Overlay Networks

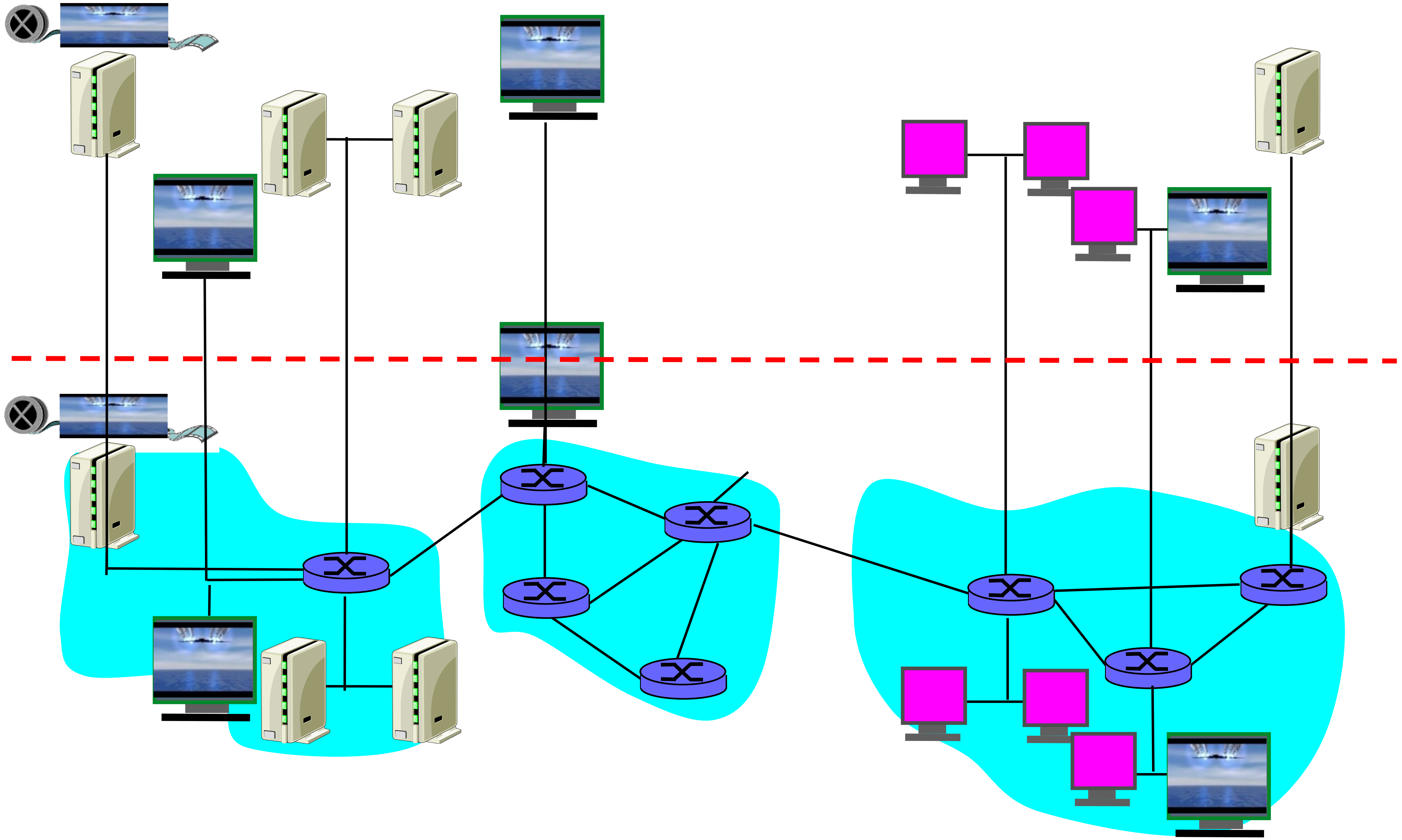
Soheil Abbasloo
Department of Computer Science
University of Toronto

