Unit - 1
Application Layer

Reference: Computer Networking : A top down approach by Kurose and Ross, 6^{th} Edition

Unit 1 - Application Layer Syllabus

- The Web and HTTP: Overview of HTTP, Non-Persistent and Persistent Connections, HTTP Message Format, User-Server Interaction-Cookies, Web Caching, The Conditional GET.

- File Transfer- FTP: FTP Commands and Replies

- Electronic Mail in the Internet: SMTP, Comparison with HTTP, Mail Access Protocols.

- DNS—The Internet's Directory Service: Services Provided by DNS, Overview of How DNS Works, DNS Records and Messages

- Peer-to Peer Applications: P2P File Distribution, Distributed Hash Tables (DHTs).

Application-Layer Protocols

- The types of msg – request, response
- syntax
- semantics
- Rules

The Web and HTTP

- HTTP – web’s application layer protocol
- HTTP implemented as client and server program.
- Web page – Base html, objects
- Object is referred by URL
- URL – hostname of server and object pathname

http://www.someSchool.edu/someDepartment/picture.gif

HTTP

- Web browser – client side of HTTP ex:- IE, chrome, firefox
- Web Server – server side of HTTP ex:- Apache and Microsoft Internet Information Server.
- Client sends HTTP request message, Server sends HTTP response message
- HTTP uses TCP and establishes the connection
- Stateless protocol

HTTP

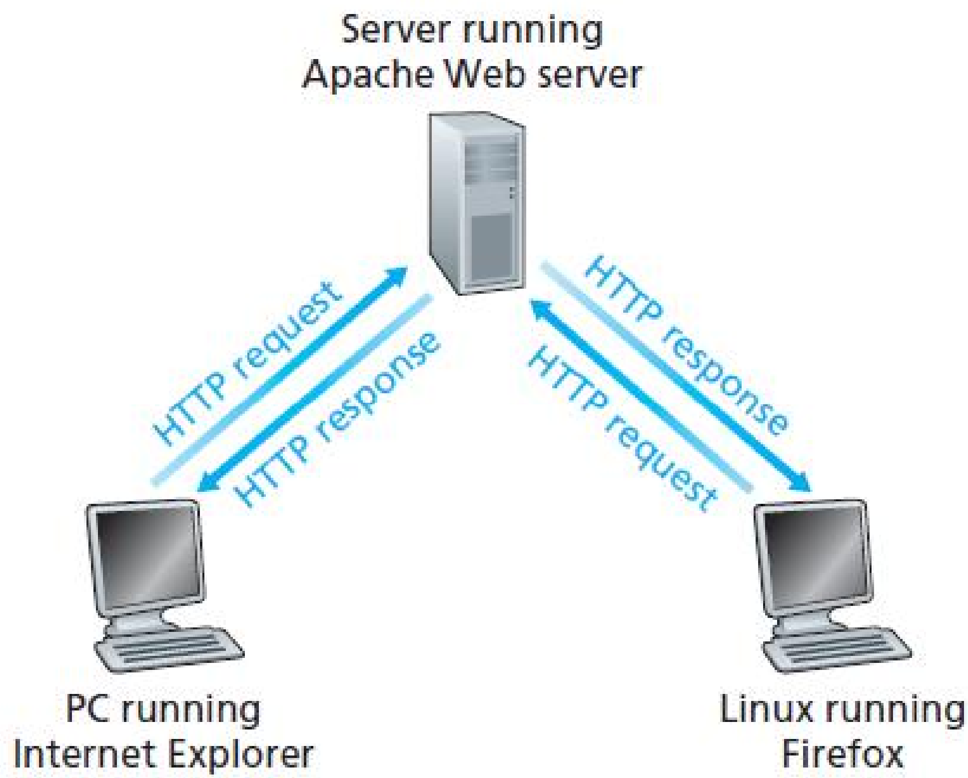


Figure 2.6 ♦ HTTP request-response behavior

Types of TCP Connections used by HTTP

- Non-Persistent Connection – Separate TCP connection is used for each request/response pair
- Persistent Connection – Same TCP connection is used for each request/response pair

HTTP with Non-Persistent Connections

- To download one base html web page and 10 objects totally 11 TCP connections are made.
- TCP connections can be serial or parallel.
- RTT - packet-propagation delay + packet queuing delays + packet-processing delays.
- Total response time is 2 RTT + transmission time of the HTML file for each TCP connection
- For each connection TCP buffers and variables should be allocated

Time estimation

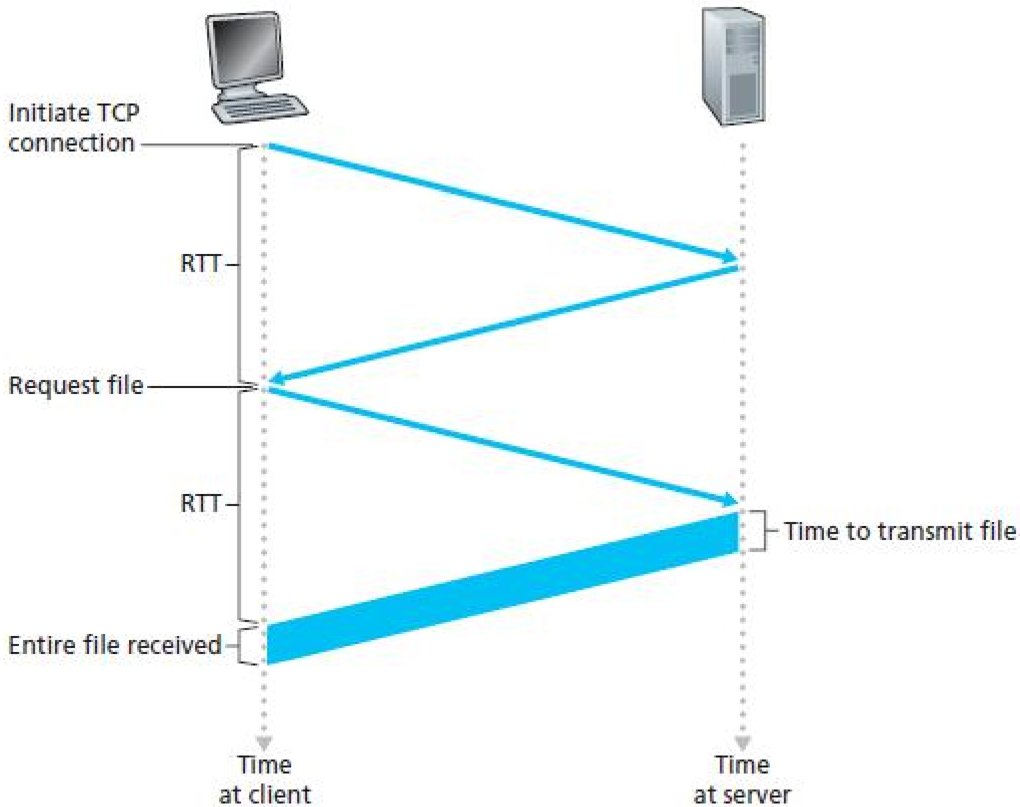


Figure 2.7 ♦ Back-of-the-envelope calculation for the time needed to request and receive an HTML file

HTTP with Persistent Connections

- Only one TCP connection is created
- All 11 objects can be sent over the same connection
- Requests can be pipelined (sent back to back)
- Default mode of HTTP

HTTP Message Format

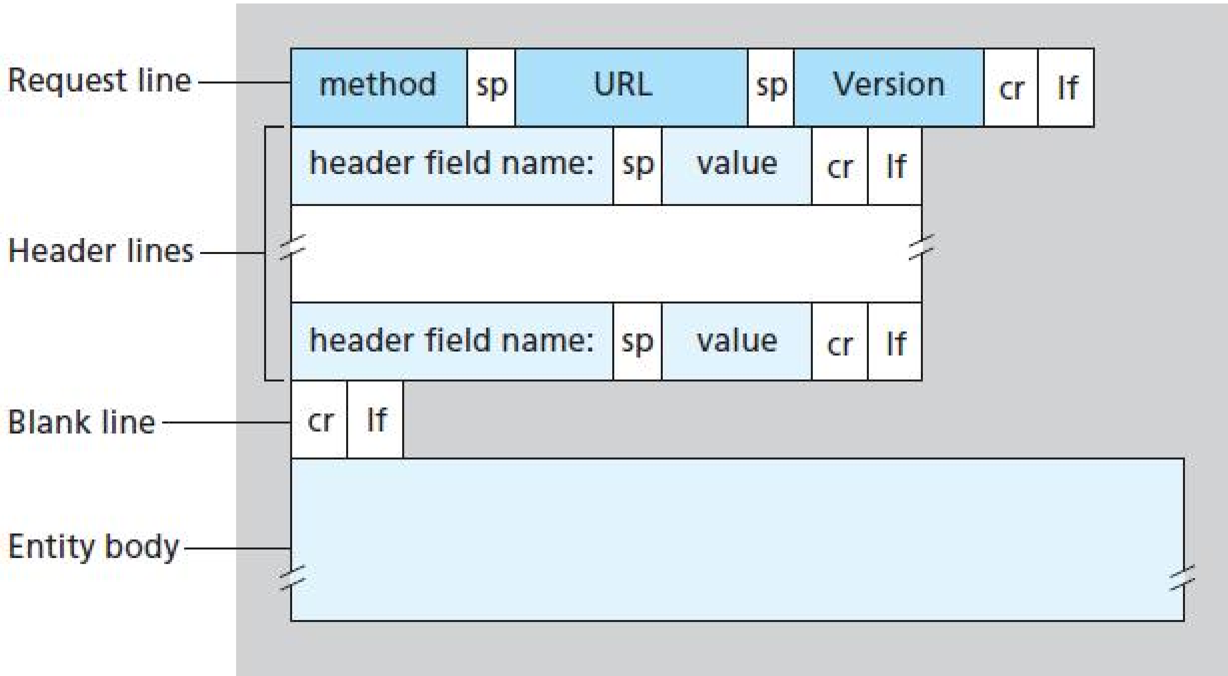


Figure 2.8 ♦ General format of an HTTP request message

HTTP Message Format

Example

```
GET /somedir/page.html HTTP/1.1
Host: www.someschool.edu
Connection: close
User-agent: Mozilla/4.0
Accept-language: fr
```

General Format

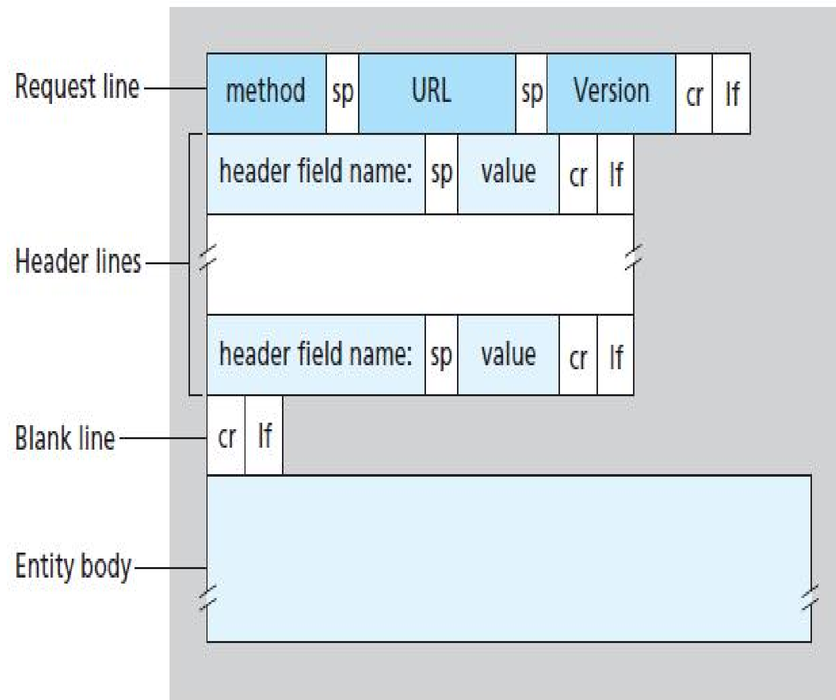


Figure 2.8 ♦ General format of an HTTP request message

Method Field

- GET – user requests an object, Can also use extended URLs like www.somesite.com/animalsearch?monkeys&bananas
- POST – user has filled a form or given search words to a search engine. The entity body contains the values entered.
- HEAD – Returns only the response with the object in it. Used for debugging
- PUT – Used to upload object onto the web server.
- DELETE – Allows user to delete object on the web server.

