Data Communication and Computer Networks

3. Application Layer PART-B

Dr. Aiman Hanna
Department of Computer Science & Software Engineering
Concordia University, Montreal, Canada

These slides has mainly been extracted, modified and updated from original slides of :
Computer Networking: A Top Down Approach, 6th edition Jim Kurose, Keith Ross
Addison-Wesley, 2013

Additional materials have been extracted, modified and updated from:
Understanding Communications and Networking, 3e by William A. Shay 2005

DNS: domain name system

people: many identifiers:

- SSN, name, passport #

Internet hosts, routers:

- IP address (32 bit) - used for addressing datagrams
- "name", e.g., www.yahoo.com - used by humans

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

Domain Name System:

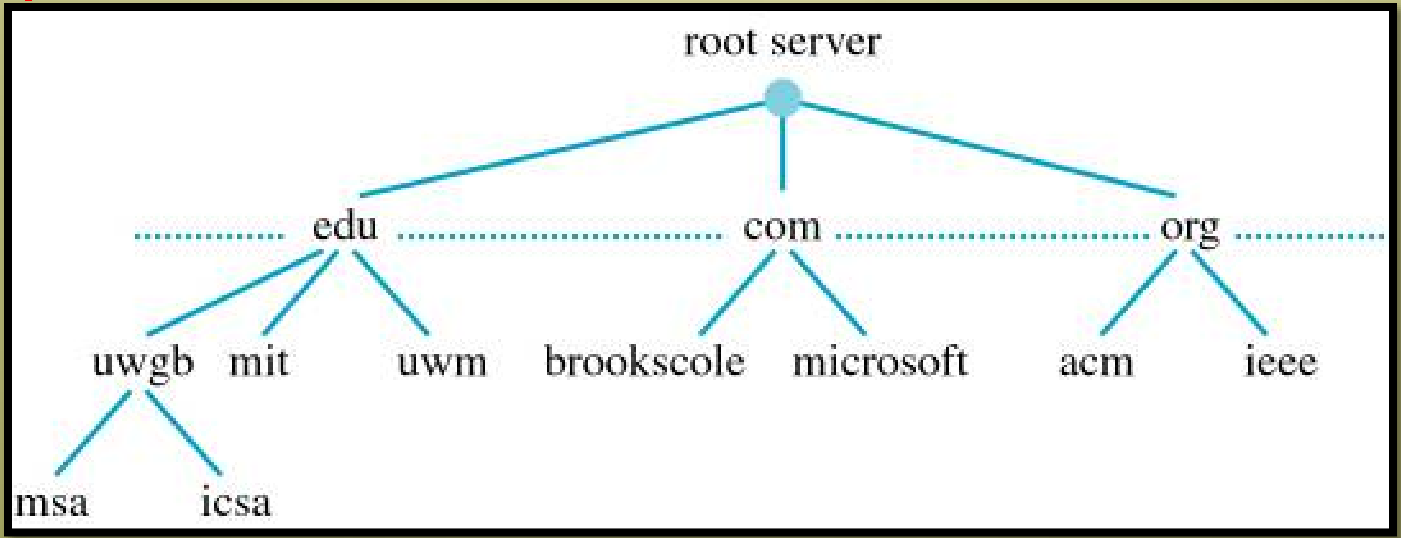
1) *distributed database* implemented in hierarchy of many *DNS servers*

2) *application-layer protocol:* which allows hosts to query the database

- name servers communicate to *resolve* names (address/name translation)
- note: core Internet function, implemented as application-layer protocol
- complexity at network's "edge"; possibly a substantial delay

DNS: domain name system

- To facilitate the lookup process, servers are organized into zones
- A request for a textual name escalates, and may also go down, until the IP address is found (or the process fails)
- *why not centralize DNS?*



DNS: services, structure

- **host aliasing**: specify domain name as an alias of another
  - i.e. point **ftp.example.com** and **www.example.com** (an FTP server and a webserver running on two ports from a single IP address) to **example.com**
  - such domain names are referred to a Canonical Names (**CNAME**s); the pointed-to domain name is an **alias**
  - a CNAME must point to another domain name; never to an IP address. In turn that pointed-to domain name points to the IP address
  - if the IP of the server ever changes, only one update to the DNS record is required for all CNAMEs

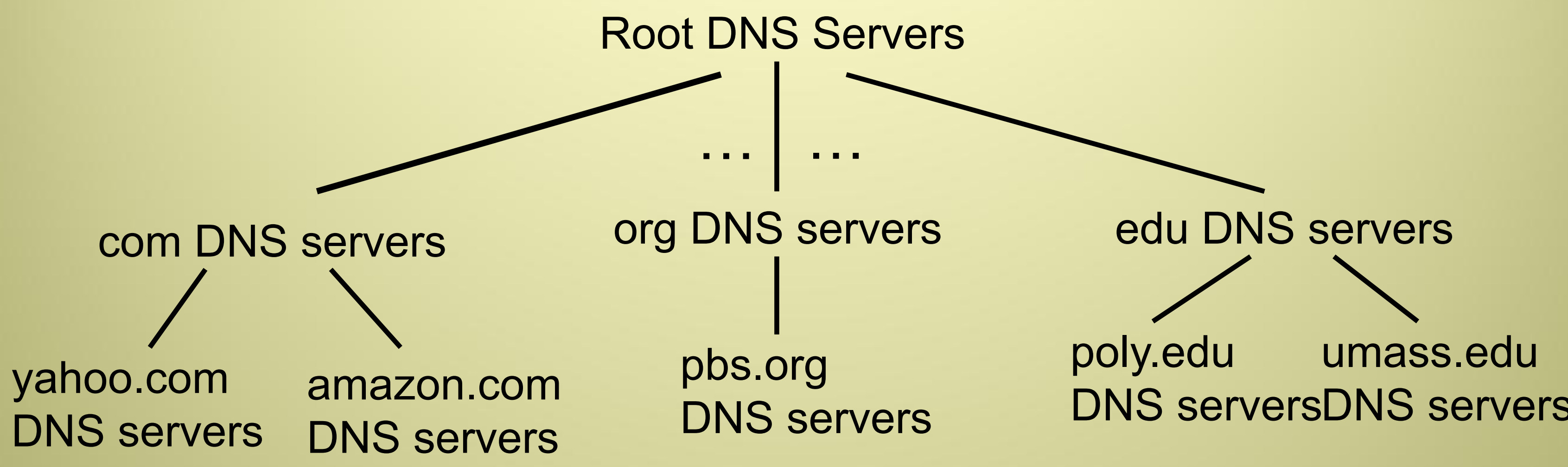
DNS: services, structure

- **mail server aliasing**
  - e-mail addresses need to be resolved as well; however the name e-mail server may be CNAME
  - i.e. an e-mail directed to linda@example.com may seem to require resolving example.com; however the mail server at example.com may actually be a CNAME (i.e relay.west-coast.example.com)
  - DNS can be invoked by e-mail application to obtain the CNAME of an alias hostname, as well as the IP address of the host

