Application Layer

Bijoy Ch. Chatterjee

South Asian University, New Delhi,
India bijoycc@sau.int,
bijoycc@ieee.org

Overview

1. Network Application Architecture
2. HyperText Transfer Protocol (HTTP)
3. File Transfer and Email protocols
4. Domain Name Service
5. Peer-to-Peer Applications

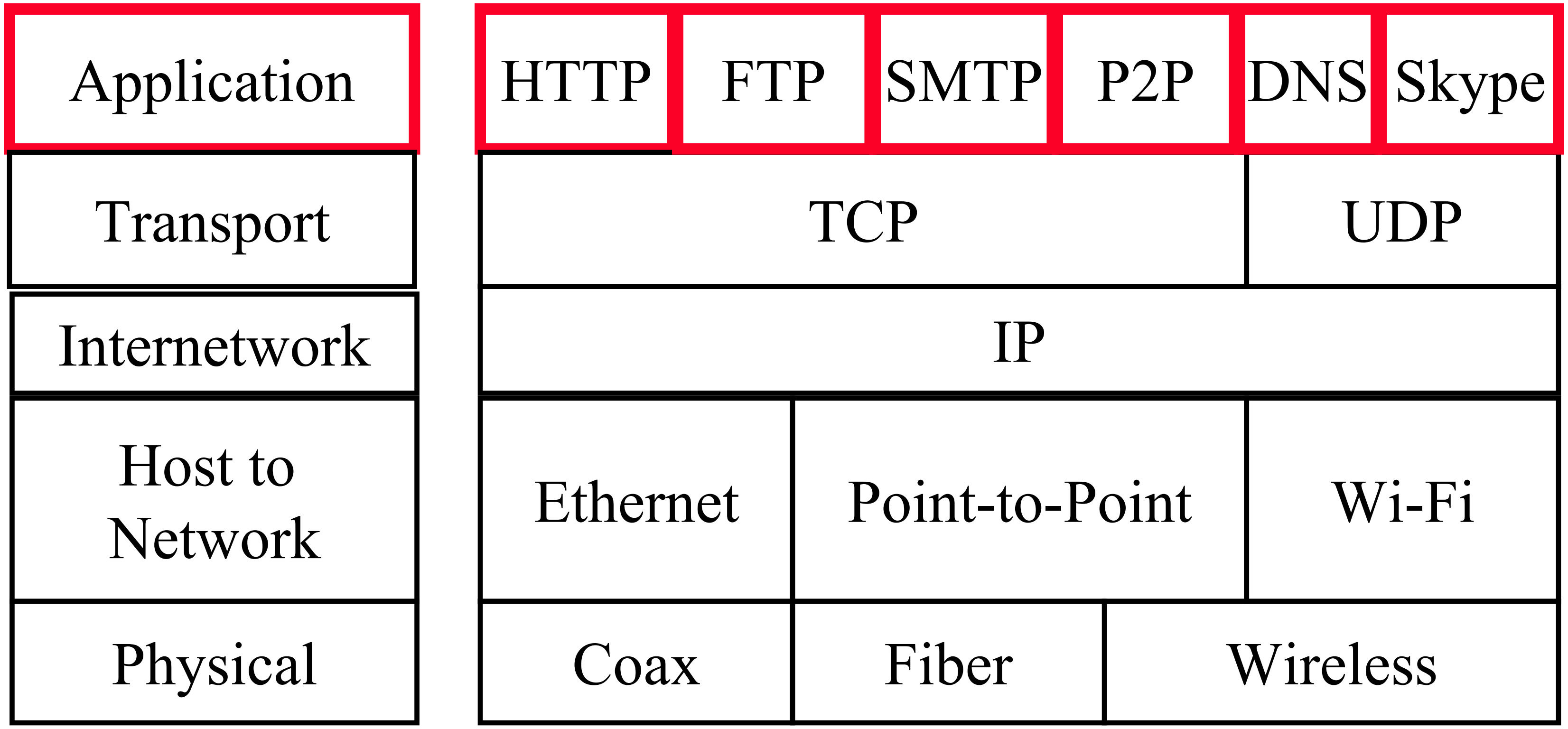
Note: This class lecture is based on Chapter 2 of the textbook (Kurose and Ross) and the figures provided by the authors.

Network Application Architectures

1. Protocol Layers
2. Client-Server vs. Peer-to-Peer
3. Process Communication
4. Names, Addresses, Ports
5. Transports

Protocol Layers

- Top-Down approach



Application

Transport

Internetwork

Host to Network

Physical

HTTP

FTP

SMTP

P2P

DNS

Skype

TCP

UDP

IP

Ethernet

Point-to-Point

Wi-Fi

Coax

Fiber

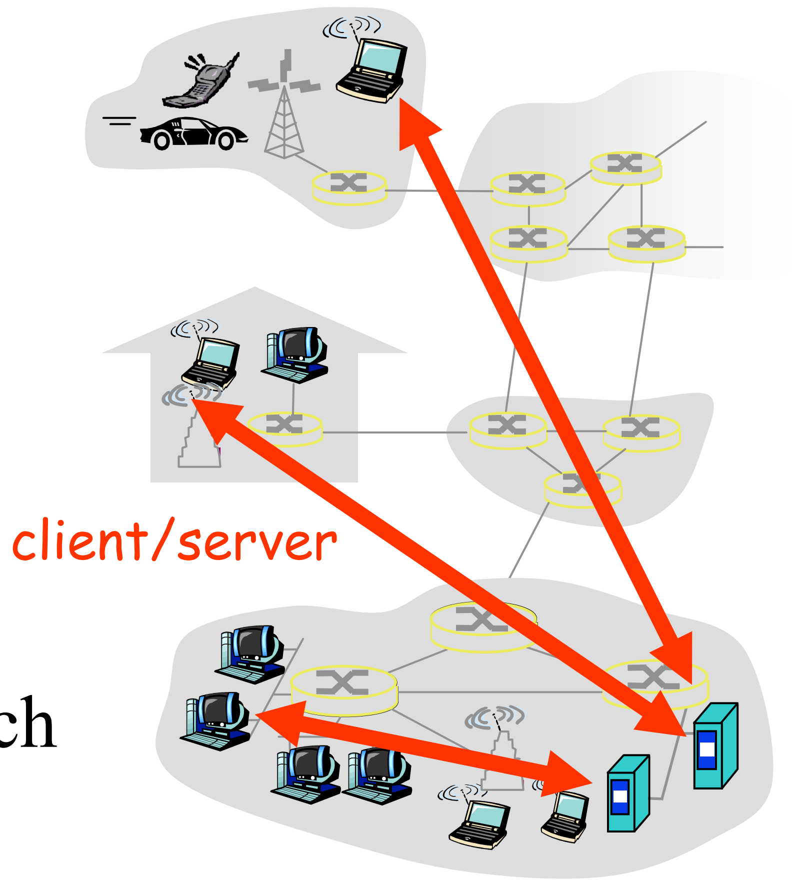
Wireless

Network Application Architectures

- Client-Server
- Peer-to-Peer

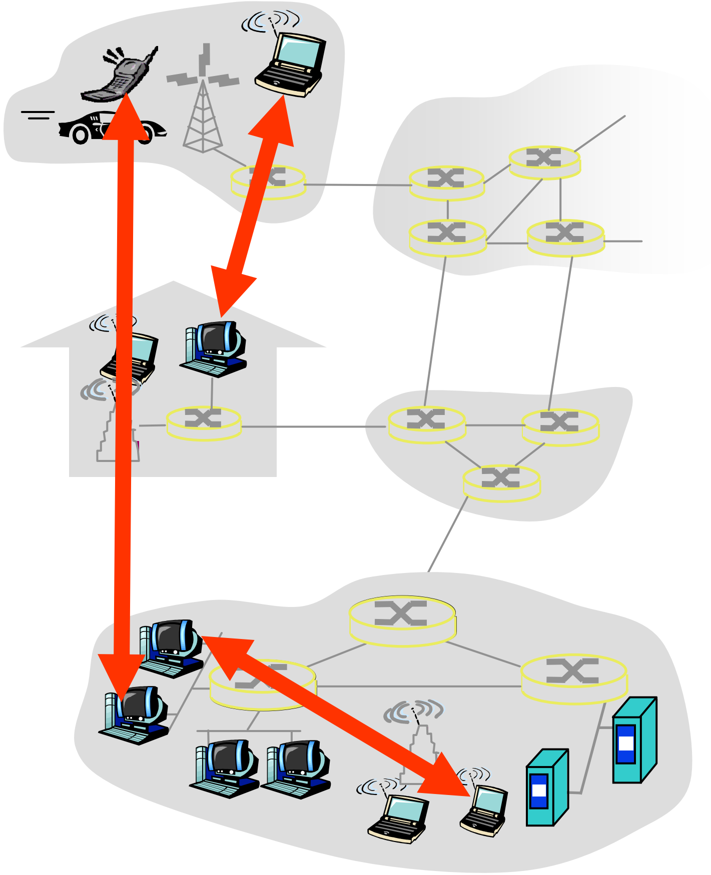
Client-Server

- Clients: Request service
- Server: Provides a service. Waits for clients
- Server is always up
- Clients do not communicate directly with each other
- Server = Data Center
- Example: Web Server, Search Engine, Social Networking



Peer-to-Peer

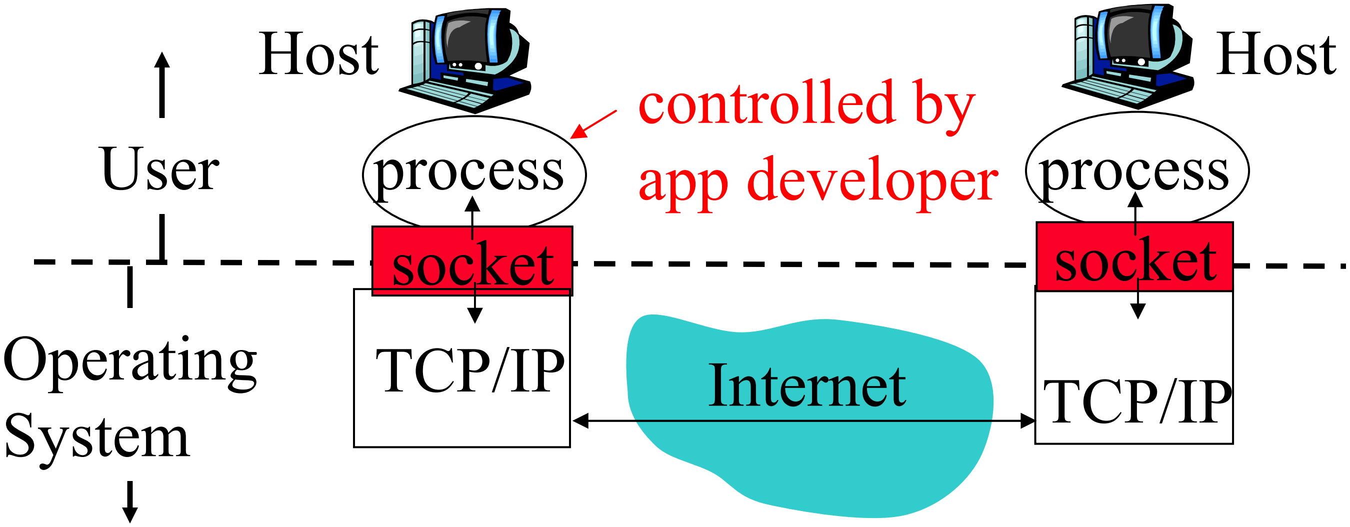
- Does not require always-on servers
- Hosts communicate directly
  ⇒ Peers
- Hosts may come on or may go off at any time
- Examples: File Sharing (Bit Torrent, eMule, LimeWire), Telephony (Skype)
- Highly scalable
- Highly symmetric traffic
  ⇒ ISP unfriendly
- Difficult to authenticate ⇒ Insecure
- Need incentives to share



