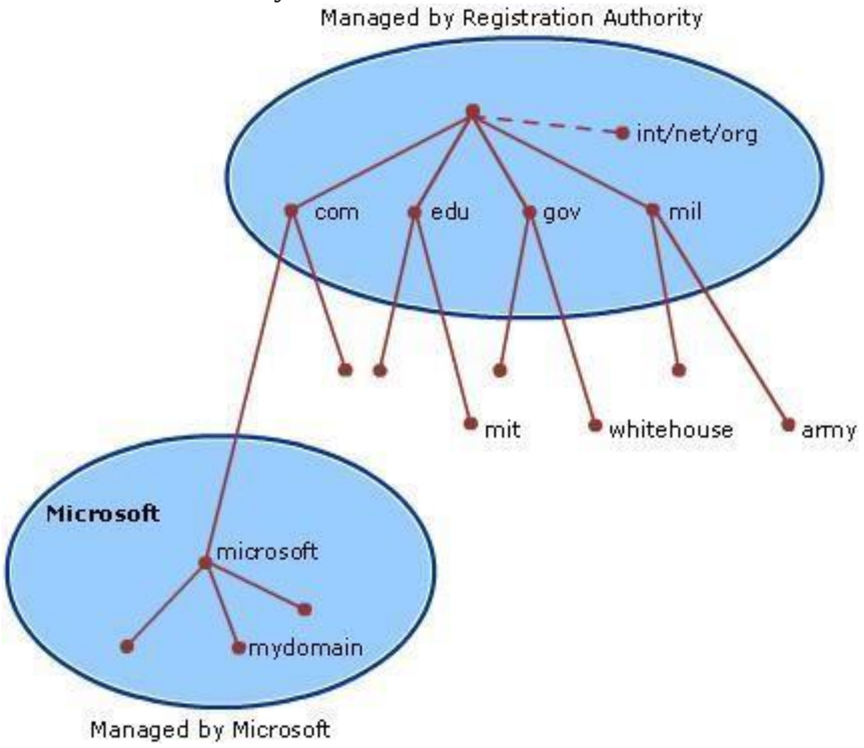
The Domain Name System is implemented as a hierarchical and distributed database containing various types of data, including host names and domain names. The names in a DNS database form a hierarchical tree structure called the domain namespace. Domain names consist of individual labels separated by dots, for example: mydomain.microsoft.com.

A Fully Qualified Domain Name (FQDN) uniquely identifies the hosts position within the DNS hierarchical tree by specifying a list of names separated by dots inthe path from the referenced host to the root. The next figure shows an example of a DNS tree with a host called my domain within the microsoft.com. domain. The FQDN for the host would be mydomain.microsoft.com**.**

Understanding the DNS Domain Namespace

The DNS domain namespace, as shown in the following figure, is based on the concept of a tree of named domains. Each level of the tree can represent either a branch or a leaf of the tree. A branch is a level where more than one name is used to identify a collection of named resources. A leaf represents a single name used once atthat level to indicate a specific resource.

DNS Domain Name Hierarchy



The previous figure shows how Microsoft is assigned authority by the Internet root servers for its own part of the DNS domain namespace tree on the Internet. DNS clients and servers use queries as the fundamental method of resolving names in the tree to specific types of resource information. This information is provided by DNS servers in query responses to DNS clients, who then extract the information and pass it to a requesting program for resolving the queried name. In the process of resolving a name, keep in mind that DNS servers often function as DNS clients, querying otherservers in order to fully resolve a queried name.

How the DNS Domain Namespace Is Organized

Any DNS domain name used in the tree is technically a domain. Most DNS discussions, however, identify names in one of five ways, based on the level and the way a name is commonly used. For example, the DNS domain name registered to Microsoft

(microsoft.com.) is known as a second-level domain. This is because the name has two parts (known as labels) that indicate it is located two levels below the root or top of the tree. Most DNS domain names have two or more labels, each of which indicates a new level in the tree. Periods are used in names to separate labels. The five categories used to describe DNS domain names by their function in the namespace are described in the following table, along with an example of each name type.