HTTP Response Message

Example

```
HTTP/1.1 200 OK
Connection: close
Date: Sat, 07 Jul 2007 12:00:15 GMT
Server: Apache/1.3.0 (Unix)
Last-Modified: Sun, 6 May 2007 09:23:24 GMT
Content-Length: 6821
Content-Type: text/html

(data data data data data ...)
```

General Format

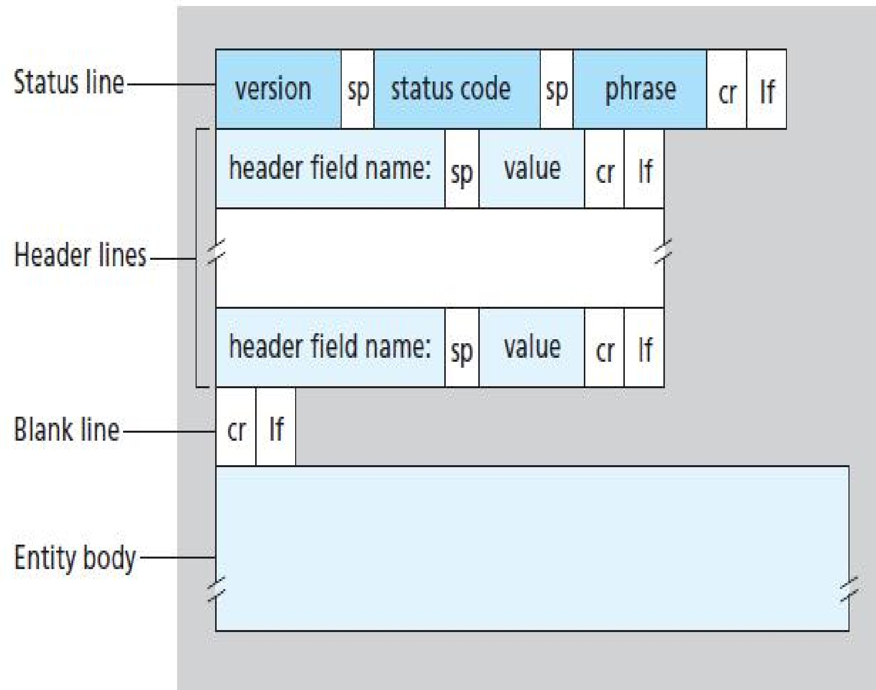


Figure 2.9 ♦ General format of an HTTP response message

Status Code and Phrases

- 200 OK
- 301 Moved Permanently – the new URL will be in Location header field.
- 400 Bad Request – Request could not be understood i.e bad syntax
- 404 Not Found
- 505 HTTP Version Not Supported

User-Server Interaction: Cookies

- Four components:

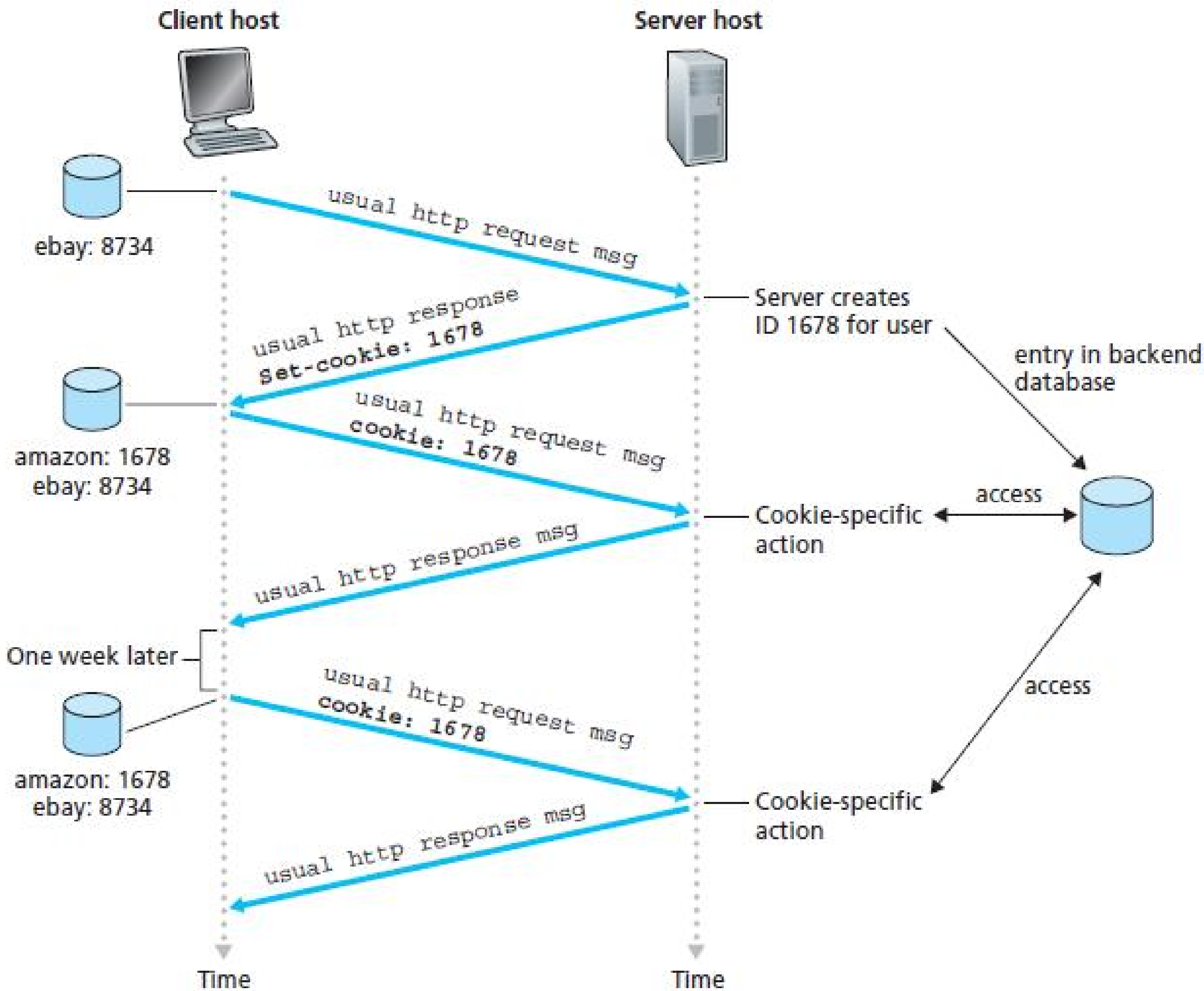
(1) a cookie header line in the HTTP response message;

(2) a cookie header line in the HTTP request message;

(3) a cookie file kept on the user's end system and managed by the user's browser;

(4) a back-end database at the Web site.

Cookies


Client host
Server host
ebay: 8734
usual http request msg
Server creates
ID 1678 for user
usual http response
Set-cookie: 1678
entry in backend
database
amazon: 1678
ebay: 8734
usual http request msg
cookie: 1678
access
Cookie-specific
action
usual http response msg
One week later
usual http request msg
cookie: 1678
access
amazon: 1678
ebay: 8734
Cookie-specific
action
usual http response msg
Time
Time

Web Caching

- Web cache also called proxy server
- Provided by ISP, campus networks etc.
- Can reduce bottleneck at the origin server.
- Reduces traffic on institution's access link thereby reducing the bandwidth required

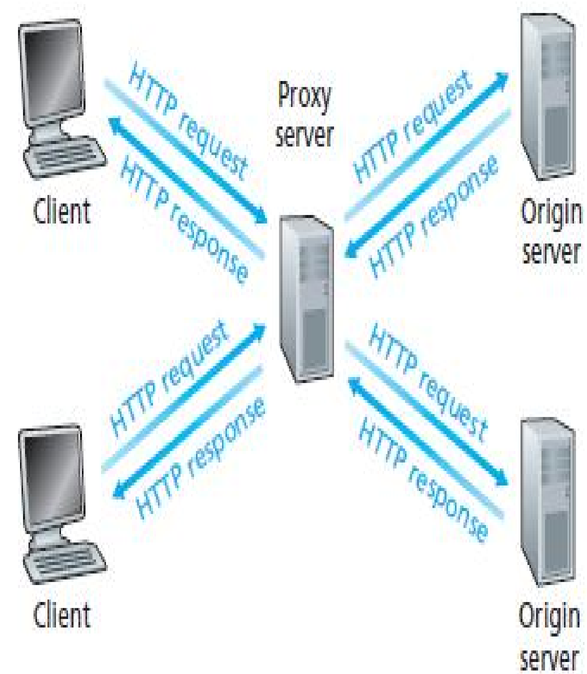


Figure 2.11 ♦ Clients requesting objects through a Web cache

Example network

- 15 requests per second
- Object size is 1Mbps
- Internet delay is 2 seconds
- Traffic intensity on LAN

$$(15 \text{ requests/sec}) \cdot (1 \text{ Mbits/request})/(100 \text{ Mbps}) = 0.15$$

- Traffic intensity on access link

$$(15 \text{ requests/sec}) \cdot (1 \text{ Mbits/request})/(15 \text{ Mbps}) = 1$$

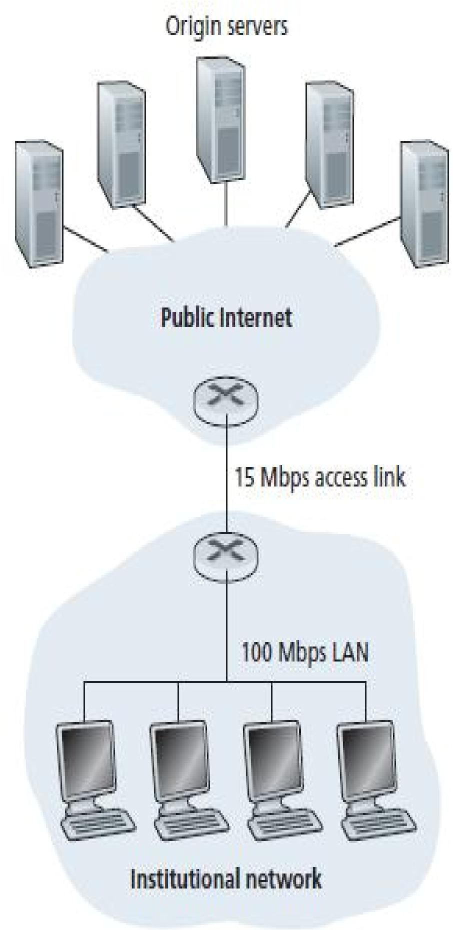


Figure 2.12 ♦ Bottleneck between an institutional network and the Internet

Using proxy server

- Suppose hit rate is 0.4

$$0.4 \cdot (0.01 \text{ seconds}) + 0.6 \cdot (2.01 \text{ seconds})$$

which is 1.2 seconds

- Better solution compared to upgrading access link to 100Mbps.

