Application Layer

Raj Jain
Washington University in Saint Louis
Saint Louis, MO 63130
Jain@wustl.edu

Audio/Video recordings of this lecture are available on-line at:

http://www.cse.wustl.edu/~jain/cse473-20/

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

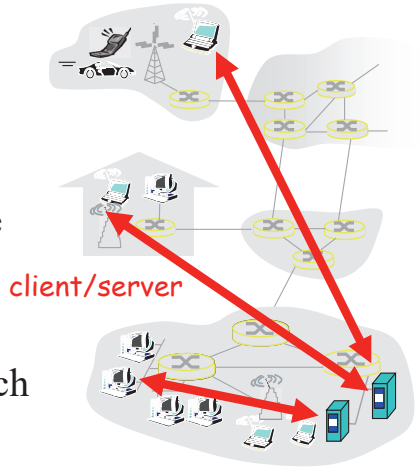
<table>
<tr><td>Application</td><td>HTTP</td><td>FTP</td><td>SMTP</td><td>P2P</td><td>DNS</td><td>Skype</td></tr>
<tr><td>Transport</td><td colspan="5">TCP</td><td>UDP</td></tr>
<tr><td>Internetwork</td><td colspan="6">IP</td></tr>
<tr><td>Host to Network</td><td colspan="2">Ethernet</td><td colspan="2">Point-to-Point</td><td colspan="2">Wi-Fi</td></tr>
<tr><td>Physical</td><td>Coax</td><td colspan="2">Fiber</td><td colspan="3">Wireless</td></tr>
</table>

Network Application Architectures

- Client-Server
- Peer-to-Peer

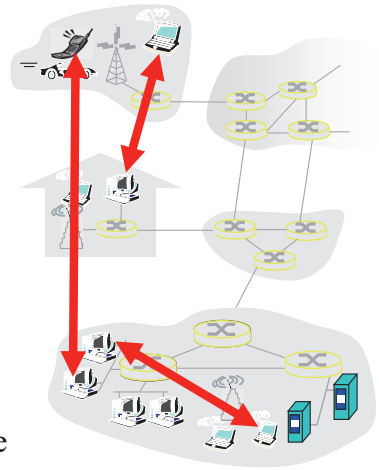
Client-Server

- 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



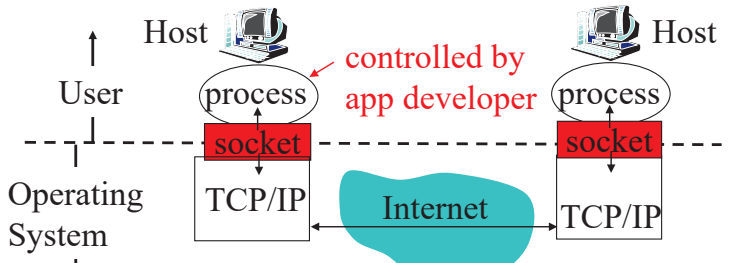
Peer-to-Peer

- 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



Names, Addresses, Ports

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

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 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

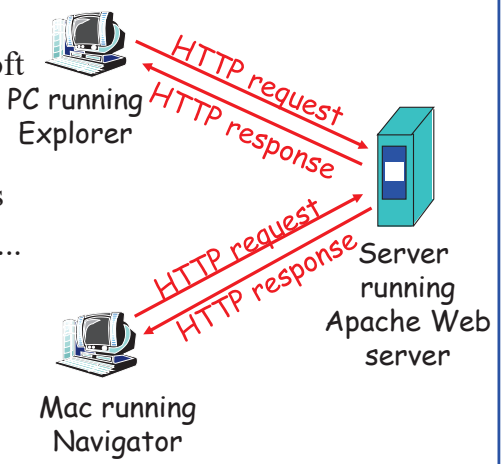
Ref: Read Section 2.1 full. Try R1-R9

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.cse.wustl.edu/~jain/cse473-09/sample.htm