Fall 2022

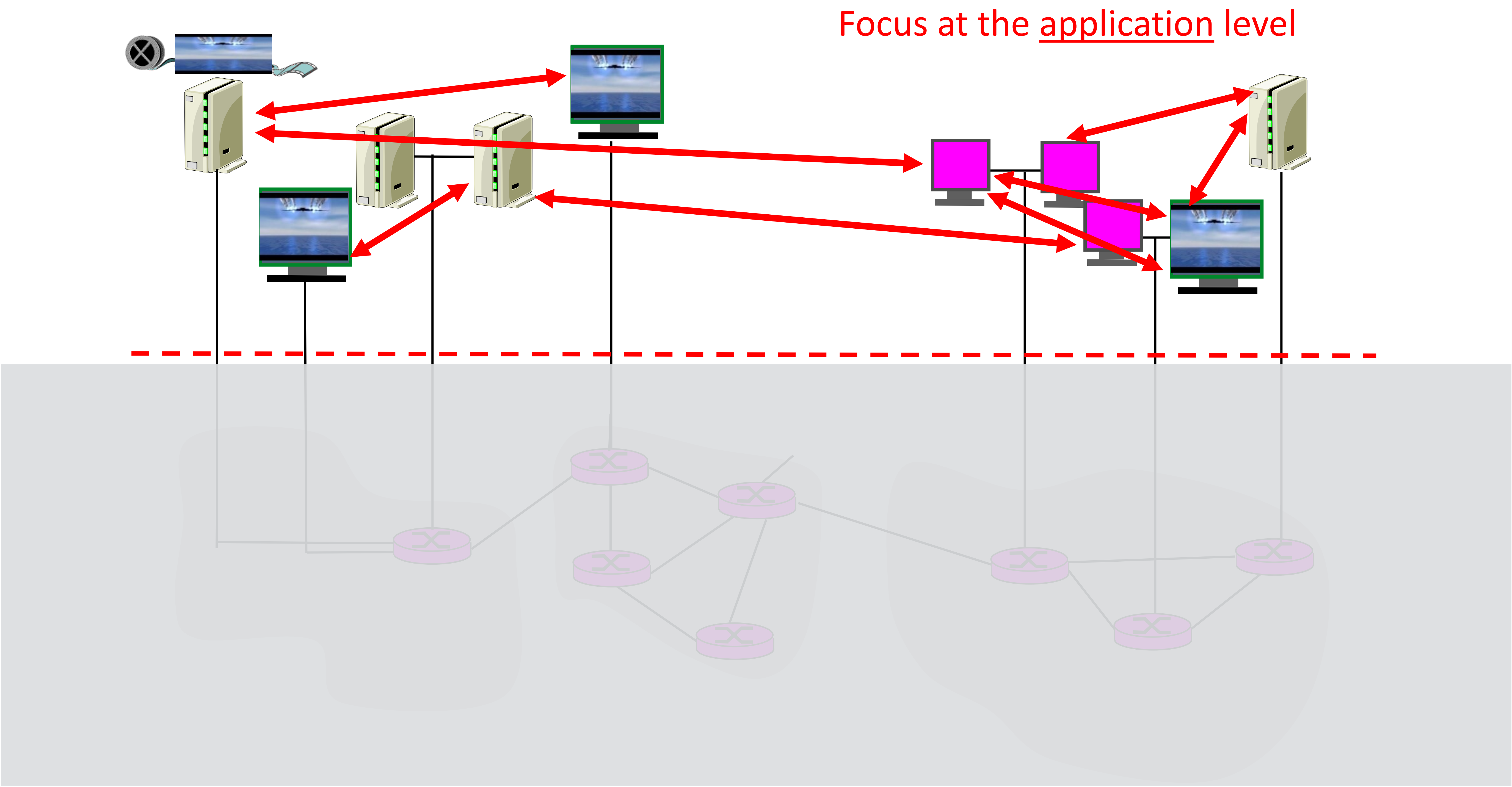
Outline

- Overlay Networks
- Routing Overlays
- P2P Networks
- Logistics of Final Exam

Overlay Networks



Overlay Networks



Overlay Networks

- A logical network built on top of a physical network
- Many logical networks may coexist at once
  - Over the same underlying network
  - And providing its own particular service
- Nodes are often end hosts
  - Acting as intermediate nodes that can forward traffic
  - Providing a service, such as access to files

Examples of Overlays

- Content Delivery Networks (CDNs)
  - Akamai (total asset: $8.12 billion)
  - Cloudflare
- Routing Overlays
  - Border Gateway Protocol (BGP) routers (with their peering relationships)
  - Application-level multicast
- P2P applications
  - Napster
  - Gnutella
  - BitTorrent

