Advanced Networking
Unit 3

-Madhavi Dave

Reliable Stream Transport Service (TCP)

Introduction

- TCP is part of the TCP/IP Internet protocol suite.
- It is an independent, general purpose protocol that can be adapted for use with other protocols.
- TCP has been so popular that one of the International Organization for Standardization's open systems protocols, TP-4

Need for Stream Delivery

- At the lowest level, computer communication networks provide unreliable packet delivery.
- Packets can be lost or destroyed when transmission errors interfere with data, when network hardware fails, or when networks become too heavily loaded to accommodate the load presented.
- Networks that route packets dynamically can deliver them out of order, deliver them after a substantial delay, or deliver duplicates.

Need for Stream Delivery

- At the highest level, application programs often need to send large volumes of data from one computer to another.
- Using an unreliable connectionless delivery system for large volume transfers becomes tedious and annoying, and it requires programmers to build error detection and recovery into each application program.

Need for Stream Delivery

- One goal of network protocol is to find general purpose solutions to the problems of providing reliable stream delivery, making it possible for experts to build a single instance of stream protocol software that all application programs use.
- Having a single general purpose protocol helps isolate application programs from the details of networking, and makes it possible to define a uniform interface for the stream transfer service.

Properties Of The Reliable Delivery Service

- ***Stream Orientation:***
  - When two application programs (user processes) transfer large volumes of data, we think of the data as a ***stream*** of bits, divided into 8-bit ***octets,*** which are informally called ***bytes.***
  - The stream delivery service on the destination machine passes to the receiver exactly the same sequence of octets that the sender passes to it on the source machine.

Properties Of The Reliable Delivery Service

- ***Virtual Circuit Connection.***
  - Before transfer can start, both the sending and receiving application programs interact with their respective operating systems, informing them of the desire for a stream transfer. Conceptually, one application places a "call" which must be accepted by the other.
  - Protocol software modules in the two operating systems communicate by sending messages across an internet, verifying that the transfer is authorized, and that both sides are ready.
  - Once all details have been settled, the protocol modules inform the application programs that a ***connection*** has been established and that transfer can begin.
  - During transfer, protocol software on the two machines continue to communicate to verify that data is received correctly. If the communication fails for any reason, both machines detect the failure and report it to the appropriate application programs.
  - We use the term ***virtual circuit*** to describe such connections because although application programs view the connection as a dedicated hardware circuit

Properties Of The Reliable Delivery Service

- ***Buffered Transfer:***
  - Each application uses whatever size pieces it finds convenient, which can be as small as a single octet.
  - At the receiving end, the protocol software delivers octets from the data stream in exactly the same order they were sent, making them available to the receiving application program.
  - To make transfer more efficient and to minimize network traffic, implementations usually collect enough data from a stream.
  - When it reaches the receiving side, the push causes TCP to make the data available to the application without delay.

Properties Of The Reliable Delivery Service

- ***Unstructured Stream:***
  - It is important to understand that the TCP/IP stream service does not honor structured data streams.
  - For example, there is no way for a payroll application to have the stream service mark boundaries between employee records, or to identify the contents of the stream as being payroll data.
  - Application programs using the stream service must understand stream content and agree on stream format before they initiate a connection.

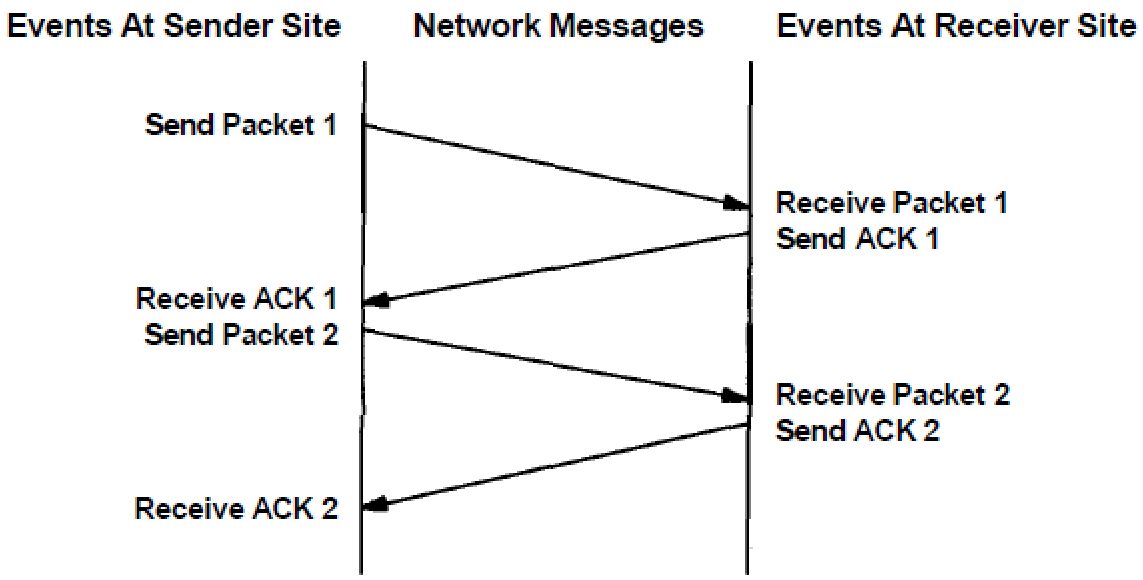
Properties Of The Reliable Delivery Service

- ***Full Duplex Connection:***
  - Connections provided by the TCP/IP stream service allow concurrent transfer in both directions. Such connections are called ***full duplex.***
  - From the point of view of an application process, a full duplex connection consists of two independent streams flowing in opposite directions, with no apparent interaction.
  - The stream service allows an application process to terminate flow in one direction while data continues to flow in the other direction, making the connection ***half duplex.***
  - The advantage of a full duplex connection is that the underlying protocol software can send control information for one stream back to the source in datagrams carrying data in the opposite direction. Such ***piggybacking*** reduces network traffic.

Providing Reliability

- Reliable stream delivery service guarantees to deliver a stream of data sent from one machine to another without duplication or data loss.
- Reliable protocols use a single fundamental technique known ***as positive acknowledgement with retransmission.***
- The technique requires a recipient to communicate with the source, sending back an ***acknowledgement*** (ACK) message as it receives data.

