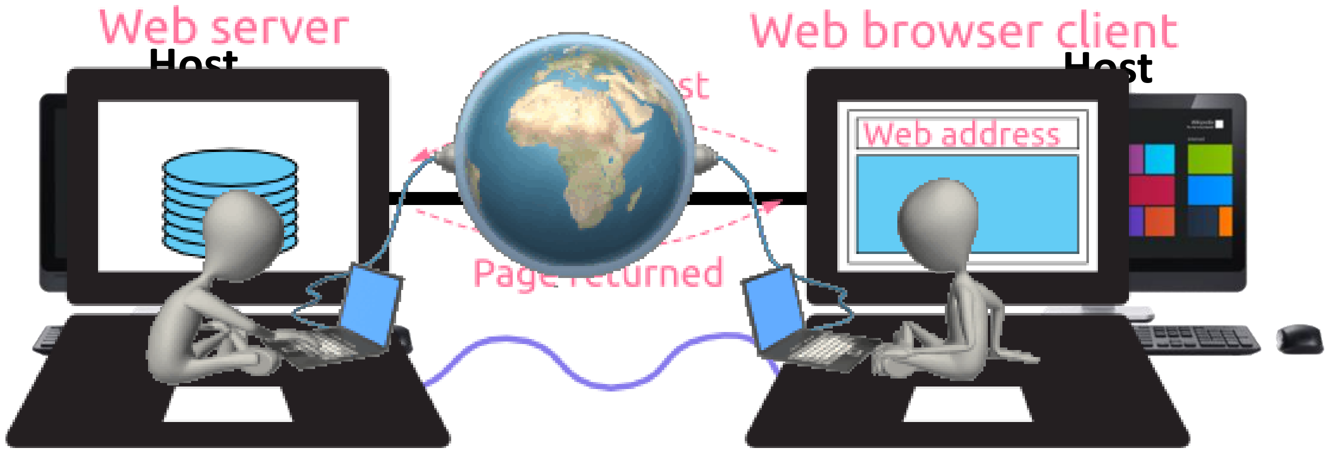
Computer Networks

Unit-6

Application Layer

Network Applications

- A Network application is an application running on one host and provides a communication to another application running on a different host.
- A network application development is writing programs that run on different end systems and communicate with each other over the network.
- In the Web application there are two different programs that communicate with each other:
  - ✓ Browser program running in the user's host.
  - ✓ Web server program running in the Web server host.



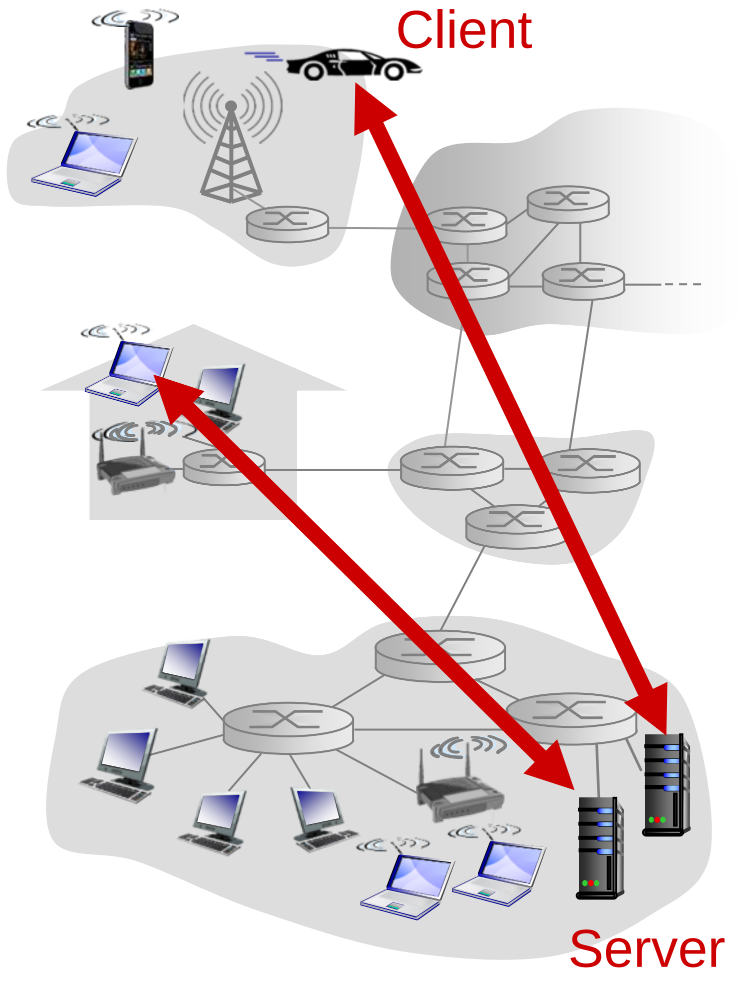
Network Applications - Examples

- Email
- Web
- Remote Login
- P2P File Sharing
- Multi-user Network Games
- Streaming Stored Video (YouTube)
- Voice Over IP (Skype)
- Real-time Video Conference
- Social Networking

Network Application Architecture

1. Client-Server architecture
2. P2P (Peer to Peer) architecture

1. Client-Server Architecture



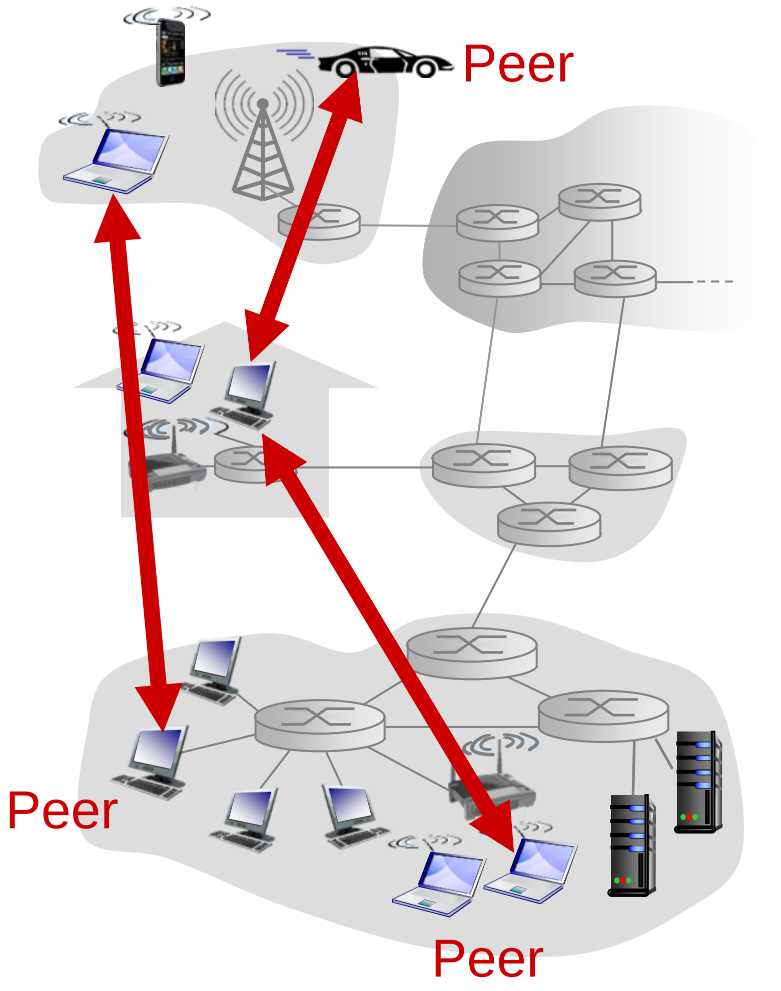
Server:

- ✓ Its always-on host.
- ✓ It has a fixed IP address.
- ✓ Large cluster of host – Data Centers.
- ✓ E.g. Web Server

Client:

- ✓ It communicate with server.
- ✓ Its not like continuously connected.
- ✓ May have dynamic IP addresses.
- ✓ Do not communicate directly with each other.
- ✓ E.g. PCs, Mobiles

2. P2P Architecture



- Peers (end systems) directly communicate.
- Get peers request service from other peers, provide service to other peers.
  - ✓ Self Scalability – New peers bring new service capacity, as well as new service demands.
- Peers are alternatingly connected and change IP addresses.
  - ✓ Complex management

Process Communicating

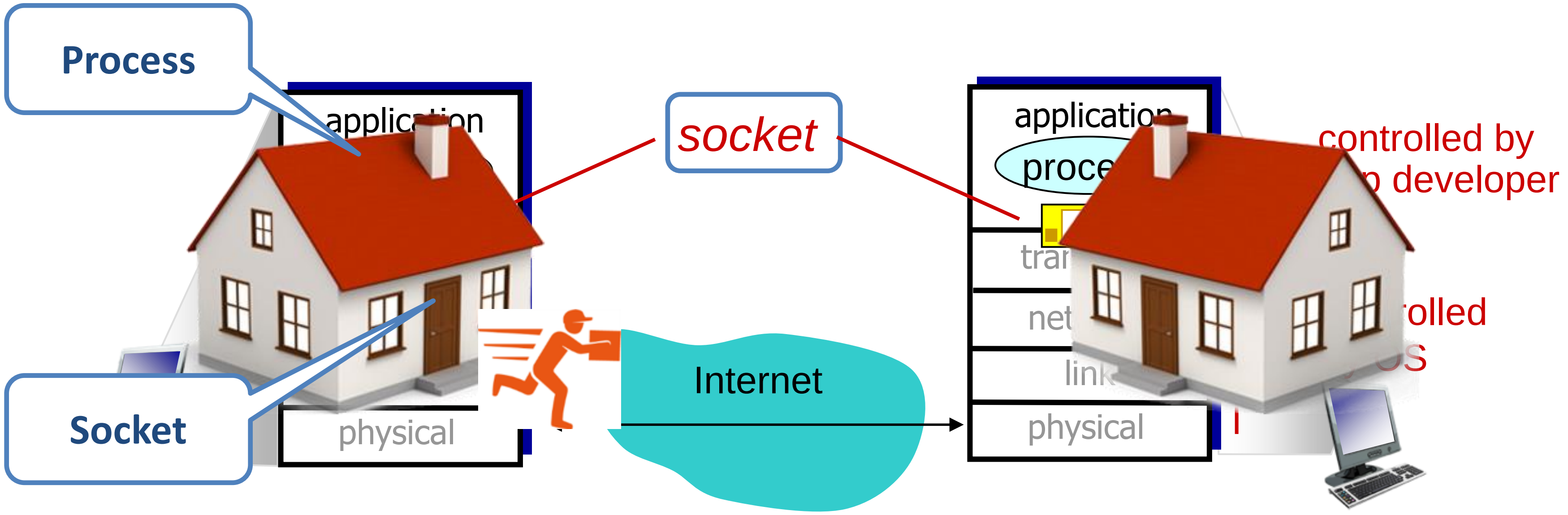
- What is Process?
- A process is an instance of a program running in a computer.
- We can say that process is program under execution.
- Within er-process commu
- Process
- Client p
- Server

≡ File Edit Search Run Compile Debug Project Options Window Help
FIRST.C
1=[↕]
Inter-Process Communication
Process A
Process B
4:19
F1 Help Alt-F
Make F10 Menu

```
#include<stdio.h>
#include<conio.h>
void main()
{
        clrscr();
        printf("H
        getch();
}
```