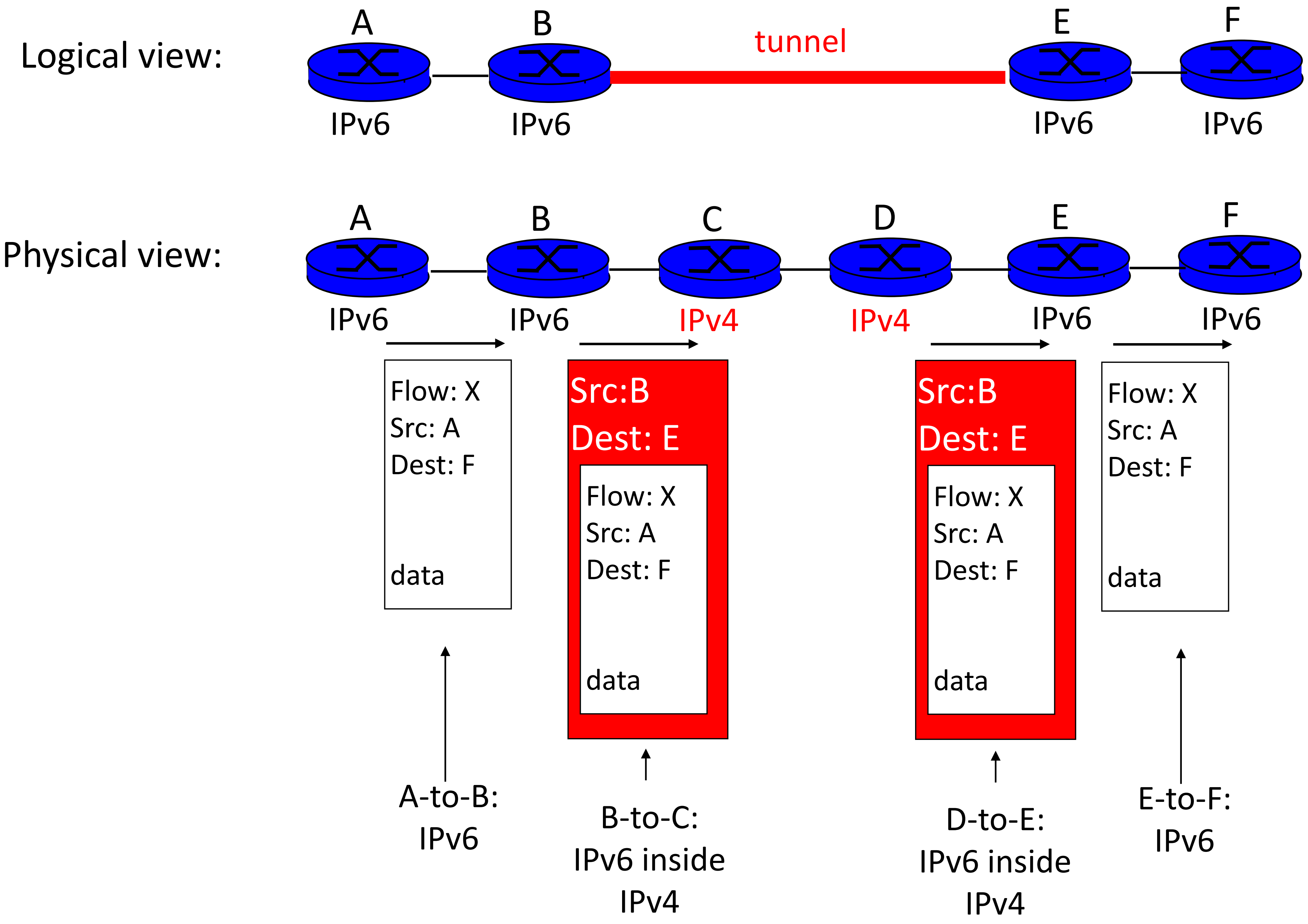
Routing Overlays

Routing Overlays

- Alternative routing strategies
  - No application-level processing at the overlay nodes
  - Packet-delivery service with new routing strategies
- Incremental enhancements to IP
  - IPv6
  - Security
  - Multicast
- Revisiting where a function belongs
  - End-system multicast: multicast distribution by end hosts

