Module 6

Internetworking

Lesson 3

Transport and Application Layer Protocols

Specific Instructional Objectives

At the end of this lesson, the students will be able to:

- Explain how UDP allows two applications running in two remote locations can communicate
- State the limitations of UDP
- Explain how TCP provides connection-oriented service
- Explain how TCP incorporates reliability in internet communication
- Explain how DNS allows the use of symbolic names instead of IP address
- Explain the use of client-server model for
    - Remote login
    - Mail transfer
    - File transfer

6.3.1 Introduction

So far we have discussed the delivery of data in the following two ways:

- **Node-to-node delivery**: At the data-link level, delivery of frames take place between two nodes connected by a point-to-point link or a LAN, by using the data-link layers address, say MAC address.
- **Host-to-host delivery**: At the network level, delivery of datagrams can take place between two hosts by using IP address.

From user's point of view, the TCP/IP-based internet can be considered as a set of application programs that use the internet to carry out useful communication tasks. Most popular internet applications include Electronic mail, File transfer, and Remote login. IP allows transfer of IP datagrams among a number of stations or hosts, where the datagram is routed through the internet based on the IP address of the destination. But, in this case, several application programs (processes) simultaneously running on a source host has to communicate with the corresponding processes running on a remote destination host through the internet. This requires an additional mechanism called *process-to-process delivery*, which is implemented with the help of a transport-level protocol. The transport level protocol will require an additional address, known as *port number*, to select a particular process among multiple processes running on the destination host. So, there is a requirement of the following third type of delivery system.

- **Process-to-process delivery**: At the transport level, communication can take place between processes or application programs by using port addresses

Basic communication mechanism is shown in Fig. 6.3.1. The additional mechanism needed to facilitate multiple application programs in different stations to communicate with each other simultaneously can be provided by a transport level protocol such as UDP or TCP, which are discussed in this lesson.

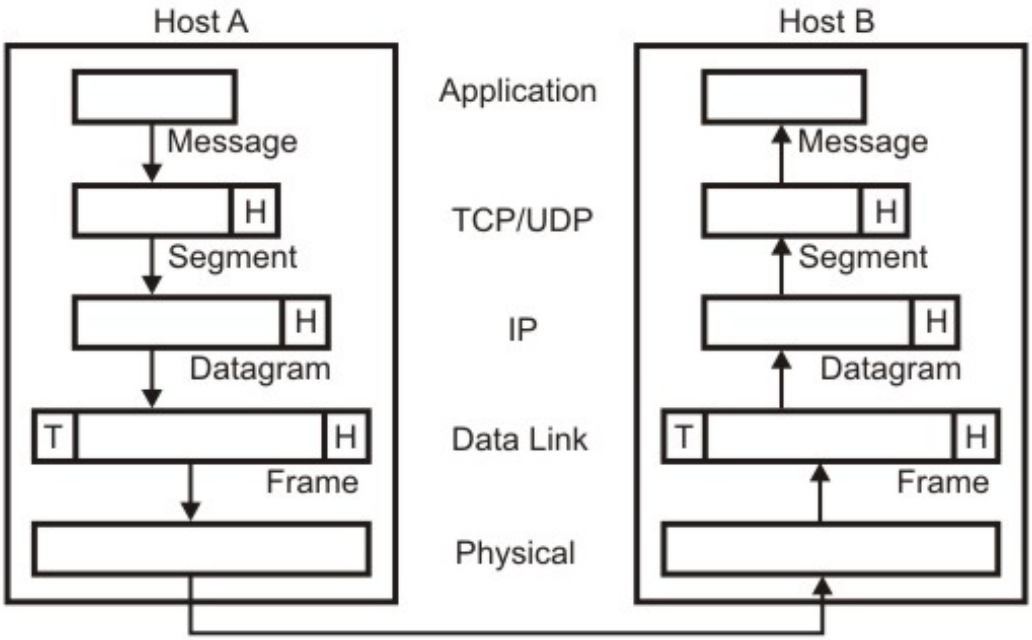


Figure 6.3.1 Communication mechanism through the internet

6.3.2 User Datagram protocol (UDP)

UDP is responsible for differentiating among multiple source and destination processes within one host. Multiplexing and demultiplexing operations are performed using the port mechanism as depicted in Fig. 6.3.2.