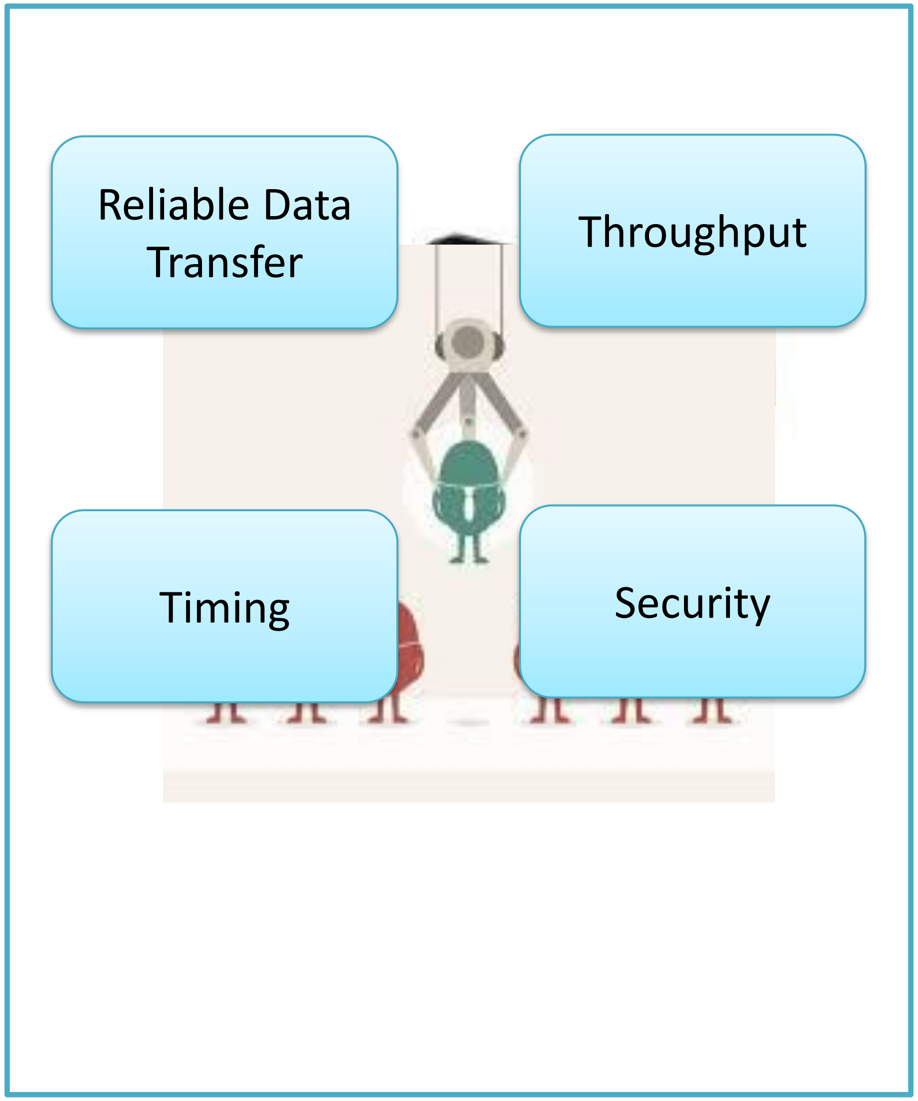
Socket

- A process sends messages into, and receives messages from; the network through a software interface called a socket.
- A process is similar to a house and its socket is similar to its door.
  - ✓ Sending process passes message out door.
  - ✓ Sending process relies on transport infrastructure on other side of door to deliver message to socket at receiving process.



Transport Services to Applications

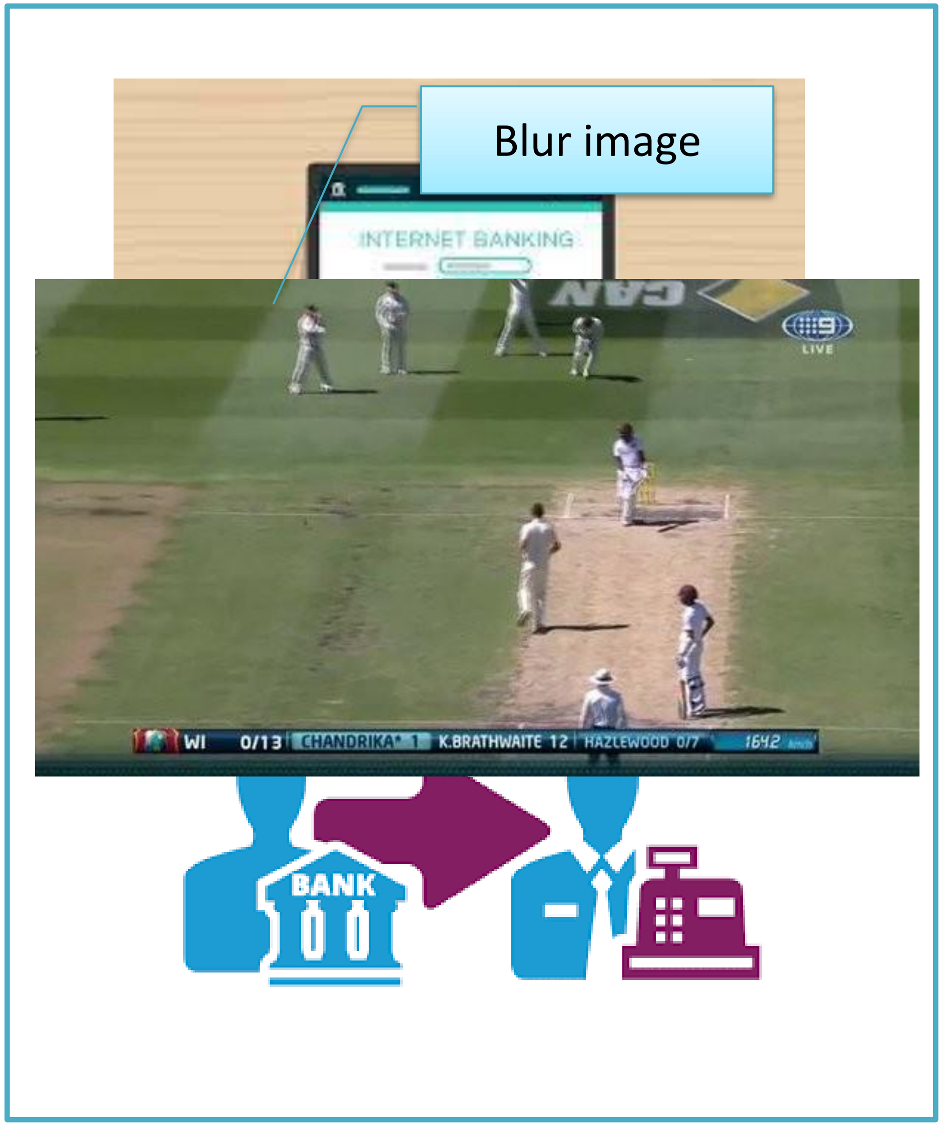
- Recall that a **socket** is the **interface** between the application process and the transport layer protocol.
- For **develop** an application, **choose** available transport layer protocol.
- Pick the protocol with the **services** that best match the needs of your application.
- **Example**: Choose either Train or Airplane transport for travel between two cities.
- Classify services with four parameters.



Transport Services to Applications

Reliable Data Transfer

- ✓ Many applications (e.g., email, file transfer, financial applications) require 100% reliable data transfer
- ✓ Required guarantee that data sent by one end of application is delivered correctly and completely to the other end of application.
- ✓ This guaranteed data delivery service is called Reliable Data Transfer.
- ✓ When it will fail to deliver reliable data transfer, it is acceptable for loss-tolerant applications.
- ✓ Loss-tolerant Applications (e.g., audio/video) can tolerate some loss.



Transport Services to Applications

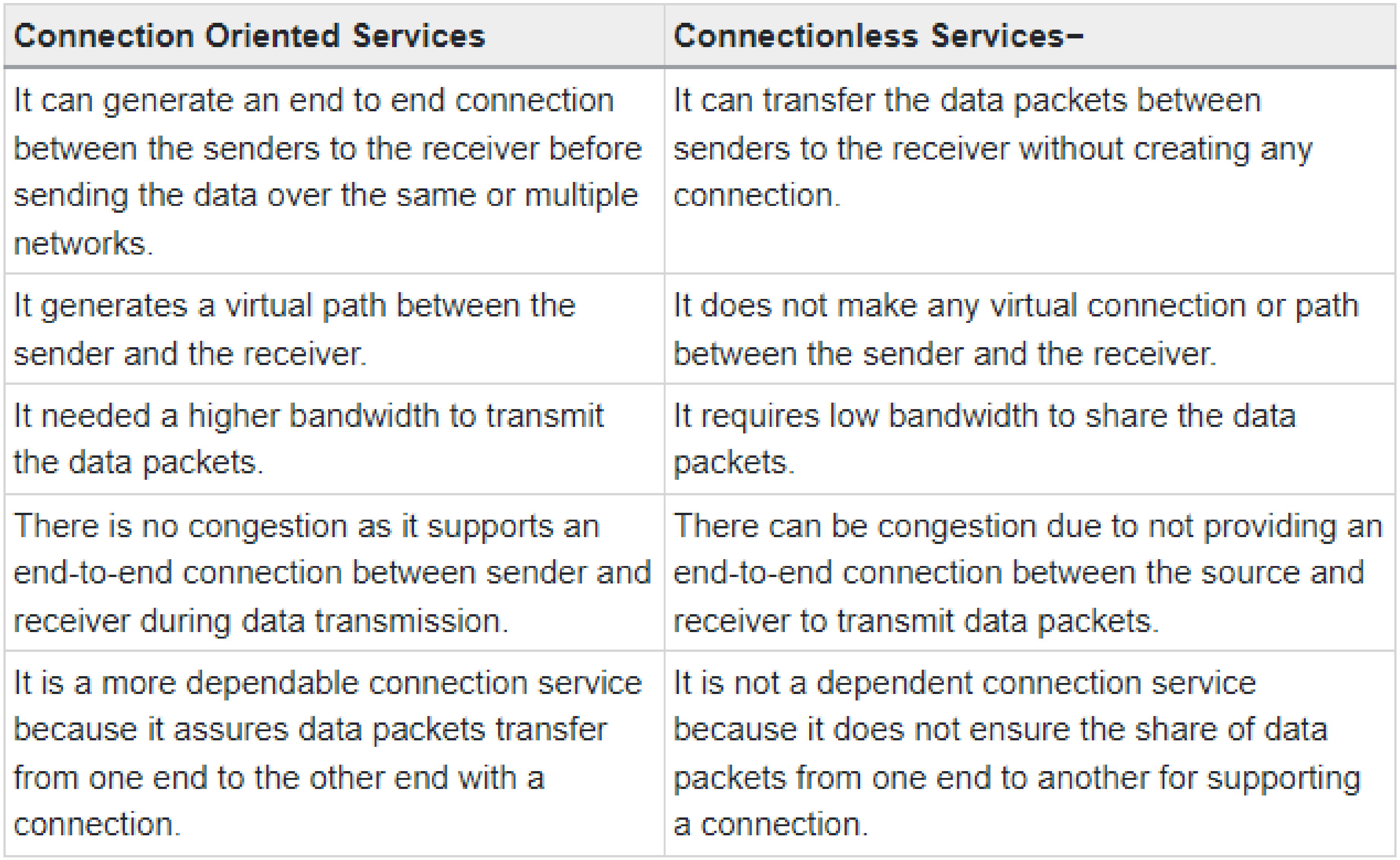
- **Throughput**
  - ✓ some apps (e.g., multimedia) require at least amount of throughput to be "effective"
  - ✓ Bandwidth sensitive application, specific throughput required.
  - ✓ Elastic application can use of as much, or as little, throughput as happens to be available.
- **Timing**
  - ✓ some apps (e.g., Internet telephony, interactive games) require low delay to be "effective"
- **Security**
  - ✓ In the sending host, encrypt all data transmitted by the sending process.
  - ✓ In the receiving host, decrypt the data before delivering the data to the receiving process.

Internet Transport Protocols Services

- **TCP Service:**
  - ✓ Connection-Oriented: A setup required between client and server processes
  - ✓ Reliable data transfer between sending and receiving process without error and proper order
  - ✓ Congestion control: To control sender when network overloaded
  - ✓ It does not provide, Timing, at least throughput guarantee (not preferred in real-time application)

- **UDP Services:**
  - ✓ Connectionless: No connection before two processes start to communicate.
  - ✓ Unreliable data transfer between sending and receiving process
  - ✓ It does not provide congestion control.
  - ✓ It Does not provide. Reliability, flow control, throughput guarantee, security.

Distinguish between Connection-Oriented and Connectionless Service



Connection Oriented Services	Connectionless Services-
It can generate an end to end connection between the senders to the receiver before sending the data over the same or multiple networks.	It can transfer the data packets between senders to the receiver without creating any connection.
It generates a virtual path between the sender and the receiver.	It does not make any virtual connection or path between the sender and the receiver.
It needed a higher bandwidth to transmit the data packets.	It requires low bandwidth to share the data packets.
There is no congestion as it supports an end-to-end connection between sender and receiver during data transmission.	There can be congestion due to not providing an end-to-end connection between the source and receiver to transmit data packets.
It is a more dependable connection service because it assures data packets transfer from one end to the other end with a connection.	It is not a dependent connection service because it does not ensure the share of data packets from one end to another for supporting a connection.