DNS: services, structure

- **load distribution**
  - heavy-loaded sites are often replicated over multiple servers
  - consequently, a *set* of IP addresses is associated with one hostname
  - DNS database contains the set of these IP addresses
  - the entire set is returned upon a query from a DNS client
  - client often sends its HTTP request to the first IP listed in the returned list
  - DNS rotations attempts to distribute the load over the replicated servers by rotating the order of the returned IP addresses

DNS: a distributed, hierarchical database

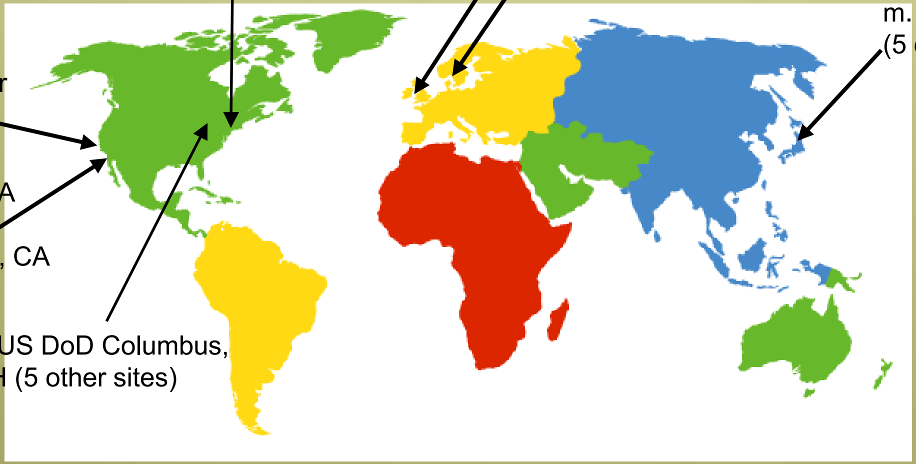


client wants IP for www.amazon.com; 1st approx:

- client queries root server to find com DNS server
- client queries .com DNS server to get amazon.com DNS server
- client queries amazon.com DNS server to get IP address for www.amazon.com

DNS: root name servers

- contacted by local name server that can not resolve name
- root name server:
  - contacts authoritative name server if name mapping not known
  - gets mapping
  - returns mapping to local name server



DNS root servers - 2012

13 root name "servers" worldwide
(**997** instances as of, July 11, 2019)

TLD & authoritative servers

top-level domain (TLD) servers:

- responsible for com, org, net, edu, aero, jobs, museums, and all top-level country domains, e.g.: uk, fr, ca, jp
- Verisign/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 name server

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

- when host makes DNS query, query is sent to its local DNS server
  - has local cache of recent name-to-address translation pairs (but may be out of date!)
  - acts as proxy, forwards query into hierarchy

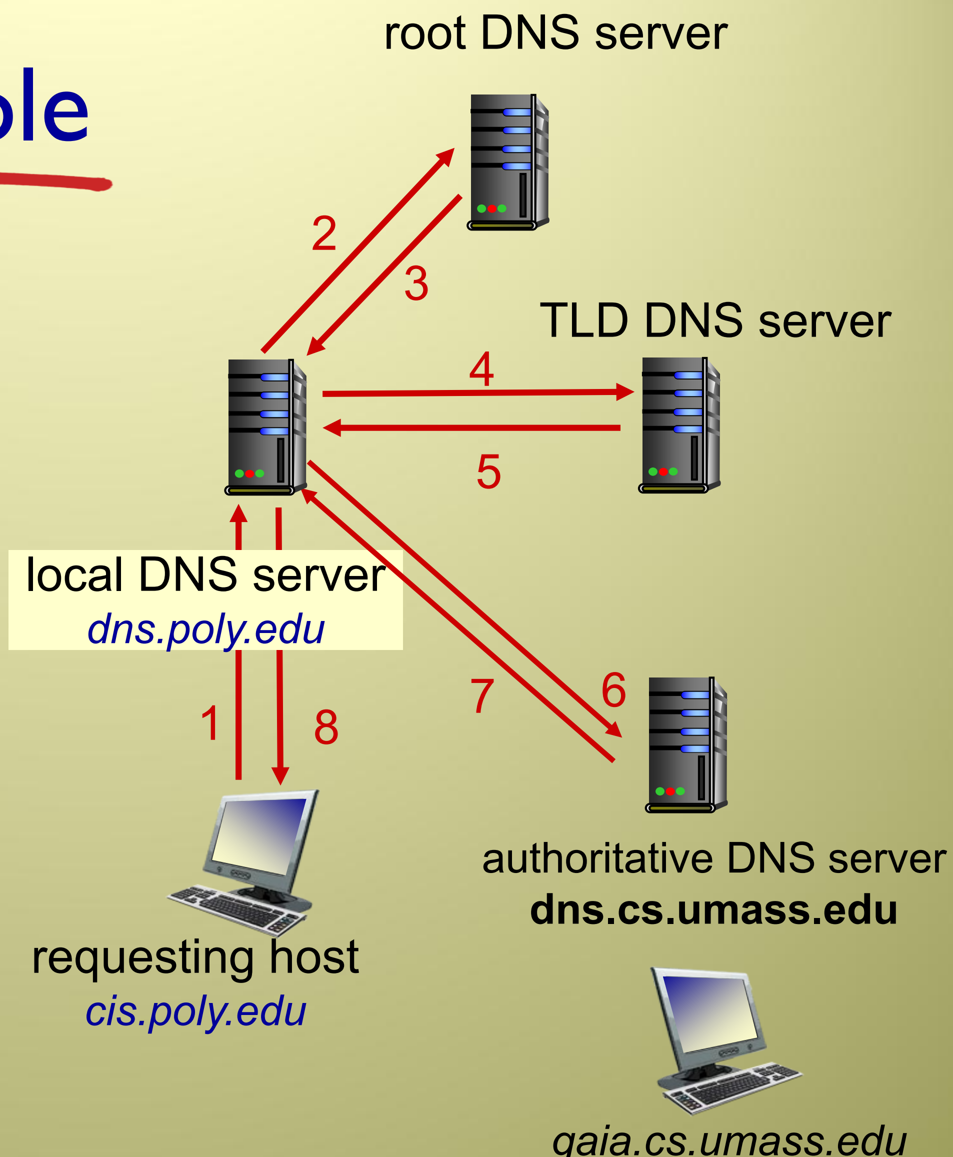
DNS name resolution example

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

Iterative query:

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

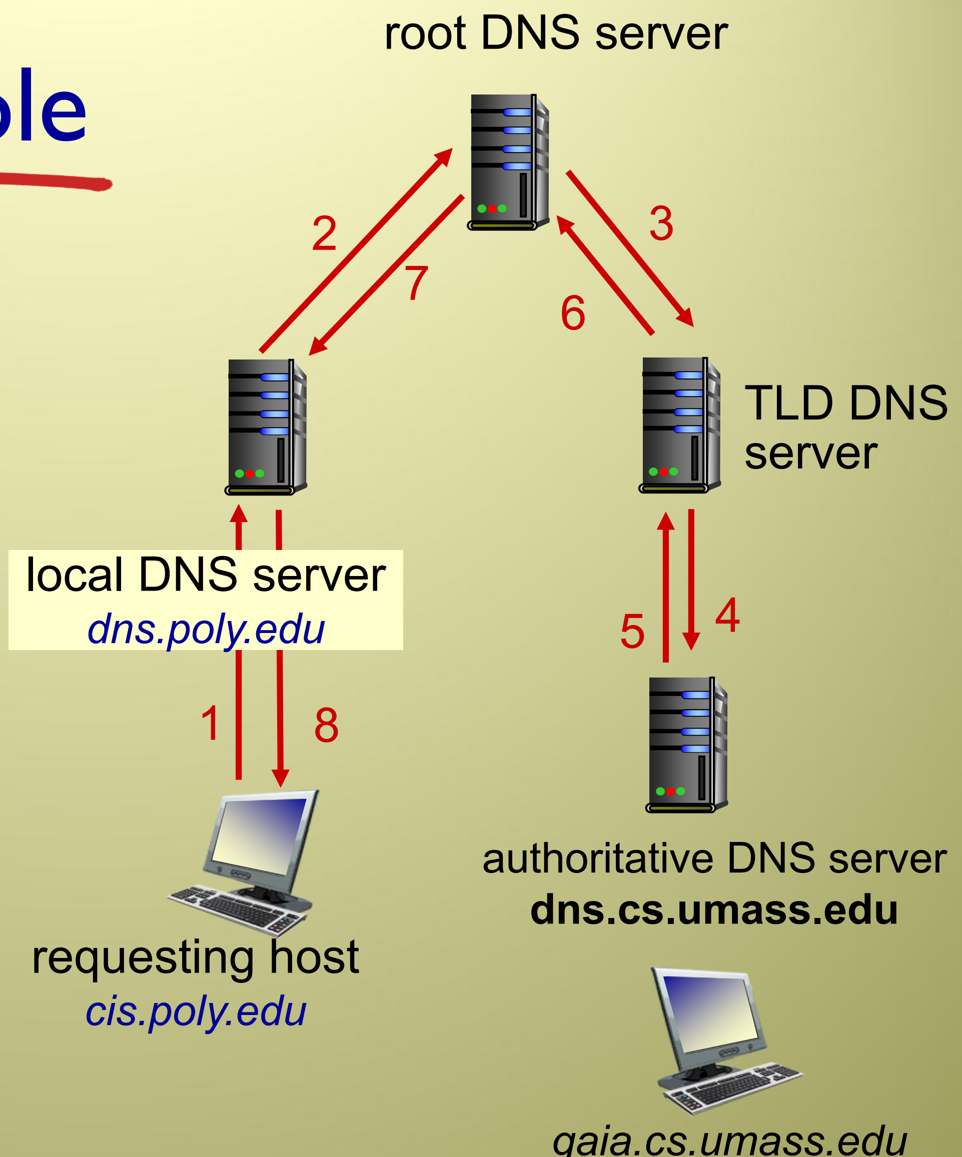
Note: call from cis.poly.edu is recursive; rest are iterative



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

- a critically important feature of DNS
- improve delay performance & reduce DNS messages in the Internet
- once (any) name server learns of an address mapping, it *caches* it
  - cache entries timeout (disappear) after some time (TTL)
  - TLD servers typically cached in local name servers
    - thus root name servers not often visited

DNS: caching, updating records

- cached entries may be *out-of-date* (best effort name-to-address translation!)
  - if name host changes IP address, may not be known Internet-wide until all TTLs expire

- update/notify mechanisms proposed IETF standard
  - RFC 2136

DNS records

DNS: distributed db storing resource records (RRs)

RR format: `(Name, Value, Type, TTL)`

type=A

- name is hostname (e.g, relay.west-coast.example.com)
- **`value`** is IP address

Example: (relay1.bar.foo.com, 145.46.93.9, A)

type=NS

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

Example: (foo.com, dns.foo.com, NS)

type=CNAME

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

Example: (foo.com, relay1.bar.foo.com, CNAME)

type=MX

- name is an alias host name
- value is a canonical name of mailserver associated with name

Example: (foo.com, mail.bar.foo.com, MX)

DNS records

- each DNS reply may carry one or **more** RRs

<u>Example</u>

❖ Assume the DNS server is an authoritative server for a particular hostname
  - server will return Type A RR
    e.g., (gaia.cs.umass.edu, 145.119.96.44, A)
  - Note: This type may be returned by any non-authoritative server as well (i.e. from a previously cashed record)

❖ Assume the DNS server is NOT an authoritative server for a particular hostname
  - server may return Type NS RR
    e.g., (umass.edu, dns.umass.edu , NS)

  As well as:
  e.g., (dns.umass.edu, 145.119.96.28, A)

