UNIT-V
Transport Layer
Application layer

Contents:

Transport layer:

Elements of Transport Protocols

Internet Transport Protocols: UDP, TCP.

Application Layer:

DNS

HTTP

SNMP

E-Mail

WWW

- The transport layer is the heart of the protocol hierarchy.
- The transport layer builds on the network layer to provide data transport from a process on a source machine to a process on a destination machine.

Services provided by the transport layer:

- The software and/or hardware within the transport layer that does the work is called the **transport entity.**
- Just as there are two types of network service, connection-oriented and connectionless, there are also two types of transport service.
- The connection-oriented transport service is similar to the connection-oriented network service in many ways. In both cases, connections have three phases: establishment, data transfer, and release.
- Addressing and flow control are also similar in both layers.

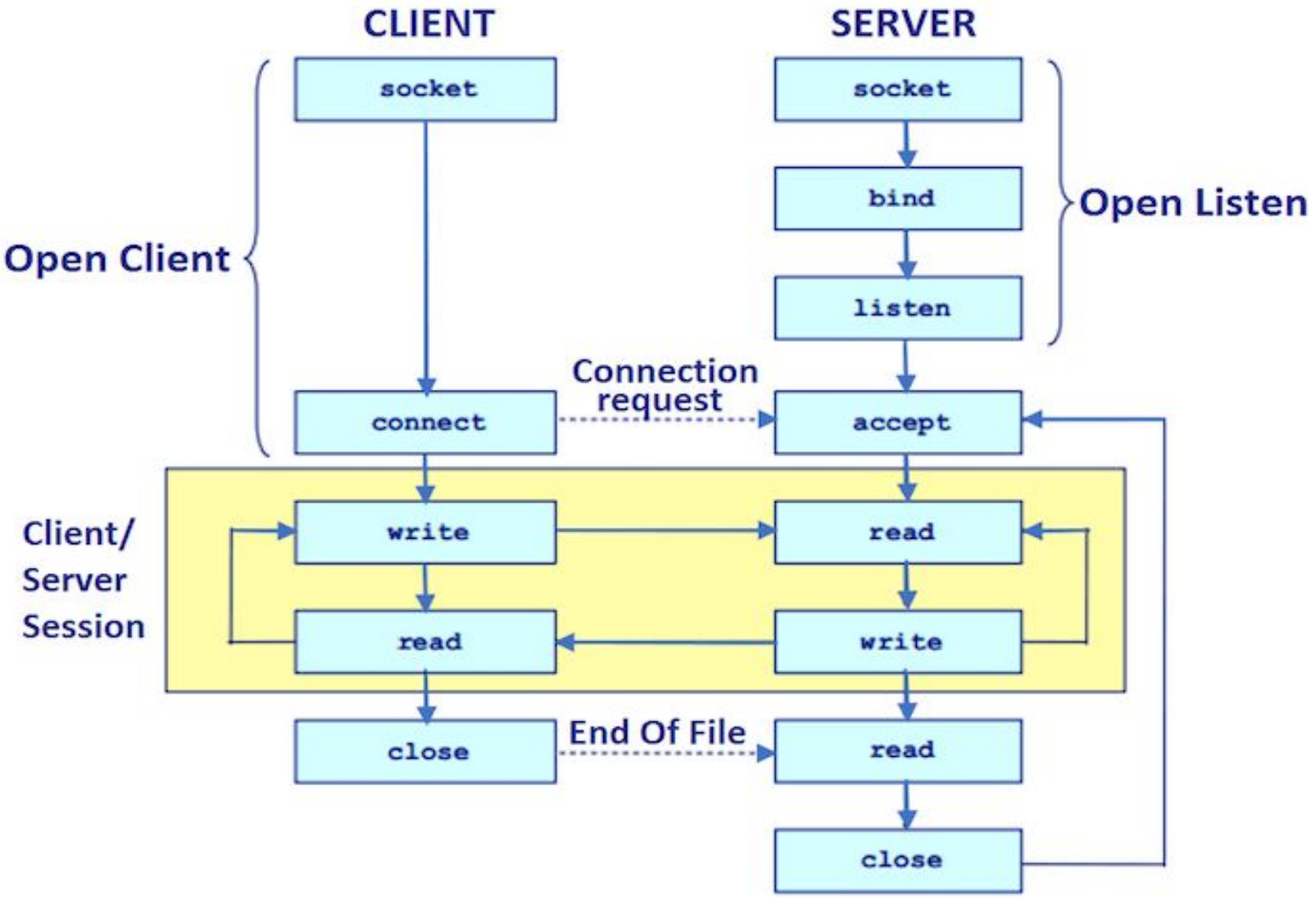
- The transport code runs entirely on the users' machines, but the network layer mostly runs on the routers. What if router fails?
- The only possibility is to put on top of the network layer another layer that improves the quality of the service.
- If, in a connectionless network, packets are lost or mangled, the transport entity can detect the problem and compensate for it by using retransmissions.
- In Earlier days, The data unit in transport layer is known as Transport Protocol Data Unit(TPDU), now it is known as **Segment**.

Berkeley Sockets:

- Sockets were first released as part of the Berkeley UNIX 4.2BSD software distribution in 1983.
- A **socket** is one endpoint of a two-way communication link between two programs running on the **network**.
- A **socket** is bound to a port number so that the **TCP** layer can identify the application that data is destined to be sent to. An endpoint is a combination of an IP address and a port number

Primitive	Meaning
SOCKET	Create a new communication endpoint
BIND	Associate a local address with a socket
LISTEN	Announce willingness to accept connections; give queue size
ACCEPT	Passively establish an incoming connection
CONNECT	Actively attempt to establish a connection
SEND	Send some data over the connection
RECEIVE	Receive some data from the connection
CLOSE	Release the connection

Figure 6-5. The socket primitives for TCP.


CLIENT
SERVER
socket
socket
bind
listen
Open Client
Open Listen
Connection request
connect
accept
Client/ Server Session
write
read
read
write
close
End Of File
read
close

Elements of Transport Protocols:

Addressing:

- When an application (e.g., a user) process wishes to set up a connection to a remote application process, it must specify which one to connect to.
- The method normally used is to define transport addresses to which processes can listen for connection requests. In the Internet, these endpoints are called ports.
- TSAP (Transport Service Access Point)- specific endpoint in transport layer
- NSAPs(Network Service Access Points)- specific endpoint in Network layer. IP addresses are examples of NSAPs.