Types of DNS Domain Names

Name Type	Description	Example
Root domain	This is the top of the tree, representing an unnamed level; it is sometimes shown as two empty quotation marks (""), indicating a null value. When used in a DNS domain name, it is stated by a trailing period (.) to designate that the name is located at the root or highest level of the domain hierarchy. In this instance, the DNS domain name is considered to be complete and points to an exact location in the tree of names. Names stated this way are called fully qualified domain names (FQDNs).	A single period (.) or a period Used at the end of a name, such as “example.microsoft.com.”

Top level domain	A name used to indicate a country/region or the type of organization using a name.	"com", which indicates a Name registered to a business for commercial use on the Internet.
Second level domain	Variable-length names registered to an individual or organization for use on the Internet. These names are always based upon an appropriate top-level domain, depending on the type of organization or geographic location where a name is used.	"microsoft.com. ", which is the second-level domain Name registered to Microsoft by the Internet DNS Domain name registrar.
Subdomain	Additional names that an organization can create that are derived from the registered second-level domain name. These include names added to grow the DNS tree of names in an organization and divide it into departments or geographic locations.	"""example.microsoft.com. ", which is a fictitious subdomain assigned by Microsoft for use in documentation example names.

Conclusion: - Hence we studied DNS in detail.

Experiment Number: C2

Lab Assignment on Unit VI:

Title: Study of DHCP.

OBJECTIVES:

- Student should be able to understand Dynamic Host Configuration Protocol.

PROBLEM STATEMENT

Installing and configure DHCP server and write a program to install the software on remote machine.

Theory: -

What is DHCP?

Dynamic Host Configuration Protocol (DHCP) is a client/server protocol that automatically provides an Internet Protocol (IP) host with its IP address and other related configuration information such as the subnet mask and default gateway. RFCs 2131 and 2132 define DHCP as an Internet Engineering Task Force (IETF) standard based on Bootstrap Protocol (BOOTP), a protocol with which DHCP shares many implementation details. DHCP allows hosts to obtain necessary TCP/IP configuration information from a DHCP server.

The Microsoft Windows Server 2003 operating system includes a DHCP Server service, which is an optional networking component. All Windows-based clients include the DHCP client as part of TCP/IP, including Windows Server 2003, Microsoft Windows XP, Windows 2000, Windows NT 4.0, Windows Millennium Edition (Windows Me), and Windows 98.

Benefits of DHCP

In Windows Server 2003, the DHCP Server service provides the following benefits:

- **Reliable IP address configuration.** DHCP minimizes configuration errors caused by manual IP address configuration, such as typographical errors, or address conflicts caused by the assignment of an IP address to more than one computer at the same time.
- **Reduced network administration.** DHCP includes the following features to reduce network administration:
  - Centralized and automated TCP/IP configuration.
  - The ability to define TCP/IP configurations from a central location.
  - The ability to assign a full range of additional TCP/IP configuration values by means of DHCP options.

- The efficient handling of IP address changes for clients that must be updated frequently, such as those for portable computers that move to different locations on a wireless network.

- The forwarding of initial DHCP messages by using a DHCP relay agent, thus eliminating the need to have a DHCP server on every subnet.

Why use DHCP

Every device on a TCP/IP-based network must have a unique unicast IP address to access the network and its resources. Without DHCP, IP addresses must be configured manually for new computers or computers that are moved from one subnet to another, and manually reclaimed for computers that are removed from the network.

DHCP enables this entire process to be automated and managed centrally. The DHCP server maintains a pool of IP addresses and leases an address to any DHCP-enabled client when it starts up on the network. Because the IP addresses are dynamic (leased) rather than static (permanently assigned), addresses no longer in use are automatically returned to the pool for reallocation.

The network administrator establishes DHCP servers that maintain TCP/IP configuration information and provide address configuration to DHCP-enabled clients in the form of a lease offer. The DHCP server stores the configuration information in a database, which includes:

- Valid TCP/IP configuration parameters for all clients on the network.

- Valid IP addresses, maintained in a pool for assignment to clients, as well as excluded addresses.

- Reserved IP addresses associated with particular DHCP clients. This allows consistent assignment of a single IP address to a single DHCP client.

- The lease duration, or the length of time for which the IP address can be used before a lease renewal is required.

ADHCP-enabled client, upon accepting a lease offer, receives:

- A valid IP address for the subnet to which it is connecting.

- Requested DHCP options, which are additional parameters that a DHCP server is configured to assign to clients. Some examples of DHCP options are Router (default gateway), DNS Servers, and DNS Domain Name. Terms and Definitions.