DNS protocol, messages

- *query* and *reply* messages, both have same *message format*

msg header

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

<table>
<tr><th>← 2 bytes →</th><th>← 2 bytes →</th></tr>
<tr><td>identification</td><td>flags</td></tr>
<tr><td># questions</td><td># answer RRs</td></tr>
<tr><td># authority RRs</td><td># additional RRs</td></tr>
<tr><td colspan="2">questions (variable # of questions)</td></tr>
<tr><td colspan="2">answers (variable # of RRs)</td></tr>
<tr><td colspan="2">authority (variable # of RRs)</td></tr>
<tr><td colspan="2">additional info (variable # of RRs)</td></tr>
</table>

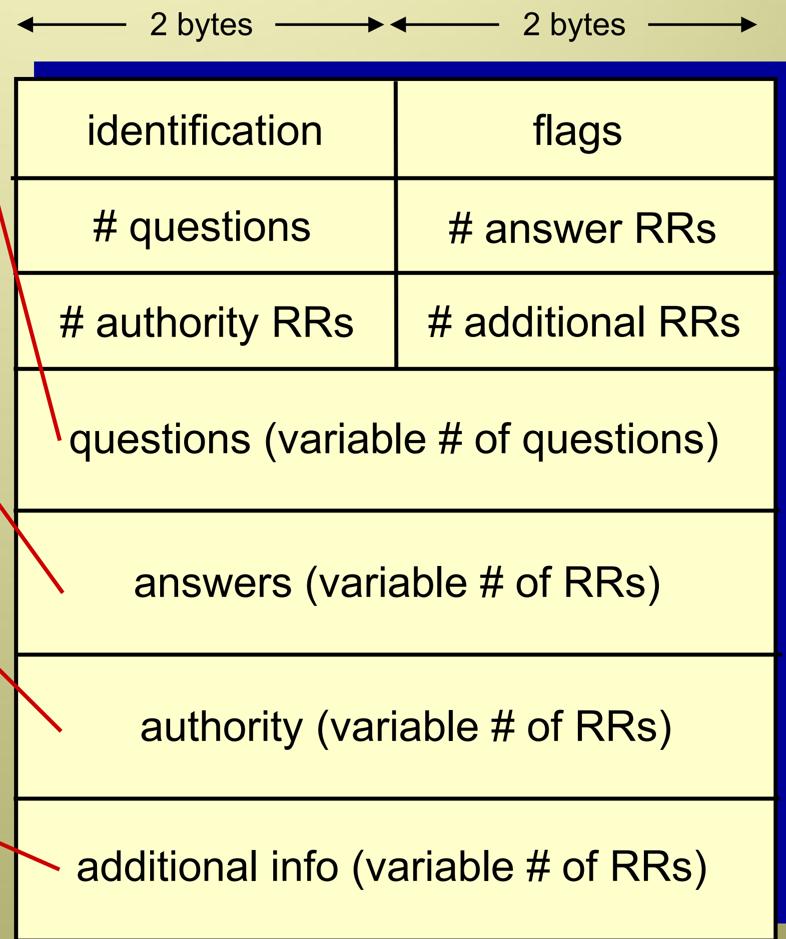
DNS protocol, messages

name, type fields for a query
i.e. host address associated with a name (type A), or the mail server for a name (type MX)

RRs in response to query
P/S: A reply may contain multiple RRs (i.e. for replicated web servers)

records for authoritative servers

additional "helpful" info that may be used
i.e. the answers section of a reply to an MX query include a RR providing the CNAME of the mail server. In the additional section may contain a type A RR providing the IP address of this CNAME



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 name 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)

- enter needed records into your authoritative DNS server:
  - type A record for www.networkuptopia.com;
  - type MX record for mail.networkutopia.com

Attacking DNS

DDoS attacks

- Bombard root servers with traffic
  - i.e. large-scale attack on October 21, 2002 using massive ***ping*** messages to the 13 root servers
  - Not successful to date
    - Traffic Filtering (configured to block *ping* messages)
    - Local DNS servers cache IPs of TLD servers, allowing root server bypass

- Bombard TLD servers
  - Potentially more dangerous
  - i.e. send a massive number of DNS quires; which will not be filtered
  - Severity can still be mitigated (at least partially) by caching in local DNS servers

Attacking DNS

Redirect attacks

- Man-in-middle
  - Intercept queries, then send bogus replies
- DNS poisoning
  - Send bogus replies to DNS server, which caches these incorrect replies
    - *Connections to the intended site can then be directed to the attacker's site*

Exploit DNS for DDoS (not directly an attack on DNS but on a targeted host)

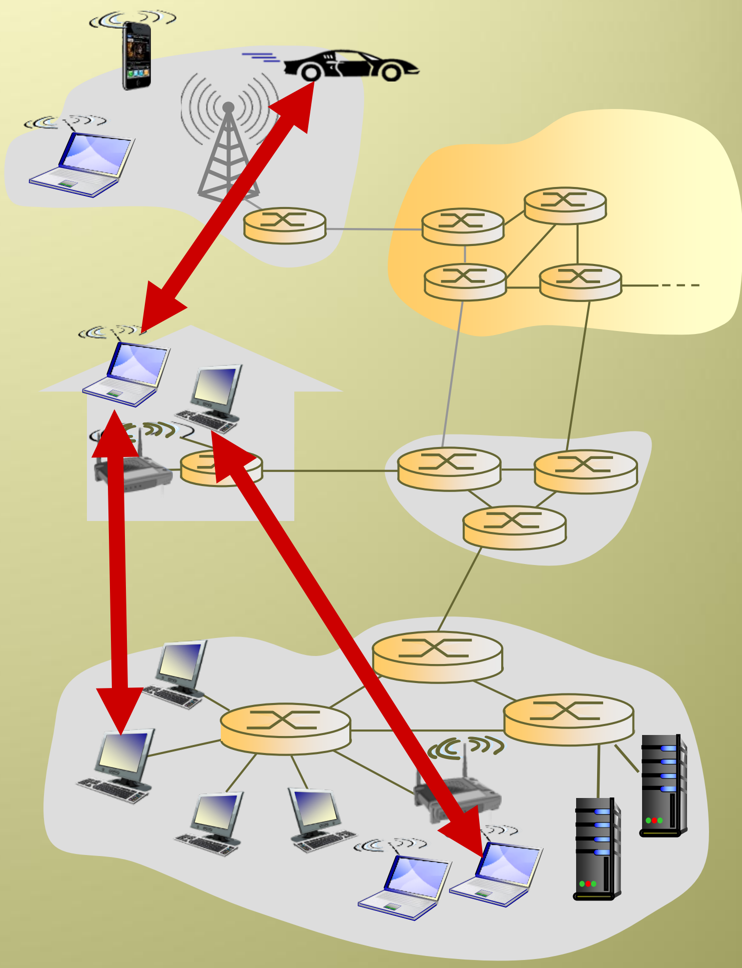
- Send queries to many authoritative DNS servers with spoofed source address as target IP
- Requires amplification (responses are much larger than queries), which overwhelm the target
  - Notice that this is done without the need for the attacker to generate much of its own traffic