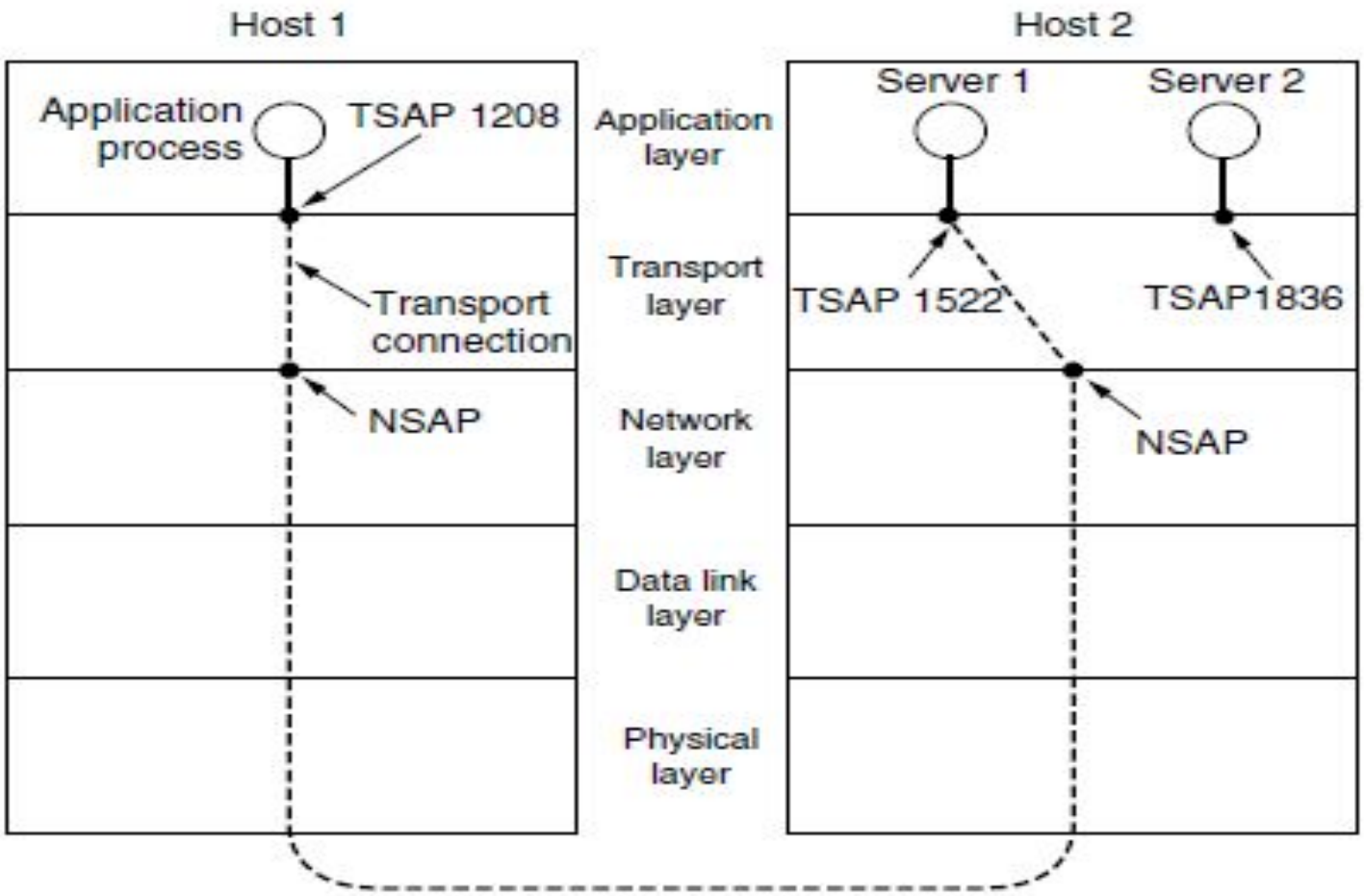


Figure 6-8. TSAPs, NSAPs, and transport connections.

A possible scenario for a transport connection is as follows:

- A mail server process attaches itself to TSAP 1522 on host 2 to wait for an incoming call. How a process attaches itself to a TSAP is out- side the networking model and depends entirely on the local operating system. A call such as our LISTEN might be used, for example.
- An application process on host 1 wants to send an email message, so it attaches itself to TSAP 1208 and issues a CONNECT request. The request specifies TSAP 1208 on host 1 as the source and TSAP 1522 on host 2 as the destination. This action ultimately results in a transport connection being established between the application process and the server.
- The application process sends over the mail message.
- The mail server responds to say that it will deliver the message.
- The transport connection is released.

Connection Establishment:

- Establishing a connection sounds easy, but it is actually surprisingly tricky.
- At first glance, it would seem sufficient for one transport entity to just send a CONNECTION REQUEST segment to the destination and wait for a CONNECTION ACCEPTED reply.
- The problem occurs when the network can lose, delay, corrupt, and duplicate packets.
- To solve this problem, Tomlinson (1975) introduced the **three-way handshake.**
- The normal setup procedure when host 1 initiates is shown in Fig. 6-11(a).
- Host 1 chooses a sequence number, *x, and sends a CONNECTION REQUEST segment containing it to host 2.*

- *Host 2* replies with an ACK segment acknowledging *x and announcing its own initial sequence* number, *y.*
- *Finally, host 1 acknowledges host 2's choice of an initial sequence* number in the first data segment that it sends.
- In Fig. 6-11(b), the first segment is a delayed duplicate
- CONNECTION REQUEST from an old connection. This segment arrives at host 2 without host 1's knowledge.
- Host 2 reacts to this segment by sending host 1 an ACK segment.
- When host 1 rejects host 2's attempt to establish a connection, host 2 realizes that it was tricked by a delayed duplicate and abandons the connection. In this way, a delayed duplicate does no damage.
- The worst case is when both a delayed CONNECTION REQUEST and an ACK are floating around in the subnet(Fig.6-11(c))

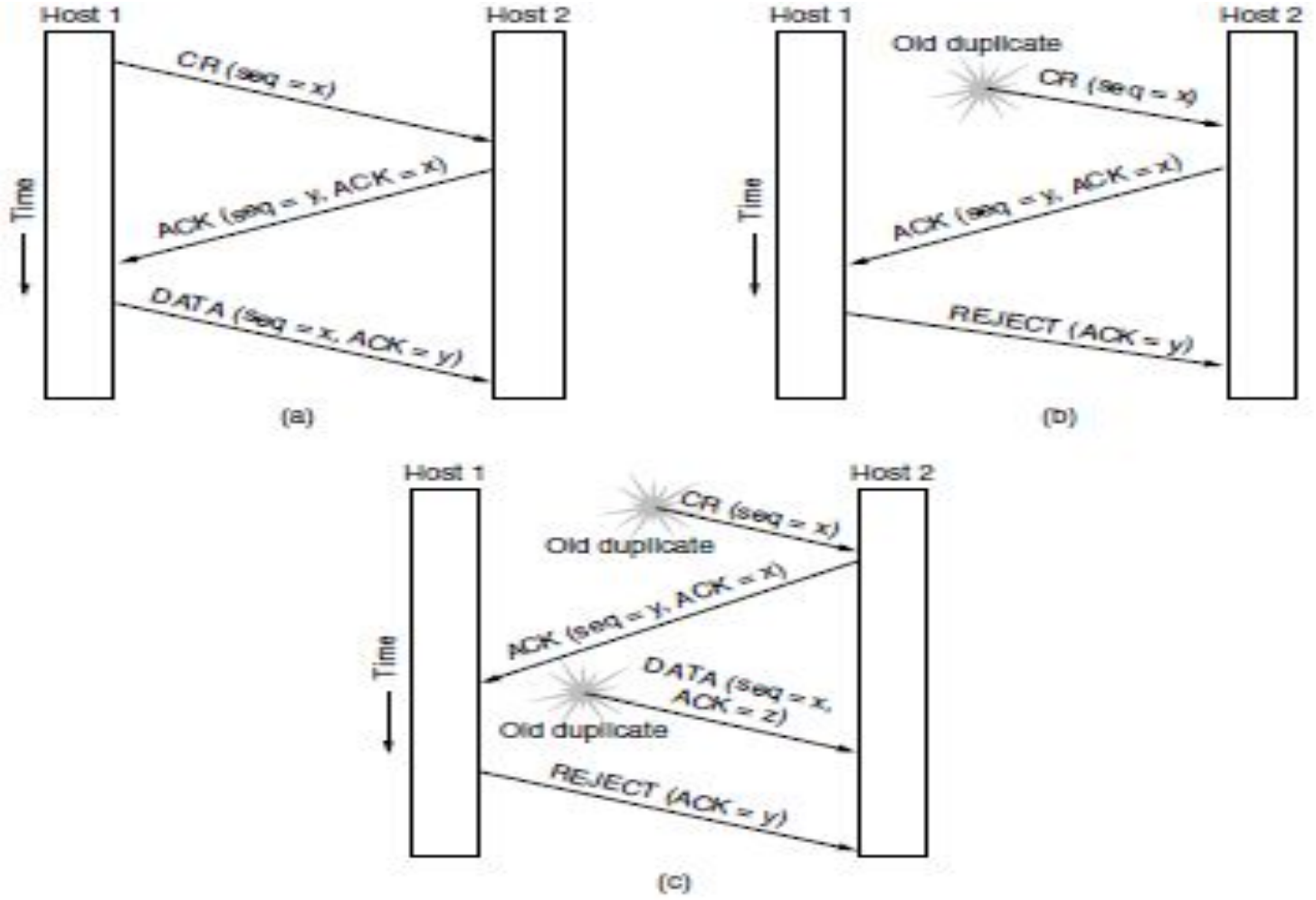


Figure 6-11. Three protocol scenarios for establishing a connection using a three-way handshake. CR denotes CONNECTION REQUEST. (a) Normal operation. (b) Old duplicate CONNECTION REQUEST appearing out of nowhere. (c) Duplicate CONNECTION REQUEST and duplicate ACK.

Connection Release:

- Releasing a connection is easier than establishing one.
- Asymmetric release is the way the telephone system works: when one party hangs up, the connection is broken.
- Symmetric release treats the connection as two separate unidirectional connections and requires each one to be released separately

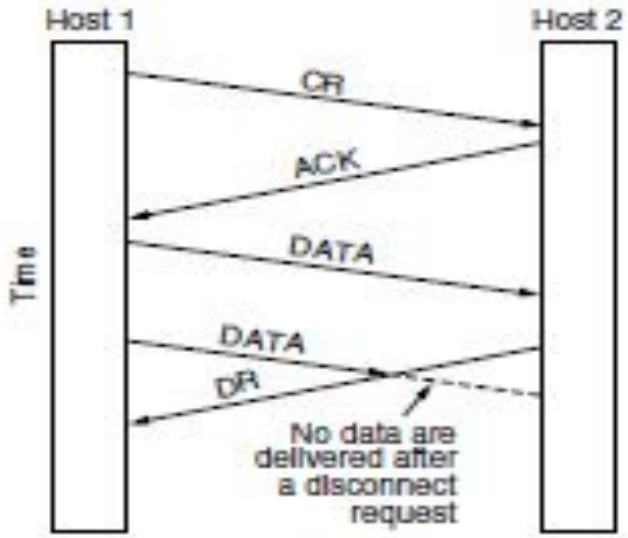


Figure 6-12. Abrupt disconnection with loss of data.