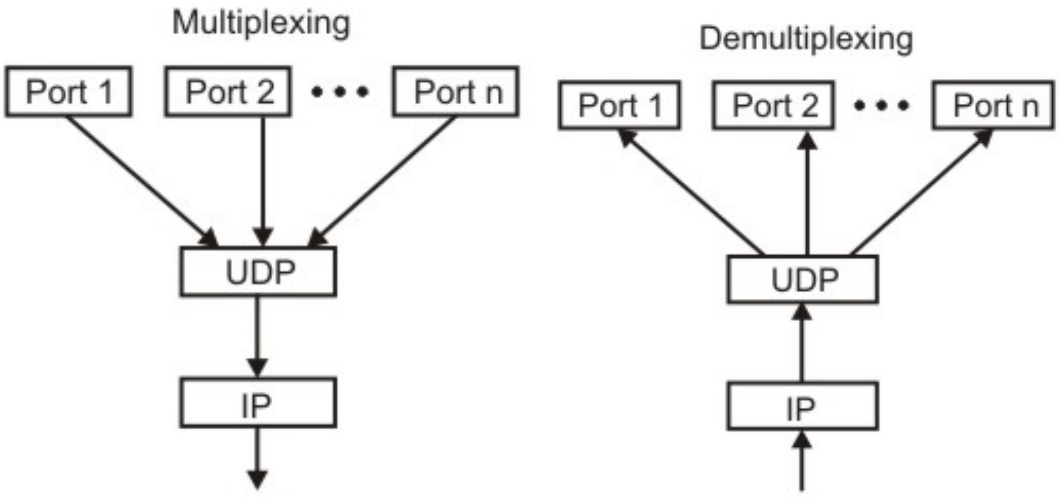


Figure 6.3.2 Multiplexing and demultiplexing mechanism of UDP

Port Numbers

Transport layer address is specified with the help a 16-bit Port number in the range of 0 and 65535. Internet Assigned Number Authority (IANA) has divided the addresses in three ranges:

- **Well-known ports:** The ports in the range from 0 to 1023 are assigned and controlled by IANA. These port numbers are commonly used as universal port numbers in the servers for the convenience of many clients the servers serve. Some commonly used well-known ports used with UDP is given in Table 6.3.1.
- **Registered ports**: Registered ports in the range from 1024 to 49151 are not assigned or controlled by IANA. However, they can only be registered with IANA to avoid duplication.
- **Dynamic ports**: Dynamic ports (49152 to 65535) are neither controlled by IANA nor need to be registered. They can be defined at the client site and chosen randomly by the transport layer software.

Table 6.3.1 Well-known ports used by UDP

Port	*Protocol*	*Description*
7	Echo	Echoes a received datagram back to the sender
9	Discard	Discards any datagram that is received
11	Users	Active users
13	Daytime	Returns the date and the time
17	Quote	Returns a quote of the day
19	Chargen	Returns a string of characters
53	Nameserver	Domain Name Service
67	Bootpc	Server port to download bootstrap information
68	Bootpc	Client port to download bootstrap information
69	TFTP	Trivial File Transfer Protocol
111	RPC	Remote Procedure Call
123	NTP	Network Time Protocol
161	SNMP	Simple Network Management Protocol
162	SNMP	Simple Network Management Protocol (trap)

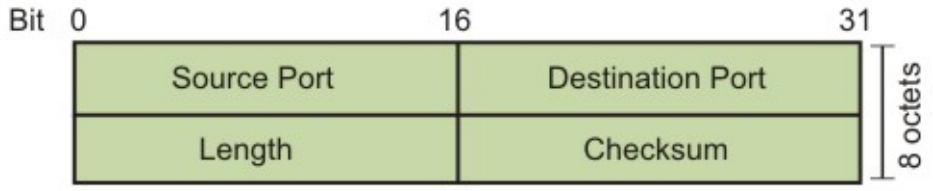


Figure 6.3.3 UDP Datagram Format

UDP Datagram

The UDP datagram format is shown in Fig. 6.3.3. A brief description of different fields of the datagram are given below:

- Source port (16 bits): It defines the port number of the application program in the host of the sender
- Destination port (16 bits): It defines the port number of the application program in the host of the receiver
- Length: It provides a count of octets in the UDP datagram, minimum length = 8
- Checksum: It is optional, 0 in case it is not in use

Characteristics of the UDP

Key characteristics of UDP are given below:

- UDP provides an unreliable connectionless delivery service using IP to transport messages between two processes
- UDP messages can be lost, duplicated, delayed and can be delivered out of order
- UDP is a thin protocol, which does not add significantly to the functionality of IP
- It cannot provide reliable stream transport service

The above limitations can be overcome by using connection-oriented transport layer protocol known as *Transmission Control Protocol* (TCP), which is presented in the following section.

6.3.3 Transmission Control Protocol (TCP)

TCP provides a connection-oriented, full-duplex, reliable, streamed delivery service using IP to transport messages between two processes.
Reliability is ensured by:

- Connection-oriented service
- Flow control using sliding window protocol
- Error detection using checksum
- Error control using go-back-N ARQ technique
- Congestion avoidance algorithms; multiplicative decrease and slow-start

TCP Datagram

The TCP datagram format is shown in Fig. 6.3.4. A brief explanation of the functions of different fields are given below:

- Source port (16 bits): It defines the port number of the application program in the host of the sender
- Destination port (16 bits): It defines the port number of the application program in the host of the receiver
- Sequence number (32 bits): It conveys the receiving host which octet in this sequence comprises the first byte in the segment
- Acknowledgement number (32 bits): This specifies the sequence number of the next octet that receiver expects to receive
- HLEN (4 bits): This field specifies the number of 32-bit words present in the TCP header
- Control flag bits (6 bits): URG: Urgent pointer
- ACK: Indicates whether acknowledge field is valid
- PSH: Push the data without buffering
- RST: Resent the connection
- SYN: Synchronize sequence numbers during connection establishment
- FIN: Terminate the connection
- Window (16 bits): Specifies the size of window
- Checksum (16 bits): Checksum used for error detection.
- User pointer (16 bits): Used only when URG flag is valid
- Options: Optional 40 bytes of information

The well-known ports used by TCP are given in Table 6.3.2 and the three types of addresses used in TCP/IP are shown in Fig. 6.3.5. TCP establishes a virtual path between the source and destination processes before any data communication by using two procedures, *connection establishment* to start reliably and *connection termination* to terminate gracefully, as discussed in the following subsection.