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=jain.jpg>
<BR>
Raj Jain
</BODY>
</HTML>
```

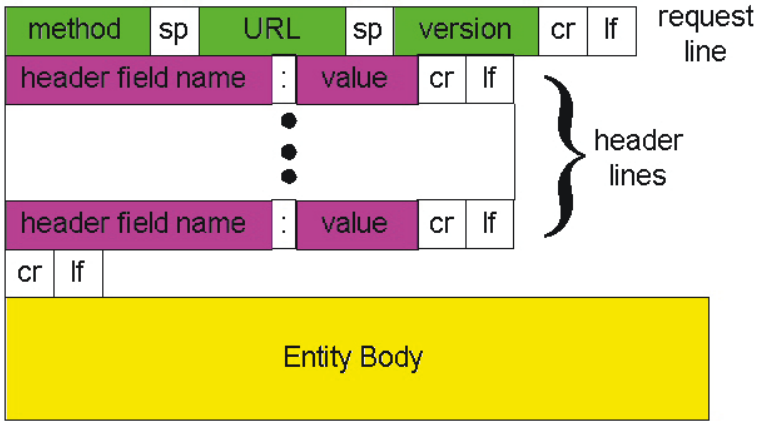

Raj Jain

Sample HTTP Request Message

GET /~jain/cse473-16/sample.htm HTTP/1.1
Host: www.cse.wustl.edu
Connection: close
User-agent: Mozilla/4.0
Accept-Language: en

- **Method** = Get
- **URL** = /~jain/cse473-16/sample.htm
- **Version** = HTTP/1.1
- **Header Fields** = Host, Connection, User-agent, …

HTTP Request Message Format



Sample HTTP Response Message

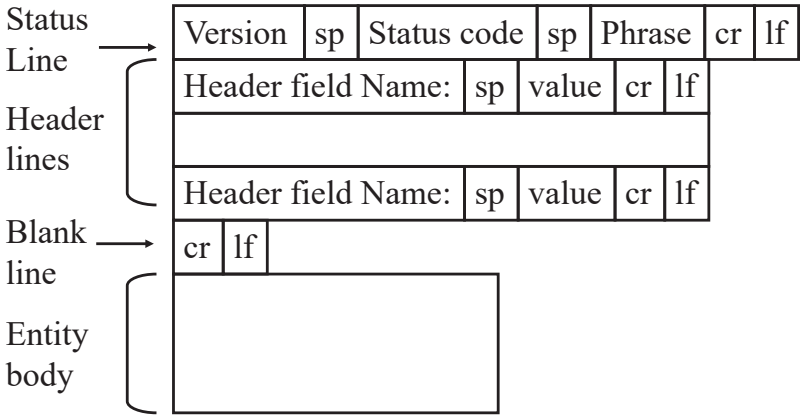
HTTP/1.1 200 OK
Connection: close
Date: Tue, 09 Sept 2009 13:00:15 GMT
Server: Apache/1.3.0 (Unix)
Last-Modified: Sun, 6 May 2009 09:23:24 GMT
Content-Length: 6500
Content-Type: Text/html

Data…

Status Codes:

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

HTTP Response Message Format



Hands-on HTTP

telnet www1.cse.wustl.edu 80
GET /~jain/cse473-19/sample.htm HTTP/1.1
Host: www1.cse.wustl.edu

HTTP/1.1 200 OK
Date: Tue, 13 Sep 2019 23:39:53 GMT
Server: Apache/2.2.3 (CentOS)
Accept-Ranges: bytes
Content-Length: 233
Connection: close
Content-Type: text/html; charset=ISO-8859-1

<HTML>
<head>
</head>
<body>
This is a sample text.
</body>
</html>

This is a sample text.

Hands-on HTTP (cont)

telnet www1.cse.wustl.edu 80
GET /~jain/cse473-08/sample.htm HTTP/1.1
Host: www1.cse.wustl.edu

HTTP/1.1 404 Not Found
Date: Tue, 13 Sep 2019 23:42:48 GMT
Server: Apache/2.2.3 (CentOS)
Content-Length: 307
Connection: close
Content-Type: text/html; charset=iso-8859-1

<!DOCTYPE HTML PUBLIC "-//IETF//DTD HTML 2.0//EN">
<html><head>
<title>404 Not Found</title>
</head><body>
<h1>Not Found</h1>
The requested URL /~jain/cse473-08/sample.htm was not found on this server.
<hr>
<address>Apache/2.2.3 (CentOS) Server at www1.cse.wustl.edu Port 80</address>
</body></html>

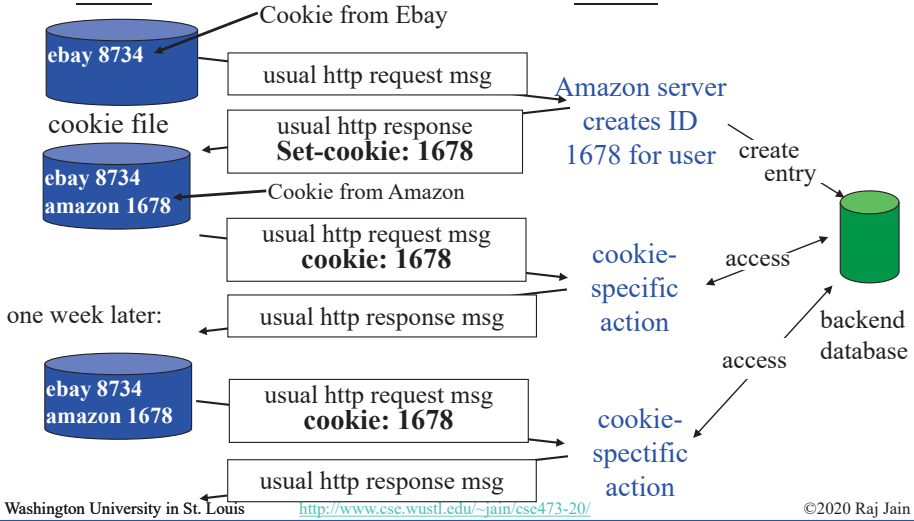
Not Found

The requested URL /~jain/cse473-08/sample.htm was not found on this server.

Apache/2.0.52 (CentOS) Server at www.cse.wustl.edu Port 80

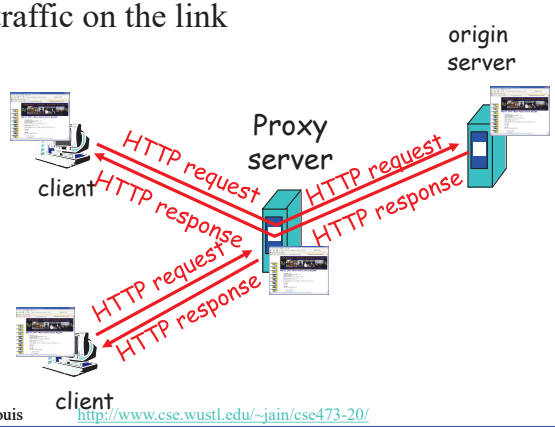
