CS 305: Computer Networks

Fall 2024

Lecture 5： Application Layer

Ming Tang

Department of Computer Science and Engineering
Southern University of Science and Technology (SUSTech)

Chapter 2: outline

2.1 principles of network applications

2.2 Web and HTTP

2.3 electronic mail

- SMTP, POP3, IMAP

2.4 DNS

2.5 P2P applications

2.6 video streaming and content distribution networks

2.7 socket programming with UDP and TCP

DNS: domain name system

People: many identifiers:
- SSN, name, passport #

Internet hosts, routers:
- hostname, e.g., www.yahoo.com - used by humans
- IP address (32 bit) - used for addressing datagrams

Q: how to map between IP address and name, and vice versa ?

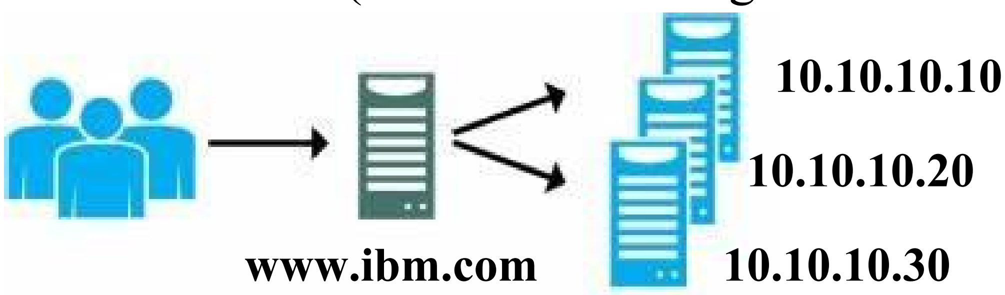
Domain Name System (DNS):
- distributed database implemented in hierarchy of many *name servers*
- application-layer protocol: hosts and name servers communicate to *resolve* names (address/name translation)

DNS Overview

- DNS Services
- DNS Structure
  - Hierarchical structure
  - Iterated and recursive query
- DNS protocol
  - DNS Records
  - Query and reply messages
- Inserting records into DNS

DNS Services

- **hostname to IP address translation**
- **host aliasing**
  - canonical, alias hostnames
  - **www.ibm.com** (alias) is really **servereast.backup2.ibm.com** (canonical)
  - From supplied alias hostname to canonical hostname
- **mail server aliasing**
- **load distribution**
  - replicated Web servers: many IP addresses correspond to one name
  - rotation distributes the traffic (rotate the ordering of IP addresses)



DNS Services

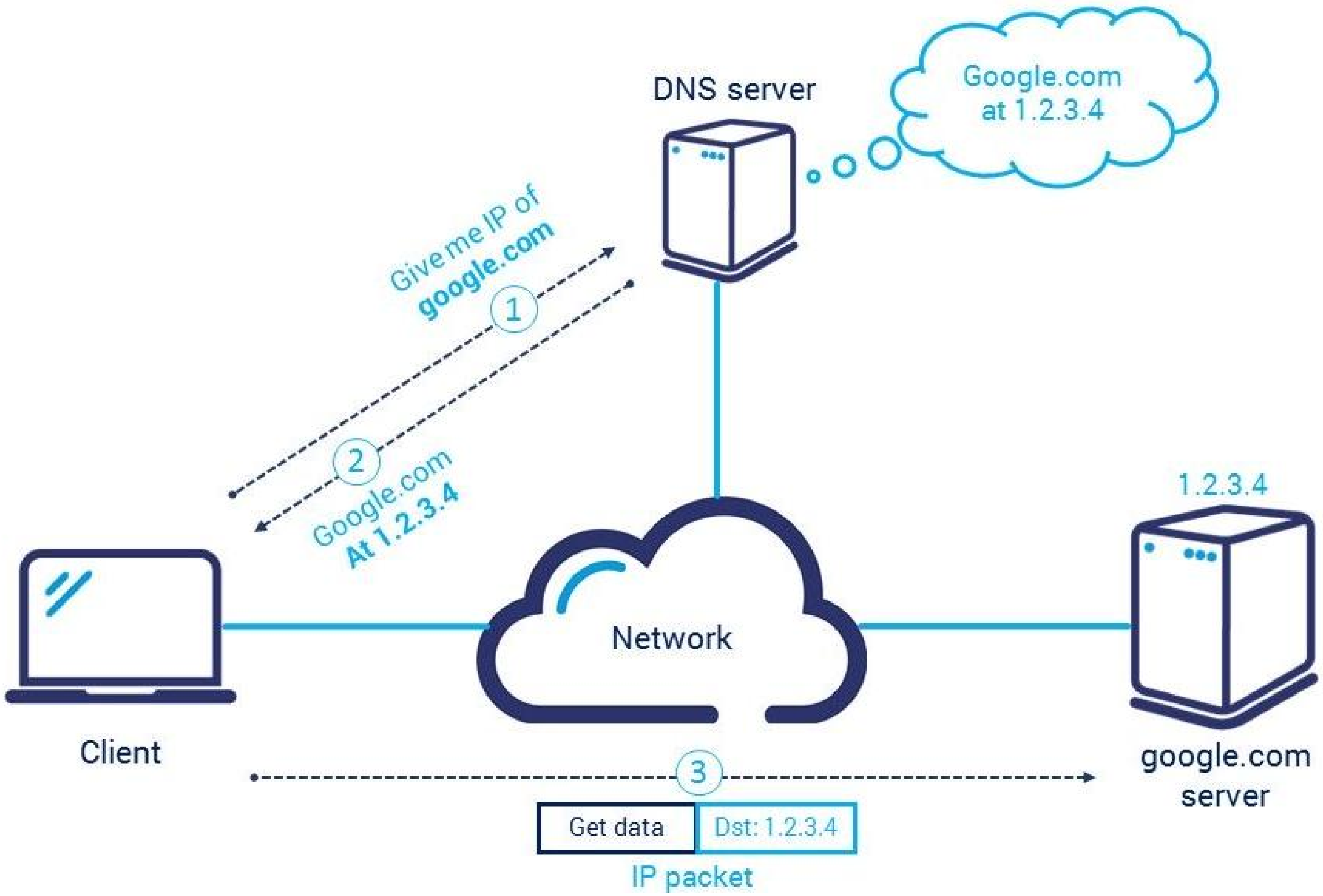
1. An application invokes the client side of DNS
   - specifying the hostname that needs to be translated
2. DNS in the user's host takes over, sending a query message into the network.
   - DNS query and reply messages
   - UDP datagrams to port 53.
3. After a delay, ranging from milliseconds to seconds, DNS in the user's host receives a DNS reply message that provides the desired mapping.
4. The mapping (hostname - IP) is then passed to the invoking application.

Why UDP?

- fast speed
- smaller data packets

DNS Services

From the perspective of the invoking application in the user's host, DNS is a black box providing a simple, straightforward translation service.



DNS Overview

- DNS Services
- DNS Structure
  - Hierarchical structure
  - Iterated and recursive query
- DNS protocol
  - DNS Records
  - Query and reply messages
- Inserting records into DNS

DNS Structure

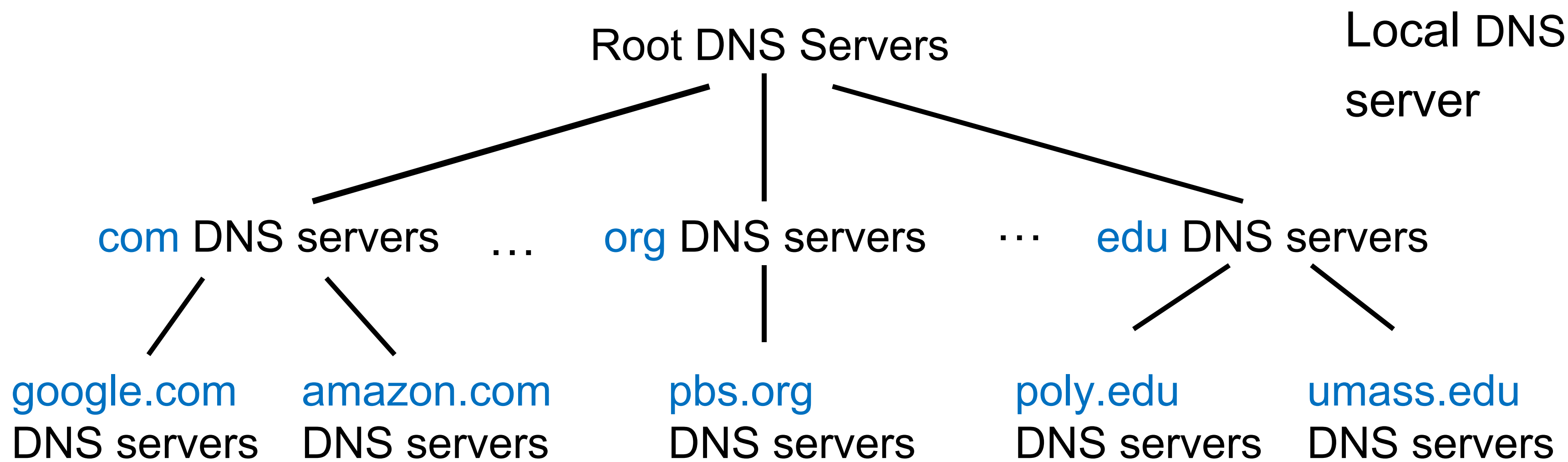
Centralized DNS：
Clients simply direct all queries to the single DNS server, and the DNS server responds directly to the querying clients.

Why not centralize DNS?

- Single point of failure
- Traffic volume
- Distant centralized database
- Maintenance: huge database, update frequently

A: doesn‘t scale!

DNS: a distributed, hierarchical database

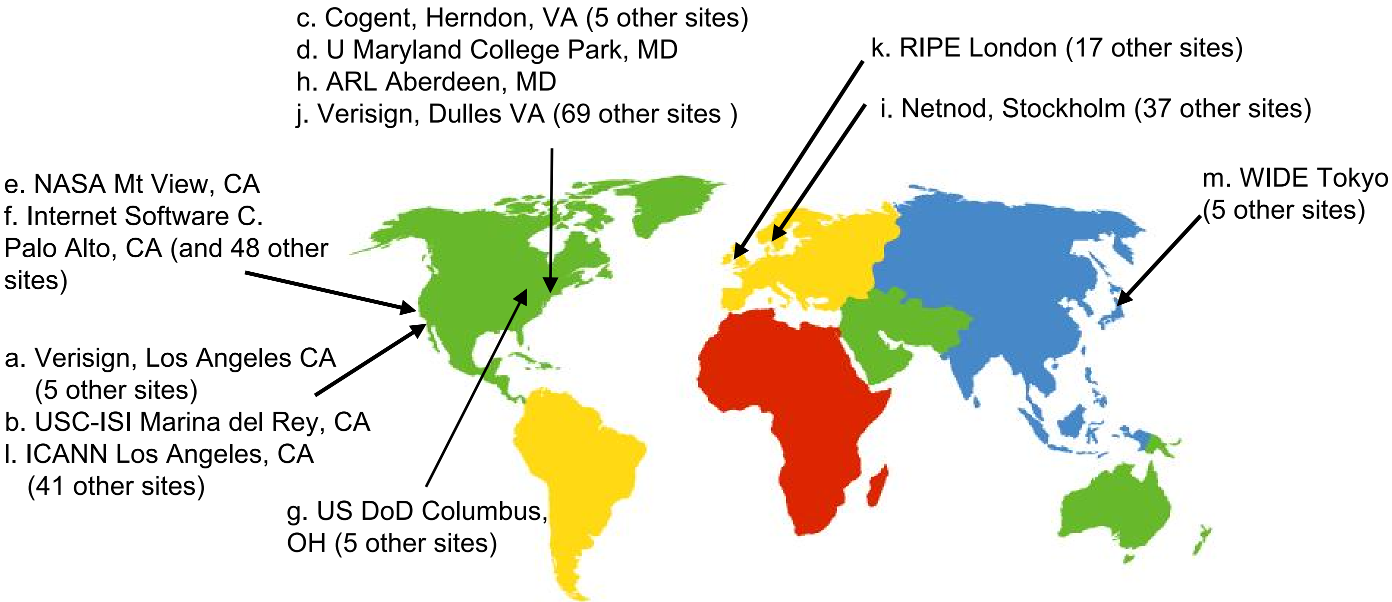


Client wants IP for www.google.com / scholar.google.com:

- Root DNS Servers: find IP address of the .com TLD DNS server
- Top-Level Domain (TLD) DNS: client queries .com DNS server to get google.com authoritative DNS server
- Authoritative DNS servers: client queries google.com DNS server to get IP address for www.google.com / scholar.google.com

DNS: root servers

- Root name server:
  - Provide the IP addresses of the TLD servers



13 logical root name "servers" worldwide

TLD, authoritative servers

Top-level domain (TLD) servers:

- Top-level domains: com, org, net, edu, aero, jobs, museums; top-level country domains: uk, fr, ca, jp
- *Network Solutions* maintains servers for .com TLD
- *Educause* for .edu TLD

Authoritative DNS servers:

- organization's own DNS server(s), providing authoritative hostname to IP mappings for organization's named hosts
- can be maintained by organization or service provider

Local DNS server

- Does not strictly belong to hierarchy
- Each ISP (residential ISP, company, university) has one
  - also called "default name server"

When a host connects to an ISP, the ISP provides the IP addresses of one or more of local DNS servers

- A host's local DNS server may be typically "close to" the host

When host makes DNS query, query is sent to local DNS server

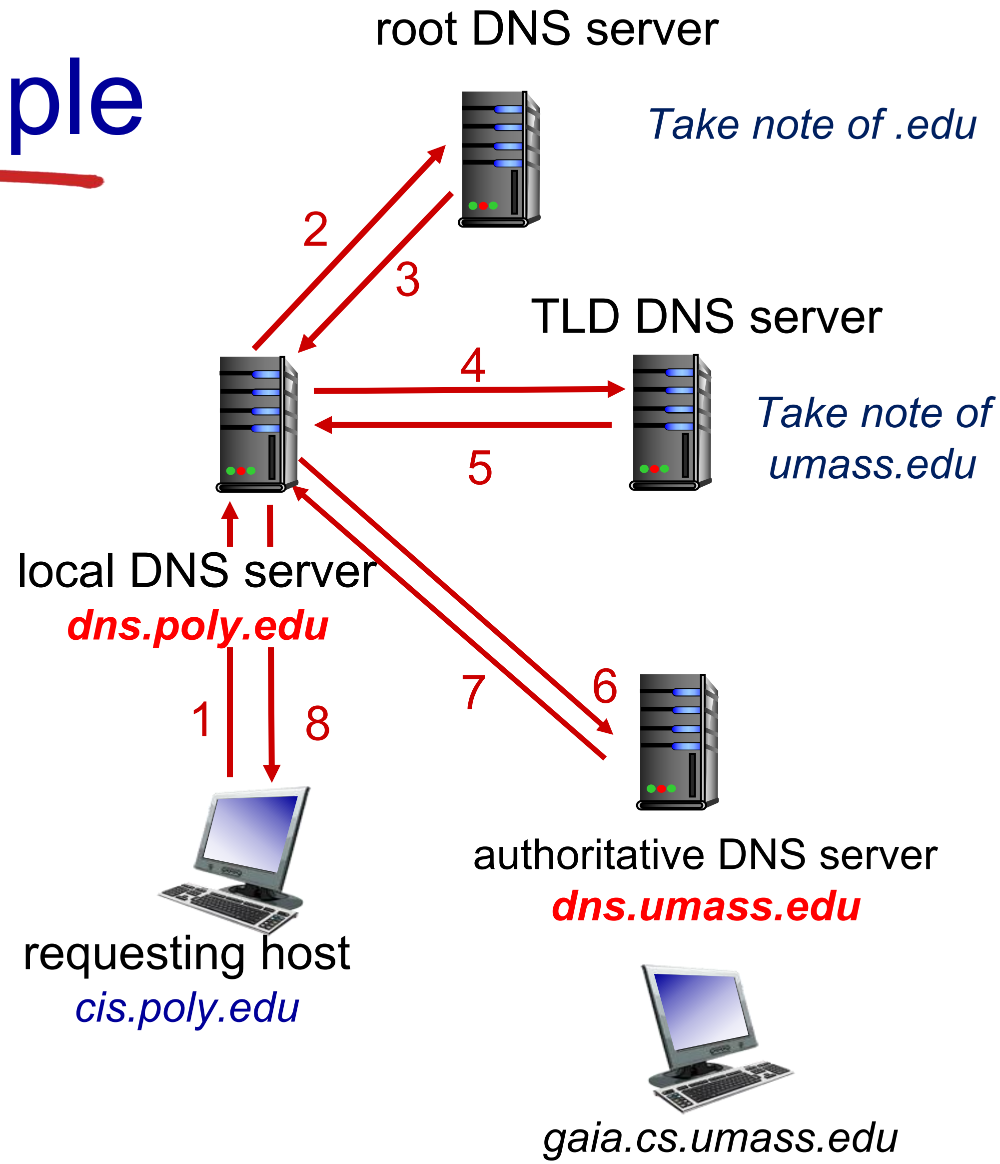
- acts as proxy, forwards query into hierarchy
- has local cache of recent name-to-address translation pairs (but may be out of date!)

DNS name resolution example

- host at cis.poly.edu wants IP address for **gaia.cs.umass.edu**

Iterated query:

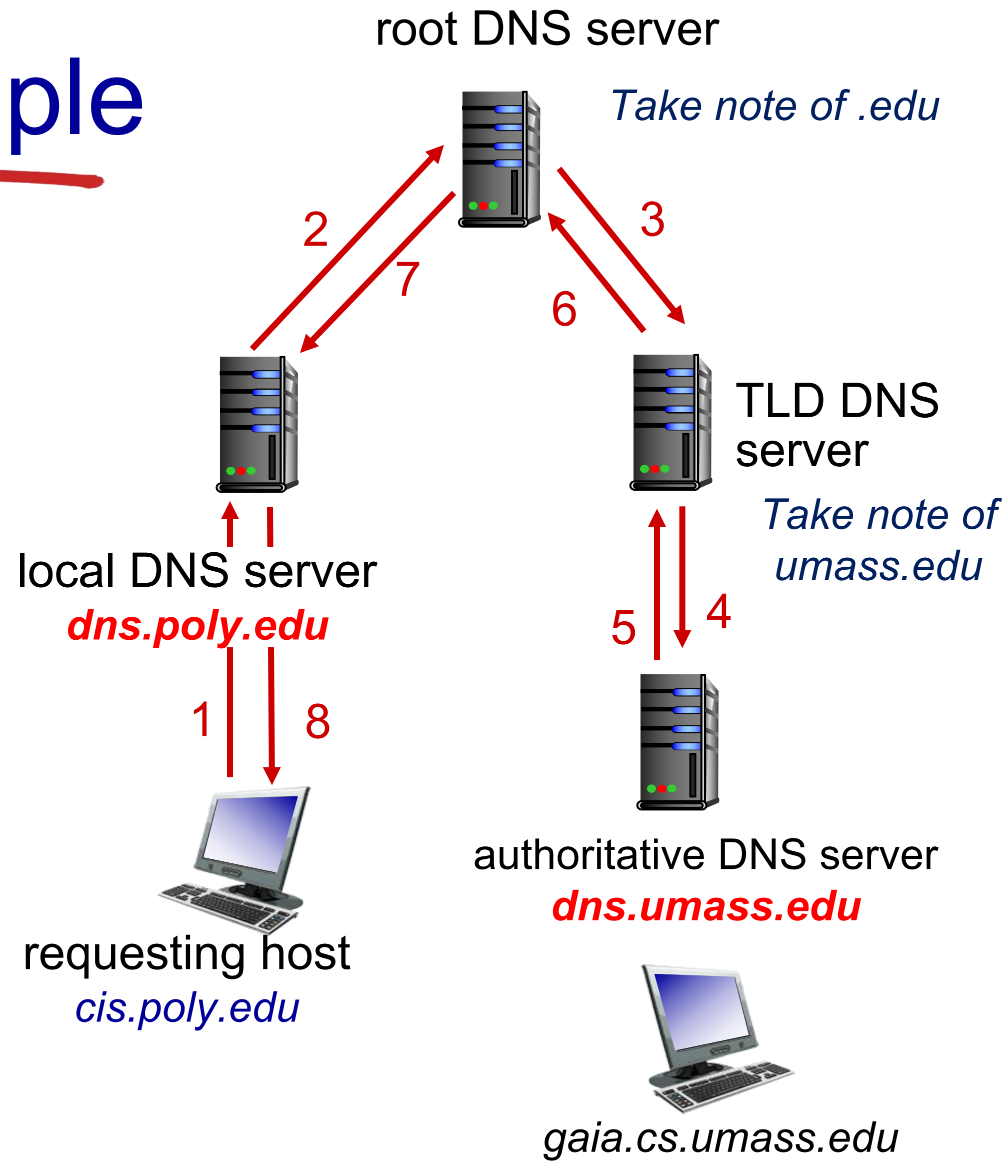
- contacted server replies with the name of another server to contact
- "I don't know this name, but ask this server"



DNS name resolution example

Recursive query:

- puts burden of name resolution on contacted name server
- heavy load at upper levels of hierarchy?



DNS: caching, updating records

- Once (any) name server learns mapping, it *caches* mapping
  - TLD servers typically cached in local name servers
  - thus root DNS servers not often visited
- Cached entries may be *out-of-date*
  - cache entries timeout (disappear) after some time (e.g., two days)
- Update/notify mechanisms proposed IETF standard
  - RFC 2136

DNS Overview

- DNS Services
- DNS Structure
  - Hierarchical structure
  - Iterated and recursive query
- DNS protocol
  - DNS Records
  - Query and reply messages
- Inserting records into DNS

DNS records

DNS: distributed database storing resource records (RR)

RR format: `(name, value, type, ttl)`

type=A

- **name** is hostname
- **value** is IP address

type=NS

- **name** is domain (e.g., foo.com)
- **value** is hostname of authoritative server for this domain (e.g., dns.foo.com)

type=CNAME

- **name** is alias name for some "canonical" (the real) name
- **www.ibm.com** is really **servereast.backup2.ibm.com**
- **value** is canonical name

type=MX

- **value** is canonical name of the mailserver with **name** (alias name)

DNS records

If a DNS server is authoritative for a particular hostname

- the DNS server will contain a Type A record for the hostname
- (Even if the DNS server is not authoritative, it may contain a Type A record in its cache.)

If a server is not authoritative for a hostname

- the server will contain a Type NS record for the domain that includes the hostname
- it will also contain a Type A record that provides the IP address of the DNS server in the Value field of the NS record.