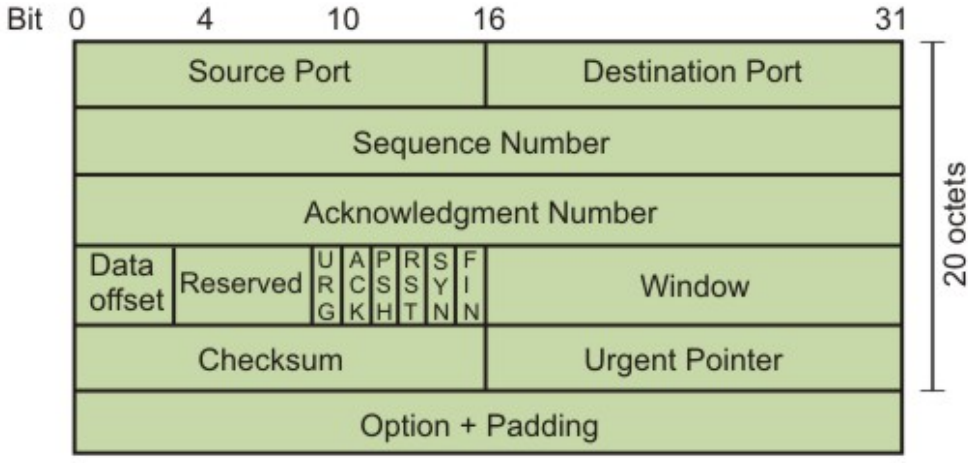


Figure 6.3.4 The TCP datagram format

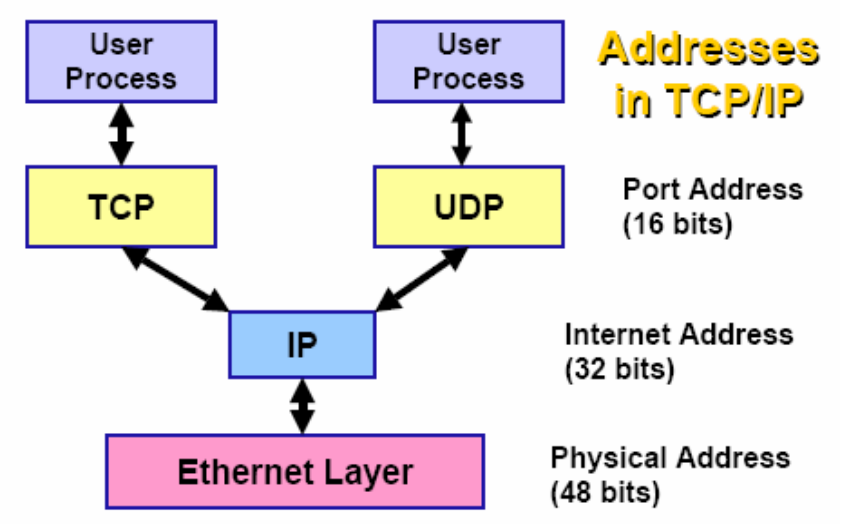


Figure 6.3.5 Three types of addresses used in TCP/IP

Table 6.3.2 Well-known ports used by TCP

Port	*Protocol*	*Description*
7	Echo	Echoes a received datagram back to the sender
9	Discard	Discards any datagram that is received
11	Users	Active users
13	Daytime	Returns the date and the time
17	Quote	Returns a quote of the day
19	Chargen	Returns a string of characters
20	FTP, Data	File Transfer Protocol (data connections)
21	FTP, Control	File Transfer Protocol (control connection)
23	TELNET	Terminal Network
25	SMTP	Simple Mail Transfer Protocol
53	DNS	Domain Name Server
67	BOOTP	BOOTP Protocol
79	Finger	Finger
80	HTTP	Hypertext Transfer Protocol
111	RPC	Remote Procedure Call