Simple transfer with positive ACK



Events At Sender Site

Network Messages

Events At Receiver Site

Send Packet 1

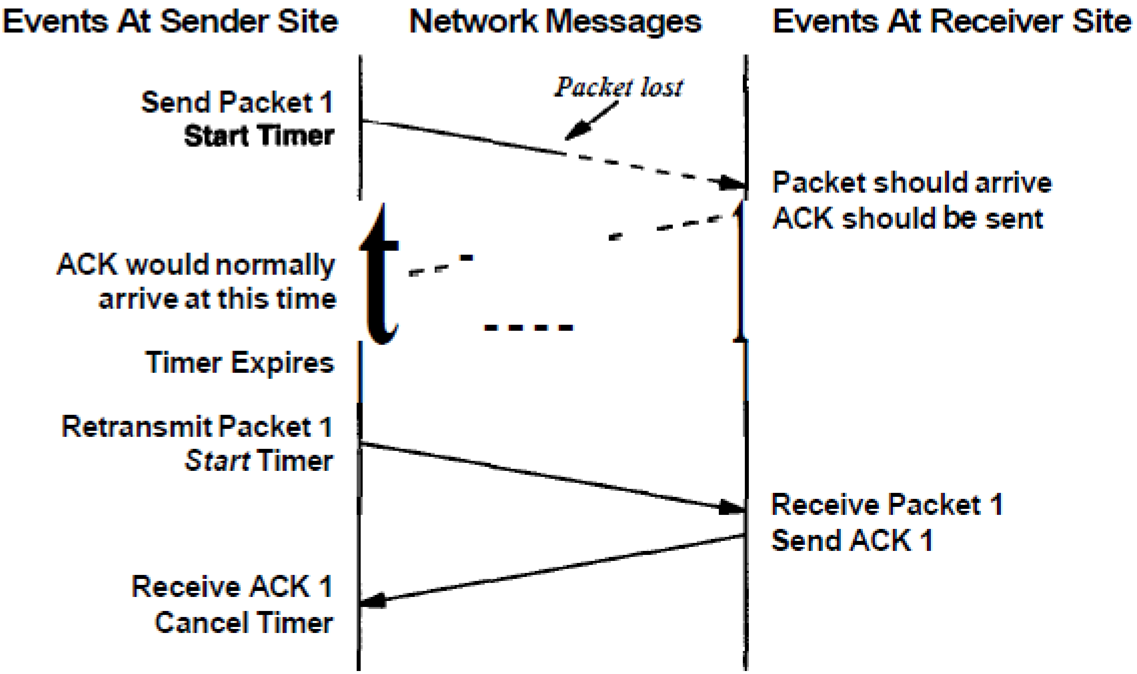
Receive Packet 1
Send ACK 1

Receive ACK 1
Send Packet 2

Receive Packet 2
Send ACK 2

Receive ACK 2

- What happens when a packet is lost or corrupted?
- The sender starts a timer after transmitting a packet.
- When the timer expires, the sender assumes the packet was lost and retransmits it.

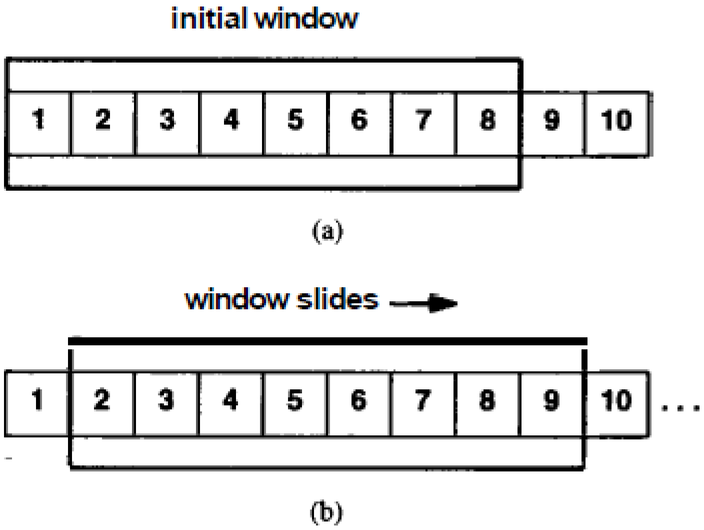

Events At Sender Site
Network Messages
Events At Receiver Site
Packet lost
Send Packet 1
Start Timer
Packet should arrive
ACK should be sent
ACK would normally
arrive at this time
Timer Expires
Retransmit Packet 1
Start Timer
Receive Packet 1
Send ACK 1
Receive ACK 1
Cancel Timer

Sliding Window Paradigm

- The sender transmits a packet and then waits for an acknowledgement before transmitting another.
- Data only flows between the machines in one direction at anytime, even if the network is capable of simultaneous communication in both directions.
- The network will be completely idle during times that machines delay responses.

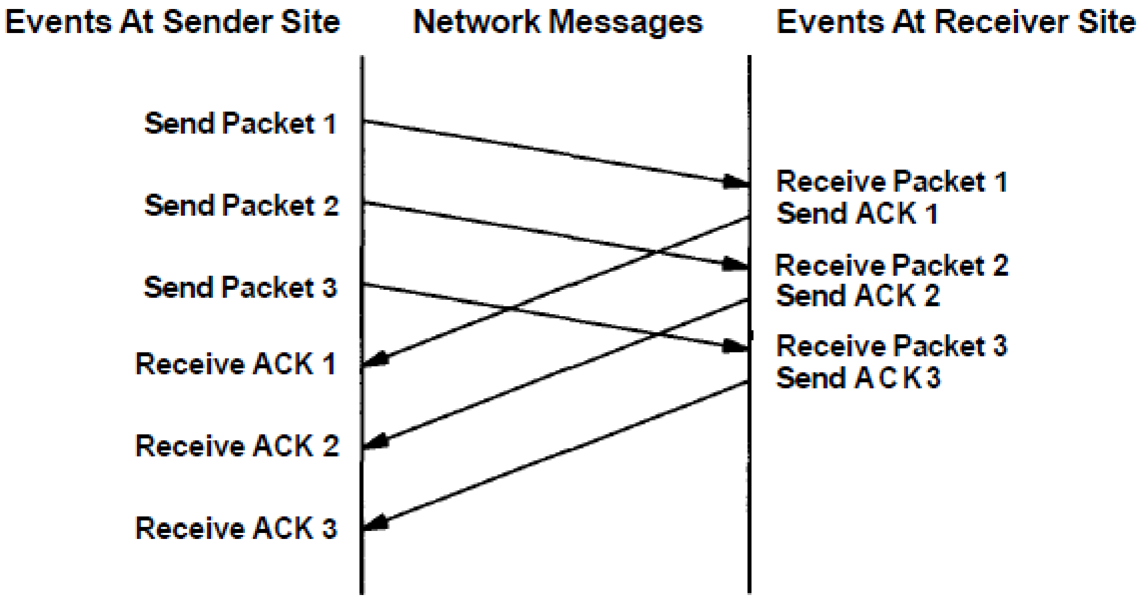
- The sliding window technique is a more complex form of positive acknowledgement and retransmission than the simple method discussed above.
- Sliding window protocols use network bandwidth better because they allow the sender to transmit multiple packets before waiting for an acknowledgement.

- The protocol places a small, fixed-size ***window*** on the sequence and transmits all packets that lie inside the window.



- We say that a packet is ***unacknowledged*** if it has been transmitted but no acknowledgement has been received.
- Technically, the number of packets that can be unacknowledged at any given time is constrained by the ***window size*** and is limited to a small, fixed number.
- Sliding window protocol with window size 8,the sender is permitted to transmit 8 packets before it receives an acknowledgement.

- The performance of sliding window protocols depends on the window size and the speed at which the network accepts packets.



Events At Sender Site

Network Messages

Events At Receiver Site

Send Packet 1

Receive Packet 1

Send ACK 1

Send Packet 2

Receive Packet 2

Send ACK 2

Send Packet 3

Receive Packet 3

Send ACK 3

Receive ACK 1

Receive ACK 2

Receive ACK 3

TCP

TCP

- The reliable stream service is so important that the entire protocol suite is referred to as TCP/IP.
- The protocol specifies the format of the data and acknowledgements that two computers exchange to achieve a reliable transfer, as well as the procedures the computers use to ensure that the data arrives correctly.
- It specifies how TCP software distinguishes among multiple destinations on a given machine, and how communicating machines recover from errors like lost or duplicated packets.
- The protocol also specifies how two computers initiate a TCP stream transfer and how they agree when it is complete.