The following table lists common terms associated with DHCP.

DHCP Terms and Definitions

Term	Definition
DHCP server	A computer running the DHCP Server service that holds information about available IP addresses and related configuration information as defined by the DHCP administrator and responds to requests from DHCP clients.
DHCP client	A computer that gets its IP configuration information by using DHCP.
Scope	A range of IP addresses that are available to be leased to DHCP clients by the DHCP Server service.
Subnetting	The process of partitioning a single TCP/IP network into a number of separate network segments called subnets.
DHCP option	Configuration parameters that a DHCP server assigns to clients. Most DHCP options are predefined, based on optional parameters defined in Request for Comments (RFC) 2132, although extended options can be added by vendors or users.
Option class	An additional set of options that can be provided to a DHCP client based on its computer class membership. The administrator can use option classes to submanage option values provided to DHCP clients. There are two types of options classes supported by a DHCP server running Windows Server 2003: vendor classes and user classes.
Lease	The length of time for which a DHCP client can use a DHCP-assigned IP address configuration.
Reservation	A specific IP address within a scope permanently set aside for leased use by a specific DHCP client. Client reservations are made in the DHCP database using the DHCP snap-in and are based on a unique client device identifier for each reserved entry.

Exclusion/exclusion range	One or more IP addresses within a DHCP scope that are not allocated by the DHCP Server service. Exclusions ensure that the specified IP addresses will not be offered to clients by the DHCP server as part of the general address pool.
DHCP relay agent	Either a host or an IP router that listens for DHCP client messages being broadcast on a subnet and then forwards those DHCP messages directly to a configured DHCP server. The DHCP server sends DHCP response messages directly back to the DHCP relay agent, which then forwards them to the DHCP client. The DHCP administrator uses DHCP relay agents to centralize DHCP servers, avoiding the need for a DHCP server on each subnet. Also referred to as a *BOOTP relay agent*.
Unauthorized DHCP server	A DHCP server that has not explicitly been authorized. Sometimes referred to as a *rogue DHCP server*. In a Windows Server 2003 domain environment, the DHCP Server service on an unauthorized server running Windows Server 2003 fails to initialize. The administrator must explicitly authorize all DHCP servers running Windows Server 2003 that operate in an Active Directory service domain environment. At initialization time, the DHCP Server service in Windows Server 2003 checks for authorization and stops itself if the server detects that it is in a
	domain environment and the server has not been explicitly authorized.
Automatic Private IP Addressing (APIPA)	A TCP/IP feature in Windows XP and Windows Server 2003 that automatically configures a unique IP address from the range 169.254.0.1 through 169.254.255.254 with a subnet mask of 255.255.0.0 when the TCP/IP protocol is configured for automatic addressing, the **Automatic private IP address** alternate configuration setting is selected, and a DHCP server is not available. The APIPA range of IP addresses is reserved by the Internet Assigned Numbers Authority (IANA) for use on a single subnet, and IP addresses within this range are not used on the Internet.
Superscope	A configuration that allows a DHCP server to provide leases from more than one scope to clients on a single physical network segment.

Multicast IP addresses	Multicast IP addresses allow multiple clients to receive data that is sent to a single IP address, enabling point-to-multipoint communication. This type of transmission is often used for streaming media transmissions, such as video conferencing.
Multicast Scope	A range of multicast IP addresses that can be assigned to DHCP clients. A multicast scope allows dynamic allocation of multicast IP addresses for use on the network by using the MADCAP protocol, as defined in RFC 2730.
BOOTP	An older protocol with similar functionality; DHCP is based on BOOTP. BOOTP is an established protocol standard used for configuring IP hosts. BOOTP was originally designed to enable boot configuration for diskless workstations. Most DHCP servers, including those running Windows Server 2003, can be configured to respond to both BOOTP requests and DHCP requests.

- DHCP Architecture
- DHCP Protocols
- DHCP Processes and Interactions

DHCP Architecture

The DHCP architecture consists of DHCP clients, DHCP servers, and DHCP relay agents on a network. The clients interact with servers using DHCP messages in a DHCP conversation to obtain and renew IP address leases.

DHCP Client Functionality

A **DHCP client** is any network-enabled device that supports the ability to communicate with a DHCP server in compliance with RFC 2131, for the purpose of obtaining dynamic leased IP configuration and related optional information.