Process Communications

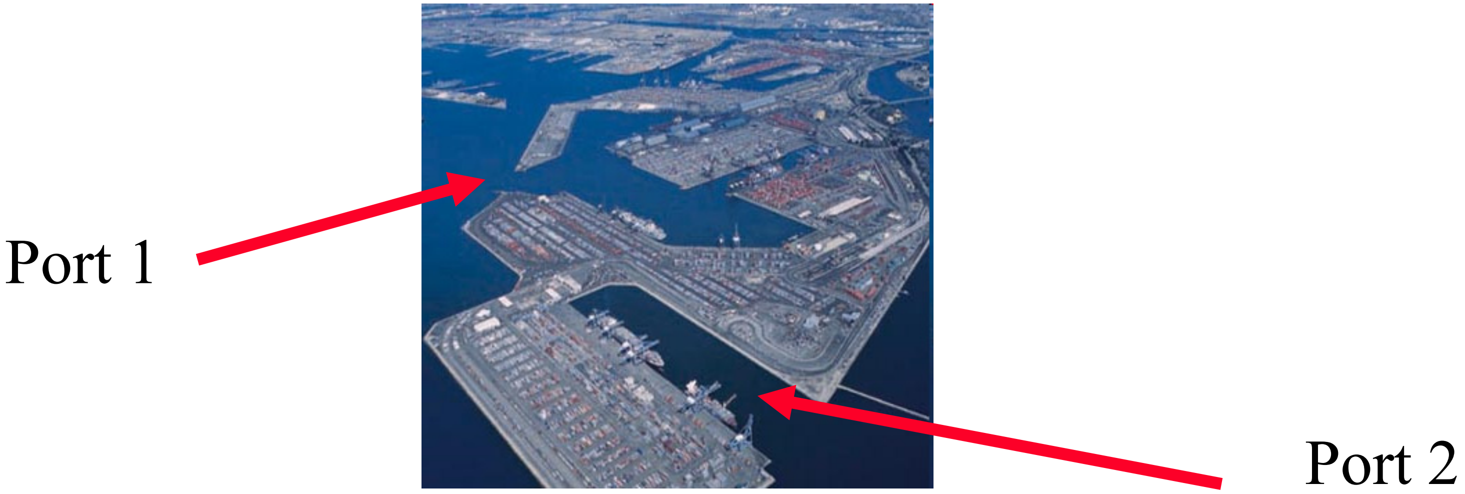
- Inter-Process Communication on the Same Host
  ⇒ Operating system provides message passing
- Unix provides application programming interface called "sockets".
- Inter-Process Communication on Different Hosts
  ⇒ Network provides message passing

A process sends messages into, and receives messages from, the network through a software interface called a **socket**.



Names, Addresses, Ports

- Domain Name System: www.google.com
- IP Address: 209.85.225.147
- 4 decimal numbers less than 256=8 bits each
  ⇒ 32-bits
- Ports: Entry point (Transport service access points)
- 21=FTP, 80=HTTP



Port 1

Port 2

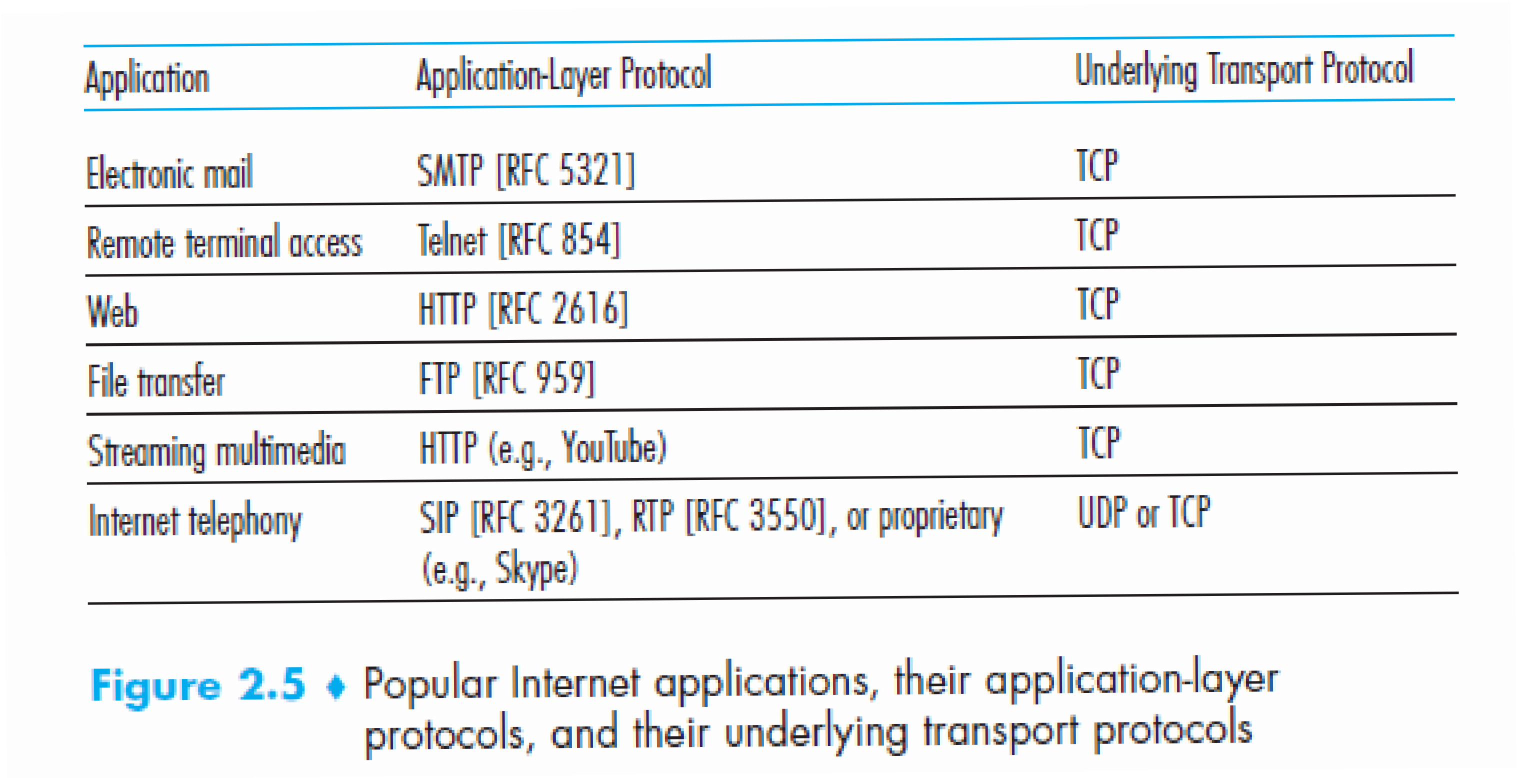
Transports

TCP	UDP
Reliable data transfer	Unreliable Data Transfer
Packet Sequence # required	Sequence # optional
Every packet is acked	Not Acked
Lost packets are retransmitted	No Retransmission
May cause long delay	Quick and Lossy
Connection-oriented service	Connection-less Service
Good for Reliable and delay-insenstive applications	Good for loss-tolerant and delay sensitive applications
Applications: email, http, ftp, Remote terminal access	Telephony, Streaming Multimedia

Application Layer Protocols

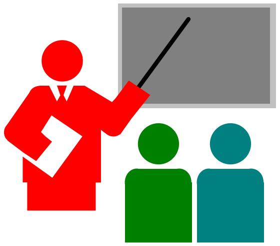
- HTTP: HyperText Transfer Protocol
- FTP: File Transfer Protocol
- SMTP: Simple Mail Transfer Protocol
- DNS: Domain Name Server
  (Control Plane Application)
- P2P: Peer-to-Peer Applications (Class of applications)
- Skype
- Each application has its own protocol, message format, semantics of fields

Application Layer Protocols



Application	Application-Layer Protocol	Underlying Transport Protocol
Electronic mail	SMTP [RFC 5321]	TCP
Remote terminal access	Telnet [RFC 854]	TCP
Web	HTTP [RFC 2616]	TCP
File transfer	FTP [RFC 959]	TCP
Streaming multimedia	HTTP (e.g., YouTube)	TCP
Internet telephony	SIP [RFC 3261], RTP [RFC 3550], or proprietary (e.g., Skype)	UDP or TCP

Figure 2.5 ♦ Popular Internet applications, their application-layer protocols, and their underlying transport protocols



Application Arch: Summary

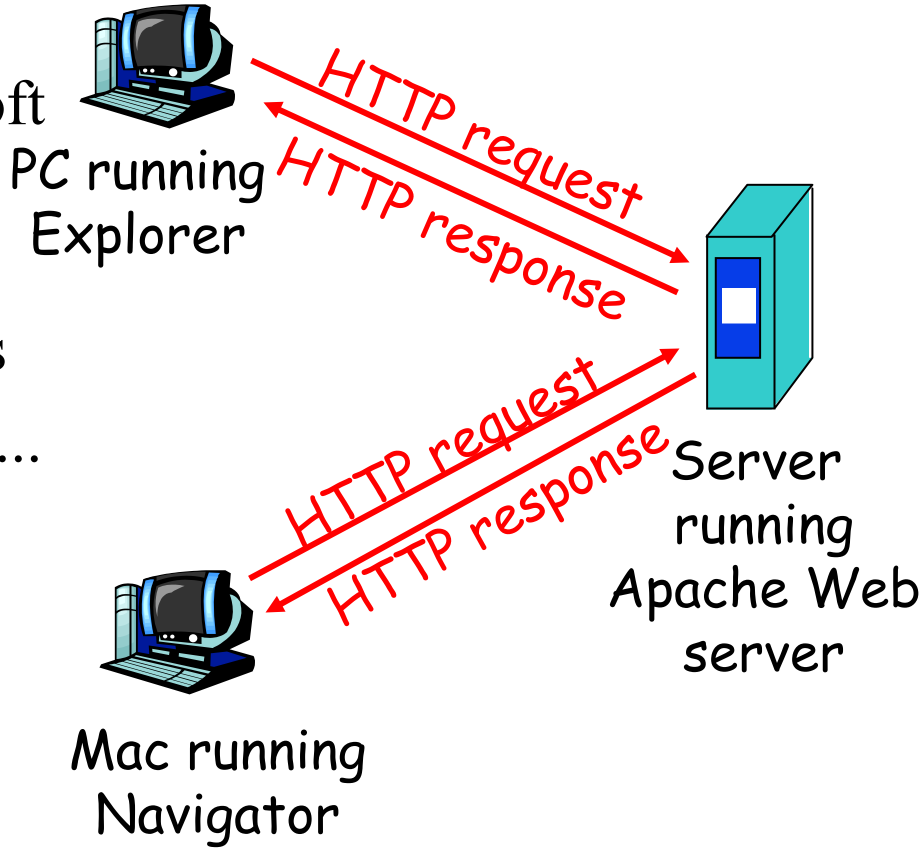
1. P2P applications are more scalable than client-server
2. Applications exchanges messages using operating system sockets
3. Applications communicate using host names, addresses, and ports
4. Applications use transports: TCP, UDP, …
5. TCP is used for reliable communication
   UDP for loss-tolerant delay-sensitive applications