- In Fig. 6-14(a), we see the normal case in which one of the users sends a DR (DISCONNECTION REQUEST) segment to initiate the connection release. When it arrives, the recipient sends back a DR segment and starts a timer, just in case its DR is lost.
- When this DR arrives, the original sender sends back an ACK segment and releases the connection. Finally, when the ACK segment arrives, the receiver also releases the connection.
- Releasing a connection means that the transport entity removes the information about the connection from its table of currently open connections

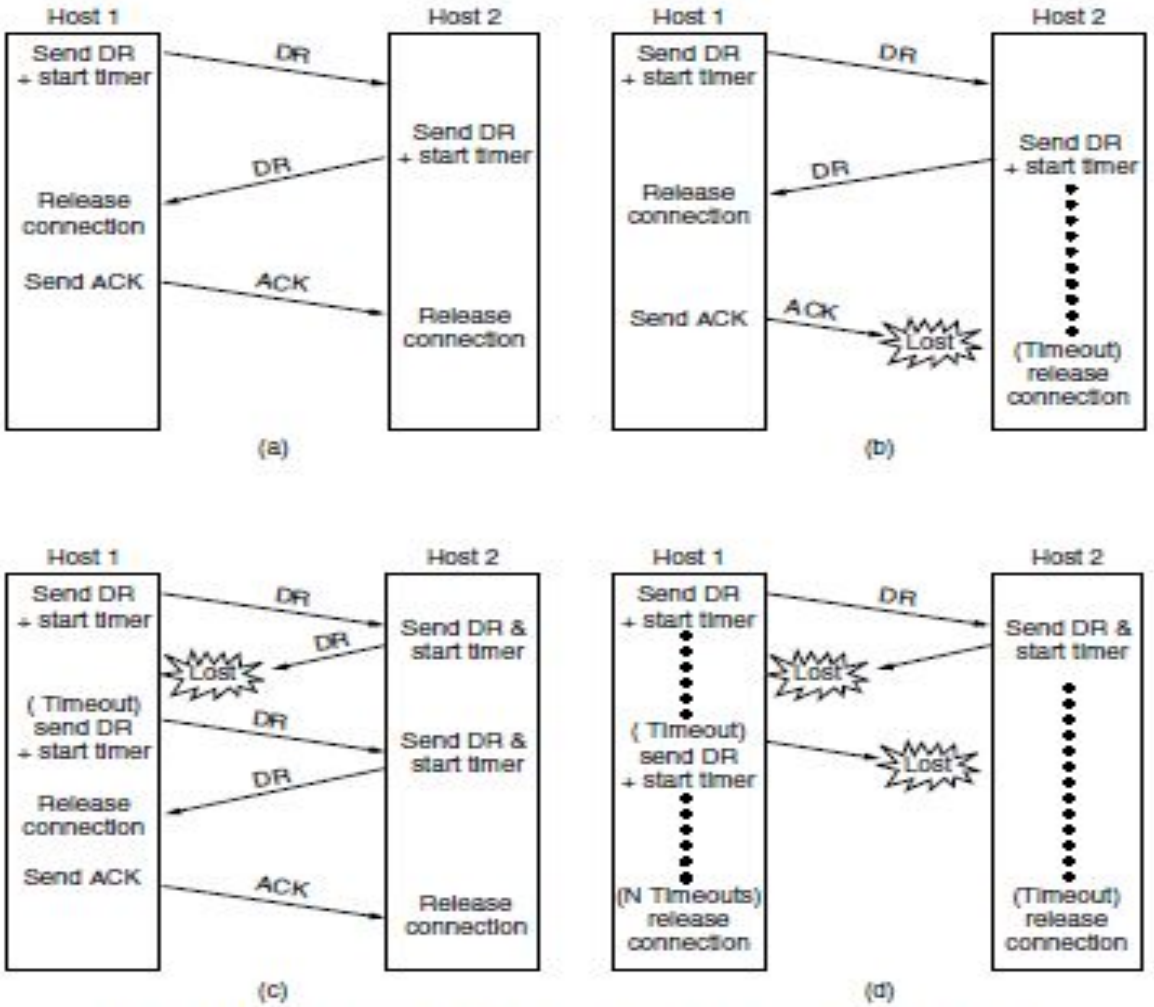
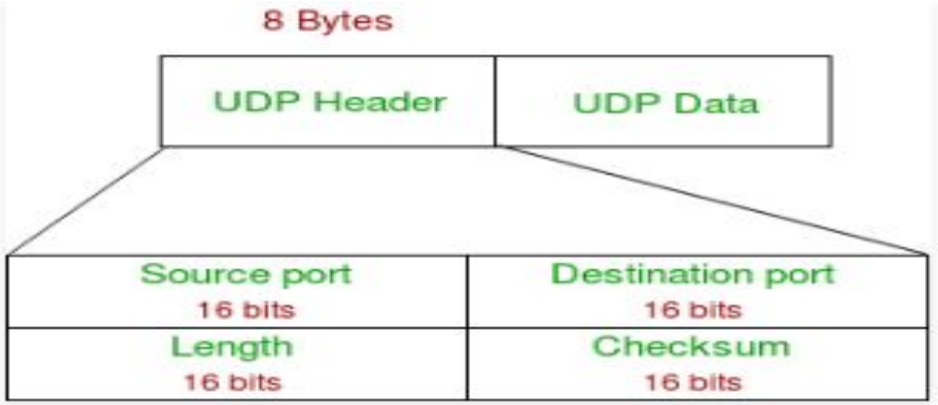


Figure 6-14. Four protocol scenarios for releasing a connection. (a) Normal case of three-way handshake. (b) Final ACK lost. (c) Response lost. (d) Response lost and subsequent DRs lost.

THE INTERNET TRANSPORT PROTOCOLS: UDP

- The Internet has two main protocols in the transport layer, a connectionless protocol and a connection-oriented one.
- The protocols complement each other. The connectionless protocol is User Datagram Protocol(UDP).
- The connection-oriented protocol is Transmission Control Protocol(TCP).
- UDP is described in RFC 768.
- UDP transmits segments consisting of an 8-byte header followed by the payload.
- The header is shown in Fig. The two ports serve to identify the endpoints within the source and destination machines. When a UDP packet arrives, its payload is handed to the process attached to the destination port

- The source port is primarily needed when a reply must be sent back to the source.
- By copying the *Source port field from the incoming segment into the Destination port field of the outgoing segment, the process sending the reply can* specify which process on the sending machine is to get it.



- **Source Port :** Source Port is 2 Byte long field used to identify port number of source.
- **Destination Port :** It is 2 Byte long field, used to identify the port of destined packet.
- **Length :** Length is the length of UDP including header and the data. It is 16-bits field.
- **Checksum :** Checksum is 2 Bytes long field.

Remote Procedure call:

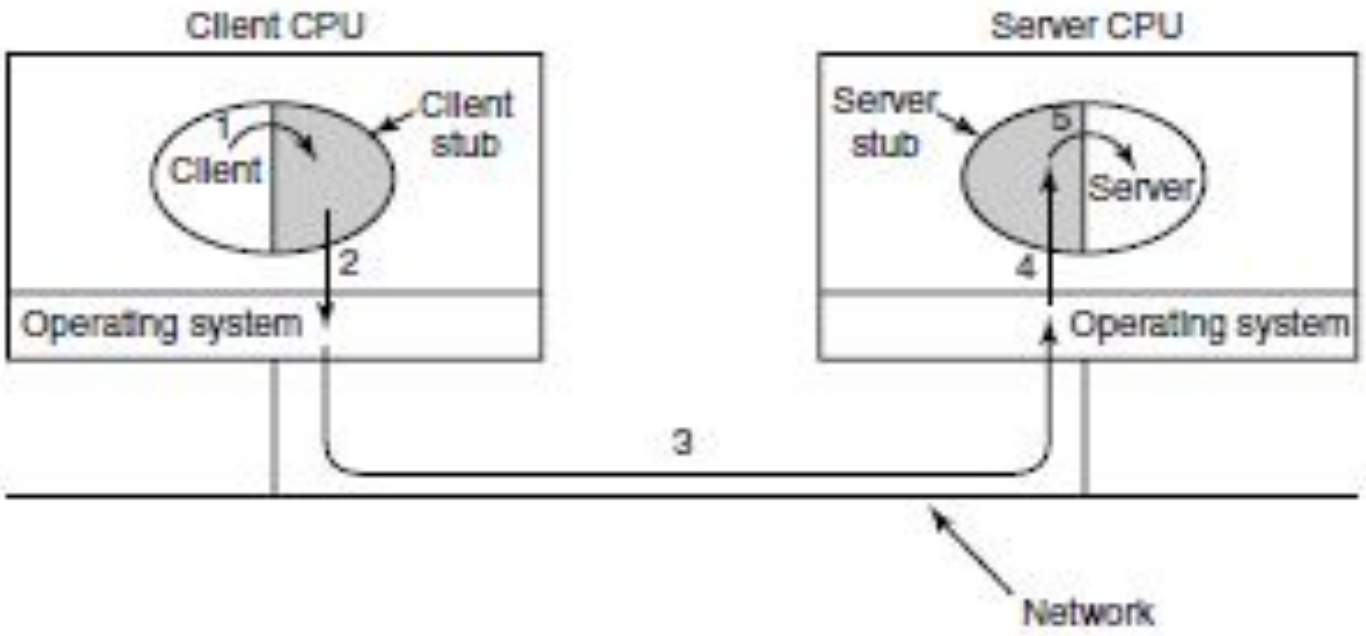


Figure 6-29. Steps in making a remote procedure call. The stubs are shaded.