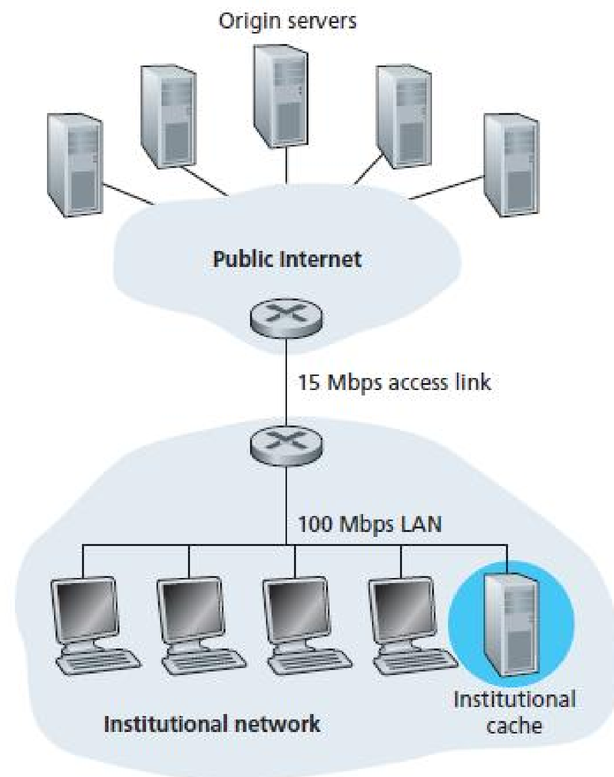


Figure 2.13 ♦ Adding a cache to the institutional network

The conditional GET

First request /reply

First Request

```
GET /fruit/kiwi.gif HTTP/1.1
Host: www.exotiquecuisine.com
```

First Response

```
HTTP/1.1 200 OK
Date: Sat, 7 Jul 2007 15:39:29
Server: Apache/1.3.0 (Unix)
Last-Modified: Wed, 4 Jul 2007 09:23:24
Content-Type: image/gif

(data data data data data ...)
```

Second request/reply

Second Request

```
GET /fruit/kiwi.gif HTTP/1.1
Host: www.exotiquecuisine.com
If-modified-since: Wed, 4 Jul 2007 09:23:24
```

Second Response

```
HTTP/1.1 304 Not Modified
Date: Sat, 14 Jul 2007 15:39:29
Server: Apache/1.3.0 (Unix)
```

FTP

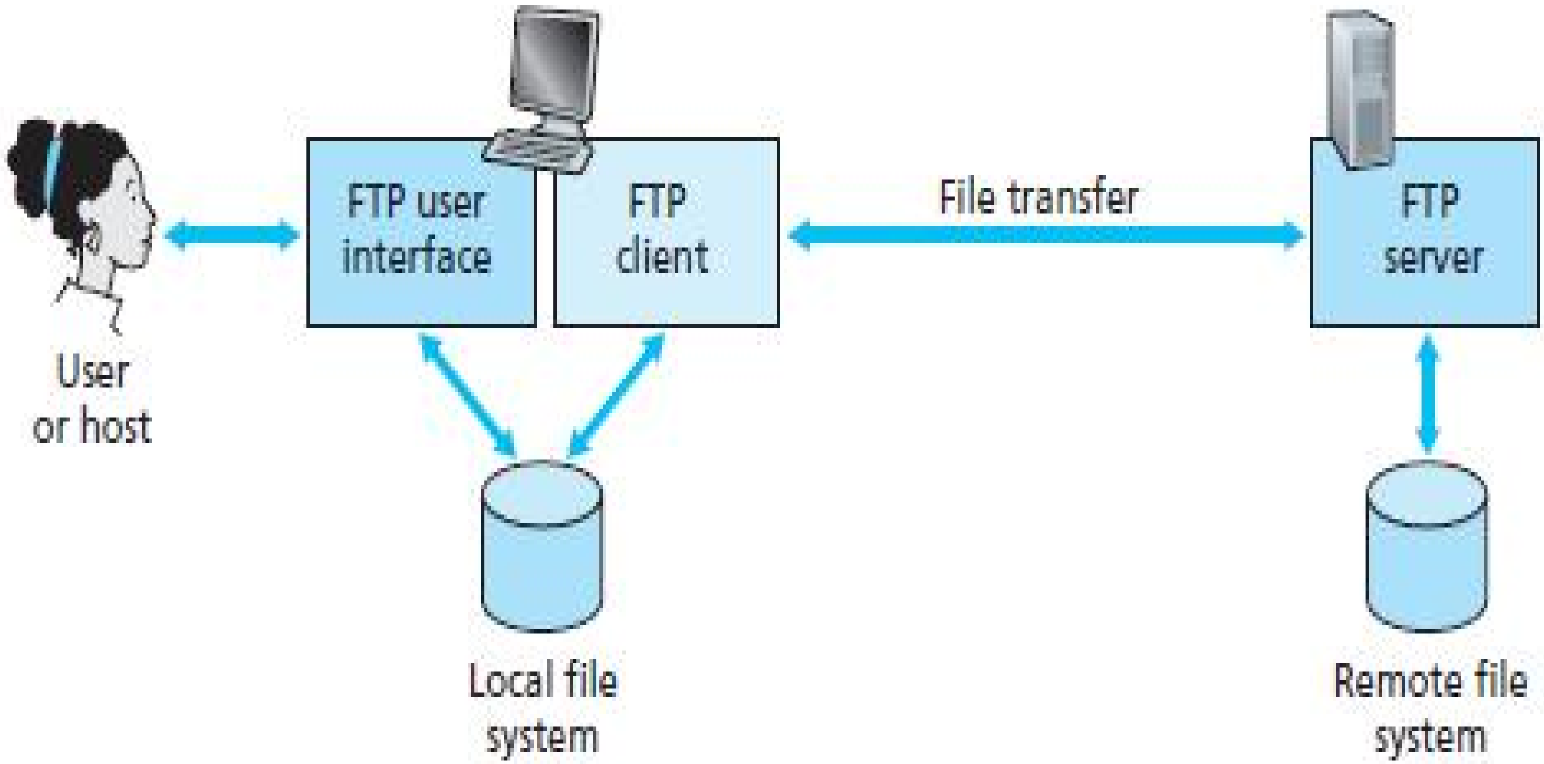


Figure 2.14 ♦ FTP moves files between local and remote file systems

TCP connections of FTP

- User id, password
- 2 parallel TCP connections viz.
- Control Connection – open always, sends id, password, put, get commands
- Data Connection – closed after each file is sent
- Out of band protocol

FTP

- HTTP – inband protocol
- Stateful protocol
- Both data and control information sent in 7 bit ASCII format.

FTP commands and Replies

FTP Commands

- USER username
- PASS password
- LIST
- RETR filename
- STOR filename

FTP Replies

- 331 Username OK, password required
- 125 Data connection already open; transfer starting
- 425 Can't open data connection
- 452 Error writing file

Comparison of HTTP & FTP

HTTP	FTP
Hyper Text Transfer Protocol	File Transfer Protocol
Used by web client and web server to transfer web pages	Used by FTP client and FTP server to transfer huge files
In band protocol i.e both data & control information is sent on the same TCP connection	Out of band protocol i.e data & control information is sent on two different TCP connections
Data can be sent in any format	Data & control information is sent in 7 bit ASCII format.
Stateless protocol – does not remember state information of previous sessions	Stateful protocol – maintains state information of previous sessions

E- mail

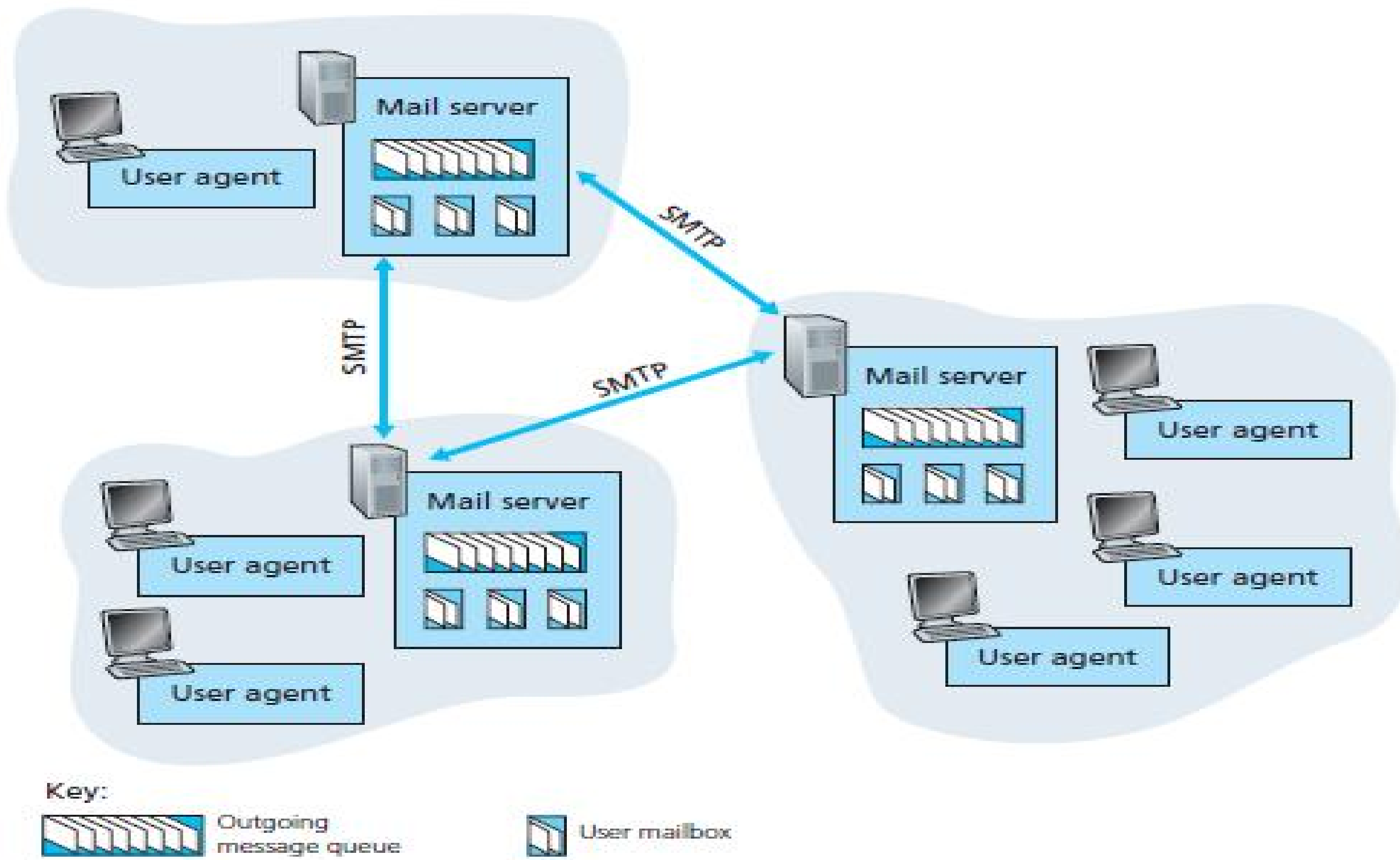


Figure 2.16 ♦ A high-level view of the Internet e-mail system

E-mail

- 3 major components – user agents, mail servers, SMTP
- User agents (mail readers)
  - GUI based user agents – microsoft outlook, apple mail, mozilla thunderbird
  - Text based email – mail, pine, elm
  - Web based interface – web browsers
- Web email – 1996 Sabeer Bhatia developed hotmail sold to microsoft.
- Mail servers – mailbox – queue (in case of delivery failure)
- A's mail box – A's mail server – B's mail server – B's mailbox

SMTP

- SMTP client, SMTP server
- Restricts the body also to be in 7 bit ASCII i.e multimedia is also encoded into 7 bit ASCII