Example: an .edu TLD server is not authoritative for gaia.cs.umass.edu

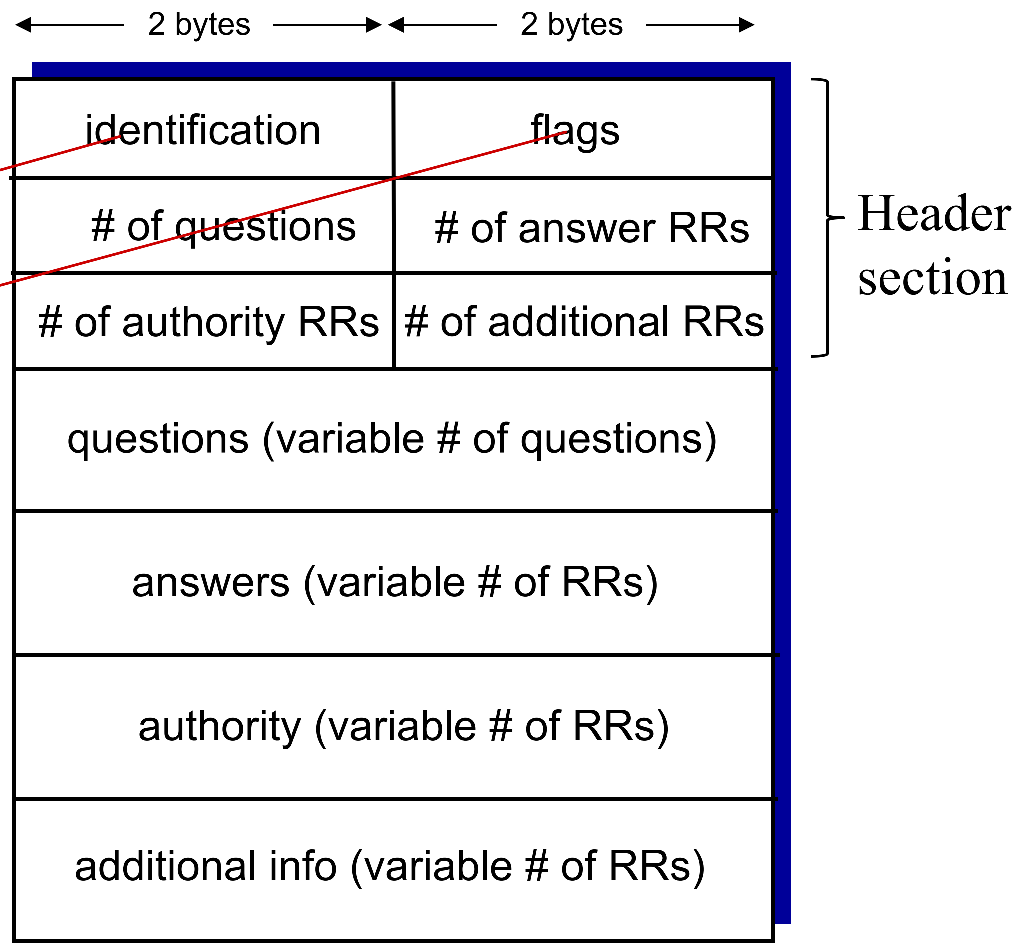
  - (umass.edu, dns.umass.edu, NS) .
  - (dns.umass.edu, 128.119.40.111, A)

DNS protocol, messages

Query and reply messages, both with same message format

message header

- identification: 16 bit number for query, reply to query uses same number
- flags:
  - query or reply
  - recursion desired
  - recursion available
  - reply is authoritative



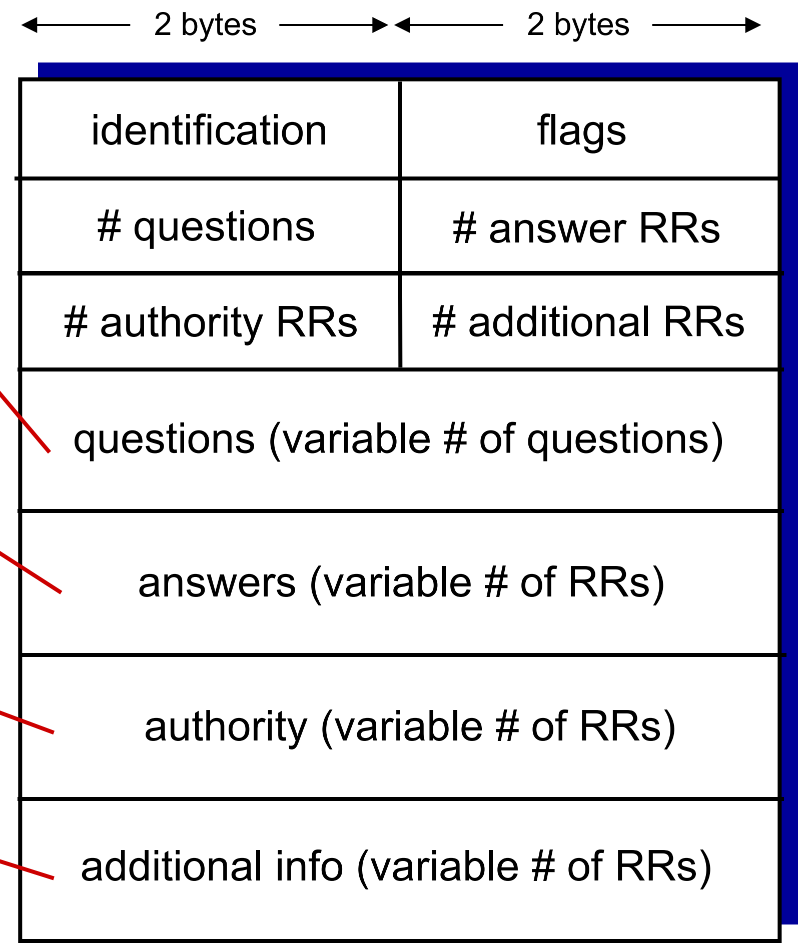
DNS protocol, messages

Name & type fields (e.g., Type A or Type MX)

RRs in response to query (a reply can return multiple RRs)

records of other authoritative servers

additional "helpful" info that may be used



DNS protocol, messages

For example, a reply to **an MX query**

Answer section: Type MX

- an RR providing the canonical hostname of a mail server.

Additional section: Type A

- the IP address for the canonical hostname of the mail server.

2 bytes	2 bytes
identification	flags
# questions	# answer RRs
# authority RRs	# additional RRs
questions (variable # of questions)	
answers (variable # of RRs)	
authority (variable # of RRs)	
additional info (variable # of RRs)	

additional "helpful" info that may be used

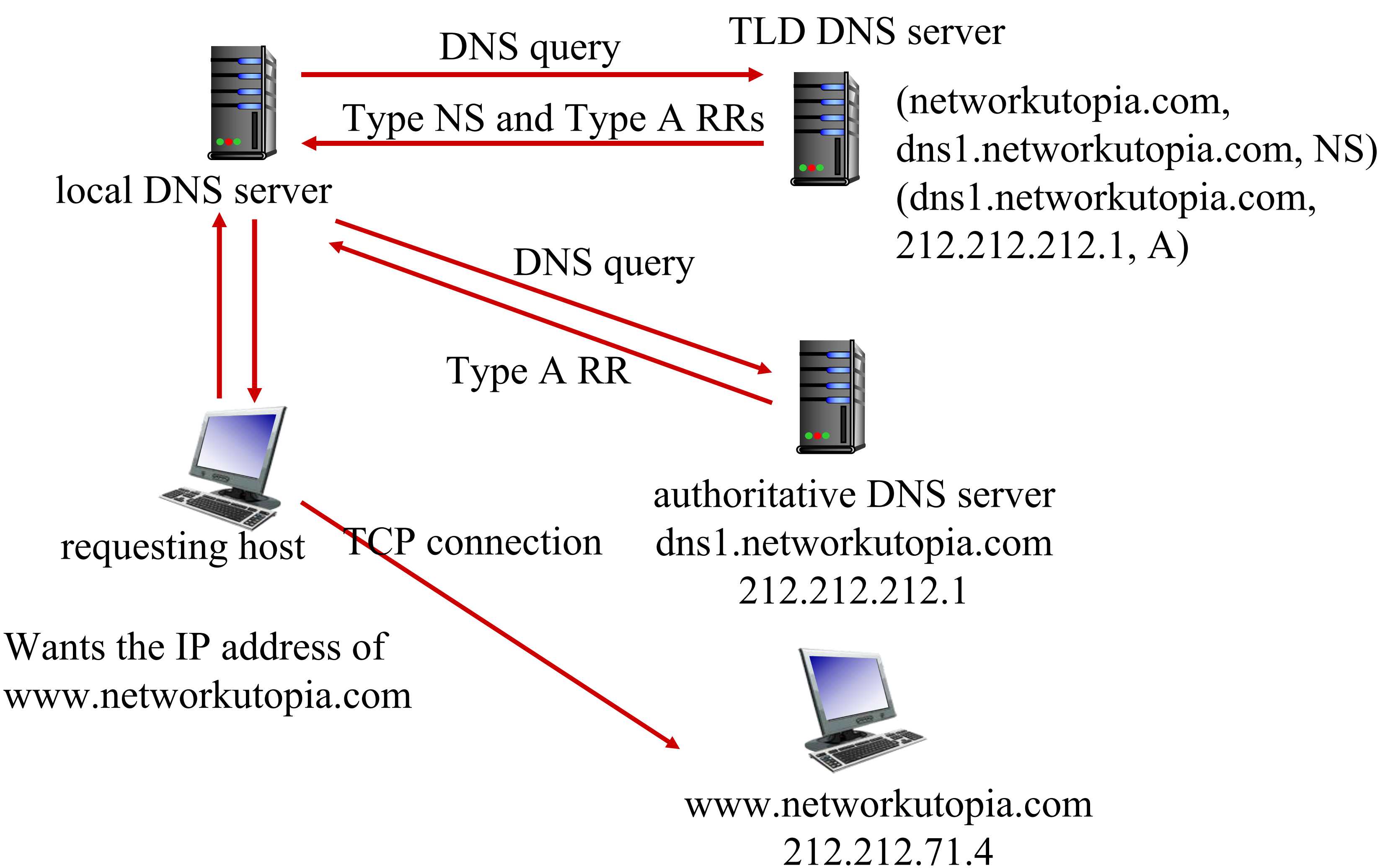
DNS Overview

- DNS Services
- DNS Structure
  - Hierarchical structure
  - Iterated and recursive query
- DNS protocol
  - DNS Records
  - Query and reply messages
- Inserting records into DNS server

Inserting records into DNS

- Example: new startup "Network Utopia"
- Register name networkuptopia.com at *DNS registrar* (e.g., Network Solutions)
  - provide names, IP addresses of authoritative DNS server (primary and secondary)
  - registrar inserts two RRs into .com TLD server:
    (networkutopia.com, dns1.networkutopia.com, NS)
    (dns1.networkutopia.com, 212.212.212.1, A)

Inserting records into DNS



Attacking DNS

Distributed denial-of-service (DDoS) attacks

- bombard root servers with traffic
  - not successful to date
  - traffic filtering
  - local DNS servers cache IPs of TLD servers, allowing root server bypass
- bombard TLD servers
  - potentially more dangerous

Redirect attacks

- man-in-middle
  - Intercept queries; bogus reply
- DNS poisoning
  - Send bogus replies to DNS server