S. No	Comparison Parameter	Connection-oriented Service	Connection Less Service
1.	Related System	It is designed and developed based on the telephone system.	It is service based on the postal system.
2.	Definition	It is used to create an end to end connection between the senders to the receiver before transmitting the data over the same or different network.	It is used to transfer the data packets between senders to the receiver without creating any connection.
3.	Virtual path	It creates a virtual path between the sender and the receiver.	It does not create any virtual connection or path between the sender and the receiver.
4.	Authentication	It requires authentication before transmitting the data packets to the receiver.	It does not require authentication before transferring data packets.
5.	Data Packets Path	All data packets are received in the same order as those sent by the sender.	Not all data packets are received in the same order as those sent by the sender.
6.	Bandwidth Requirement	It requires a higher bandwidth to transfer the data packets.	It requires low bandwidth to transfer the data packets.
7.	Data Reliability	It is a more reliable connection service because it guarantees data packets transfer from one end to the other end with a connection.	It is not a reliable connection service because it does not guarantee the transfer of data packets from one end to another for establishing a connection.
8.	Congestion	There is no congestion as it provides an end-to-end connection between sender and receiver during transmission of data.	There may be congestion due to not providing an end-to-end connection between the source and receiver to transmit of data packets.
9.	Examples	Transmission Control Protocol (TCP) is an example of a connection-oriented service.	User Datagram Protocol (UDP), Internet Protocol (IP), and Internet Control Message Protocol (ICMP) are examples of connectionless service.

Internet Applications

- Popular internet applications with their application layer and their underlying transport protocol.

Applications	**Application-Layer Protocol**	**Underlying Transport Protocol (Service)**
Email	SMTP	TCP
Remote Terminal Access	Telnet	TCP
Web	HTTP	TCP
File Transfer	FTP	TCP
Streaming Media	HTTP(YouTube), RTP	TCP or UDP
Internet Telephony	SIP, RTP(Skype)	Typically UDP

Loss-tolerant

No loss, Elastic Bandwidth

Application-Layer Protocols

- **an application-layer protocol defines:**
  - ✓ **The types of messages exchanged**, for example, request messages and response messages
  - ✓ **The syntax of the various message types**, such as the fields in the message and how the fields are delineated
  - ✓ **The semantics of the fields**, that is, the meaning of the information in the fields
  - ✓ **Rules for determining when and how a process sends messages and responds to messages**

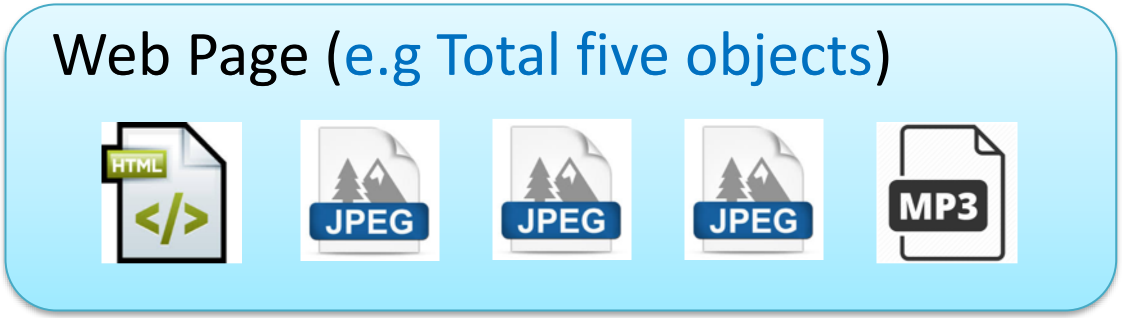
WEB & HTTP

Web

- Early 1990, Internet was used only by researchers, academics, and university students.
- New application WWW arrived in 1994 by Tim Berners-Lee.
- World Wide Web - is an information where documents and other web resources are identified by URL, interlinked by hypertext links, and can be accessed via the Internet.
- On demand available, What they want, When they want it.
- Unlike TV and Radio.
- Navigate through Websites.

Web and HTTP

- Web page consists of objects.
- Object can be HTML file, JPEG image, Java applet, audio file etc....
- Web page consists of base HTML-file which includes several referenced objects.

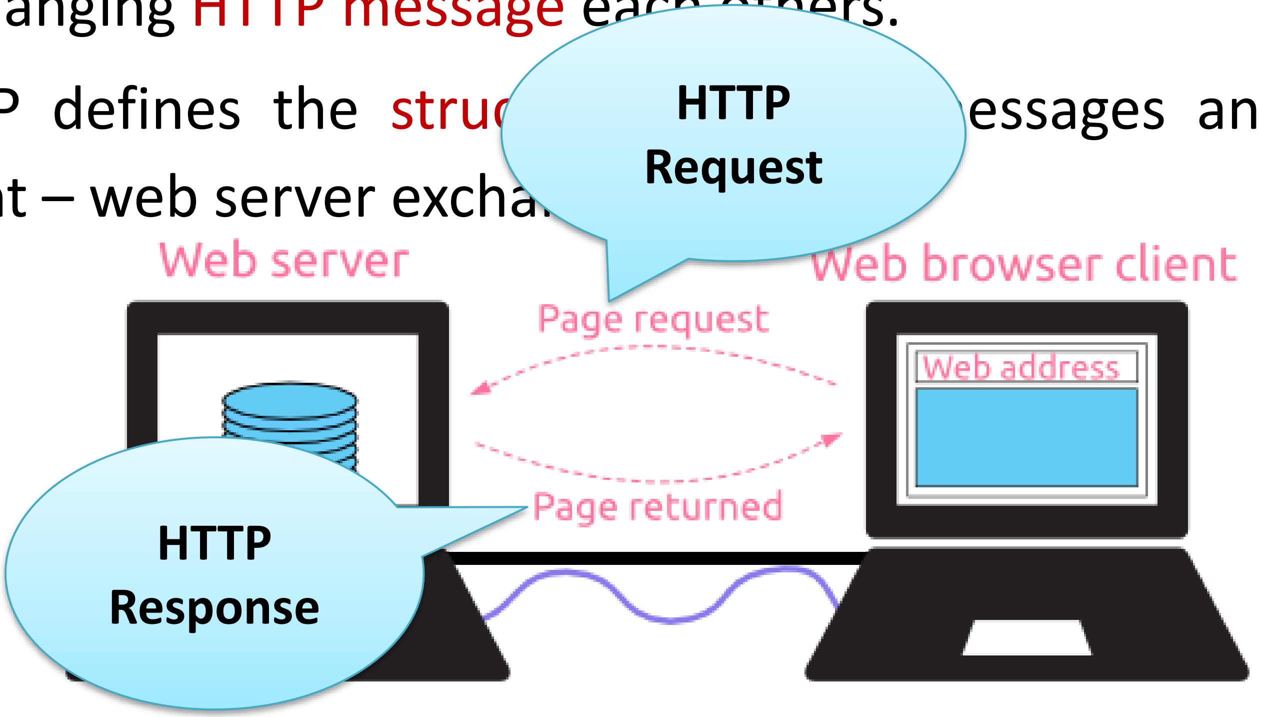


- each object is addressable by a Uniform Resource Locator (URL), like;

$$\underbrace{\texttt{www.someschool.edu}}_{\text{host name}}\underbrace{\texttt{/someDept/pic.gif}}_{\text{path name}}$$

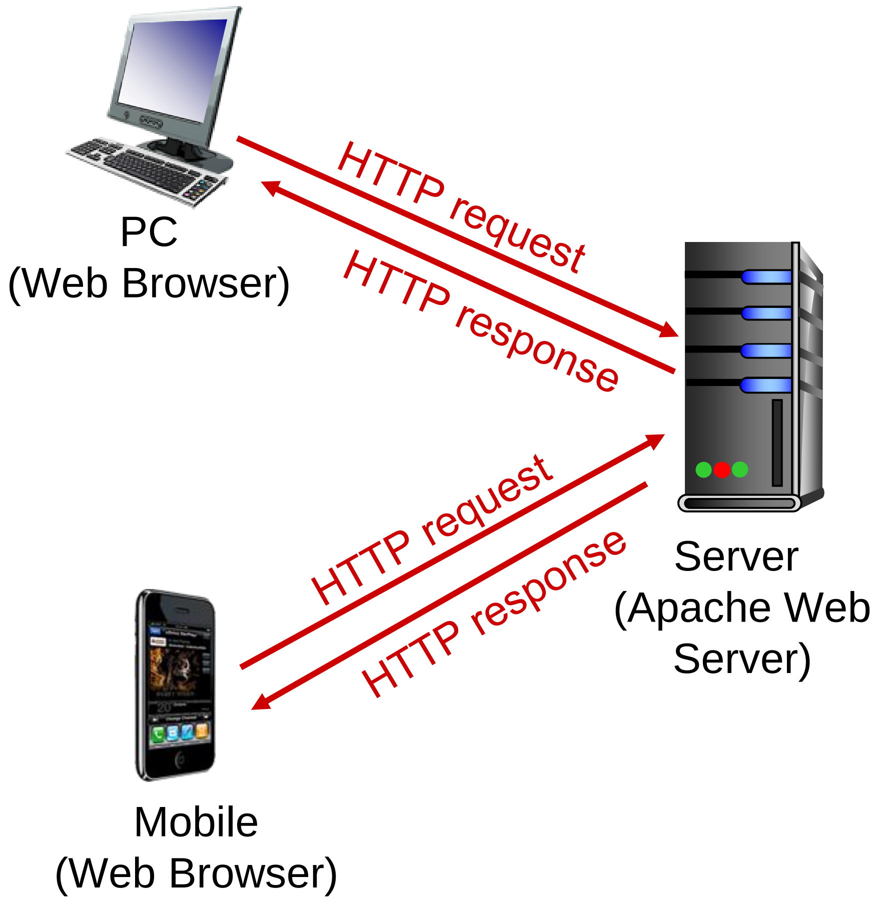
HTTP

- HyperText Transfer Protocol – Application layer protocol
- it is implemented in two programs.
  1. Client Program
  2. Server Program
- Exchanging HTTP message each others.
- HTTP defines the struc essages and how web client – web server excha



HTTP – Cont…

- HTTP
  - ✓ Hyper-Text Transfer Protocol
  - ✓ It is Application layer protocol
  - ✓ Client: A browser that requests, receives, (using HTTP protocol) and "displays" Web objects.
  - ✓ E.g. PC, Mobile
  - ✓ Server*:* Web server sends (using HTTP protocol) objects in response to requests.
  - ✓ E.g. Apache Web Server



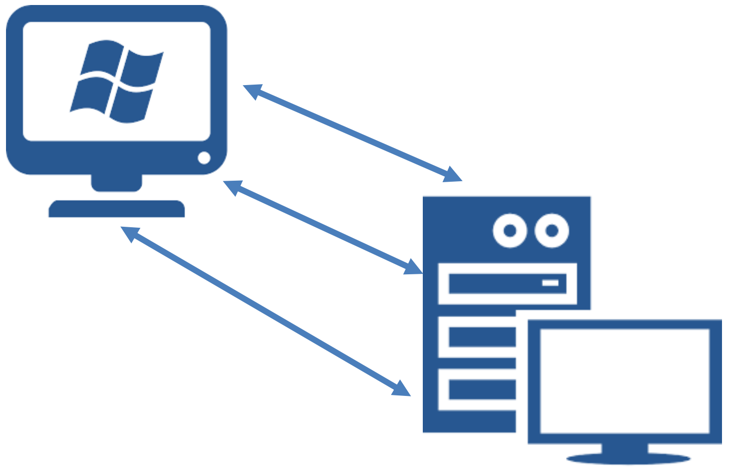
HTTP - Cont...

- A client initiates TCP connection (creates socket) to server using port 80.
- A server accepts TCP connection from client.
- HTTP messages (application-layer protocol messages) exchanged between browser (HTTP client) and Web server (HTTP server).
- HTTP is "stateless protocol", server maintains no information about past client requests.
- HTTP connection types are:
  1. Non-persistent HTTP
  2. Persistent HTTP

NON-PERSISTENT HTTP & PERSISTENT HTTP CONNECTION

Non-persistent & Persistent Connection

- In Client-Server communication, Client making a series of requests to server, Server responding to each of the requests.
- Series of requests may be made back to back or periodically at regular time interval.
- So, Application developer need to make an important decision;
    - ✓ Should each request/response pair be sent over a separate TCP connection.
    - ✓ **OR** should all of the requests and corresponding responses be sent over same TCP connection?