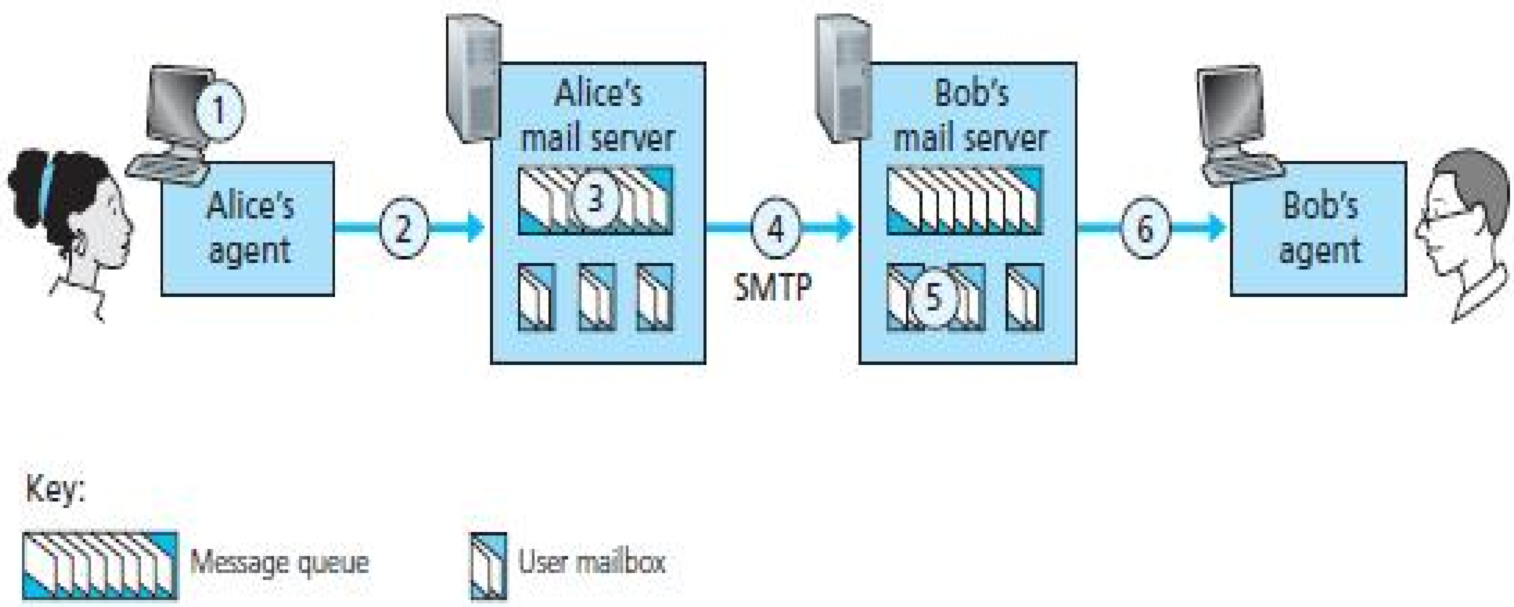


Figure 2.17 ♦ Alice sends a message to Bob

SMTP

- Does not use any intermediate mail server
- Perform handshake
- Uses persistent TCP connections

```
S: 220 hamburger.edu
C: HELO crepes.fr
S: 250 Hello crepes.fr, pleased to meet you
C: MAIL FROM: <alice@crepes.fr>
S: 250 alice@crepes.fr ... Sender ok
C: RCPT TO: <bob@hamburger.edu<
S: 250 bob@hamburger.edu ... Recipient ok
C: DATA
S: 354 Enter mail, end with "." on a line by itself
C: Do you like ketchup?
C: How about pickles?
C: .
S: 250 Message accepted for delivery
C: QUIT
S: 221 hamburger.edu closing connection
```

Comparison of HTTP and SMTP

HTTP

- Hyper Text Transfer Protocol
- HTTP transfers files from a Web server to a Web client
- Uses persistent TCP connections
- **Pull protocol-** the TCP connection is initiated by the machine that wants to receive the file from the server.
- Message can be in any format.
- Each object is encapsulated in its own HTTP response message

SMTP

- Stands for Simple Text Transfer Protocol
- SMTP transfers files from one mail server to another mail server or from sender's user agent to his mail server.
- Uses persistent TCP connections
- **Push protocol** - the TCP connection is initiated by the sending mail server to the receiving mail server
- Message should be in 7 bit ASCII format only
- All message's objects are placed into one message only.

E- mail

- E-mail message format

```
From: alice@crepes.fr
To: bob@hamburger.edu
Subject: Searching for the meaning of life.
```

- Mail Access Protocol

  - POP3 - Post Office Protocol – Version 3

  - IMAP – Internet Mail Access Protocol

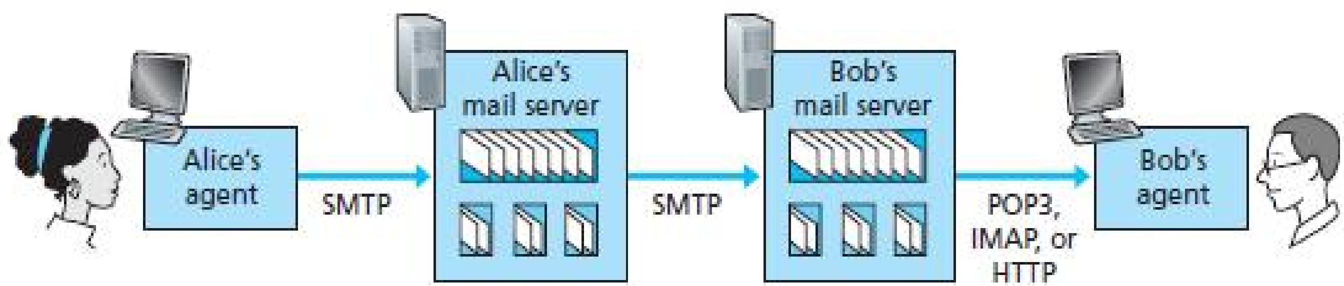


Figure 2.18 ♦ E-mail protocols and their communicating entities

POP3

- 3 phases
  - Authorization phase – username, password
  - Transaction phase
    - mark/unmark messages for deletion, obtain mail statistics.
    - Commands – List, retr, dele
    - responses +OK or –Err
    - Download and delete
    - Download and Keep
  - Update phase – After quit command, deletes the marked messages.
- User agent maintains state but POP3 server is stateless
- Can create folders, rename or move them only in local machine and not on the server

Transaction Example

```
C: list
S: 1 498
S: 2 912
S: .
C: retr 1
S: (blah blah ...
S: .................
S: ..........blah)
S: .
C: dele 1
C: retr 2
S: (blah blah ...
S: .................
S: ..........blah)
S: .
C: dele 2
C: quit
S: +OK POP3 server signing off
```

IMAP

- Can create folders, rename or move them only on the remote mail server, default folder is Inbox.
- IMAP sever maintains state information across sessions.
- Can obtain components of message

Comparison of POP3 & IMAP

POP3

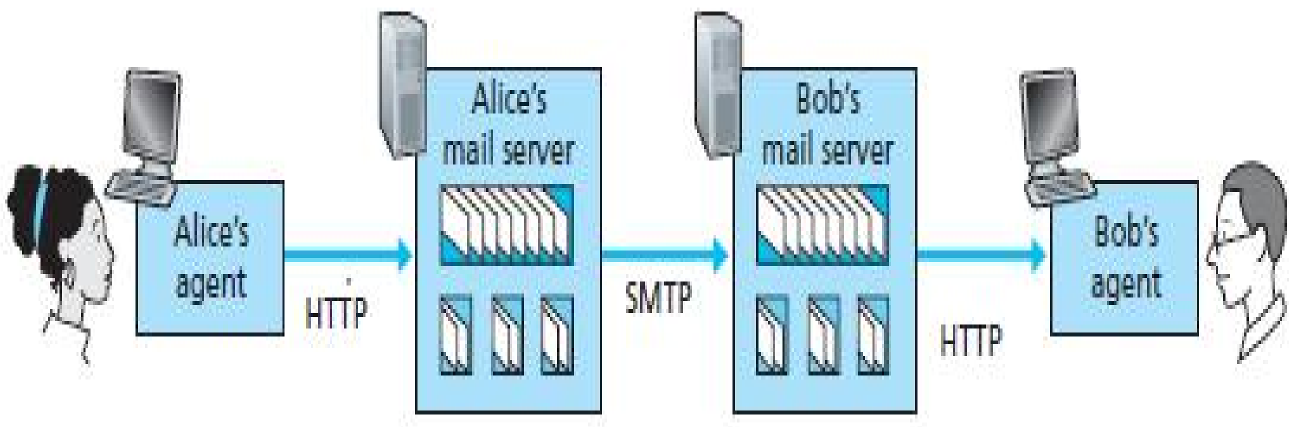
- Post office protocol
- POP3 Servers are stateless
- 2 modes - Download & keep, Download & delete
- Cannot create folders and manage them on servers.
- Entire mail has to be downloaded.

IMAP

- Internet mail access protocol
- IMAP servers are stateful
- Download & keep
- Can create folders and manage them on servers.
- Can obtain components of mail message

Web – based e-mail

- User agent – web browser
- Uses HTTP and SMTP protocols.



Domain Name system

- Host name, ip address
- DNS- distributed hierarchical database & application layer protocol
- Runs on UDP port no. 53
- Uses BIND (Berkeley Internet Name Domain) software
- DNS client and DNS server

DNS Services

- Host name to ipaddress translation
- Host aliasing – Alias hostname, canonical hostname
- Mail server aliasing – Both mail server & web server can have the same name.
- Load Distribution – DNS rotates list of ipaddresses before sending to host.

Centralized design for DNS - Problems

- Single point of failure
- Traffic volume
- Distant centralized database – more delay, & network traffic
- Maintenance – updations
- Not scalable

