NEAR EAST UNIVERSITY

Faculty of Engineering

Department of Computer Engineering

NETWORK ROUTING

Graduation Project
COM-400

Student: Mohammed Khillah(20011715)

Supervisor: Assc.Prof.Dr.Rahib Abiyev

Nicosia- 2005

NEAR EAST UNIVERSITY

Faculty of Engineering

Department of Computer Engineering

NETWORK ROUTING

Graduation Project
COM-400

Student: Mohammed Khillah(20011715)

Supervisor: Assc.Prof.Dr.Rahib Abiyev

Nicosia- 2005

ACKNOWLEDGMENT

First of all I am happy to complete the task which I had given with blessing of God and also I am grateful to all the people in my life who have, supported me, advised me. Taught me and who have always encouraged me to follow my dreams and ambitions.

I wish to thank my supervisor, ***Assoc.Prof.Dr. Rahib Abiyev****, for intellectual support, encouragement, and enthusiasm, which made this project possible, and his patience for correcting both my stylistic and scientific errors.*

And thank my dearest parents who encouraged me to continue beyond my undergraduate studies, to my father who proceeded before me and to my mother who encouraged me along the way.

To all my friends, especially Mr Samir Jabr, Ismail Alhemss(virus),and Murad Al- Zubi for sharing wonderful moments, advice, and for making me feel at home.

And above, I thank God for giving me stamina and courage to achieve my objectives.

MOHAMMED KHILLAH

TABLE OF CONTENTS:

ITRODUCTION

A basic understanding of computer networks is requisite in order to understand the principles of network security, and as the internet is growing,the community has changed from a small tight group of academic users to a loose gathering of people on a global network, so that the moving of information between the groups by the network sharing became a popular way, then, the need to find the optimal paths to rout the information has come to be one of the important topics all over the world of information transportation .

This thesis includes ***four chapters*** covering the main topics related in the following Structure:

Chapter 1, Will discuss the network as whole**:** What is Networking, Why and how did Networking Start, The ISO/OSI Reference Model, Types of Networks, Network Devices, The Internet, Overview of TCP/IP.

Chapter 2, Describes the underlying concepts widely used in routing protocols: What Is Routing, Routing Components, Path Determination, Switching, Routing Algorithms, Routing Metrics, Network Protocols.

Chapter 3, Will discuss the dynamic routing of the information during the internetwork sharing : How Routers Work, Packet Processing Interface to Interface, Routing Information Protocol, Distance Vector, Counting to Infinity, Open Shortest Path First, Dijkstra's Algorithm, Flooding Algorithm, OSPF Protocol, Border Gateway Protocol.

Chapter 4, Will discuss and solve the network optimization problems : What Is Network Optimization, Network Modification Analysis, Measuring Network Application Efficiency, Sizing a Network Communication Link, Network Flow Problem, Network Linear Program, Ensuring that Total Supply Equals Total Demand.

Finally in conclusion the obtained important results for the thesis are given.

CHAPTER ONE

INTRODUCTION TO NETWORKING

1.1 Introduction to Networking

A basic understanding of computer networks is requisite in order to understand the principles of network security. In this section, we will cover some of the foundations of computer networking. Following that, we will take a more in-depth look at Routing`s concepts,the problem of routing in computer network.

1.2 What is Networking

Networking is the interconnection of workstations, peripherals terminals and other devices.

One of the most common types of networks is the Local Area Network or LAN.

In networking, it is possible for different types of computers to communicate.

It is not important what type of computer is used on a network.

It may be a Macintosh computer or a PC or a mainframe.

In networking, what is important is that all the devices speak the same language, or protocol.

1.3 Why and how did Networking Start

Applications written for business helped create the PC industry.

Early computers were standalone devices.

In other words, each computer operated on its own, independently from other computers.

It soon became apparent that this was not an efficient or cost effective way for businesses to operate.

A solution was needed that would successfully address three problems: duplication of equipment and resources.

Inability to communicate efficiently and the lack of network management.

One early solution to these problems was the creation of local area networks, or LANs.

Because they connected workstations, peripherals, terminals, and other devices in a single building, LANs made it possible for businesses using computer technology to efficiently share such things as files and printers.

As the use of computers by businesses grew, however, it soon became apparent that even LANs were not sufficient.

In a LAN system, each department or business was an electronic island.

1.4 Why were networking standards needed

Early development of LANs, MANs, and WANs was chaotic in many ways.

The early 1980s saw tremendous expansion in networking.

As companies realized how much money could be saved and how much they could gain in productivity by using network technology,

They began adding networks and expanding existing networks almost as rapidly as new network technologies and products were introduced.

By the mid-1980s, growing pains from this expansion were being felt.

Because many of the emerging network technologies were built using different hardware and software implementations,

one problem that soon surfaced was that many of the new network technologies were incompatible. Increasingly,

it became difficult for networks using different specifications to communicate with each other

What was needed was a way to move information efficiently and quickly from one business to another.

The solution was the creation of metropolitan area networks, Or MANs, and wide area networks, or WANs.

Because WANs connected networks that served users across a large geographic area, they made it possible for businesses to communicate with each other even though they were geographically distant from each other.

1.5 What model was developed to describe NetworKing

To address the problem, the International Organization for Standardization (ISO) researched networks schemes like DECNET

, SNA, and TCP/IP.

As a result of this research, the ISO recognized there was a need to create a network model that would help vendors create networks that would work compatibly and interoperably with other networks.

The OSI Reference Model, released in 1984, was the descriptive scheme they created. By creating the OSI model, the ISO was providing vendors with a set of standards thus ensuring greater compatibility and interoperability between the various types of network technologies that were being produced by many companies around the world.

1.6 The ISO/OSI Reference Model

The *International Standards Organization* (ISO) *Open Systems Interconnect* (OSI) Reference Model defines seven layers of communications types, and the interfaces among them. (See Figure 1.1) Each layer depends on the services provided by the layer below it, all the way down to the physical network hardware, such as the computer's network interface card, and the wires that connect the cards together.

An easy way to look at this is to compare this model with something we use daily which is the telephone. In order for you and I to talk when we are out of earshot, we need a device like a telephone. (In the ISO/OSI model, this is at the application layer.) The telephones, of course, are useless unless they have the ability to translate the sound into electronic pulses that can be transferred over wire and back again. (These functions are provided in layers below the application layer.) Finally, we get down to the physical connection, both must be plugged into an outlet that is connected to a switch that's part of the telephone system's network of switches.

If preson A places a call to person B, person A picks up the receiver, and dials person B's number. This number specifies which central office to which to send my request, and then which phone from that central office to ring. Once person B answers the phone, they begin talking, and their session has begun. Conceptually, computer networks function exactly the same way.

It isn't important to memorize the ISO/OSI Reference Model's layers; but it is useful to know that they exist, and that each layer can not work without the services provided by the layer below it.

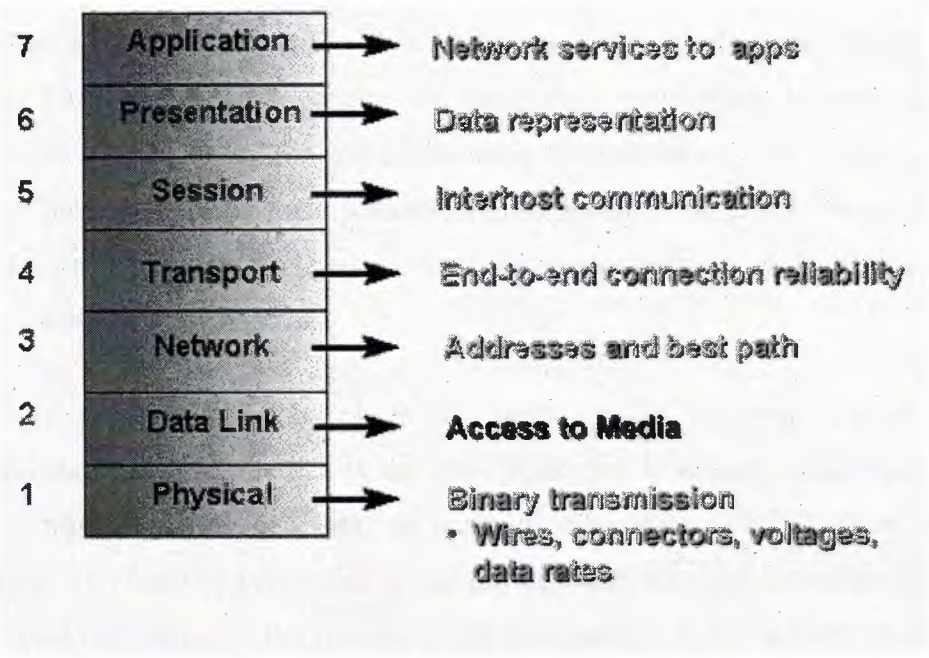


Figure 1.1. OSI Reference Model

The *physical layer* of the model consists of the actual medium through which bits are transmitted from one location to another, in other words, the fabric of the network itself. The connection between two network stations may be in the form of copper or some other electrically conductive cable, fiber optic, radio signals, microwaves, lasers, infrared, or any other medium practically suited to the environment. The OSI model makes no distinctions concerning the actual hardware involved, but the physical layer comprises every component that is needed to realize the connection. This includes any and all connectors, hubs, transceivers, network interfaces, and ancillary hardware, as well as the physical medium or cable itself, if any. This layer also includes the environmental specifications necessary to maintain the validity of the medium, as well as the method of signaling used to transmit bits to a remote location.

The *data link layer* as the interface between the network medium and the higher protocols, the data link layer is responsible for the final packaging of the upper-level binary data into discrete packets before it goes to the physical layer. Its frame is outer most on the packet and contains the basic addressing information that allows it to be transmitted to its destination. The data link layer also controls access to the network medium. This is a crucial element of local area networking because dozens of workstations may be vying for use of the same medium at any one time. Were all of these stations to transmit their packets simultaneously, the result would be chaos. Protocols operating at this layer may also provide other services, such as error checking and correction and flow control.

The *network layer* is where the most crucial dividing line in network communications occurs, for this is the only layer that is actually concerned with the complete transmission of packets, or *protocol data units* (*PDUs*), from source to destination. The functions provided by the physical and data link layers are local. They are designed only to move the packets to the next station on the network medium. The primary task of the network layer is to provide the routing functionality by which packets can be sent across the boundaries of the local network segment to a destination that may be located on an adjacent network or on one thousands of miles away. What's more, the route actually taken by the packet must often be selected from many possible options, based on the relative efficiency of each.

The *transport layer*, as its primary function, provides the balance of the essential services not provided by the network layer protocol. A full-featured CO protocol at the network layer results in a relatively simple transport layer protocol, but as the functionality at the network layer diminishes, the complexity of the transport layer increases. The transport layer's task, therefore, is to provide whatever functions are necessary to elevate the network's *quality of service* (QOS) to a level suitable for the communications required of it

We now arrive at the *session layer* and pass beyond all concerns for transmission reliability, error checking, flow control, and the like. All that can be done in these areas has been done by the time that the transport layer functions have been completed. The session layer is the most misunderstood service in the OSI model, and a

great deal of discussion has gone into the question of whether its functions even warrant a layer of their own. Because of its name, it is often thought (mistakenly) to be concerned with the network logon procedure and related matters of security. The other common description is that it is concerned with matters of "dialogue control and dialogue separation." This is actually true, but more often than not, these expressions are left undefined in such treatments.

Sixth in line, the *presentation layer* acts as the interpreter for network communication. The presentation layer prepares the data for transmission by using one or more of a number of resources, including compression, encryption, or a complete translation of the data into a form more suitable for the currently-implemented communications methods.

Finally, the *application layer*, as the highest of the OSI levels, is tasked with providing the front-end of the computing experience for the user. The application layer is responsible for everything that the user will see, hear, and feel in the course of the networking process-everything from sending and receiving electronic mail, establishing Telnet or FTP sessions, to managing remote network resources.

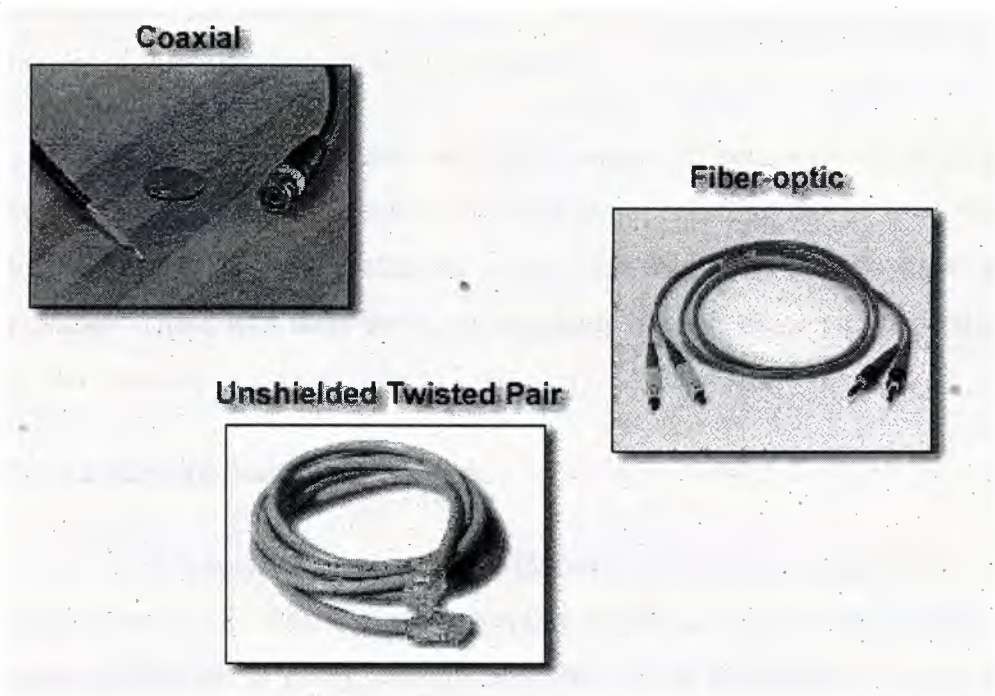


Figure 1.2. Various type of network media

1.7 Types of Networks

In this section some useful categorizations of networks are introduced:

1- Categorization by geographical coverage.

2- Categorization by topology.

1.7.1 Categorization By Geographical Coverage

Depending on the distances signals have to travel different technologies are used to run the connections. That's why it makes sense to distinguish computer networks by the area they cover.

1.7.1.1 Local Area Network (LAN)

A LAN is a network that covers a small area only: a house, a factory site, or a small number of near buildings. It has most often only one owner. However, the size restriction is by area only, and not by number! Large companies can easily have hundreds of workstations in a single LAN.

Hence all the computers are nearby, many different ways of designing the cable connection can be applied, and some methods of cabelling can be used, that would be too expensive for long distances. Local Area Networks usually have a *symmetric topology*. That's why there are many standards (namely those on symmetric topologies as star, ring, bus, etc.) that refer to LANs only.

1.7.1.2 Metropolitan Area Network

A Metropolitan Area Network (MAN) covers larger geographic areas, such as cities or school districts. By interconnecting smaller networks within a large geographic area, information is easily disseminated throughout the network. Local libraries and government agencies often use a MAN to connect to citizens and private industries.

1.7.1.3 Wide Area Network (WAN)

A WAN is a network that covers la large area; typically countries or continents. WANs are used to interconnect LANs over long distances. They usually have an *irregular topology*.

When examining a WAN the main interest is put on *transmission lines* and the *switching elements*, but not on the local "ends" of the WAN. Lines and switches together are called the communication subnet (*short: subnet*); it performs the data exchange in the network.

Besides data exchange in WANs application programs can be run. The machines that do that are referred to as hosts; Hosts perform applications in the network.

1.7.2 Categorization By Topology

1.7.2.1 Bus Topology

A *bus topology*, shown in Figure 1.2, features all networked nodes interconnected peer-to-peer using a single, open-ended cable. These ends must be terminated with a resistive load--that is, *terminating resistors*. This singe cable can support only a single channel. The cable is called the *bus*.

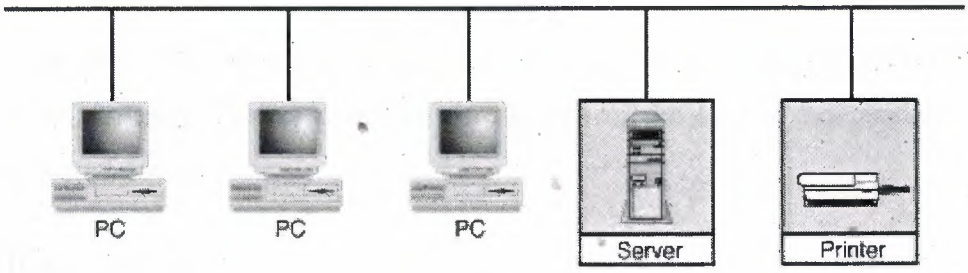


Figure 1.3. Typical bus topology.

The typical bus topology features a single cable, supported by no external electronics, that interconnects all networked nodes peer to peer. All connected devices listen to the bussed transmissions and accept those packets addressed to them. The lack of any external electronics, such as repeaters, makes bus LANs simple and inexpensive.

The downside is that it also imposes severe limitations on distances, functionality, and scaleability.

1.7.2.2 Star Topology

Star topology LANs have connections to networked devices that radiate out from a common point--that is, the *hub*, as shown in Figure 1.3. Unlike ring topologies, physical or virtual, each networked device in a star topology can access the media independently. These devices have to share the hub's available bandwidth. An example of a LAN with a star topology is Ethernet.