HTTP

1. Concepts
2. Sample Web Page
3. HTTP Messages
4. Cookies
5. Proxy Servers
6. Conditional GET

HTTP Concepts

- **Client**=Browser, e.g., Internet Explorer, Firefox
- **HTTP Server**, e.g., Microsoft Internet Information Service (IIS), Apache
- **Web Page**=Group of objects
- **Object**=Text, Images, files, ...
  - **URL**: Uniform Resource Locator http://www.sau.int/faculty/faculty-profile.html?staff_id=84

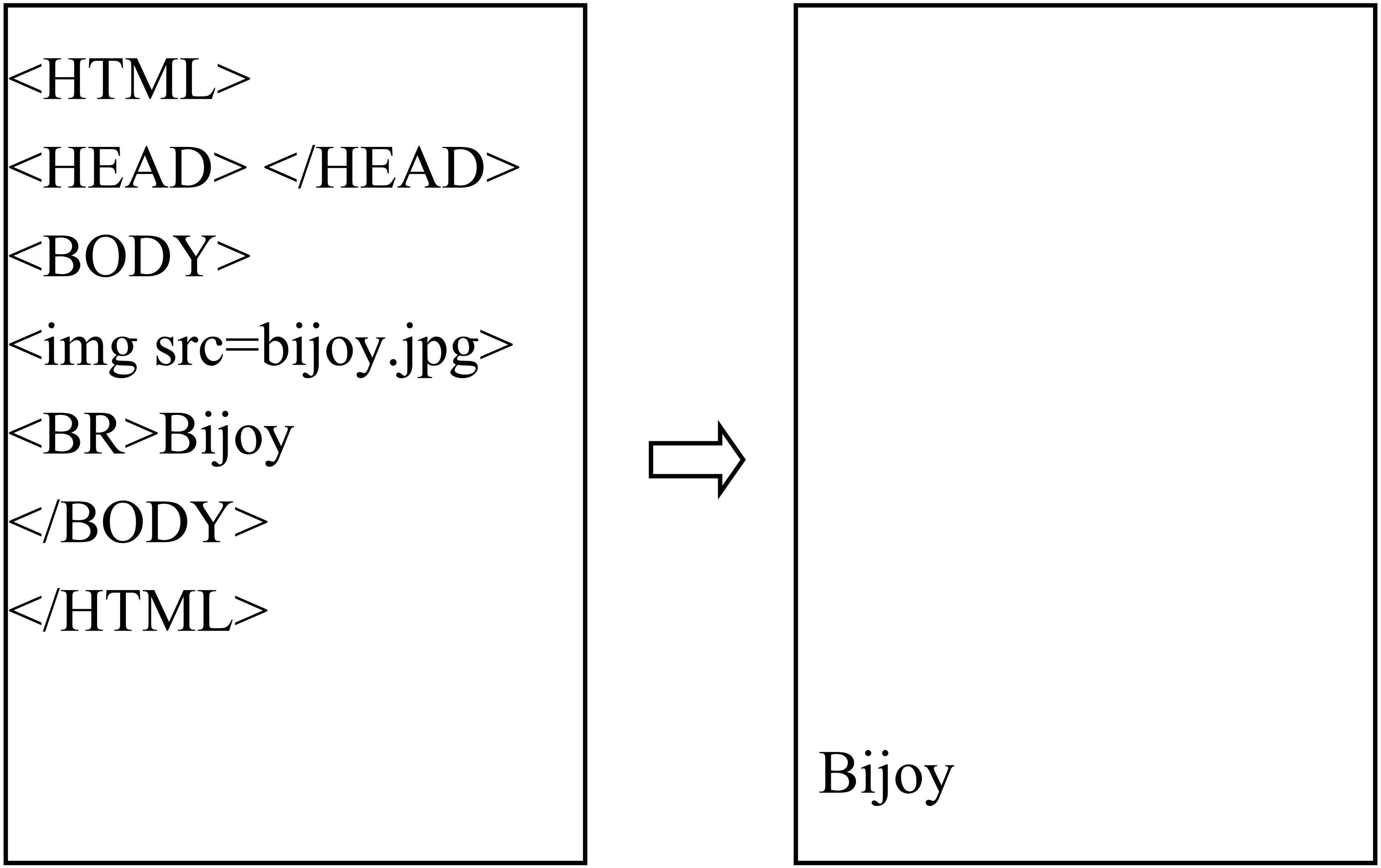


HTTP

- Uses TCP
- **Stateless**: Server does not remember previous history
- **Non-Persistent**: Open new TCP connection, get one object, close
- **Persistent**: Open one TCP connection, get all objects, close

  Server leaves the connection open after sending an object and closes on timeout
- Web pages are written in HyperText Markup Language (**HTML**)

Sample Web Page



```
<HTML>
<HEAD> </HEAD>
<BODY>
<img src=bijoy.jpg>
<BR>Bijoy
</BODY>
</HTML>
```

Bijoy

Sample HTTP Request Message

GET /~bijoycc/*cs203/sample.htm HTTP/1.1*

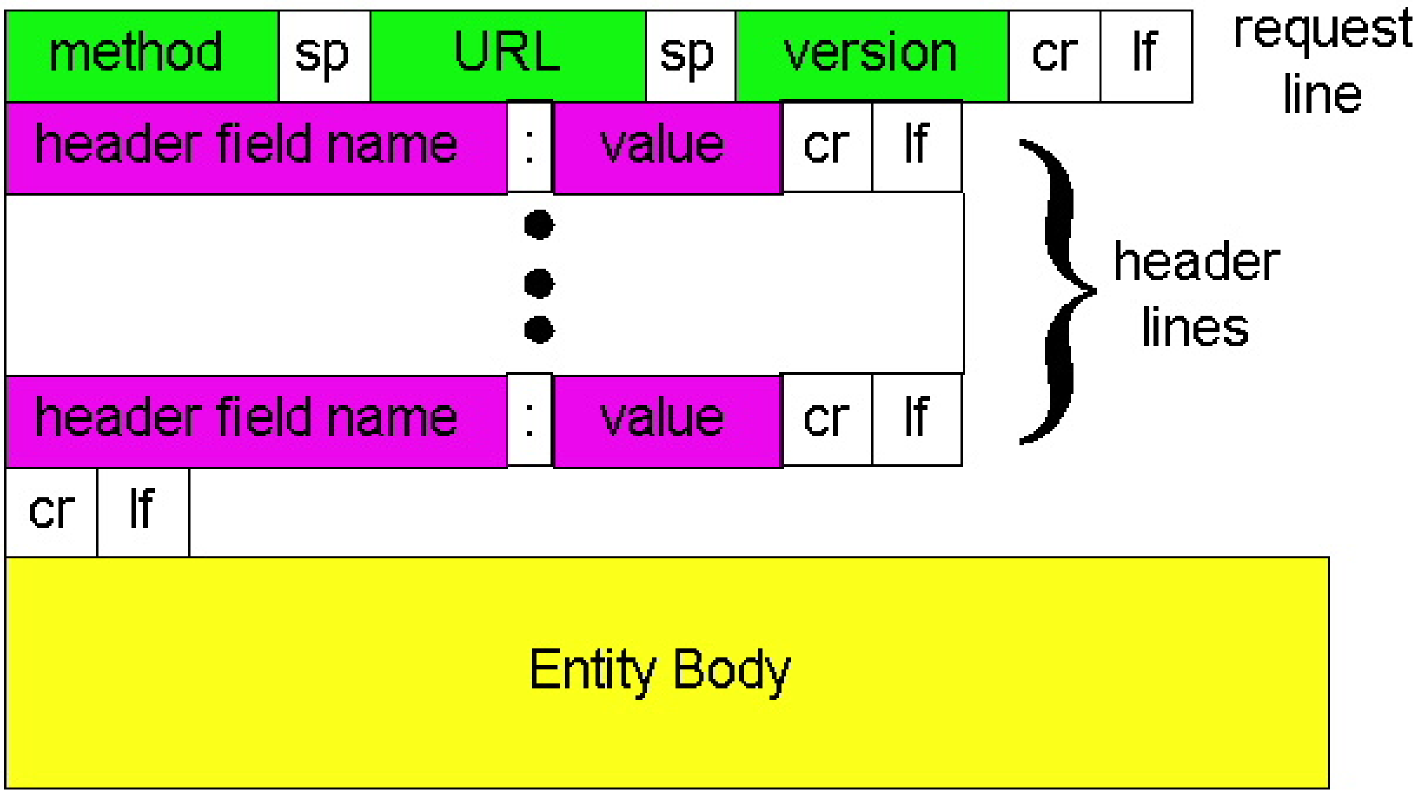
Host: www.sau.int

Connection: close

User-agent: Mozilla/4.0 Accept-Language: en

- **Method** = Get
- **URL** = /bijoycc/*cs203/sample.htm*
- **Version** = HTTP/1.1
- **Header Fields** = Host, Connection, User-agent, …

HTTP Request Message Format



method sp URL sp version cr lf

request line

header field name : value cr lf

header field name : value cr lf

header lines

cr lf

Entity Body

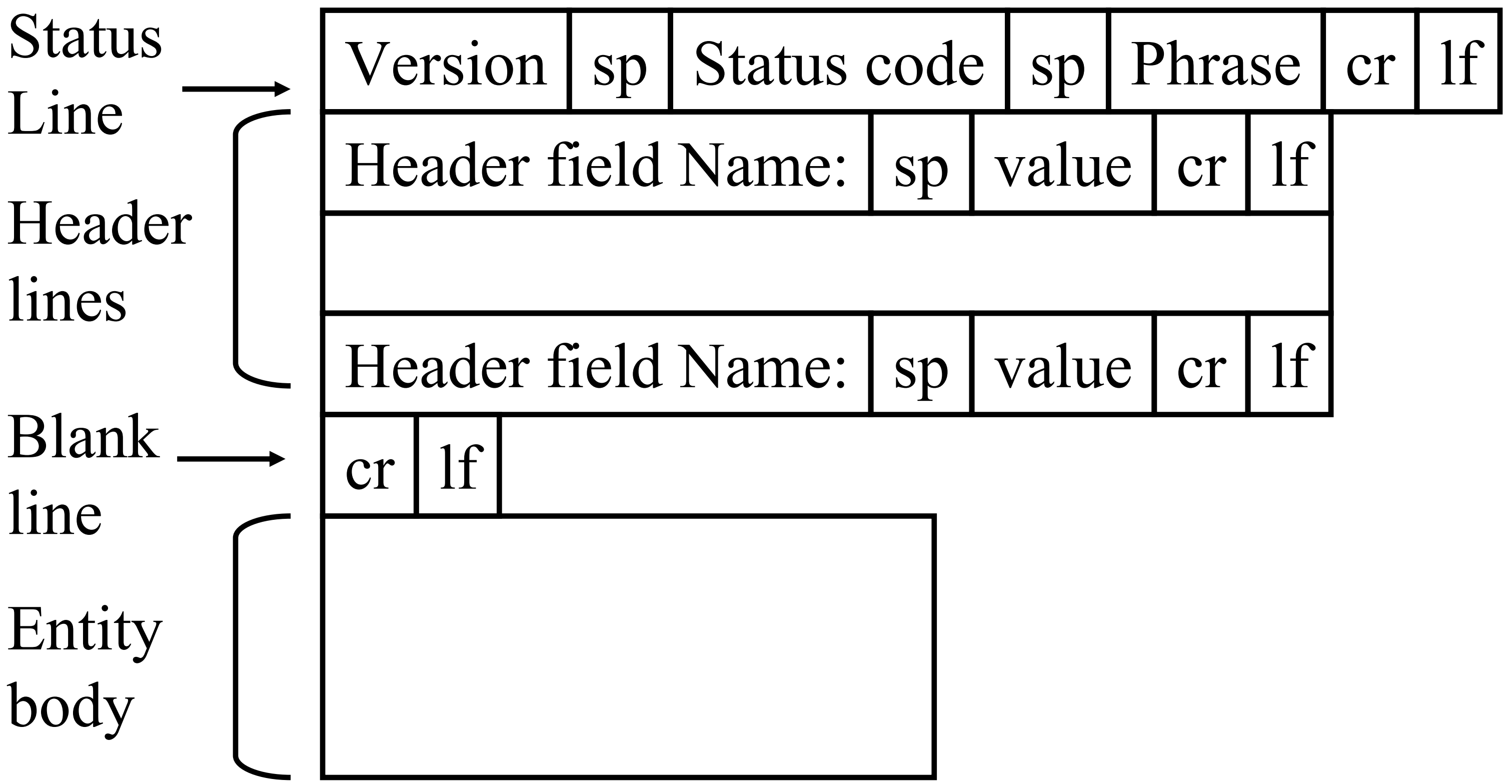
Sample HTTP Response Message

```
HTTP/1.1 200 OK
Connection: close
Date: Tue, 09 Sept 2009 13:00:15 GMT
Server: Apache/1.3.0 (Unix)
Last-Modified: Sun, 6 May 2009 09:23:24 GMT
Content-Length: 6500
Content-Type: Text/html

Data…
```