Exploit DNS for DDoS

- target IP
- Redirect an unsuspecting Web user to attack Web site

Chapter 2: outline

2.1 principles of network applications

2.2 Web and HTTP

2.3 electronic mail

- SMTP, POP3, IMAP

2.4 DNS

2.5 P2P applications

2.6 video streaming and content distribution networks

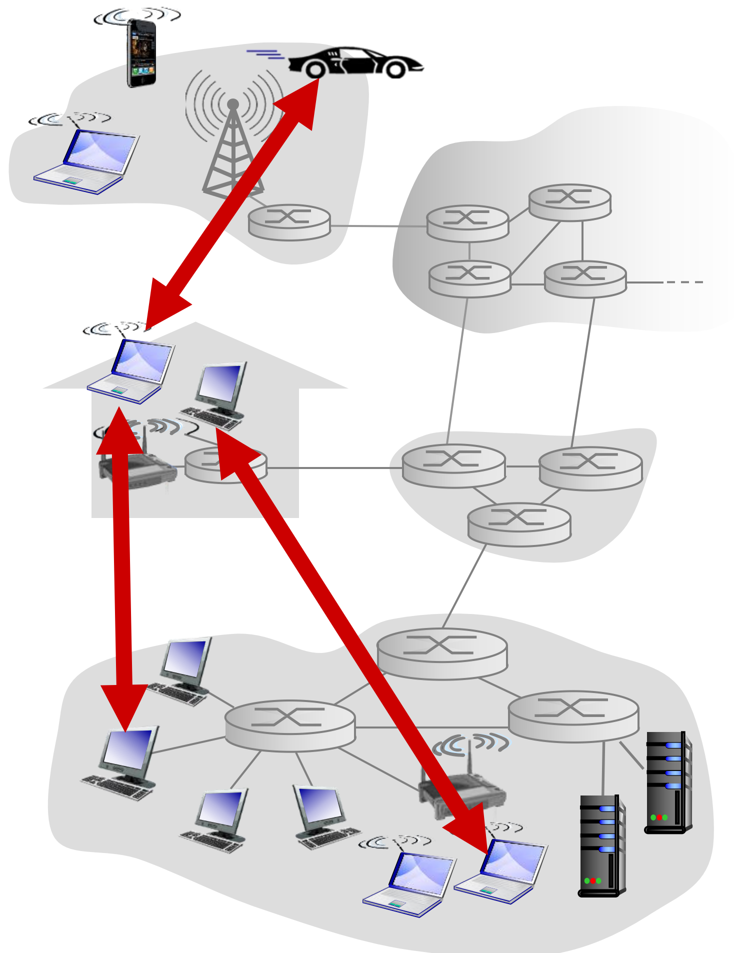
2.7 socket programming with UDP and TCP

Pure P2P architecture

- *no* always-on server
- arbitrary end systems directly communicate
- peers are intermittently connected and change IP addresses

Examples:

- file distribution (BitTorrent)
- Streaming (KanKan)
- VoIP (Skype)



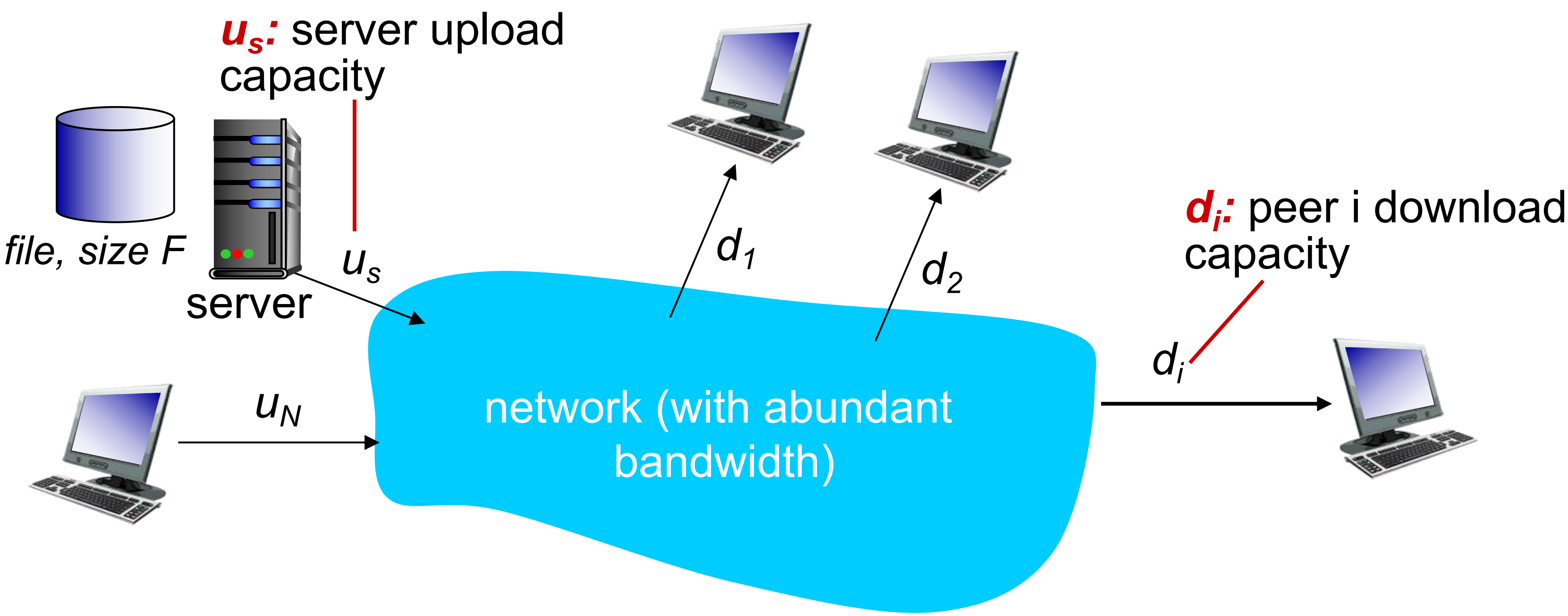
DNS Overview

- P2P vs Client Server
- BitTorrent

File distribution: client-server vs P2P

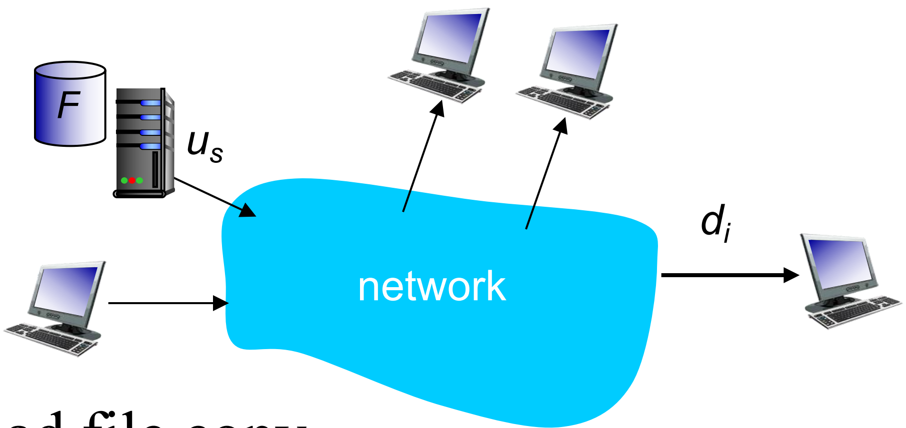
Question: How much time to distribute file (size F) from one server to N *peers*?

- peer upload/download capacity is limited resource
- **Distribution time:** the time it takes to get a copy of the file to all N peers.



File distribution time: client-server

- Server transmission: must sequentially send (upload) N file copies:
  - time to send one copy: F/u_s
  - time to send N copies: NF/u_s



- Client: each client must download file copy
  - d_{min} = min client download rate
  - maximum client download time: F/d_{min}

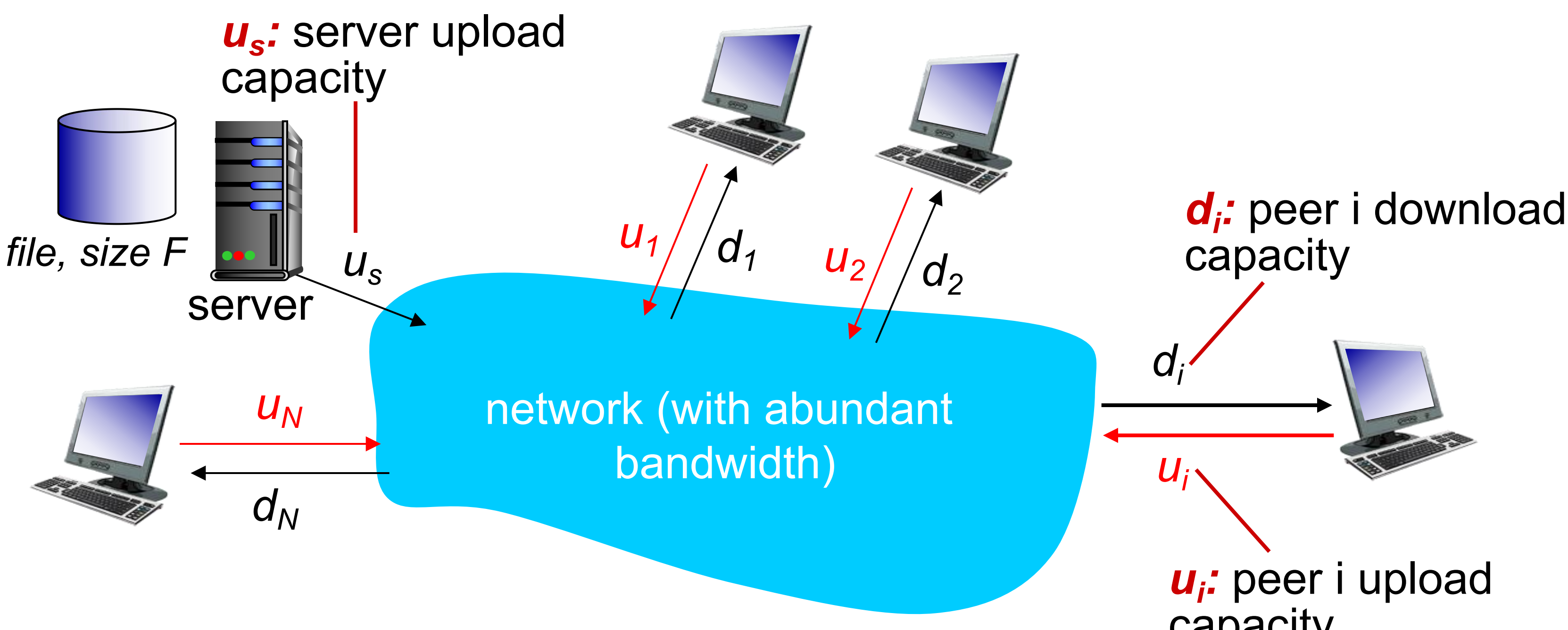
time to distribute F to N clients using client-server approach

$$D_{c\text{-}s} \geq max\{NF/u_{s,},F/d_{min}\}$$

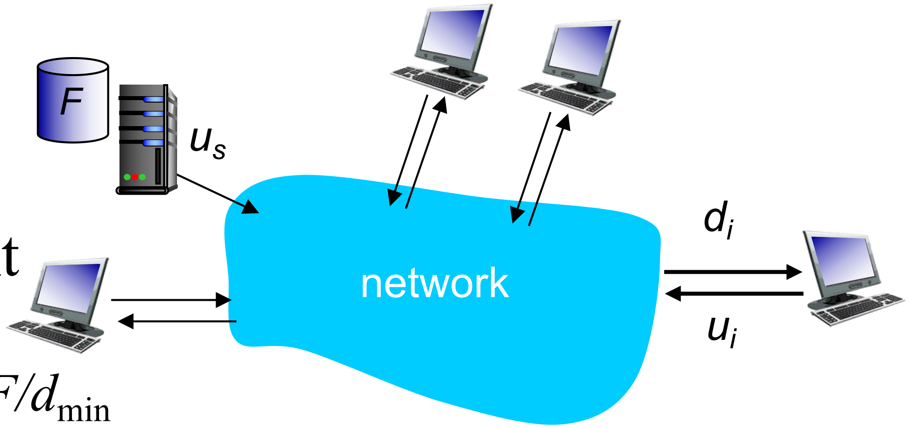
increases linearly in N

File distribution time: P2P

In P2P model, clients are <u>both downloaders and uploaders</u>.