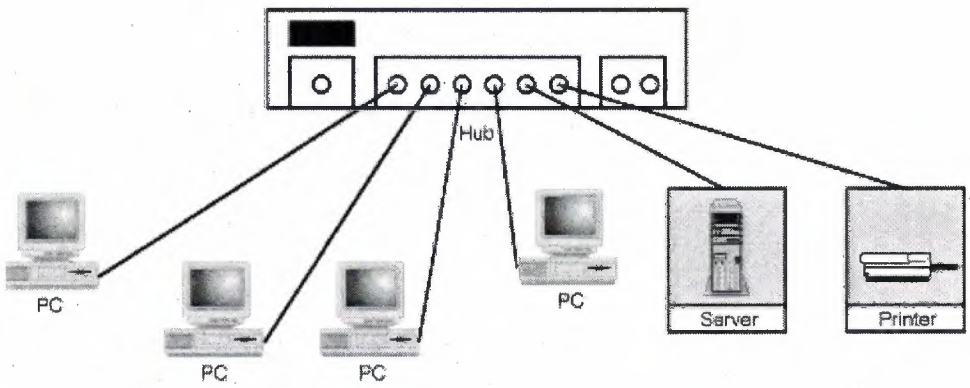


Figure 1.4. Star topology.

A small LAN with a star topology features connections that radiate out from a common point. Each connected device can initiate media access independent of the other connected devices.

1.7.2.3 Ring Topology

The *ring topology* started out as a simple peer-to-peer LAN topology. Each networked workstation had two connections: one to each of its nearest neighbors (see Figure 1.4). The interconnection had to form a physical loop, or ring. Data was transmitted unidirectionally around the ring. Each workstation acted as a repeater, accepting and responding to packets addressed to it, and forwarding on the other packets to the next workstation "downstream."

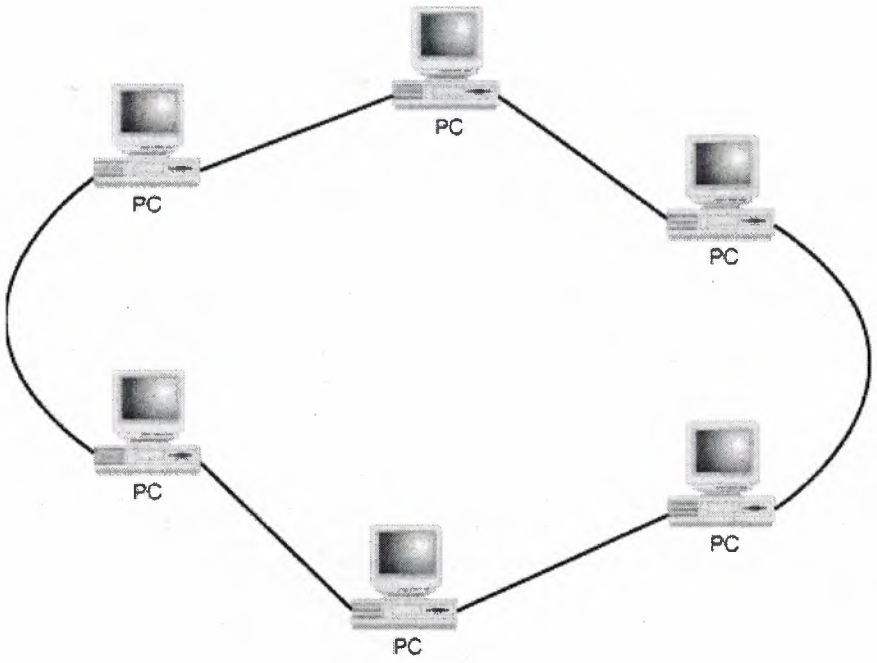


Figure 1.5. Peer-to-peer ring topology.

1.8 Network Devices

Hubs, bridges and routers are getting very intelligent, they have more and more configuration options and are increasingly complex. This is useful for additional features, but the added complexity increases the security risk. On critical subnets, it's important correctly configure network devices: only enable needed services, restrict access to configuration services by port/interface/IP address, disable broadcasts, source routing, choose strong (non default) passwords, enable logging, choose carefully who has user/enable/admin access, etc.

1.8.1 Hub

As its name implies, a *hub* is a center of activity. In more specific network terms, a hub, or concentrator, is a common wiring point for networks that are based around a star topology. Arcnet, 10base-T, and 10base-F, as well as many other

proprietary network topologies, all rely on the use of hubs to connect different cable runs and to distribute data across the various segments of a network (See Figure 1.5.). Hubs basically act as a signal splitter. They take all of the signals they receive in through one port and redistribute it out through all ports. Some hubs actually regenerate weak signals before re-transmitting them. Other hubs retime the signal to provide true synchronous data communication between all ports. Hubs with multiple 10base-F connectors actually use mirrors to split the beam of light among the various ports.

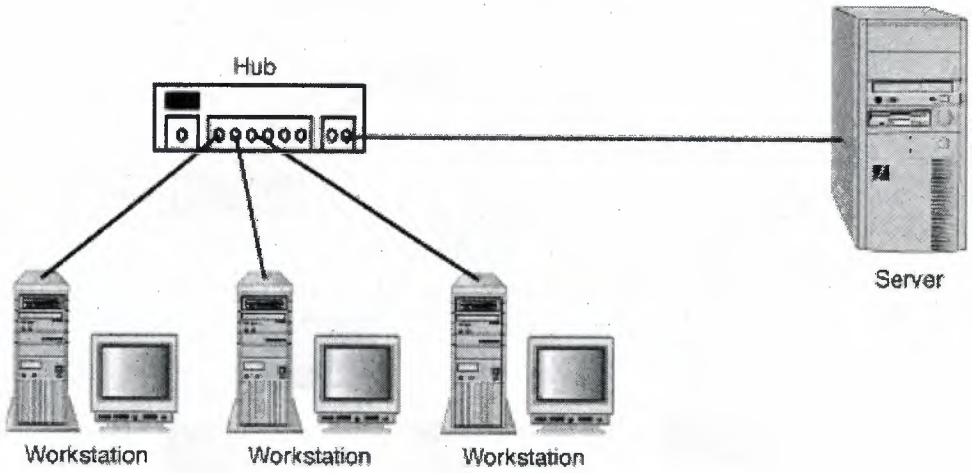


Figure 1.6. A basic diagram of a 10base-T network. Notice the hub, which is the device to which all systems initially connect.

1.8.2 Bridge

A bridge is a device that passes all data on the ethernet, token ring, or whatever type of LAN you have over the WAN to the other LAN which operate at the data link layer, connect two LANs (local area networks) together, and forward frames according to their MAC (media access control) address. Often the concept of a router is more familiar than that of a bridge; it may help to think of a bridge as a "low-level router" (routers operate at the network layer, forwarding by addresses such as an IP address).

A remote bridge connects two remote LANs (bridge 1 and 2 in Figure 1.6) over a link that is normally slow (for example, a telephone line), while a local bridge connects two locally adjacent LANs together (bridge 3 in Figure 1.6). With a local

bridge, performance is an issue, but for a remote bridge, the capability to operate over a long connecting line is often more important.

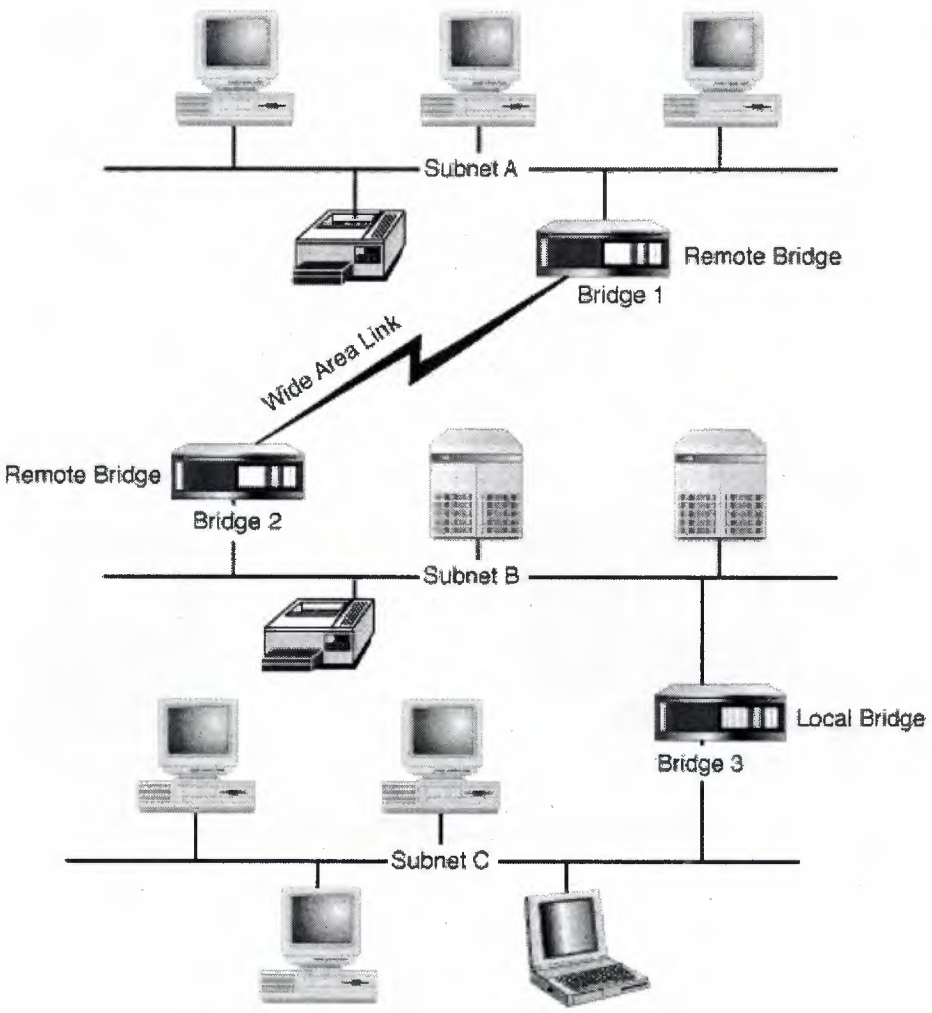


Figure 1.7. A sample network with local and remote bridges.

1.8.3 Router

Routers are devices that are installed on the LAN much as bridges are; a router connects to both the WAN and the LAN. The difference between a router and a bridge is in the way it handles the data it receives. In the bridging world, data bits on the LAN (called packets) are passed across the WAN with minimum effort on the bridge. The bridge doesn't look at the packets very closely to examine the data, because it doesn't care what the data is; it just passes the packets over to the other side of the WAN. Routers, on the other hand, examine the data sent in the packets to see whether it needs

to go over the WAN or if it should stay in the LAN. Think of a data application, e-mail for instance, as if it were a letter being sent over the LAN.

1.9 How does encapsulation allow computer to communicate data

To understand how networks are structured and how they function, you should remember that all communications on a network originate at a source and are being sent to a destination.

The information that is sent on a network is referred to as data or data packets.

If one computer (host A) wants to send data to another computer (host B), the data must first be packaged in a process called encapsulation.

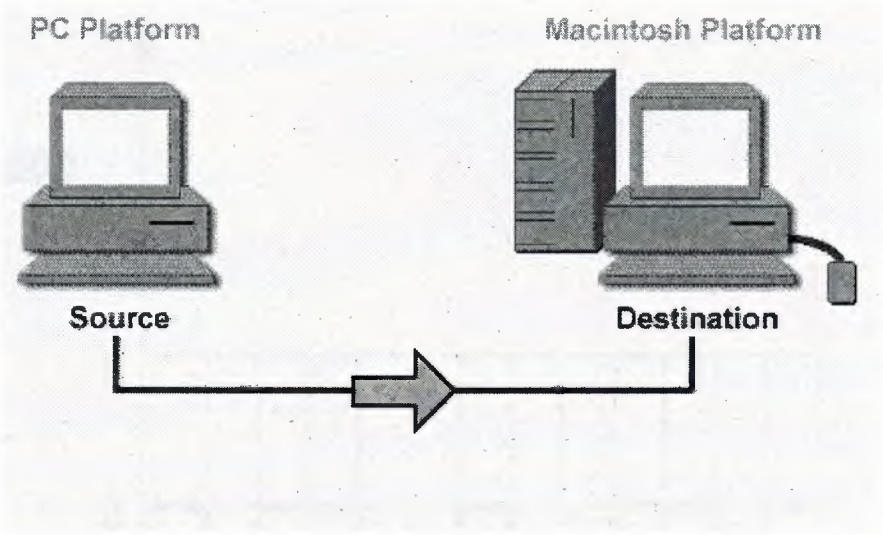


Figure 1.8. Data packet

1.10 How is information stored in Computer

Information in computers is stored using the binary number system, in which the only possible symbols, or binary digits, or "bits", are 1 and 0.

These bits - many of which are called data - are used to represent information, like text, pictures, and sounds.

In the physical layer, a 1 bit is often represented by the presence of voltage (electrical pressure) on a copper conducting cable or light in an optical fiber.

To help you picture these bits, imagine measuring the voltage at one point on the cable as time goes on (for a fiber, imagine measuring the light intensity versus time).

Your measurements would allow you to create a graph of voltage versus time (for a fiber, light intensity versus time).

How the bits (1s and 0s) might be represented on the cable is shown in the graphic.

There are many ways bits can be represented with voltages.

This process is called encoding.

Many of the LANs use "Manchester Encoding."

In this type of encoding bits are represented by different voltage patterns than the ones shown in the graphic.

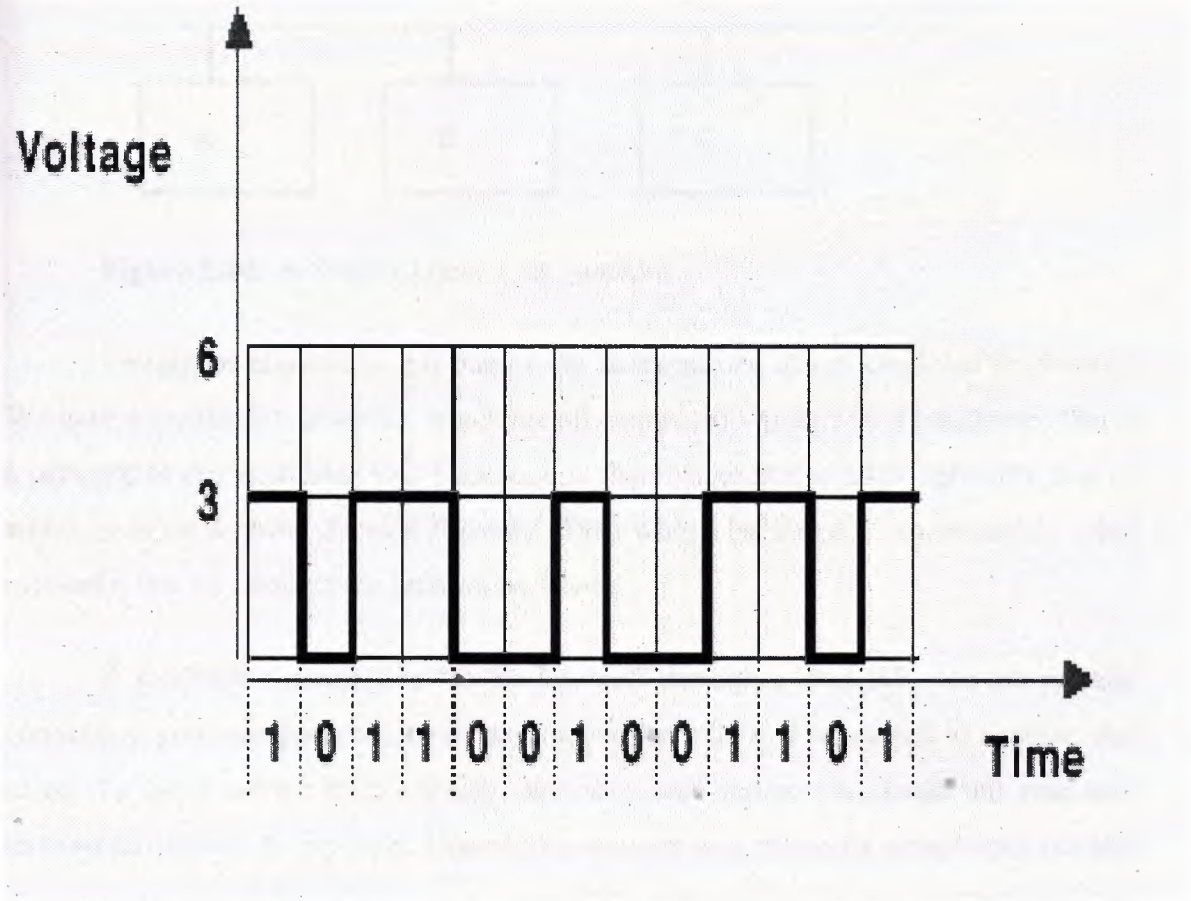


Figure 1.9. Digital signal

1.11 What is The Internet

The Internet is the world's largest network *of networks*. When you want to access the resources offered by the Internet, you do not really connect to the Internet; you connect to a network that is eventually connected to the Internet backbone, a network of extremely fast (and incredibly overloaded!) network components. This is an important point: the Internet is a network of *networks* -- not a network of hosts.

A simple network can be constructed using the same protocols and such that the Internet uses without actually *connecting* it to anything else. Such a basic network is shown in Figure 1.7.

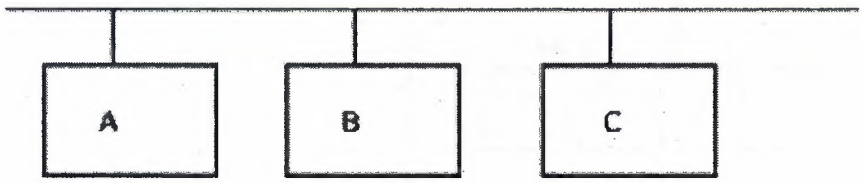


Figure 1.10. A Simple Local Area Network

I might be allowed to put one of my hosts on one of my employer's networks. We have a number of networks, which are all connected together on a backbone , that is a network of our networks. Our backbone is then connected to other networks, one of which is to an *Internet Service Provider* (ISP) whose backbone is connected to other networks, one of which is the Internet backbone.

If you have a connection "to the Internet" through a local ISP, you are actually connecting your computer to one of their networks, which is connected to another, and so on. To use a service from my host, such as a web server, you would tell your web browser to connect to my host. Underlying services and protocols would send *packets* (small datagrams) with your query to your ISP's network, and then a network they are connected to, and so on, until it found a path to my employer's backbone, and to the exact network my host is on. My host would then respond appropriately, and the same would happen in reverse: packets would traverse all of the connections until they found their way back to your computer, and you were looking at my web page.