Status Codes:

- ❑ 200 OK
- ❑ 301 Moved Permanently
- ❑ 400 Bad Request
- ❑ 404 Not Found
- ❑ 505 HTTP Version Not Supported

HTTP Response Message Format



Status Line

Version | sp | Status code | sp | Phrase | cr | lf

Header lines

Header field Name: | sp | value | cr | lf

Header field Name: | sp | value | cr | lf

Blank line

cr | lf

Entity body

GET and POST requests

GET - Requests data from a specified resource
POST - Submits data to be processed to a specified resource

Some other notes on GET requests:

- GET requests can be cached
- GET requests remain in the browser history
- GET requests can be bookmarked
- GET requests should never be used when dealing with sensitive data

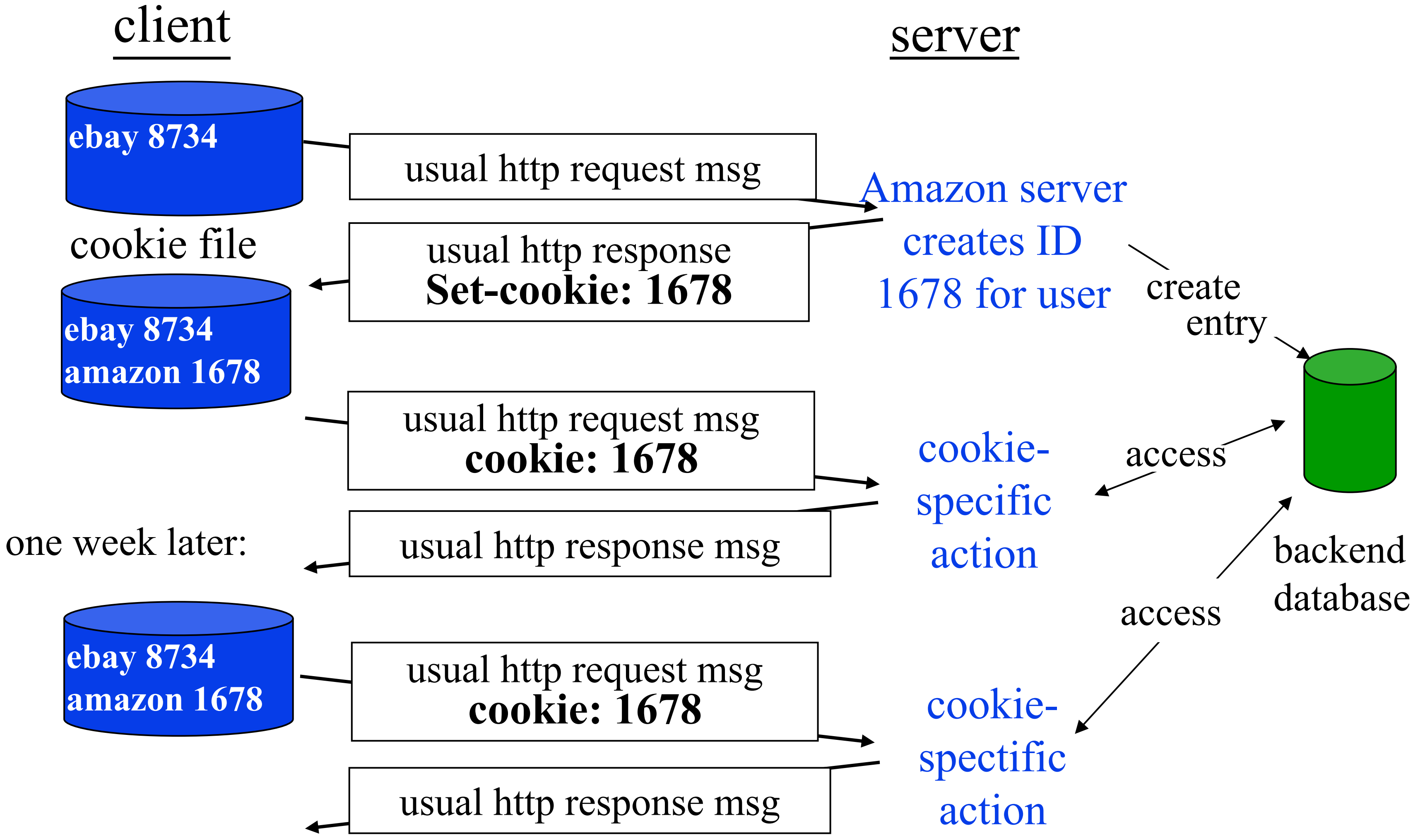
- GET requests have length restrictions
- GET requests should be used only to retrieve data

Some other notes on POST requests:

- POST requests are never cached
- POST requests do not remain in the browser history
- POST requests cannot be bookmarked
- POST requests have no restrictions on data length

Cookies

- Allow servers to remember previous information
  C: \Documents and Settings\bijoy\Cookies\



client

server

ebay 8734

usual http request msg

Amazon server creates ID 1678 for user

create entry

usual http response
Set-cookie: 1678

cookie file

ebay 8734
amazon 1678

usual http request msg
cookie: 1678

cookie-specific action

access

usual http response msg

backend database

one week later:

access

ebay 8734
amazon 1678

usual http request msg
cookie: 1678

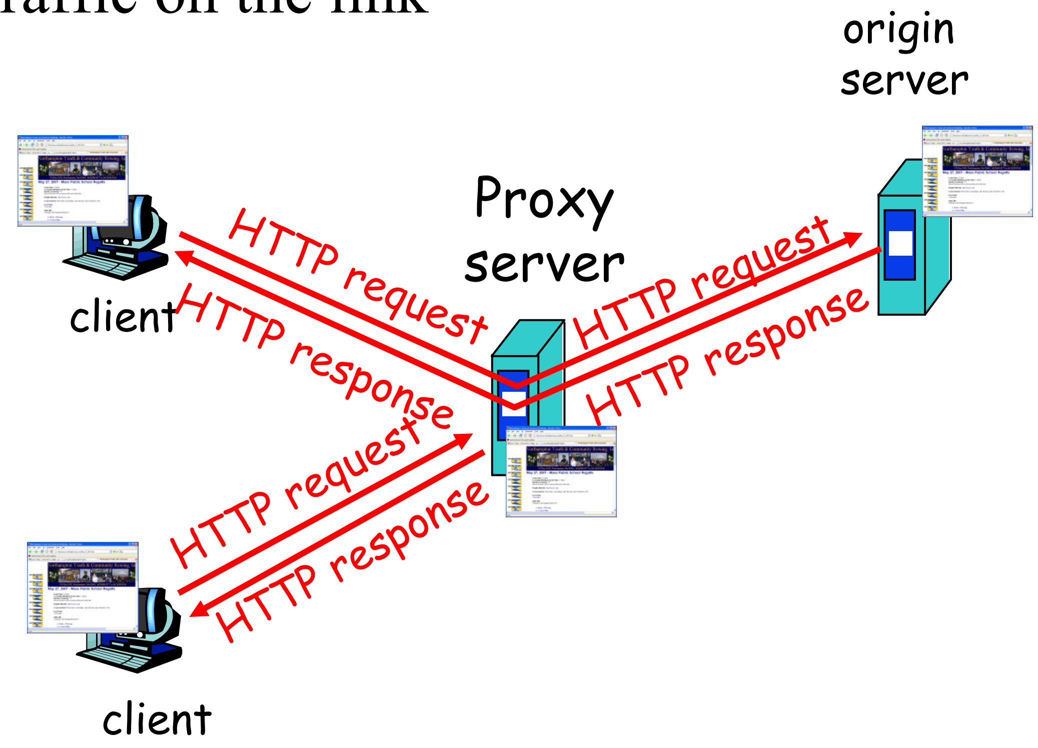
cookie-spectific action

usual http response msg

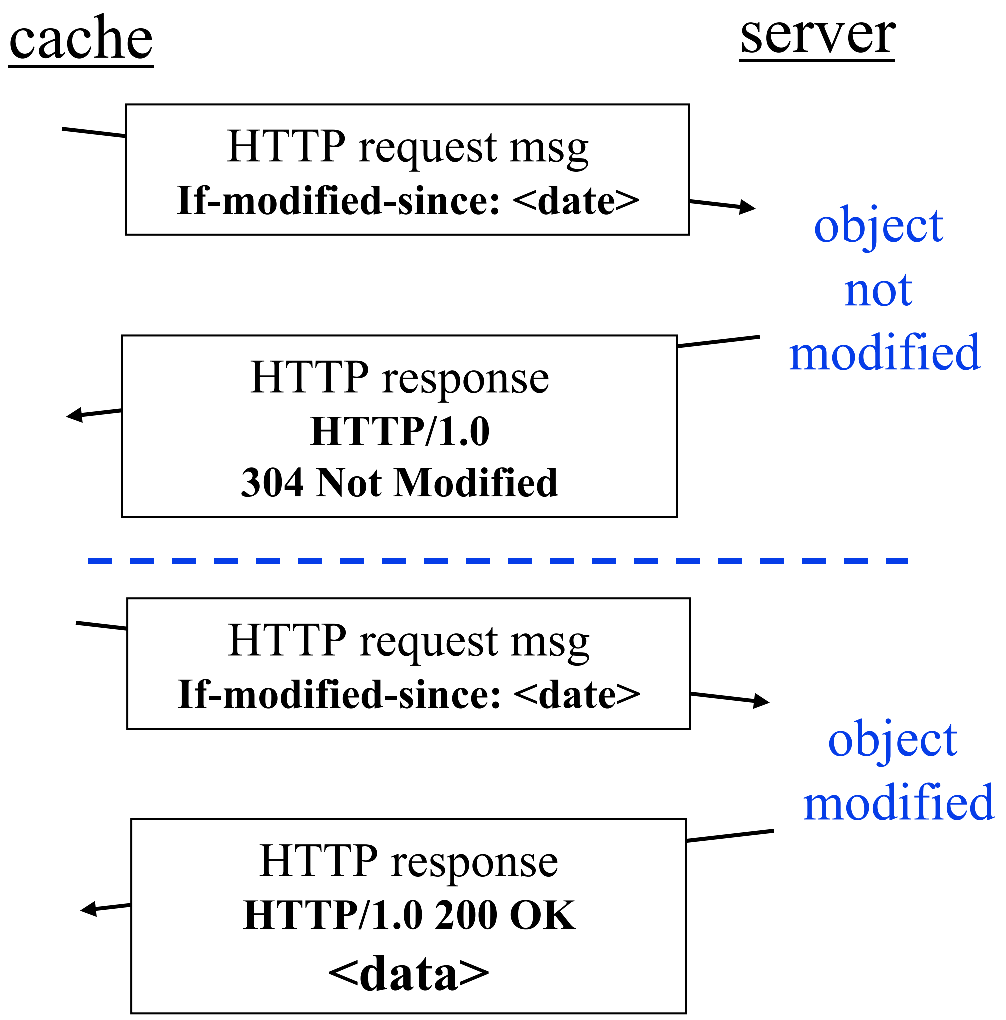
Proxy Server: Web Caching

- All requests are sent to proxy server
- Proxy server caches objects
- Only new objects are requested from origin server
- Fast, Lower traffic on the link

A **Web cache**—also called a **proxy server**—is a network entity that satisfies HTTP requests on the behalf of an origin Web server.