1. Non-persistent HTTP

- A non-persistent connection is closed after the server sends the requested object to the client.
- The connection is used exactly for one request and one response.
- For downloading multiple objects it required multiple connections.
- Non-persistent connections are the default mode for HTTP/1.0.
- Example:
- Transferring a webpage from server to client, webpage consists of a base HTML file and 10 JPEG images.
- Total 11 object are reside on server.

1. Non-persistent HTTP – Cont….

URL: `www.someSchool.edu/someDepartment/home.index`

1a. HTTP client initiates TCP connection to HTTP server (process) at www.someSchool.edu on port 80

1b. HTTP server at host www.someSchool.edu waiting for TCP connection at port 80. "accepts" connection, notifying client

2. HTTP client sends HTTP *request message* (containing URL) into TCP connection socket. Message indicates that client wants object someDepartment/home.index

3. HTTP server receives request message, forms *response message* containing requested object, and sends message into its socket

4. HTTP server closes TCP connection.

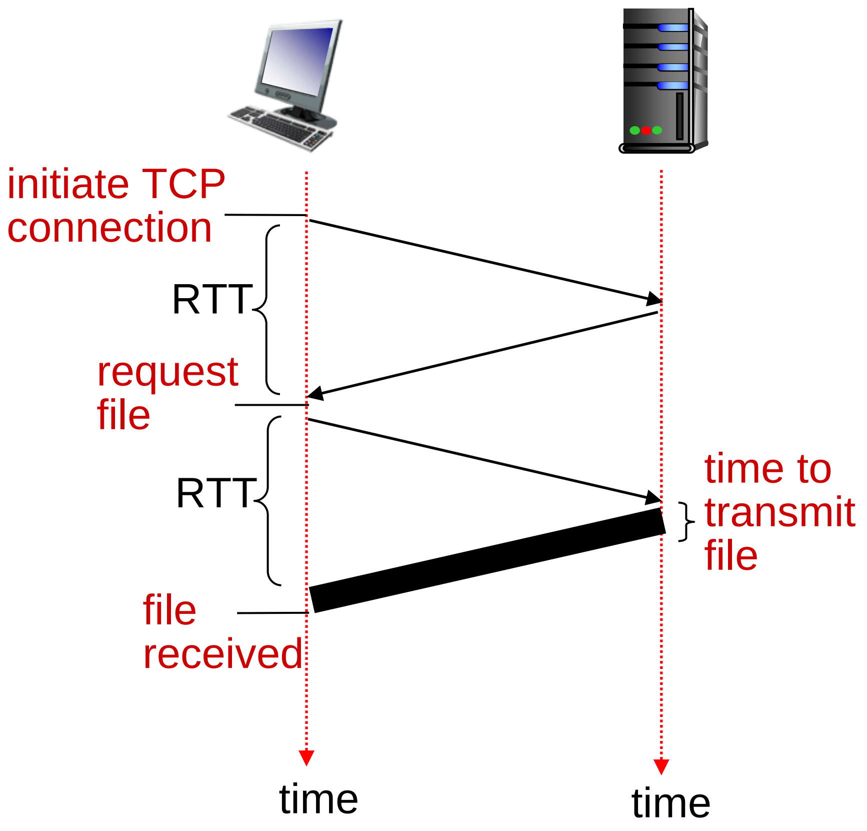
5. HTTP client receives response message containing html file, displays html. Parsing html file, finds 10 referenced jpeg objects

6. Steps 1-5 repeated for each of 10 jpeg objects

Time

1. Non-persistent HTTP: Response time

- **RTT(round-trip time):** A time for a small packet to travel from client to server and vice versa.
- **HTTP response time:**
  - ✓ one RTT to initiate TCP connection.
  - ✓ one RTT for HTTP request and first few bytes of HTTP response to return.
  - ✓ File transmission time



Non-persistent HTTP response time = 2RTT + file transmission time

2. Persistent HTTP

- Server leaves the TCP connection open after sending responses.
- Subsequent HTTP messages between same client and server sent over open connection.
- The server closes the connection only when it is not used for a certain configurable amount of time.
- It requires as little as one round-trip time (RTT) for all the referenced objects.
- With persistent connections, the performance is improved by 20%.
- Persistent connections are the default mode for HTTP/1.1.

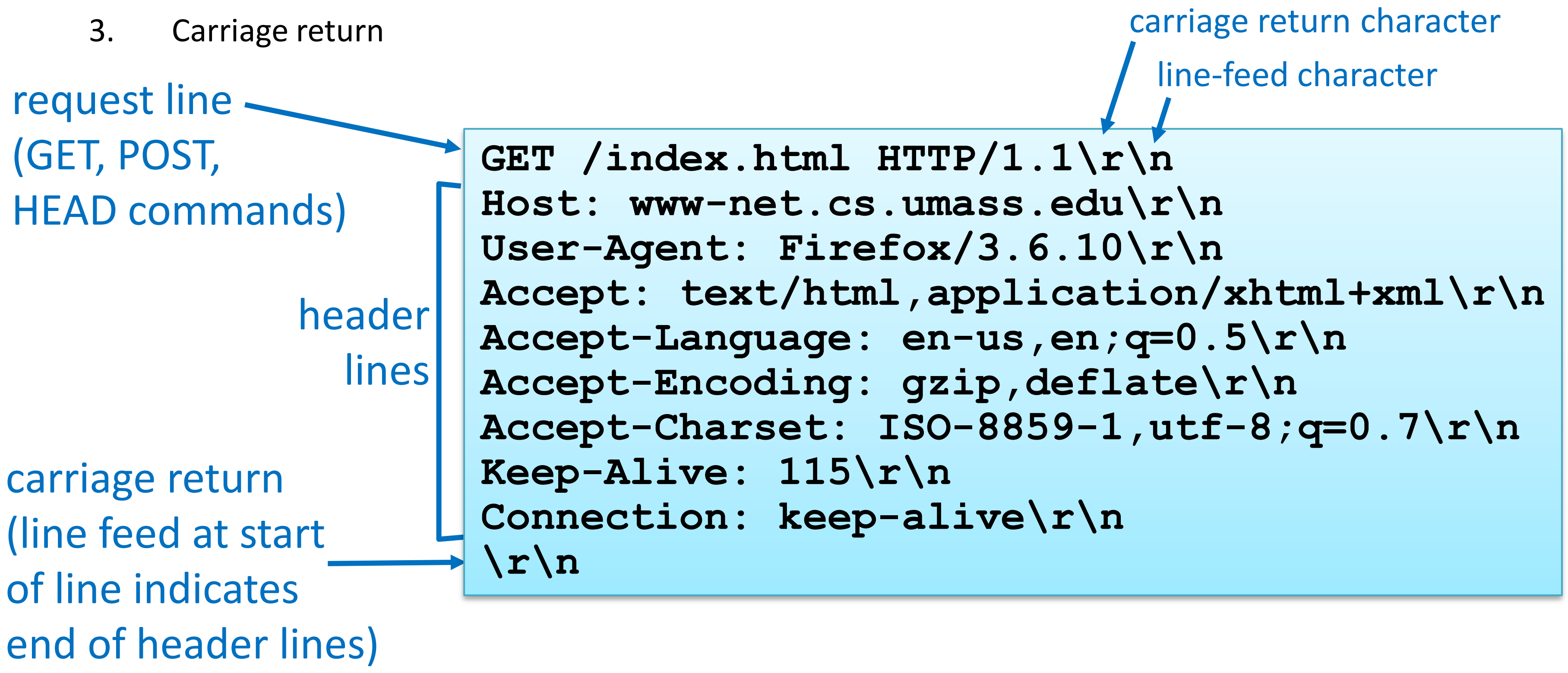
HTTP Message Format

- Two types:
  1. Request Message
  2. Response Message

1. HTTP Request Message

- It is in ASCII format which means that human-
- readable format.
- HTTP request message consist three part:
  1. Request line
  2. Header line
  3. Carriage return

GET
POST
PUT
HEAD
DELETE
PATCH
OPTIONS



carriage return character
line-feed character
request line (GET, POST, HEAD commands)
header lines
carriage return (line feed at start of line indicates end of header lines)

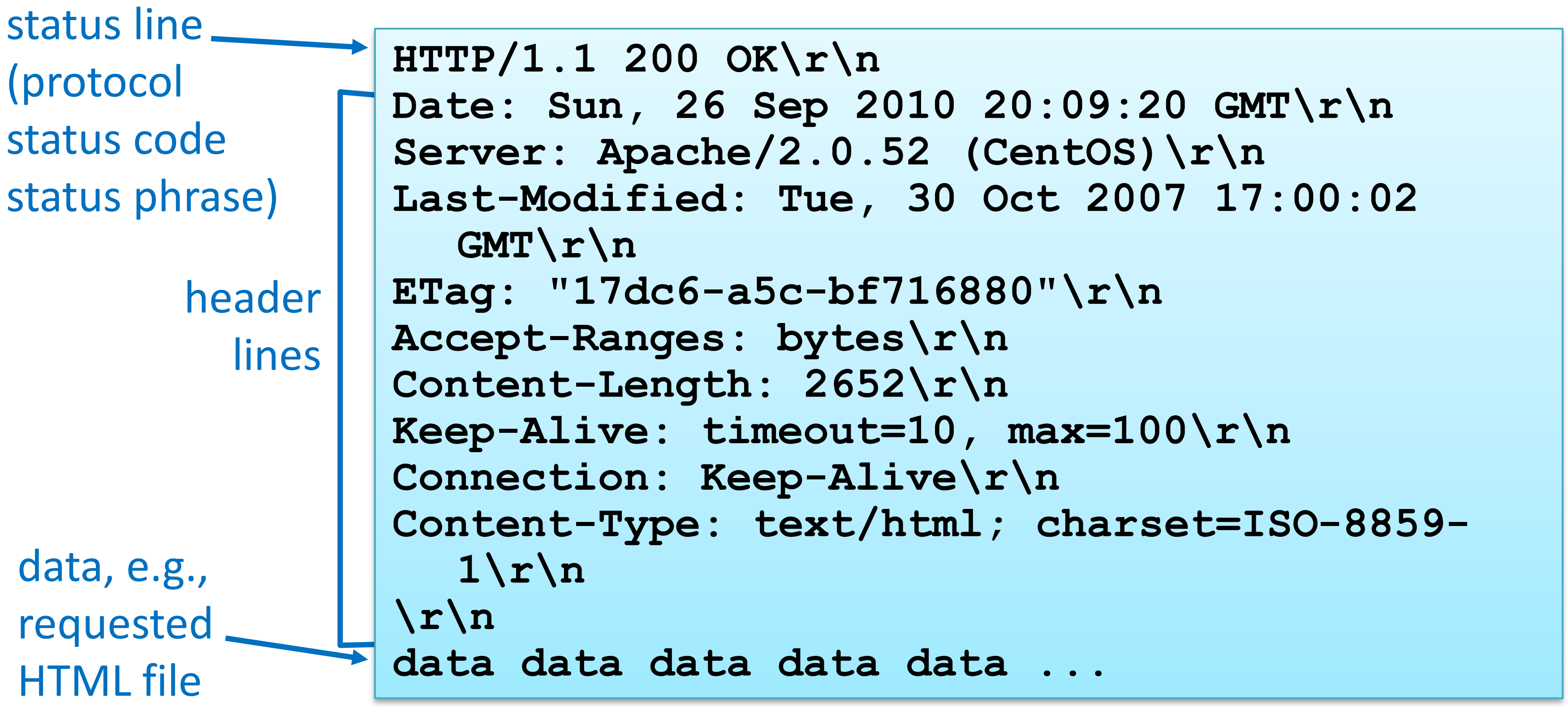
```
GET /index.html HTTP/1.1\r\n
Host: www-net.cs.umass.edu\r\n
User-Agent: Firefox/3.6.10\r\n
Accept: text/html,application/xhtml+xml\r\n
Accept-Language: en-us,en;q=0.5\r\n
Accept-Encoding: gzip,deflate\r\n
Accept-Charset: ISO-8859-1,utf-8;q=0.7\r\n
Keep-Alive: 115\r\n
Connection: keep-alive\r\n
\r\n
```

1. HTTP Request Message - Format

- The request line has three fields: Method field, URL field, and HTTP version field.
- The method field can take on several different values, including GET, POST, HEAD, PUT, and DELETE.
- In above message, browser is requesting the object `/somedir/page.html` and version is self-explanatory; browser implements version HTTP/1.1.
- The header line Host: `www-net.cs.umass.edu` specifies the host on which the object resides.
- User agent indicate browser name and version.