Hierarchy of DNS Servers

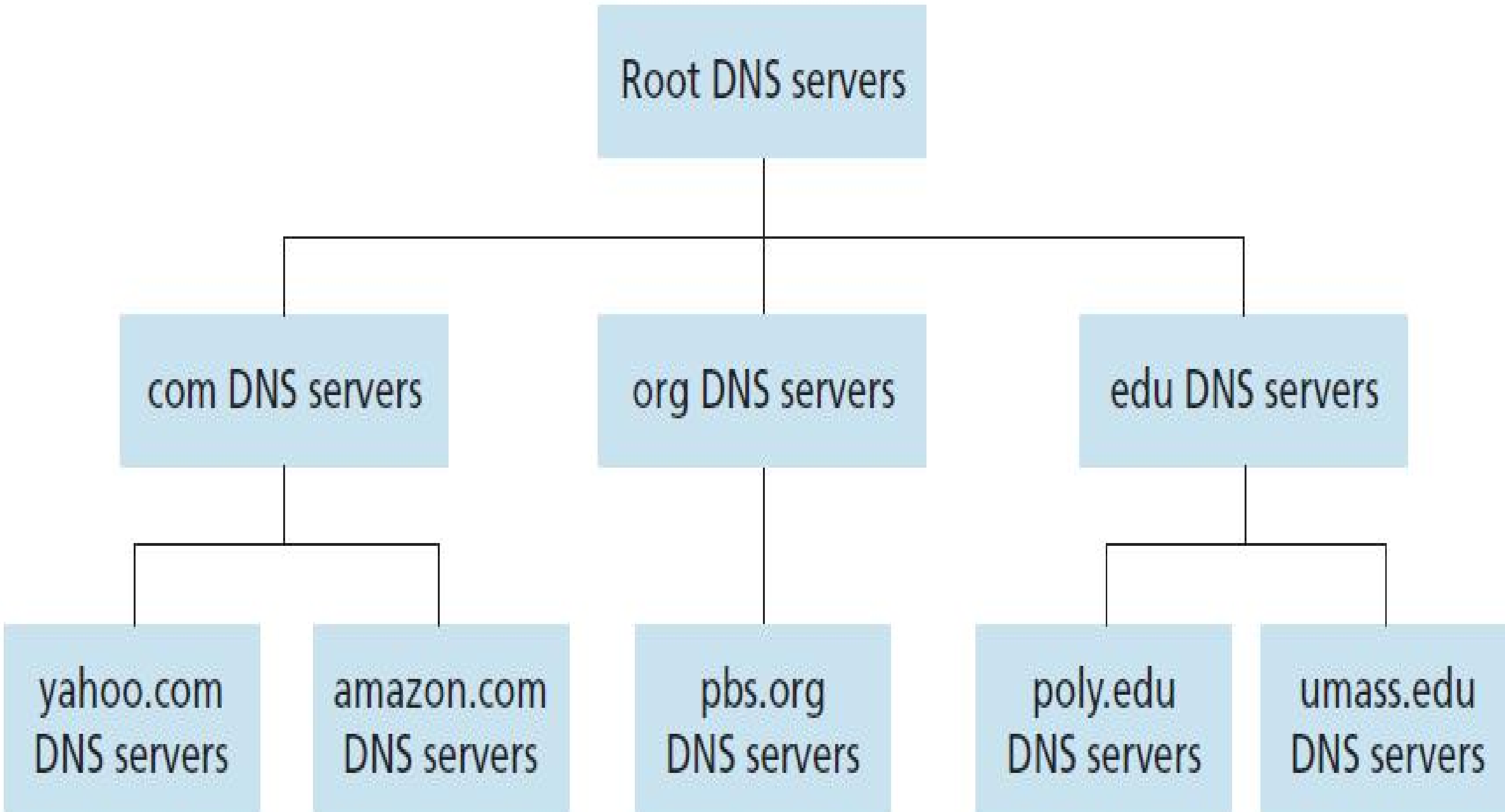


Figure 2.19 ♦ Portion of the hierarchy of DNS servers

Distributed Hierarchical Database

- Root DNS Servers – 13 of them mostly in North America
- Top Level Domain (TLD) Servers – com,edu,org,net,gov,uk,jp,in
- Authoritative Servers
- Local DNS servers -organization or ISP

Iterative queries in DNS Servers

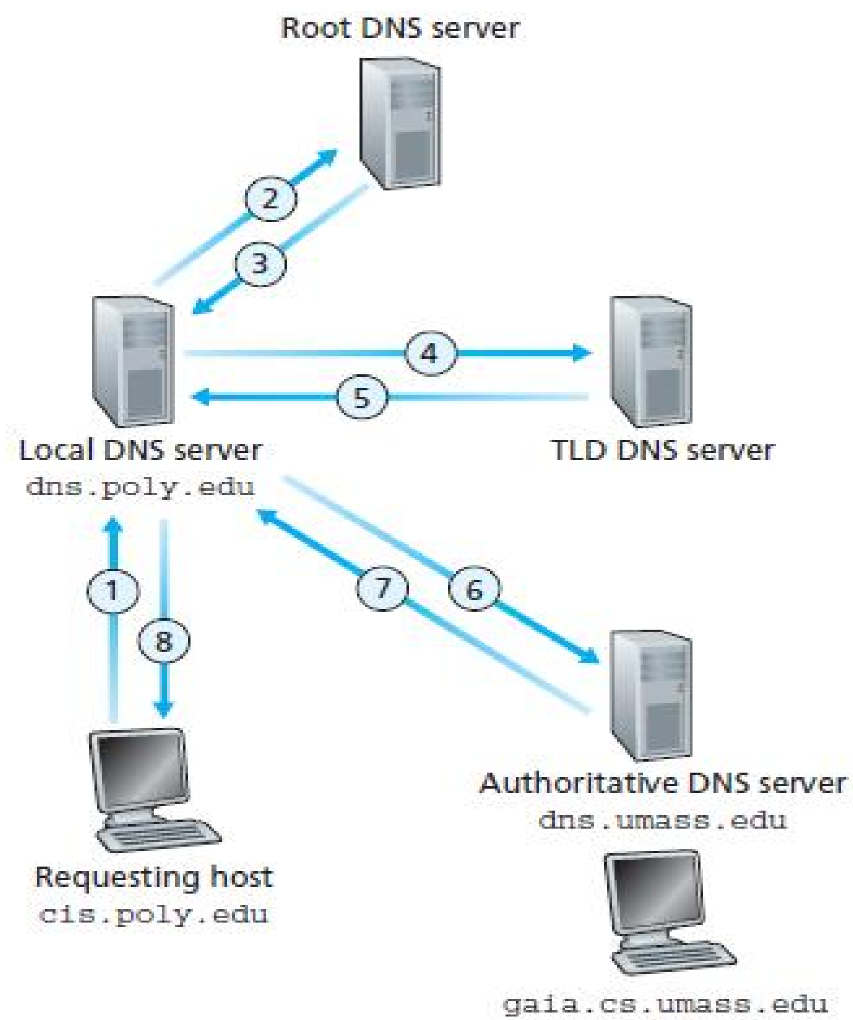


Figure 2.21 ♦ Interaction of the various DNS servers

Recursive queries in DNS Servers

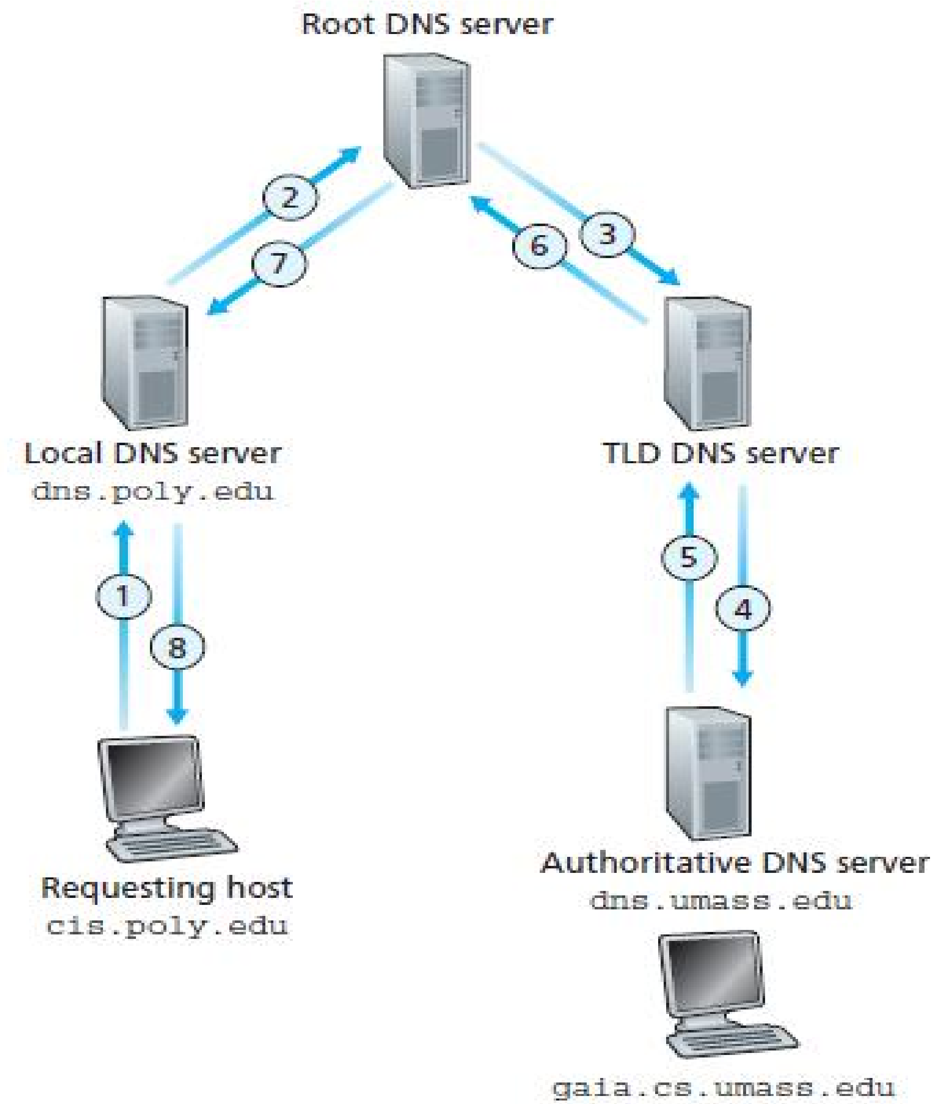


Figure 2.22 ♦ Recursive queries in DNS

DNS records

4 tuple resource record - (Name, Value, Type, TTL)

- Type=A, then Name = hostname and Value = its IP address (relay1.bar.foo.com, 145.37.93.126, A)

- Type=NS, then Name=domain (such as foo.com) and Value=hostname of its authoritative DNS server (foo.com, dns.foo.com, NS)

- Type=CNAME, then Name = Alias hostname and Value = canonical hostname (foo.com, relay1.bar.foo.com, CNAME)

- Type=MX, then Name = Alias hostname of mail server, Value=its canonical name (foo.com, mail.bar.foo.com, MX)

DNS Query & Reply Messages

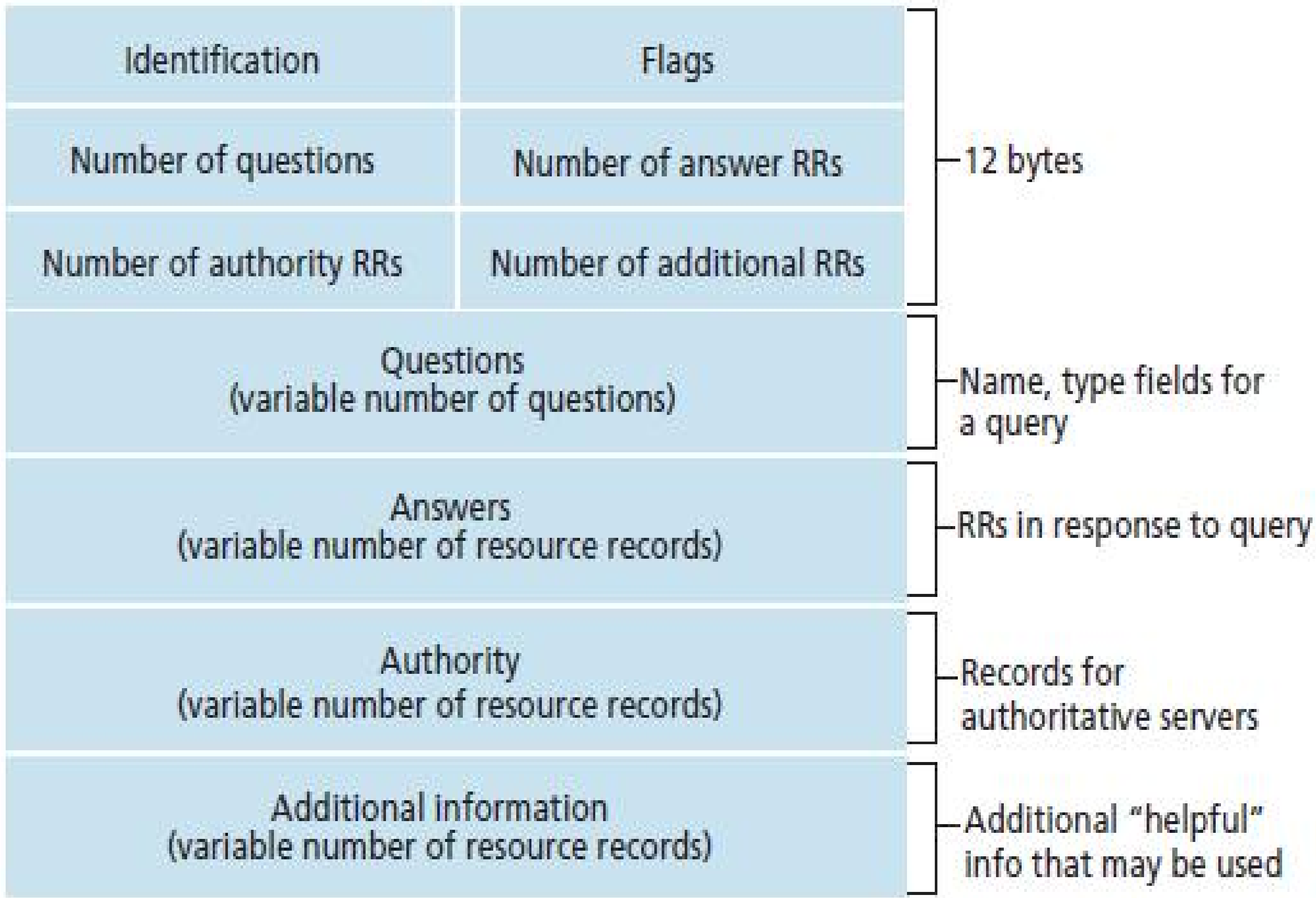


Figure 2.23 ♦ DNS message format

DNS message fields

- Identifier
- Flags
  - 1 bit – query(0), reply(1)
  - 1 bit – authoritative flag, set when reply is from authoritative server
  - 1 bit – recursion desired flag (if it is query), recursion available (if it is reply)
- Questions – Queries with name and type field
- Answers – reply sometimes multiple RRs
- Authority – records of other authoritative servers
- Additional section – helpful records, Type A record with type MX record.
- NSlookup