Pure P2P architecture

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

examples:

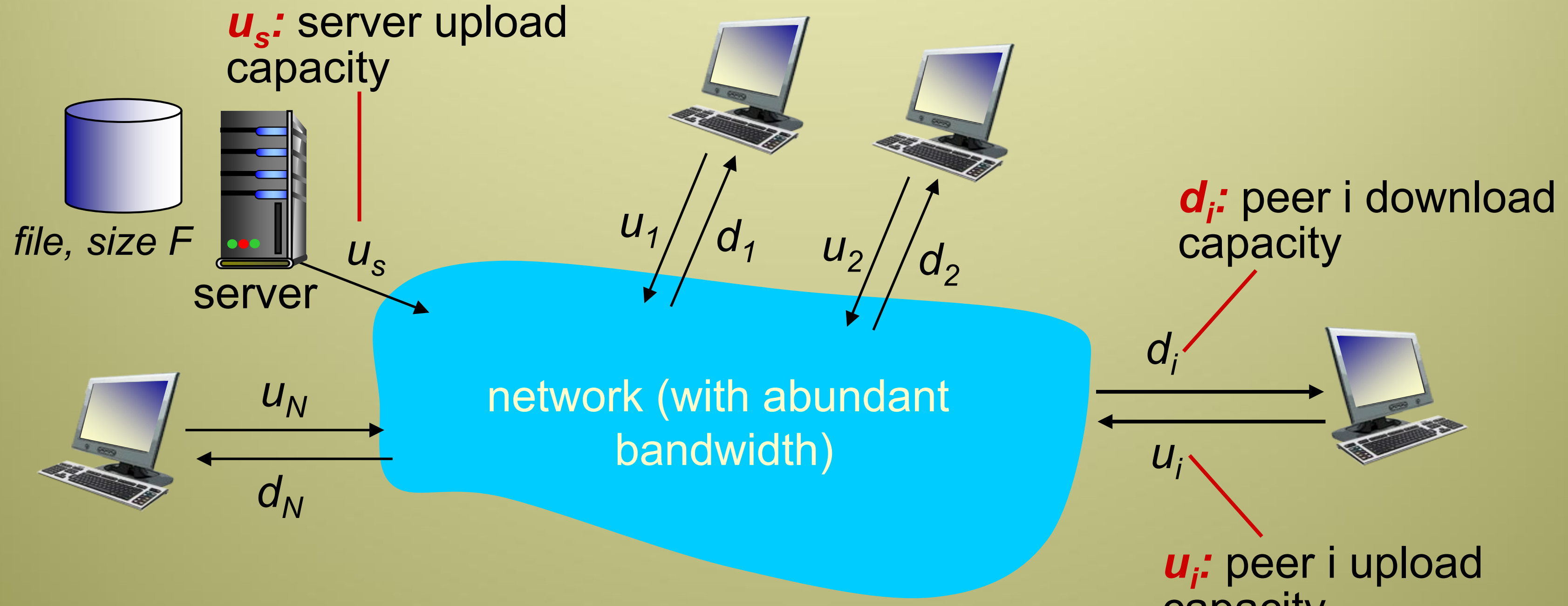
- file distribution (BitTorrent)
- Streaming (KanKan)



File distribution: client-server vs P2P

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

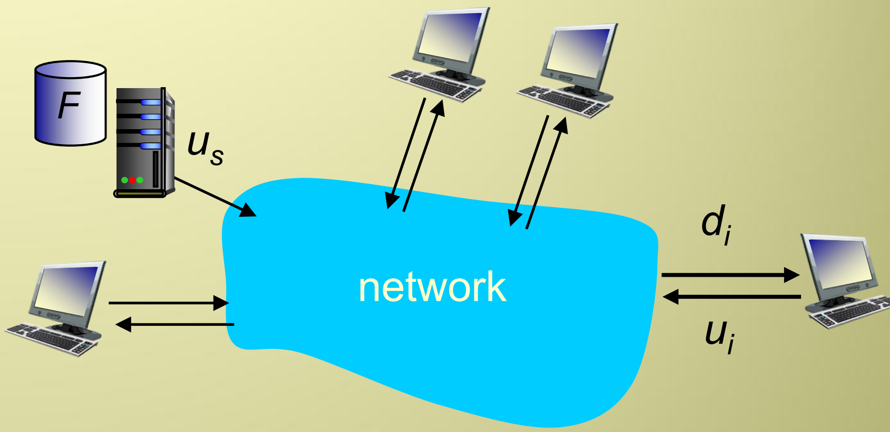
- peer upload/download capacity is limited resource



File distribution time: client-server

- *server transmission:* must sequentially send (upload) N file copies:
  - total transmission: FN bits
  - time to send one copy: F/u_s
  - time to send N copies is: NF/u_s

  ➔ distribution time is, at least: NF/u_s



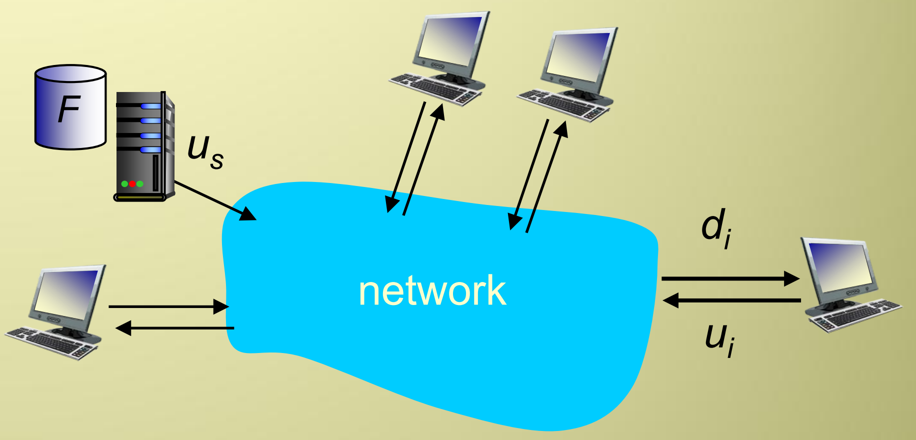
- *client:* each client must download file copy
- Let d_{min} denote the download rate of the peer with the slowest download rate
  - $d_{min} = min\{d_1, d_2, \ldots, d_N\}$
  - min-client download time: F/d_{min}