DHCP provides support for client computers running any of the following Microsoft operating systems:

- Windows NT version 4.0
- Windows 2000
- Windows XP
- Windows Server 2003
- Windows 98

- Windows Millennium Edition

Automatic IP Configuration

DHCP supports Automatic Private IP Addressing (APIPA), which enables computers running Windows 2000, Windows XP, and Windows Server 2003 to configure an IP address and subnet mask if a DHCP server is unavailable at system startup and the **Automatic private IP address** Alternate Configuration setting is selected. This feature is useful for clients on small private networks, such as a small-business office or a home office.

The DHCP Client service on a computer running Windows XP and Windows Server 2003 uses the following process to auto-configure the client:

1. The DHCP client attempts to locate a DHCP server and obtain an IP address and configuration.

2. If a DHCP server cannot be found or does not respond after one minute, the DHCP client checks the settings on the **Alternate Configuration** tab of the properties of the TCP/IP protocol.

   If **Automatic private IP address** is selected, the DHCP client auto-configures its IP address and subnet mask by using a selected address from the Microsoft-reserved Class B network, 169.254.0.0, with the subnet mask 255.255.0.0. The DHCP client tests for an address conflict to ensure that the IP address is not in use on the network. If a conflict is found, the client selects another IP address. The client retries auto-configuration up to 10 times.

   If **User Configured** is selected, the DHCP client configures a static IP address configuration. The DHCP client tests for an address conflict to ensure that the IP address is not already in use on the network. If a conflict is found, the DHCP client indicates the error condition to the user.

3. When the DHCP client succeeds in self-selecting an address, it configures its network interface with the IP address. The client then continues to check for a DHCP server in the background every five minutes. If a DHCP server responds, the DHCP client abandonsits self-selected IP address and uses the address offered by the DHCP server (and any other DHCP option information that the server provides) to update its IP configuration settings.

If the DHCP client obtained a lease from a DHCP server on a previous occasion, and the lease is still valid (not expired) at system startup, the client tries to renew its lease. If, during the renewal attempt, the client fails to locate any DHCP server, it attempts to ping the default gateway listed in the lease, and proceeds in one of the following ways:

- If the ping is successful, the DHCP client assumes that it is still located on the same network where it obtained its current lease, and continues to use the lease as long as the lease is still valid. By default the client then attempts, in the background, to renew its lease when 50 percent of its assigned lease time has expired.

- If the ping fails, the DHCP client assumes that it has been moved to a network where a DHCP server is not available. The client then auto-configures its IP address by using the settings on the **Alternate Configuration** tab. When the client is auto-configured, it attempts to locate a DHCP server and obtain a lease every five minutes.

Interactions between Client and Server

DHCP servers and DHCP clients communicate through a series of DHCP messages. To obtain a lease, the DHCP client initiates a conversation with a DHCP server using a series of these DHCP messages.

DHCP Messages

The following list includes the eight types of messages that can be sent between DHCP clients and servers. For more information about the structure and specifics of each of these packets, see "DHCP Message Format" later in this section.

DHCP Discover

Broadcast by a DHCP client when it first attempts to connect to the network. The DHCP Discover message requests IP address information from a DHCP server.

DHCP Offer

Broadcast by each DHCP server that receives the client DHCP Discover message and has an IP address configuration to offer to the client. The DHCP Offer message contains an unleased IP address and additional TCP/IP configuration information, such as the subnet mask and default gateway. More than one DHCP server can respond with a DHCP Offer message. The client accepts the best offer, which for a Windows DHCP client is the first DHCP Offer message that itreceives.

DHCP Request

Broadcast by a DHCP client after it selects a DHCP Offer. The DHCP Request message contains the IP address from the DHCP Offer that it selected. If the client is renewing or rebinding to a previous lease, this packet might be unicast directly to the server.

DHCP Ack

Broadcast by a DHCP server to a DHCP client acknowledging the DHCP Request message. At this time, the server also forwards any options. Upon receipt of the DHC PAck, the client can use the leased IP address to participate in the TCP/IP network and complete its system startup. This message is typically broadcast, because the DHCP client does not officially have an IP address that it can use at this point. If the DHCPAck is in response to a DHCPInform, then the message is unicast directly to the host that sent the DHCPInform message.