Inserting DNS records

- Registrar – ex: Network solutions
- ICANN accredits registrars - http://www.internic.net.
- Suppose networkutopia.com is the new website
- Provide names of primary & secondary authoritative servers

- Ex: dns1.networkutopia.com, dns2.networkutopia.com,
  212.212.212.1, and 212.212.212.2.

- Type NS and type A record should be entered in TLD servers
  (networkutopia.com, dns1.networkutopia.com, NS)
  (dns1.networkutopia.com, 212.212.212.1, A)

- Enter Type A resource record for the Web server www.networkutopia.com and the Type MX resource record the mail server mail.networkutopia.com into authoritative DNS servers

DNS Vulnerabilities

- DDoS bandwidth flooding attack – attack in 2002 to root servers by ICMP messages – avoid by packet filters at root servers & caching at TLD servers.

- DNS queries attack to TLD servers

- Man-in-middle attack – take DNS queries & send bogus replies.

- DNS poisoning – tricking DNS to store bogus records in its cache

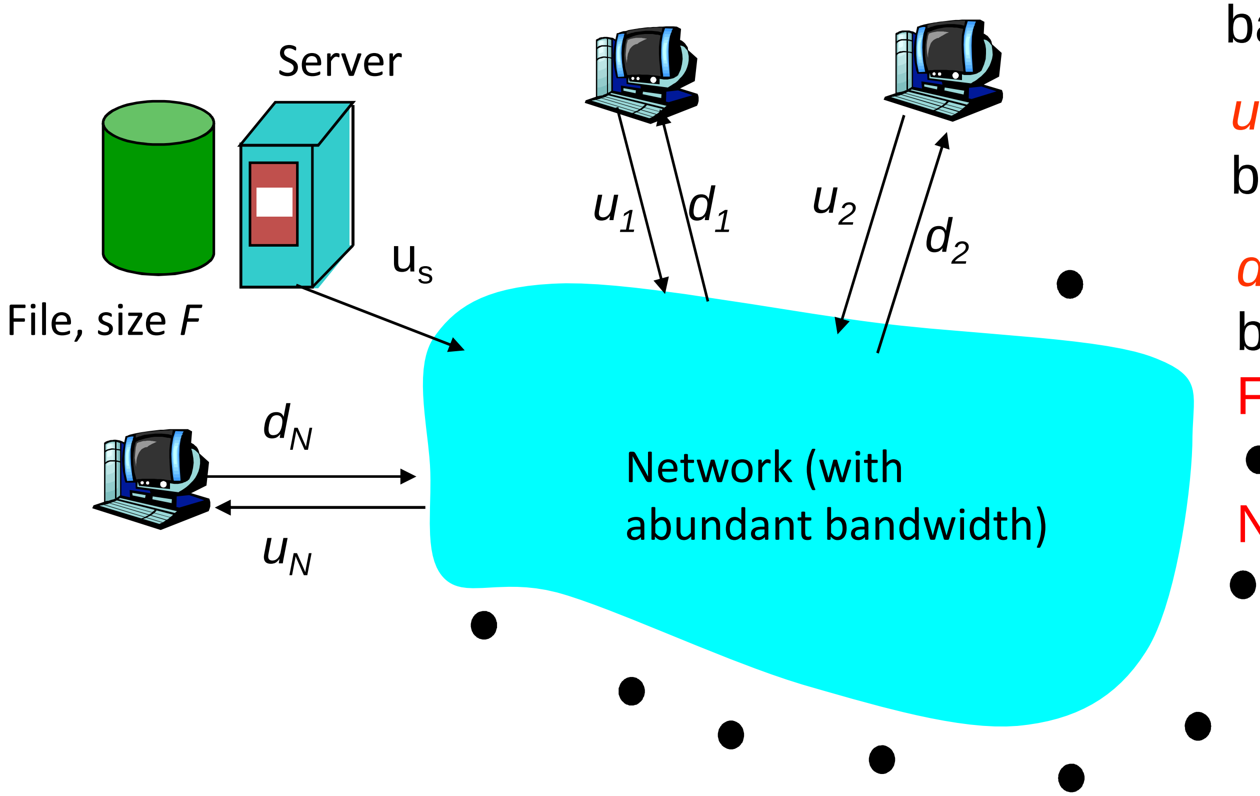
- Using DNS servers to attack a host – spoof the source address of DNS queries with the target host, DNS replies will be sent to target overwhelming it.

Peer to Peer applications

- P2P file distribution
- Bit torrent
- Distributed Hash Table
- P2P internet telephony with skype

File Distribution: Server-Client vs P2P

Question : How much time to distribute file from one server to *N* *peers*?



u_s: server upload bandwidth

u_i: peer i upload bandwidth

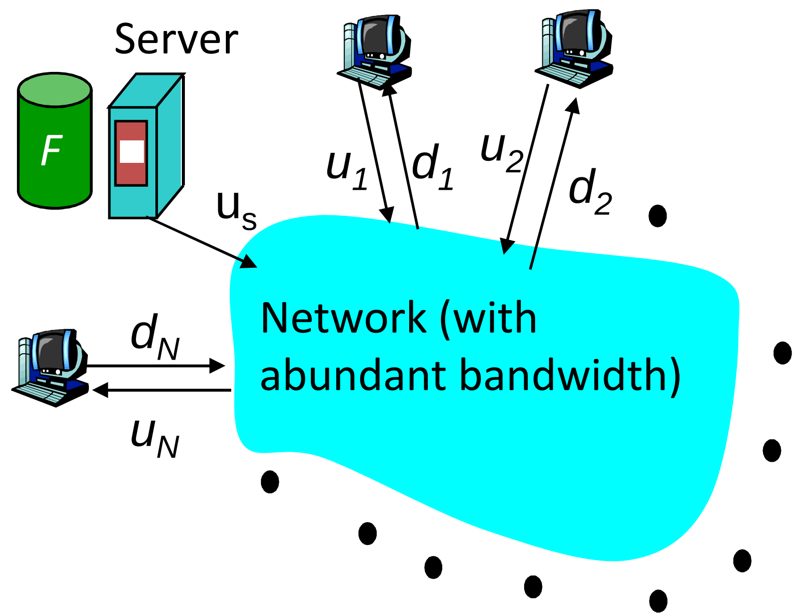
d_i: peer i download bandwidth

F : file size

N: no. of peers

File distribution time: server-client

- server sequentially sends N copies:
  - NF/u_s time
- client i takes F/d_i time to download

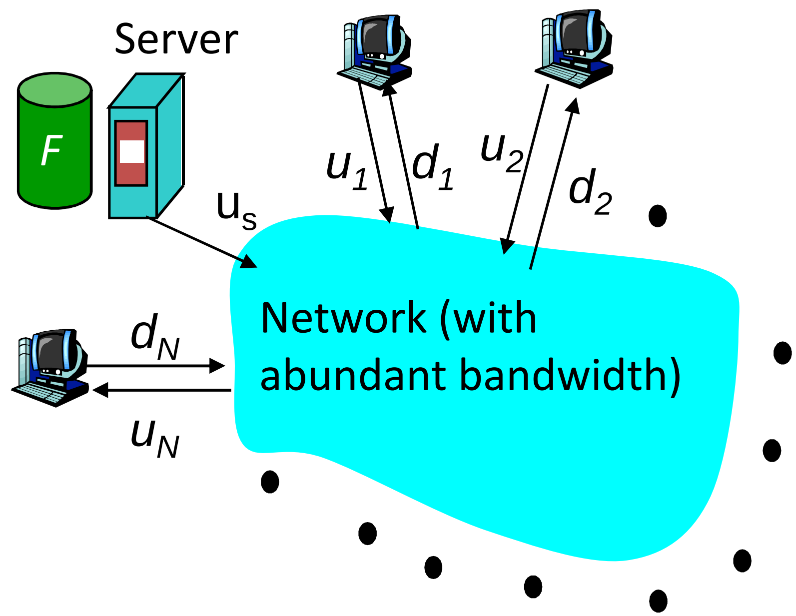


Time to distribute *F* to *N* clients using client/server approach

$$D_{cs} = \max \left\{ \frac{NF}{u_s}, \frac{F}{d_{\min}} \right\}$$

File distribution time: P2P

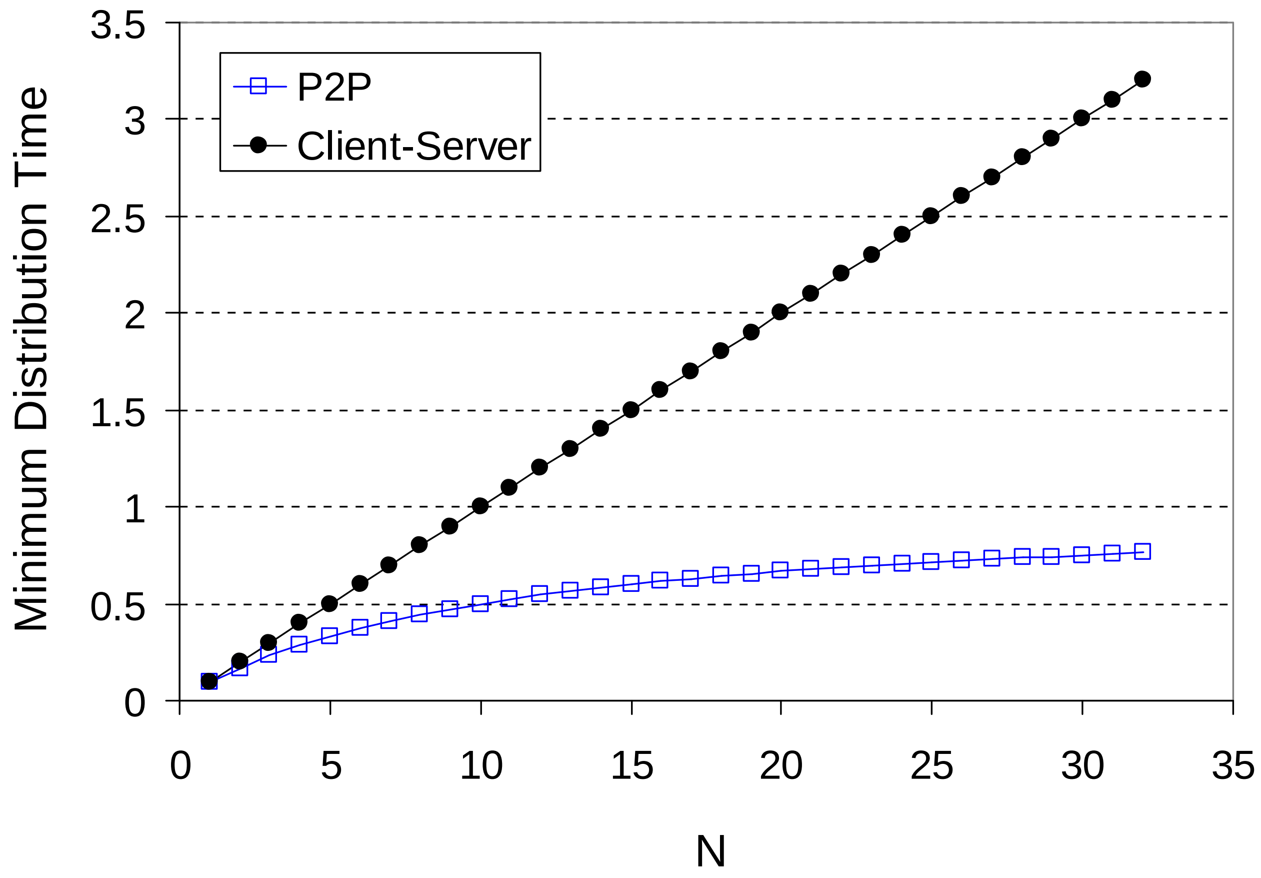
- server must send one copy: F/u_s time
- client i takes F/d_i time to download
- NF bits must be downloaded (aggregate)
  - fastest possible upload rate: $u_s + \sum u_i$