In Figure 1.8, the network shown in is designated "LAN 1" and shown in the bottom-right of the picture. This shows how the hosts on that network are provided connectivity to other hosts on the same LAN, within the same company, outside of the company, but in the same ISP *cloud* , and then from another ISP somewhere on the Internet.

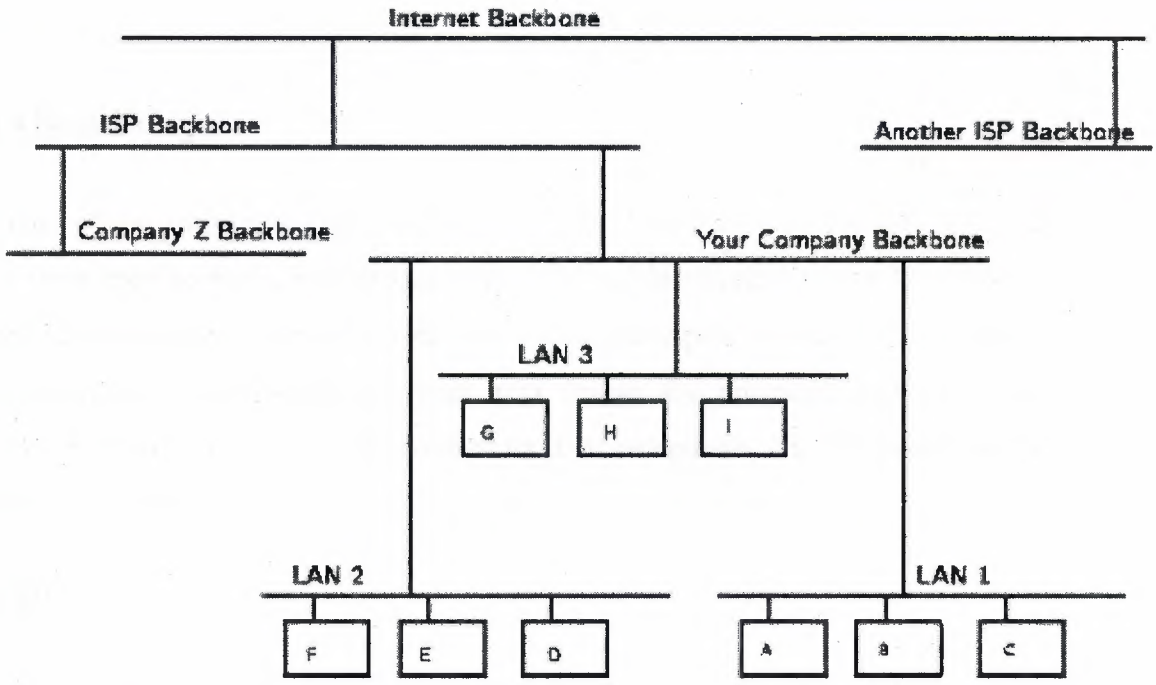


Figure 1.11. A Wider View of Internet-connected Network

The Internet is made up of a wide variety of hosts, from supercomputers to personal computers, including every imaginable type of hardware and software. How do all of these computers understand each other and work together?

1.12 Overview of TCP/IP

TCP/IP (Transport Control Protocol/Internet Protocol) is the language of the Internet. Anything that can learn to speak TCP/IP can play on the Internet. This is functionality that occurs at the Network (IP) and Transport (TCP) layers in the ISO/OSI Reference Model. Consequently, a host that has TCP/IP functionality (such as Unix,

OS/2, MacOS, or Windows NT) can easily support applications (such as Netscape's Navigator) that uses the network.

TCP/IP protocols are not used only on the Internet. They are also widely used to build private networks, called internets, that may or may not be connected to the global Internet. An internet that is used exclusively by one organization is sometimes called an intranet

1.12.1 Open Design

One of the most important features of TCP/IP isn't a technological one: The protocol is an open protocol, and anyone who wishes to implement it may do so freely. Engineers and scientists from all over the world participate in the *IETF* (Internet Engineering Task Force) working groups that design the protocols that make the Internet work. Their time is typically donated by their companies, and the result is work that benefits everyone.

1.12.2 IP

IP is a "network layer" protocol. This is the layer that allows the hosts to actually talk to each other. Such things as carrying datagrams, mapping the Internet address to a physical network address , and routing, which takes care of making sure that all of the devices that have Internet connectivity can find the way to each other.

1.12.2 IP Address

IP addresses are analogous to telephone numbers – when you want to call someone on the telephone, you must first know their telephone number. Similarly, when a computer on the Internet needs to send data to another computer, it must first know its IP address. IP addresses are typically shown as four numbers separated by decimal points, or "dots". For example, 10.24.254.3 and 192.168.62.231 are IP addresses.

If you need to make a telephone call but you only know the person's name, you can look them up in the telephone directory (or call directory services) to get their telephone number. On the Internet, that directory is called the Domain Name System or

DNS for short. If you know the name of a server, say www.cert.org, and you type this into your web browser, your computer will then go ask its DNS server what the numeric IP address is that is associated with that name.

1.12.3.1 Static And Dynamic Addressing

Static IP addressing occurs when an ISP permanently assigns one or more IP addresses for each user. These addresses do not change over time. However, if a static address is assigned but not in use, it is effectively wasted. Since ISPs have a limited number of addresses allocated to them, they sometimes need to make more efficient use of their addresses.

Dynamic IP addressing allows the ISP to efficiently utilize their address space. Using dynamic IP addressing, the IP addresses of individual user computers may change over time. If a dynamic address is not in use, it can be automatically reassigned to another computer as needed.

1.12.3.2 Attacks Against IP

A number of attacks against IP are possible. Typically, these exploit the fact that IP does not perform a robust mechanism for *authentication*, which is proving that a packet came from where it claims it did. A packet simply claims to originate from a given address, and there isn't a way to be sure that the host that sent the packet is telling the truth. This isn't necessarily a weakness, *per se*, but it is an important point, because it means that the facility of host authentication has to be provided at a higher layer on the ISO/OSI Reference Model. Today, applications that require strong host authentication (such as cryptographic applications) do this at the application layer.

1.12.3.3 IP Spoofing

This is where one host claims to have the IP address of another. Since many systems (such as router access control lists) define which packets may and which

packets may not pass based on the sender's IP address, this is a useful technique to an attacker: he can send packets to a host, perhaps causing it to take some sort of action.

1.12.4 TCP and UDP Ports

TCP (Transmission Control Protocol) and UDP (User Datagram Protocol) are both protocols that use IP. Whereas IP allows two computers to talk to each other across the Internet, TCP and UDP allow individual applications (also known as "services") on those computers to talk to each other.

In the same way that a telephone number or physical mail box might be associated with more than one person, a computer might have multiple applications (e.g. email, file services, web services) running on the same IP address. Ports allow a computer to differentiate services such as email data from web data. A port is simply a number associated with each application that uniquely identifies that service on that computer. Both TCP and UDP use ports to identify services. Some common port numbers are 80 for web (HTTP), 25 for email (SMTP), and 53 for Dmain Name System (DNS).

1.12.5 TCP

TCP is a transport-layer protocol. It needs to sit on top of a network-layer protocol, and was designed to ride atop IP. (Just as IP was designed to carry, among other things, TCP packets.) Because TCP and IP were designed together and wherever you have one, you typically have the other, the entire suite of Internet protocols are known collectively as TCP/IP. TCP itself has a number of important features that we'll cover briefly.

1.12.5.1 Guaranteed Packet Delivery

Probably the most important is guaranteed packet delivery. Host A sending packets to host B expects to get acknowledgments back for each packet. If B does not send an acknowledgment within a specified amount of time, A will resend the packet.

Applications on host B will expect a data stream from a TCP session to be complete, and in order. As noted, if a packet is missing, it will be resent by A, and if packets arrive out of order, B will arrange them in proper order before passing the data to the requesting application.

This is suited well toward a number of applications, such as a telnet session. A user wants to be sure every keystroke is received by the remote host, and that it gets every packet sent back, even if this means occasional slight delays in responsiveness while a lost packet is resent, or while out-of-order packets are rearranged.

It is not suited well toward other applications, such as streaming audio or video, however. In these, it doesn't really matter if a packet is lost (a lost packet in a stream of 100 won't be distinguishable) but it *does* matter if they arrive late (i.e., because of a host resending a packet presumed lost), since the data stream will be paused while the lost packet is being resent. Once the lost packet is received, it will be put in the proper slot in the data stream, and then passed up to the application.

1.12.6 UDP

UDP (User Datagram Protocol) is a simple transport-layer protocol. It does not provide the same features as TCP, and is thus considered "unreliable". Again, although this is unsuitable for some applications, it does have much more applicability in other applications than the more reliable and robust TCP.

1.12.6.1 Lower Overhead than TCP

One of the things that makes UDP nice is its simplicity. Because it does not need to keep track of the sequence of packets, whether they ever made it to their destination, etc., it has lower overhead than TCP. This is another reason why it's more suited to

streaming-data applications: there's less screwing around that needs to be done with making sure all the packets are there, in the right order, and that sort of thing.

1.12.7 Domain Name System (DNS)

DNS is a distributed database system used to match host names with IP addresses. A host normally requests the IP address of a given domain name by sending a UDP message to the DNS server which responds with the IP address or with information about another DNS server.

1.12.8 Telnet

Telnet provides simple terminal access to a host computer. The user is normally authenticated based on user name and password. Both of these are transmitted in plain text over the network however, and is therefore susceptible to capture.

1.12.9 File Transfer Protocols

FTP - The file transfer protocol is one of the most widely and heavily used Internet applications . FTP can be used to transfer both ASCII and binary files. Separate channels are used for commands and data transfer. Anonymous FTP allows external users to retrieve files from a restricted area without prior arrangement or authorisation. By convention users log in with the userid "anonymous" to use this service. Some sites request that the user's electronic mail address be used as the password.

CHAPTER 2
ROUTING CONCEPTS

2.1 Overview

This chapter introduces the underlying concepts widely used in routing protocols. Topics summarized here include routing protocol components and algorithms. In addition, the role of routing protocols is briefly contrasted with the role of routed or network protocols. Subsequent chapter, "Routing Protocols," address specific routing protocols in more detail, while the optimization problem are discussed at the 3rd chapter of this project.

2.2 What Is Routing

Routing is the act of moving information across an internetwork from a source to a destination. Along the way, at least one intermediate node typically is encountered. Routing is often contrasted with bridging, which might seem to accomplish precisely the same thing to the casual observer. The primary difference between the two is that bridging occurs at Layer 2 (the link layer) of the OSI reference model, whereas routing occurs at Layer 3 (the network layer). This distinction provides routing and bridging with different information to use in the process of moving information from source to destination, so the two functions accomplish their tasks in different ways.

The topic of routing has been covered in computer science literature for more than two decades, but routing achieved commercial popularity as late as the mid-1980s. The primary reason for this time lag is that networks in the 1970s were simple, homogeneous environments. Only relatively recently has large-scale internetworking become popular.

2.3 Routing Components

Routing involves two basic activities: determining optimal routing paths and transporting information groups (typically called packets) through an internetwork. In the context of the routing process, the latter of these is referred to as packet switching. Although packet switching is relatively straightforward, path determination can be very complex.

2.4 Path Determination

Routing protocols use metrics to evaluate what path will be the best for a packet to travel. A metric is a standard of measurement, such as path bandwidth, that is used by routing algorithms to determine the optimal path to a destination. To aid the process of path determination, routing algorithms initialize and maintain routing tables, which contain route information. Route information varies depending on the routing algorithm used.

Routing algorithms fill routing tables with a variety of information. Destination/next hop associations tell a router that a particular destination can be reached optimally by sending the packet to a particular router representing the "next hop" on the way to the final destination. When a router receives an incoming packet, it checks the destination address and attempts to associate this address with a next hop.

Figure 1-1 depicts a sample destination/next hop routing table.

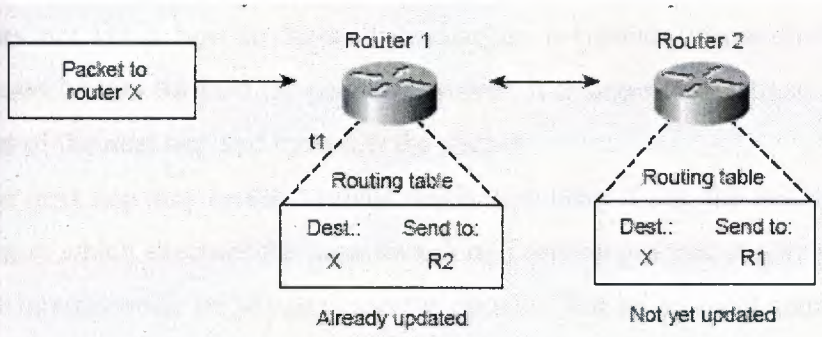


Figure 1-1 Destination/Next Hop Associations Determine the Data's Optimal Path

Routing tables also can contain other information, such as data about the desirability of a path. Routers compare metrics to determine optimal routes, and these metrics differ depending on the design of the routing algorithm used. A variety of common metrics will be introduced and described later in this chapter.

Routers communicate with one another and maintain their routing tables through the transmission of a variety of messages. The routing update message is one such message

that generally consists of all or a portion of a routing table. By analyzing routing updates from all other routers, a router can build a detailed picture of network topology. A link-state advertisement, another example of a message sent between routers, informs other routers of the state of the sender's links. Link information also can be used to build a complete picture of network topology to enable routers to determine optimal routes to network destinations.

2.5 Switching

Switching algorithms is relatively simple; it is the same for most routing protocols. In most cases, a host determines that it must send a packet to another host. Having acquired a router's address by some means, the source host sends a packet addressed specifically to a router's physical (Media Access Control [MAC]-layer) address, this time with the protocol (network layer) address of the destination host.

As it examines the packet's destination protocol address, the router determines that it either knows or does not know how to forward the packet to the next hop. If the router does not know how to forward the packet, it typically drops the packet. If the router knows how to forward the packet, however, it changes the destination physical address to that of the next hop and transmits the packet.

The next hop may be the ultimate destination host. If not, the next hop is usually another router, which executes the same switching decision process. As the packet moves through the internetwork, its physical address changes, but its protocol address remains constant, as illustrated in Figure 1-2.

The preceding discussion describes switching between a source and a destination end system. The International Organization for Standardization (ISO) has developed a hierarchical terminology that is useful in describing this process. Using this terminology, network devices without the capability to forward packets between subnetworks are called *end systems (ESs)*, whereas network devices with these capabilities are called *intermediate systems (ISs)*. ISs are further divided into those that can communicate within routing domains (*intradomain ISs*) and those that communicate both within and between routing domains (*interdomain ISs*). A routing domain generally is considered a portion of

an internetwork under common administrative authority that is regulated by a particular set of administrative guidelines. Routing domains are also called autonomous systems.With certain protocols, routing domains can be divided into routing areas, but intradomain routing protocols are still used for switching both within and between areas.

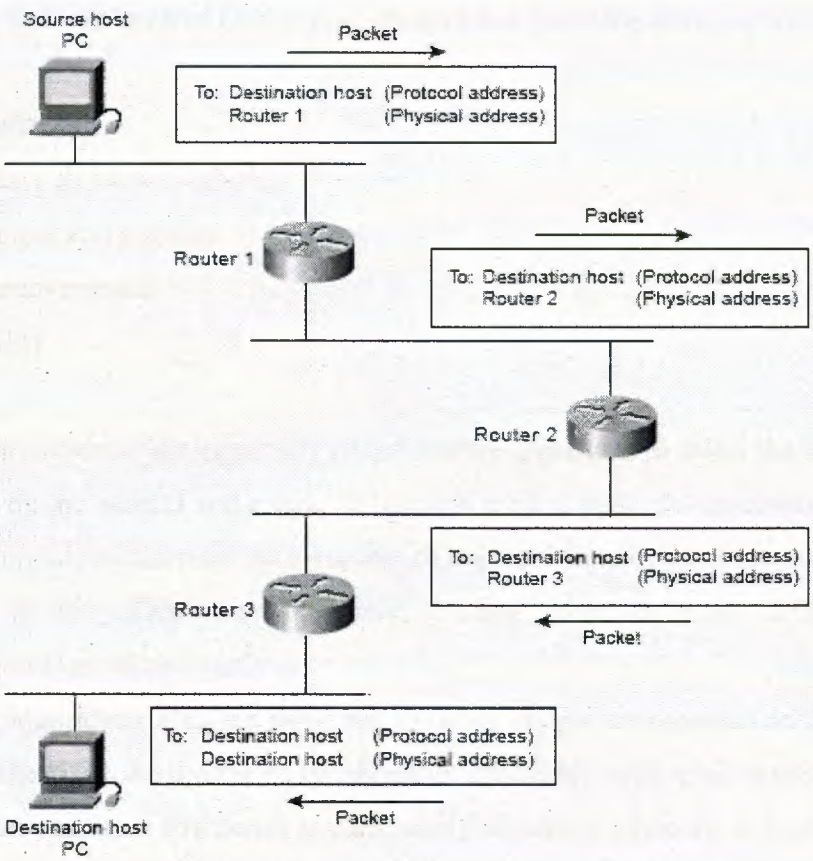


Figure 1-2 Numerous Routers May Come into Play During the Switching Process

2.6 Routing Algorithms

Routing algorithms can be differentiated based on several key characteristics. First, the particular goals of the algorithm designer affect the operation of the resulting routing

protocol. Second, various types of routing algorithms exist, and each algorithm has a different impact on network and router resources.

Finally, routing algorithms use a variety of metrics that affect calculation of optimal routes. The following sections analyze these routing algorithm attributes.

2.6.1 Design Goals

Routing algorithms often have one or more of the following design goals:

- Optimality
- Simplicity and low overhead
- Robustness and stability
- Rapid convergence
- Flexibility

Optimality refers to the capability of the routing algorithm to select the best route, which depends on the metrics and metric weightings used to make the calculation. For example, one routing algorithm may use a number of hops and delays, but it may weigh delay more heavily in the calculation. Naturally, routing protocols must define their metric calculation algorithms strictly.