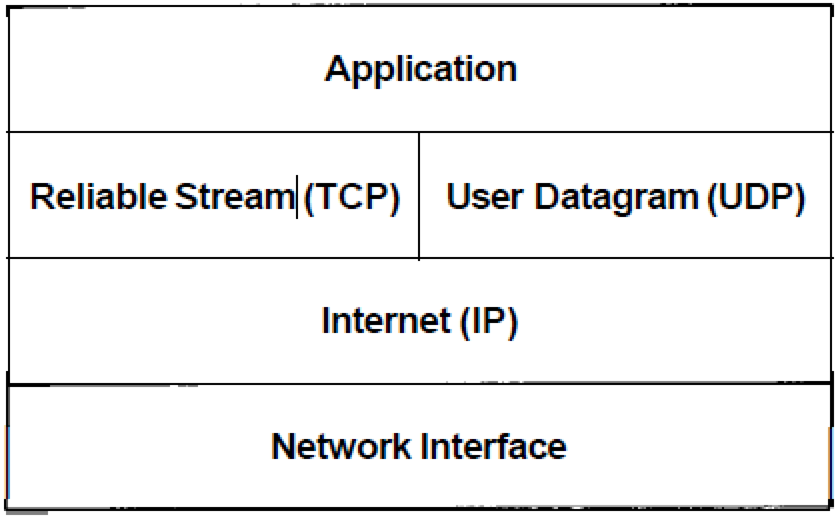
- Although the TCP specification describes how application programs use TCP in general terms, it does not dictate the details of the interface between an application program and TCP.
- it does not specify the exact procedures application programs invoke to access these operations.

- TCP assumes little about the underlying communication system, TCP can be used with a variety of packet delivery systems, including the IP datagram delivery service.
- For example, TCP can be implemented to use dialup telephone lines, a local area network, a high speed fiber optic network, or a lower speed long haul network.
- In fact, the large variety of delivery systems TCP can use is one of its strengths.

Ports, Connections, And Endpoints

- TCP uses ***protocol port*** numbers to identify the ultimate destination within a machine.
- Each port is assigned a small integer used to identify.

Conceptual Layering



Application

Reliable Stream (TCP) | User Datagram (UDP)

Internet (IP)

Network Interface

- TCP ports are much more complex because a given port number does not correspond to a single object.
- Instead, TCP has been built on the ***connection abstraction,*** in which the objects to be identified are virtual circuit connections, not individual ports.
- Understanding that TCP uses the notion of connections is crucial because it helps explain the meaning and use of TCP port **User Datagram (UDP)** numbers

- TCP uses the connection, not the protocol port, as its fundamental abstraction; connections are identified by a pair of endpoints.
- We have said that a connection consists of a virtual circuit between two application programs.
- Application program serves as the connection "endpoint."

- There is a connection from machine ***(18.26.0.36)*** at MIT to machine ***(128.10.2.3)*** at Purdue University, it might be defined by the endpoints:
- ***(18.26.0.36, 1069)*** and ***(128.10.2.3, 25).***

Passive And Active Opens

- TCP is a connection-oriented protocol that requires both endpoints to agree to participate.
- Before TCP traffic can pass across an internet, application programs at both ends of the connection must agree that the connection is desired.
- To do so, the application program on one end performs a ***passive open*** function by contacting its operating system and indicating that it will accept an incoming connection.

- The operating system assigns a TCP port number for its end of the connection.
- The application program at the other end must then contact its operating system using an ***active open*** request to establish a connection.
- The two TCP software modules communicate to establish and verify a connection.
- Once a connection has been created, application programs can begin to pass data; the TCP software modules at each end exchange messages that guarantee reliable delivery.

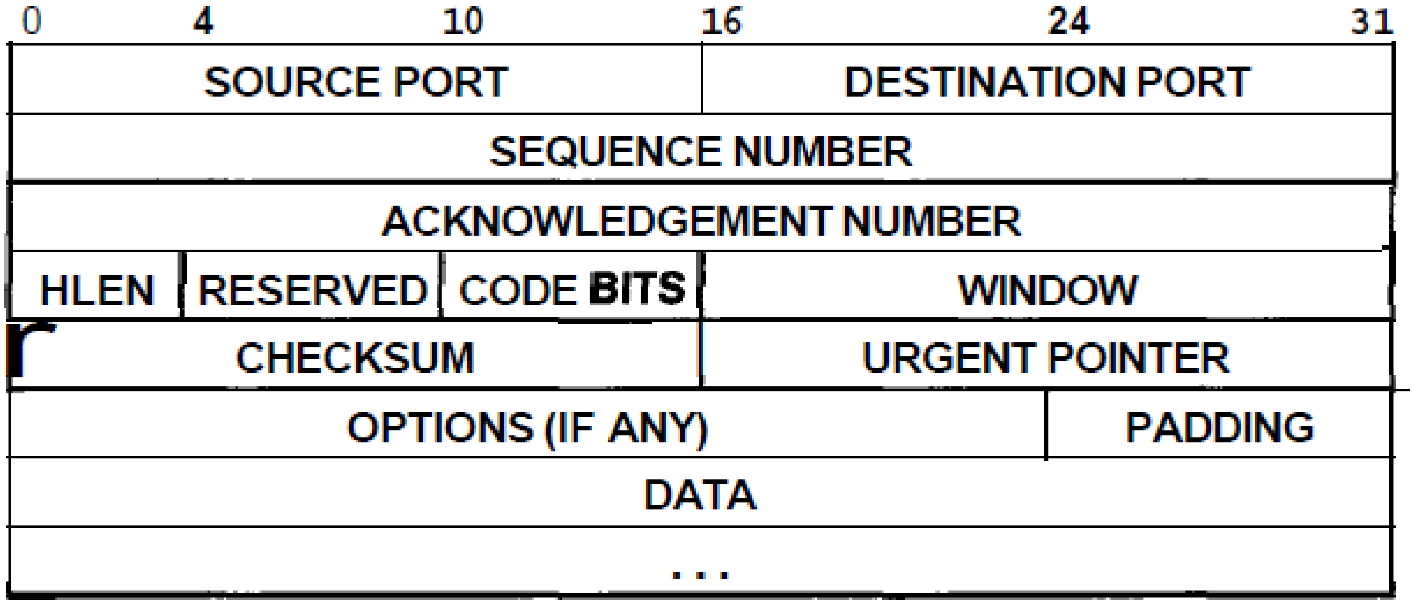
Segments, Streams, And Sequence Numbers

- TCP views the data stream as a sequence of octets or bytes that it divides into segments for transmission.
- Each segment travels across an internet in a single IP datagram.
- TCP uses a specialized sliding window mechanism to solve two important problems: efficient transmission and flow control.