- When a process on machine 1 calls a procedure on machine 2, the calling process on 1 is suspended and execution of the called procedure takes place on 2.
- Information can be transported from the caller to the callee in the parameters and can come back in the procedure result.
- No message passing is visible to the application programmer. This technique is known as **RPC (Remote Procedure Call).**
- To call a remote procedure, the client program must be bound with a small library procedure, called the **client stub.** Similarly the server is bound with a procedure called the **server stub.**

The actual steps in making an RPC:

- The client calling the client stub.
- The client stub packing the parameters into a message and making a system call to send the message. Packing the parameters is called **marshaling.**
- The operating system sending the message from the client machine to the server machine.
- The operating system passing the incoming packet to the server stub.
- The server stub calling the server procedure with the unmarshaled parameters.
- The reply traces the same path in the other direction.

THE INTERNET TRANSPORT PROTOCOLS: TCP

- **TCP (Transmission Control Protocol)** was specifically designed to provide a reliable end-to-end byte stream over an unreliable internetwork.
- TCP was formally defined in RFC 793 in September 1981.
- A port is the TCP name for a TSAP

Port	Protocol	Use
20, 21	FTP	File transfer
22	SSH	Remote login, replacement for Telnet
25	SMTP	Email
80	HTTP	World Wide Web
110	POP-3	Remote email access
143	IMAP	Remote email access
443	HTTPS	Secure Web (HTTP over SSL/TLS)
543	RTSP	Media player control
631	IPP	Printer sharing

- All TCP connections are full duplex and point-to-point.
- Full duplex means that traffic can go in both directions at the same time. Point-to-point means that each connection has exactly two end points.
- TCP does not support multicasting or broadcasting.
- TCP entity accepts user stream and divides it into IP datagrams

TCP Header:

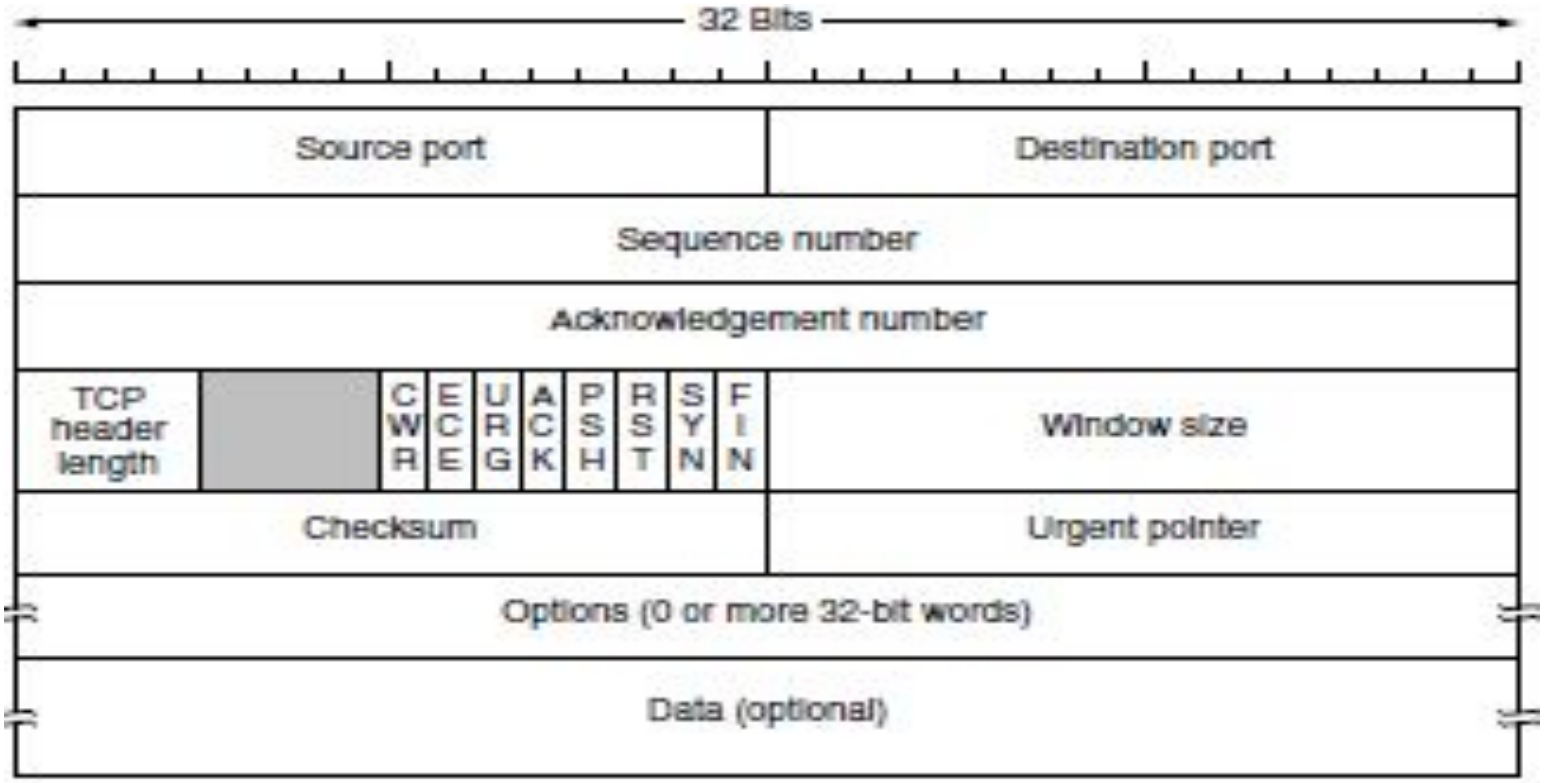


Figure 6-36. The TCP header.

- **Source Port (16-bits)** - It identifies source port of the application process on the sending device.
- **Destination Port (16-bits)** - It identifies destination port of the application process on the receiving device.
- **Sequence Number (32-bits)** - Sequence number of data bytes of a segment in a session.
- **Acknowledgement Number (32-bits)** - When ACK flag is set, this number contains the next sequence number of the data byte expected and works as acknowledgement of the previous data received.

- **CWR** - When a host receives packet with ECE bit set, it sets Congestion Windows Reduced to acknowledge that ECE received.
- **ECE** -It has two meanings:

  If SYN bit is clear to 0, then ECE means that the IP packet has its CE (congestion experience) bit set.

  If SYN bit is set to 1, ECE means that the device is ECT capable.
- **URG** - It indicates that Urgent Pointer field has significant data and should be processed.
- **ACK** - It indicates that Acknowledgement field has significance. If ACK is cleared to 0, it indicates that packet does not contain any acknowledgement.
- **PSH** - When set, it is a request to the receiving station to PUSH data (as soon as it comes) to the receiving application without buffering it.
- **RST** - Reset flag has the following features:

  It is used to refuse an incoming connection.

  It is used to reject a segment.

  It is used to restart a connection.
- **SYN** - This flag is used to set up a connection between hosts.
- **FIN** - This flag is used to release a connection and no more data is exchanged thereafter. Because packets with SYN and FIN flags have sequence numbers, they are processed in correct order.

- **Windows Size** - This field is used for flow control between two stations and indicates the amount of buffer (in bytes) the receiver has allocated for a segment, i.e. how much data is the receiver expecting.
- **Checksum** - This field contains the checksum of Header, Data and Pseudo Headers.
- **Urgent Pointer** - It points to the urgent data byte if URG flag is set to 1.
- **Options** - It facilitates additional options which are not covered by the regular header. Option field is always described in 32-bit words. If this field contains data less than 32-bit, padding is used to cover the remaining bits to reach 32-bit boundary.