Routing algorithms also are designed to be as simple as possible. In other words, the routing algorithm must offer its functionality efficiently, with a minimum of software and utilization overhead. Efficiency is particularly important when the software implementing the routing algorithm must run on a computer with limited physical resources.

Routing algorithms must be *robust*, which means that they should perform correctly in the face of unusual or unforeseen circumstances, such as hardware failures, high load conditions, and incorrect implementations. Because routers are located at network junction points, they can cause considerable problems when they fail. The best routing algorithms are often those that have withstood the test of time and that have proven stable under a variety of network conditions.

In addition, routing algorithms must converge rapidly. *Convergence* is the process of agreement, by all routers, on optimal routes. When a network event causes routes to

either go down or become available, routers distribute routing update messages that permeate networks, stimulating recalculation of optimal routes and eventually causing all routers to agree on these routes. Routing algorithms that converge slowly can cause routing loops or network outages.

In the routing loop displayed in Figure 5-3, a packet arrives at Router 1 at time t1. Router 1 already has been updated and thus knows that the optimal route to the destination calls for Router 2 to be the next stop. Router 1 therefore forwards the packet to Router 2, but because this router has not yet been updated, it believes that the optimal next hop is Router 1. Router 2 therefore forwards the packet back to Router 1, and the packet continues to bounce back and forth between the two routers until Router 2 receives its routing update or until the packet has been switched the maximum number of times allowed.

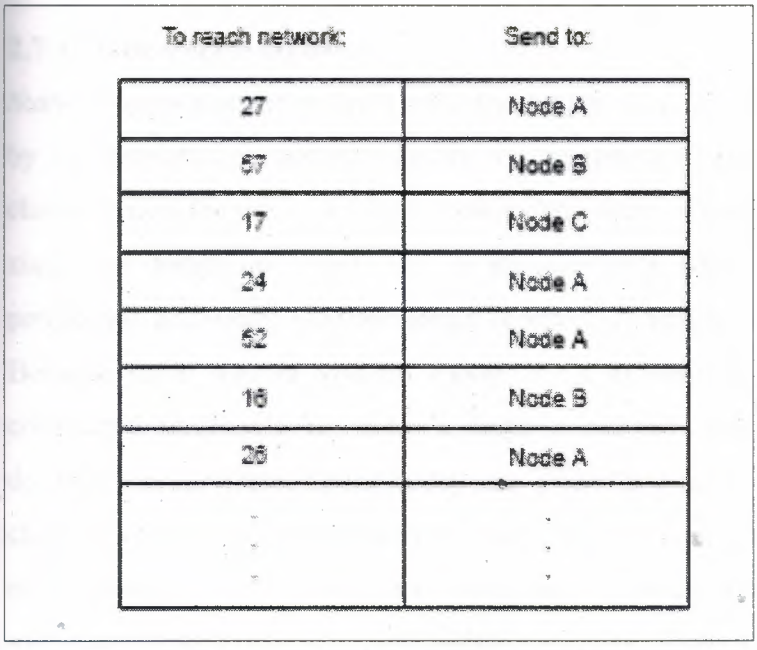


Figure 1-3 Slow Convergence and Routing Can Hinder prores

Routing algorithms should also be flexible, which means that they should quickly and accurately adapt to a variety of network circumstances. Assume, for example, that a network segment has gone down. As many routing algorithms become aware of the

problem, they will quickly select the next-best path for all routes normally using that segment. Routing algorithms can be programmed to adapt to changes in network bandwidth, router queue size, and network delay, among other variables.

2.7 Algorithm Types

Routing algorithms can be classified by type. Key differentiators include these:

- Static versus dynamic
- Single-path versus multi path
- Flat versus hierarchical
- Host-intelligent versus router-intelligent
- Intradomain versus interdomain
- Link-state versus distance vector

2.7.1 Static Versus Dynamic

Static routing algorithms are hardly algorithms at all, but are table mappings established by the network administrator before the beginning of routing. These mappings do not change unless the network administrator alters them. Algorithms that use static routes are simple to design and work well in environments where network traffic is relatively predictable and where network design is relatively simple.

Because static routing systems cannot react to network changes, they generally are considered unsuitable for today's large, constantly changing networks. Most of the dominant routing algorithms today are *dynamic routing algorithms*, which adjust to changing network circumstances by analyzing incoming routing update messages. If the message indicates that a network change has occurred, the routing software recalculates routes and sends out new routing update messages. These messages permeate the network, stimulating routers to rerun their algorithms and change their routing tables accordingly.

Dynamic routing algorithms can be supplemented with static routes where appropriate. A router of last resort (a router to which all unroutable packets are sent), for example, can

be designated to act as a repository for all unroutable packets, ensuring that all messages are at least handled in some way.

2.7.2 Single-Path Versus Multipath

Some sophisticated routing protocols support multiple paths to the same destination. Unlike single-path algorithms, these multipath algorithms permit traffic multiplexing over multiple lines. The advantages of multipath algorithms are obvious: They can provide substantially better throughput and reliability.

This is generally called load sharing.

2.7.3 Flat Versus Hierarchical

Some routing algorithms operate in a flat space, while others use routing hierarchies. In a *flat routing system*, the routers are peers of all others. In a hierarchical routing system, some routers form what amounts to a routing backbone. Packets from nonbackbone routers travel to the backbone routers, where they are sent through the backbone until they reach the general area of the destination. At this point, they travel from the last backbone router through one or more nonbackbone routers to the final destination.

Routing systems often designate logical groups of nodes, called domains, autonomous systems, or areas.

In *hierarchical systems*, some routers in a domain can communicate with routers in other domains, while others can communicate only with routers within their domain. In very large networks, additional hierarchical levels may exist, with routers at the highest hierarchical level forming the routing backbone.

The primary advantage of hierarchical routing is that it mimics the organization of most companies and therefore supports their traffic patterns well. Most network communication occurs within small company groups (domains). Because intradomain routers need to know only about other routers within their domain, their routing algorithms can be simplified, and, depending on the routing algorithm being used,routing update traffic can be reduced accordingly.

2.7.4 Host-Intelligent Versus Router-Intelligent

Some routing algorithms assume that the source end node will determine the entire route. This is usually referred to as *source routing*. In source-routing systems, routers merely act as store-and-forward devices, mindlessly sending the packet to the next stop.

Other algorithms assume that hosts know nothing about routes. In these algorithms, routers determine the path through the internetwork based on their own calculations. In the first system, the hosts have the routing intelligence. In the latter system, routers have the routing intelligence.

2.7.5 Intradomain Versus Interdomain

Some routing algorithms work only within domains; others work within and between domains. The nature of these two algorithm types is different. It stands to reason, therefore, that an optimal intradomain-routing algorithm would not necessarily be an optimal interdomain-routing algorithm.

2.7.6 Link-State Versus Distance Vector

Link-state algorithms (also known as shortest path first algorithms) flood routing information to all nodes in the internetwork. Each router, however, sends only the portion of the routing table that describes the state of its own links. In link-state algorithms, each router builds a picture of the entire network in its routing tables. Distance vector algorithms (also known as Bellman-Ford algorithms) call for each router to send all or some portion of its routing table, but only to its neighbors. In essence, link-state algorithms send small updates everywhere, while distance vector algorithms send larger updates only to neighboring routers. *Distance vector* algorithms know only about their neighbors.

Because they converge more quickly, link-state algorithms are somewhat less prone to routing loops than distance vector algorithms. On the other hand, link-state algorithms require more CPU power and memory than distance vector algorithms. Link-state algorithms, therefore, can be more expensive to implement and support. Link-state protocols are generally more scalable than distance vector protocols.

2.8 Routing Metrics

Routing tables contain information used by switching software to select the best route. But how, specifically, are routing tables built? What is the specific nature of the information that they contain?

How do routing algorithms determine that one route is preferable to others?

Routing algorithms have used many different metrics to determine the best route. Sophisticated routing algorithms can base route selection on multiple metrics, combining them in a single (hybrid) metric. All the following metrics have been used:

- Path length
- Reliability
- Delay
- Bandwidth
- Load
- Communication cost

Path length is the most common routing metric. Some routing protocols allow network administrators to assign arbitrary costs to each network link. In this case, path length is the sum of the costs associated with each link traversed. Other routing protocols define hop count, a metric that specifies the number of passes through internetworking products, such as routers, that a packet must take en route from a source to a destination.

Reliability, in the context of routing algorithms, refers to the dependability (usually described in terms of the bit-error rate) of each network link. Some network links might go down more often than others.

After a network fails, certain network links might be repaired more easily or more quickly than other links. Any reliability factors can be taken into account in the assignment of the reliability ratings, which are arbitrary numeric values usually assigned to network links by network administrators.

Routing delay refers to the length of time required to move a packet from source to destination through the internetwork. Delay depends on many factors, including the bandwidth of intermediate network links, the port queues at each router along the way, network congestion on all intermediate network links, and the physical distance to be

traveled. Because delay is a conglomeration of several important variables, it is a common and useful metric.

Bandwidth refers to the available traffic capacity of a link. All other things being equal, a 10-Mbps Ethernet link would be preferable to a 64-kbps leased line. Although bandwidth is a rating of the maximum attainable throughput on a link, routes through links with greater bandwidth do not necessarily provide better routes than routes through slower links. For example, if a faster link is busier, the actual time required to send a packet to the destination could be greater.

Load refers to the degree to which a network resource, such as a router, is busy. Load can be calculated in a variety of ways, including CPU utilization and packets processed per second. Monitoring these parameters on a continual basis can be resource-intensive itself.

Communication cost is another important metric, especially because some companies may not care about performance as much as they care about operating expenditures. Although line delay may be longer, they will send packets over their own lines rather than through the public lines that cost money for usage time.

2.9 Network Protocols

Routed protocols are transported by routing protocols across an internetwork. In general, routed protocols in this context also are referred to as network protocols. These network protocols perform a variety of functions required for communication between user applications in source and destination devices, and these functions can differ widely among protocol suites. Network protocols occur at the upper five layers of the OSI reference model: the network layer, the transport layer, the session layer, the presentation layer, and the application layer.

Confusion about the terms *routed protocol* and *routing protocol* is common. Routed protocols are protocols that are routed over an internetwork. Examples of such protocols are the Internet Protocol (IP), DECnet, AppleTalk, Novell NetWare, OSI, Banyan VINES, and Xerox Network System (XNS). Routing protocols, on the other hand, are protocols that implement routing algorithms. Put simply, routing protocols are used by

intermediate systems to build tables used in determining path selection of routed protocols. Examples of these protocols include Interior Gateway Routing Protocol (IGRP), Enhanced Interior Gateway Routing Protocol (Enhanced IGRP), Open Shortest Path First (OSPF), Exterior Gateway Protocol (EGP), Border Gateway Protocol (BGP), Intermediate System-to-Intermediate System (IS-IS), and Routing Information Protocol (RIP). Routed and routing protocols are discussed in detail later in this project.

CHAPTER 3
ROUTING PROBLEM

3.1 Terminology (Cont.)

1. Dynamic Routing

– Routers and hosts use an IGP or EGP to update their routing tables periodically

2.Typically used in network backbone and WANs, and groups of LANs

3. Static Routing

– Routers and hosts are administratively configured with some number of routes that will not change

4. Typically used on hosts and on "edge routers"

5. Policy-Based Routing

– Routing decisions are not just made upon topological, connectivity or traffic considerations

– Local administrative policy may determine or influence routing.

6. E.g., "send all packets from customer X on network Y"

7. Most networks use some combination of all three

3.2 History

1. In the old ARPANET, routing was static.

2. As the ARPANET grew, the routing became more dynamic, but with all routers sharing a single protocol.

3. As the Internet became the "network of networks" routing was separated into interior and exterior domains.

– Each AS could determine the IGP that suited it best

– A standard EGP was used between AS's

4. Today

– RIP and OSPF are the most widely used IGP's

– IS-IS is another IGP that is generally available

– EGP was the first EGP (confused?) but has been replaced with BGP (which is now in version 4)

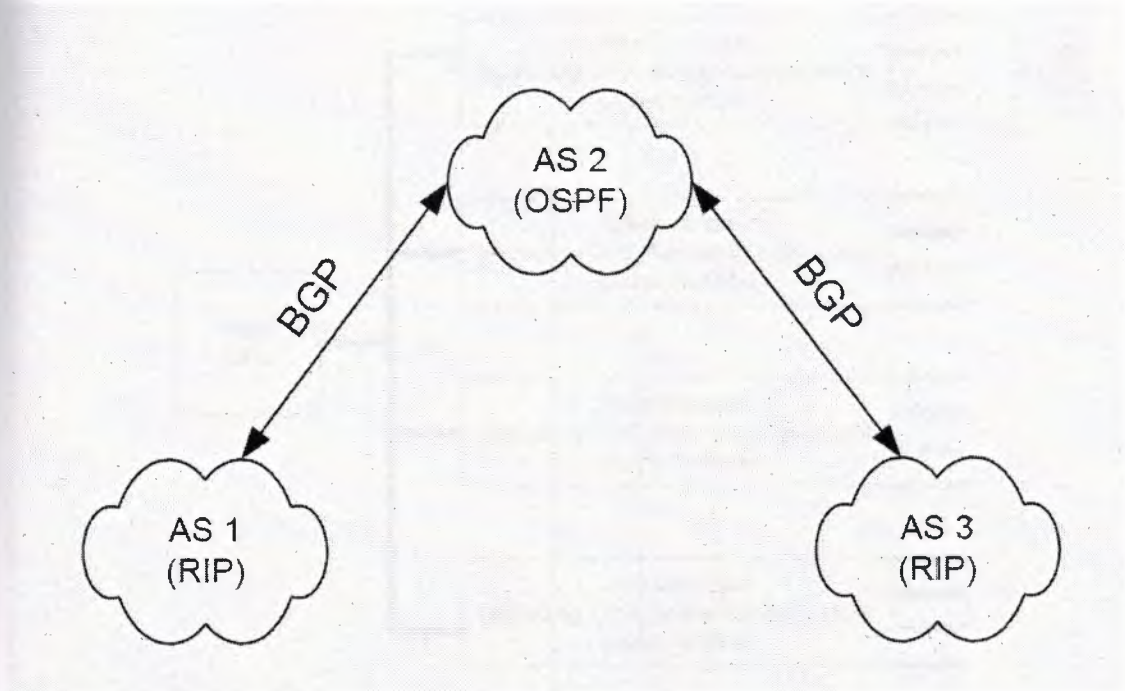


Figure 3.1. Example Routing Architecture

3.3 How Routers Work

3.3.1 History

1. Routing in some form or another is as old as the Internet

– ARPANET IMP's were gateways to local networks

2. Supported a variety of protocol stacks, not much TCP/IP until 80's

3. IP routing as we know it dates to about 1985

– Cisco popularized IP routers as an efficient way of connecting LANs on a campus

4. Four basic generations of IP routers

– Circa 1985-1990: PC/workstation based, simple, slow

– Circa 1990-1995: Some CPU offload onto interfaces

– Circa 1995-2000: CPU and bus out of forwarding path

– Circa 2000-??: Network and general-purpose processors allow semi-off-the-shelf design instead of using custom ASICs

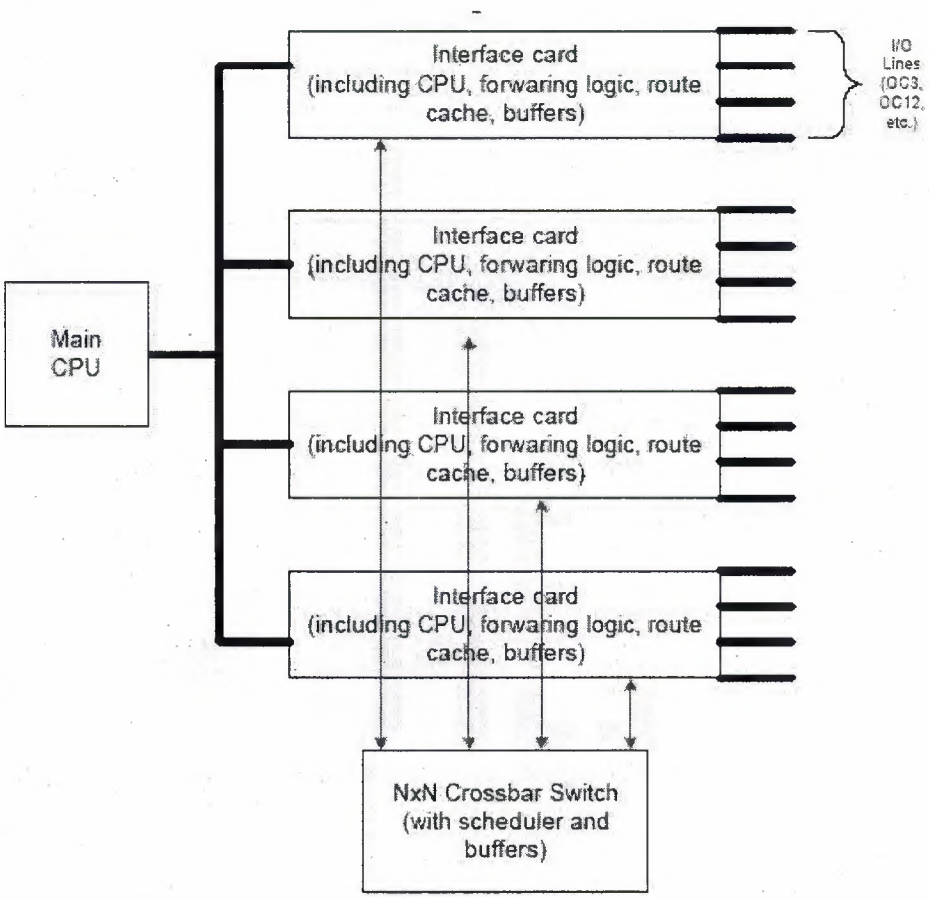


Figure 3.2. High-End Router Design

(As of about 2000)