TCP Segment Format

- Segments are exchanged to establish connections, transfer data, send acknowledgements, advertise window sizes, and close connections.
- Because TCP uses piggybacking, an acknowledgement traveling from machine ***A*** to machine ***B*** may travel in the same segment as data traveling from machine ***A*** to machine ***B,*** even though the acknowledgement refers to data sent from ***B*** to ***A***

TCP Segment Format



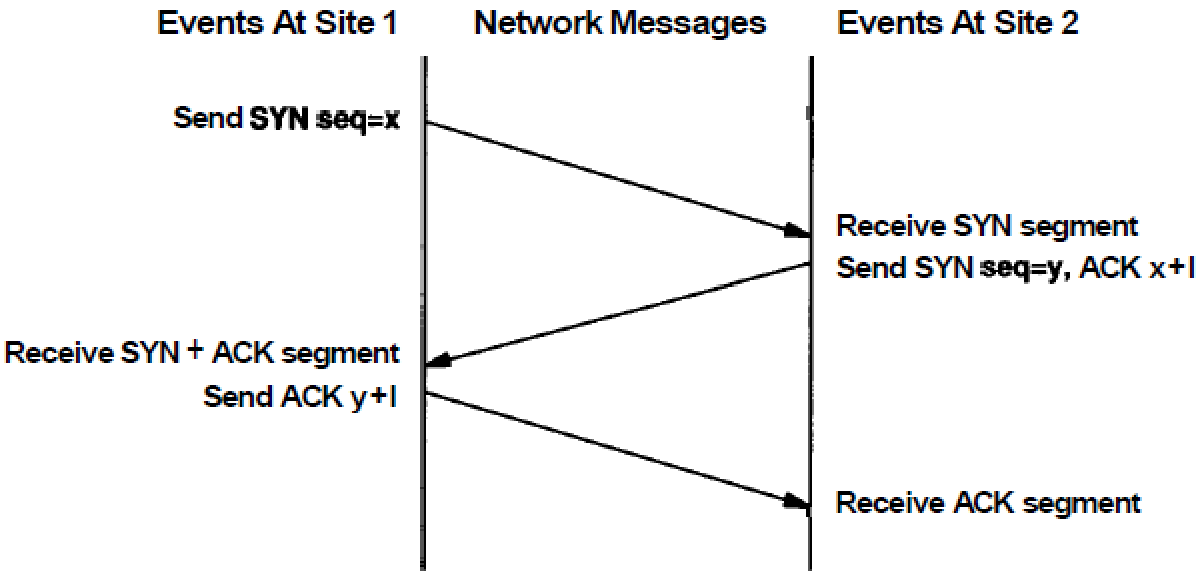
0	4	10	16	24	31
SOURCE PORT			DESTINATION PORT		
SEQUENCE NUMBER					
ACKNOWLEDGEMENT NUMBER					
HLEN	RESERVED	CODE BITS	WINDOW		
CHECKSUM			URGENT POINTER		
OPTIONS (IF ANY)				PADDING	
DATA					
. . .					

Timeout And Retransmission

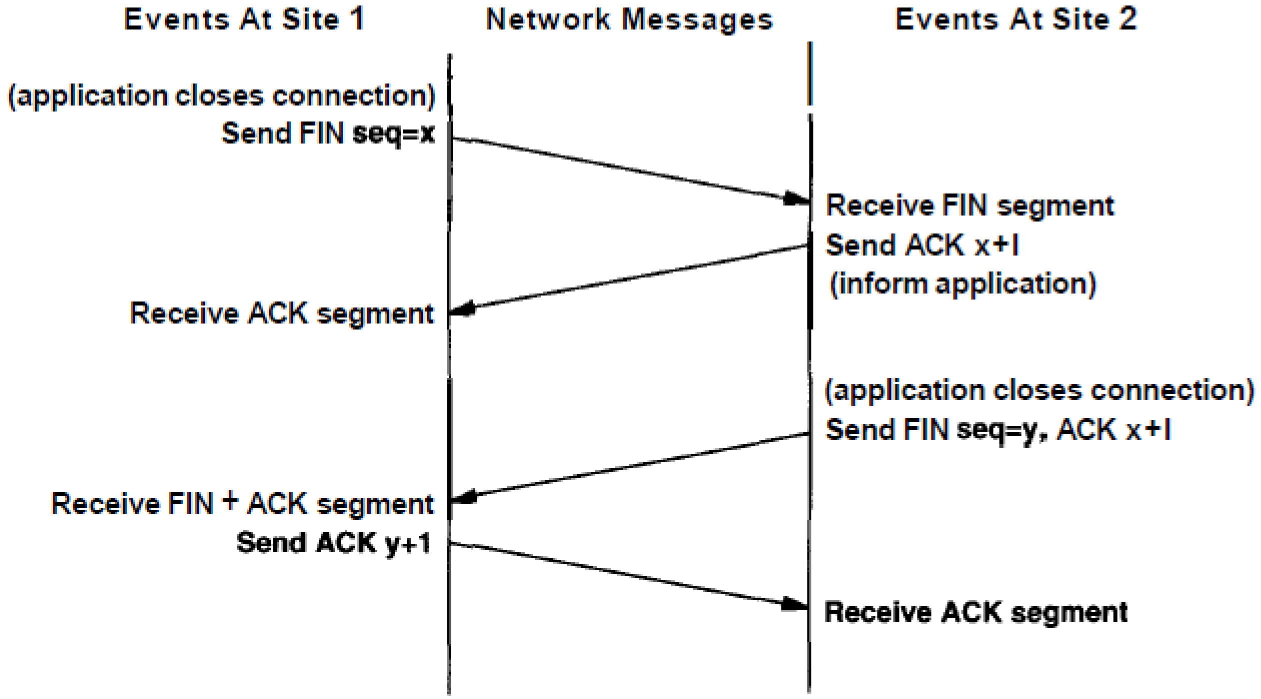
- TCP expects the destination to send acknowledgements whenever it successfully receives new octets from the data stream. Every time it sends a segment, TCP starts a timer and waits for an acknowledgement.
- If the timer expires before data in the segment has been acknowledged, TCP assumes that the segment was lost or corrupted and retransmits it.
- TCP computes an elapsed time known as a ***sample round trip time*** or ***round trip sample.***

Establishing A TCP Connection

- TCP uses a ***three-way handshake***



Closing a TCP Connection



Events At Site 1

Network Messages

Events At Site 2

(application closes connection)
Send FIN seq=x

Receive FIN segment
Send ACK x+I
(inform application)

Receive ACK segment

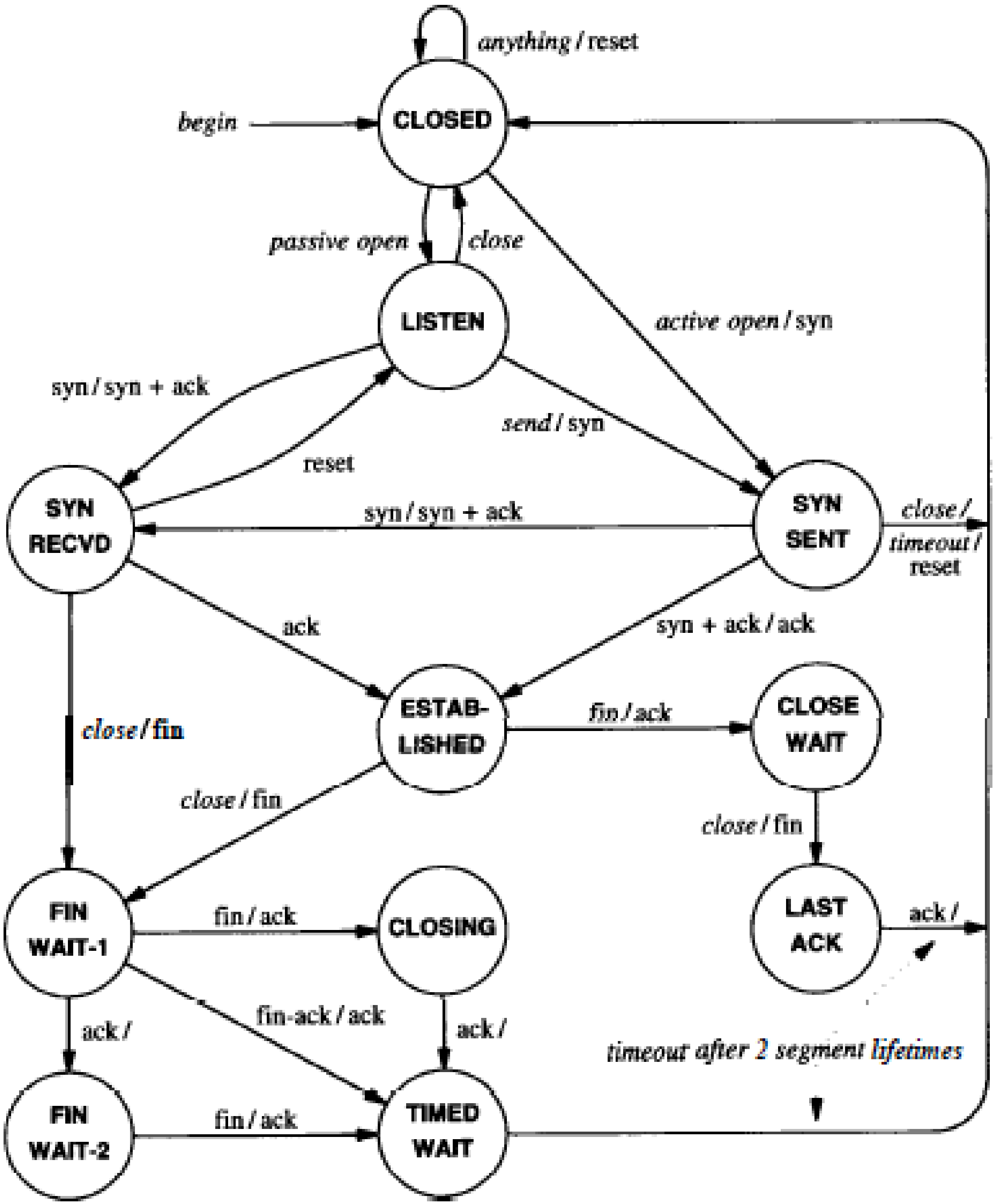
(application closes connection)
Send FIN seq=y, ACK x+I