  ➔ distribution time is, at least: F/d_{min}

File distribution time: client-server

➔ To send, distribution time is, at least: NF/u_s

➔ To receive, distribution time is, at least: F/d_{min}



❖ Let D_{cs} denote distribution for client-server architecture

$$D_{cs} >= max\{ NF/u_{s,}, F/d_{min} \}$$

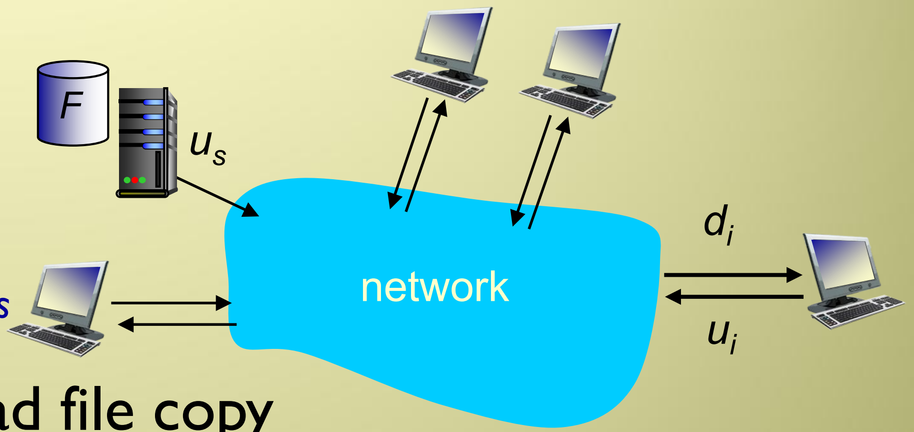
increases linearly in N

$$D_{cs} = max\{ NF/u_{s,}, F/d_{min} \}$$

Lower-bound (best case) distribution time

File distribution time: P2P

- each peer can assist the server in distributing the file
- *server transmission:* must upload at least one copy
  - ➔ time to send one copy: F/u_s



- *client:* each client must download file copy
  - min client download time: F/d_{min}
  - ➔ distribution time is, at least: F/d_{min}
- *Total upload capacity of the system as a whole:*

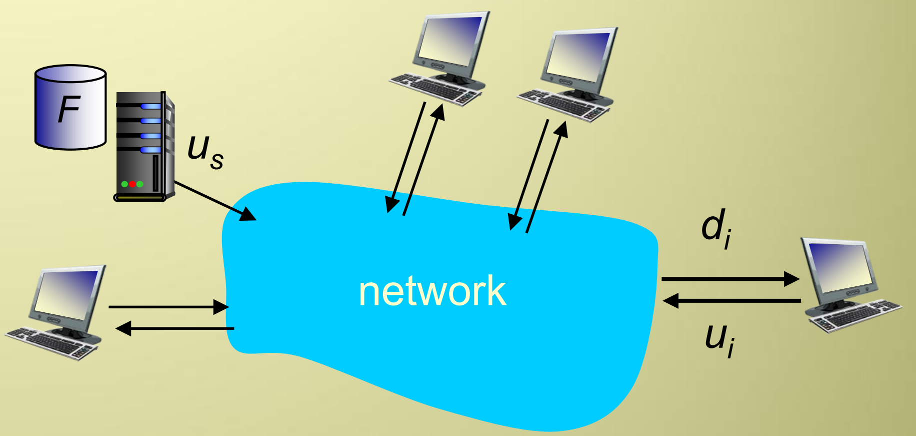
  $u_{total} = u_s + u_1 + u_2 + u_3 + \ldots + u_N$

- *System still needs to deliver F bits to N peers, totaling to **NF** bits, which cannot be done at a rate faster than $\mathbf{u_{total}}$*

➔ distribution time is, at least: **NF/** $\mathbf{u_{total}}$

File distribution time: P2P

- ➔ server time to send one copy: F/u_s
- ➔ client distribution time is, at least: F/d_{min}
- ➔ System-wide distribution time is, at least: NF/u_{total}



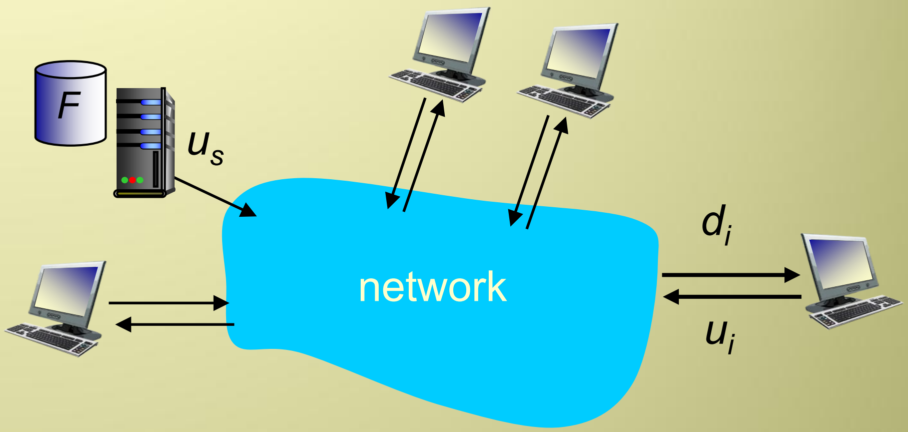
- ❖ Let D_{P2P} denote distribution for P2P architecture

$$\boxed{D_{P2P} >= max\{ F/u_s , F/d_{min} , NF/(u_s + \sum_{i=1}^{N} u_i) \}}$$