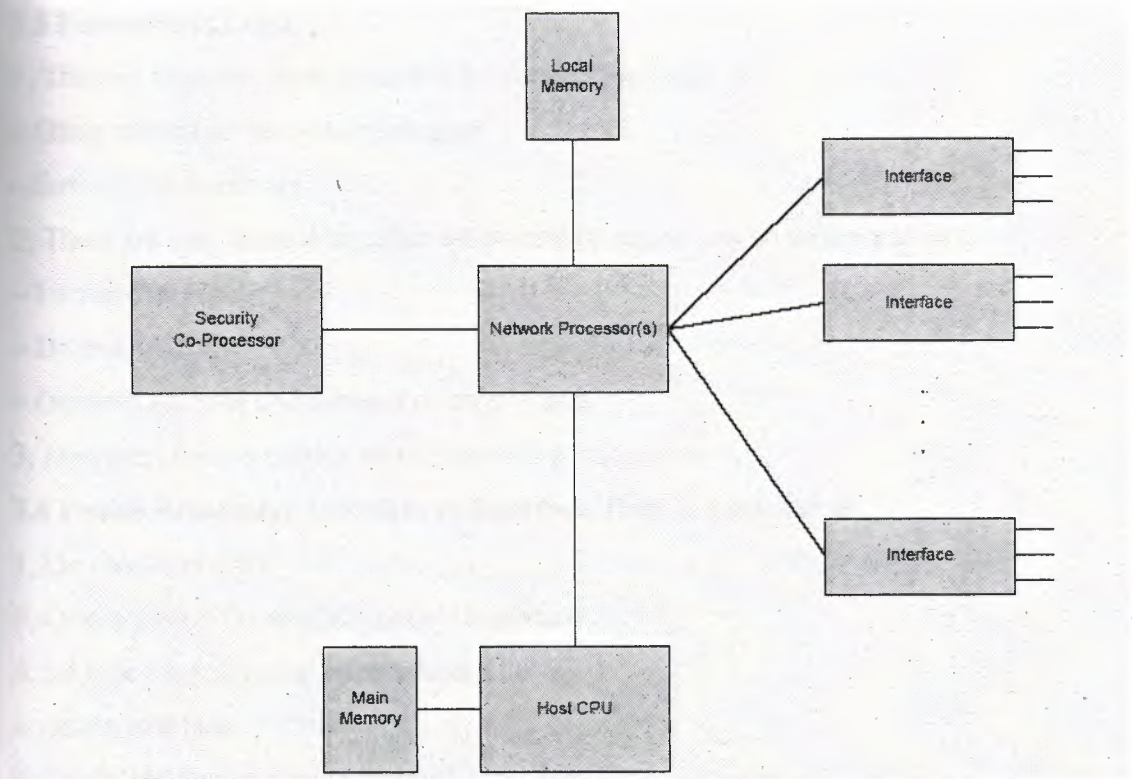


Figure 3.3.High-End Router Design Using

Network Processors

3.4 Today's Routers

1. Four classes (distinctions not always clear)

– Backbone ($100K - $500K or more)

– Medium / large enterprise ($15K - $150K)

– Small enterprise ($1K - $20K)

– Residential (< $200)

2. Cable, DSL, set top boxes, etc.

3. Just about all IP stacks can do routing

– Its hard to build one that doesn't!

– Win98 SE, Win2K, Win XP, FreeBSD, and Linux PCs make nice small routers

3.5 Forwarding Logic

1. The key to router performance is the forwarding logic

– Often called the "forwarding engine"

– Software or hardware

2. There are only three things that a forwarding engine can do with a packet

– Forward it

– Drop it

– Queue it for now and forward or drop it later

3. However, there's quite a lot of processing that occurs

3.6 Packet Processing Interface to Interface, Part 1: Link Layer

1. Do link layer CRC

2. Queue packet for reading by the IP module

3. Include the following information

– ngress interface

4. For ICMP (saves a route lookup)

– Media length

5. For IP header check

– Link layer destination type (uni-, multi-, broadcast)

6. ICMP must not respond to broadcasts or multicasts

– Link layer source address

7. For ICMP (may save an ARP)

3.7 Packet Processing Interface to Interface, Part 2: IP Header Check

1. Media length >= 20 bytes

2. Version = 4 (or 6)

3. Header length >= 5

4. Header length <= total length

5. Checksum validation

– Mandatory according to RFC1812, but many routers can be configured to skip this step!

6. If link layer address was broad cast, IP address must be broadcast or multicast

7. If link layer address was multicast, IP address must be multicast

8. Discard Martians

3.8 Packet Processing Interface to Interface, Part 3: Determine Destination

1. The packet is for this router if

– Destination matches any of the routers' IP addresses

– Broadcast

– If multicast, check further

– Any of these cases except possibly multicast, packet is not forwarded

2. Otherwise, do route lookup

– Simplest lookup is based on destination IP

3. Longest prefix-match algorithm (several variations, some complex)

– May also include source IP, TOS, source port, destination port, protocol, other fields

– If more than one route, choose one with best metric

4. Lowest load, least number of hops, etc

5. All metrics the same? Do something intelligent!

3.9 Packet Processing Interface to Interface, Part 4: Write Header, Give to Interface

1. If TTL is 0 or 1, must not forward

2. Decrement TTL by at least one

3. Fragment if necessitated by egress interface

4. Modify TOS if policy says so

5. Re-compute checksum

6. Send to switching fabric…done!

3.10 Design for Speed

1. "Make the common case fast"

2. Forwarding engines are designed to operate at line speed on "normal" IP packets

– Streams of small packets can stress a router to its limits

– Minimize reading and writing latencies with a wide bus

3. "Abnormal" packets are given to main CPU for more complicated processing

– IP options

– ICMP

– Packets destined to this router

– Requires bus traversal and more computation

4. Slow!

3.11 Routing Information Protocol (RIP)

3.11.1 Introduction

1. RIP is a distance-vector dynamic routing protocol

– Based on theory developed by Bellman in 1957 that was further refined by Ford and Fulkerson in 1962

– Goal: Minimize the number of hops (distance) to the destination by routing packets via the proper interface (vector)

2. The most widely deployed IGP

3. Easy to implement

– Two messages and one table

4. Cannot be used in large networks (>15 hops)

5. Subject to transient instability

6. The best starting place is a simple example, so...

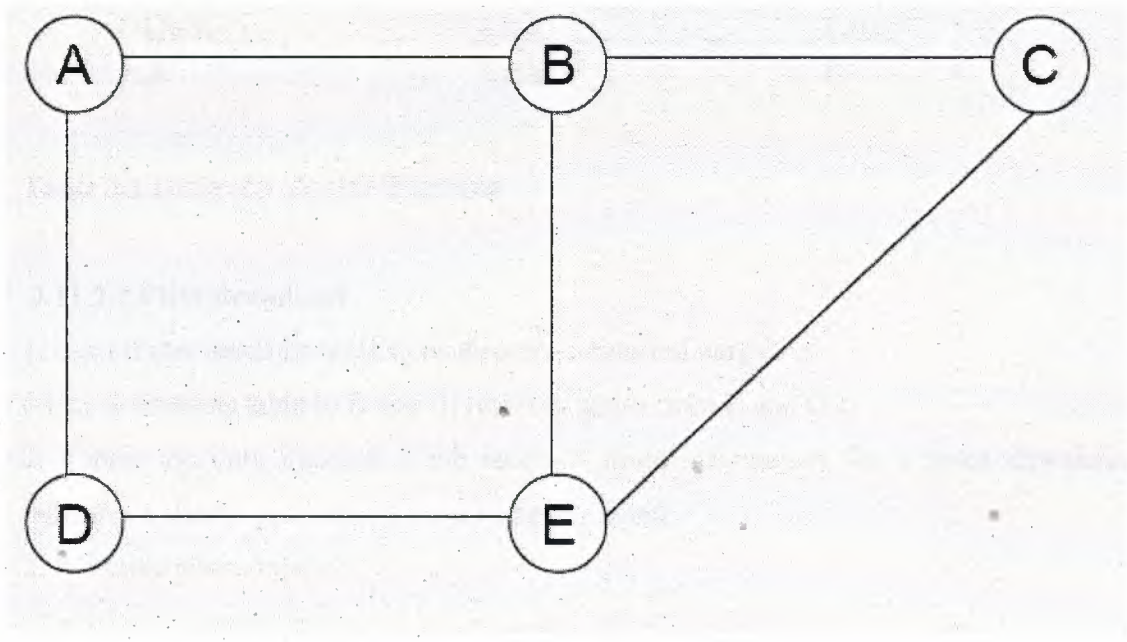


Figure 3.4. Distance Vector Example

7. Toy network of 5 routers and 6 links

– We'll ignore hosts

8. We'll assume that all links are the same distance

9. The link between x and y is called "link xy"

– I.e., AD, BE, AB, BC, CE, DE

3.11.2 Distance Vector Example:

3.11.2.1 Startup

1. All routers boot simultaneously

2. The routers do not know anything about the network

3. They each contain only a single routing table entry to

Them selves

– Assume no static routes except loop back

- For example, Router A:

Dest	Link	Cost
A	local	0

Table 3.1. Distance Vector Example

3.11.2.2 First Broadcast

1. Each router sends its table to its directly-connected neighbors

– I.e., A sends its table to B and D, receives tables from B and D

2. Tables are only updated if the received route information for a given destination indicates a shorter path than the one currently listed

3. A's table now contains:

Dest	Link	Cost
A	local	0
B	AB	1
D	AD	1

3.11.2.2.1 First Broadcast (Cont.)

1. The rest of the tables:

B

Dest	Link	Cost
B	local	0
A	AB	1
C	BC	1
E	BE	1

D

Dest	Link	Cost
D	local	0
A	AD	1
E	DE	1

C

Dest	Link	Cost
C	local	0
B	BC	1
E	CE	1

E

Dest	Link	Cost
E	local	0
B	BE	1
C	CE	1
D	DE	1

Table 3.2. First Broadcast

3.11.2.3 Second Broadcast

1. Each router sends its table to its neighbors again:

A

Dest	Link	Cost
A	local	0
B	AB	1
C	AB	2
D	AD	1
E	AB	2

B

Dest	Link	Cost
B	local	0
A	AB	1
C	BC	1
D	AB	2
E	BE	1

D

Dest	Link	Cost
D	local	0
A	AD	1
B	AD	2
C	DE	2
E	DE	1

C

Dest	Link	Cost
C	local	0
A	BC	2
B	BC	1
D	CE	2
E	CE	1

E

Dest	Link	Cost
E	local	0
A	BE	2
B	BE	1
D	CE	1
C	DE	1

24

Table 3.3. Second Broadcast

3.11.2.4 Stability

1. At this point, routing in the network is stable

– Routing updates will continue, periodically

– No changes made because shorter paths cannot be found

2. When multiple equal-cost paths exist, only the first will be used

– Update algorithm

3. If new cost(X,Y) < old cost(X,Y) then old cost(X,Y) = new cost(X,Y)

4. As long as the network is stable, distance vector protocols work reasonably well

– When an outage occurs, they may take a while to converge, if at all!

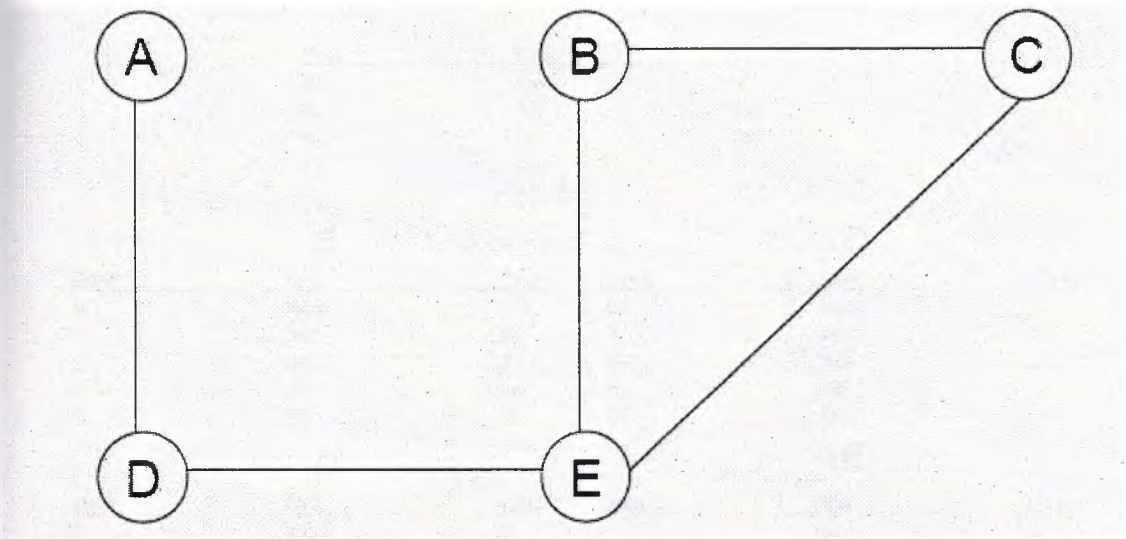


Figure 3.5. Link AB Goes Down

5. A and B will both immediately update their routing tables and send these updated tables to their neighbors
6. The routing will take a couple of steps to converge

3.11.2.5 Updated Routing Tables

A

Dest	Link	Cost
A	local	0
B	AB	INF
C	AB	INF
D	AD	1
E	AB	INF

B

Dest	Link	Cost
B	local	0
A	AB	INF
C	BC	1
D	AB	INF
E	BE	1

D

Dest	Link	Cost
D	local	0
A	AD	1
B	AD	2
C	DE	2
E	DE	1

C

Dest	Link	Cost
C	local	0
A	BC	2
B	BC	1
D	CE	2
E	CE	1

E

Dest	Link	Cost
E	local	0
A	BE	2
B	BE	1
C	CE	1
D	DE	1

Table 3.4. Updated Routing Tables

3.11.2.6 A and B Broadcast Their Tables

A

Dest	Link	Cost
A	local	0
B	AB	INF
C	AB	INF
D	AD	1
E	AB	INF

B

Dest	Link	Cost
B	local	0
A	AB	INF
C	BC	1
D	AB	INF
E	BE	1

D

Dest	Link	Cost
D	local	0
A	AD	1
B	AD	INF
C	DE	2
E	DE	1

C

Dest	Link	Cost
C	local	0
A	BC	INF
B	BC	1
D	CE	2
E	CE	1

E

Dest	Link	Cost
E	local	0
A	BE	INF
B	BE	1
C	CE	1
D	DE	1

28

Table 3.4.A and B Broadcast Their Tables

3.11.2.7 C, D, and E Broadcast Their Tables

A

Dest	Link	Cost
A	local	0
B	AB	INF
C	AD	3
D	AD	1
E	AD	2

B

Dest	Link	Cost
B	local	0
A	AB	INF
C	BC	1
D	BE	2
E	BE	1

D

Dest	Link	Cost
D	local	0
A	AD	1
B	DE	2
C	DE	2
E	DE	1

C

Dest	Link	Cost
C	local	0
A	BC	INF
B	BC	1
D	CE	2
E	CE	1

E

Dest	Link	Cost
E	local	0
A	DE	2
B	BE	1
C	CE	1
D	DE	1

29

Dest	Link	Cost
C	local	0
A	BC	INF
B	BC	1
D	CE	2
E	CE	1

Dest	Link	Cost
E	local	0
A	BE	INF
B	BE	1
C	CE	1
D	DE	1

28

Table 3.5.C, D, and E Broadcast Their Tables

3.11.2.8 Final Broadcast Updates A, B, and C

A

Dest	Link	Cost
A	local	0
B	AD	3
C	AD	3
D	AD	1
E	AD	2

B

Dest	Link	Cost
B	local	0
A	BE	3
C	BC	1
D	BE	2
E	BE	1

D

Dest	Link	Cost
D	local	0
A	AD	1
B	DE	2
C	DE	2
E	DE	1

C

Dest	Link	Cost
C	local	0
A	CE	3
B	BC	1
D	CE	2
E	CE	1

E

Dest	Link	Cost
E	local	0
A	DE	2
B	BE	1
C	CE	1
D	DE	1

30

Table 3.6. Final Broadcast Updates A, B, and C

3.12 Problems With Distance Vector

1. In our example, a single outage occurred

– Required 3 sets of broadcasts to fix

– In the mean time, packets are delayed and/or dropped

2. In a larger network, it could take much, much longer

– Dependent on topology and link cost, weird things can happen

3. Additionally, distance vector is prone to "count to infinity"

– Part of the network becomes isolated

– The isolated routers keep telling each other that they know how to get to the rest of the network

4. But they don't!

– Routing loops are created

3.13 Counting to Infinity

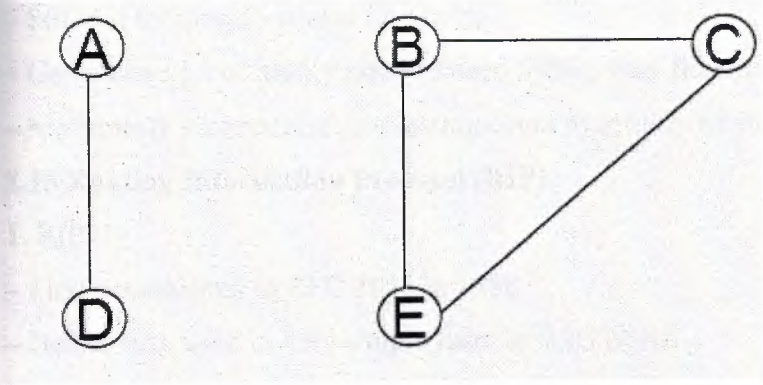


Figure 3.6. Counting to Infinity

1. Continuing our previous example.

– AB went down then the network converged...

2. Link DE goes down

– If D immediately sends an update to A, telling A that all costs to B, C, and E are infinite, then the system converges

– If A sends an update first, D will think it can reach B, C, and E through A!!

– Since all of A's paths are via D, when D sends the next routing update, A's costs will increase by 2

– Etc., etc.