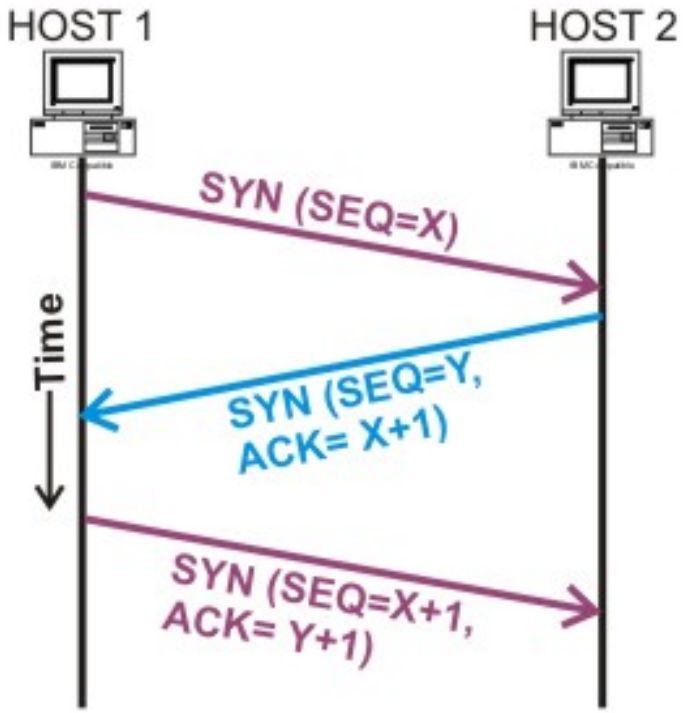
Connection-oriented service

TCP performs data communication in full-duplex mode, that is both the sender and receiver processes can send segments simultaneously. For connection establishment in full-duplex mode, a four-way protocol can be used. However, the second and third steps can be combined to form a three-way handshaking protocol with the following three steps as shown in Fig. 6.3.6.

Step 1: The client sends SYN segment, which includes, source and destination port numbers, and an *initialization sequence number* (ISN), which is essentially the byte number to be sent from the client to the server.

Step 2: The server sends a segment, which is a two-in-one segment. It acknowledges the receipt of the previous segment and it also acts as initialization segment for the server.

Step3: The sends an ACK segment, which acknowledges the receipt of the second segment



X, Y = Initialization sequence numbers

Figure 6.3.6 Protocol for connection establishment

Similarly for connection termination, a four-way handshaking protocol is necessary for termination of connection in both directions as shown in Fig. 6.3.7. The four steps are as follows:

Step 1: The client sends a FIN segment to the server.

Step 2: The server sends an ACK segment indicating the receipt of the FIN segment and the segment also acts as initialization segment for the server.

Step3: The server can still continue to send data and when the data transfer is complete it sends a FIN segment to the client.

Step4: The client sends an ACK segment, which acknowledges the receipt of the FIN segment sent by the server.

Both the connections are terminated after this four-way handshake protocol.

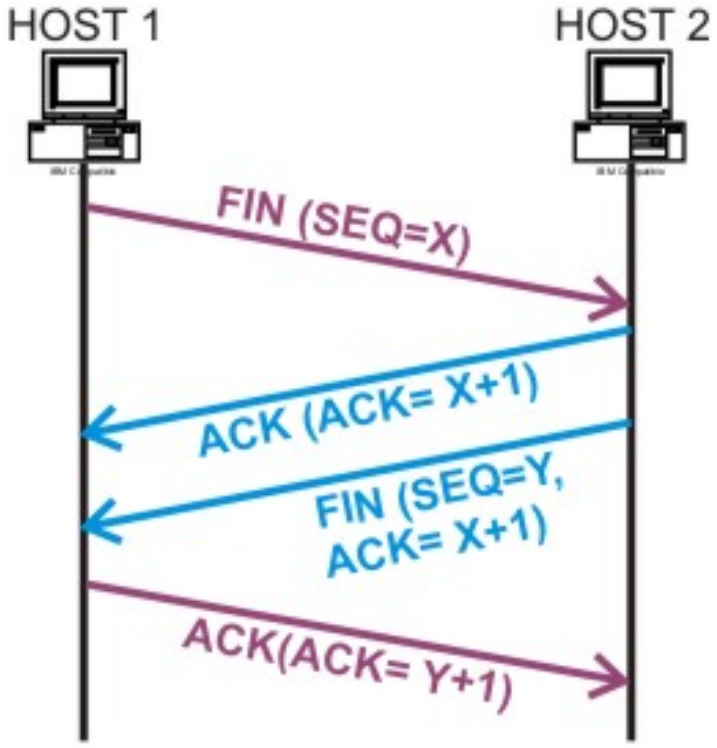


Figure 6.3.7 Protocol for connection termination

Reliable Communication

To ensure reliability, TCP performs flow control, error control and congestion control.

Flow control: TCP uses byte-oriented sliding window protocol, which allows efficient transmission of data and at the same time the destination host is not overwhelmed with data. The flow control operation is in Fig. 6.3.8. As shown in the figure, the receiver has a buffer size of 8 Kbytes. After receiving 4 K bytes, the window size is reduced to 4 Kbytes. After receiving another 3 K bytes, the window size reduces to 1 K bytes. After the buffer gets empty by 4 K bytes, the widow size increases to 7 K bytes. So it may be noted that the window size is totally controlled by the receiver window size, which can be increased or decreased dynamically by the destination. The destination host can send acknowledgement any time.