TCP Connection Establishment:

- Connections are established in TCP by means of the three-way handshake
- Client initiates the connection and sends the segment with a Sequence number.
- Server acknowledges it back with its own Sequence number and ACK of client's segment which is one more than client's Sequence number.
- Client after receiving ACK of its segment sends an acknowledgement of Server's response.
- In the second case, two hosts simultaneously attempt to establish a connection between the same two sockets

- The result of these events is that just one connection is established, not two, because connections are identified by their end points.
- If the first setup results in a connection identified by (*x, y) and the second one does too, only one* table entry is made, namely, for (*x, y).*

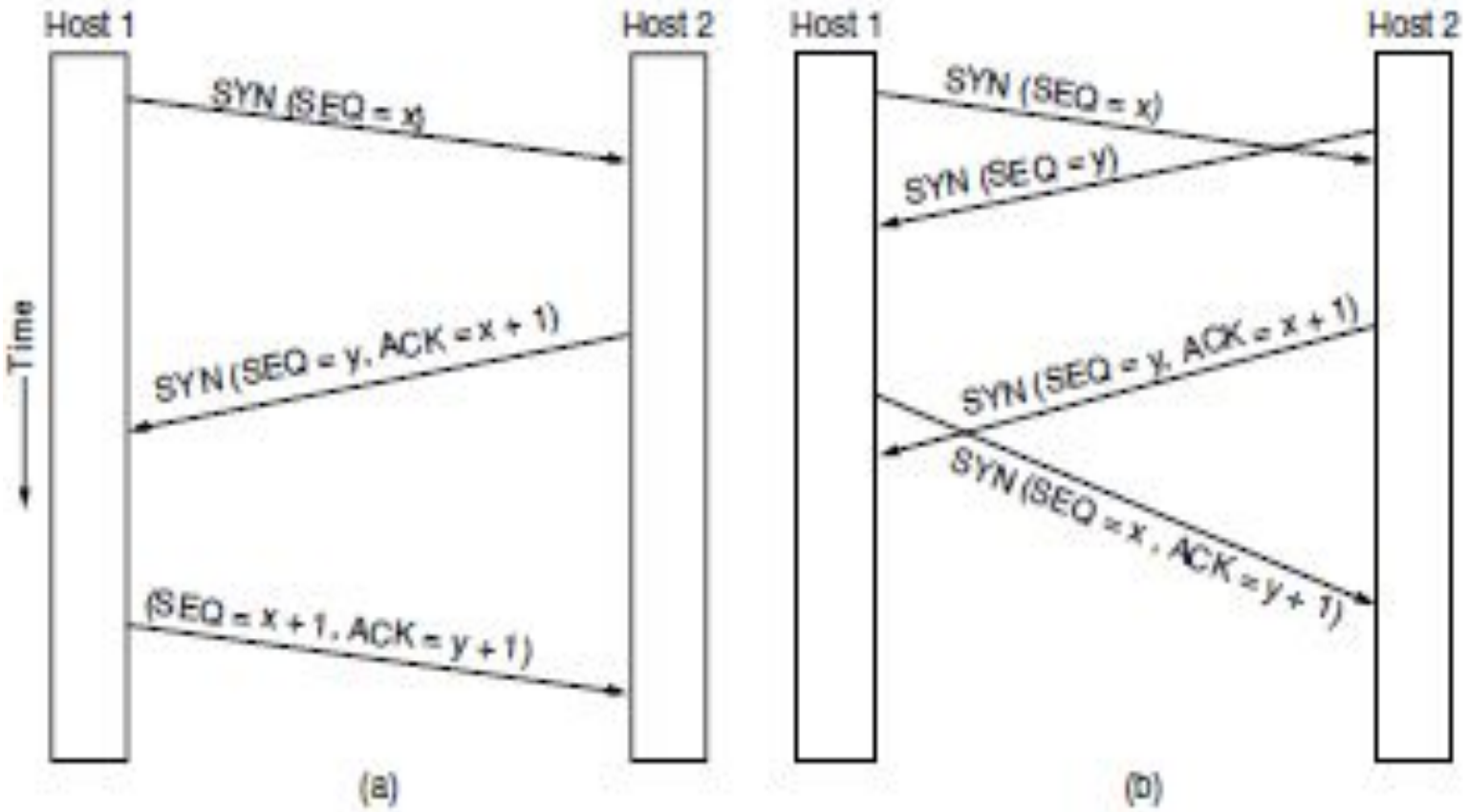


Figure 6-37. (a) TCP connection establishment in the normal case. (b) Simultaneous connection establishment on both sides.

TCP Connection Release:

- To release a connection, either party can send a TCP segment with the *FIN bit set, which means that it has no* more data to transmit.
- When the *FIN is acknowledged, that direction is shut down* for new data. Data may continue to flow indefinitely in the other direction, however.
- When both directions have been shut down, the connection is released.
- Normally, four TCP segments are needed to release a connection: one *FIN and* one *ACK for each direction.*
- *However, it is possible for the first ACK and the second FIN to be contained in the same segment, reducing the total count to three.*

Application layer:

Domain Name System(DNS):

- Network understands only numerical addresses, some mechanism is required to convert the names to network addresses. In the following sections, we will study how this mapping is accomplished in the Internet.
- It is primarily used for mapping host names to IP addresses but can also be used for other purposes.

The way DNS is used is as follows:

- To map a name onto an IP address, an application program calls a library procedure called the **resolver**, passing it the name as a parameter.

- The resolver sends a query containing the name to a local DNS server, which looks up the name and returns a response containing the IP address to the resolver, which then returns it to the caller.

The DNS Name Space:

- For the Internet, the top of the naming hierarchy is managed by an organization called **ICANN (Internet Corporation for Assigned Names and Numbers).**
- The Internet is divided into over 250 top-level domains, where each domain covers many hosts.

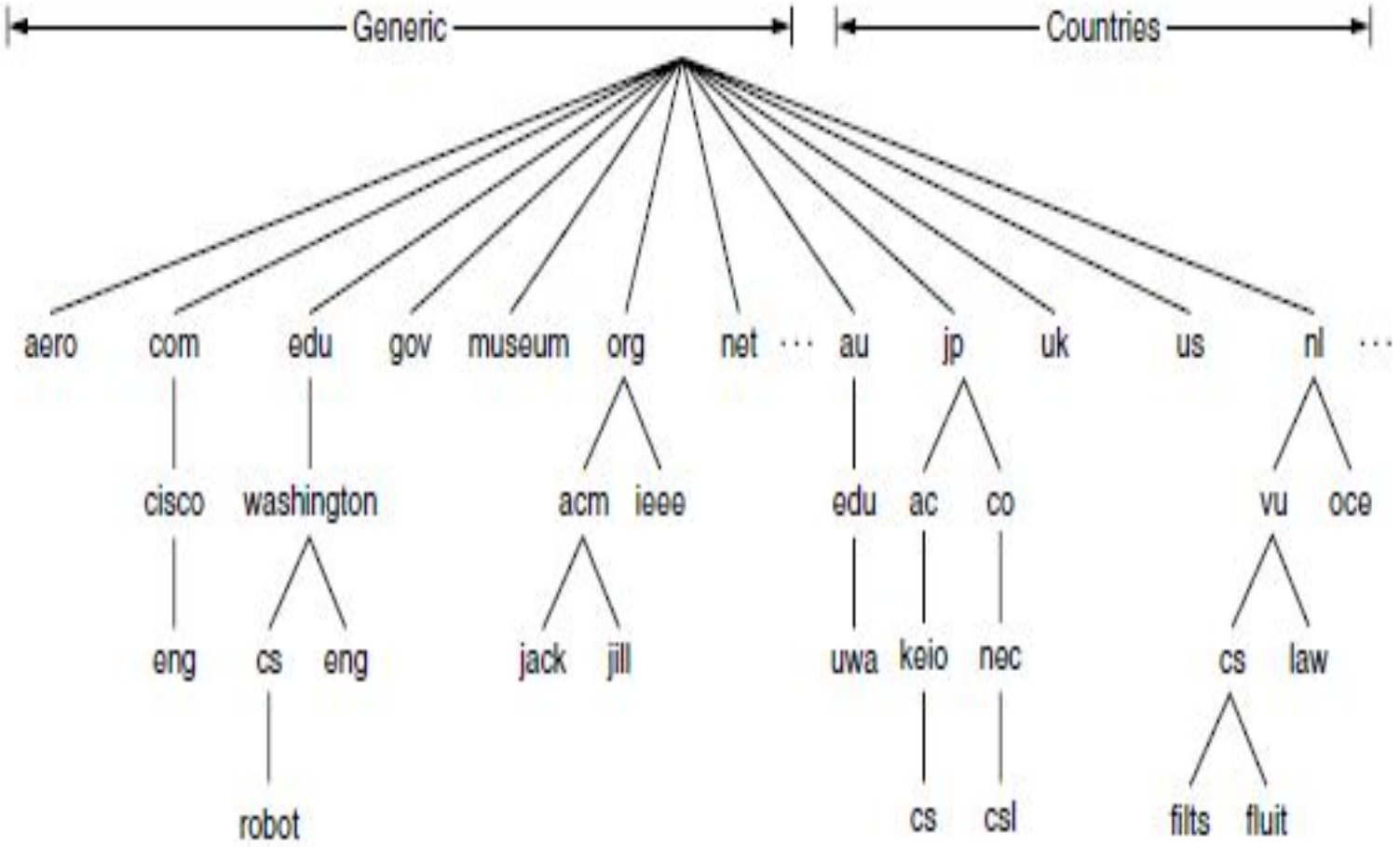


Figure 7-1. A portion of the Internet domain name space.