2. HTTP Response Message

- HTTP response message consist of three part:
  1. Status line
  2. Header line
  3. Data (Entity body)



```
HTTP/1.1 200 OK\r\n
Date: Sun, 26 Sep 2010 20:09:20 GMT\r\n
Server: Apache/2.0.52 (CentOS)\r\n
Last-Modified: Tue, 30 Oct 2007 17:00:02
   GMT\r\n
ETag: "17dc6-a5c-bf716880"\r\n
Accept-Ranges: bytes\r\n
Content-Length: 2652\r\n
Keep-Alive: timeout=10, max=100\r\n
Connection: Keep-Alive\r\n
Content-Type: text/html; charset=ISO-8859-
   1\r\n
\r\n
data data data data data ...
```

2. HTTP Response Message - Format

- The status line has three fields: protocol version field, status code and corresponding status message.
- In below example, the status line indicates that the server is using HTTP/1.1 and that everything is OK.

```
HTTP/1.1 200 OK\r\n
Date: Sun, 26 Sep 2010 20:09:20 GMT\r\n
Server: Apache/2.0.52 (CentOS)\r\n
Last-Modified: Tue, 30 Oct 2007 17:00:02 GMT\r\n
ETag: "17dc6-a5c-bf716880"\r\n
Accept-Ranges: bytes\r\n
Content-Length: 2652\r\n
Keep-Alive: timeout=10, max=100\r\n
Connection: Keep-Alive\r\n
Content-Type: text/html; charset=ISO-8859-1\r\n \r\n
data data data data data ...
```

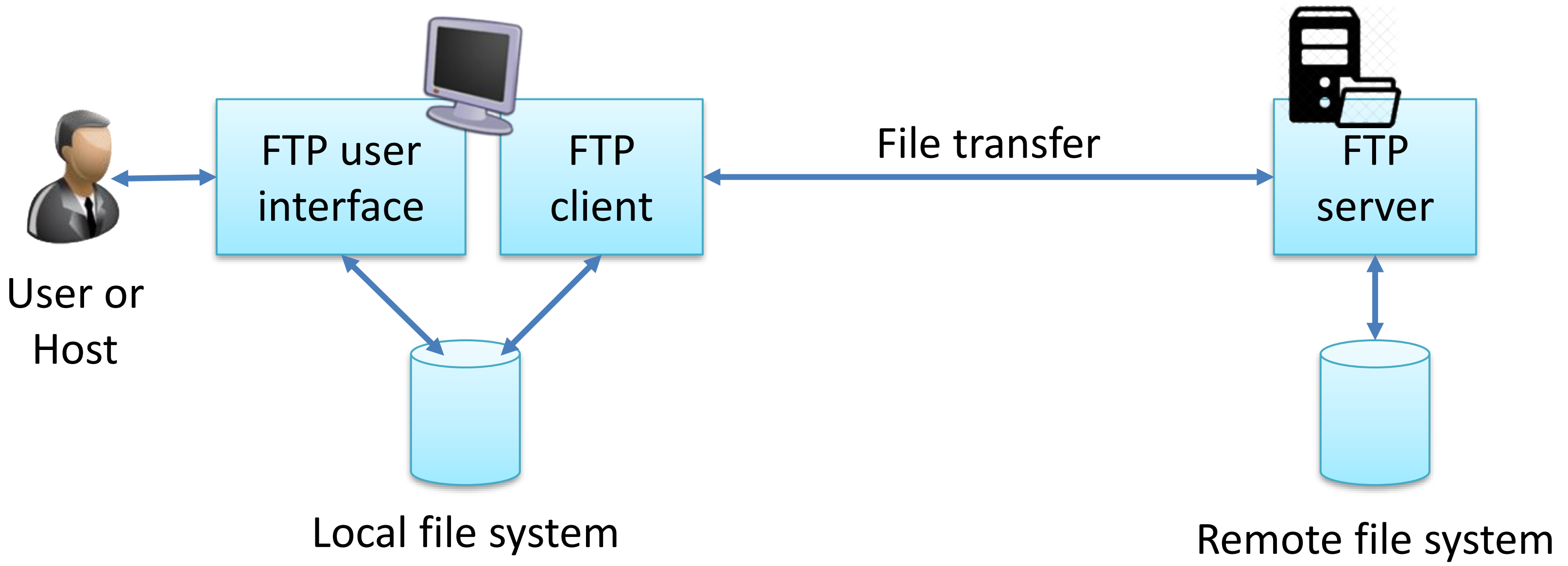
[illegible]

HTTP Response Status Codes

- A status code appears in 1st line in server-to-client response message.
- Some sample codes:
  - ✓ 200 OK
    - Request succeeded, requested object later in this message
  - ✓ 301 Moved Permanently
    - Requested object moved, new location specified later in this message(Location)
  - ✓ 400 Bad Request
    - Request message not understood by server
  - ✓ 404 Not Found
    - Requested document not found on this server
  - ✓ 505 HTTP Version Not Supported
    - Requested http version not support

FTP (File Transfer Protocol)

- File Transfer Protocol (FTP) is the commonly used protocol for exchanging files over the Network or Internet. **Example: Filezilla**
- FTP uses the Internet's TCP/IP protocols to enable data transfer.
- FTP uses client-server architecture.
- FTP promotes sharing of files via remote computers with reliable and efficient data transfer.

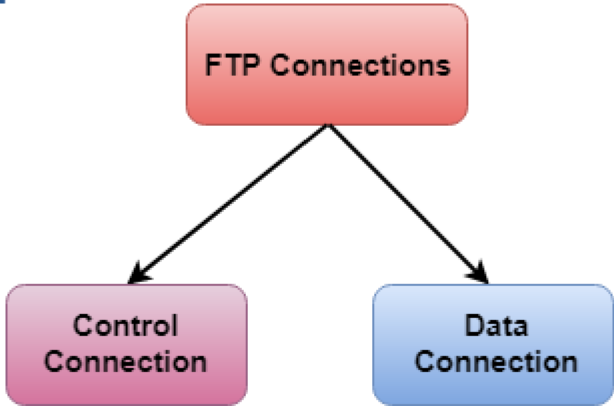


FTP (File Transfer Protocol) – Cont…

- FTP client connect FTP server at port 21 using TCP.
- FTP uses two parallel TCP connections to transfer a file,

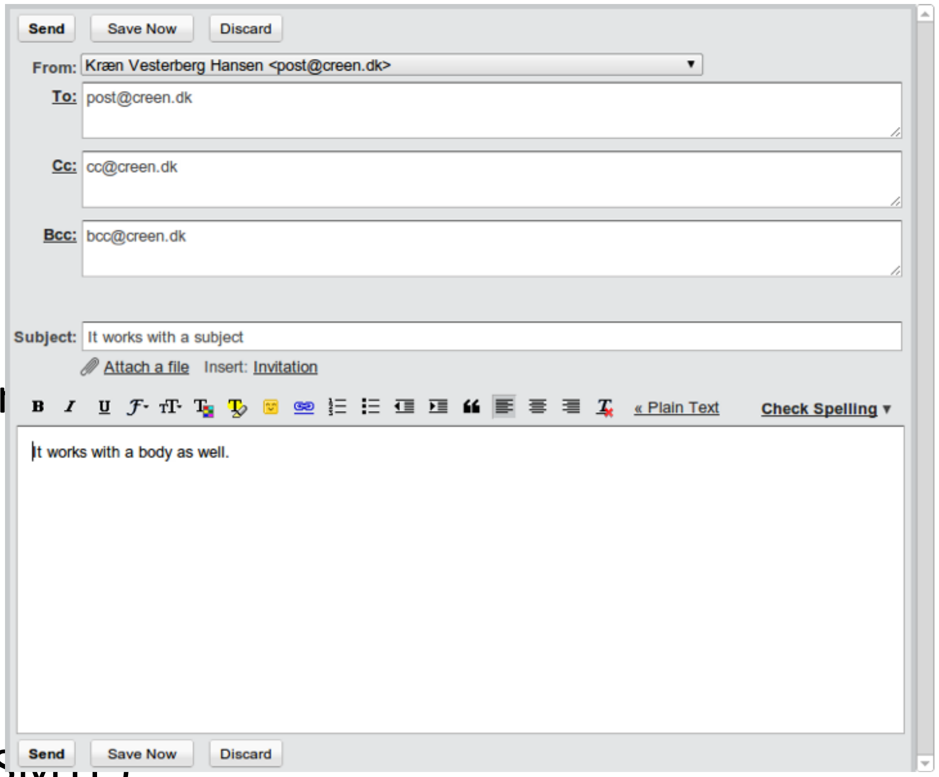
1. Control Connection: Used for sending control information between two hosts.
2. Data Connection: To send a file.

- Control Information like user identification, password, commands to change remote directory, commands to “put” and “get” files
- Client will browses remote file directory, sends commands over control connection.
- FTP server mai e’ r directory, earlier authentication.



Electronic Mail (Email)

- Email is an asynchronous communication medium in which people send and read messages as convenient for them.
- Modern Email has many powerful features like:
  - ✓ A messages with attachments
  - ✓ Hyperlinks
  - ✓ HTML-formatted text
  - ✓ Embedded photos
- Email is fast, easy to distribute, and i
- High level view of Internet mail syste
  1. User agents
  2. Mail servers
  3. Simple Mail Transfer Protocol (S



From: Kræn Vesterberg Hansen <post@creen.dk>
To: post@creen.dk
Cc: cc@creen.dk
Bcc: bcc@creen.dk
Subject: It works with a subject

It works with a body as well.

Email - Cont…

User Agent

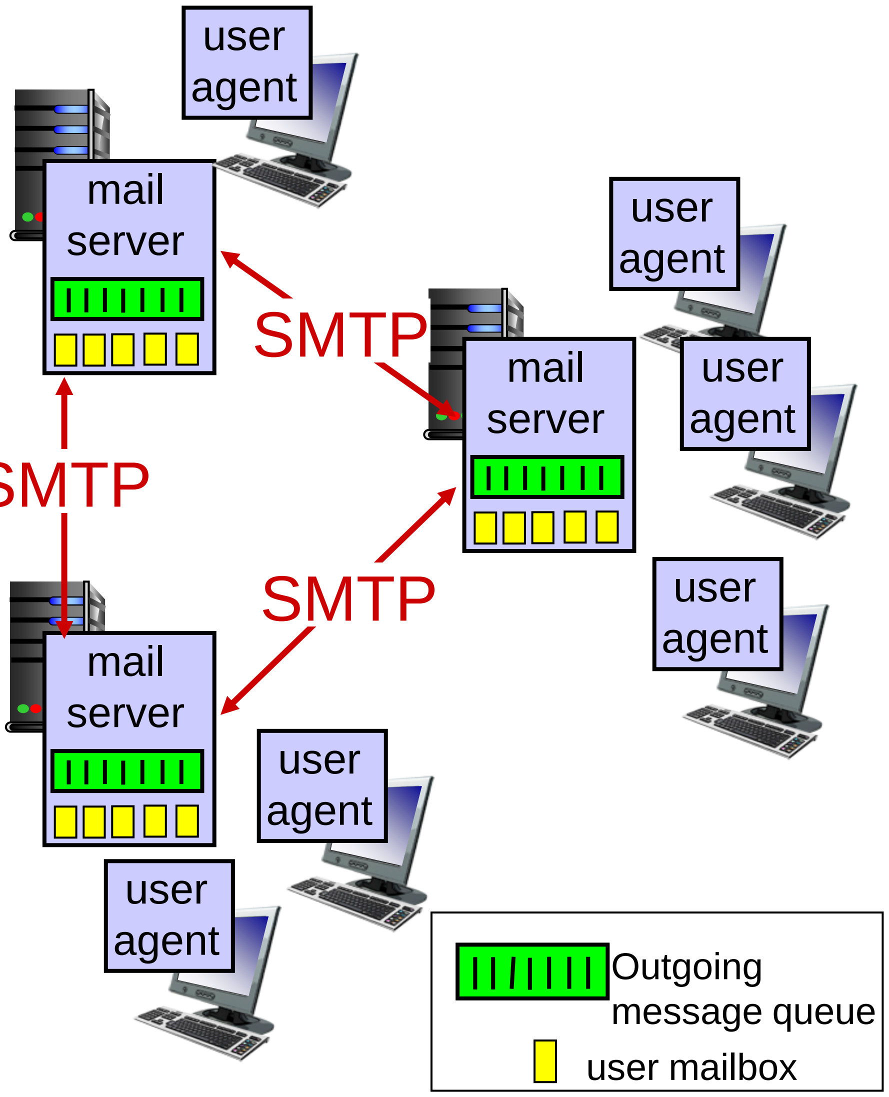
- User agents allow users to read, reply to, forward, save, and compose messages.
- E.g. Microsoft Outlook and Apple Mail.