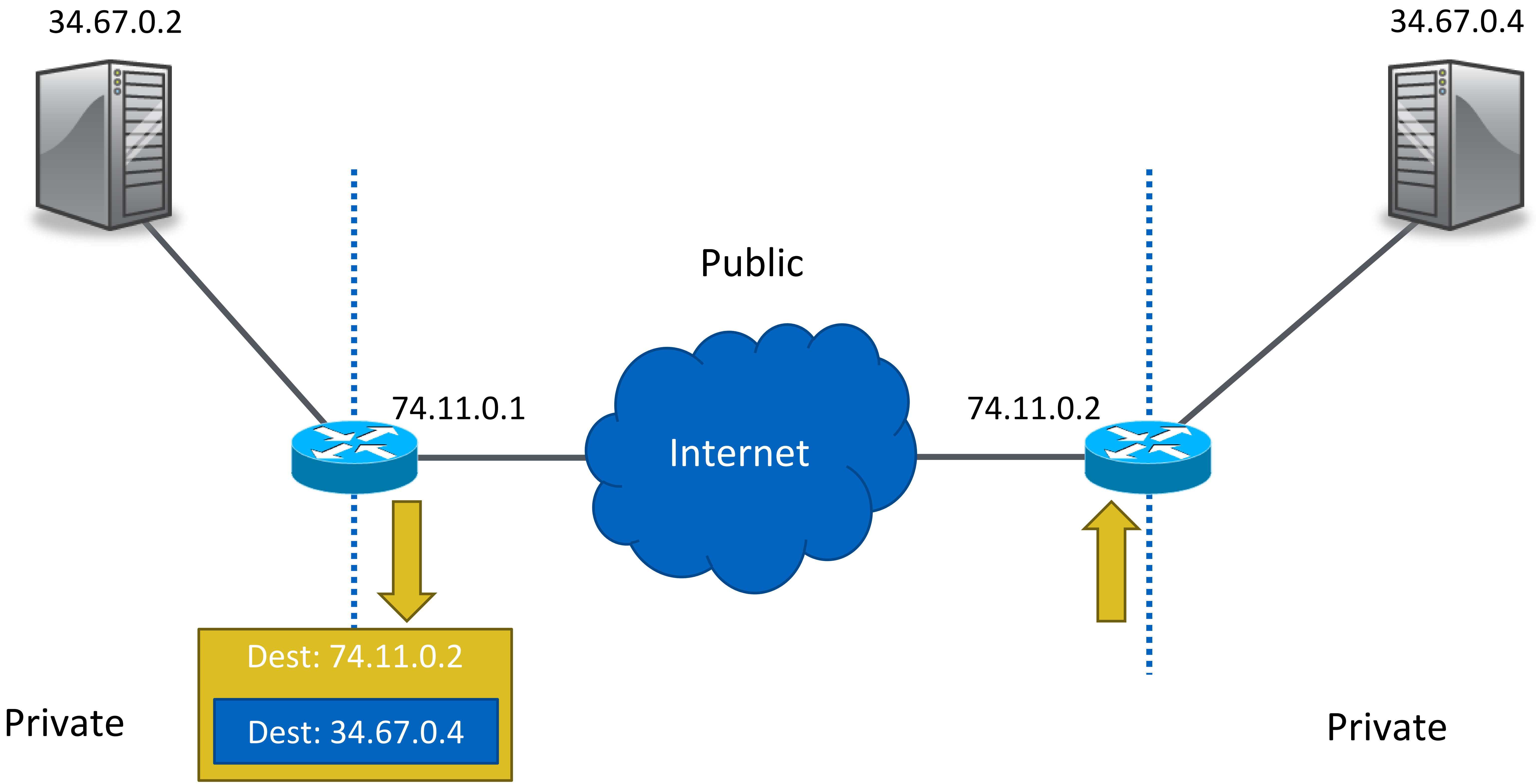
Example: Incremental Deployment

6Bone Deploying IPv6 over IP4



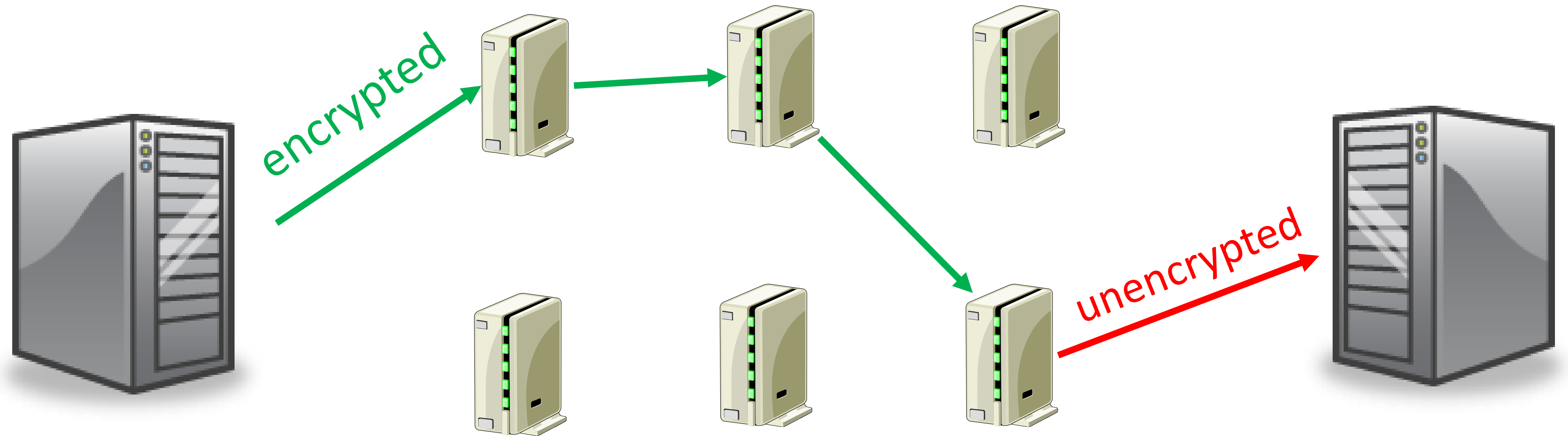
Example: Security

VPN



Tor Project

- An overlay to enhance anonymity and privacy
  - Volunteer operated servers
- How Tor Works
  - Obtain a list of Tor nodes from a directory
  - Pick a random path to destination server
  - Select a different path for other servers