3.14 Trying to Avoid Count to Infinity

1. Set "infinity" to be a finite (and small) number

– RIP uses 16, which is why RIP is limited to networks with 15 or fewer hops

2. Use Split Horizon

– Don't advertise to X a route to Y if you go through X to get to Y

– With "poison reverse"

3. If you go through X to get to Y, advertise to X an infinite-cost route to Y

4. Issue: potentially large routing updates sent because each router also lists all the routes that it *can't* get to

– Works fine for loops of size two

5. Still counts to infinity when loops are larger

6. Use triggered updates

– Whenever a router receives an update, it immediate sends them on to its neighbors: faster convergence

– Still not foolproof - packet loss hurts

– Generates a lot of highly synchronized traffic, may flood network

– Moderately successful in preventing count to infinity when loops are larger than two

3.15 Routing Information Protocol (RIP)

1. RIPv1

– First documented in RFC 1058 in 1988

– Before that, used in very simple form in BSD UNIX

2. Uses UDP port 520 for sending and receiving

– An ephemeral source port can be used for queries

3. Packets consist of control information followed by a list of routes and their associated metrics

4. Route updates normally sent every 30 seconds

– Faster for triggered updates, but with extra delay of 1-5 seconds to avoid synchronization

5. A route is timed out (metric set to infinity) if no updates are heard within 180 seconds

– After 120 more seconds, route is deleted

3.15.1 RIPv1 Fields

1. Command

– Request (1)

– Response (2)

2. Version: Always 1 for RIP v1

3. Address Family: 2 for IP

4. MBZ: "Must Be Zero"

5. Route entries (0 to 25 per packet)

– IP Address

6. Address or network

7. 0.0.0.0 for default route

– Metric

8. Only can take on values 0-15; 16 indicates infinity

9. Almost always a hop count; range is too small for much else

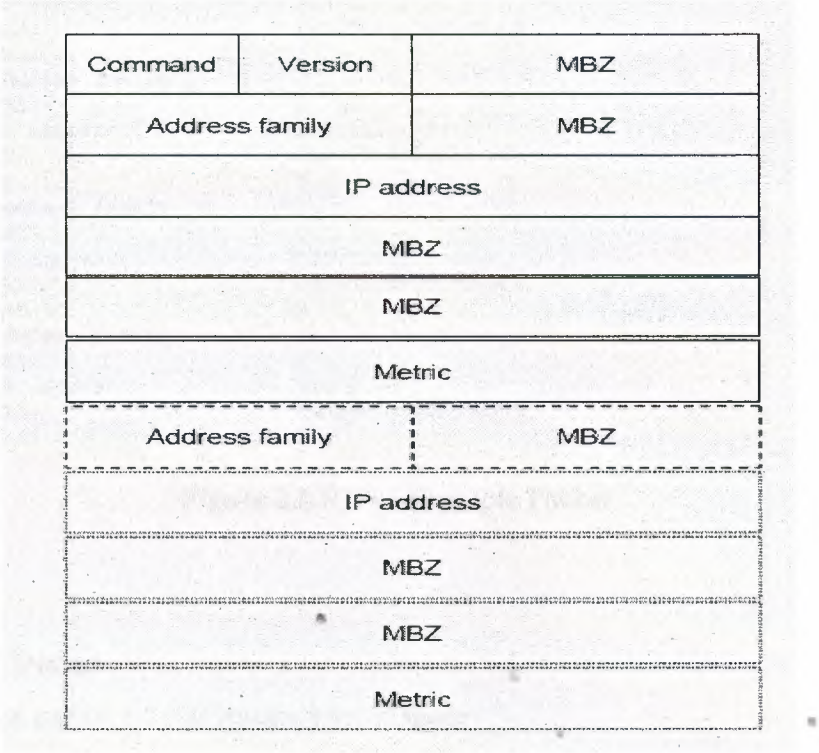


Figure 3.7.RIPv1 Fields

```
-------------------------------------------------------------
                        IP Header
-------------------------------------------------------------
Source address:         149.112.36.254
Destination address:    149.112.36.255
-------------------------------------------------------------
                        UDP Header
-------------------------------------------------------------
Source port:            520 (RIP)
Destination port:       520 (RIP)
-------------------------------------------------------------
                        RIPv1 Header
-------------------------------------------------------------
Command:                2 (reply)
Version:                1
MBZ:                    0
Address family:         2
MBZ:                    0
IP address:             211.213.1.0
MBZ:                    0x0000000000000000
Metric:                 3
Address family:         2
MBZ:                    0
IP address:             213.211.1.0
MBZ:                    0x0000000000000000
Metric:                 3
Address family:         2
MBZ:                    0
IP address:             213.212.1.0
MBZ:                    0x0000000000000000
Metric:                 3
Address family:         2
MBZ:                    0
IP address:             219.219.1.0
MBZ:                    0x0000000000000000
Metric:                 3
Address family:         2
MBZ:                    0
IP address:             0.0.0.0
MBZ:                    0x0000000000000000
Metric:                 2
```

Figure 3.8.RIPv1 Example Packet

3.15.2 RIPv1 Design

1. Note that the packet format wastes a lot of space

2. There are a number of reasons for this

– RIP was originally designed so that it could be used with non-IP networks

3. This never really happened, as other protocols designed their own version of RIP

4. E.g., IPXRIP, XNS RIP, etc.

5. Eight empty bytes in each route entry are for support of the other addressing families

– Fields tend to fall on 32-bit boundaries

6. Easier for processing on 32-bit processors

3.15.3 RIPv1 Processing

1. A request packet lists all of the networks or hosts that the sender wants routing entries for

2. The recipient parses the table adding the appropriate metric into each entry, then sends the table back to the requestor

3. A request with no route entries gets no response

4. A "special request" has an IP address of 0.0.0.0 and a single route entry with a metric of 16

– This indicates that the sender wants all routing entries

3.15.4 Problems with RIPv1

1. No security

– Anyone can send bogus routing updates

– Most implementations allow the operator to configure a "trusted routers" list, but this could require a lot of manual configuration

2. No support for CIDR

– Need net masks as well as networks

3. By the time that RIP needed updating, OSPF and other link state routing protocols were already considered superior

– A conundrum: RIP had a large installed base

– Should the IETF "encourage" use of an inferior protocol by updating it?

– The practical answer was "yes"

4. RIP v1 was broken and needed fixing

5. Not everyone was going to use OSPF anyway

6. Political and administrative issues, rather than technical

3.16 RIPv2

1. Defined as proposed, draft, and full standard in RFC's 1388, 1723, and 2453, respectively

2. Additional features

– Weak authentication

– Stronger authentication: RFC 2082

– CIDR support

– Same general packet format but less wasted space

– Support for interaction with an EGP

– General route updates are multicast to 224.0.0.9 rather than broadcast

3. Still widely used today, despite its limitations

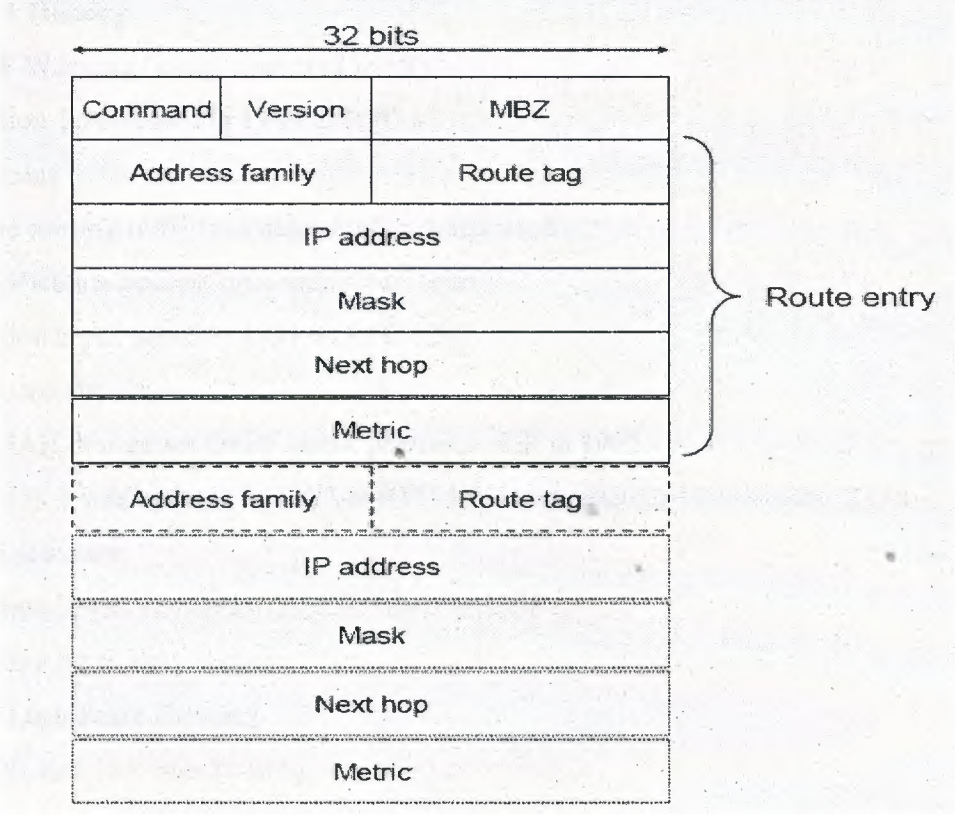


Figure3.9.RIPv2 Packet Format

3.16.1 RIPv2 New Fields

1. Route tag

– Generic field that must be preserved and re-advertised with the route

– Allows for RIP-EGP interaction

2. Some routes can be tagged "internal" and others can be tagged "external"

3. The tag usually takes the form of a autonomous system (AS) number

4. Exact interpretation is not specified

5. Mask

– The subnet mask of the address part of the entry

– Allows for hierarchical route specifications and aggregation

6. Next hop

– The router to which destinations matching this route entry should be forwarded

3.17 Open Shortest Path First (OSPF)

3.17.1 History

1. IETF Working Group chartered in 1987

2. Version 1 published in 1989 as RFC 1131

– Problems

3. Some routing traffic not deleted after being used

4. Specification unclear on a number of issues

5. Version 2 published in 1991 as RFC 1247

– Proposed standard

6. The IAB designates OSPF as the preferred IGP in 1992

7. Version 2 was updated in 1993 as RFC 1583 and again in 1997 as RFC 2178

– Draft standard

8. Version 2 was elevated to full Standard in 1998

– Note the long, slow process!

3.17.2 Link State Routing

1. OSPF uses link state routing

– Significantly different than distance vector routing

2. Each router keeps a complete map of the network, rather than just how to get to each of the other routers

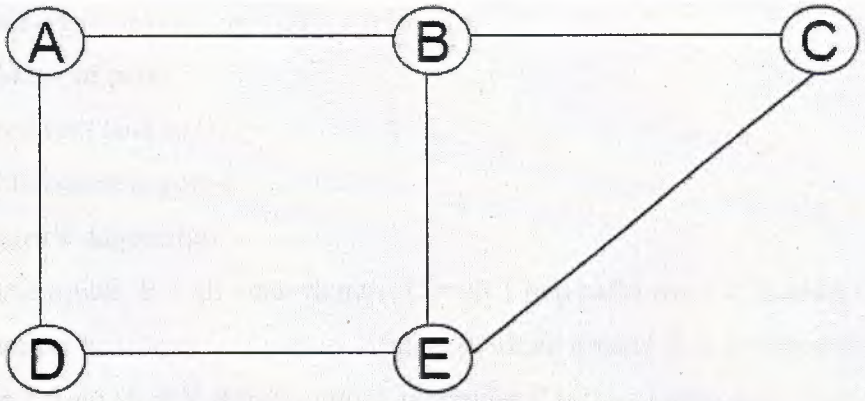
– This map is used to populate routing tables

– All routers should have exactly the same map

3. Routing updates are "flooded" to all nodes

4. Shortest paths between any two points are very easy to compute

5. Fast convergence when the network topology changes



From->To	Link	Cost
A -> B	AB	1
A -> D	AD	1
B -> A	AB	1
B -> C	BC	1
B -> E	BE	1
C -> B	CB	1
C -> E	CE	1
D -> A	AD	1
D -> E	DE	1
E -> B	BE	1
E -> C	CE	1
E -> D	DE	1

Figure 3.10. Link State Example

3.17.3 Shortest Path Calculation

1. Once the map is known to a router, it can compute the shortest path to all other routers

2. Variables:

– S: Source node

– E: All evaluated nodes (shortest path known)

– R: Remaining nodes (shortest path not known)

– O: Sorted list of paths

– P: Shortest cost path in O

– V: The last router in path P

3.18 Dijkstra's Algorithm

1. S = source router, R = all other routers, O = all 1 hop paths from S, in order of cost

2. If O is empty or lowest cost path is infinite, mark all routers in R as unreachable, stop

3. Remove P from O, if V is in E, goto 2, otherwise P is the shortest path from S to V, put V into E with the path and the cost, remove V from R

4. Build new paths by concatenating P with the links that emanate from P, with appropriate metrics added

5. Insert new paths in O, in order

6. Go to 2

3.19 Flooding Algorithm

1. Receive a link state update message

2. If it refers to an entry not yet in the local table

a) Add the entry to the database

b) Broadcast the message on all outgoing interfaces except the one the message was received on

3. Else if the metric in the entry is lower than the one in the database

a) Replace the entry to the database

b) Broadcast the message on all outgoing interfaces except the one the message was received on

4. Else if the metric is lower in the database than the one in the entry

a) Broadcast a new message reflecting the metric from the database on the receiving interface

3.20 Why is Link State Better Than Distance-Vector

1. Fast convergence

– Speed is proportional to number of nodes in the network

2. Loopless convergence

– After flooding all routes are stable, no count to infinity

3. Support of multiple metrics

– Throughput, delay, loss, cost, policy, security…

– But all routers should use the same metric, otherwise loops may occur

4. Support for multiple equivalent paths

– In theory, but not so easy in practice

3.21 OSPF Areas

1. OSPF allows a autonomous system's network to be divided into routing areas

– An area contains one or more subnets

– Each area runs the OSPF algorithm independently of the others

– There must be a "backbone" area

– All areas must connect to the backbone

– "Stub" areas have only one exit point

2. Types of OSPF routers

– Area border routers (ABRs): routers between multiple areas

– Autonomous system boundary routers (ASBRs): routers that exchange routes with a different autonomous system

3. Typically using an EGP

3.21.1 OSPF Protocol

1. Uses IP protocol type 89 (not UDP or TCP!)

2. Three sub-protocols

– Hello: Link maintenance

– Exchange: Initial exchange of routing tables

– Flooding: Incremental routing table updates

3. All OSPF packets begin with a common header

4. Authentication methods

– None

– Simple 8 character password

– MD5 hash of packet and secret key

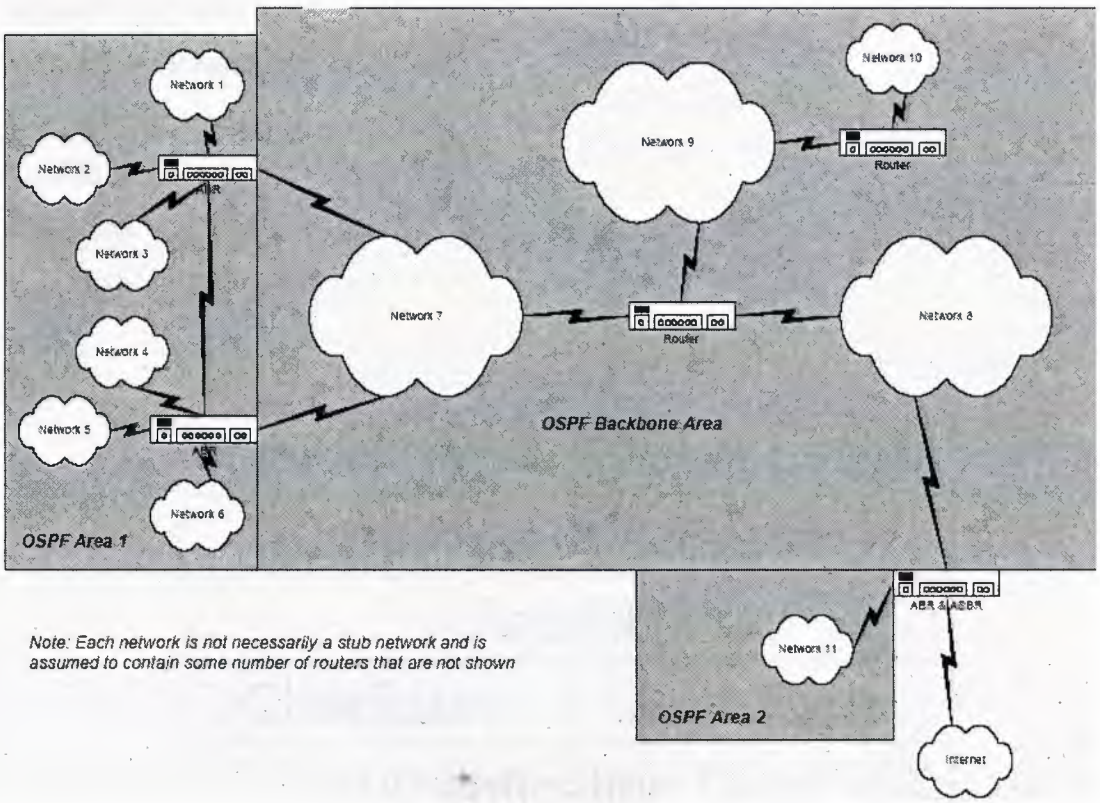


Figure 3.11.OSPF Area Example

3.22.2 OSPF Common Header Fields

1. Version (1 byte)

– 1 or 2

2. Type (1 byte)

– 1 for hello, 2 = exchange, 4 = flood update, 5 = flood ack

3. Packet length (2 bytes)

– Length in bytes of the entire OSPF packet

4. Sender's IP address (4 bytes)

– IP address of the router that sent the packet

5. Area ID (4 bytes)

– An IP network number from the AS

6. Checksum (2 bytes)

– Same as IP checksum

7. Auth type

-Type of authentication

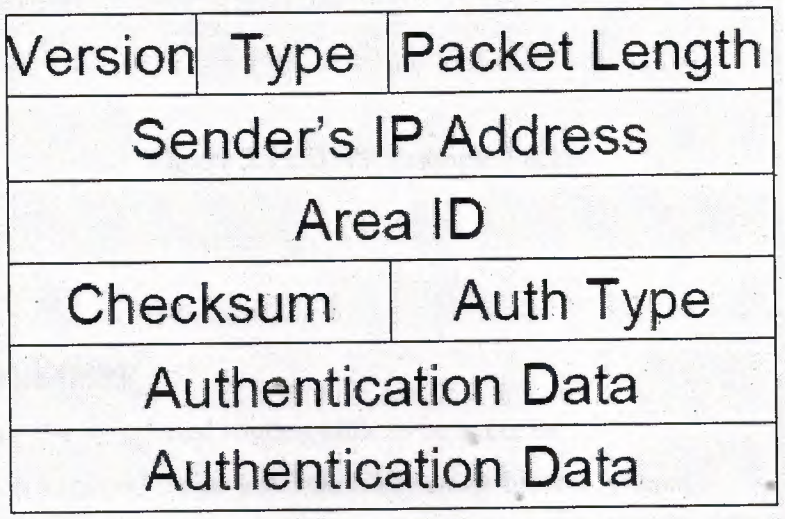


Figure 3.12.OSPF Common Header Fields

```
-------------------------------------------------------------------------------
                        IP Header
-------------------------------------------------------------------------------
Version:                4
Header length:          5 (20 bytes)
TOS:                    0xc0
Total length:           64
Identification:         65488
Fragmentation offset:   0
Unused bit:             0
Don't fragment bit:     0
More fragments bit:     0
Time to live:           1
Protocol:               89 (OSPF)
Header checksum:        40544
Source address:         149.112.164.254
Destination address:    224.0.0.5
-------------------------------------------------------------------------------
                        OSPF Header
-------------------------------------------------------------------------------
Version:                2
Type:                   1 (hello)
Length:                 44
Router ID:              149.112.79.242
Area ID:                149.112.14.0
Checksum:               15196
Authentication:         0 (none)
Authentication data:    0x0000000000000000
Netmask:                255.255.255.0
Interval:               10
Options:                0x00
Router priority:        1
Dead interval:          40
Designated router:      149.112.164.254
Backup router:          0.0.0.0
```