Mail servers:

- A mailbox contains incoming messages for user.
- A message queue of outgoing (to be sent) mail messages.

SMTP

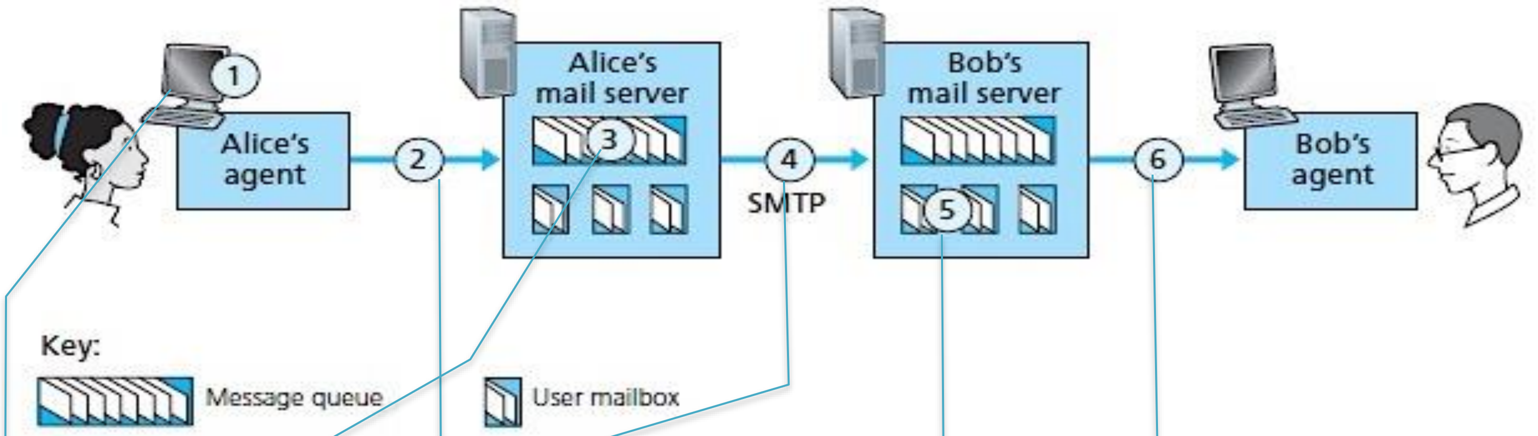
- It is a principal application layer protocol between mail servers to send email messages.
  - ✓ client: sending mail to server
  - ✓ server: receiving mail from other different mail server



SMTP

- Simple Mail Transfer Protocol used in sending and receiving e-mail.
- It use TCP to reliably transfer email message from client to server using port 25.
- It restricts the body (not just the headers) of all mail messages to simple 7-bit ASCII.
- SMTP does not use intermediate mail servers for sending mail.
- If receiving end mail server is down, the message remains in sending end mail server and waits for a new attempt.

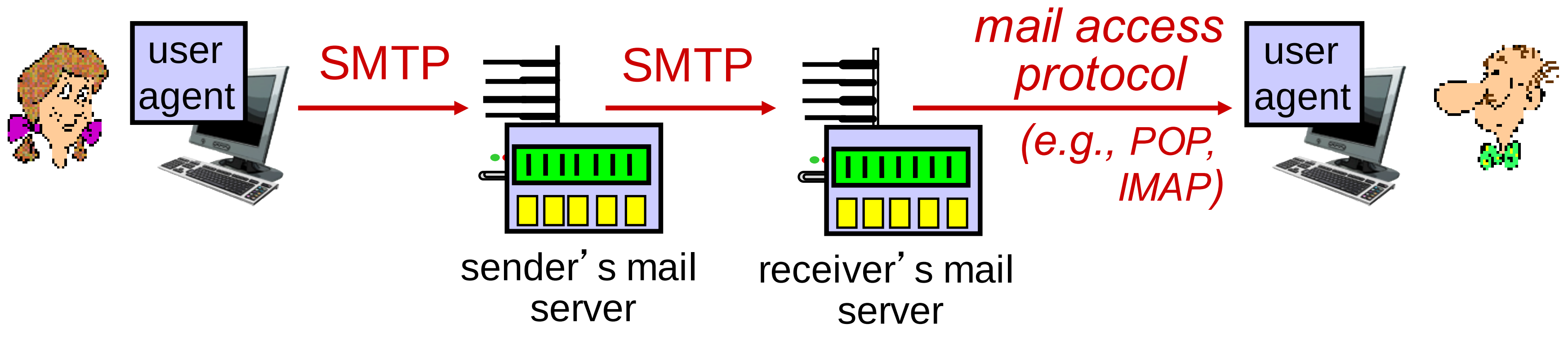
SMTP - Example



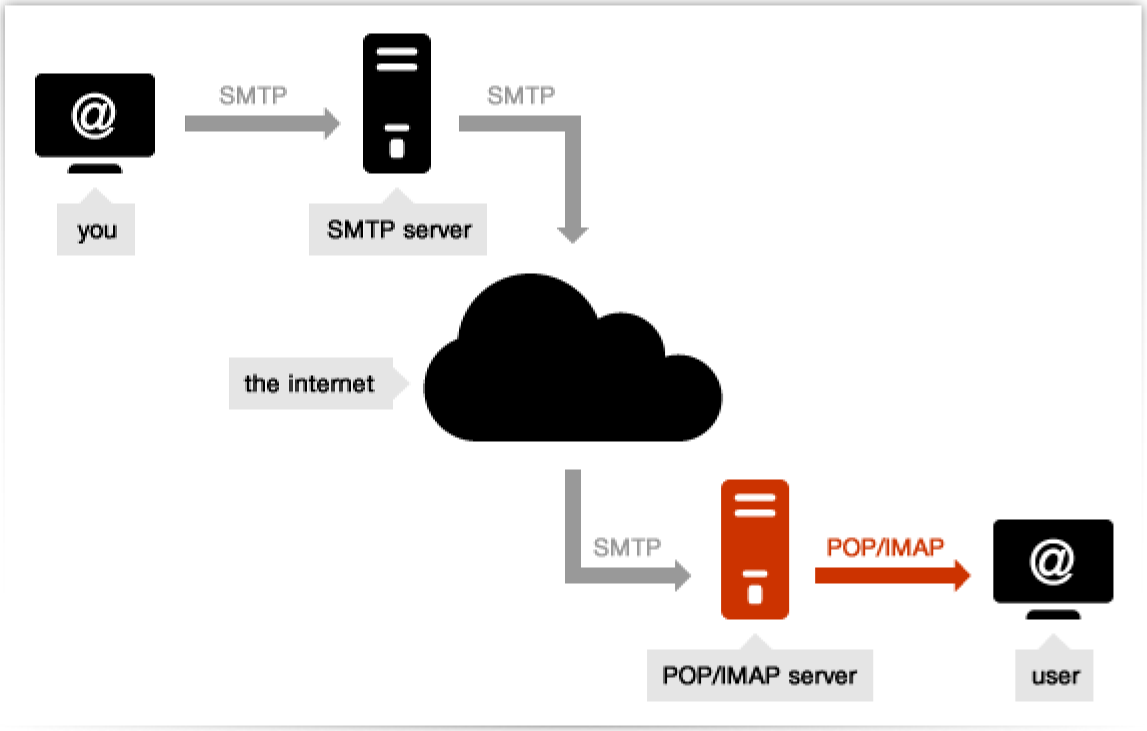
1. Alice uses user agent to compose message to it@mbit.edu.in
2. Alice's user agent sends message to her mail server; message placed in message queue.
3. Client side of SMTP opens TCP connection with Bob's mail server.
4. SMTP client sends Alice's message over the TCP connection.
5. Bob's mail server places the message in Bob's mailbox.
6. Bob invokes his user agent to read message.

6) Bob invokes his user agent to read message

Mail Access Protocols (POP3 and IMAP)

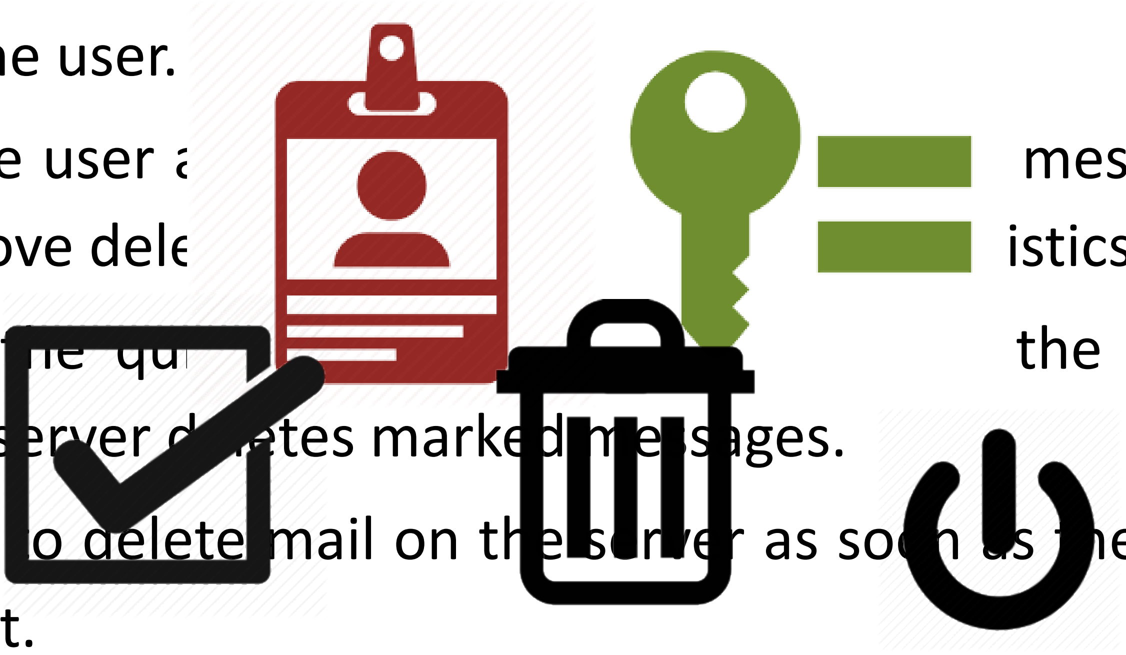


- POP3
  - ✓ Post Office Protocol – Version 3
- IMAP
  - ✓ Internet Mail Access Protocol
- A mail access protocol, such as POP3, is used to transfer mail from the recipient's mail server to the recipient's user agent.


SMTP
SMTP
you
SMTP server
the internet
SMTP
POP/IMAP
POP/IMAP server
user

POP3 – Post Office Version 3

- POP3 is an extremely simple mail access protocol.
- With the TCP connection established, POP3 progresses through three phases: authorization, transaction and update.
- In authorization, the user agent sends a username and a password to authenticate the user.
- In transaction, the user a messages for deletion, remove dele istics.
- In update, after the qu the POP3 session; the mail server deletes marked messages.
- POP3 is designed to delete mail on the server as soon as the user has downloaded it.