Error Control: Error control in TCP includes mechanism for detecting corrupted segments with the help of checksum field. Acknowledgement method is used to confirm the receipt of uncorrupted data. If the acknowledgement is not received before the time-out, it is assumed that the data or the acknowledgement has been corrupted or lost. It may

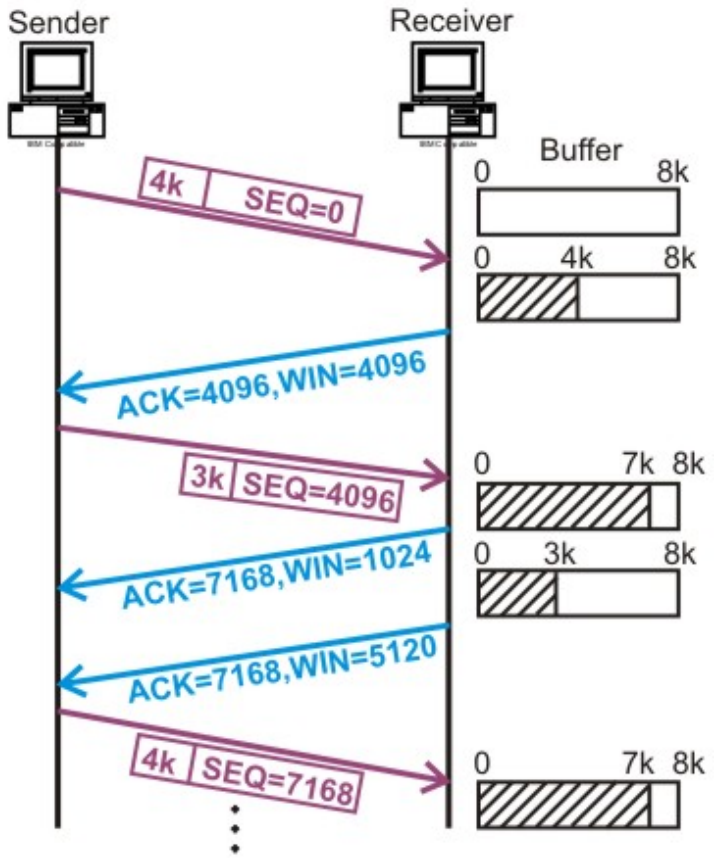


Figure 6.3.8 Flow control in TCP

be noted that there is no negative acknowledgement in TCP. To keep track of lost or discarded segments and to perform the operations smoothly, the following four timers are used by TCP:

- Retransmission; it is dynamically decided by the round trip delay time.
- Persistence; this is used to deal with window size advertisement.
- Keep-alive; commonly used in situations where there is long idle connection between two processes
- Time-waited; it is used during connection terminations

Congestion control: To avoid congestion, the sender process uses two strategies known as slow-start and additive increase, and the send one is known as multiplicative decrease as shown in Fig. 6.3.9. To start with, the congestion widow size is set to the maximum segment size and for each segment that is acknowledged, the size of the congestion window size is increased by maximum segment size until it reaches one-half of the allowable window size. Ironically, this is known as *slow-start*, although the rate of increase is exponential as shown in the figure. After reaching the threshold, the window size is increased by one segment for each acknowledgement. This continues till there is no time-out. When a time-out occurs, the threshold is set to one-half of the last congestion window size

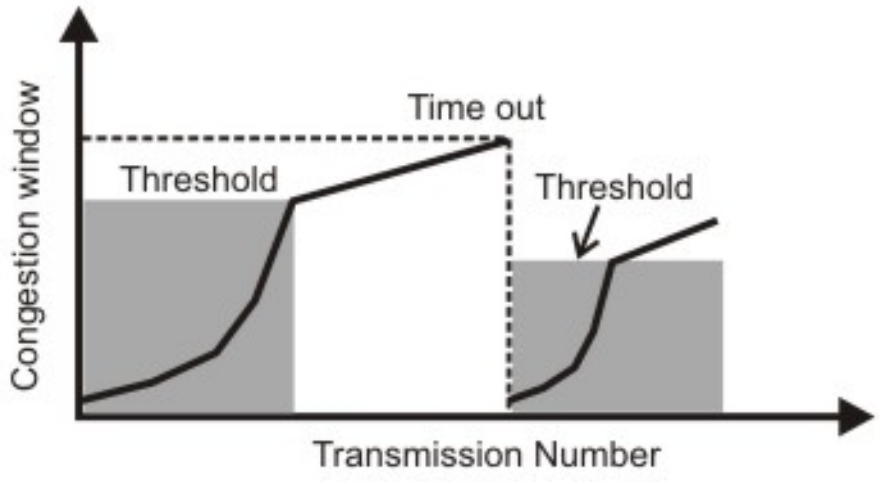


Figure 6.3.9 Congestion control in TCP

6.3.4 Client-Server Paradigm and its Applications

The way the application programs communicate with each other is based on client-server model as shown in Fig. 6.3.10. This provides the foundation on which distributed algorithms are developed. A client process formulates a request, sends it to the server and then waits for response. A server process awaits a request at a well-known port that has been reserved for the service and sends responses. Two identifiers, namely IP address and the port number, are necessary for process-to-process communication. The combination of the two is called a *socket address.* A pair of socket addresses; one of the client and the other of the server, are necessary for the transport-layer protocol. This allows multiplexing and demultiplexing by the transport layer as we have already discussed. There are several applications such as Doman Name System, Telnet, FTP, Email and SNMP, based on client-server paradigm are briefly discussed in the following subsections.