Figure 3.12.OSPF Example Packet

3.22.3 Issues With OSPF

1. OSPF relies on the distributed routing table to be accurate

– Implementation bugs and major link instability cause bizarre failures

2. Hackers can still spoof bogus route updates

– Most networks don't use hashed authentication

3. Load balancing between equal metric paths is difficult

– Out of order delivery

– Oscillation

4. Complexity

– Five messages

– Three algorithms (Dijkstra, flooding, exchange)

– A lot of code

3.23 Border Gateway Protocol (BGP)

3.23.1 Background

1. As the ARPANET evolved in to the Internet, a number of changes were made to the routing architecture

– The Internet was broken up into AS's

– Routing protocols were divided into IGP's and EGP's

2. EGP was the first EGP

– Something like distance-vector routing between AS's

– Poor loop avoidance, count to infinity

3. The Inter-Domain Routing Protocol (IDRP) was proposed

– Something like link-state between AS's

– Loop avoidance, but there were too many AS's for it to scale well

– No support for the varying metrics that service providers need

3.23.2 Enter BGP

1. BGP version 4 is the current EGP of choice

– First three versions did not support CIDR

2. Distance-vector and link-state drawbacks eliminated in a new technique called "path-vector"

– Each BGP routing update contains the complete list of AS's that it has traversed

– Loop avoidance is easy

3. BGP route advertisements contain a list of AS's that the advertisement has already traversed

4. When a border router receives a route advertisement it scans for its own AS number

– If it is already listed, it will not advertise the route

– If not, it adds its AS number to the end of the list, and advertises the route to the rest of its neighbors

– Arbitrary metrics can be applied

5. Lowest cost, highest speed, policy based, etc.

3.24 Routing Path Advertisements

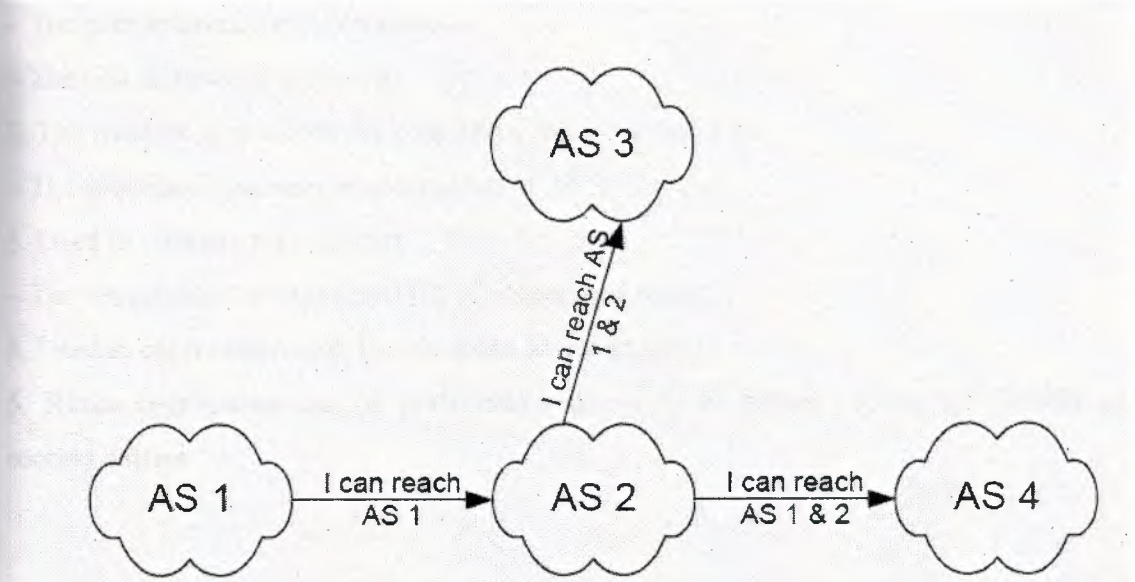


Figure 3.13. Routing Path Advertisements

3.25 Border Router State

1. Each border router maintains a list of all AS's that it can reach

2. This information is periodically transmitted to border routers at neighboring AS's with BGP updates

3. Since there are thousands of AS's, these lists can grow *very* large

– Thus, TCP is used for reliable delivery

4. BGP-4 introduced route aggregation with CIDR

– Networks with common prefixes can be aggregated "supernetted" into a since route entry

– Route advertisement size is reduced

3.26 Advertising Aggregated Routes

1. A problem with advertising aggregated routes

– The path information is obscured

– The hop distance is obscured

2. The solution is to divide the path into a *sequence* and a *set*

– The sequence contains the ordered list of AS's

3. Used to estimate hop distance

– The set contains an unordered list of aggregated routes

4. Used in conjunction with the sequence to detect loops

5. Route aggregation can be performed recursively to further reduce the number of routing entries

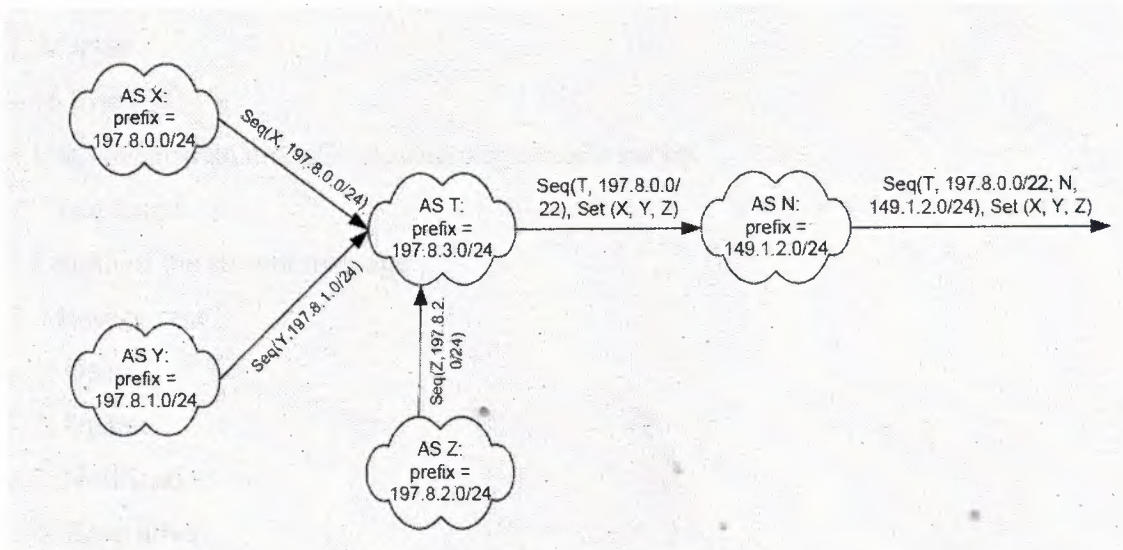


Figure 3.13. Route Aggregation

3.27 BGP Common Header Format

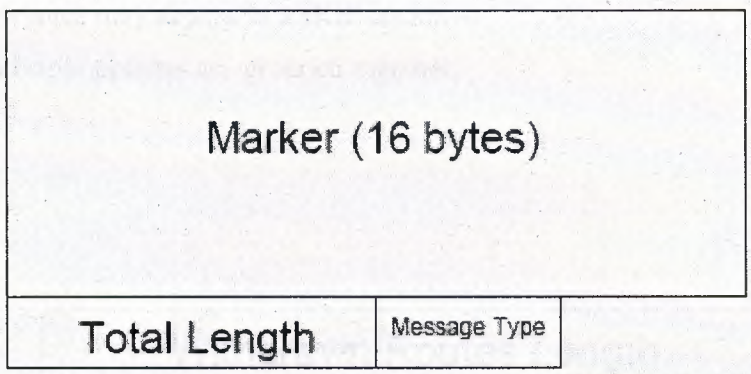


Figure 3.14.BGP Common Header Format

1. Marker

– 16 bytes, all 1's

– Used to separate multiple messages in a single packet

2. Total length

– Length of the current message

3. Message type

– 1: Open

– 2: Update

– 3: Notification

– 4: Keep alive

3.28 BGP Message Types

1. The 4 BGP message types

– OPEN

2. Initiates a session

– KEEPALIVE

3. Heartbeat for an established session

– UPDATE

4. Sends routing tables and associated information to a peer

– NOTIFICATION

5. Sends an error message to a peer

6. Multiple messages may appear in a BGP packet

– Typically, multiple updates are grouped together

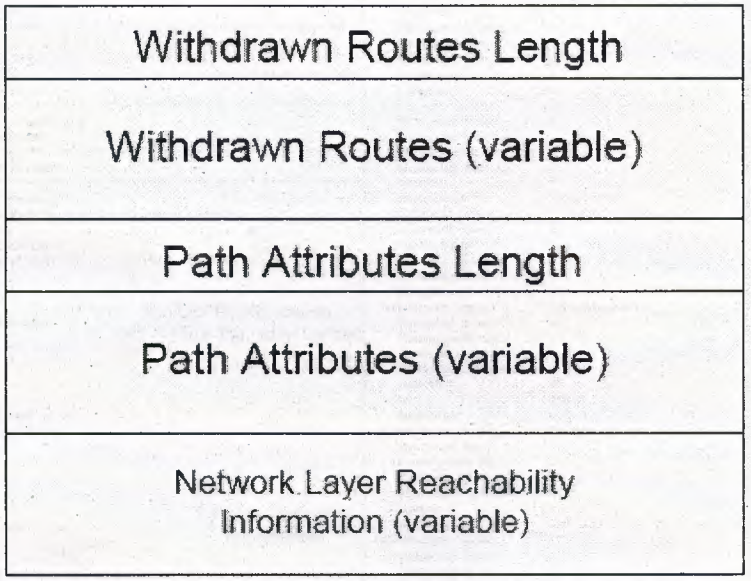


Figure 3.15.BGP UPDATE Packet Format

7. Withdrawn routes length

– 2 bytes

– Length of withdrawn routes field

8. Withdrawn routes

– Variable Length

– List of previously advertised routes that are no longer valid (in prefix form)

9. Path attributes length

– 2 bytes

– Length of path attributes field

10. Path attributes

– Variable length

– List of sets, sequences, and associated parameters

11. NLRI

– Variable length

– Prefixes that the path attributes apply to

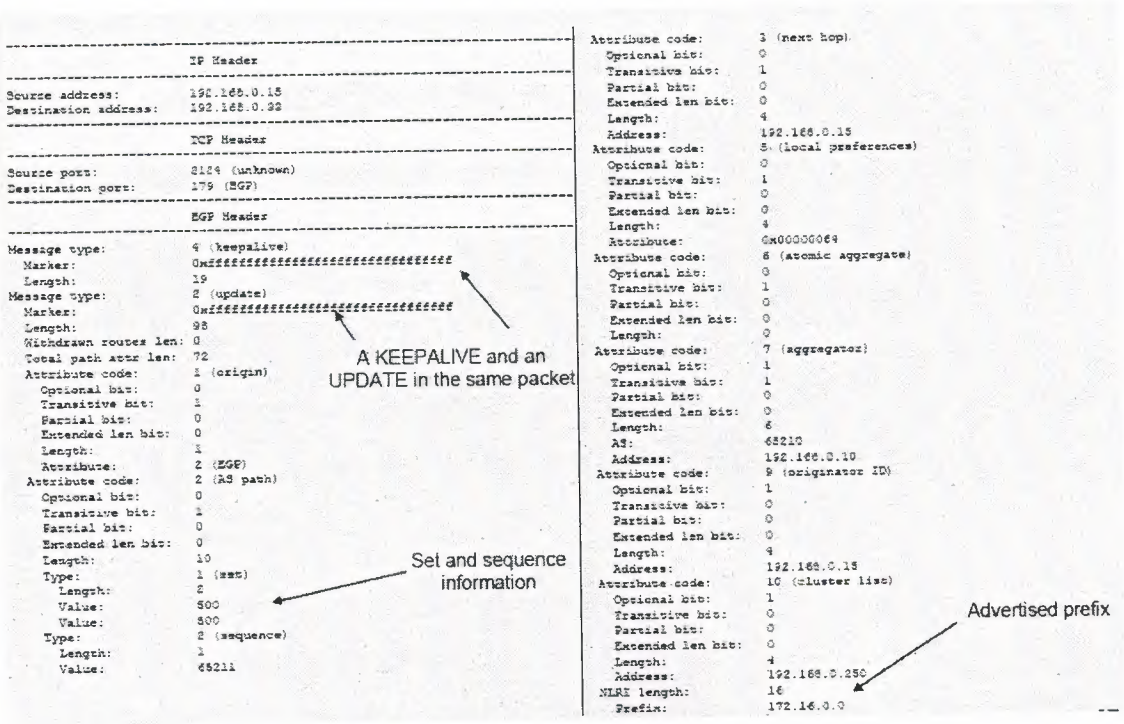


Figure 3.16. Example BGP Packet

3.29 Other Issues

1. Communicating across the AS

– All border routers in an AS typically know about all other border routers within the AS

2. A fully-connected graph

3. Large amounts of manual configuration

4. Duplicate paths to an AS?

– Routing decisions can be based on arbitrary policy

5. Hop distance

6. Politics

7. Economics

8. Etc.

9. IPv6 deployment

– Even with route aggregation, inter-domain routing tables may grow enormous

CHAPTER FOUR

Network Optimization Problems

4.1 Introduction

Linear programming problems defined on networks have many special properties. These properties allow the simplex method to be implemented more efficiently, making it possible to solve large problems efficiently. The structure of a basis, as well as the steps in the simplex method, can be interpreted directly in terms of the network, providing further insight into the workings of the simplex method. These relationships between the simplex method and the network form one of the major themes of this chapter. We use them to derive the network simplex method, a refinement of the simplex method specific to network problems.

Network problems arise in many settings. The network might be a physical network, such as a road system or a network of telephone lines. Or the network might only be a modeling tool, perhaps reflecting the time restrictions in scheduling a complicated construction project. A number of these applications are discussed in Section 2.

4.2 What Is Network Optimization

What has happened to our simple network? Do we struggle daily to make our network perform like it still has youth, stability, and endurance? Do users call the complaints: "The network is running slow"; or "I've been waiting ten minutes for my document to print"; or "Our file server just went down." Is this the same network that used to have one file server, 25 workstations, and only one network protocol?

What you have now is an in Internet work. Your network runs TCP/IP, SNA, NetWare, and NetBEUI protocols. Your simple network has grown and now comprises five separate token rings with over 400 workstations and 30 file servers.

The easy days are over. A network that used to be easy to manage now requires a complicated; intricate set of steps involving calculated technical moves to manage this monster you created-this Internet work.

Techniques are available to help you get a handle on your network-to under-stand how it is running and performing, and how it is utilizing the innovative technologies you implemented yesterday. You can also use certain techniques based on calculated measurements to implement changes so that your network will run and perform better. This is known as ***network optimization***.

Network optimization is the process of measuring to a defined level a network's workload characteristics and then making modifications to the network's layout, design, and configuration to improve its overall performance.

Most often network optimization involves using a ***protocol analyzer*** to evaluate the operation of the complete network, including all its hardware and software components. After evaluating the operational state of the network, the next step is to tune all the components. The goal is to make the network components work together so that your Internet work can perform the mission critical operations for which it was intended-at a higher performance level.

Protocol analyzers enable you to optimize and view data on your network to help you understand how that data is performing. Protocol analyzers are mostly independent of protocols, the network operating stem you are running, or the of applications on your network. Protocol analyzers need to be configured, and you need a certain skill set to use them effectively; nonetheless, these tools are extremely valuable.

It is important to understand that a protocol analyzer actually enables you to examine the protocols on a network. The ***protocols*** are the actual transmission vehicles used to get data from one point to another within the Internet work, regardless of the entity doing the transmitting on the network. The only way to actually view and troubleshoot any problems in the protocols themselves is with a protocol analyzer.

Some tools available today enable you to view just the statistics from the protocols and how they affect the network. These tools could be appropriately labeled ***monitoring tools***, but they are not true protocol analyzers. Protocol analyzer is more in depth than a monitoring tool, and enables you to view the internals of the protocol and its operations.