$$D_{P2P} = \max\left\{\frac{F}{u_s}, \frac{F}{d_{min}}, \frac{NF}{u_s + \sum_{i=1}^{N} u_i}\right\}$$

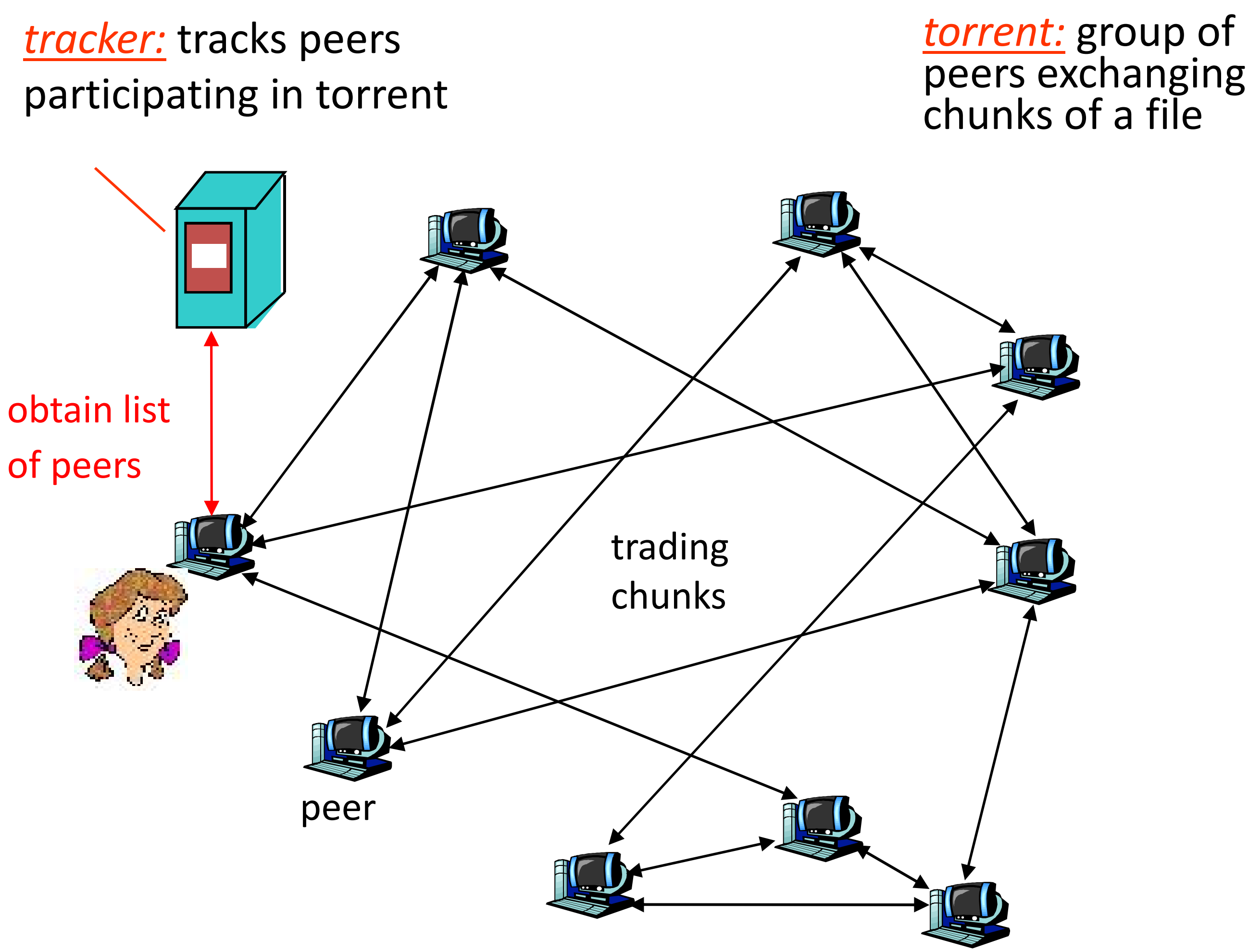
Server-client vs. P2P: example

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

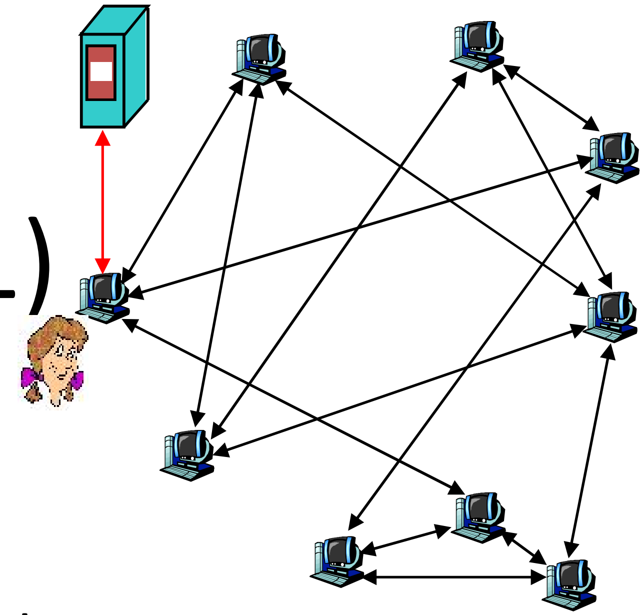


File distribution: BitTorrent

- P2P file distribution



BitTorrent (1)



- file divided into 256KB *chunks*.
- peer joining torrent:
  - has no chunks, but will accumulate them over time
  - registers with tracker to get list of peers, connects to subset of peers (“neighbors”)
- while downloading, peer uploads chunks to other peers.
- peers may come and go
- once peer has entire file, it may (selfishly) leave or (altruistically) remain

BitTorrent (2)

Pulling Chunks

- at any given time, different peers have different subsets of file chunks
- periodically, a peer (Alice) asks each neighbor for list of chunks that they have.
- Alice sends requests for her missing chunks
  - rarest first

Sending Chunks: tit-for-tat

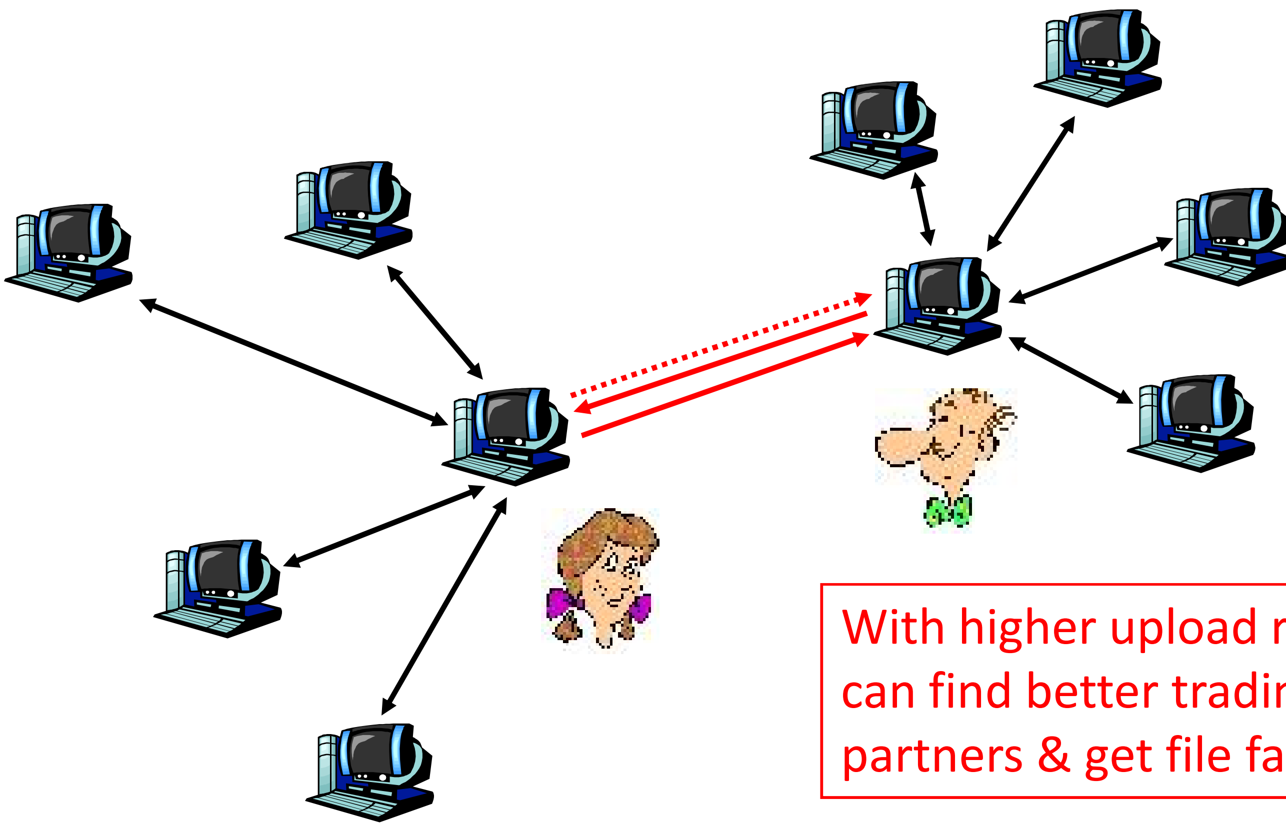
- ❒ Alice sends chunks to four neighbors (unchoked) currently sending her chunks *at the highest rate*
  - ❖ re-evaluate top 4 every 10 secs
- ❒ every 30 secs: randomly select another peer, starts sending chunks
  - ❖ newly chosen peer may join top 4
  - ❖ "optimistically unchoke"

BitTorrent: Tit-for-tat

(1) Alice "optimistically unchokes" Bob

(2) Alice becomes one of Bob's top-four providers; Bob reciprocates

(3) Bob becomes one of Alice's top-four providers



With higher upload rate, can find better trading partners & get file faster!