Domain Name System

Although IP addresses are convenient and compact way for identifying machines and are fundamental in TCP/IP, it is unsuitable for human user. Meaningful high-level symbolic names are more convenient for humans. Application software permits users to use symbolic names, but the underlying network protocols require addresses. This requires the use of names with proper syntax with efficient translation mechanism. A concept known as *Domain Name System* (DNS) was invented for this purpose. DNS is a naming scheme that uses a hierarchical, domain-based naming scheme on a distributed database system. The basic approach is to divide the internet into several hundred top-level domains, which come in two flavors - *generic* and *countries*. Nearly all organizations in USA, are under generic name, where each domain is partitioned into subdomains, and these are further partitioned, and so on, as represented in the form of a tree as shown in Fig. 6.3.11. The leaves of the tree represent domains that contain no subdomains, represent single hosts, or a company or contains a thousand of hosts. Naming follows

organizational boundaries, not physical networks. The hierarchical naming system, which is used by DNS has many advantages over flat addressing scheme used earlier. Key features of the two approaches are highlighted below:

Figure 6.3.10 Client-server model

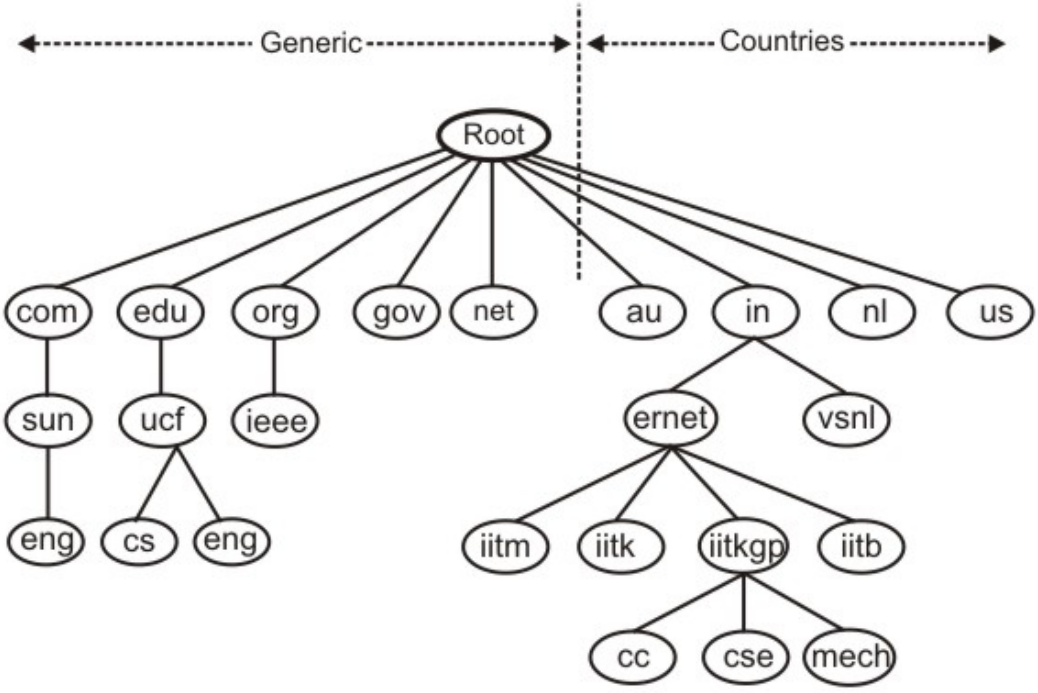


Figure 6.3.11 Partial Domain Name Space

Flat namespace

- Each machine is given a unique (by NIC) name
- Special file is used to keep name-address mapping
- All hosts must know the current mapping for all other hosts with which they want to communicate
- Large mapping file, if communication with a large number of machines is required

- Not a good scheme for communicating to arbitrary machines over large networks such as Internet

Hierarchical Namespace

- Break complete namespace into domains
- Domains broken up recursively into one or more subdomains, each of which is basically a domain again
- Further division to create any level of hierarchy – Namespace Tree
- Delegate task of name allocation/resolution of parts of the tree to distributed name servers

Name-address Resolution

Although the names used by the DNS is very convenient to humans, it cannot be used for communication through the internet. This requires mapping a name to an address known as *Name-address Resolution*. The mapping of the name to the address can be done using a *name server*, where a look-up table is maintained. A single name server could contain the entire DNS database and respond to all queries about it. However, the server would be very much overloaded and when it would fail, the entire Internet would be crippled. To avoid this problem, the entire name space is divided into non-overlapping zones. Each zone contains some part of the tree and also contains *name servers* holding the authorization information about the zone. In practice, a zone will have a primary name server and one or more secondary name servers, which get their information from the primary name servers. This is how smaller databases are maintained in a distributed manner as shown in Fig. 6.3.12.

To map a name onto an IP address, an application program calls a library procedure known as *resolver*. The resolver sends a UDP packet to a local DNS server, which searches for the name in its database. If the name is found, it returns the IP address to the resolver, which in turn informs it to the client. After having the IP address, the client then establishes a TCP connection with a destination node. However, if the local DNS server does not have the requested information, it seeks the help from other servers and finally reports back. This is known as *recursive resolution*, as shown in Fig. 6.3.13. The client may not ask for a recursive answer and in that case the mapping can be done iteratively. If a server is an authority for the name, the reply is sent. Otherwise, it sends the IP address of another server that is likely to resolve the query. The client sends query to the second server and so on. This process is known as iterative resolution.