Example: Multicast

Definition

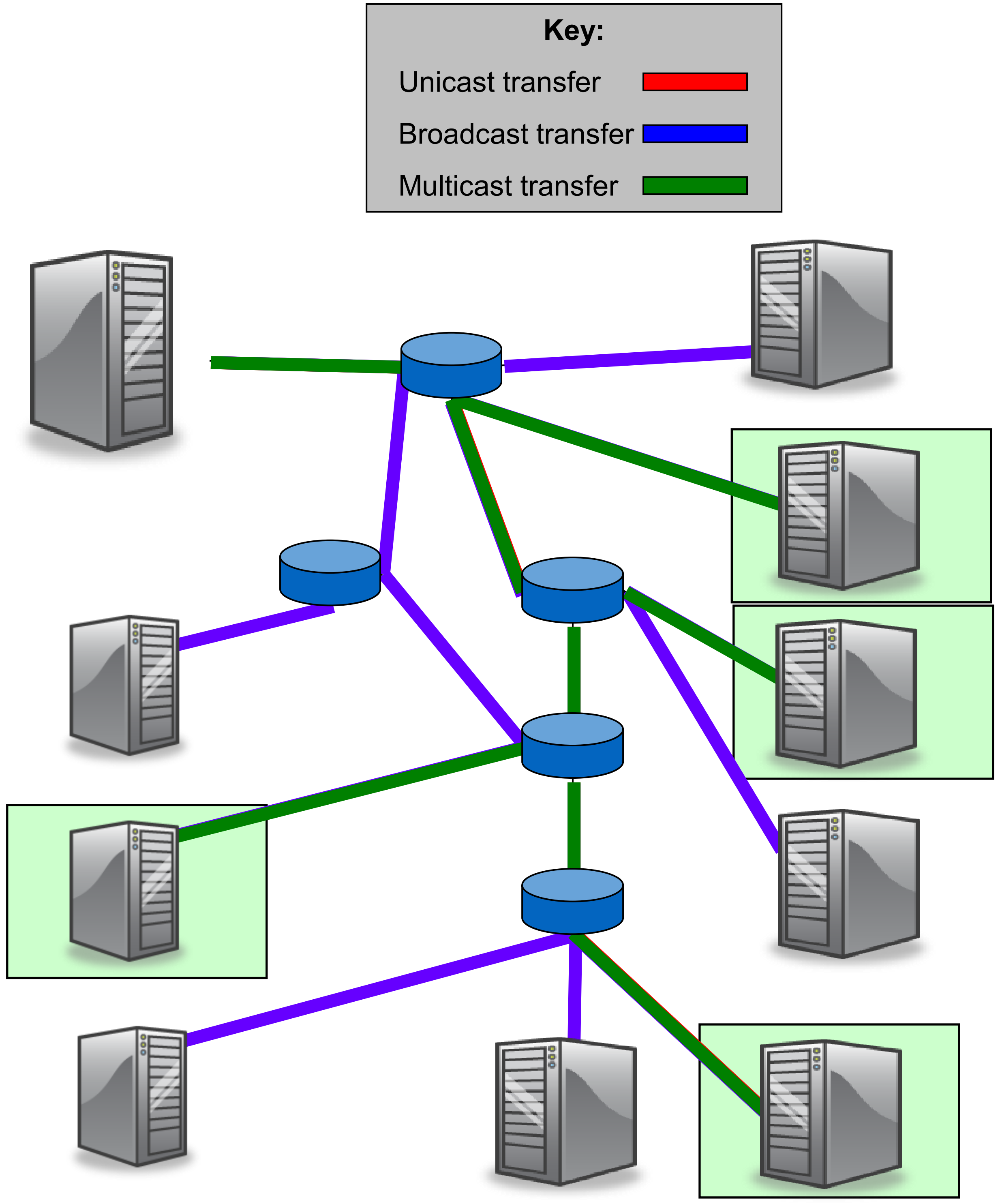
- Unicast
  - One-to-one
  - Destination – unique receiver host address
- Broadcast
  - One-to-all
  - Destination – address of network

- Multicast
  - One-to-many
  - Multicast group must be identified
  - Destination – address of group

Why not simply use unicast multiple times?!