DHCPNack

Broadcast by a DHCP server to a DHCP client denying the client's DHCPRequest message. This might occur if the requested address is incorrect because the client moved to a new subnet or because the DHCP client's lease has expired and cannot be renewed.

DHCPDecline

Broadcast by a DHCP client to a DHCP server, informing the server that the offered IP address is declined because it appears to be in use by another computer.

DHCPRelease

Sent by a DHCP client to a DHCP server, relinquishing an IP address and canceling the remaining lease. This is unicast to the server that provided the lease.

DHCPInform

Sent from a DHCP client to a DHCP server, asking only for additional local configuration parameters; the client already has a configured IP address. This message type is also used by DHCP servers running Windows Server 2003 to detect unauthorized DHCP servers.

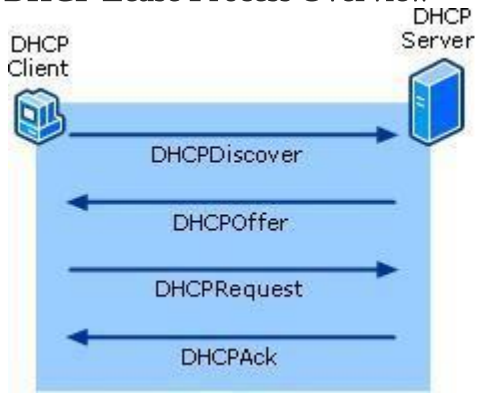
DHCP Lease Process

A DHCP-enabled client obtains a lease for an IP address from a DHCP server. Before the lease expires, the DHCP client must renew the lease or obtain a new lease. Leases are retained in the DHCP server database for a period of time after expiration. By default, this grace period is four hours and cleanup occurs once an hour for a DHCP server running Windows Server 2003. This protects a clients lease in case the client and server are in different time zones, the internal clocks of the client and server computers are not synchronized, or the client is off the network when the lease expires.

Obtaining a New Lease

A DHCP client initiates a conversation with a DHCP server when it is seeking a new lease, renewing a lease, rebinding, or restarting. The DHCP conversation consists of a series of DHCP messages passed between the DHCP client and DHCP servers. The following figure shows an overview of this process when the DHCP server and DHCP client are on the same subnet.

DHCP Lease Process Overview



1. The DHCP client requests an IP address by broadcasting a DHCPDiscover message to the local subnet.

2. The client is offered an address when a DHCP server responds with a DHCPOffer message containing an IP address and configuration information for lease to the client. If no DHCP server responds to the client request, the client sends DHCPDiscover messages at intervals of 0, 4, 8, 16, and 32 seconds, plus a random interval of between -1 second and 1 second. If there is no response from a DHCP server after one minute, the client can proceed in one of two ways:

   - If the client is using the Automatic Private IP Addressing (APIPA) alternate configuration, the client self-configures an IP address for its interface.

   - If the client does not support alternate configuration, such as APIPA, or if IP auto-configuration has been disabled, the client network initialization fails.

   In both cases, the client begins a new cycle of DHCPDiscover messages in the background every five minutes, using the same intervals as before (0, 4, 8, 16, and 32 seconds), until it receives a DHCPOffer message from a DHCP server.

3. The client indicates acceptance of the offer by selecting the offered address and broadcasting a DHCP Request message in response.

4. The client is assigned the address and the DHCP server broadcasts a DHCP Ack messagein response, finalizing the terms of the lease.

<u>Conclusion: -</u> Hence we studied DHCP in detail

PROBLEM STATEMENT

Title: To study steps S/MIME email security through Microsoft® Office Outlook

Objective: Students should be able to understand S/MIME email security through Microsoft® Office Outlook

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

S/MIME (Secure/Multipurpose Internet Mail Extensions) is a standard for securely sending and receiving emails with encryption and digital signatures. It helps protect the confidentiality and integrity of email communication.

Here's a basic guide to implementing S/MIME email security through Microsoft Office Outlook:

1. Obtain a Digital Certificate**:**

- Before you can start using S/MIME, you need a digital certificate. You can obtain one from a trusted certificate authority (CA) or through your organization's IT department.

2. **Install the Digital Certificate:**

- Once you have the digital certificate, you need to install it in Outlook. This typically involves importing the certificate file into your Windows certificate store.