4.3 Network Modification Analysis

After any major network modification, a critical final step is to retest the network with a protocol analyzer to get a benchmark of the network's performance level. Only one change should be made to a network at a time, followed by retesting the network with a protocol analyzer to see the effect on performance.

Testing the network before and after the change enables you to evaluate and document the performance effect of the network modification. For example, if a protocol analyzer captured a particular workstation to file server read at one minute and there now has been a 16-megabytes upgrade in file server memory~, the network read should be remeasured; this is called post analysis. After the analyzer test is complete, the results of the post analysis show a workstation4~file-server read of 20 second this is a 40 second increase in performance for the particular task.

4.4 Measuring Network Application Efficiency

Optimizing application efficiency is a goal to be established when a particular application is having problems on the network. The usual symptom is a user or group of users complaining about a particular application's performance. There are fairly easy ways to benchmark the application's performance with respect to its standard configuration as it is documented, and how it is actually operating on a particular network.

Some protocol analyzers and network monitoring tools can be used to trap certain application access events. After a particular application event is captured, you can determine if it is performing to its optimum level.

For all applications, some application processing occurs in the workstation, and some occurs in the file server. Through examination with a protocol analyzer of a complete traffic event of an application access, an analyst can watch the request to the file server for the application, the access from the file server, and the return to the particular workstation. The specific interprocess times at the workstation and at the file server can be stamped and benchmarked for their actual interprocess time.

File search processes also occur for the application, and certain hand-off- techniques are available for reading and writing files across the network involved. A protocol analyzer helps to pinpoint whether the application is working the way it was originally intended by

the designer or manufacturer. The efficiency of the application can only be measured against its original specifications from the designer or manufacturer; if a particular application is performing at its optimum level, that specifically needs to be verified.

Additionally, some applications may not work well on certain internet work configurations. The network operation staff may have a perception of how an application will work, but the analyst must capture some events of the application's actual operation on the network to verify its overall integrity. It is possible that a certain application can contribute to bandwidth peaks on a network. Figure 3.2 depicts a protocol analyzer measuring an application load on a network.

At times, applications may perform general file searches inefficiently. These are easily picked up by examining the network for multiple file searches during searches for an application. Application tuning approaches can be used to examine ways to reconfigure an application so that it performs at its peak.

4.5 Sizing a Network Communication Link

An important goal of network optimization is to properly size all the communication links in a network. For instance, in WANs , it may be necessary to evaluate the size of a certain wide area network point to wide area network point link. But even with small networks, some communication links from building to building may not be maximized to their optimum level.

A variety of optimization techniques are discussed later in this book that you can use to examine the communication upon a wide area or local area network. Basically, communication link sizing involves looking at the capacity of the communication link with a protocol analyzer, and then utilizing the protocol analyzer to calculate how much of the communication link actually is being used.

For example, if a 56Kbps link connects two wide area network points, you can use a protocol analyzer to examine how much of that ^{56}Kbps link is being used. If the analyzer captures a set of transmission events for one second on the 56Kbps link, and analysis finds that only 32Kbps is actually being transmitted in one second across the link, a total of approximately 24Kbps of actual data space for timing can be implemented upon the communication link to maximize its capacity. Figure 3.4 depicts an example of this approach for maximizing a communication link.

Note that the goal is not always to completely maximize the capacity of a communication link-that in itself can cause a performance problem-but to actually use as much of the capacity as possible without taking a performance hit. In this example, it is reasonable to say that at least lO-15KB of space is available on this particular 56KB link. Optimization of this network link would involve utilizing as much as possible of that available capacity.

4.6 Network Flow Problem

The most general network optimization problem that we treat in this chapter is called the minimum cost network flow problem. It is a linear program of the form.

$$\text{Minimize} \quad Z = C^T x$$

$$X$$

$$\text{Subject to} \quad Ax = b$$

$$\lambda \leq x \leq u$$

Where £ and u are vectors of lower and upper bounds on x. We allow components λ and u to take on the values $-\infty$ and $+\infty$, respectively, to indicate that a variable can be arbitrarily small or large.

The notation for describing network problems is slightly different than for the programs we have discussed so far. The network has seven (the small black circles) and eleven arcs connecting the nodes. The nodes numbered 1—7. An arc between nodes i and j is denoted as (i, j). So, for example, this network includes the arcs (1, 2) and (4, 5). In this example, the existence of an arc (i, j) means that it is possible to drive from node i to node j, but not from node j to node i There is a difference between arc (i, j) and arc(j, i). For each arc (i, j) the linear program will have a corresponding variable $x_{i,j}$, and cost coefficient $c_{i,j}$. The variable x records the flow in arc(i,j) of the network, and for this

application might represent the number of cars on a road. In this problem there are eleven variables, one for each road.

In the general network problem there will be a variable for each arc in the network, and an equality constraint for each node. We assume that there are m nodes and n arcs in the network, so that A is an m x n matrix. The bounds on the variables represent the upper and lower limits on flow on an arc. Often the lower bound will be zero.

The i-th row of the constraint matrix A corresponds to a constraint at the i-th node:

$$\text{(Flow out of Node i)} - \text{(Flow into Node i)} = b_i ,$$

or in algebraic terms

$$\sum x_{i,j} - \sum X_{k,i} = b_i,$$

This reverses the more common usage of n for the number of nodes and m for the number of arcs that is used in many references on network problems. Where n refers to the number of variables and m refers to the number of constraints.

Where the respective summations are taken over all arcs leading out of and into node i If b. > 0 then node i is called a source since it adds flow to the network If b <0 then the node is called a sink since it removes flow from the network. If b, = 0 then the node is called a transshipment node, a node where flow is conserved. A component c of the cost vector $c_{i,j}$ records the cost of shipping one unit of flow over arc (i,j).

Example 4.1: Network Linear Program

We now use it to represent flows of oil through pipes. Suppose that *50* barrels of oil are being produced at node 1, and that they must be shipped through a system of pipes to nodes 6 and 7 (20 barrels to node 6 and 30 barrels to node 7). The costs of pumping a barrel of oil along each arc are marked on the figure. The flow on each arc has a lower bound of zero and an upper bound of 30.

Node 1 is a source and nodes 6 and 7 are sinks. The other nodes are transshipment nodes. The cost of each arc is marked in the figure. The corresponding minimum cost linear program is

Minimize $z = 12X_{1,2} + 15X_{1,3} + 9X_{2,4} + 8X_{2,6} + 7X_{3,4} + 6X_{3,5} + 8X_{4,5} + 4X_{4,6} + 5X_{5,6} + 3X_{5,7} + 11X_{6,7}$

X

Supject to

$$x_{1,2} + x_{1,3} = 50$$

$$x_{2,4} + x_{2,6} - x_{1,2} = 0$$

$$x_{3,4} + x_{3,5} + x_{1,3} = 0$$

$$x_{4,5} + x_{4,6} - x_{2,4} - x_{3,4} = 0$$

$$x_{5,6} + x_{5,7} - x_{3,5} - x_{4,5} = 0$$

$$x_{6,7} - x_{2,6} - x_{4,6} - x_{5,6} = 20$$

$$-x_{5,7} - x_{6,7} = 30$$

$$0 \leq x \leq 30 .$$

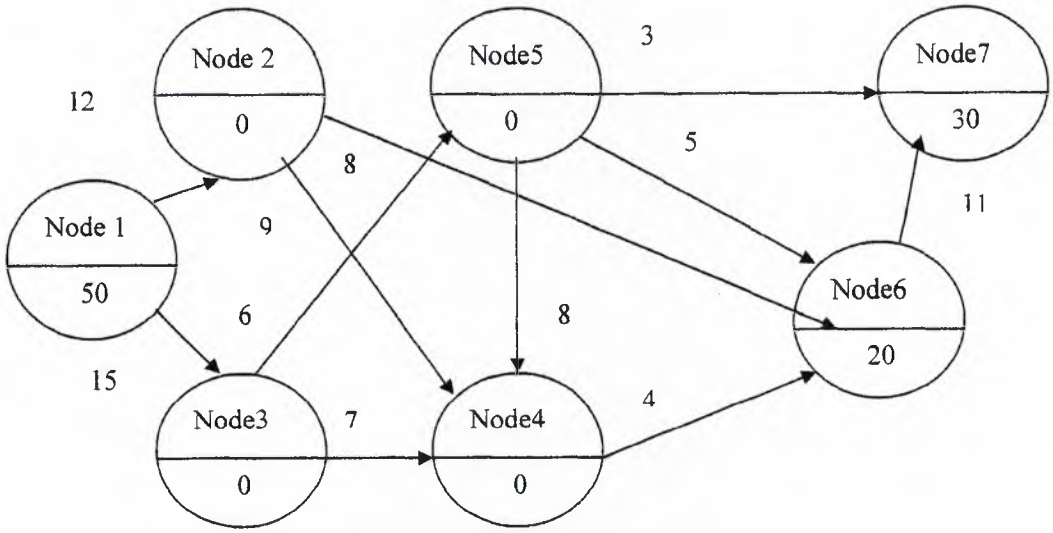


Figure 4.1.Minimization (Network flow problem)

Table 4.1.Show the solution using **WINQSB** Package

From/To	Node1	Node2	Node3	Node4	Node5	Node6	Node7	Supply
Node1		12	15					50
Node2				9		8		0
Node3				7	6			0
Node4					8	4		0
Node5						5	3	0
Node6							11	0
Node7								0
Demand	0	0	0	0	0	20	30	

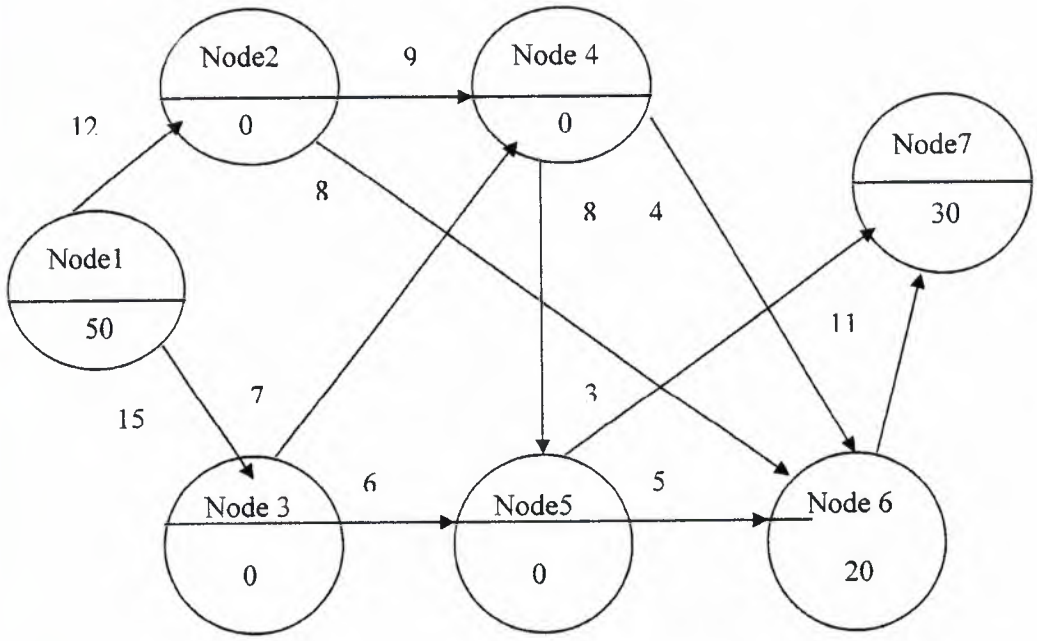


Figure 4.2. Minimization(Network Flow problem)

Table4.2. Show the Solution using WINQSB Package

01-03-2005	From	To	Flow	Unit Cost	Total Cost	Reduced Cost
1	Node 1	Node 2	20	12	240	0
2	Node 1	Node 3	30	15	450	0
3	Node 2	Node 6	20	8	160	0
4	Node 3	Node 5	30	6	180	0
5	Node 5	Node 7	30	3	90	0
	Total	Objective	Function	Value=	1120	

The constraints are listed in order of node number, where the left-hand side of the constraint corresponds to (flow out) — (flow in). If we order the variables as in the objective function, then in matrix-vector form the linear program could be written with cost vector

$$C = (12\ 15\ 9\ 8\ 7\ 6\ 8\ 4\ 5\ 3\ 11)^T$$

Right-hand side vector

$$C = (50\ 0\ 0\ 0\ 0\ -20\ -30)^T$$

A=

$$\begin{pmatrix} 1 & 1 & & & & & & & & & \\ -1 & & 1 & 1 & & & & & & & \\ & -1 & & & 1 & 1 & & & & & \\ & & -1 & -1 & & & 1 & 1 & & & \\ & & & & -1 & -1 & & & 1 & 1 & \\ & & -1 & & & & -1 & -1 & & & 1 \\ & & & & & & & & -1 & -1 & \end{pmatrix}$$

Each column A has two nonzero entries , +1 and -1. Each column corresponds to an arc (i,j) , +1 appears in row i and -1 in row j to indicate that the arc carries flow out of node i and into node j.

The total supply in the network is given by the formula

$$S = \sum_{(i,j)\in A} bi$$

And the total demand by

$$D = -\sum_{(i,j)\in A} bi$$

Will assume that S = D, that total supply equals total demand. A network always be modified to guarantee this. If S > D, that is, there is excess iy, then an artificial node is added to the network with demand S — D, and cia! arcs are added connecting every source to this artificial node; each such Such arc has its associated cost coefficient equal to zero. (This assumes that there is cost associated with excess production.) If there is excess demand, then an

Node is added with supply D — 5, together with artificial arcs connecting artificial node with every sink. These new arcs have cost coefficients that respond to the cost (if any) of unmet demand.

Example 4.2; Ensuring that Total Supply Equals Total Demand

If artificial sources and sinks are added appropriately, together with the assoc artificial arcs, then the networks are brought into balance. No cost has been associated with either 'supply or excess demand.

We have written above that there are lower and upper bounds on every flow:

$* \leq X \leq u$. For the sake of simplicity, when deriving the network simplex method we will assume that the variables are only constrained to be non-negative $x \geq 0$

Simplifying assumption can be made without any loss of generality. (The reason for this are outlined here, although their justification is left to the Problems).

A simple change of variables can be used to transform $* \leq x$ to $0 \leq x$ .Upper bounds can also be eliminated. The technique used to eliminate upper bounds.

Does increase the size of the linear program, which can lead to an increase in solution time. The constraint matrix A in a network problem is sparse. As observed in Example 8.1, each column of A has precisely twice nonzero entries, +1 and —1. If (i, j) is an arc in the network then the corresponding column in A has +1 in row i and —1 in row j. This implies that if all the rows of A are added together the result is a vector of all zeroes. Thus, the rank of A is at most m - 1, where in is the number of nodes in the network. We prove in the next Section that the rank of A is exactly m - 1.

If we add together all the rows in the equality constraints Ax b then, by the above remarks, the left-hand side will be zero. The right-hand side will sum to S - D, supply minus demand, and so it also will be zero. Therefore, any one of the constraints is redundant. The rank deficiency in A does not lead to an inconsistent system of linear equations.

There are a number of special forms of the minimum-cost network flow problem that are of independent interest. Special-purpose algorithms have bee developed for these problems, some of which are dramatically more efficient than the general-purpose simplex method. For this reason, giving them individualized treatment has resulted in many practical benefits. We will not discuss these special-purpose algorithms in this book. But we mention some of these special problems to give some idea of the range of applications of network models.

CONCLUSION

In this thesis, it has given a detailed view on the network routing and its application; it also has provided a solution of the information routing problem using Routing Information Protocol (RIP), and Open Shortest Path First (OSPF), as a dynamic solutions of the problem , also has provided a solution of the optimization problems.

Next paragraph the important result obtained from this thesis:

- In computer networks, each router has a table that shows where to send each incoming packet. This table is created when the router comes online, and used for all packets. A system that updates this routing table periodically is called a dynamic system, whereas a system that does not change the routing table is called a static system.

- Routing algorithms are the algorithms which decide the route a packet is going to follow namely its source to its destination.

- In computer network routing link state better than distance-vector, that is link state of routing is Fast convergence, Loopless convergence, Supports of multiple metrics, and supports for multiple equivalent paths.

- Network optimization is the process of measuring to a defined level a network's workload characteristics and then making modifications to the network's layout, design, and configuration to improve its overall performance.

- Network problem there will be a variable for each arc in the network, and an equality constraint for each node.
- Does increase the size of the linear program, which can lead to an increase in solution time.

REFERENCES

[1] F. Baker, "Requirements for IP Version 4 Routers," Internet RFC 1812, Jun. 1995

[2] C. Hedrick, "Routing Information Protocol," Internet RFC 1058, Jun. 1988.

[3] C. Huitema, *Routing in the Internet*, 2nd Edition, Prentice Hall, 2000.

[4] G. Malkin, "RIP Version 2," Internet RFC 2453, Nov. 1998.

[5] P. Miller, *TCP/IP* Explained, Digital Press, 1997.

[6] J. Moy, "OSPF Version 2," Internet RFC 2328, Apr. 1998.

[7] J. Moy, *OSPF: Anatomy of a Routing Protocol*, Addison-Wesley, 1998.

[8] C. Partridge, "Designing and Building Gigabit and Terabit Internet Routers," *SIGCOMM '98*

Tutorial, Sep. 1998.

[9] Y. Rekhter and T. Li, "A Border Gateway Protocol (BGP-4)," Internet RFC 1771, Mar. 1995.

[10] Y. Rekhter and P. Gross, "Application of the Border Gateway Protocol in the Internet," Internet RFC 1772, Mar. 1995.

[11] R. Stevens, *TCP/IP Illustrated, Vol. 1*, Addison-Wesley, 1994.

[12] J. W. Stewart, *BGP4: Inter-Domain Routing in the Internet*, Addison-Wesley, 1999.

[13] D. P. Bertsekas and R. Gallager. *Data Networks*. Prentice-Hall, 1992. Second Edition.

[14] R. Bott and R. J. Duffin. On the algebra of networks. *Transactions of the AMS*, 74:99–109, 1953.

[15] D. Braess. Uber ein paradoxon der verkehrsplanung. *Unternehmensforschung*, 12:258–268, 1968.