Cookies

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

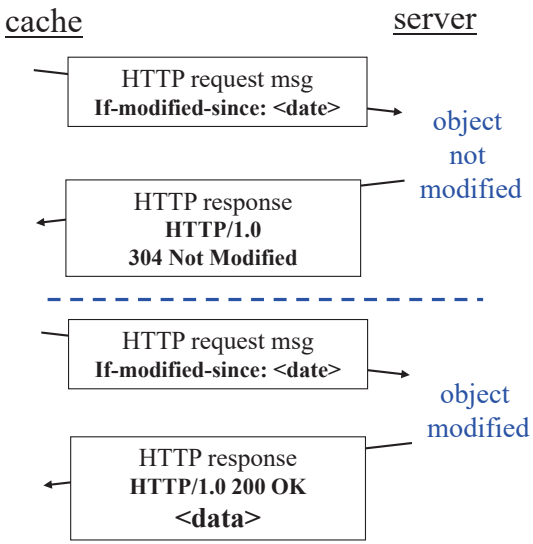


Proxy Server: Web Caching

- 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



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

Ref: Read Section 2.2 Full. Try R10-R14.

Homework 2A: HTTP

[10 points] The text below shows the reply sent from the server in response to the HTTP GET message. Answer the following questions, indicating where in the message below you find the answer.

HTTP/1.1 200 OK
Date: Tue, 07 Mar 2019 12:39:45GMT
Server: Apache/2.0.52 (Fedor)
Last-Modified: Sat, 5 Jan 2019 18:27:46 GMT
Etag: "526c3-f22-a88a4c80"
Accept-ranges: bytes
Content-Length: 4071
Keep-Alive: timeout=max=100
Connection: Keep-Alive
Content-Type: text/html; charset=ISO-8859-1

<!doctype html publi "-//w3c//dtd html 4.0 transitional//en">
<html>
<head>
<much more document text following here (not shown)>

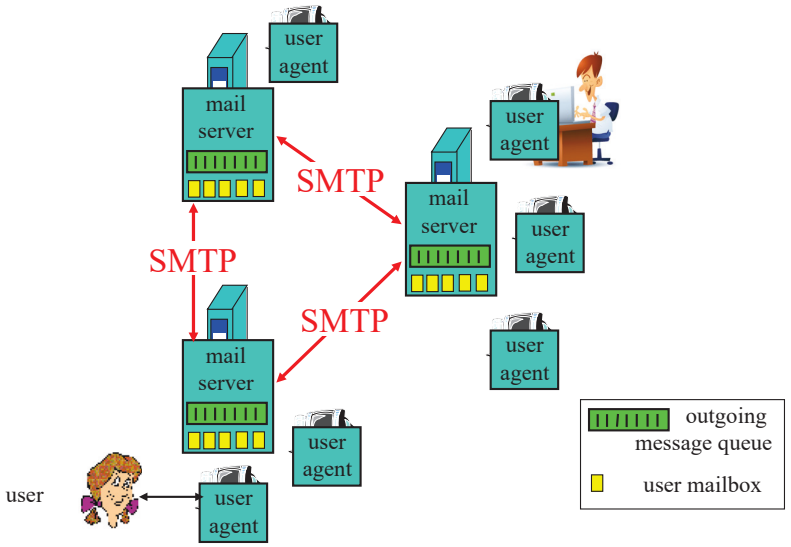
A. Was the server able to successfully find the document or not? What time was the document reply provided?
B. When was the document last modified?
C. How many bytes are there in the document being returned?
D. What are the first 5 bytes of the document being returned?
E. Did the server agree to a persistent connection?

Lab 2A: Domains

[10 points] Submit answers for the following: (See hints in the parenthesis.)

1. Find the IP addresses of www.google.com and www.yahoo.com (ping)
2. Modify the hosts file to map www.google.com to yahoo's IP address and ping to www.google.com. Notice what address it is pinging to. Remove the modification to the host file, open a new command window and repeat. (Windows: c:\windows\system32\drivers\etc\hosts)
3. Find the domain name and country of 128.252.165.7 (http://www.webyield.net/domainquery.html)
4. Find the owner of wustl.edu domain (http://www.networksolutions.com/whois/index.jsp)
5. Find the name server of wustl.edu domain (http://www.networksolutions.com/whois/index.jsp)

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