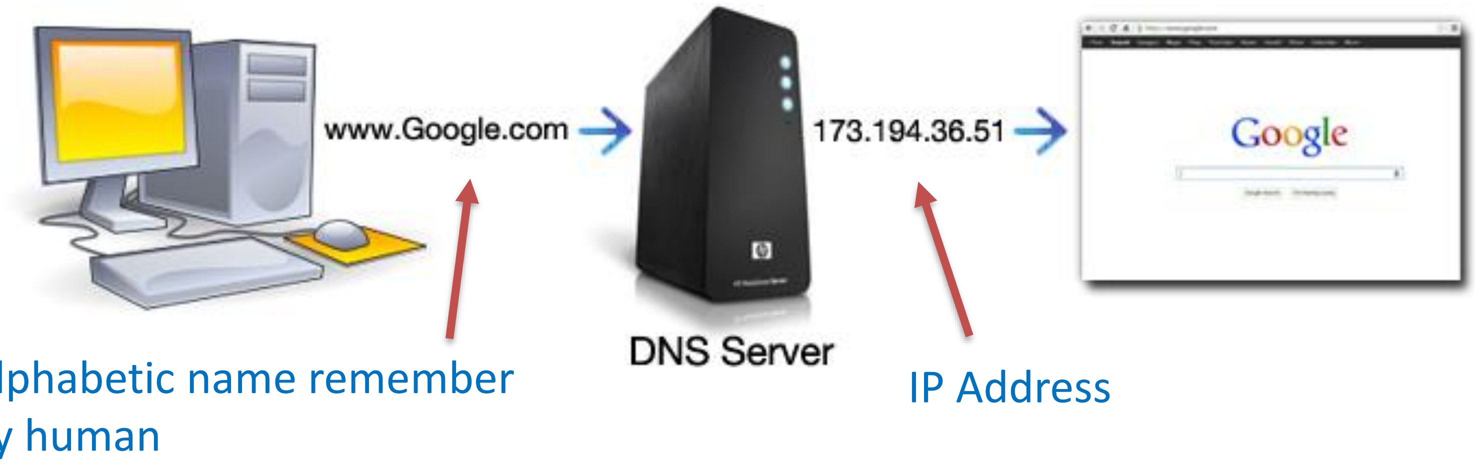
POP3 Vs. IMAP4

- Disadvantages of POP3
  - ✓ Does not allow users to organize mails on server
  - ✓ No separate Folders on server
  - ✓ No Partial checking content of mail before downloading
- Features of IMAP4
  - ✓ User can partially download email
  - ✓ Users can create delete or rename the mail box on server
  - ✓ User can search contents of the email
  - ✓ User can also check email header prior to downloading
  - ✓ User can create folders to organize the emails in the hierarchy

IMAP - Internet Mail Access Protocol

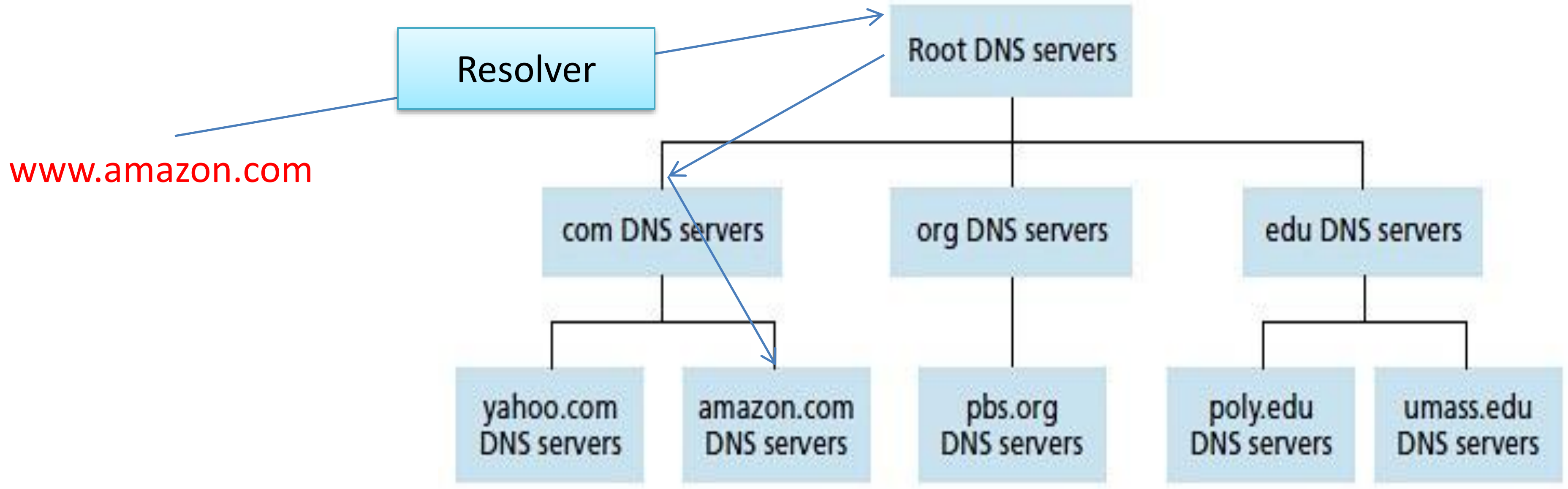
- To keeps all messages in one place: at server
- The recipient can then move and organize the message into a new, user-created folder, read the message, delete the message, move messages from one folder to another and so on.
- To allow users to search remote folders for messages matching specific criteria.
- Also permit a user agent to obtain components of messages, When low-bandwidth connection between the user agent and its mail server.
- In this case, user not to download all of the messages in its mailbox, particularly avoiding long messages like an audio or video clip.

DNS - Domain Name System



- It is an internet service that translates domain names into IP addresses.
- It is application-layer protocol.
- DNS service must translate the domain name into the corresponding IP address.
- In DNS system, If one DNS server doesn't know how to translate a particular domain name, it asks another one, and so on, until the correct IP address is returned.

DNS - Example



- DNS client wants to determine the IP address for the hostname `www.amazon.com`
- The client first contacts one of the root servers, which returns IP addresses for TLD servers - top-level domain .com.
- Then contacts TLD servers, which returns the IP address of an **authoritative server** for `www.amazon.com`
- Finally, contacts one of the authoritative servers for `www.amazon.com`, which returns the IP address for the hostname `www.amazon.com`.

- In order for the user’s host to be able to send an HTTP request message to the Web server www.someschool.edu, the user’s host must first obtain the IP address of www.someschool.edu. This is done as follows.

  1. The same user machine runs the client side of the DNS application.

  2. The browser extracts the hostname, www.someschool.edu, from the URL and

  passes the hostname to the client side of the DNS application.

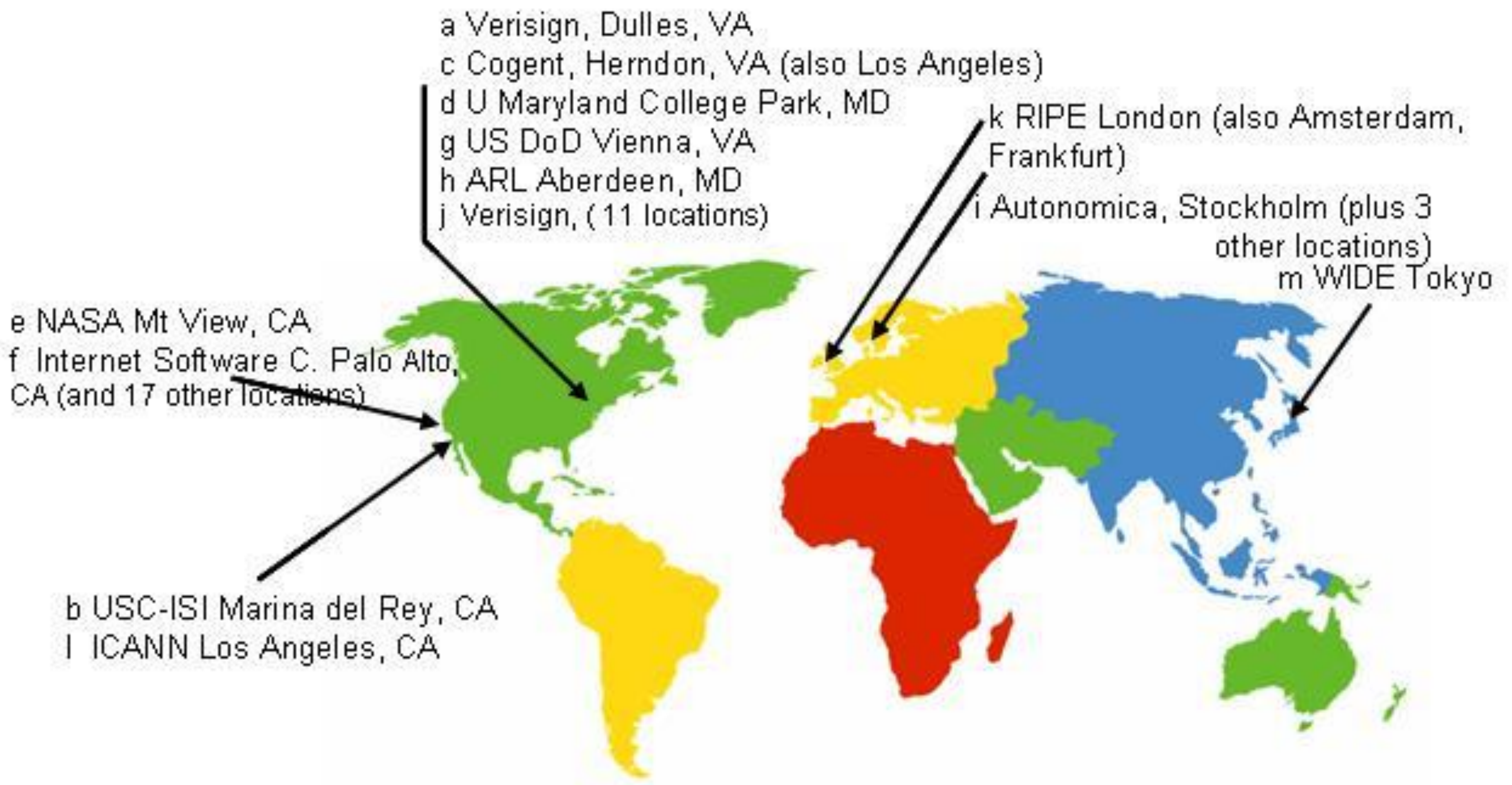
  3. The DNS client sends a query containing the hostname to a DNS server.

  4. The DNS client eventually receives a reply, which includes the IP address for the hostname.

  5. Once the browser receives the IP address from DNS, it can initiate a TCP connection to the HTTP server process located at port 80 at that IP address.

DNS: A distributed - hierarchical database

- Root DNS Servers – Total 13

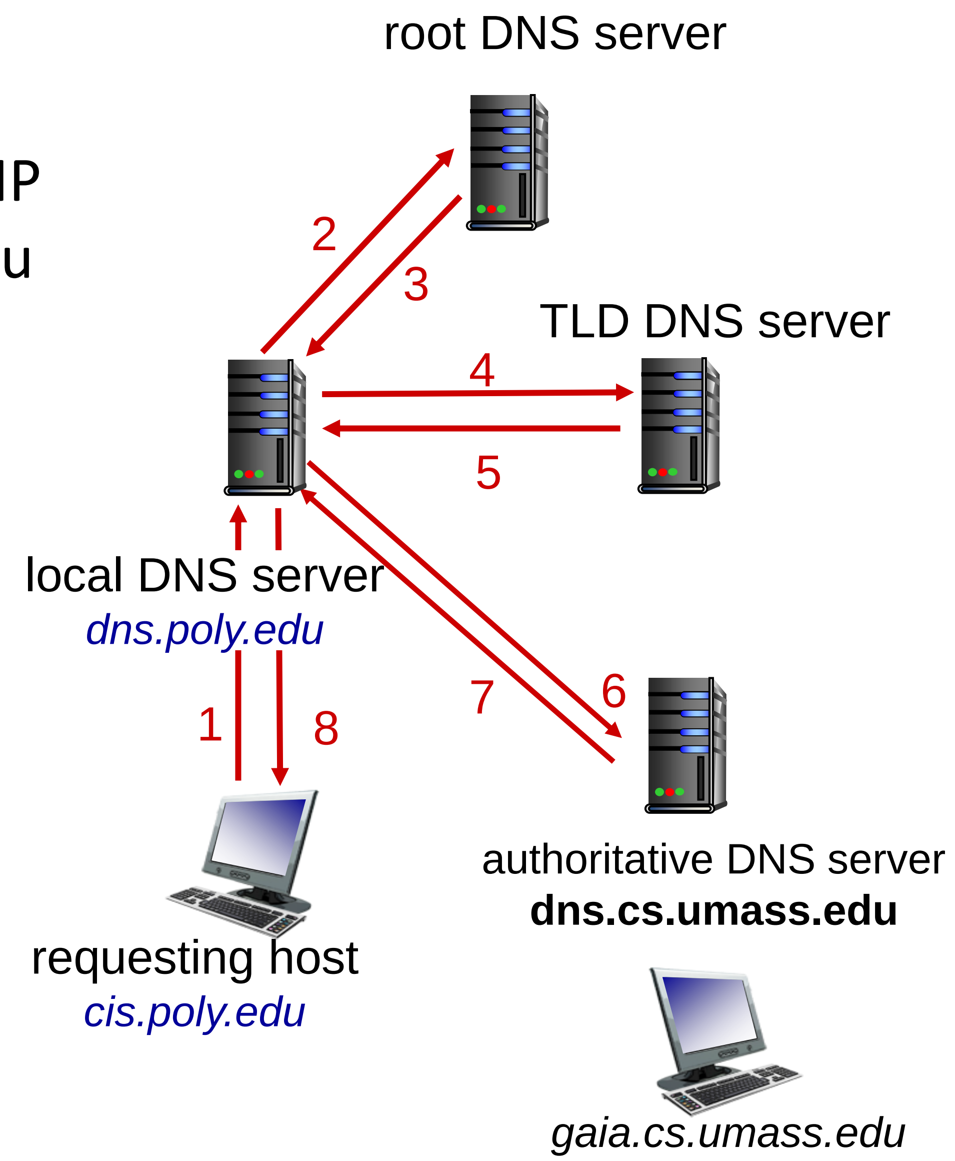

a Verisign, Dulles, VA
c Cogent, Herndon, VA (also Los Angeles)
d U Maryland College Park, MD
g US DoD Vienna, VA
h ARL Aberdeen, MD
j Verisign, (11 locations)
k RIPE London (also Amsterdam, Frankfurt)
i Autonomica, Stockholm (plus 3 other locations)
m WIDE Tokyo
e NASA Mt View, CA
f Internet Software C. Palo Alto, CA (and 17 other locations)
b USC-ISI Marina del Rey, CA
l ICANN Los Angeles, CA

DNS – Cont…

- Top-level domain (TLD) servers:
  - ✓ It is responsible for com, org, net, edu, aero, jobs, museums, and all top-level country domains, e.g.: uk, fr, ca, jp
  - ✓ Network Solutions maintains servers for .com TLD
  - ✓ Education for .edu TLD
- Authoritative DNS servers:
  - ✓ To organization's own DNS servers, providing authoritative hostname to IP mappings for organization's named hosts.
  - ✓ It can be maintained by organization or service provider.
- Local DNS name servers:
  - ✓ It does not strictly belong to hierarchy
  - ✓ when host makes DNS query, query is sent to its local DNS server.
    - It acts as proxy, forwards query into hierarchy.