File distribution time: P2P



- Server transmission: must upload at least one copy
  - time to send one copy: F/u_s
- Client downloading: each client must download file copy
  - maximum client download time: $F/d_{\min}$
- Clients and server: delivering a total of NF bits
  - max upload rate (limiting max download rate) is $u_s + \Sigma u_i$

time to distribute F to N clients using P2P approach

$$D_{P2P} \geq max\{F/u_{s,},F/d_{min,},NF/(u_s + \Sigma u_i)\}$$

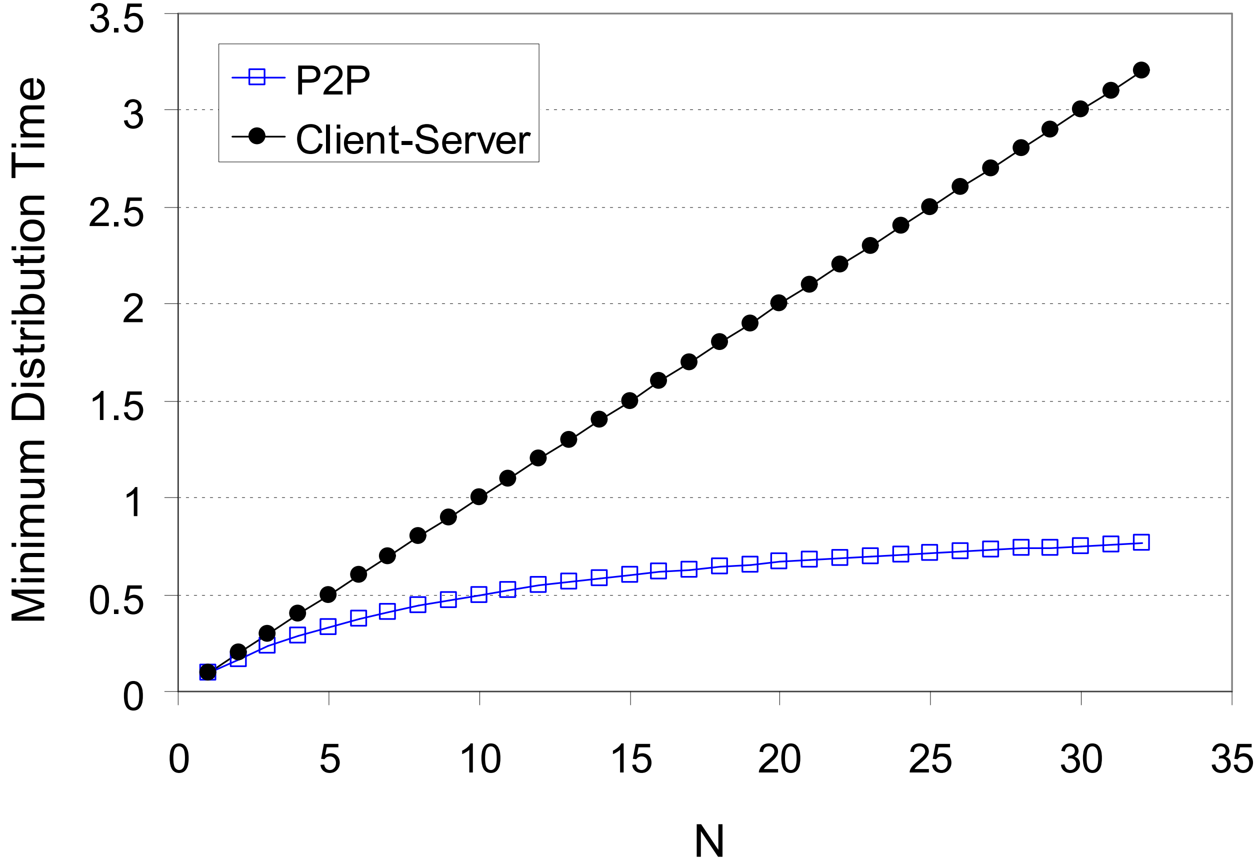
If each peer can redistribute a bit as soon as it receives the bit, then there is a scheme that actually achieves this lower bound

increases linearly in N …

… but so does this, as each peer brings service capacity

Client-server vs. P2P: example

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



DNS Overview

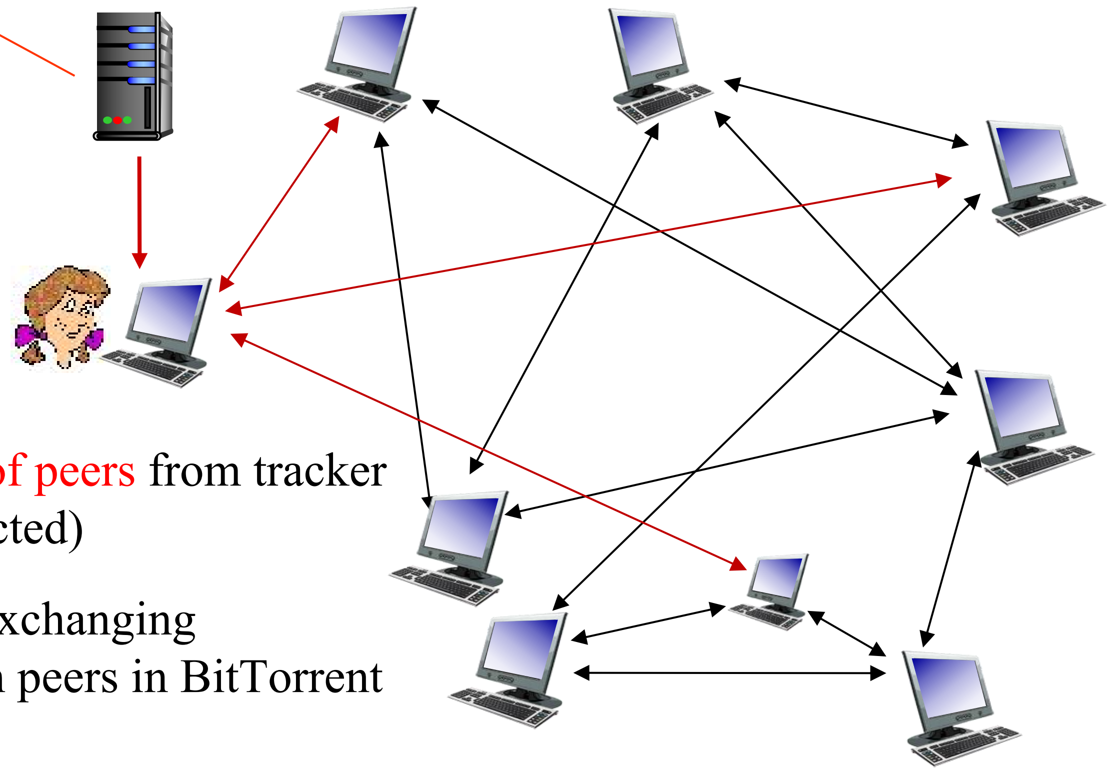
- P2P vs Client Server
- BitTorrent

P2P file distribution: BitTorrent

- File divided into 256Kb chunks
- Peers in BitTorrent send/receive file chunks

tracker: tracks peers participating in BitTorrent

torrent: group of peers exchanging chunks of a file

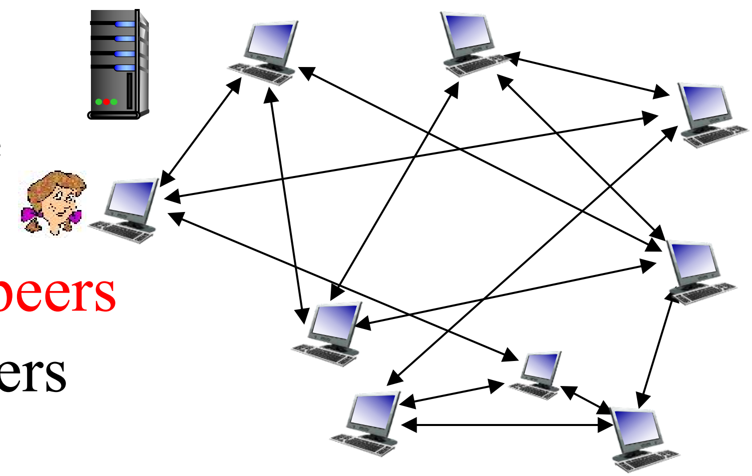


Alice arrives …

… obtains list of peers from tracker (randomly selected)

… and begins exchanging file chunks with peers in BitTorrent

P2P file distribution: BitTorrent



- Peer joining BitTorrent:
  - has no chunks, but will accumulate them over time from other peers
  - registers with tracker to get list of peers
  - TCP connections with subset of peers ("neighbors")

- While downloading, peer uploads chunks to other peers
  - Peers may leave
  - Peers may come, initiating connections with Alice

- Once peer has entire file, it may (selfishly) leave or (altruistically) remain in BitTorrent

BitTorrent: requesting, sending file chunks

Q1: which chunks should she request first from her neighbors?

Q2: to which of her neighbors should she send requested chunks?

requesting chunks:

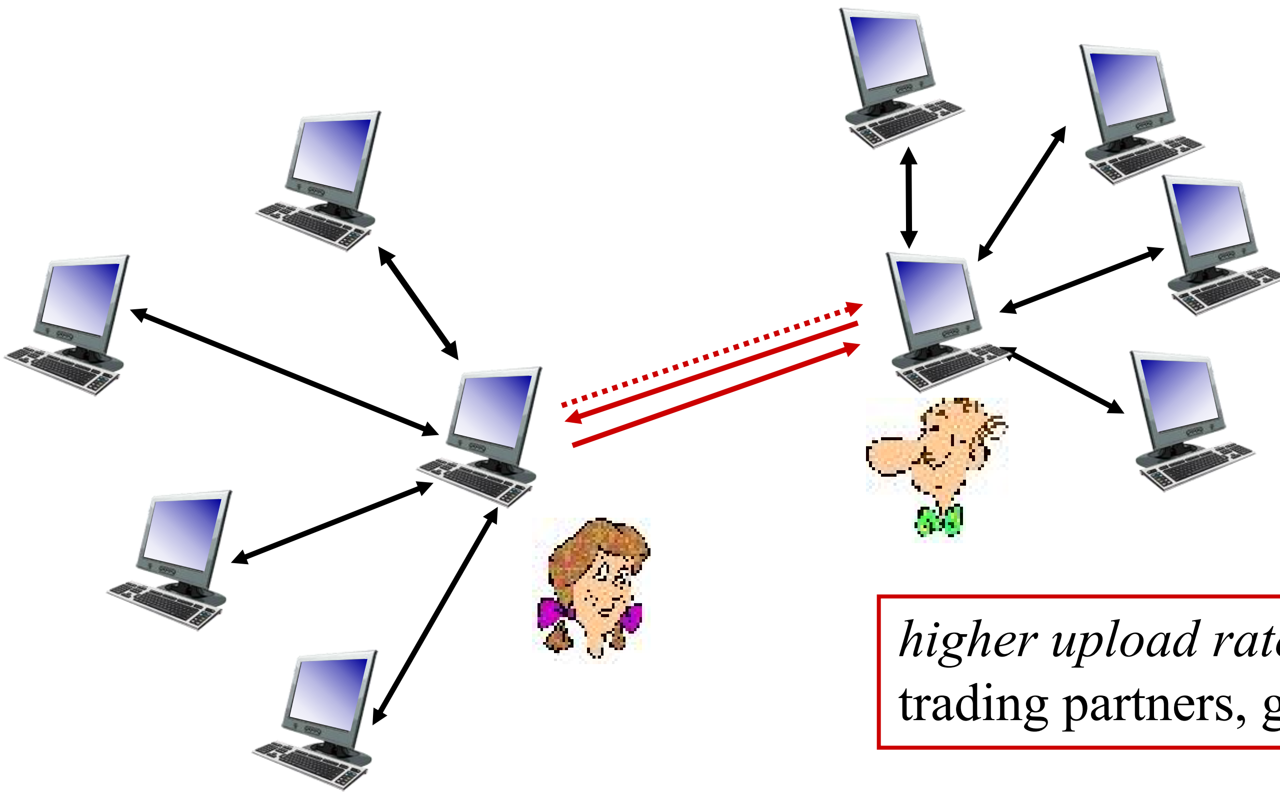
- at any given time, different peers have different subsets of file chunks
- periodically, Alice asks each "neighbor" for list of chunks that they have
- Alice requests missing chunks from peers, rarest first

sending chunks: tit-for-tat

- Alice sends chunks to those four peers currently sending her chunks at highest rate
  - other peers are choked by Alice (do not receive chunks from her)
  - re-evaluate every 10 secs
- every 30 secs: randomly select one additional peer, starts sending chunks
  - "optimistically unchoke" this peer
  - newly chosen peer may join top 4

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



higher upload rate: find better trading partners, get file faster!