To avoid another search when a query is received for a name that is not in its domain, the information is stored in the cash memory of the server. This mechanism is known as *caching.* This improves the efficiency of resolution. However, the mapping is not stored in the cache memory indefinitely. A *time-to-live* TTL) counter is associated with each mapping and when the time expires, the mapping is purged.

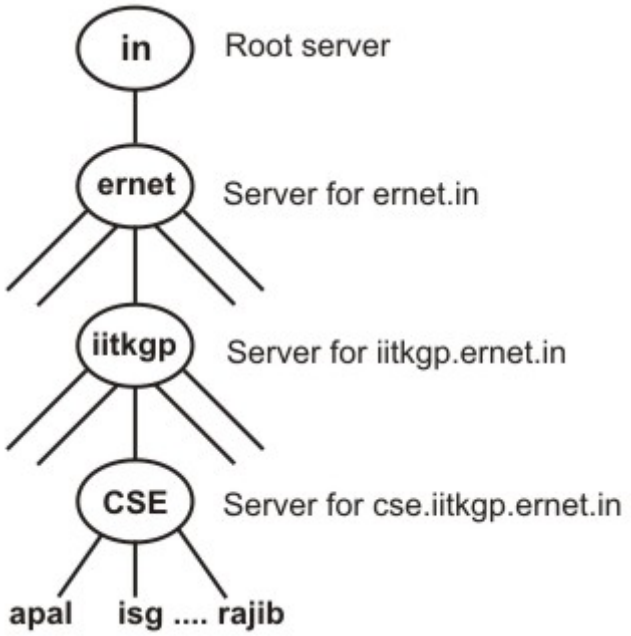


Figure 6.3.12 DNS servers

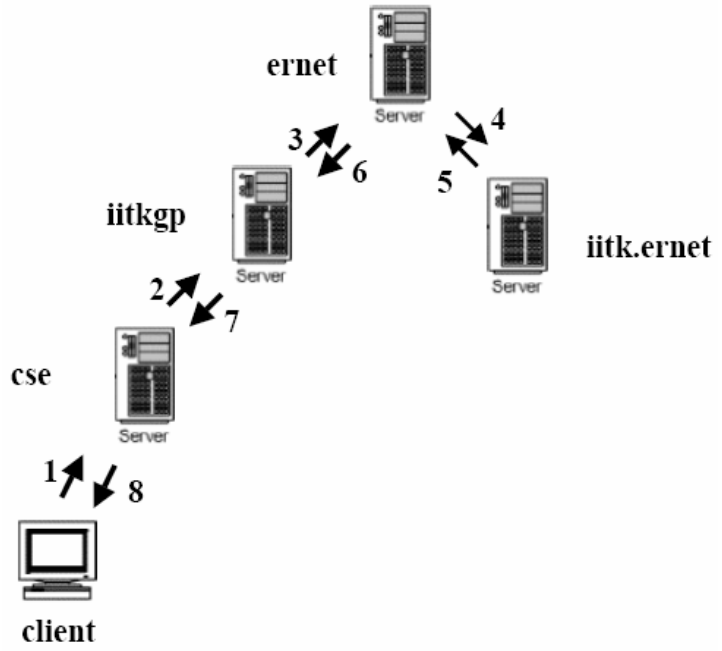


Figure 6.3.13 Recursive resolution performed in ARP protocol

Electronic Mail

Electronic mail is among the most widely available application services. Each user, who intends to participate in email communication, is assigned a mailbox, where out-going and incoming messages are buffered, allowing the transfer to take place in the

background. The message contains a header that specifies the sender, recipients, and subject, followed by a body that contains message. The TCP/IP protocol that supports electronic mail on the internet is called *Simple Mail Transfer Protocol* (SMTP), which supports the following:

- Sending a message to one or more recipients
- Sending messages that include text, voice, video, or graphics

A software package, known as *User Agent*, is used to compose, read, reply or forward emails and handle mailboxes. The email address consists of two parts divided by a @ character. The first part is the local name that identifies mailbox and the second part is a domain name.

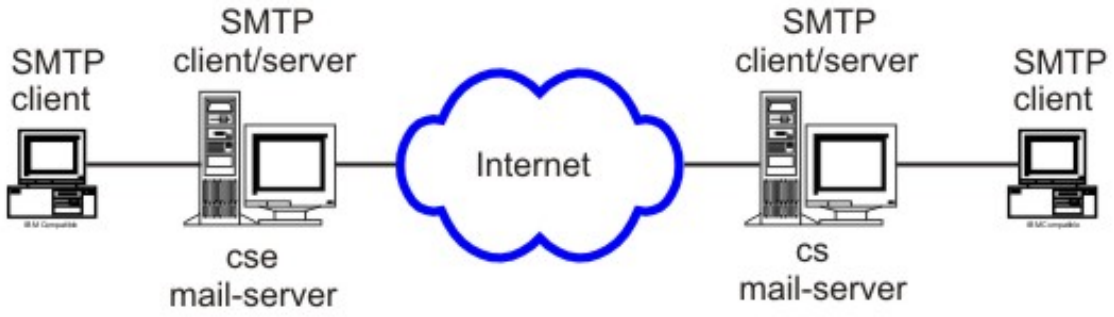


Figure 6.3.14 Simple Mail Transfer Protocol (SMTP)