Receive FIN + ACK segment
Send ACK y+1

Receive ACK segment

TCP State Machine

- Like most protocols, the operation of TCP can best be explained with a theoretical model called a finite state machine.
- Circles representing states and arrows representing transitions between them.
- The label on each transition shows what TCP receives to cause the transition and what it sends in response.
- For example, the TCP software at each endpoint begins in the ***CLOSED*** state.
- Application programs must issue either a passive open command (to wait for a connection from another machine), or an active open command (to initiate a connection).
- An active open command forces a transition from the ***CLOSED*** state to the ***SYN SENT*** state.
- When TCP follows the transition, it emits a **SYN** segment.
- When the other end returns a segment that contains a **SYN** plus ACK, TCP moves to the ***ESTABLISHED*** state and begins data transfer.


anything / reset
begin
CLOSED
passive open
close
LISTEN
active open / syn
syn / syn + ack
send / syn
reset
SYN RECVD
syn / syn + ack
SYN SENT
close /
timeout / reset
ack
syn + ack / ack
close / fin
ESTAB-LISHED
fin / ack
CLOSE WAIT
close / fin
close / fin
LAST ACK
ack /
FIN WAIT-1
fin / ack
CLOSING
fin-ack / ack
ack /
ack /
timeout after 2 segment lifetimes
FIN WAIT-2
fin / ack
TIMED WAIT

- The ***TIMED WAIT*** state reveals how TCP handles some of the problems incurred with unreliable delivery.
- TCP keeps a notion of maximum segment lifetime ***(MSL),*** the maximum time an old segment can remain alive in an internet.
- To avoid having segments from a previous connection interfere with a current one, TCP moves to the ***TIMED WAIT*** state after closing a connection.
- It remains in that state for twice the maximum segment lifetime before deleting its record of the connection.
- If any duplicate segments happen to arrive for the connection during the timeout interval, TCP will reject them. However, to handle cases where the last acknowledgement was lost, TCP acknowledges valid segments and restarts the timer.
- Because the timer allows TCP to distinguish old connections from new ones, it prevents TCP from responding with a ***RST*** (reset) if the other end retransmits a ***FIN*** request.

Reserved TCP Port Numbers

- TCP combines static and dynamic port binding, using a set of ***wellknown port assignments*** for commonly invoked programs (e.g., electronic mail), but leaving most port numbers available for the operating system to allocate as programs need them.
- Although the standard originally reserved port numbers less than 256 for use as well-known ports, numbers over 1024 have now been assigned.
- It should be pointed out that although TCP and UDP port numbers are independent, the designers have chosen to use the same integer port numbers for any service that is accessible from both UDP and TCP.
- For example, a domain name server can be accessed either with TCP or with UDP.
- In either protocol, port number 53 has been reserved for servers in the domain name system.

Decimal	Keyword	UNIX Keyword	Description
0			Reserved
1	TCPMUX		TCP Multiplexor
7	ECHO	echo	Echo
9	DISCARD	discard	Discard
11	USERS	systat	Active Users
13	DAYTIME	daytime	Daytime
15		netstat	Network status program
17	QUOTE	qotd	Quote of the Day
19	CHARGEN	chargen	Character Generator
20	FTP-DATA	ftp-data	File Transfer Protocol (data)
21	FTP	ftp	File Transfer Protocol
22	SSH	ssh	Secure Shell
23	TELNET	telnet	Terminal Connection
25	SMTP	smtp	Simple Mail Transport Protocol
37	TIME	time	Time
43	NICNAME	whois	Who Is
53	DOMAIN	nameserver	Domain Name Server
67	BOOTPS	bootps	BOOTP or DHCP Server
77		rje	any private RJE service
79	FINGER	finger	Finger
80	WWW	www	World Wide Web Server
88	KERBEROS	kerberos	Kerberos Security Service
95	SUPDUP	supdup	SUPDUP Protocol
101	HOSTNAME	hostnames	NIC Host Name Server
102	ISO-TSAP	iso-tsap	ISO-TSAP
103	X400	x400	X.400 Mail Service
104	X400-SND	x400-snd	X.400 Mail Sending
110	POP3	pop3	Post Office Protocol Vers. 3
111	SUNRPC	sunrpc	SUN Remote Procedure Call
113	AUTH	auth	Authentication Service
117	UUCP-PATH	uucp-path	UUCP Path Service
119	NNTP	nntp	USENET News Transfer Protocol
123	NTP	ntp	Network Time Protocol
139	NETBIOS-SSN		NETBIOS Session Service
161	SNMP	snmp	Simple Network Management Protc

Silly Window Syndrome And Small Packets

- The sending and receiving applications operate at different speeds.
- When a connection is first established, the receiving TCP allocates a buffer of K bytes, and uses the ***WINDOW*** field in acknowledgement segments to advertise the available buffer size to the sender.
- If the sending application generates data quickly, the sending TCP will transmit segments with data for the entire window.
- Eventually, the sender will receive an acknowledgement that specifies the entire window has been filled, and no additional space remains in the receiver's buffer.

- When it learns that space is available, the sending TCP responds by transmitting a segment that contains one octet of data.
- The sending TCP must compose a segment that contains one octet of data, place the segment in an IP datagram, and transmit the result.
- When the receiving application reads another octet, TCP generates another acknowledgement, which causes the sender to transmit another segment that contains one octet of data.
- The resulting interaction can reach a steady state in which TCP sends a separate segment for each octet of data.