Sample SMTP Exchange

C: telnet mail.seas.wustl.edu 25
S: 220 POSTOFFICE.seas.wustl.edu Microsoft ESMTP MAIL Service, Version: 6.0.3790.46 75 ready at Tue, 13 Sep 2011 18:34:56 -0500
C: HELO acm.org
S: 250 POSTOFFICE.seas.wustl.edu Hello [128.252.19.232]
C: MAIL FROM: jain@acm.org
S: 250 2.1.0 jain@acm.org....Sender OK
C: RCPT TO: jain@wustl.edu
S: 250 2.1.5 jain@wustl.edu
C: DATA
S: 354 Start mail input; end with <CRLF>.<CRLF>
C: This is test email.
This serves as an exmaple for CS473 class.
.
S: 250 2.6.0 <MAIL2j97vPYGrN7kf0V00000aff@POSTOFFICE.seas.wustl.edu> Queued mail for delivery
C: QUIT
S: 221 2.0.0 POSTOFFICE.seas.wustl.edu Service closing transmission channel

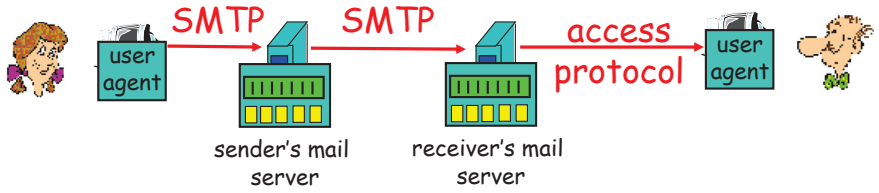
Try the above client sequence by *telnet mail.seas.wustl.edu 25*

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



POP3 protocol

Authorization phase

S: +OK POP3 server ready
C: user bob
S: +OK
C: pass hungry
S: +OK user successfully logged on

Transaction phase

C: list
S: 1 498
S: 2 912
S: .
C: retr 1
S: <message 1 contents>
S: .
C: dele 1
C: retr 2
S: <message 2 contents>
S: .
C: dele 2
C: quit
S: +OK POP3 server signing off

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

Mail: Summary

1. SMTP is the protocol to **send** email
2. SMTP uses only **7-bit ASCII** messages
3. POP3, IMAP, or HTTP is used to **receive** email

Ref: Read Section 2.3 Full. Try R15-R20

Homework 2B: Mail

[12 points] Consider accessing your e-mail with POP3.

a) Suppose you have configured your POP mail client to operate in the download and delete mode. Complete the following transaction to retrieve both messages, and sign off. Show the complete sequence of messages. (Fill in ? and successive messages)

C: list
S: 1 500
S: 2 901
S: .
C: retr 1
S: blah blah …
S: … Blah
S: .
?
?

b) Repeat part a if you have programmed your POP client in download and keep mode.

c) Suppose five minutes later you again access POP to retrieve new e-mail. Suppose that in the five-minute interval no new message have been sent to you. Provide a transcript of this second POP session for both options a and b above.