3. **Configure Outlook for S/MIME:**

- Open Microsoft Office Outlook.
- Go to the "File" tab or the "Office Account" section, depending on your version of Outlook.
- Click on "Options" or "Account Settings."
- In the Account Settings window, select your email account and click on "Change" or "More Settings."
- Look for the "Security" or "Security Settings" tab.
- Enable S/MIME by selecting the option to "Add digital signature to outgoing messages" and "Encrypt contents and attachments for outgoing messages."
- Choose the digital certificate you installed earlier for signing and encryption.

4. **Set Default Encryption and Signature Options:**

- While still in the Security settings, you can set default options for signing and encrypting emails.
- Configure whether to sign and encrypt messages by default, or allow manual control for each email.

Compose and Send Secure Emails:

- When composing a new email, you'll see options to digitally sign and encrypt the message. These options are usually in the toolbar or ribbon at the top of the email composition window.
- You can choose to sign, encrypt, or both, depending on your preferences and the sensitivity of the message.

5. **Receiving Secure Emails:**

- When you receive an encrypted email, Outlook will automatically decrypt it if you have the matching private key for the certificate used by the sender.
- Digitally signed emails will show a signature icon, indicating that the email has not been tampered with.

6. **Verifying Signatures:**

- You can right-click on a digitally signed email and choose to verify the digital signature to ensure the authenticity of the sender.

Remember that S/MIME requires coordination between you and your recipients. Both parties need to have valid digital certificates and properly configured email clients to fully utilize the security features.

Please consult the documentation specific to your version of Outlook for detailed step-by-step instructions, as the user interface and options may vary.

Conclusion: Thus Studied study steps S/MIME email security through Microsoft® Office Outlook**.**

Experiment Number: C4
Lab Assignment on Unit VI:

Title: To study the IPsec (ESP and AH) protocol

Objective:

- Students will able 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.

Theory:

Studying the IPsec protocol (Encapsulating Security Payload - ESP and Authentication Header - AH) by capturing packets using Wireshark is a great way to gain practical insights into how the protocol works. Here's a step-by-step guide to capturing and analyzing IPsec packets using Wireshark:

1. **Prepare Your Environment:**
- Set up a network environment where IPsec communication is occurring. This could be a lab environment or a test network.
- Ensure that you have two devices configured to communicate over IPsec: the source (sender) and the destination (receiver) devices.

2. **Install Wireshark:**
- If you haven't already, download and install Wireshark from the official website: https://www.wireshark.org/download.html

3. **Start Capturing Packets:**
- Open Wireshark and select the network interface you want to capture packets on (usually your Ethernet or Wi-Fi adapter).
- Click the "Capture" button to start capturing packets.

4. **Generate IPsec Traffic:**
- Initiate communication between the source and destination devices. This could involve sending encrypted data using IPsec.

5. **Stop Capturing Packets:**
- After generating some IPsec traffic, return to Wireshark and click the "Stop" button to halt packet capture.

6. **Filter IPsec Packets:**
- Use Wireshark's display filters to narrow down the captured packets to only those related to IPsec. Common filter expressions include **esp** for ESP packets and **ah** for AH packets.

7. **Analyze Captured Packets:**
- Select an IPsec packet from the captured list to analyze its details.
- Wireshark will display various layers of the packet, including Ethernet, IP, and the IPsec (ESP/AH) headers.
- You can expand each layer to see the specific fields and values, which can help you understand the structure of the IPsec headers and the data they contain.

8. **Decode Encrypted Data (ESP Only):**
- If you captured ESP packets, Wireshark will not be able to decrypt the actual payload data, as it's encrypted. However, you can analyze the ESP headers and see information about the encrypted traffic.

9. **Examine AH Headers (if captured):**

- If you captured AH packets, analyze the AH headers to understand the authentication and integrity mechanisms.

10. **Learn and Research:**

- To gain a deeper understanding of IPsec, study the IPsec protocol specifications and documentation.
- Research the details of ESP and AH headers, encryption algorithms, authentication methods, and key exchange mechanisms.

11. **Experiment and Repeat:**

- Modify your IPsec configuration, generate different types of traffic, and capture packets again to observe the variations in packet headers and behavior.

Remember that IPsec is a complex protocol, and studying it thoroughly may require a solid understanding of networking, security, and cryptographic concepts. Wireshark is a powerful tool that can help you visualize and analyze the packet-level details of IPsec communication.

Conclusion: Thus studied IPsec (ESP and AH) protocol using wireshark tool.