There are various kinds of DOMAIN :

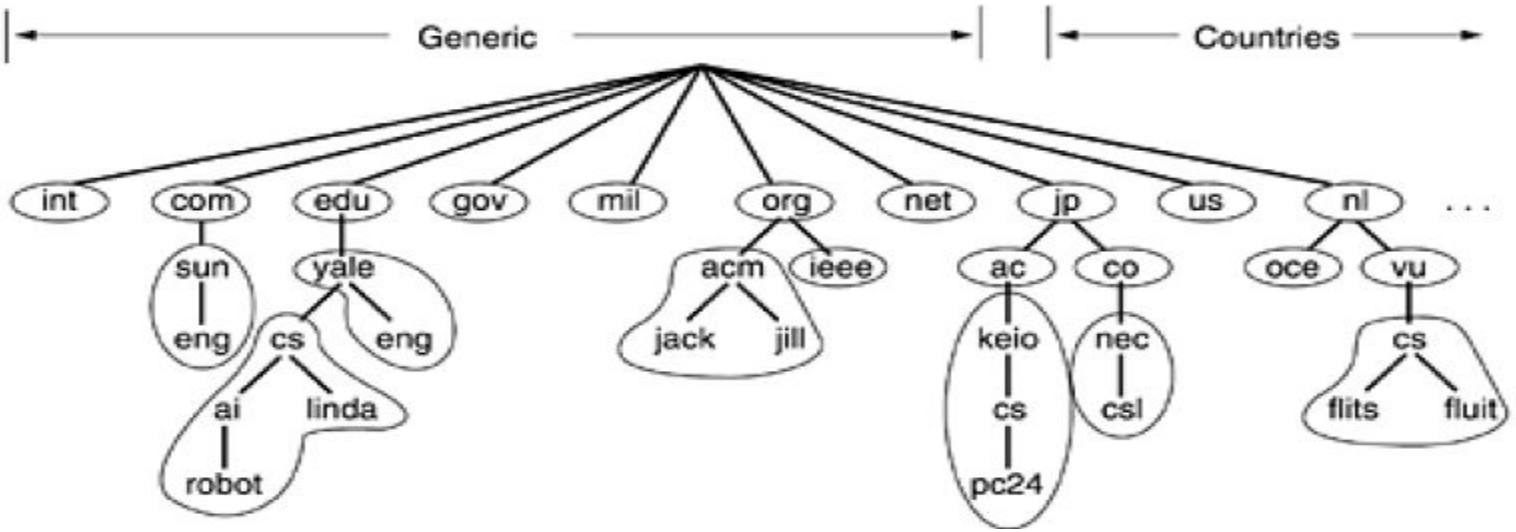
- Generic domain : .com(commercial) .edu(educational) .mil(military) .org(non profit organization) .net(similar to commercial) all these are generic domain.
- Country domain .in (india) .us .uk.

Name Servers:

- A single name server could contain the entire DNS database and respond to all queries about it. In practice, this server would be so overloaded as to be useless.
- To avoid the problems, the DNS name space is divided into nonoverlapping **zones**

- Normally, a zone will have one primary name server, which gets its information from a file on its disk, and one or more secondary name servers, which get their information from the primary name server

Figure 7-4. Part of the DNS name space showing the division into zones.

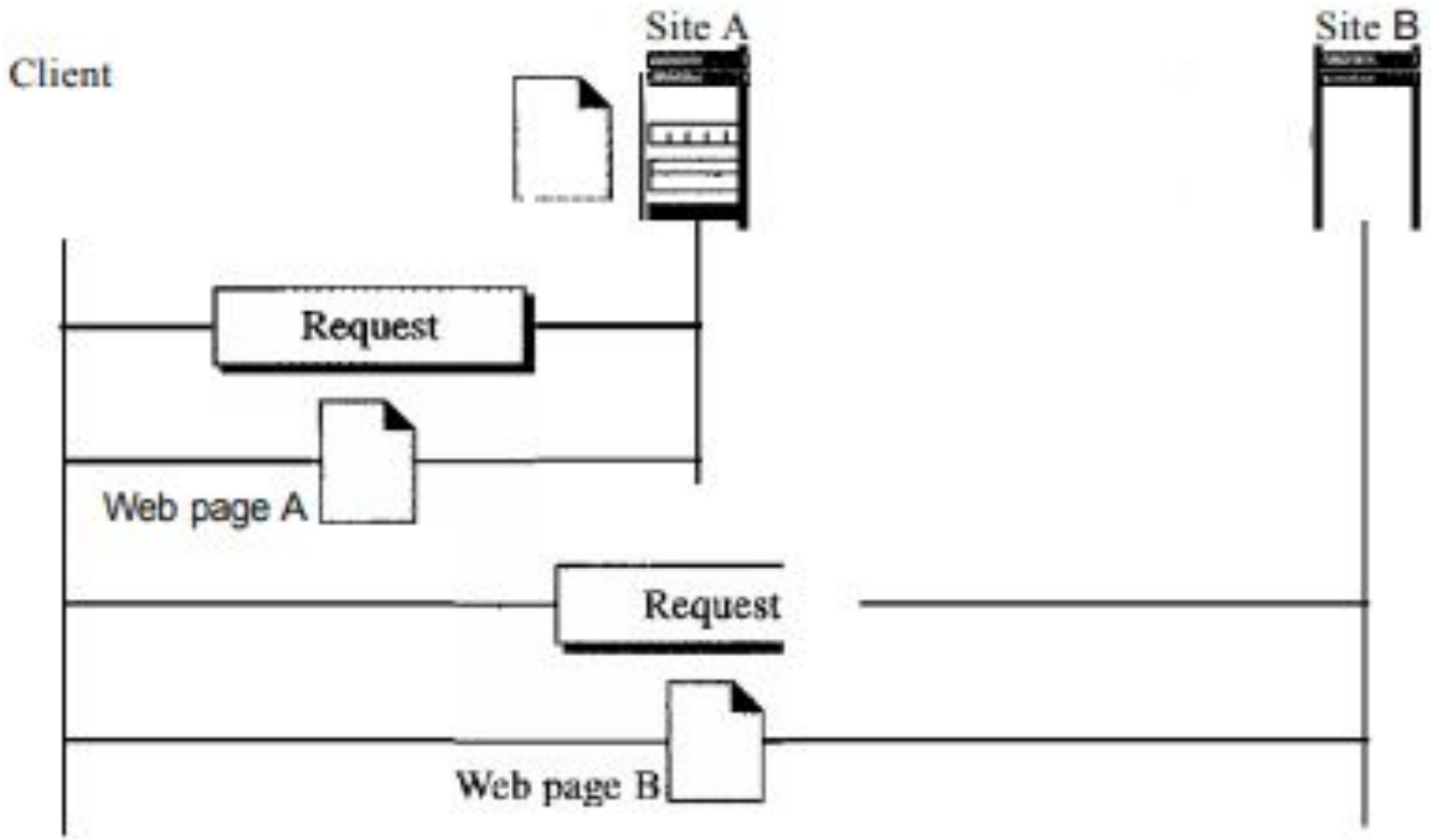


World wide Web:

- The World Wide Web (WWW) is a repository of information linked together from points all over the world.
- The WWW has a unique combination of flexibility, portability, and user-friendly features that distinguish it from other services provided by the Internet.

Architecture:

- The WWW today is a distributed client/server service, in which a client using a browser can access a service using a server.
- The service provided is distributed over many locations called sites.
- Each site holds one or more documents, referred to as Web pages. Each Web page can contain a link to other pages in the same site or at other sites. The pages can be retrieved and viewed by using browsers.