Chapter 2: outline

2.1 principles of network applications

2.2 Web and HTTP

2.3 electronic mail

- SMTP, POP3, IMAP

2.4 DNS

2.5 P2P applications

2.6 video streaming and content distribution networks

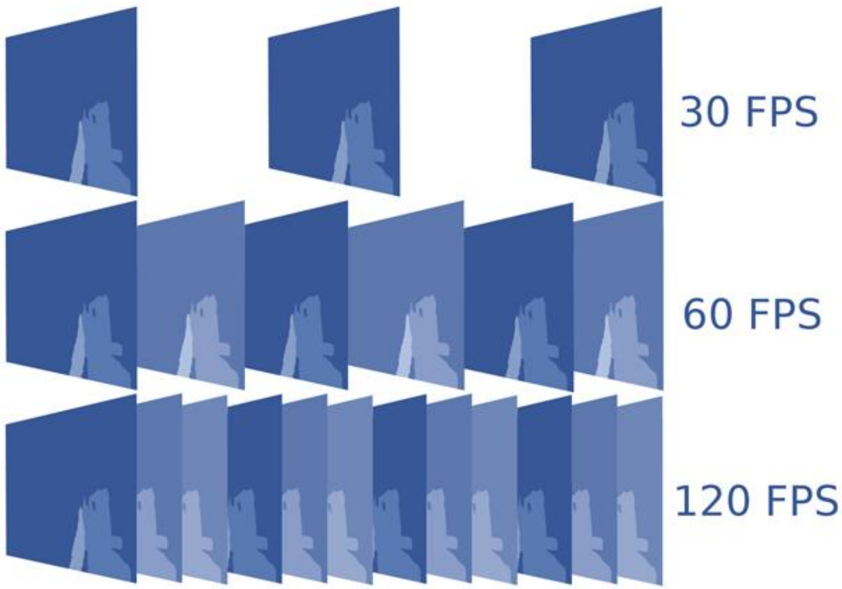
2.7 socket programming with UDP and TCP

Video Streaming and CDNs: ~~context~~

- Video traffic: major consumer of Internet bandwidth
  - Netflix, YouTube: 37%, 16% of downstream residential ISP traffic
  - ~1B YouTube users, ~75M Netflix users
- Challenge: scale - how to reach ~1B users?
  - single mega-video server won't work (why?)
- Challenge: heterogeneity
  - different users have different capabilities (e.g., wired versus mobile; bandwidth rich versus bandwidth poor)
- Solution: distributed, application-level infrastructure

Multimedia: video

- Video: sequence of images displayed at constant rate
  - e.g., 24 images/sec
- Digital image: array of pixels
  - each pixel represented by bits

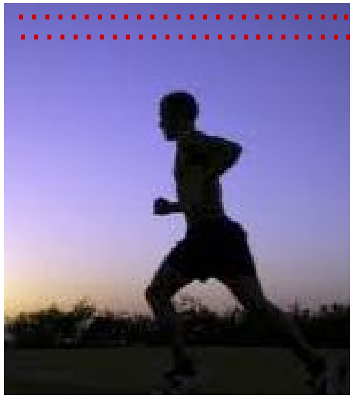


Multimedia: video

Coding (Compression): use redundancy *within* and *between* images to decrease # bits used to encode image

- spatial (within image)
- temporal (from one image to next)

Spatial coding example: instead of sending N values of same color (all purple), send only two values: color value (purple) and number of repeated values (N)



frame *i*

frame *i+1*

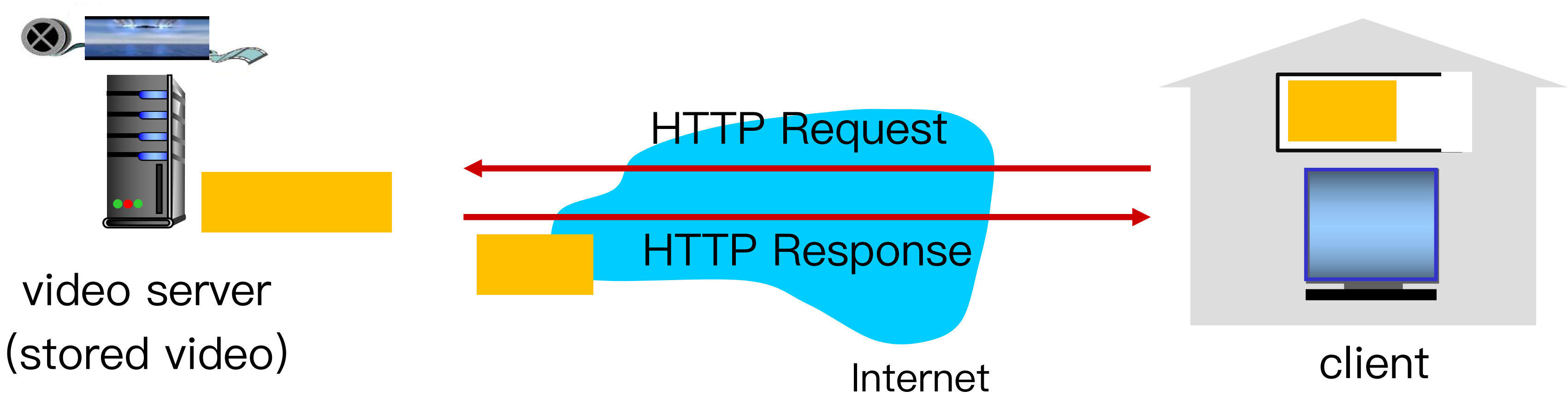
temporal coding example: instead of sending complete frame at i+1, send only differences from frame i

Multimedia: video

Type	Video Bitrate, Standard Frame Rate (24, 25, 30)	Video Bitrate, High Frame Rate (48, 50, 60)
2160p (4k)	35-45 Mbps	53-68 Mbps
1440p (2k)	16 Mbps	24 Mbps
1080p	8 Mbps	12 Mbps
720p	5 Mbps	7.5 Mbps
480p	2.5 Mbps	4 Mbps
360p	1 Mbps	1.5 Mbps

- CBR: (constant bit rate): video encoding rate fixed
- VBR: (variable bit rate): video encoding rate changes as amount of spatial, temporal coding changes

HTTP Streaming



All clients receive the same encoding of the video:

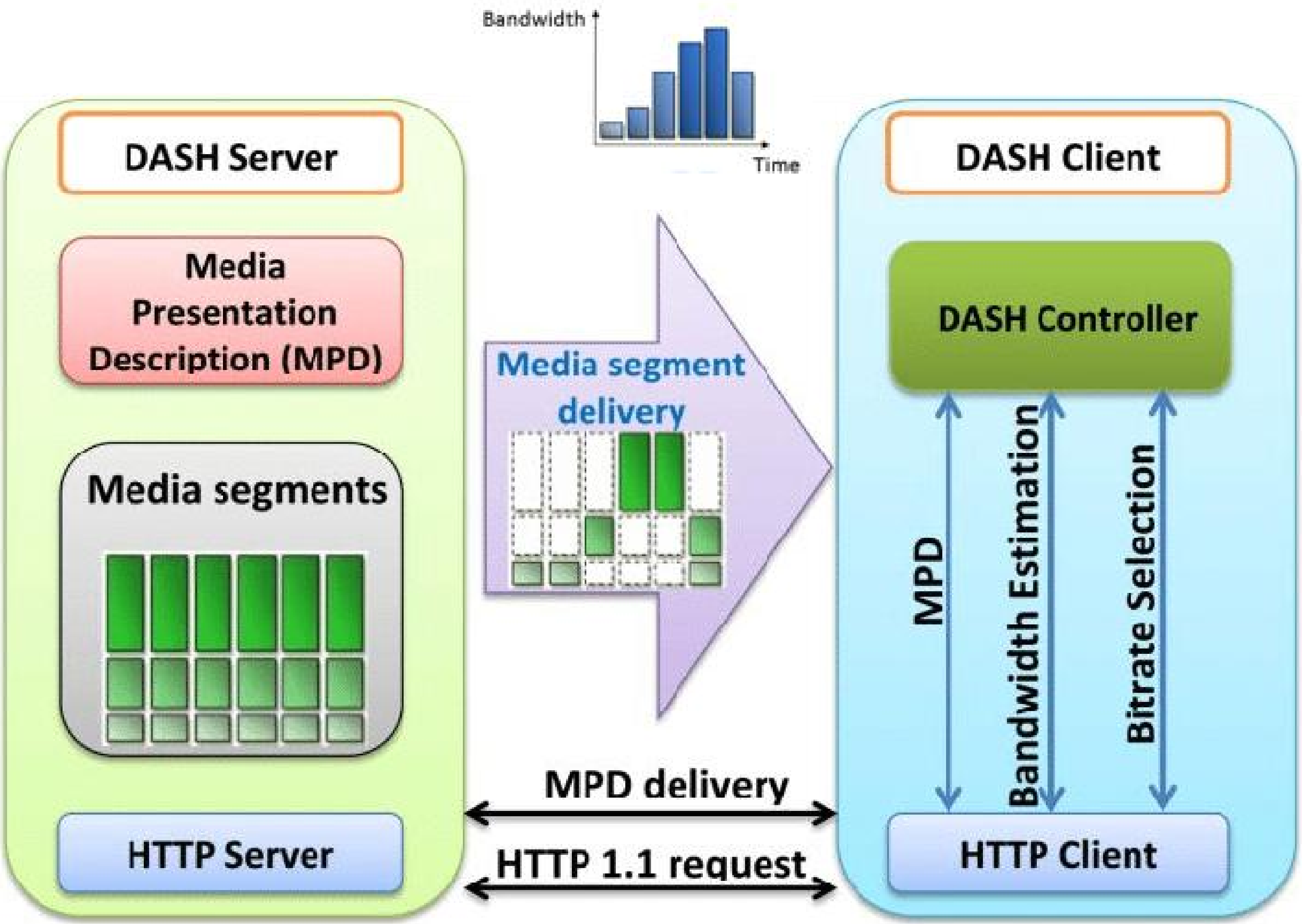
- Human users may have different requirements
- Clients may have different available bandwidth, which may be time-varying

How to deal with this?