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.wustl.edu ⇒ 128.252.87.149
- Distributed database of all hosts in the universe
- Other Services:
  - **Host Aliasing**: www.rajjain.com or www.cse.wustl.edu/~jain/
  - **Mail Server Aliasing**: MX record (e.g., jain@wustl.edu)
  - **Load Distribution**: Multiple addresses, rotated

DNS Example

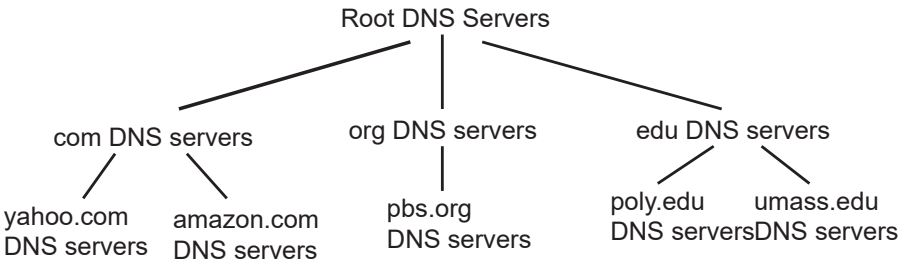
F:\>**nslookup www.wustl.edu**
Server: ns00.ip.wustl.edu
Address: 128.252.0.1

Name: www.wustl.edu
Address: 128.252.87.149

F:\>**nslookup www.google.com**
Server: ns00.ip.wustl.edu
Address: 128.252.0.1

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

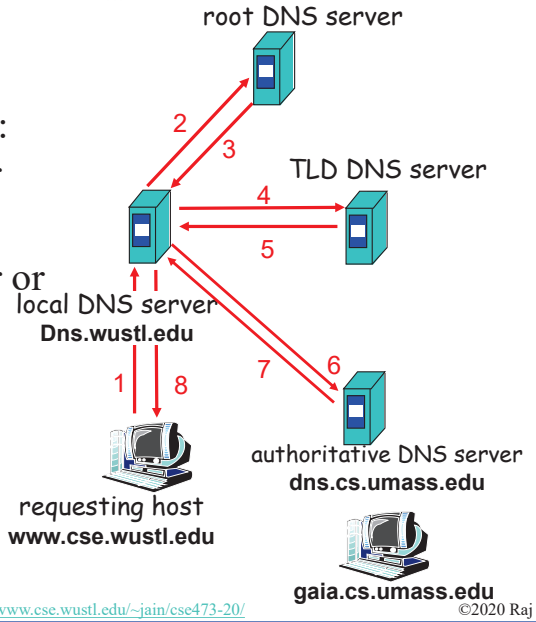
DNS Hierarchy



- Root DNS Servers
- 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



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

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 Registration

- Many Registrars
- Internet Corporation for Assigned Names and Numbers (ICANN) accredits registrars
- www.internic.net
- Registrars provide authoritative name servers, A and MX records for the domain

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

Ref: Read Section 2.4 Full.

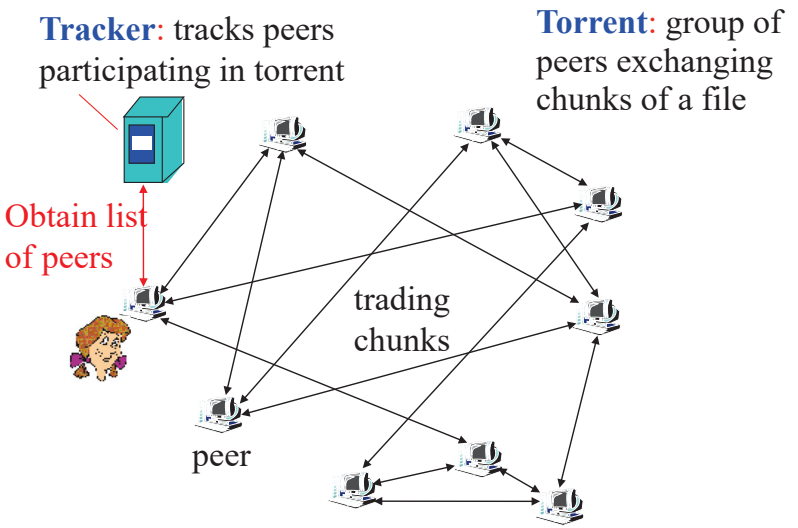
Homework 2C: DNS

- [4 points]
- Is it possible for an organization's web server and mail server to have exactly the same hostname? (Briefly explain why or why not?)
- What would be the type of RR that contains the host name of the mail server?

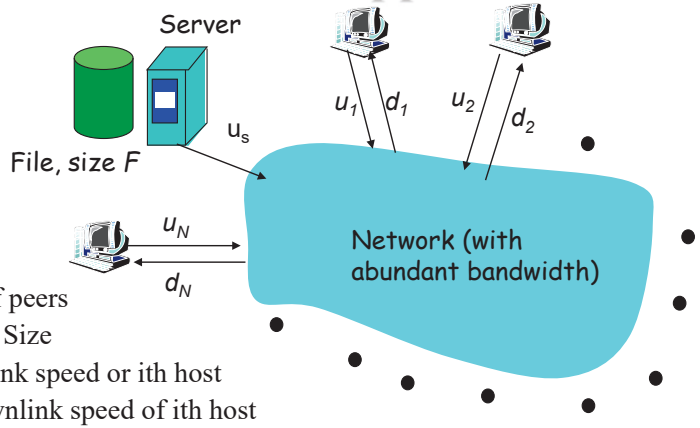
Peer-to-Peer Applications

1. Client Server vs. P2P Scalability
2. P2P File Distribution (BitTorrent)

P2P File Distribution (BitTorrent)



Peer-to-Peer Applications

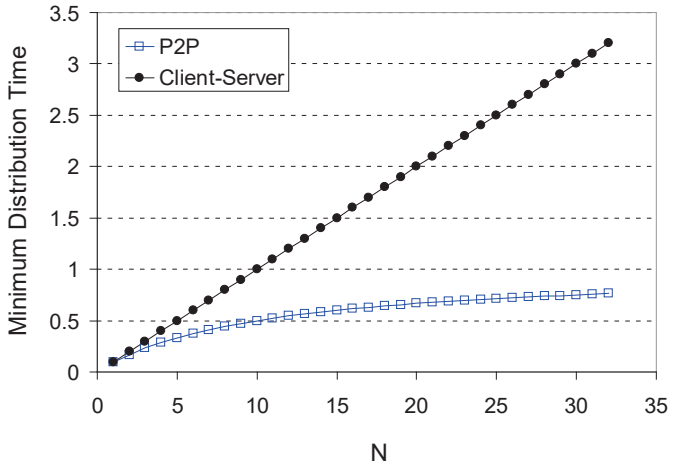


- N= # of peers
- F= File Size
- u_i= uplink speed or ith host
- d_i= downlink speed of ith host
- $d_{min}=\min\{d_1, d_2, ..., d_N\}$
- $D_{cs} \geq \max\{ NF/u_s, F/d_{min}\}$
- $D_{P2P} \geq \max\{F/u_s, F/d_{min}, NF/(u_s+\Sigma u_i)\}$