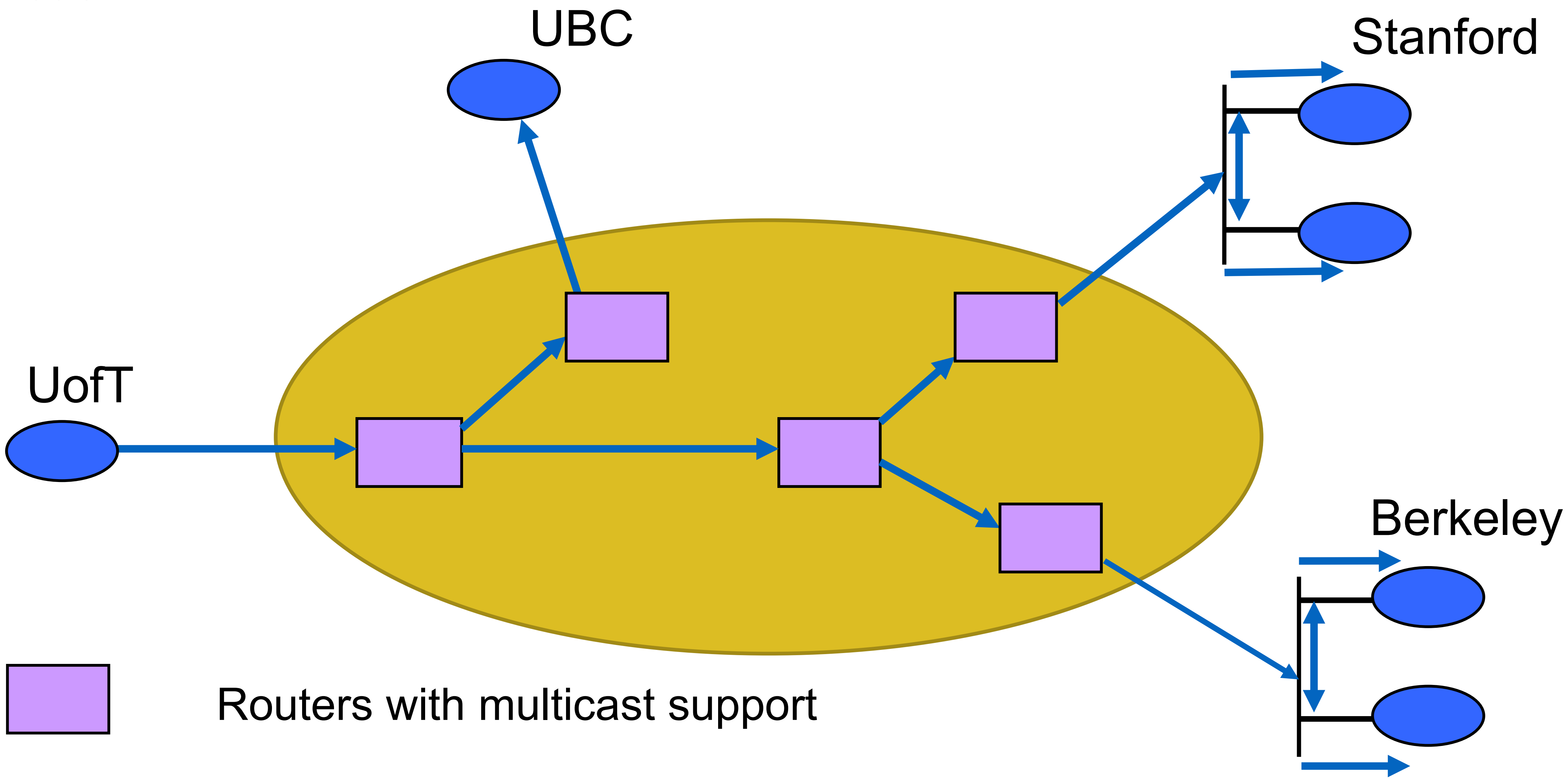
Lots of applications:

Delivery of news, stock quotes
Remote conferencing
Streaming audio and video to many participants …