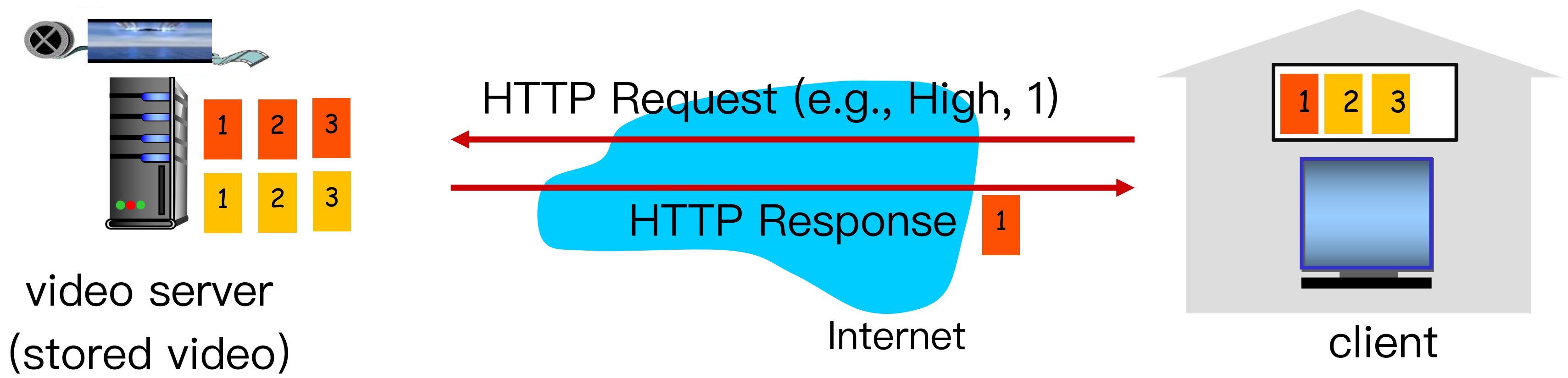
Streaming multimedia: DASH

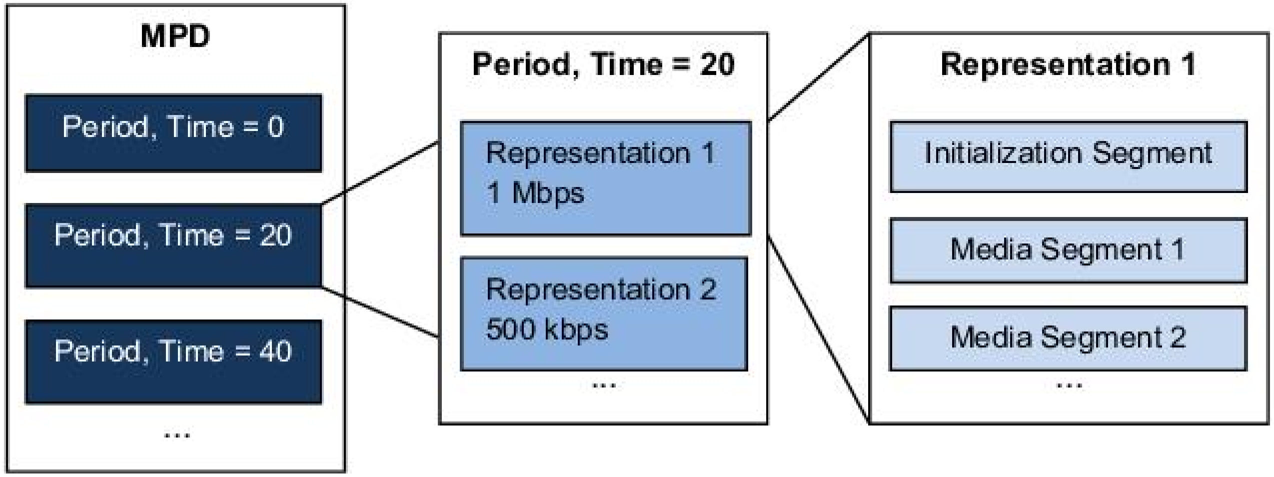
- DASH: Dynamic, Adaptive Streaming over HTTP
- Server:
  - divides video file into multiple chunks
  - each chunk stored, encoded at different rates
  - manifest file: provides URLs for different chunks encoded at different rates
- Client:
  - periodically measures server-to-client bandwidth
  - consulting manifest, requests one chunk at a time
    - chooses maximum coding rate sustainable given current bandwidth
    - can choose different coding rates at different points in time (depending on available bandwidth at time)

Streaming multimedia: DASH



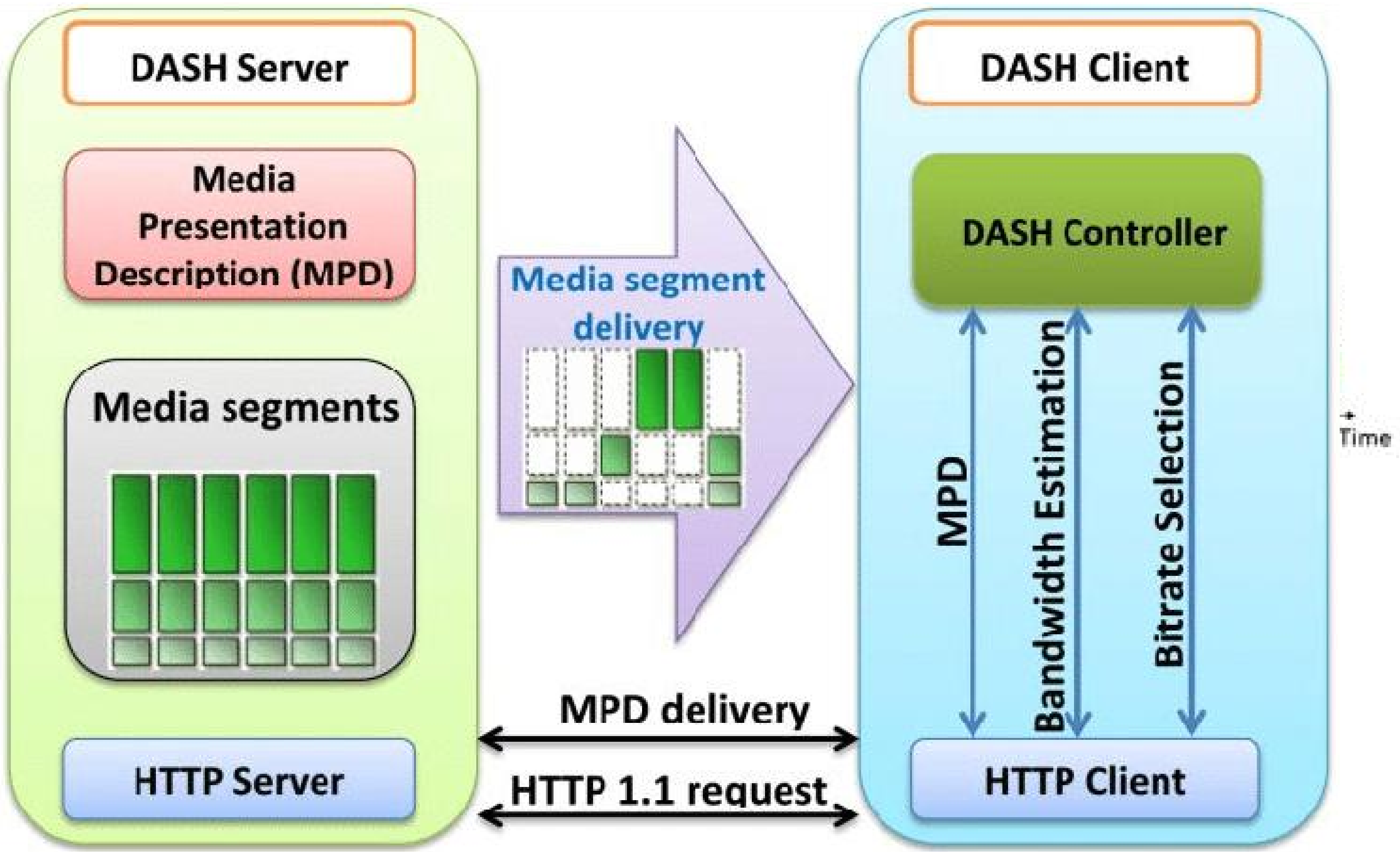
Streaming multimedia: DASH





Media presentation description (MPD), also known as manifest

Streaming multimedia: DASH



Streaming multimedia: DASH

"intelligence" at client: client determines

- when to request chunk (so that buffer starvation, or overflow does not occur)
- what encoding rate to request (higher quality when more bandwidth available)
- where to request chunk (can request from URL server that is "close" to client or has high available bandwidth)