Telnet

Telnet is a simple remote terminal protocol that provides a remote log-on capability, which enables a user to log on to a remote computer and behaves as if it is directly connected to it. The following three basic services are offered by TELNET:

- It defines a network virtual terminal that provides a standard interface to remote systems
- It includes a mechanism that allows the client and server to negotiate options from a standard set
- It treats both ends symmetrically

File Transfer Protocol (FTP)

File Transfer Protocol (FTP) is a TCP/IP client-server application for transfer files between two remote machines through internet. A TCP connection is set up before file transfer and it persists throughout the session. It is possible to send more than one file before disconnecting the link. A control connection is established first with a remote host before any file can be transferred. Two connections required are shown in Fig. 6.3.15. Users view FTP as an interactive system

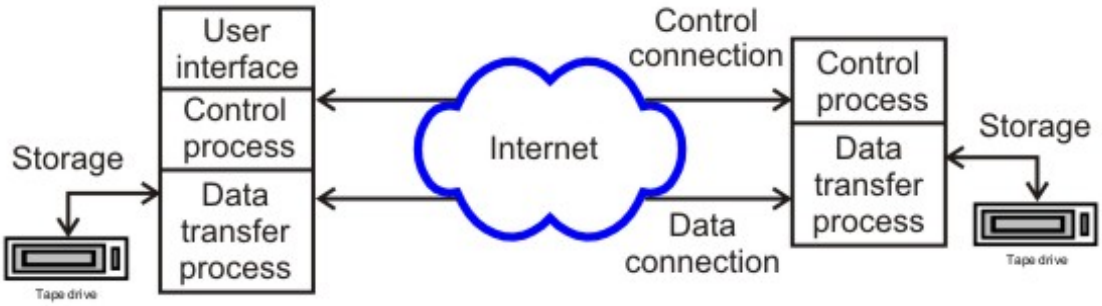


Figure 6.3.15 File Transfer Protocol (FTP)

Simple Network Management Protocol (SNMP)

Network managers use network management software that help them to locate, diagnose and rectify problems. Simple Network Management Protocol (SMTP) provides a systematic way for managing network resources. It uses transport layer protocol for communication. It allows them to monitor switches, routers and hosts. There are four components of the protocol:

- Management of systems
- Management of nodes; hosts, routers, switches
- Management of Information Base; specifies data items a host or a router must keep and the operations allowed on each (eight categories)
- Management of Protocol; specifies communication between network management client program a manager invokes and a network management server running on a host or router

Review Questions

Q1. What is the relationship between TCP/IP and Internet?
Ans: Internet is a network of different types of network. TCP/IP is a set of rules and procedures that govern the exchange of messages between hosts linked to different networks. TCP/IP creates an environment as if all hosts are connected to a single logical network.

Q2. Distinguish between TCP and UDP?
Ans: Both TCP and UDP belong to transport layer. The UDP is simpler with much less overhead. UDP provides unreliable connectionless service. On the other hand, TCP provides connection oriented reliable service with the help of suitable flow control and error control protocols. As a consequence, TCP has much more overhead.

Q3. What is the main function of UDP protocol?
Ans: UDP protocol provides user programs the ability to communicate using unreliable connectionless packet delivery service with minimum overhead.

Q4. Why pseudo-header is added in a UDP datagram?
Ans: As the UDP datagram does not contain source and destination address information, a pseudo-header is added with these information to verify that the UDP datagram has reached its correct destination.

Q5. What protocol is used by TCP for flow control?
Ans. TCP uses sliding window protocol for flow control.

Q6. What ARQ protocol is used in TCP?
Ans. Go-back-N ARQ.

Q7. What is piggybacking?
Ans. Instead of sending a separate packet for positive/negative acknowledgement, piggybacking technique utilizes the full-duplex communication environment of TCP. The positive/negative acknowledgement information is added to a normal packet sent by the receiving side. It helps to save precious network bandwidth.

Q8. How TCP establishes and terminates connection?
Ans. TCP establishes connection using a three-way handshaking protocol and connection is terminated by a 2-way/4-way handshaking protocol.

Q9. What are the advantages of DNS?
Ans: Key advantage of DNS is the use of a hierarchical naming system and the use of distributed database to store the huge amount of address information in many servers. The host that needs mapping of name to address can contact the closest server holding the required information.

Q10. What kind of paradigm is used by the application layer protocols?
Ans. Client-Server paradigm is used by all the application layer protocols.

References

1. **Douglas E. Comer, Internetworking With TCP/IP (Volime I), Rrentice Hall of India, 1995**
2. **Behrouz A. Forouzan, Data Communications and Networking, 3rd Edition, Tata McGraw-Hill Publishing Company Limited, 2004**