Example: Multicast

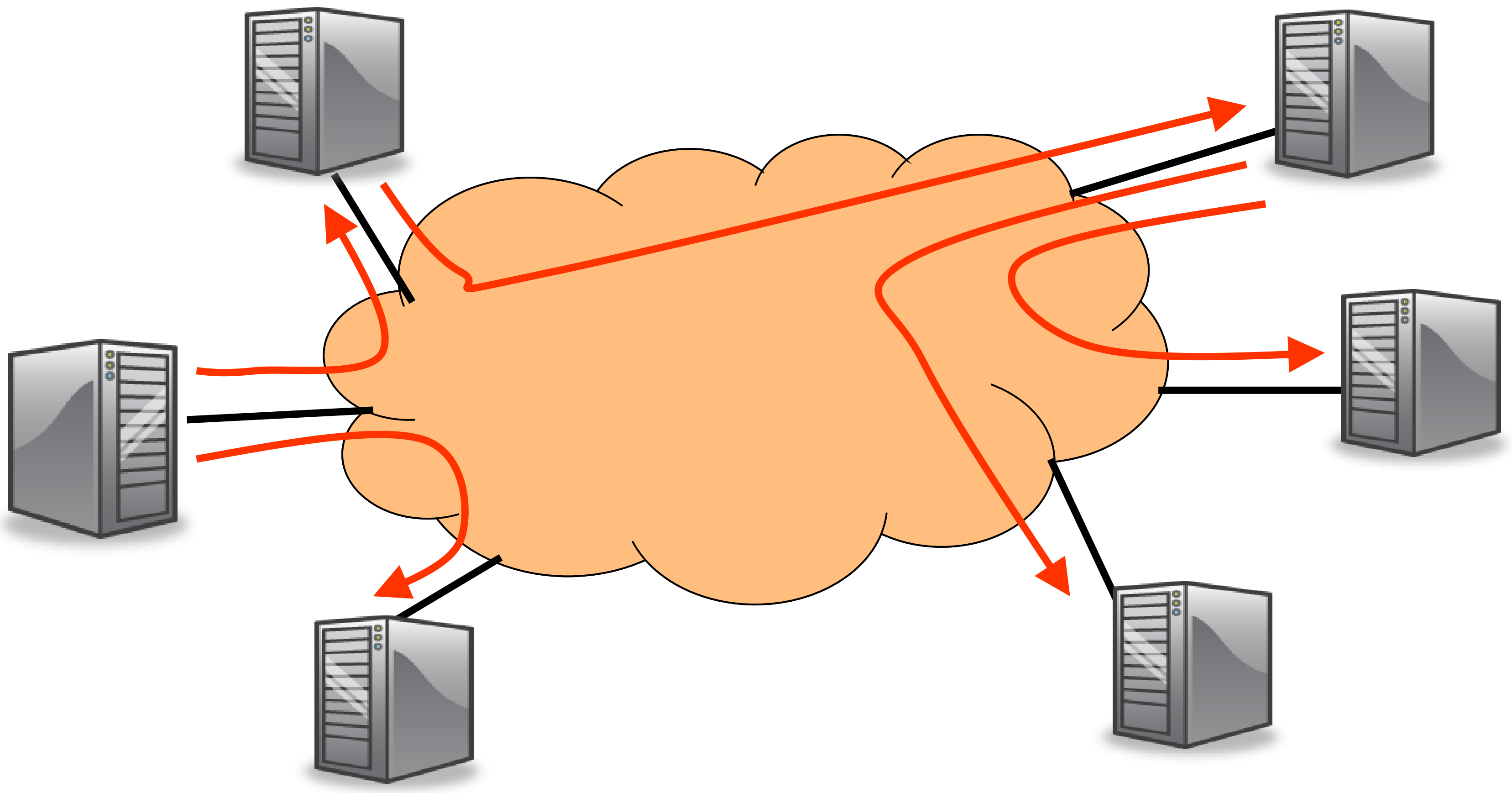
IP multicast



Highly efficient bandwidth usage **BUT**
Technical and business challenges
Should multicast be a network-layer service?