Server, Client, Network

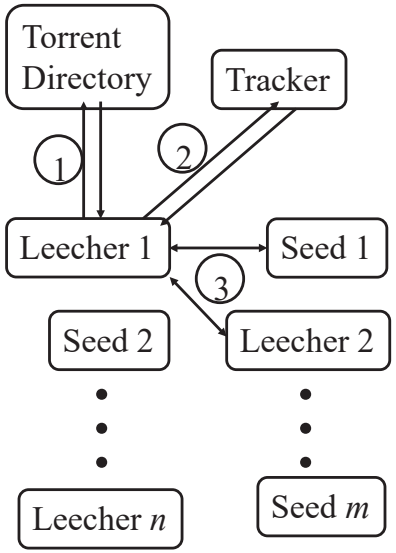
Client Server vs. P2P Scalability

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)

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**

Ref: Read Section 2.5 full. Try R21-R23.

Homework 2D: P2P

[4 points] P26. Suppose Bob joins a BitTorrent torrent, but he does not want to upload any data to any other peers (so called free-riding).

A. Bob claims that he can receive a complete copy of the file that is shared by the swarm. Is Bob's claim possible? Why or Why not?

B. Bob further claims that he can further make his "free-riding" more efficient by using a collection of multiple computers (with distinct IP addresses) in the computer lab in his department. How can he do that?

Streaming Video

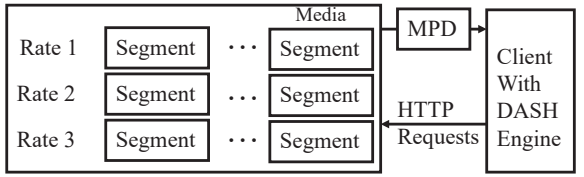
- Video traffic is 80% of consumer traffic
- Video: 25-30 Frames/sec
- Video can be compressed:
  - Spatial: next pixel is similar to this

  - Temporal: Pixel in the next frame is similar to this
- Variable bit rate (VBR)/Constant bit rate (CBR)
  - Motion Picture Expert Group (MPEG) 1: 1.5 Mbps
  - MPEG2: 3-6 Mbps
  - MPEG4 (.mp4): Less than 1 Mbps

Ref: Cisco Visual Networking Index: Forecast and Methodology, 2014-2019 White Paper, http://www.cisco.com/c/en/us/solutions/collateral/service-provider/ip-ngn-ip-next-generation-network/white_paper_c11-481360.html

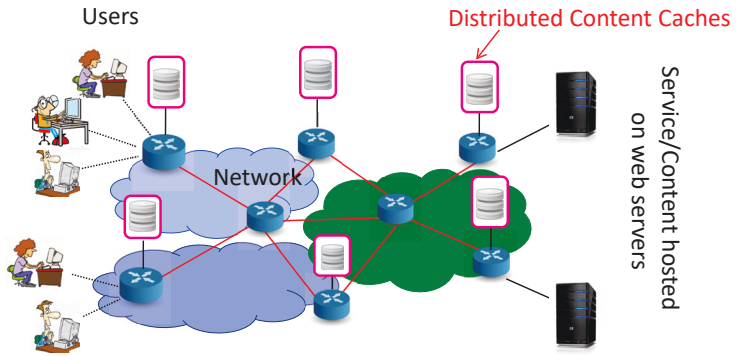
Dynamic Adaptive Streaming over HTTP (DASH)

- DASH provides an efficient method for video streaming
- Standard Web Servers: No changes required to servers, Content Distribution Networks (CDN), or HTTP protocol.
- Mobile client controls what is downloaded using a "**media presentation description (MPD)**" file defined by DASH
- MPD contains URLs for segments
- Client measures throughput and requests segments as needed. Allows fast forward, rewind, etc.



Content Distribution Networks (CDN)

- To reduce latency to worldwide users, the data is replicated at many sites