- Transferring small segments consumes unnecessary network bandwidth and introduces unnecessary computational overhead.
- The transmission of small segments consumes unnecessary network bandwidth because each datagram carries only one octet of data; the ratio of header to data is large.
- Computational overhead arises because TCP on both the sending and receiving computers must process each segment.
- Early TCP implementations exhibited a problem known as silly window syndrome in which each acknowledgement advertises a small amount of space available and each segment carries a small amount of data**.**

Avoiding Silly Window Syndrome

- A heuristic used on the sending machine avoids transmitting a small amount of data in each segment.
- Another heuristic used on the receiving machine avoids sending small increments in window advertisements that can trigger small data packets.
- Although the heuristics work well together, having both the sender and receiver avoid silly window helps ensure good performance in the case that one end of a connection fails to correctly implement silly window avoidance.

Receive-Side Silly Window Avoidance

- A receiver maintains an internal record of the currently available window, but delays advertising an increase in window size to the sender until the window can advance a significant amount.
- Receive-side silly window prevents small window advertisements in the case where a receiving application extracts data octets slowly.
- For example, when a receiver's buffer fills completely, it sends an acknowledgement that contains a zero window advertisement.
- As the receiving application extracts octets from the buffer, the receiving TCP computes the newly available space in the buffer.
- Instead of sending a window advertisement immediately, the receiver waits until the available space reaches one half of the total buffer size or a maximum sized segment.

- Receive-Side Silly Window Avoidance: Before sending an updated window advertisement after advertising a zero window, wait for space to become available that is either at least 50% of the total buffer size or equal to a maximum sized segment.

Send-Side Silly Window Avoidance

- To achieve the goal, a sending TCP must allow the sending application to make multiple calls to ***write,*** and must collect the data transferred in each call before transmitting it in a single, large segment.
- That is, a sending TCP must delay sending a segment until it can accumulate a reasonable amount of data. The technique is known as ***clumping.***

- Send-Side Silly Window Avoidance: When a sending application generates additional data to be sent over a connection for which previous data has been transmitted but not acknowledged, place the new data in the output buffer as usual, but do not send additional segments until there is sufficient data to fill a maximum-sized segment.
- If still waiting to send when an acknowledgement arrives, send all data that has accumulated in the buffer. Apply the rule even when the user requests a push operation.

VPN (Private Network Interconnections)

Private Network

- An organization builds its own TCP/IP internet separate from the global Internet is referred as Private Network.
- A private network uses routers to interconnect networks at each site, and leased digital circuits to interconnect the sites.
- All data remains private because no outsiders have access to any part of a private network.

Hybrid Network

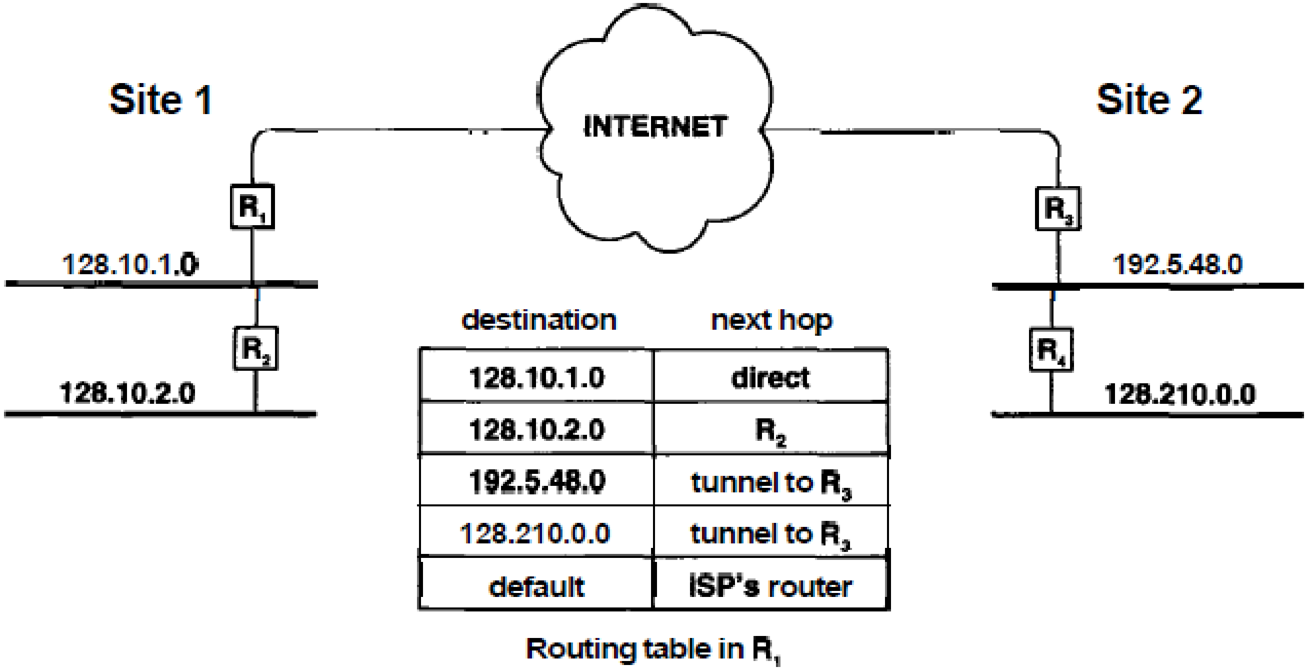
- The complete isolation is not always desirable. Thus, many organizations choose a ***hybrid network*** architecture that combines the advantages of private networking with the advantages of global Internet connectivity.
- That is, the organization uses globally valid IP addresses and connects each site to the Internet.
- The advantage is that hosts in the organization can access the global Internet when needed, but can be assured of privacy when communicating internally.

Virtual Private Network

- The organization that uses the global Internet to connect its sites can keep its data private using VPN.
- A VPN is ***private*** in the same way as a private network -the technology guarantees that communication between any pair of computers in the VPN remains concealed from outsiders.
- A VPN is ***virtual*** because it does not use leased circuits to interconnect sites.
- Instead, a VPN uses the global Internet to pass traffic from one site to another.

- Two basic techniques make a VPN possible: ***tunneling*** and ***encryption.***
- To guarantee privacy, a VPN encrypts each outgoing datagram before encapsulating it in another datagram for transmission.

VPN Addressing and Routing



destination	next hop
128.10.1.0	direct
128.10.2.0	R_2
192.5.48.0	tunnel to R_3
128.210.0.0	tunnel to R_3
default	ISP's router

Routing table in R_1

Figure 20.3 A VPN that spans two sites and R_1's routing table. The tunnel from R, to R_3 is configured like a point-to-point leased circuit.

VPN with private addresses

- A VPN offers an organization the same addressing options as a private network.
- If hosts in the VPN do not need general Internet connectivity, the VPN can be configured to use arbitrary IP addresses; if hosts need Internet access, a hybrid addressing scheme can be used.
- A minor difference is that when private addressing is used, one globally valid IP address is needed at each site for tunneling.
- Each application gateway handles only one specific service; multiple gateways are required for multiple services.

Network Address Translation (NAT)

- A technology has been created that solves the general problem of providing **IP** level access between hosts at a site and the rest of the Internet, without requiring each host at the site to have a globally valid **IP** address.
- ***Network Address Translation(NAT)*** requires a site to have a single connection to the global Internet and at least one globally valid IP address.

- NAT translates the addresses in both outgoing and incoming datagrams by replacing the source address in each outgoing datagram and replacing the destination address in each incoming datagram with the private address of the correct host.
- All datagram come from the NAT box and all responses return to the NAT box.
- From the view of internal hosts, the NAT box appears to be a router that can reach the global Internet.