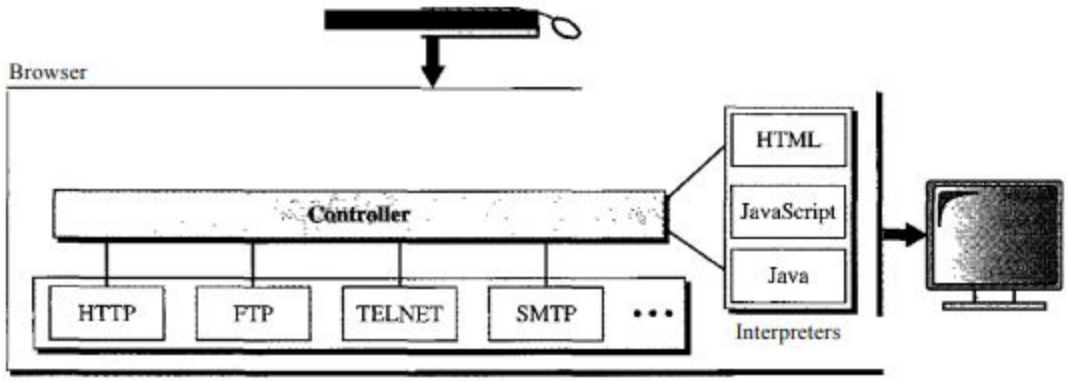

Client
Site A
Site B
Request
Web page A
Request
Web page B

Client(Browser):

- Each browser usually consists of three parts: a controller, client protocol, and interpreters.
- The controller receives input from the keyboard or the mouse and uses the client programs to access the document.
- After the document has been accessed, the controller uses one of the interpreters to display the document on the screen.
- The client protocol can be one of the protocols described previously such as FTP or HTTP (described later in the chapter). The interpreter can be HTML, Java, or JavaScript, depending on the type of document.

Server:

The Web page is stored at the server. Each time a client request arrives, the corresponding document is sent to the client. To improve efficiency, servers normally store requested files in a cache in memory; memory is faster to access than disk.



Uniform Resource Locator:

- A client that wants to access a Web page needs the address.
- HTTP uses locators.
- The uniform resource locator (URL) is a standard for specifying any kind of information on the Internet.
- The URL defines four things: protocol, host computer, port, and path.
- The **protocol** is the client/server program used to retrieve the document(Eg: FTP or HTTP).

- The host is the computer on which the information is located.
- The URL can optionally contain the port number of the server. If the *port is* included, it is inserted between the host and the path.
- Path is the pathname of the file where the information is located.

Cookies:

- An Cookie is a small piece of data sent from a website and stored on the user's computer by the user's web browser while the user is browsing.
- Cookies were designed to be a reliable mechanism for websites to remember stateful information or to record the user's browsing activity.

Creation and Storage of Cookies:

- When a server receives a request from a client, it stores information about the client in a file or a string.
- The server includes the cookie in the response that it sends to the client.
- When the client receives the response, the browser stores the cookie in the cookie directory, which is sorted by the domain server name.

WEB DOCUMENTS:

- The documents in the WWW can be grouped into three broad categories: static, dynamic, and active.

Static Documents:

- Static documents are fixed-content documents that are created and stored in a server.
- The client can get only a copy of the document.

Dynamic Documents:

- A dynamic document is created by a Web server whenever a browser requests the document.
- When a request arrives, the Web server runs an application program or a script that creates the dynamic document.
- A fresh document is created for each request.

Active Documents:

- For many applications, we need a program or a script to be run at the client site. These are called active documents.
- When a browser requests an active document, the server sends a copy of the document or a script. The document is then run at the client (browser) site.

HTTP:

- Hypertext Transfer Protocol (HTTP) is an application-level protocol.
- HTTP is a TCP/IP based communication protocol, that is used to deliver data (HTML files, image files, query results, etc.) on the World Wide Web. The default port is TCP 80, but other ports can be used as well.
- The most current version of HTTP is 1.1.

Features:

HTTP is connectionless.

HTTP is media independent.

HTTP is stateless

HTTP Connections:

Non-persistent connection is known as HTTP 1.0 and Persistent connection is known as HTTP 1.1.

- **Non-Persistent Connection**: It requires connection setup again and again for each object to send.

The following lists the steps in this strategy:

1. The client opens a TCP connection and sends a request.
2. The server sends the response and closes the connection.
3. The client reads the data until it encounters an end-of-file marker; it then closes the connection.

- **Persistent connection**: It does not require connection setup again and again. Multiple objects can use connection. The server leaves the connection open for more requests after sending a response

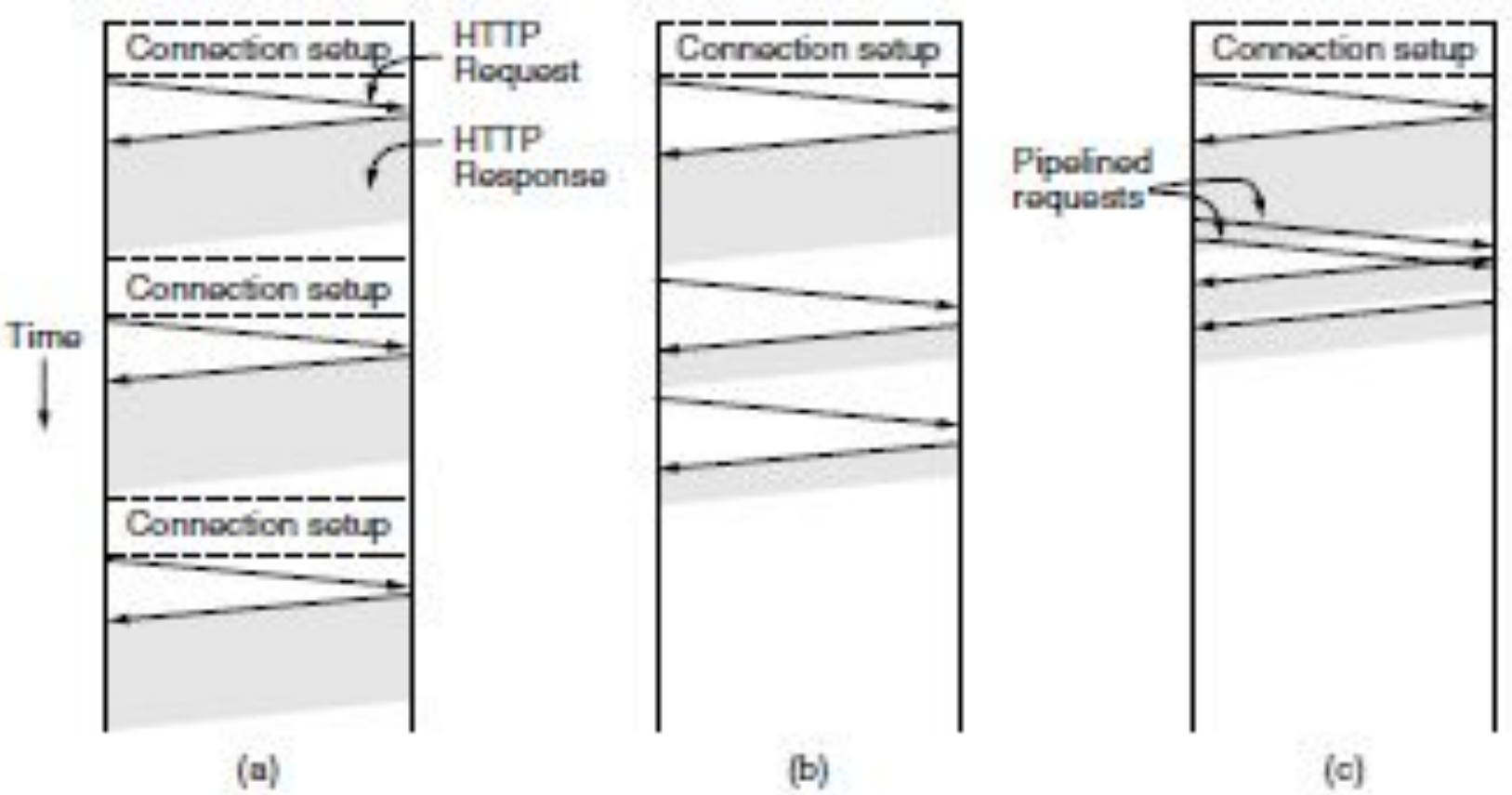


Figure 7-36. HTTP with (a) multiple connections and sequential requests. (b) A persistent connection and sequential requests. (c) A persistent connection and pipelined requests.

HTTP Transaction:

The client initializes the transaction by sending a request message. The server replies by sending a response.

Messages:

- The formats of the request and response messages are similar.
- A request message consists of a request line, a header, and sometimes a body. A response message consists of a status line, a header, and sometimes a body.

Request and Response Messages:

Request Line
Headers
A blank line
Body(present only in some messages)

Status line
Headers
A blank line
Body(present only in some messages)

Status code: This field is used in the response message. It consists of three digits.

Request type: This field is used in the request message. In version 1.1 of HTTP, several request types are defined

Request type Methods:

Method	Description
GET	Read a Web page
HEAD	Read a Web page's header
POST	Append to a Web page
PUT	Store a Web page
DELETE	Remove the Web page
TRACE	Echo the incoming request
CONNECT	Connect through a proxy
OPTIONS	Query options for a page

Figure 7-37. The built-in HTTP request methods.