Distributed Hash Table (DHT)

- DHT = distributed P2P database
- Database has (key, value) pairs;
  - key: ss number; value: human name
  - key: content type; value: IP address
- Peers query DB with key
  - DB returns values that match the key
- Peers can also insert (key, value) peers

Trivial implementation

- Randomly scatter all key-value pairs across all peers
- Maintain ip address of all peers.
- Drawback
  - Unscalable
  - One query is sent to all peers

DHT Implementation

- Assign integer identifier to each peer in range $[0,2^n-1]$.
  - Each identifier can be represented by n bits.
- Require each key to be an integer in same range.
- To get integer keys, hash original key.
  - eg, key = h("Led Zeppelin IV")
  - This is why they call it a distributed "hash" table

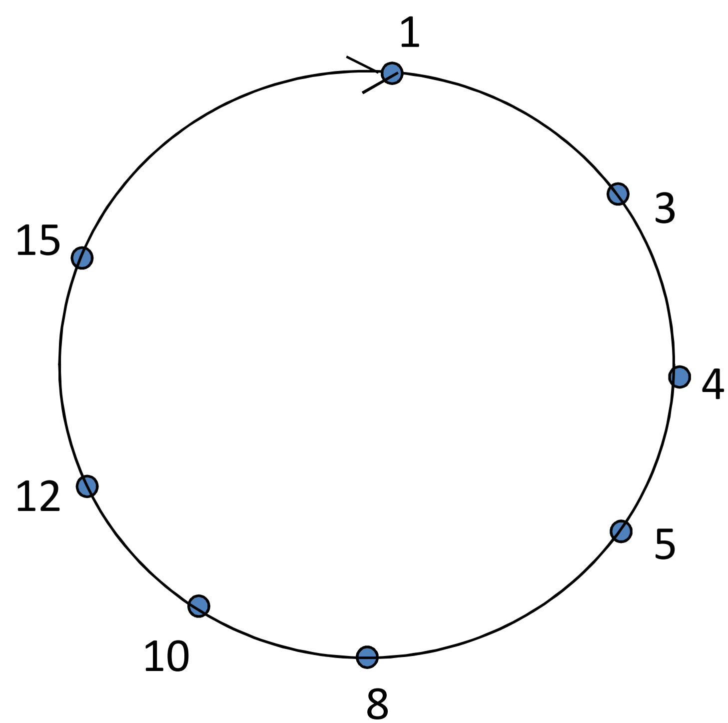
How to assign keys to peers?

- Central issue:
  - Assigning (key, value) pairs to peers.
- Rule: assign key to the peer that has the closest ID.
- closest is the immediate successor of the key.
- Ex: n=4; peers: 1,3,4,5,8,10,12,14;
  - key = 13, then successor peer = 14
  - key = 15, then successor peer = 1

Drawback

Should keep tack of all peer ids and its ip addresses to find nearest peer – not scalable

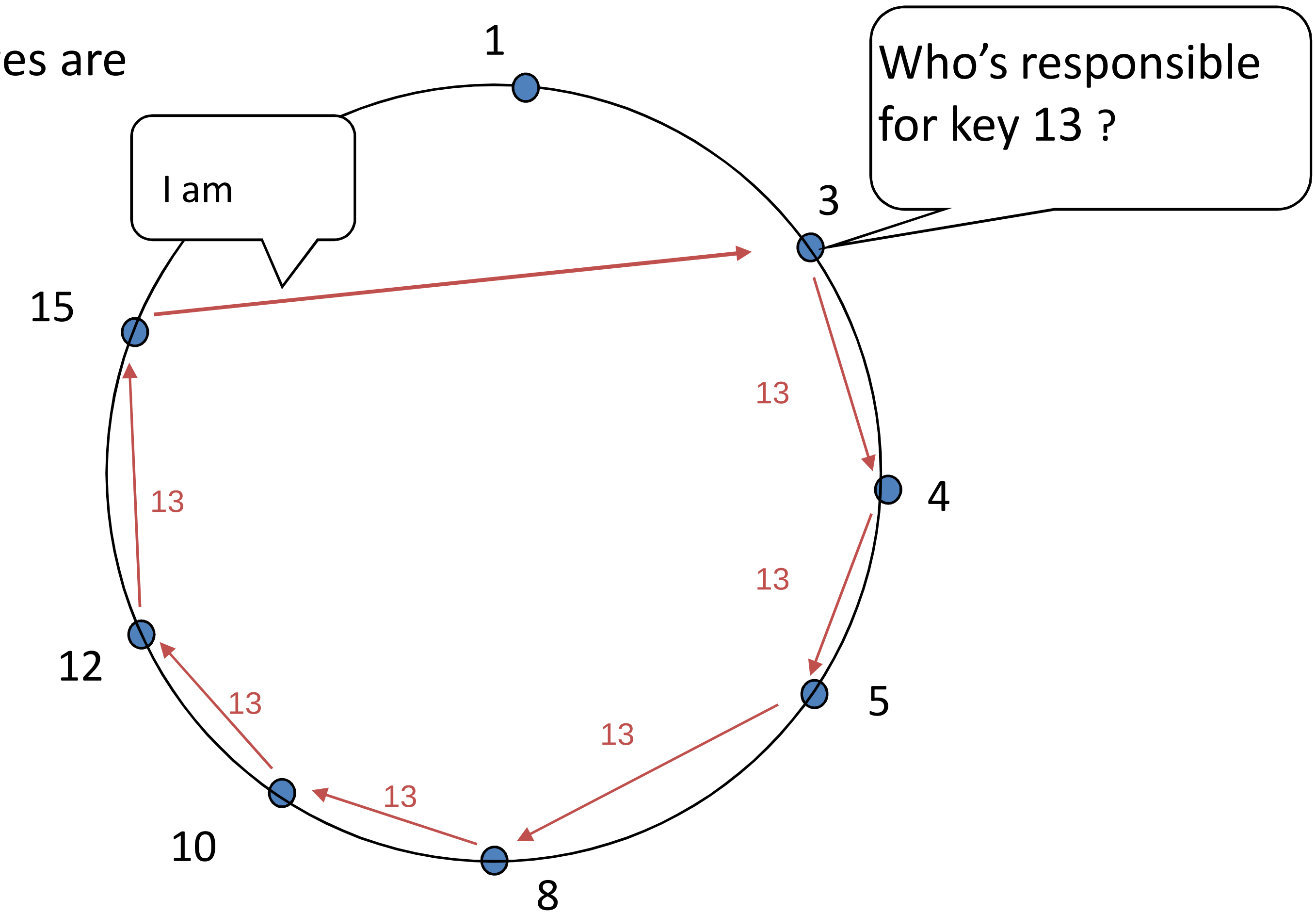
Circular DHT (1)



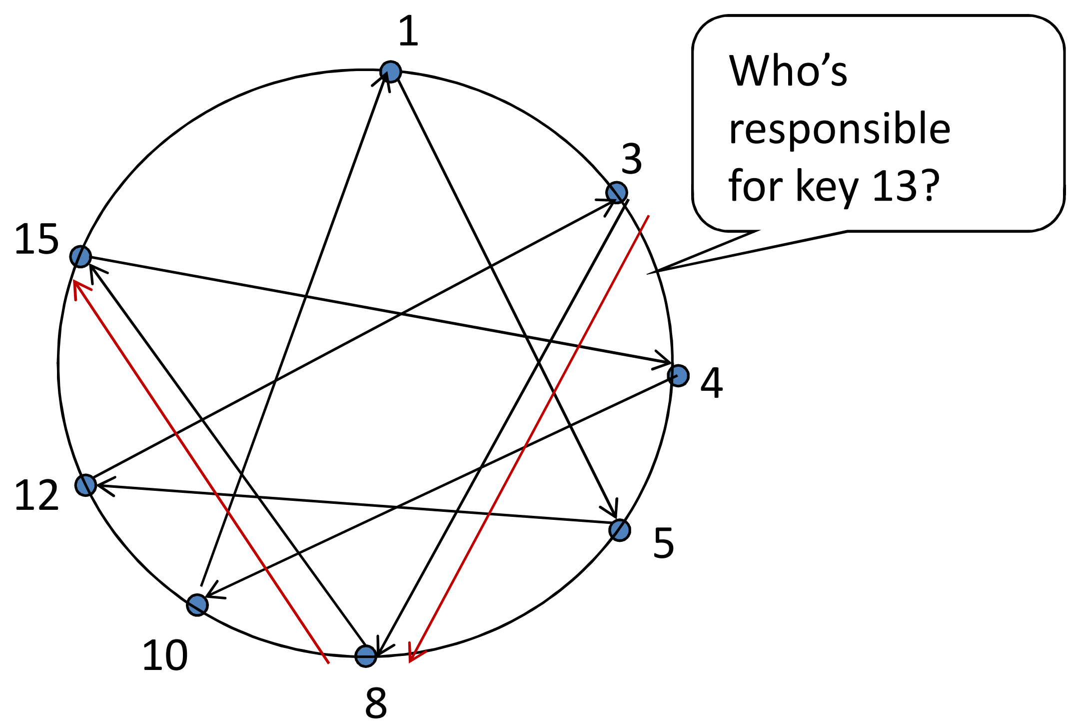
- Each peer *only* aware of immediate successor and predecessor.
- “Overlay network”

Circular DHT (2)

On avg N/2 messages are sent to resolve query, when there are N peers

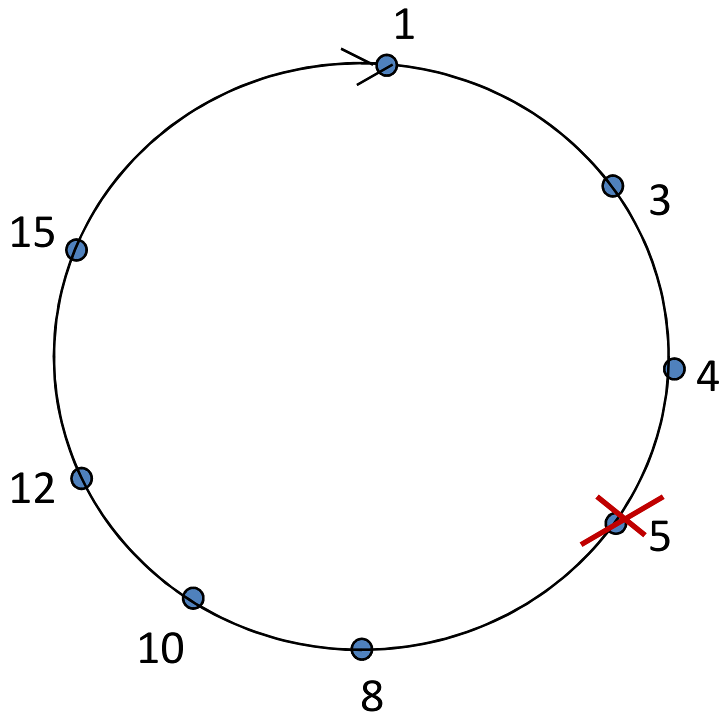


Circular DHT with Shortcuts



- Each peer keeps track of IP addresses of predecessor, successor, short cuts.
- Reduced from 6 to 2 messages.
- How to choose shortcuts? How many shortcuts? - Research
- Possible to design shortcuts such that no. of peers and no. of messages per query is O(log N)

Peer Churn (1)



- To handle peer churn, require each peer to know the IP address of its two successors.
- Each peer periodically pings its two successors to see if they are still alive.

- Peer 5 abruptly leaves
- Peer 4 detects; makes 8 its immediate successor; asks 8 who its immediate successor is; makes 8's immediate successor its second successor.

Peer Churn (2)

- Suppose peer 13 wants to join the network and knows only peer 1.
- The request is sent to peer 1 which forwards till peer 12 which identifies the location for 13 (peer 12 first successor is 15).
- Peer 12 makes 13 as its first successor and 15 as its second successor
- Peer 15 makes 13 as its predecessor
- Peer 13 makes 12 as its predecessor, 15 as its first successor and gets information of its second successor (peer 1) from peer 15.

END of Unit 1