- The chief advantage of NAT arises from its combination of generality and transparency.
- NAT is more general than application gateways because it allows an arbitrary internal host to access an arbitrary service on a computer in the global Internet.
- NAT is transparent because it allows an internal host to send and receive datagrams using a private address.

NAT Translation Table Creation

- NAT maintains a translation table that it uses to perform the mapping.
- Each entry in the table specifies two items: the IP address of a host on the Internet and the internal IP address of a host at the site.
- The NAT translation table must be in place before a datagram arrives from the Internet.
  - ***Manual initialization.***
  - ***Outgoing datagrams.***
  - ***Incoming name lookups.***

Interaction Between NAT And ICMP

- Even straightforward changes to an IP address can cause unexpected side-effects in higher layer protocols.
- For example, suppose an internal host uses ***ping*** to test reachability of a destination on the Internet.
- The host expects to receive an ICMP ***echo reply*** for each ICMP ***echo request*** message it sends.
- Thus, NAT must forward incoming echo replies to the correct host.
- However, NAT does not forward all ICMP messages that arrive from the Internet.
- If routes in the NAT box are incorrect, for example, an ICMP ***redirect*** message must be processed locally.
- Thus, when an ICMP message arrives from the Internet, NAT must first determine whether the message should be handled locally or sent to an internal host.
- Before forwarding to an internal host, NAT translates the ICMP message.

Interaction Between NAT And Applications

- Although ICMP makes NAT complex, application protocols have a more serious effect.
- In general, NAT will not work with any application that sends IP addresses or protocol ports as data.
- NAT affects ICMP and higher layer protocols; except for a few standard applications like FTP, an application protocol that passes IP addresses or protocol port numbers as data will not operate correctly across NAT.

Conceptual Address Domains

- We have described NAT as a technology that can be used to connect a private network to the global Internet.
- In fact, NAT can be used to interconnect any two address domains**.**
- The individual can assign private addresses to the computers at home, and use NAT between the home network and the corporate intranet.
- The corporation can also assign private addresses and use NAT between its intranet and the global Internet.

DHCP

Introduction

- To overcome some of the drawbacks of RARP, researchers developed the BOOTstrap Protocol (BOOTP).
- Later, the Dynamic Host Configuration Protocol (DHCP) was developed as a successor to BOOTP.
- Because the two protocols are closely related, most of the description applies to both.
- DHCP extends the functionality to provide dynamic address assignment.

Using IP To Determine An IP Address

- BOOTP uses UDP to carry messages and UDP messages are encapsulated in **IP** datagrams for delivery.
- An application program can use the limited broadcast IP address to force IP to broadcast a datagram on the local network before IP has discovered the IP address of the local network or the machine's IP address.

Need For Dynamic Configuration

- BOOTP was designed for a relatively static environment in which each host has a permanent network connection.
- A manager creates a BOOTP configuration file that specifies a set of BOOTP parameters for each host.
- The file does not change frequently because the configuration usually remains stable.
- With the advent of wireless networking and portable computers such as laptops and notebooks, it has become possible to move a computer from one location to another quickly and easily and BOOTP does not support it.

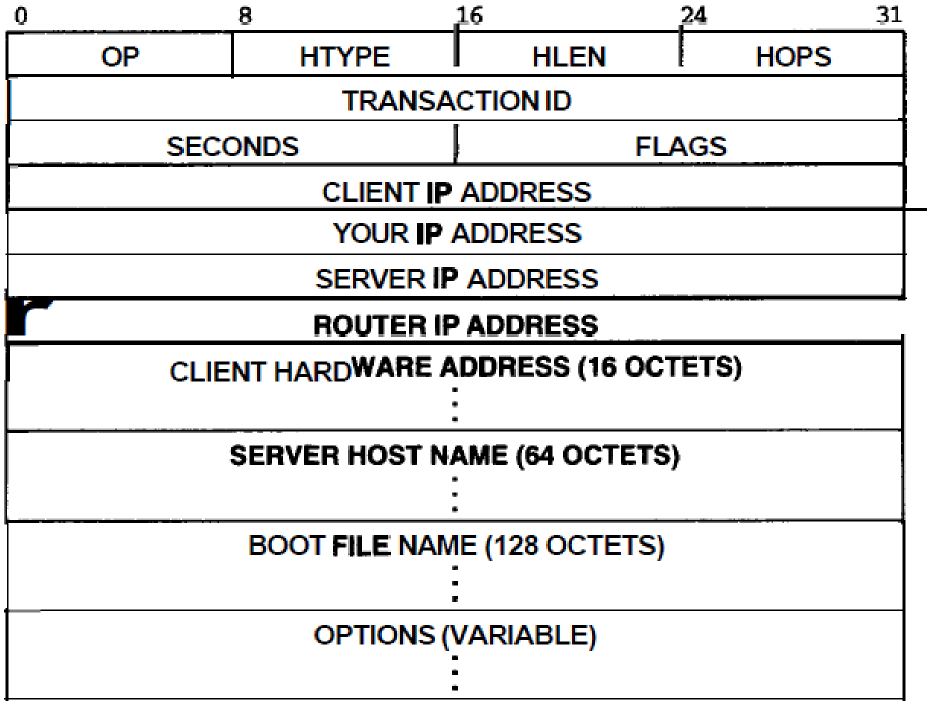
Dynamic Host Configuration

- IETF has designed *Dynamic Host Configuration Protocol (DHCP) which* extends BOOTP in two ways
  - DHCP allows a computer to acquire all the configuration information it needs in a single message
  - DHCP allows a computer to obtain IP address quickly and dynamically
- DHCP allows three types of address assignment
  - manual configuration
  - automatic configuration
  - dynamic configuration