Conditional GET



cache

server

HTTP request msg
If-modified-since: <date>

object not modified

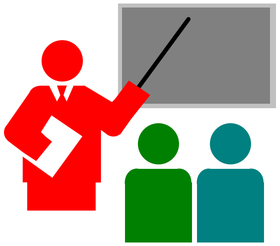
HTTP response
HTTP/1.0
304 Not Modified

HTTP request msg
If-modified-since: <date>

object modified

HTTP response
HTTP/1.0 200 OK
<data>

The copy of an object residing in the cache may be out-of-date. The object housed in the Web server may have been modified since the copy was cached at the client. Fortunately, HTTP has a mechanism that allows a cache to verify that its objects are up to date. This mechanism is called the conditional GET.



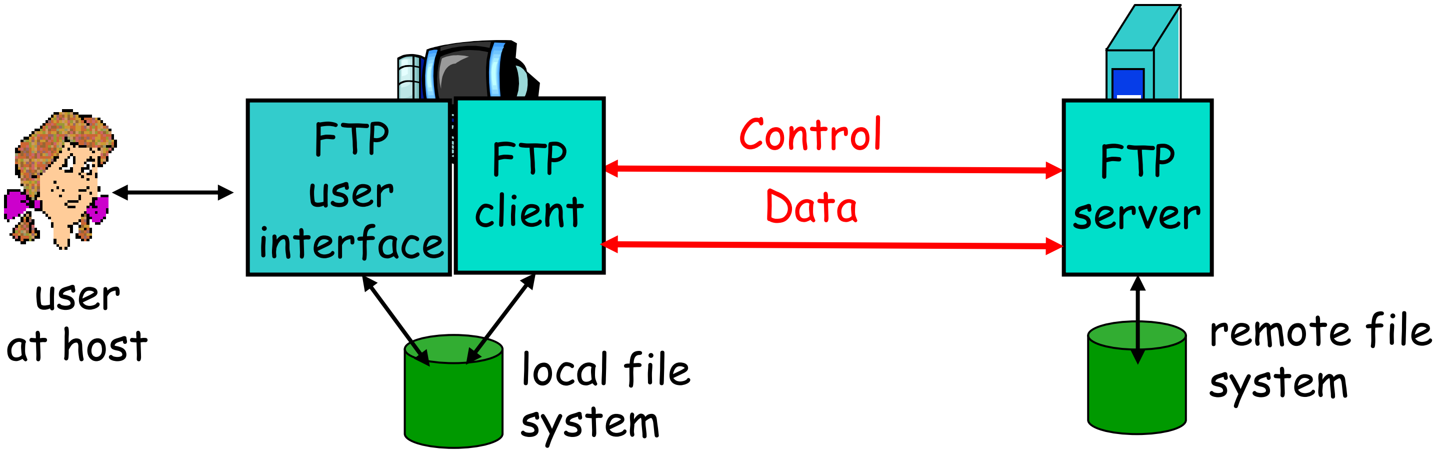
HTTP: Summary

1. HTTP is a client-server protocol.
   Uses text-based messages
2. Web pages are generally written in HTML
3. HTTP uses non-persistent/persistent TCP connections
4. Cookies allow servers to maintain state
5. Proxy servers improve performance by caching frequently used pages
6. Conditional gets allows proxy servers to reduce Internet traffic

File Transfer and Electronic Mail

1. File Transfer Protocol (FTP) Commands and Replies
2. Simple Mail Transfer Protocol (SMTP)
3. HTTP vs. SMTP
4. Mail Access Protocols
   - ❑ Post-Office Protocol version 3 (POP3) protocol
   - ❑ Internet Mail Access Protocol (IMAP)

FTP

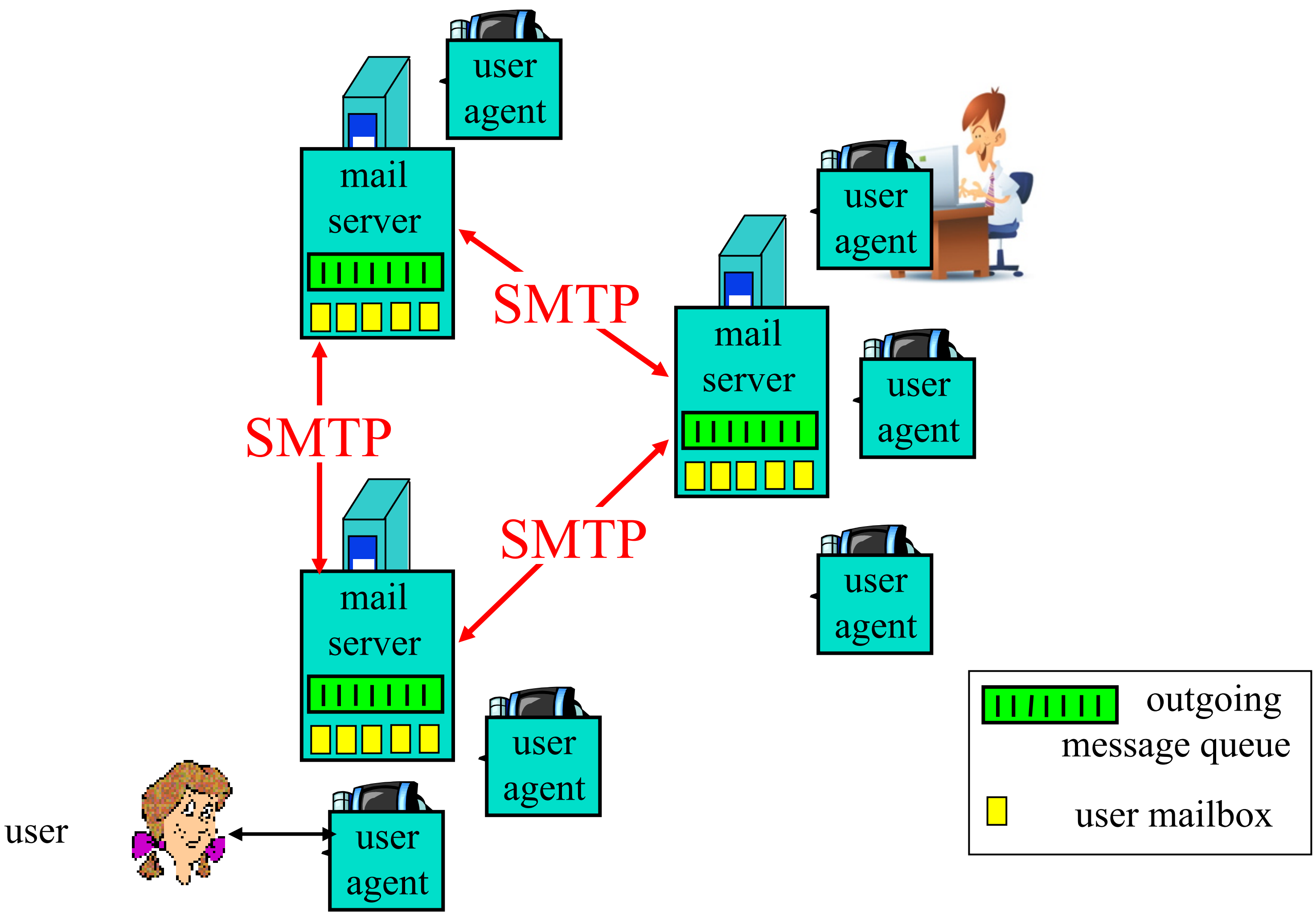


- File transfer protocol. Uses Port 21
- Uses two parallel TCP connections: Control and Data
  ⇒Control is out-of-band
- HTTP uses in-band control
- Control connection is persistent
- Data connections are non-persistent: A new TCP connection is opened for each data file
- Stateful: Server keeps track of user name, directory, ...

FTP Commands and Replies

- **USER** username
- **PASS** password
- **LIST**
- **RETR** filename
- **STOR** filename
- **Status**:
  - 331 Username OK, password required
  - 125 Data connection already open; transfer starting
  - 425 Con't open data connection
  - 452 Error writing file

Electronic Mail



SMTP

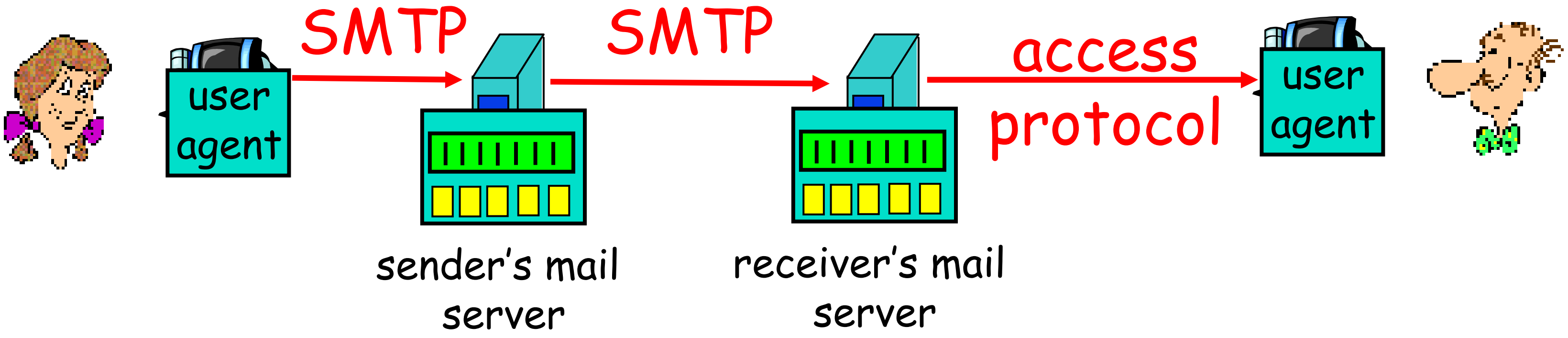
- Simple Mail Transfer Protocol
- Old Protocol: Allows only 7-bit ASCII messages
- All binary objects have to be converted to ASCII
- Uses port 25 at the server

HTTP vs. SMTP

HTTP	SMTP
Persistent/Non-Persistent TCP	Persistent TCP
Mostly Pull	Mostly Push
Accepts binary objects	Accepts only 7-bit ASCII
One Object/response	Multiple objects/message

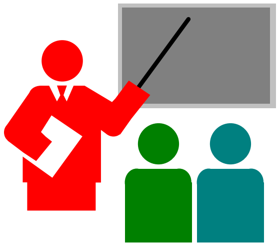
Mail Access Protocols

- SMTP can be used to send messages to destination user agent
  ⇒ Requires destination to be always accessible
- Post Office Protocol - Version 3 (POP3)
- Internet Mail Access Protocol (IMAP)
- HTTP