- Users are directed to nearby site by DNS
- netflix.com -> cdn_stl.com or cdn_sfo.com, …



Homework 2E: DASH

- [2 points] A DASH system stores video at 5 different qualities (rates) and 15 minute segments. How many URLs will be required for a 2-hour movie?

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
6. **DASH** allows clients to request different video segments as needed
7. **CDN**'s directs users to to nearby copy via DNS

Ref: In addition to previous readings, read Sections 2.6.1-2.6.3. Try R24-R25.

Lab 2B: UDP Pinger

- [50 points] In this lab, you will learn the basics of socket programming for UDP in Python. You will learn how to send and receive datagram packets using UDP sockets and also, how to set a proper socket timeout. Throughout the lab, you will gain familiarity with a Ping application and its usefulness in computing statistics such as packet loss rate.
- You will first study a simple Internet ping server written in the Python, and implement a corresponding client. The functionality provided by these programs is similar to the functionality provided by standard ping programs available in modern operating systems. However, these programs use a simpler protocol, UDP, rather than the standard Internet Control Message Protocol (ICMP) to communicate with each other. The ping protocol allows a client machine to send a packet of data to a remote machine, and have the remote machine return the data back to the client unchanged (an action referred to as echoing). Among other uses, the ping protocol allows hosts to determine round-trip times to other machines.
- You are given the complete code for the Ping server below. Your task is to write the Ping client.

Lab 2B (Cont)

Server Code

The following code fully implements a ping server. You need to compile and run this code before running your client program. *You do not need to modify this code.*

In this server code, 30% of the client's packets are simulated to be lost. You should study this code carefully, as it will help you write your ping client.