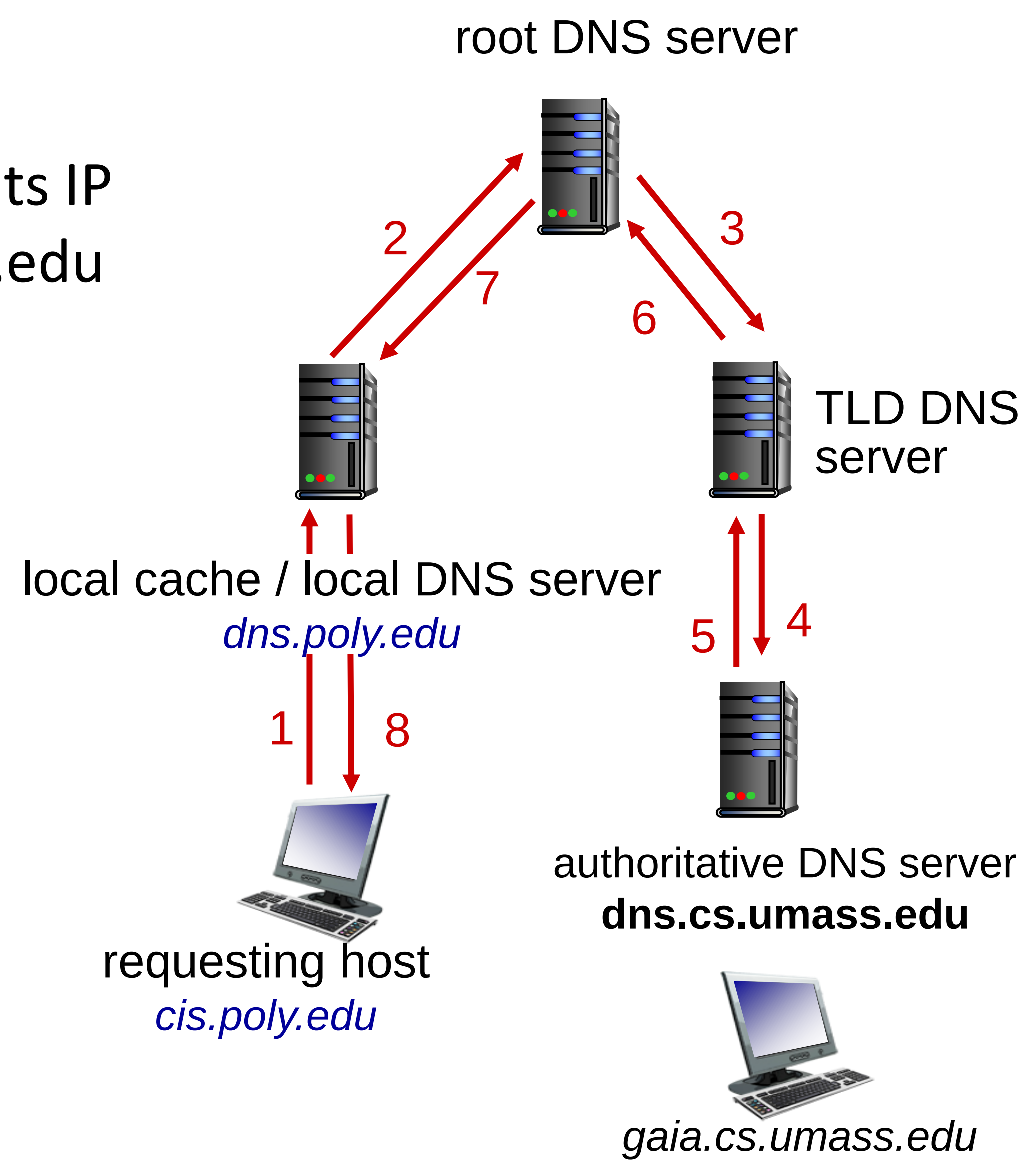
DNS name resolution

- Iterated query:
- A host at cis.poly.edu wants IP address for gaia.cs.umass.edu



DNS name resolution

- Recursive query:
- A host at cis.poly.edu wants IP address for gaia.cs.umass.edu

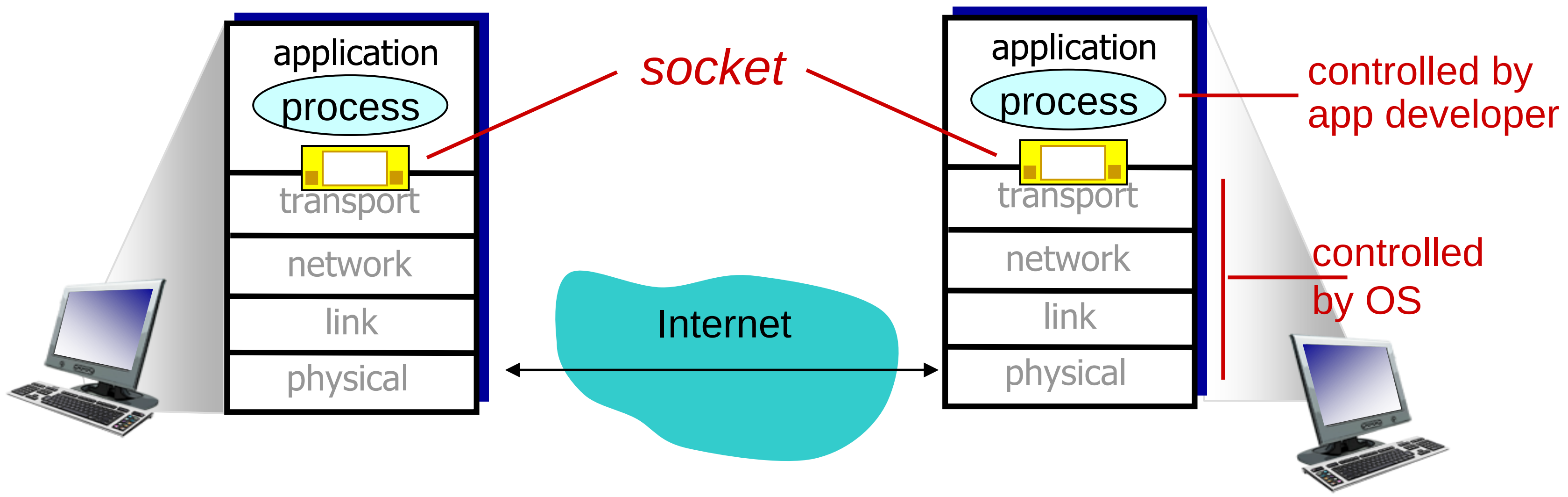


DNS – Cont…

- Distributed database design is more preferred over centralized design to implement DNS in the Internet.
- **A single point of failure:** If the DNS server crashes then the entire Internet will not stop.
- **Traffic volume:** With millions of device and users accessing its services from whole globe at the same time.
- A Single DNS Server cannot handle huge DNS traffic but with distributed system its distributed and reduce overload on server.
- **Distant centralized database:** A single DNS server cannot be "close to" all the querying clients.
- If it is in New York City, then all queries from Australia must travel to the other side of the globe, perhaps over slow and congested links cause significant delays.
- **Maintenance:** To keep records for all Internet hosts. it would have to be updated frequently to account for every new host.

Socket Programming

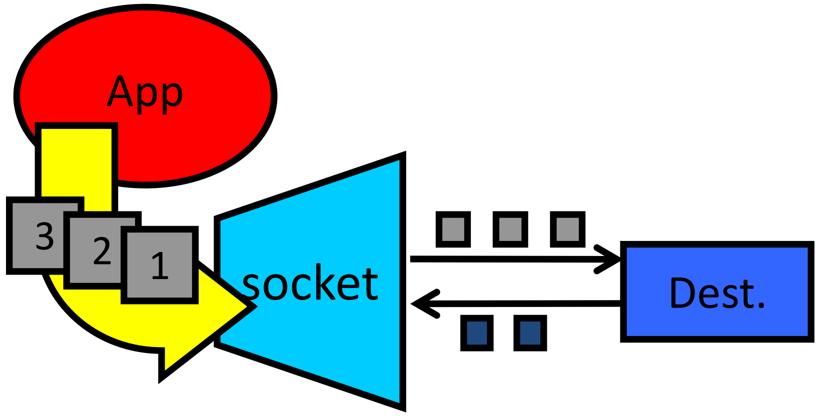
- Socket is interface between application and network.
  - ✓ An application creates a socket.
  - ✓ Two type of socket:
    1. TCP Socket – Reliable Transmission
    2. UDP Socket – Unreliable Transmission
- Once configured the application can pass data to the socket for transmission and receive data from the socket (transmitted through the network by some other host).

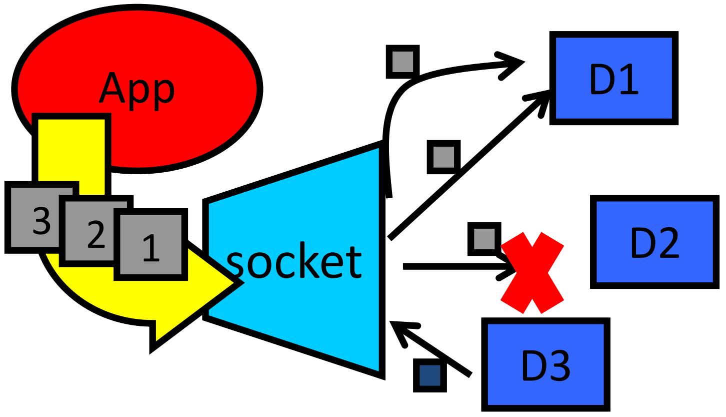


Type of Socket

- SOCK_STREAM
  - ✓ E.g. TCP
  - ✓ Reliable delivery
  - ✓ In-order guaranteed
  - ✓ Connection-oriented
  - ✓ Bidirectional

- SOCK_DGRAM
  - ✓ E.g. UDP
  - ✓ Unreliable delivery
  - ✓ No order guarantees
  - ✓ Connection-less
  - ✓ Unidirectional





Outline - Summary

- Principles of Computer Applications
  - ✓ Browser, Web Server, Email, P2P Applications etc…
- Application Layer (TCP – UDP Services)
- Web (Web Pages – Objects like html, jpeg, mp3, etc…)
- HTTP (TCP connection, port-80, persistent & non-persistent conn.), Request & Response Message format, Cookies, Web caches, FTP, Port-21
- E-mail (User agent, Mail Server, SMTP port - 25), POP3, IMAP
- DNS (Domain names to IP Address), hierarchy structure
- Socket programming with TCP and UDP (TCP – Sock_Stream, UDP – Sock_DGram)

THANK YOU