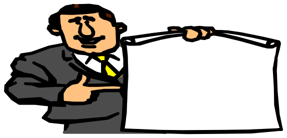
IMAP

- Internet Mail Access Protocol
- More sophisticated than POP3
- Allows users to maintain folders on the server
- Messages can be moved from one folder to another
- Users can get only headers or other components of the message
- Official IMAP site: www.imap.org



FTP and Mail: Summary

1. FTP is a client-server protocol. Uses Port 21.
2. FTP uses two parallel TCP connections for control and data
3. SMTP is the protocol send email
4. SMTP uses only 7-bit ASCII messages
5. POP3, IMAP, or HTTP is used to receive email



Domain Name Service

1. DNS Hierarchy
2. How DNS Works?
3. DNS Records
4. DNS Message Format
5. DNS Registration
6. DNS Vulnerability

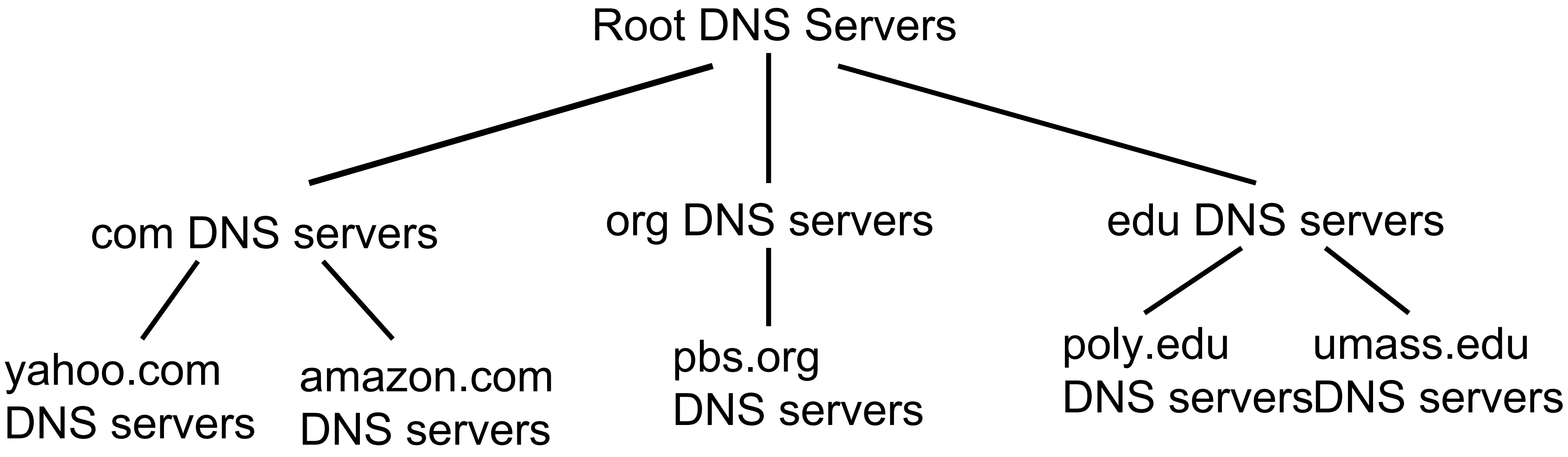
DNS

- Domain Name Service
- DNS servers translate a host name to IP address
  E.g., www.sau.int ⇒ 192.168.43.1
- Distributed database of all hosts in the universe
- Other Services:
  - **Host Aliasing**: www.bijoycc.site
    or www.cs.sau.int/~bijoy/
  - **Mail Server Aliasing**: MX record (e.g., bijoycc@sau.int)
  - **Load Distribution**: Multiple addresses, rotated

DNS Example

```
C:\>nslookup www.sau.int

Server:  Unknown

Address:  192.168.43.1

C:\>nslookup www.google.com

Server: Unknown,  Address:  128.252.0.1

Non-authoritative answer:
Name:    www.google.com
Addresses:  74.125.225.48, 74.125.225.52, 74.125.225.50, 74.125.225.49,
74.125.225.51
```

DNS Hierarchy

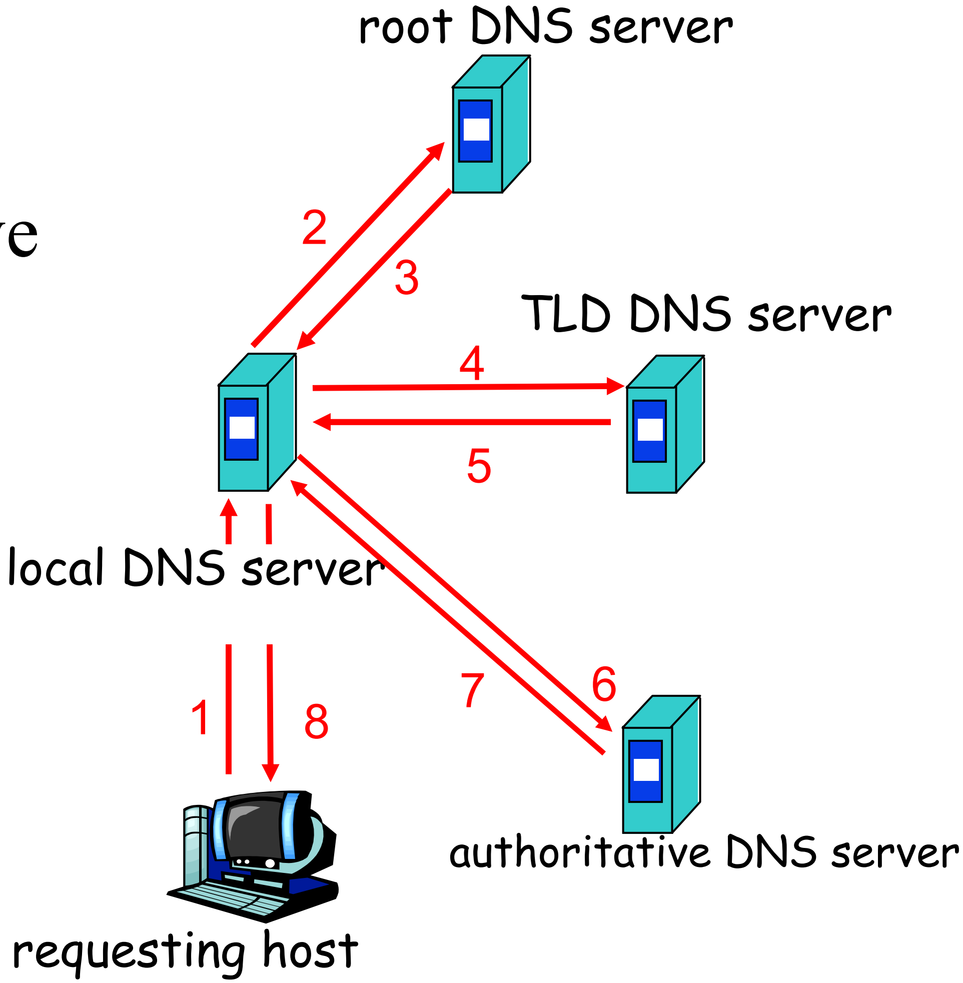


- Root DNS Servers: Root servers, or DNS root servers, are name servers that are responsible for the functionality of the DNS as well as the entire Internet. Total 13 roots server are available
- Top-level Domain (TLD) servers
- Authoritative DNS Servers

How DNS Works?

- Redirects
- **Recursive queries**: Give me an answer
- **Iterative queries**: Give me an answer or a hint
- DNS responses are cached

The query from the requesting host to the local DNS server is recursive, and the remaining queries are iterative.



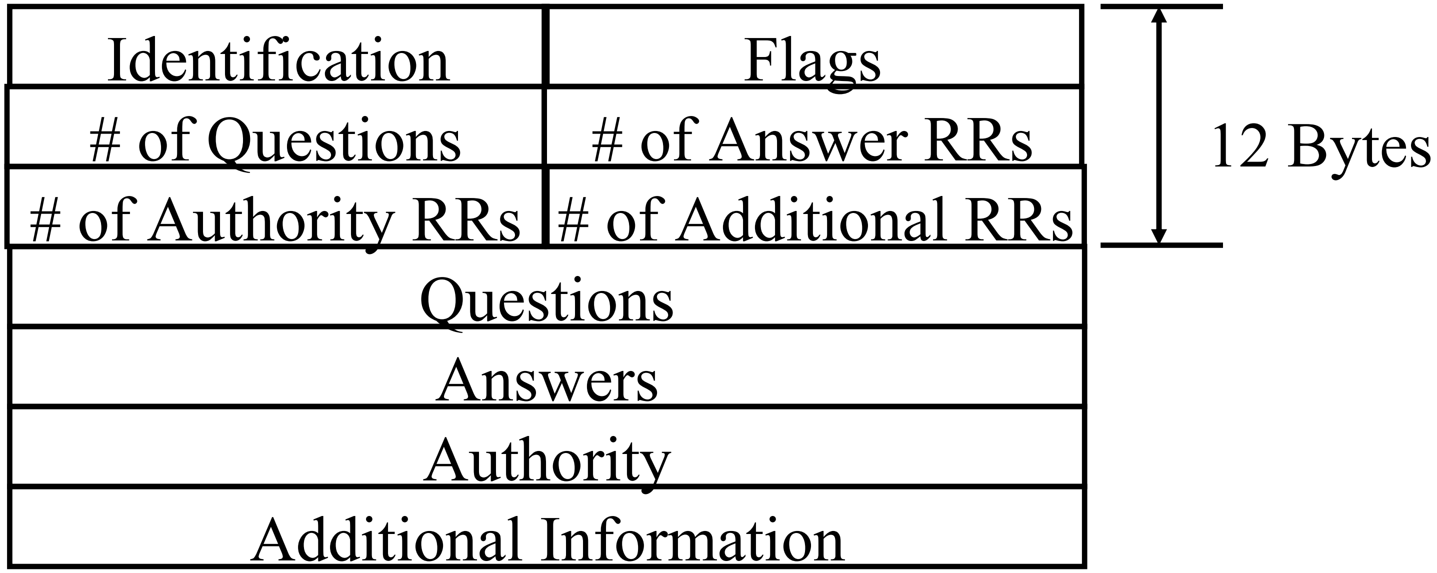
DNS Records

- Resource Records=(Name, Value, Type, TTL)
- Type=A: IP Address for the host name
- Type=NS: Name server for the domain name
- TYPE=CNAME: Canonical name for a host name
- Type=MX: Canonical name of mail server

The TTL serves to tell the recursive server or local resolver how long it should keep said record in its cache.