Chapter 2: outline

2.1 principles of network applications

2.2 Web and HTTP

2.3 electronic mail

- SMTP, POP3, IMAP

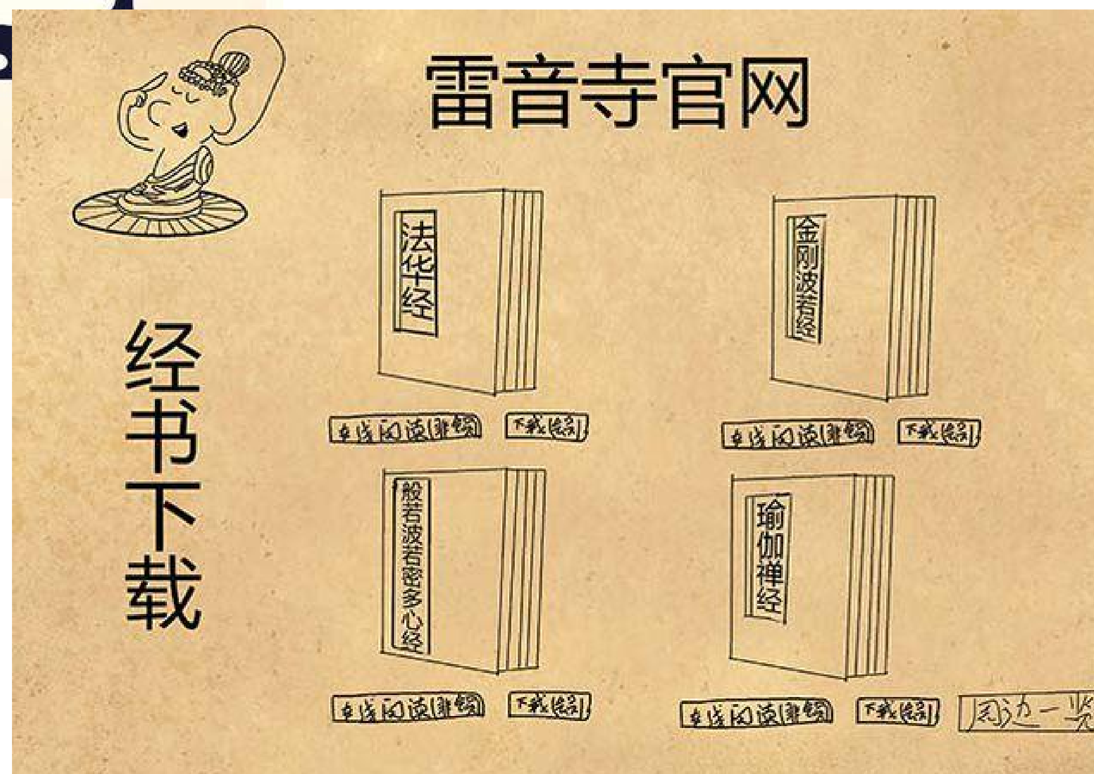
2.4 DNS

2.5 P2P applications

2.6 video streaming and content distribution networks

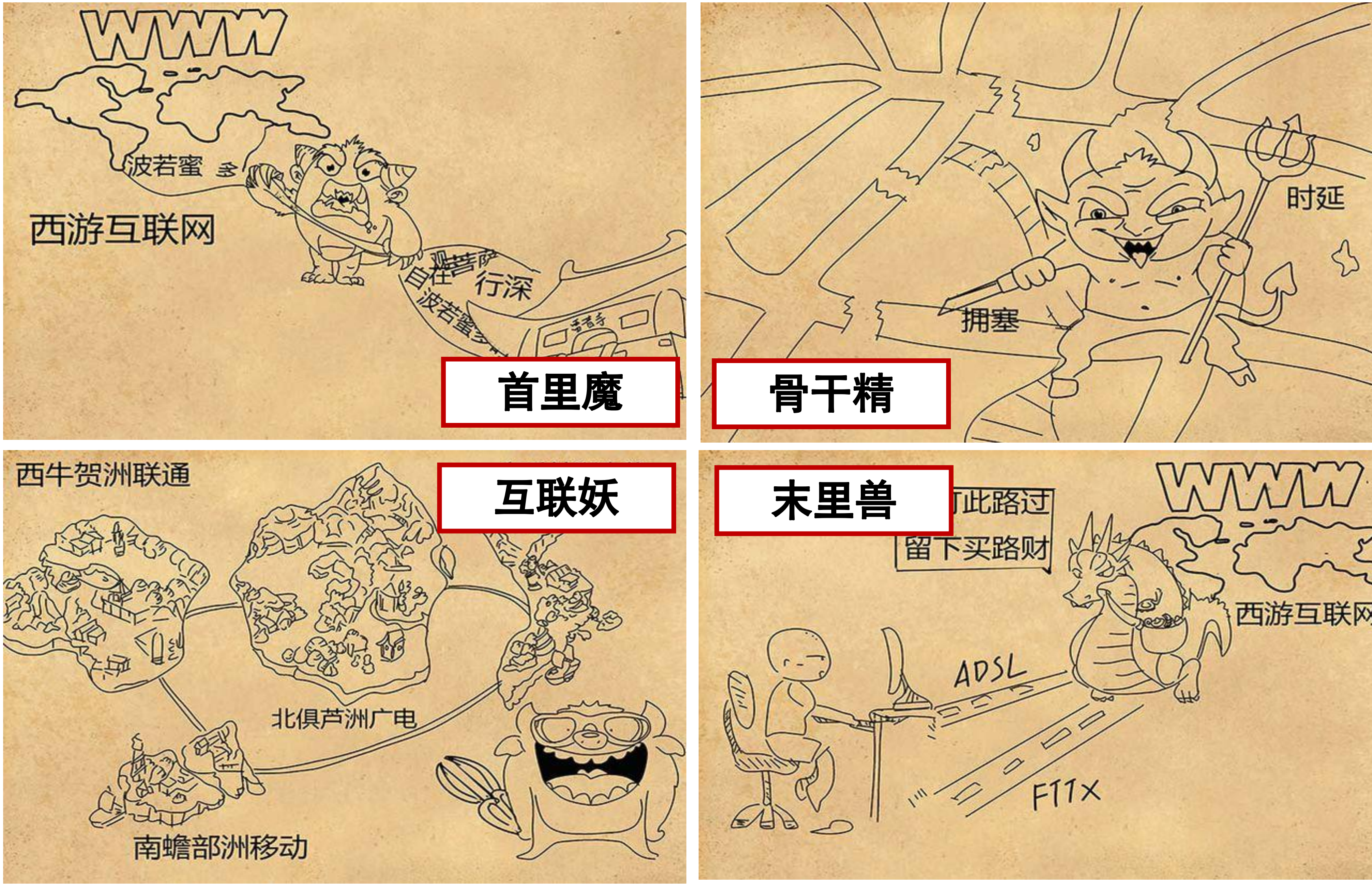
2.7 socket programming with UDP and TCP

Content distribution networks

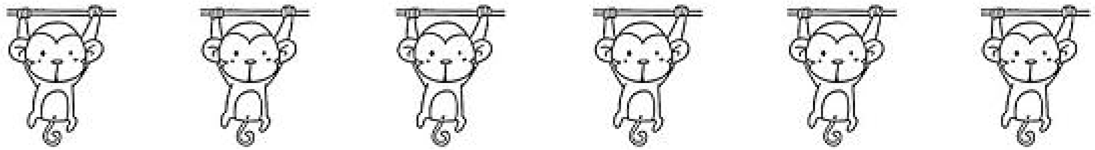


来源：小黑羊JoinWings
https://baijia.baidu.com/s?old_id=126615&wfr=pc&fr=app_list

Content distribution networks



Content distribution networks



Content distribution networks

- Challenge: how to stream content (selected from millions of videos) to hundreds of thousands of simultaneous users?

- Option 1: single, large "mega-server"
  - single point of failure
  - huge traffic
  - long path to distant clients
  - multiple copies of video sent over outgoing link

….quite simply: this solution doesn't scale

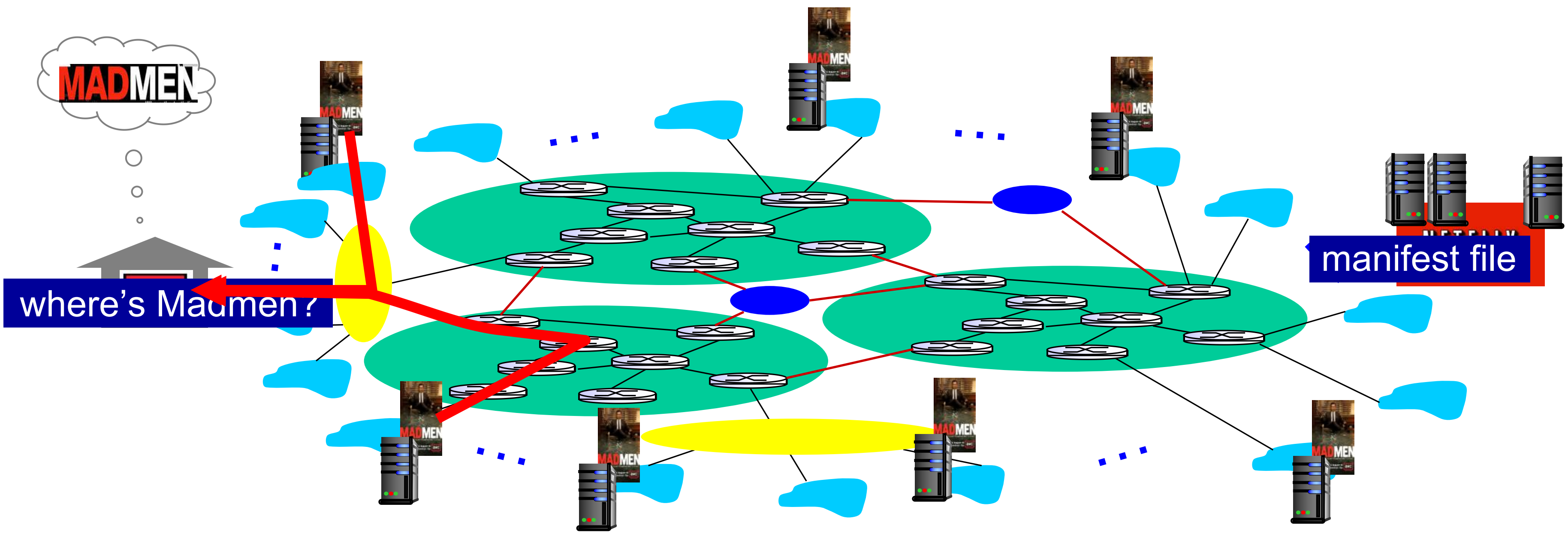
Content distribution networks

- Challenge: how to stream content (selected from millions of videos) to hundreds of thousands of simultaneous users?

- Option 2: Content distribution networks (CDN) store/serve multiple copies of videos at multiple geographically distributed sites
  - Enter deep: push CDN servers deep into many access networks; inside ISPs
    - close to users
    - used by Akamai, 1700 locations
  - Bring home: smaller number (10's) of larger clusters in Internet Exchange Point (IXP); outside ISPs
    - used by Limelight

Content Distribution Networks (CDNs)

- CDN: stores copies of content at CDN nodes
  - e.g. Netflix stores copies of MadMen
- subscriber requests content from CDN
  - directed to nearby copy, retrieves content
  - may choose different copy if network path congested



Content Distribution Networks (CDNs)

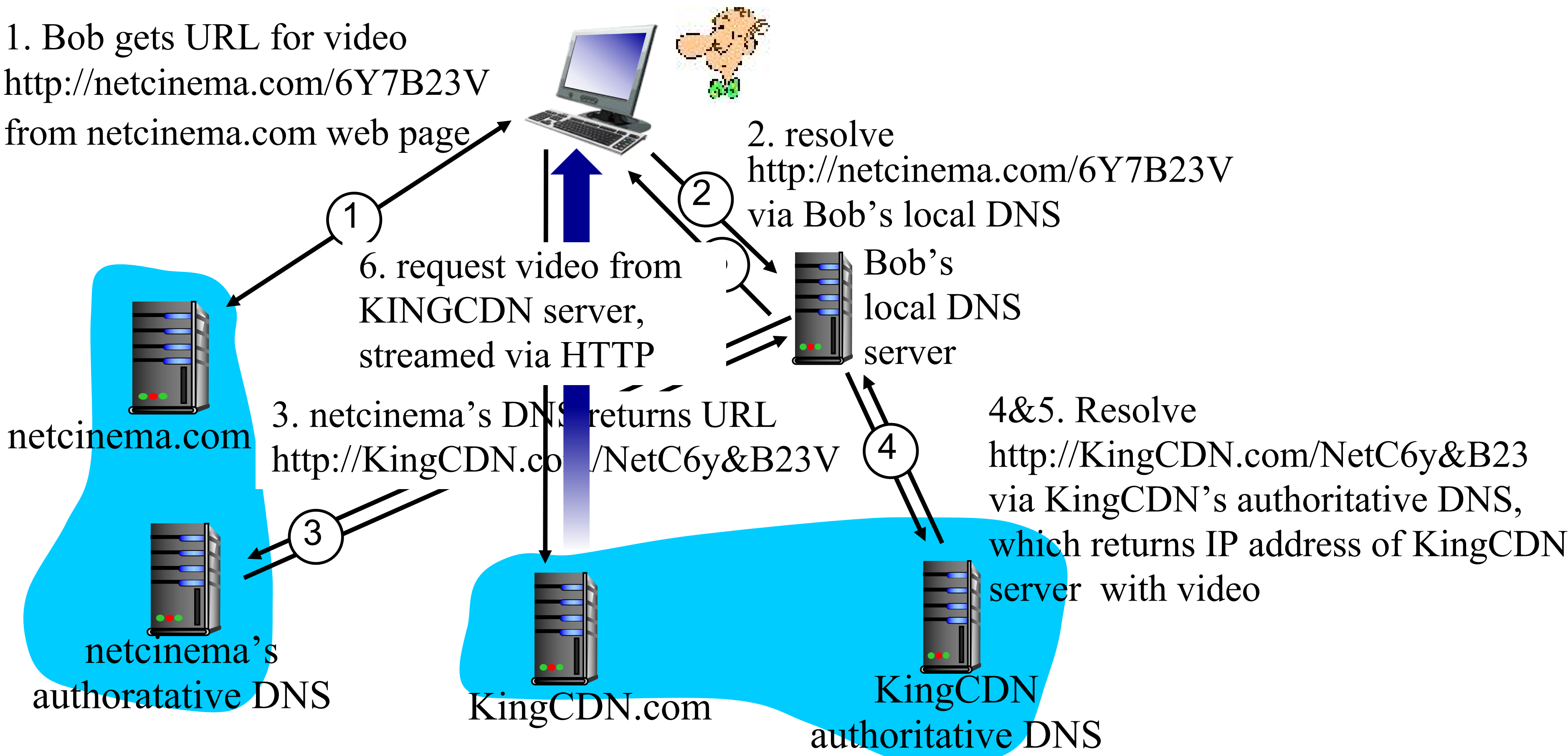
Challenges: Coping with a congested Internet

- what content to place in CDN node?
  - Simple pull strategy: request, then store
- from which CDN node to retrieve content?
  - Cluster selection strategy
- the operation for retrieving content?
  - CDN operation

CDN Operation

Bob (client) requests video http://netcinema.com/6Y7B23V

- video stored in CDN at http://KingCDN.com/NetC6y&B23V



CDN: Cluster Selection Strategy

One simple strategy is to assign the client to the cluster that is **geographically closest**:

- When a DNS request is received from a particular LDNS, the CDN chooses the geographically closest cluster
- may not be the closest cluster in terms of the length or number of hops
- ignore the variation in delay and available bandwidth over time

Periodic **real-time measurements** of delay and loss performance between their clusters and clients:

- a CDN can have each of its clusters periodically send probes to all of the LDNSs around the world.
- many LDNSs are configured to not respond to such probes.

Chapter 2: outline

2.1 principles of network applications

2.2 Web and HTTP

2.3 electronic mail

- SMTP, POP3, IMAP

2.4 DNS

2.5 P2P applications

2.6 video streaming and content distribution networks

2.7 socket programming with UDP and TCP