```
# UDPPingerServer.py
# We will need the following module to generate randomized lost packets
import random
from socket import *
# Create a UDP socket
# Notice the use of SOCK_DGRAM for UDP packets
serverSocket = socket(AF_INET, SOCK_DGRAM)
# Assign IP address and port number to socket
serverSocket.bind(('', 12000))
```


Lab 2B (Cont)

```
while True:
# Generate random number in the range of 0 to 10
rand = random.randint(0, 10)
# Receive the client packet along with the address it is coming from
message, address = serverSocket.recvfrom(1024)
# Capitalize the message from the client
message = message.upper()
# If rand is less is than 4, we consider the packet lost and do not respond
if rand < 4:
continue
# Otherwise, the server responds
serverSocket.sendto(message, address)
```

The server sits in an infinite loop listening for incoming UDP packets. When a packet comes in and if a randomized integer is greater than or equal to 4, the server simply capitalizes the encapsulated data and sends it back to the client.

Lab 2B (Cont)

Packet Loss

UDP provides applications with an unreliable transport service. Messages may get lost in the network due to router queue overflows, faulty hardware or some other reasons. Because packet loss is rare or even non-existent in typical campus networks, the server in this lab injects artificial loss to simulate the effects of network packet loss. The server creates a variable randomized integer which determines whether a particular incoming packet is lost or not.

Client Code

You need to implement the following client program.

The client should send 10 pings to the server. Because UDP is an unreliable protocol, a packet sent from the client to the server may be lost in the network, or vice versa. For this reason, the client cannot wait indefinitely for a reply to a ping message. You should get the client wait up to one second for a reply; if no reply is received within one second, your client program should assume that the packet was lost during transmission across the network. You will need to look up the Python documentation to find out how to set the timeout value on a datagram socket.

Lab 2B (Cont)

Specifically, your client program should

(1) send the ping message using UDP (Note: Unlike TCP, you do not need to establish a connection first, since UDP is a connectionless protocol.)

(2) print the response message from server, if any

(3) calculate and print the round trip time (RTT), in seconds, of each packet, if server responses

(4) otherwise, print "Request timed out"

During development, you should run the UDPPingerServer.py on your machine, and test your client by sending packets to *localhost* (or, **127.0.0.1**). After you have fully debugged your code, you should see how your application communicates across the network with the ping server and ping client running on different machines.

Message Format

The ping messages in this lab are formatted in a simple way. The client message is one line, consisting of ASCII characters in the following format:

Ping *sequence_number time*

where *sequence_number* starts at 1 and progresses to 10 for each successive ping message sent by the client, and *time* is the time when the client sends the message.

Lab 2B (Cont)

What to Hand in

You will hand in the complete client code and screenshots at the client verifying that your ping program works as required.

Reading List

- Read Chapter 3 of the textbook for the next lecture.

Acronyms

- ASCII American Standard Code for Information Interchange
- CBR Constant bit rate
- CDN Content Distribution Network
- DASH Dynamic Adaptive Streaming
- DNS Domain Name System
- FTP File Transfer Protocol
- GMT Greenwich Mean Time
- HTML Hyper-Text Markup Language
- HTTP Hyper-Text Transfer Protocol
- ICANN International Corporation for Assigned Names and Numbers
- ID Identifier
- IMAP Internet Message Access Protocol
- IP Internet Protocol
- ISO International Standards Organization
- ISP Internet Service Provider
- kB Kilo Byte

Acronyms (Cont)

- MPD Media Presentation Description
- MPEG Moving Picture Expert Group
- NAT Network Address Translator
- NS Name Service
- PC Personal Computer
- POP Point of Presence
- RR Resource Record
- SMTP Simple Mail Transfer Protocol
- TCP Transmission Control Protocol
- TLD Top Level Domain
- TTL Time to Live
- UDP Universal Data Protocol
- URL Uniform Resource Locator
- VBR Variable bit rate

Scan This to Download These Slides

Raj Jain
http://rajjain.com

http://www.cse.wustl.edu/~jain/cse473-20/i_2app.htm

Related Modules

CSE 567: The Art of Computer Systems Performance Analysis
https://www.youtube.com/playlist?list=PLjGG94etKypJEKjNAa1n_1X0bWWNyZcof

CSE473S: Introduction to Computer Networks (Fall 2011),
https://www.youtube.com/playlist?list=PLjGG94etKypJWOSPMh8Azcgy5e_10TiDw

CSE 570: Recent Advances in Networking (Spring 2013)
https://www.youtube.com/playlist?list=PLjGG94etKypLHyBN8mOgwJLHD2FFIMGq5

CSE571S: Network Security (Spring 2011),
https://www.youtube.com/playlist?list=PLjGG94etKypKvzfVtutHcPFJXumyyg93u

Video Podcasts of Prof. Raj Jain's Lectures,
https://www.youtube.com/channel/UCN4-5wzNP9-ruOzQMs-8NUw