DNS Message Format

- **Questions**: Name, type
- **Answers**: Name, type, value, TTL
- **Authority**: Other authoritative servers
- **Additional**: Other information, e.g., IP address of canonical name in MX response

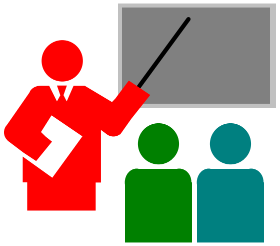


Identification | Flags
# of Questions | # of Answer RRs
# of Authority RRs | # of Additional RRs
Questions
Answers
Authority
Additional Information

12 Bytes

DNS Vulnerability

- Distributed Denial of service attack on Name server
- DNS cache poisoning – A server gives wrong answer



DNS: Summary

1. DNS is used to resolve names to IP address
2. Also provides Name aliasing (CNAME), Mail Server (MX) records
3. DNS is a distributed database
   ⇒ Servers ask other servers for answers when needed
4. Recursive (answer only) or iterative (answer or hint) queries
5. Root Servers, Top level domain servers, Authoritative servers

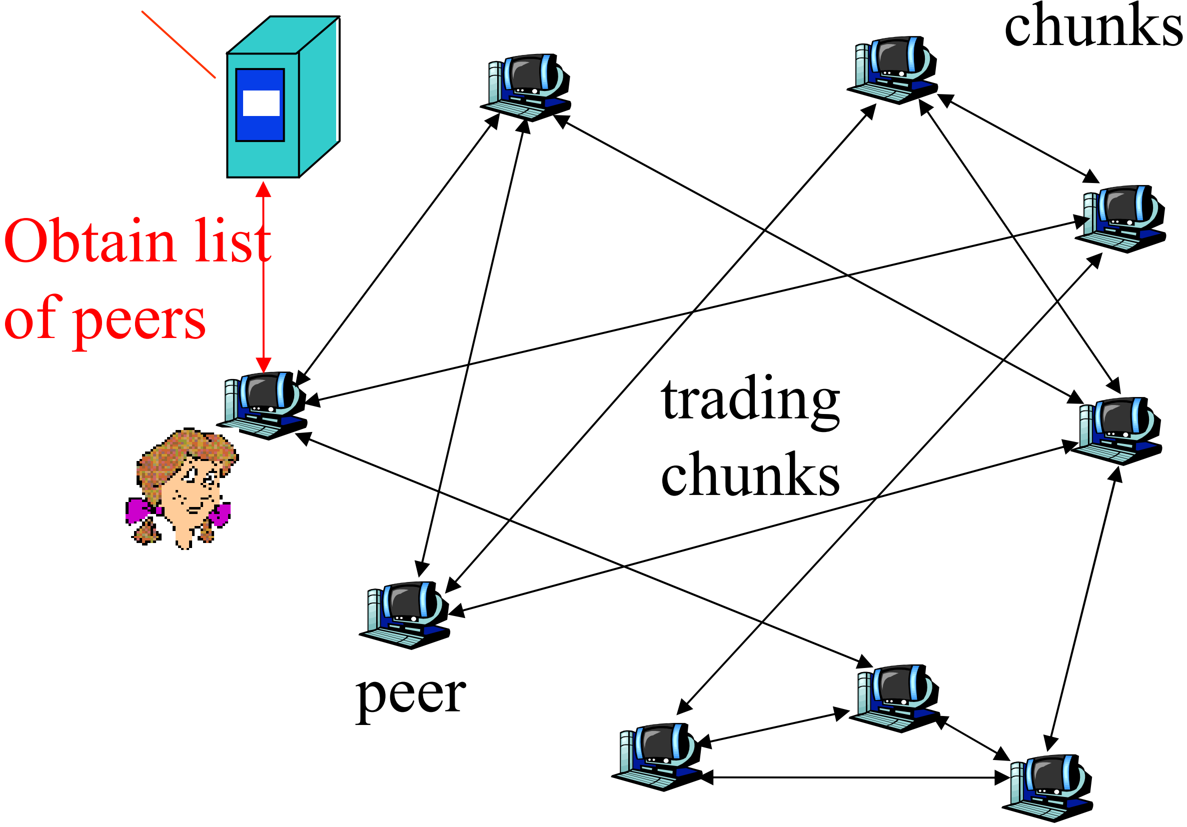
Peer-to-Peer Applications

1. Client Server vs. P2P Scalability
2. P2P File Distribution (BitTorrent)
3. Distributed Hash Tables (DHTs)
4. Peer Churn
5. Network Address Translation (NAT)
6. Skype

P2P File Distribution (BitTorrent)

Tracker: tracks peers participating in torrent

Torrent: group of peers exchanging chunks of a file

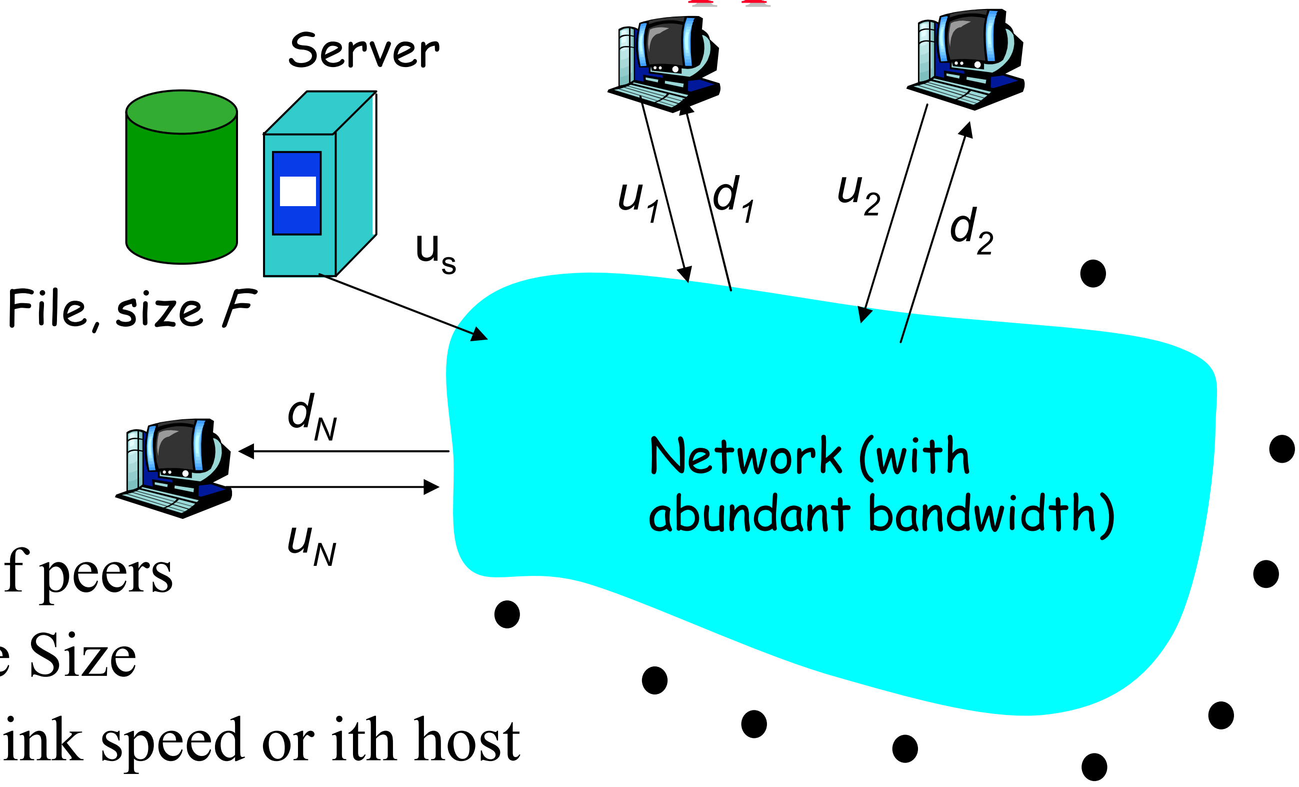


Obtain list of peers

trading chunks

peer

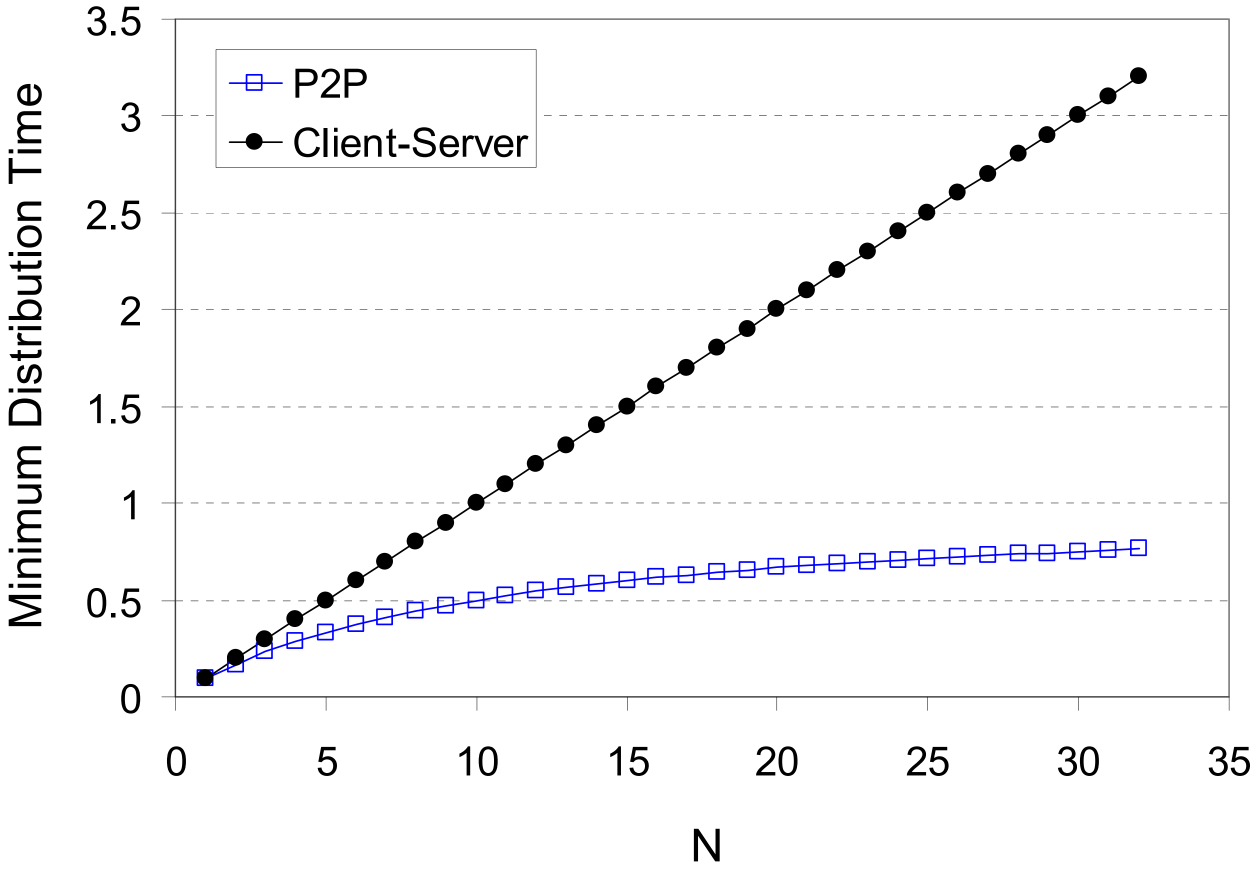
Peer-to-Peer Applications



- N= # of peers
- F= File Size
- u_i= uplink speed or ith host
- d_i= downlink speed of ith host
- d_{min}=min$\{d_1, d_2, \ldots, d_N\}$
- $D_{cs} \geq \max\{ NF/u_s, F/d_{min}\}$
- $D_{P2P} \geq \max\{F/u_s, F/d_{min}, NF/(u_s+\Sigma u_i)\}$