increases linearly in *N* …

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

File distribution time: P2P



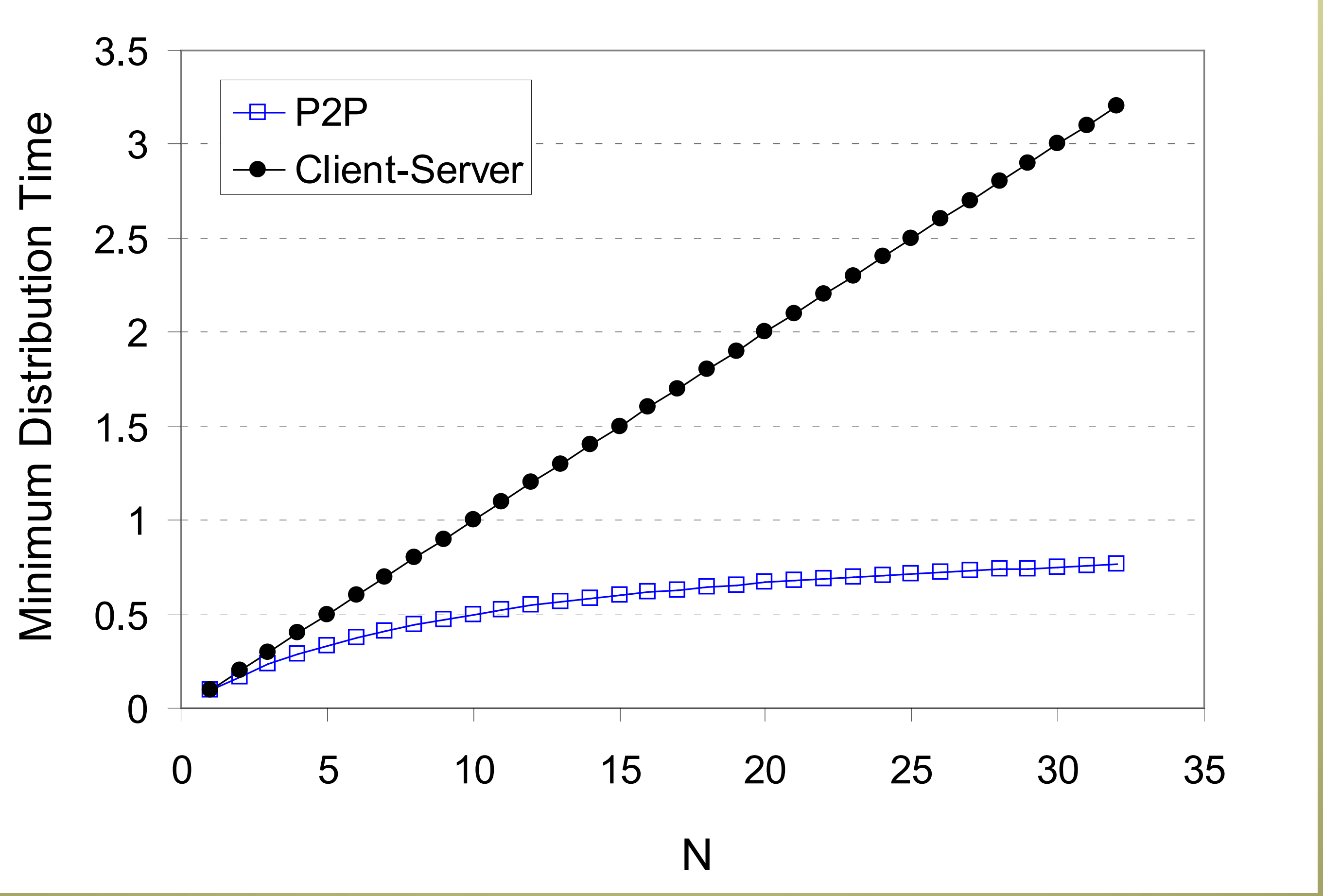
❖ Lower-bound (actual minimum distribution time) for P2P distribution can then be given as:

$$\boxed{D_{P2P} = \max\{\, F/u_s\,,\ F/d_{min}\,,\ NF/(u_s + \sum_{i=1}^{N} u_i)\,\}}$$