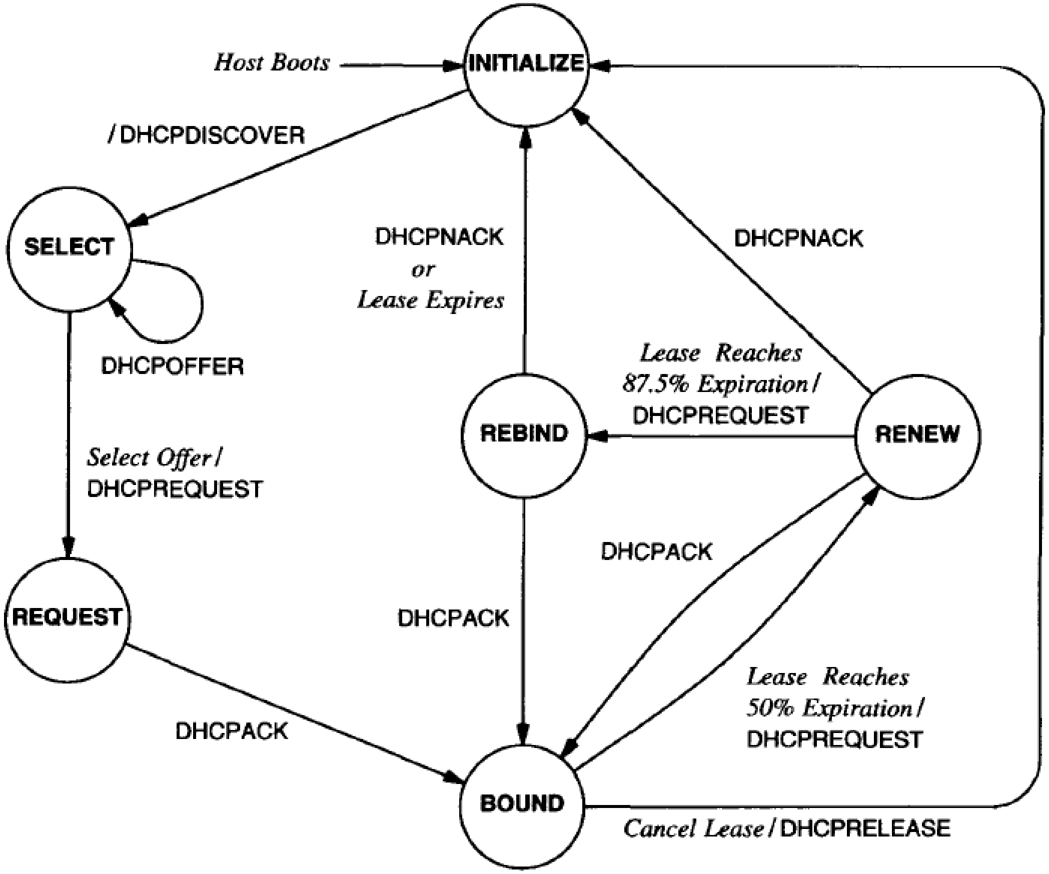
Dynamic IP Address Assignment

- Dynamic address assignment is not a one-to-one mapping, and the server does not need to know the identity of a client**.**
- In particular, a DHCP server can be configured to permit an arbitrary computer to obtain an **IP** address and begin communicating.
- Thus, DHCP makes it possible to design systems that auto configure.
- To make auto configuration possible, a DHCP server begins with a set of IP addresses that the network administrator gives the server to manage.
- The administrator specifies the rules by which the server operates.
- A DHCP client negotiates use of an address by exchanging messages with a server.
- In the exchange, the server provides an address for the client, and the client verifies that it accepts the address.
- Once a client has accepted an address, it can begin to use that address for communication.

DHCP Message Format



0	8	16	24 31
OP	HTYPE	HLEN	HOPS
TRANSACTION ID			
SECONDS		FLAGS	
CLIENT IP ADDRESS			
YOUR IP ADDRESS			
SERVER IP ADDRESS			
ROUTER IP ADDRESS			
CLIENT HARDWARE ADDRESS (16 OCTETS) ⋮			
SERVER HOST NAME (64 OCTETS) ⋮			
BOOT FILE NAME (128 OCTETS) ⋮			
OPTIONS (VARIABLE) ⋮			


INITIALIZE
Host Boots
/ DHCPDISCOVER
SELECT
DHCPOFFER
Select Offer /
DHCPREQUEST
REQUEST
DHCPACK
BOUND
Cancel Lease / DHCPRELEASE
DHCPNACK
or
Lease Expires
REBIND
DHCPACK
DHCPNACK
Lease Reaches
87.5% Expiration /
DHCPREQUEST
RENEW
DHCPACK
Lease Reaches
50% Expiration /
DHCPREQUEST

DNS

Domain Name System

- The earliest computer systems forced users to understand numeric addresses for objects like system tables and peripheral devices.
- Timesharing systems advanced computing by allowing users to invent meaningful symbolic names for both physical objects and abstract objects

Flat Namespace

- In this each name consisted of a sequence of characters without any further structure.
- The main advantage of a flat namespace is that names are convenient and short.
- The main disadvantage is that a flat namespace cannot generalize to large sets of machines for both technical and administrative reasons.

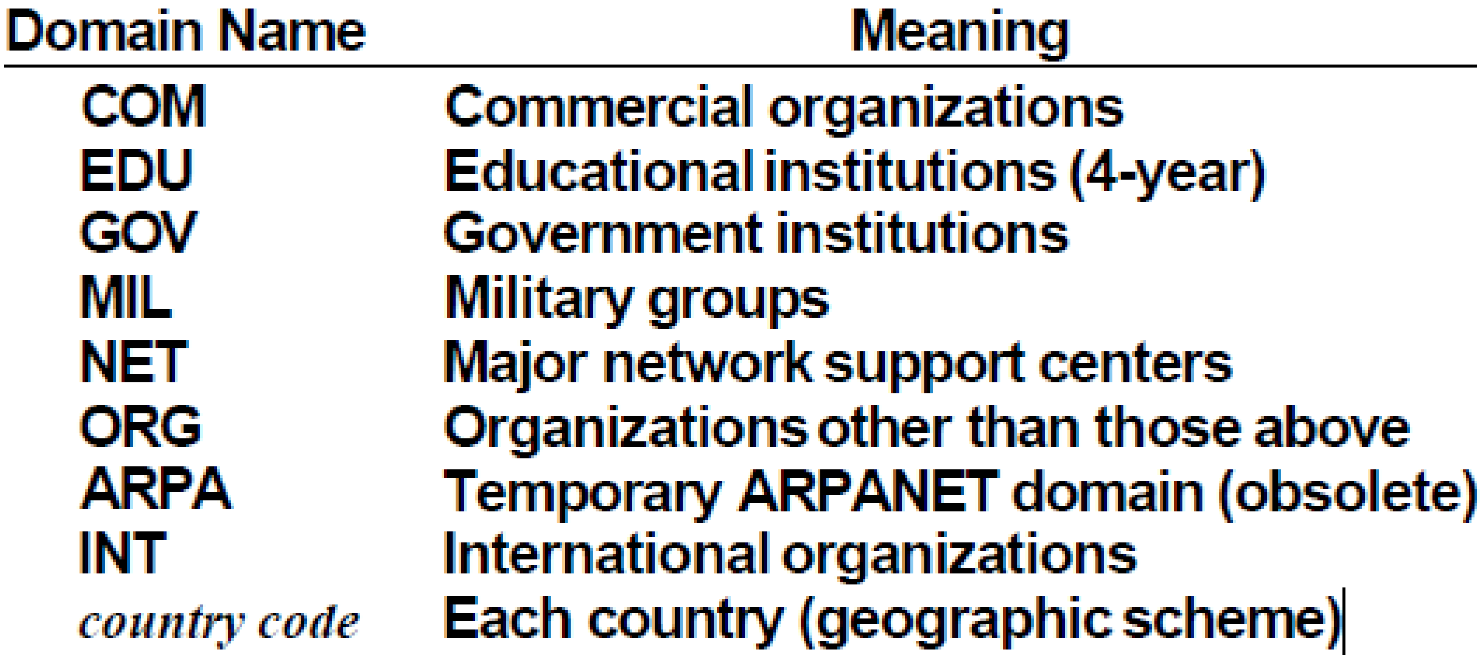
Hierarchical Namespace

- TCP/IP using this scheme for address mapping.
- The partitioning of a namespace must be defined in a way that supports efficient name mapping and guarantees autonomous control of name assignment.
- Besides making it easy to delegate authority, the hierarchy of a large organization introduces autonomous operation.
- Authority always passes down the corporate hierarchy, information can flow across the hierarchy from one office to another.

Internet Domain Names

- The mechanism that implements a machine name hierarchy for TCP/IP internets is called the ***Domain Name System***
- Any suffix of a label in a domain name is also called a ***domain.***
- The first section considers the name syntax, and later sections examine the implementation.

Official And Unofficial Internet Domain Names



Domain Name	Meaning
COM	Commercial organizations
EDU	Educational institutions (4-year)
GOV	Government institutions
MIL	Military groups
NET	Major network support centers
ORG	Organizations other than those above
ARPA	Temporary ARPANET domain (obsolete)
INT	International organizations
country code	Each country (geographic scheme)