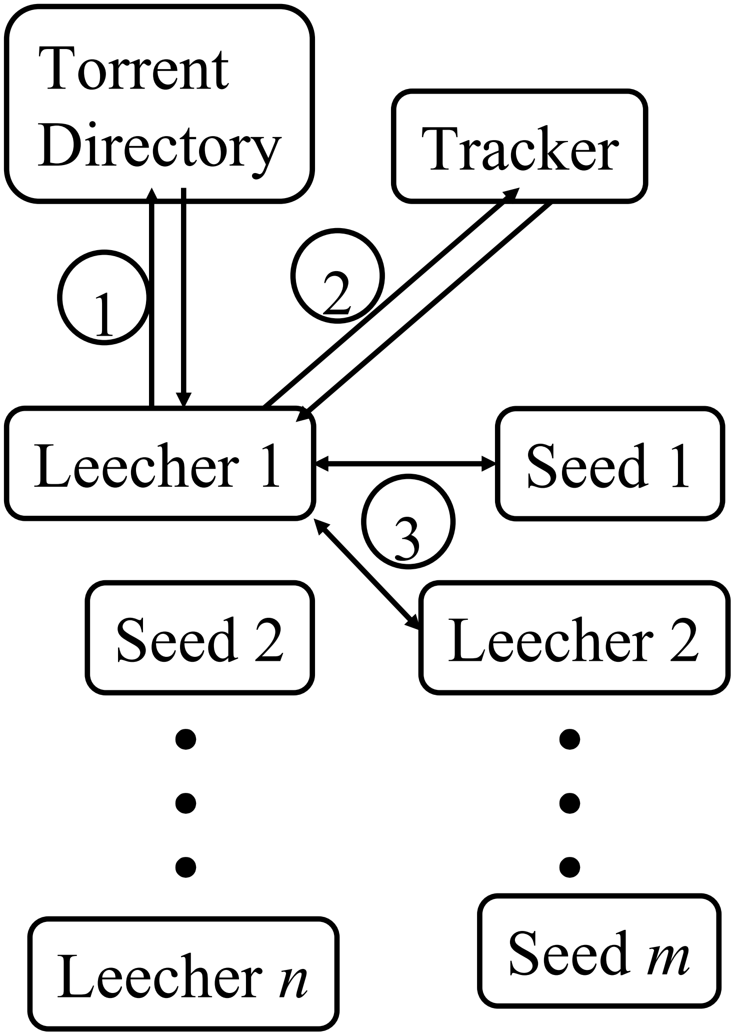
Client Server vs. P2P Scalability

Client upload rate = u, F/u = 1 hour, $u_s = 10u$, $d_{min} \geq u_s$



BitTorrent P2P File Distribution

- **Peers**=nodes participating in a file distribution
- **Torrent**=Set of all peers
- **Torrent File** =a file containing information about the tracker, object ID, and file
- Files are segmented into equal size **chunks** (256kB)
- **Seeds**=Peers that have the complete file
- **Leechers**=Peers that have incomplete file
- **Tracker**=Has list of all peers



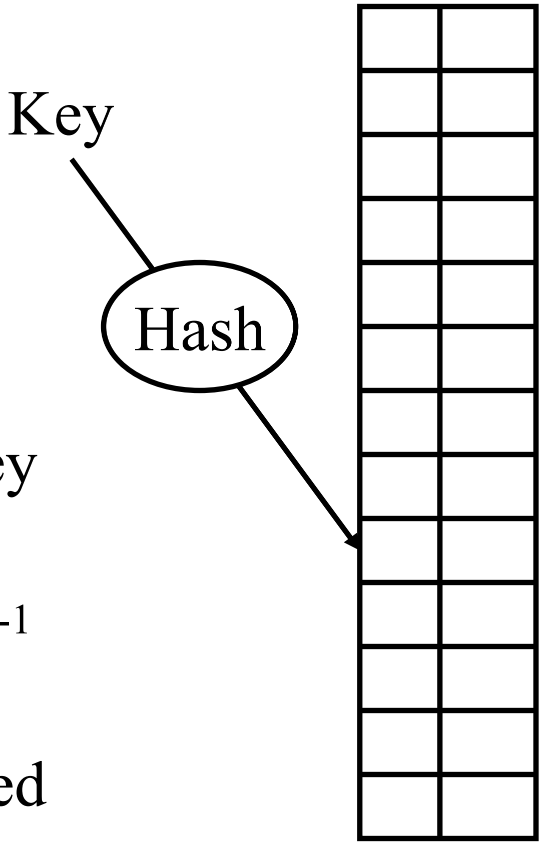
BitTorrent File Distribution (Cont)

1. Alice uses torrent directories (search engines) to find a torrent for "Raj Jain's Lecture"
2. Alice contact the tracker to get the current list of peers
   Tracker may provide random subset (say 50) peers
3. Alice sets up TCP connections with these peers in parallel and gets a map of available chunks

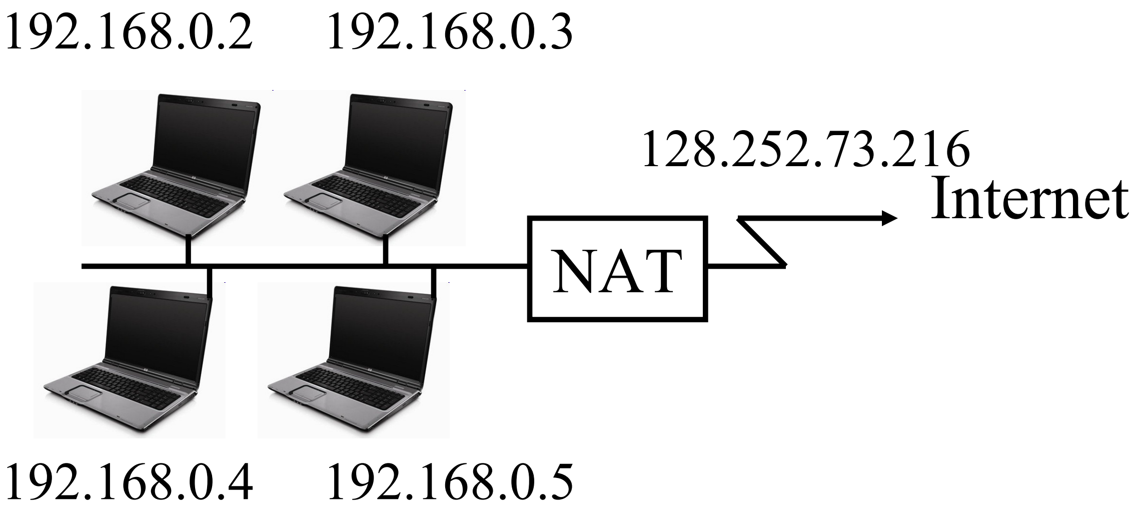
- ❑ Requests least available chunks first (**rarest first**)
- ❑ Every 10 seconds, Alice calculates the receiving rates
- ❑ Sends to (**Unchokes**) the top 4 senders
- ❑ Every 30 seconds, Alice sends to one randomly selected peer (**optimistically unchokes**)
  ⇒ Helps find high-rate neighbors
- ❑ Ref: www.bittorrent.org http://en.wikipedia.org/wiki/BitTorrent_(protocol)

Distributed Hash Tables (DHTs)

- Hash Tables are used to find an entry in a table of (key, value) pairs in the database
- Distributed Hash Tables are used for very large databases
- Database is partitioned and stored in many peers
- Hash the key to find the peer that has the key
- Peers are numbered from 0 thru 2^{n-1} (n-bit)
- Objects (keys) are numbered from 0 thru 2^{n-1} (n-bit)
- If some peers are missing, the keys are stored at the nearest peer.
- Ref: http://en.wikipedia.org/wiki/Distributed_hash_table



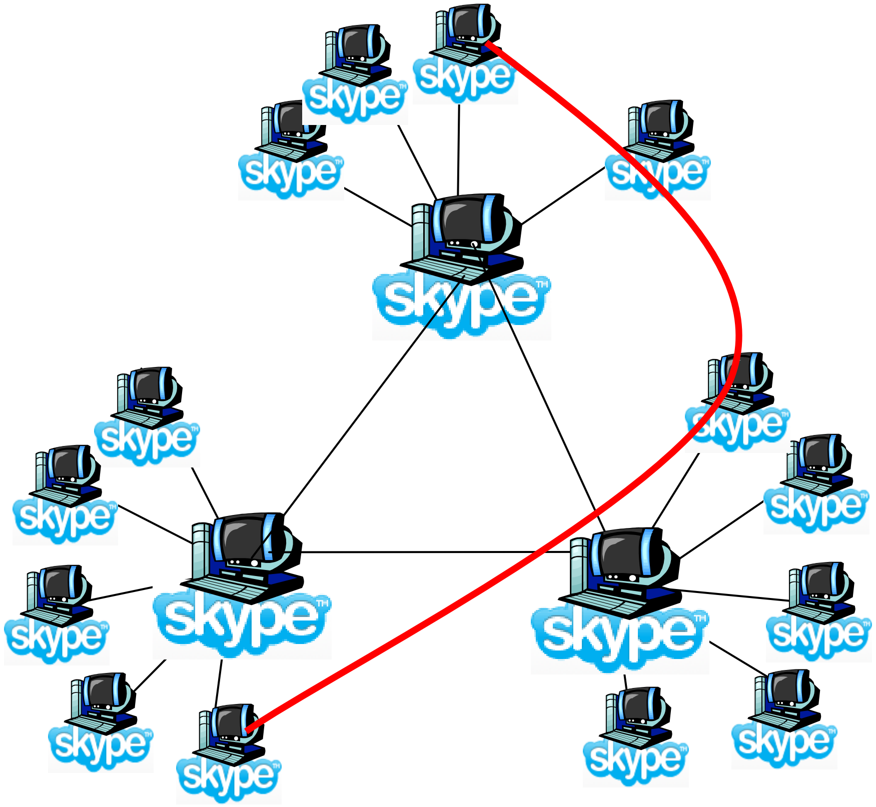
Network Address Translation (NAT)

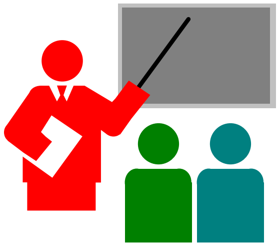


- Private IP addresses 192.168.x.x
- Can be used by anyone inside their networks
- Cannot be used on the public Internet
- NAT overwrites source addresses on all outgoing packets and overwrites destination addresses on all incoming packets
- Only outgoing connections are possible

Skype

- P2P telephony application
  Sky Peer-to-Peer
  ⇒ Skyper ⇒ Skype
- Super-peers keep an index (DHT?) of all active Skype users
- Relays are used to connect NAT'ed nodes





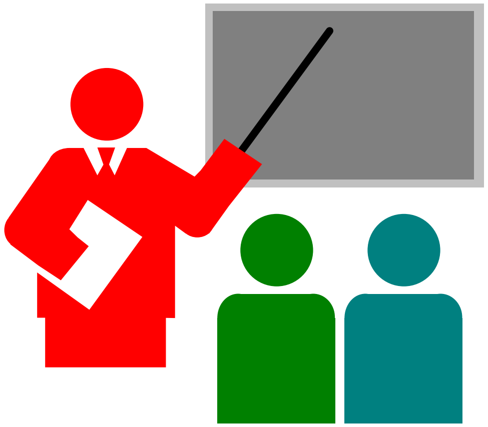
P2P Applications: Summary

1. P2P applications are more scalable

⇒ More efficient when the number of peers is large

2. BitTorrent has peers, trackers, seeds, and leechers
3. BitTorrent unchokes 4 top uploaders and one random node for load balancing
4. Distributed hash tables are used to manage large distributed databases used in P2P applications
5. Skype uses super-nodes to keep track of active users and provides relays for users behind NATs.

Application Layer: Summary



1. Applications use TCP/UDP ports for communication
2. HTTP/FTP/SMTP are client-server protocols and use TCP connections
3. HTTP is stateless but cookies allows servers to maintain state
4. Proxy servers improve performance by caching
5. BitTorrent is a P2P file distribution protocol and uses trackers to keep list of peers