Table 27.2 *Status codes*

Code	*Phrase*	*Description*
	Informational	
100	Continue	The initial part of the request has been received, and the client may continue with its request.
101	Switching	The server is complying with a client request to switch protocols defined in the upgrade header.
	Success	
200	OK	The request is successful.
201	Created	A new URL is created.
202	Accepted	The request is accepted, but it is not immediately acted upon.
204	No content	There is no content in the body.

Code	Phrase	Description
		Redirection
301	Moved permanently	The requested URL is no longer used by the server.
302	Moved temporarily	The requested URL has moved temporarily.
304	Not modified	The document has not been modified.
		Client Error
400	Bad request	There is a syntax error in the request.
401	Unauthorized	The request lacks proper authorization.
403	Forbidden	Service is denied.
404	Not found	The document is not found.
405	Method not allowed	The method is not supported in this URL.
406	Not acceptable	The format requested is not acceptable.
		Server Error
500	Internal server error	There is an error, such as a crash, at the server site.
501	Not implemented	The action requested cannot be performed.
503	Service unavailable	The service is temporarily unavailable, but may be requested in the future.

Header :The header exchanges additional information between the client and the server.

- Each header line has a header name, a colon, a space, and a header value.
- A header line belongs to one of four categories: general header, request header, response header, and entity header
- **General header:** The general header gives general information about the message and can be present in both a request and a response.

Request header: The request header can be present only in a request message. It specifies the client's configuration and the client's preferred document format.

Response header: The response header can be present only in a response message. It specifies the server's configuration and special information about the request.

Entity header: The entity header gives information about the body of the document.

Proxy Server :

- HTTP supports proxy servers.
- A proxy server is a computer that keeps copies of responses to recent requests.
- The HTTP client sends a request to the proxy server.
- The proxy server checks its cache.
- If the response is not stored in the cache, the proxy server sends the request to the corresponding server.
- Incoming responses are sent to the proxy server and stored for future requests from other clients.
- The proxy server reduces the load on the original server, decreases traffic, and improves latency.

Electronic Mail:

One of the most popular Internet services is electronic mail(e-mail).

- Architecture consists of two kinds of subsystems: the **user agents,** which allow people to read and send email, and the **message transfer agents,** which move the messages from the source to the destination.
- We will also refer to message transfer agents informally as **mail servers.**
- The user agent is a program that provides a graphical interface, or sometimes a text- and command-based interface that lets users interact with the email system(compose, display and organize messages).

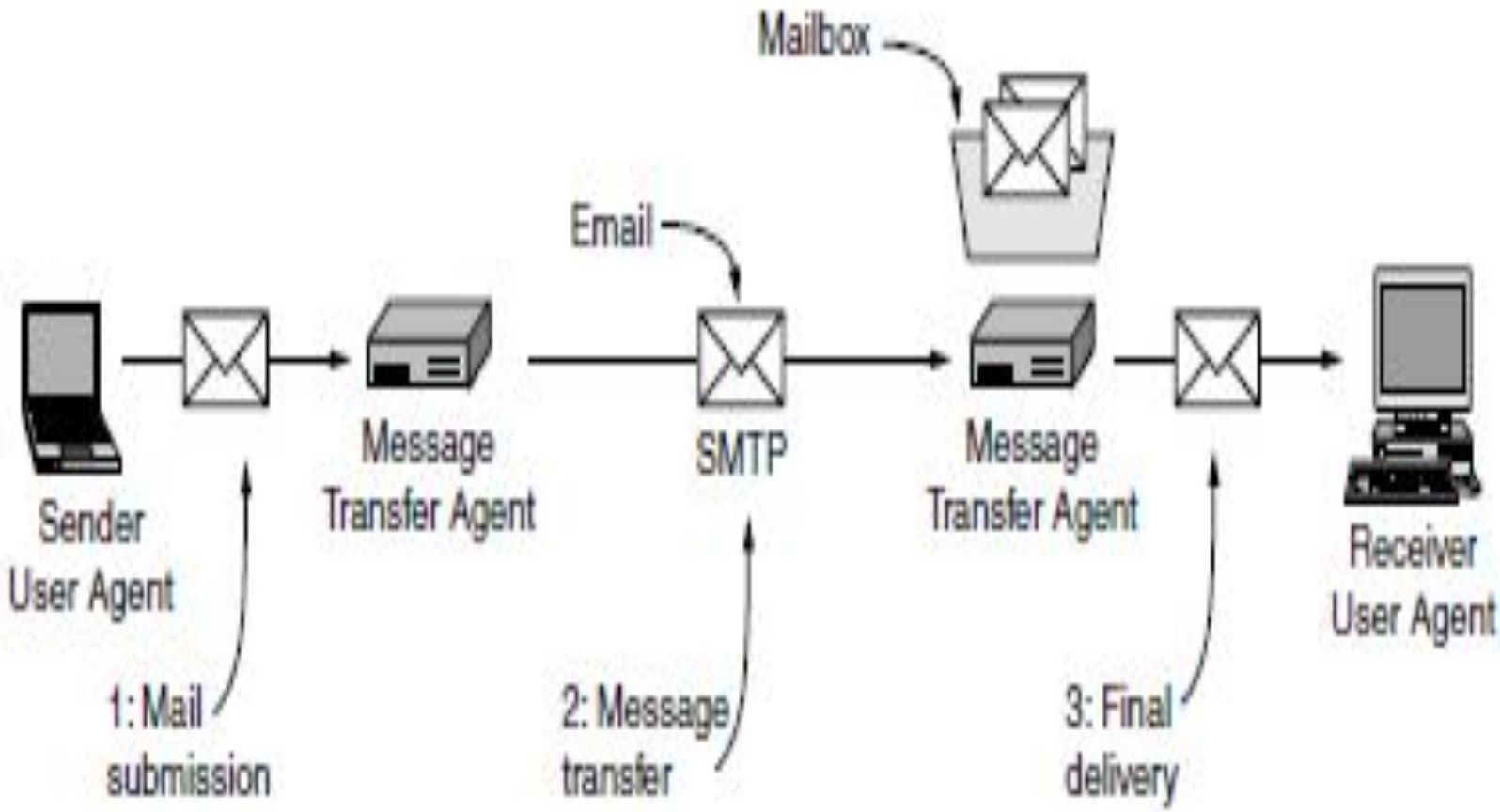


Figure 7-7. Architecture of the email system.

- The message transfer agents are typically system processes. They run in the background on mail server machines.
- Their job is to automatically move email through the system from the originator to the recipient with **SMTP (Simple Mail Transfer Protocol).**
- SMTP sends mail over connections and reports back the delivery status and any errors.
- Linking user agents and message transfer agents are the concepts of mailboxes.
- **Mailboxes** store the email that is received for a user. They are maintained by mail servers. User agents simply present users with a view of the contents of their mailboxes.
- To do this, the user agents send the mail servers commands to manipulate the mailboxes, inspecting their contents, deleting messages, and so on. The retrieval of mail is the final delivery.

Message Access Agent: POP and IMAP

The first and the second stages of mail delivery use SMTP because SMTP is a push protocol; it pushes the message from the client to the server.

The third stage needs a pull protocol; the client must pull messages from the server.

The third stage uses a message access agent.

- Currently two message access protocols are available: Post Office Protocol, version 3 (POP3)
- Internet Mail Access Protocol, version 4 (IMAP4).

- POP3 has two modes: the delete mode and the keep mode.
- IMAP4 is similar to POP3, but it has more features:
- A user can search the contents of the e-mail.
- A user can partially download e-mail(If bandwidth is low).
- A user can create, delete, or rename mailboxes.
- A user can create a hierarchy of mailboxes in a folder for e-mail storage

Simple Network Management Protocol:

- The Simple Network Management Protocol (SNMP) is a framework for managing devices in an internet using the TCP/IP protocol suite.
- It provides a set of fundamental operations for monitoring and maintaining an internet.
- A manager, usually a host, controls and monitors a set of agents, usually routers.
- The manager is a host that runs the SNMP client program.
- The agent is a router or host that runs the SNMP server program.

- SNMP uses the services of two other protocols: Structure of Management Information(SMI) and Management Information Base (MIB).
- SMI names objects, defines the type of data that can be stored in an object, and encodes the data.
- MIB is a collection of groups of objects that can be managed by SNMP.
- SNMP defines eight types of packets: GetRequest, GetNextRequest, SetRequest, GetBulkRequest, Trap, InformRequest, Response, and Report.

SNMP functions in three ways:

1. A manager can retrieve the value of an object defined in an agent.
2. A manager can store a value in an object defined in an agent.
3. An agent can send an alarm message to the manager.