Lower-bound (best case) distribution time

Client-server vs. P2P: example

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

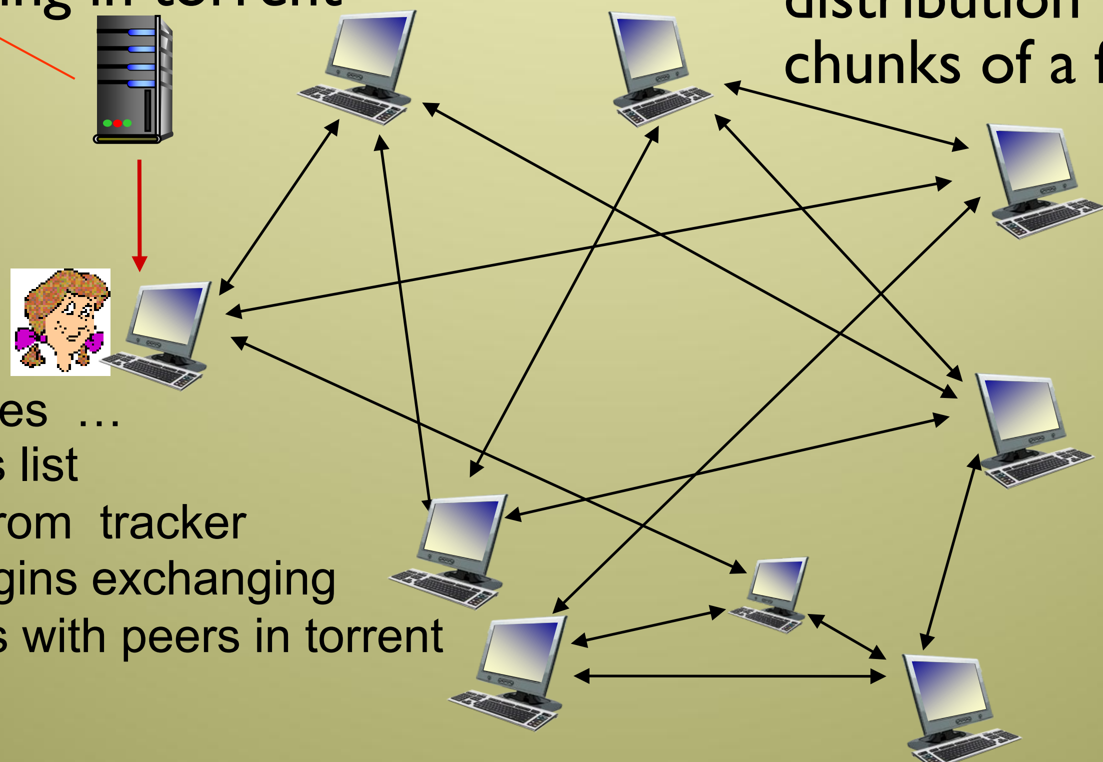


P2P file distribution: BitTorrent

- file divided into 256KByte chunks
- peers in torrent send/receive file chunks

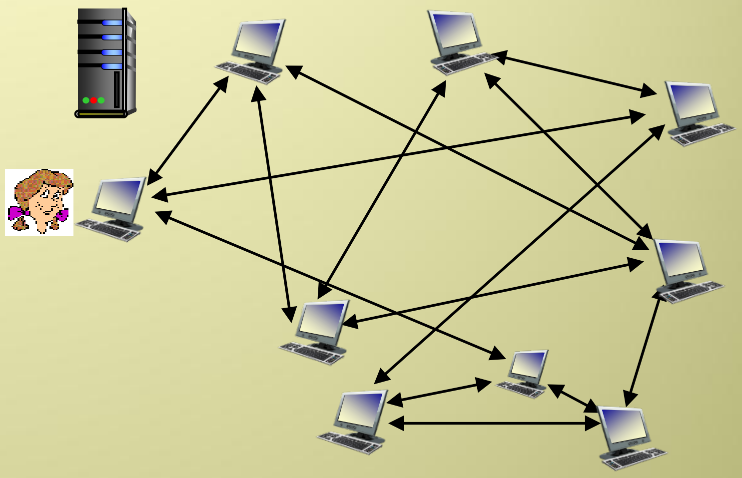
tracker: tracks peers participating in torrent

torrent: group of peers participating in the distribution (exchanging chunks of a file)



Alice arrives …
… obtains list
of peers from tracker
… and begins exchanging
file chunks with peers in torrent

P2P file distribution: BitTorrent



- peer joining torrent:
  - has no chunks, but will accumulate them over time from other peers
  - registers with **tracker** to get list of peers, connects to subset of peers ("neighbors")
- while downloading, peer uploads chunks to other peers
- peer may change peers with whom it exchanges chunks
- *churn:* peers may come and go
- once peer has entire file, it may (selfishly) leave or (altruistically) remain in torrent

BitTorrent: requesting, sending file chunks

requesting chunks:

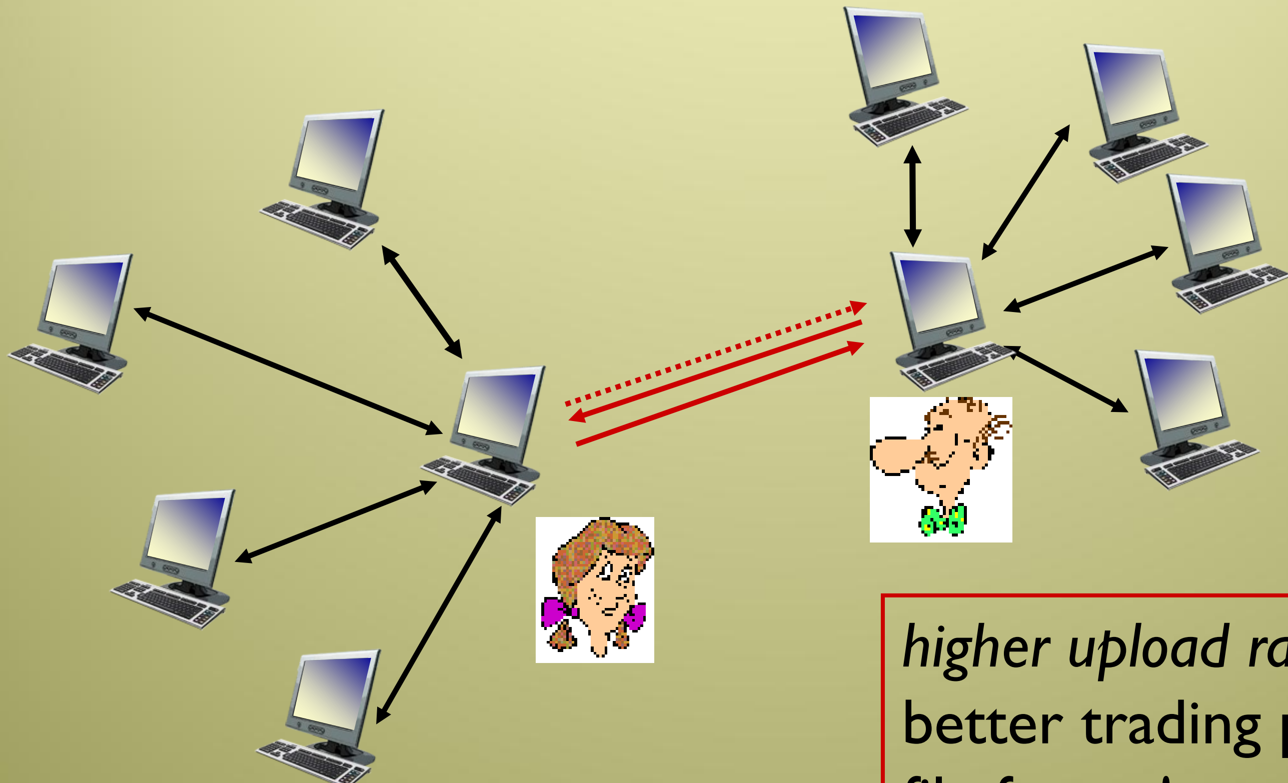
- at any given time, different peers have different subsets of file chunks
- periodically, Alice asks each peer for list of chunks that they have
- The provided list from each of the peers allows Alice to make 2 important decisions:
  - which chunks should she request first (**rarest first**)
  - to which of her neighbors should she send requested chunks

sending chunks: tit-for-tat

- Alice sends chunks to the four peers currently sending her chunks *at highest rate*
  - other peers are **choked** by Alice (do not receive chunks from her)
  - re-evaluate top 4 every10 secs
- every 30 secs: randomly select another 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 !