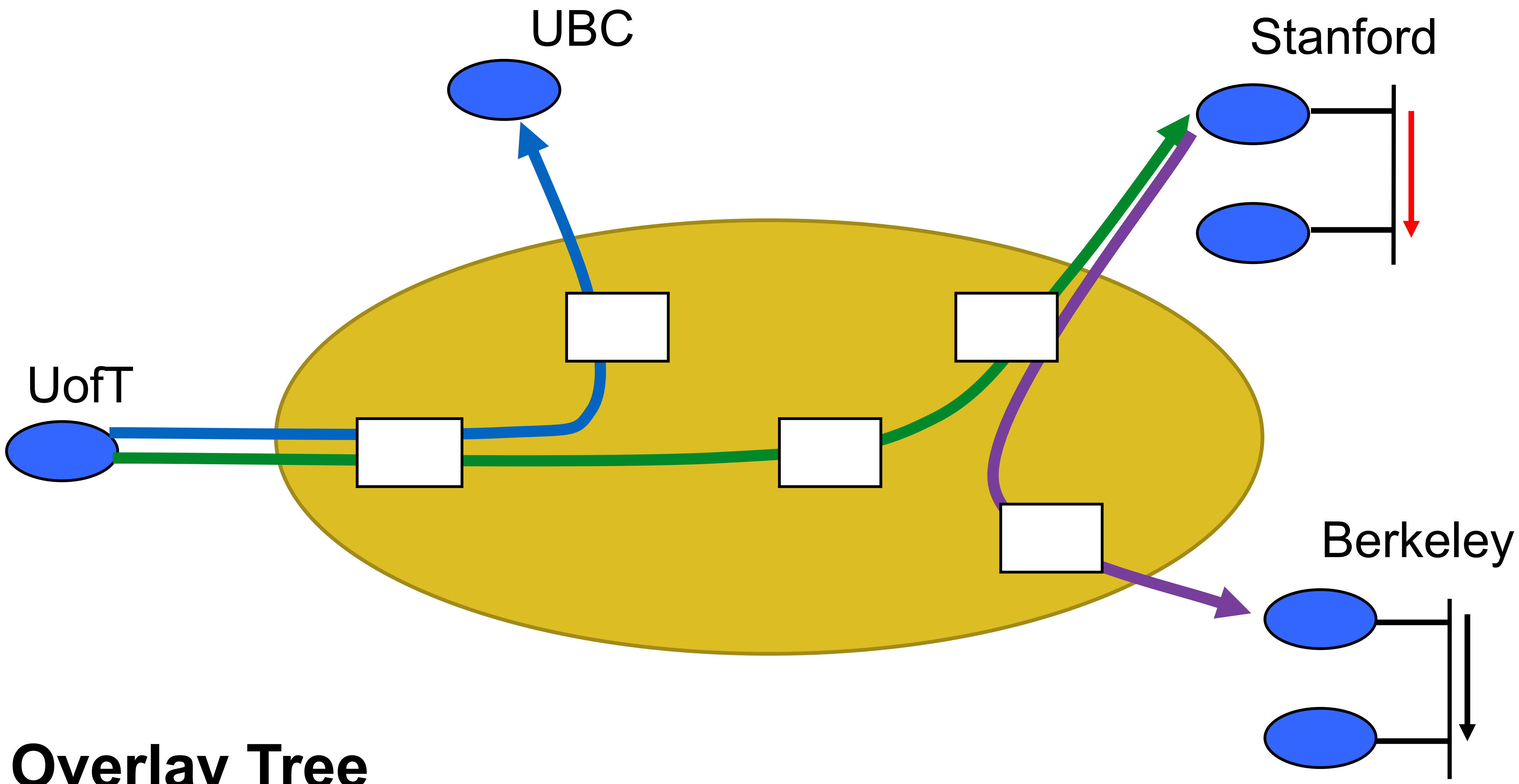
End-Host-Based Multicast

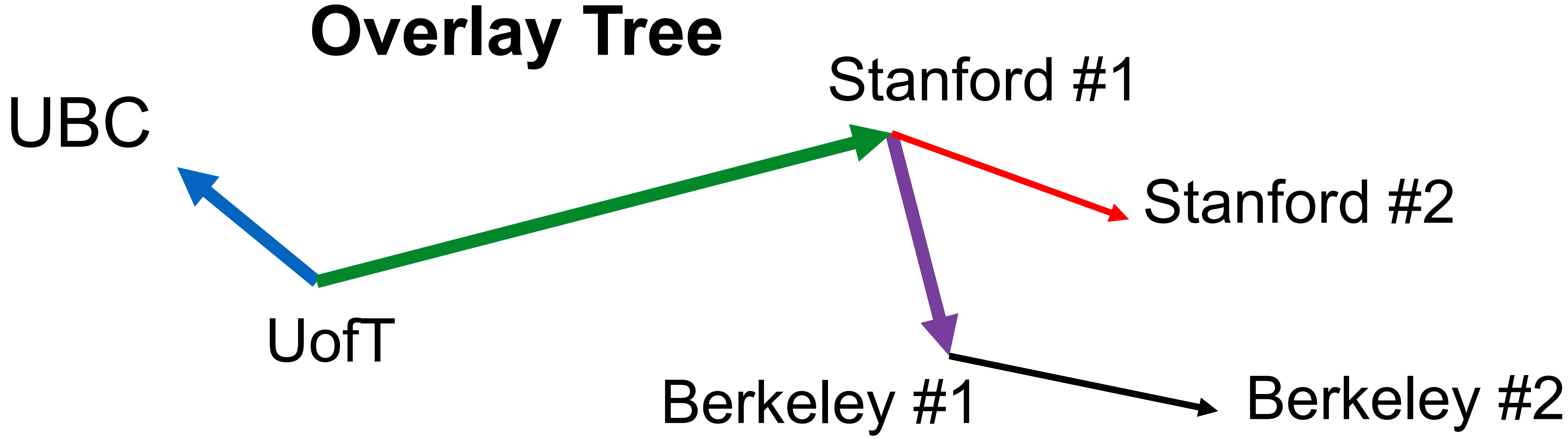
- Making a **logical** network on top of the existing physical network
- Multicast tree of end hosts
  - Allow end hosts to form their own multicast tree
  - Hosts receiving the data help forward to others



Example: Multicast

End-Host-Based multicast





Question

- Assume there are no services like Netflix, Spotify, …

- You wanna start a new file sharing business to provide **free content**.
- How do you architect that?

- Assume nothing is illegal!

P2P Networks

Peer-to-Peer

- Client-server: Traditional Internet Service model
  - Many clients, 1 (or more) server(s)
  - Web servers, file downloads, video streaming
- **P2P:** A simple alternative
  - Users bring their own resources to the table
  - A cooperative model: clients = peers = servers
- **How did it start?**
  - A killer application: file distribution
  - Free music over the Internet! (*not exactly legal…*)
- Key idea: share storage, content, and bandwidth of individual users
  - Lots of them
- Big challenge: coordinate all of these users

Peer-to-Peer:
3 Key Requirements

- Help users determine what they want
  - Some form of search
  - P2P version of Google
- Locate that content
  - Which node(s) hold the content?
  - P2P version of DNS (map name to location)
- Download the content
  - Should be efficient
  - P2P form of Akamai

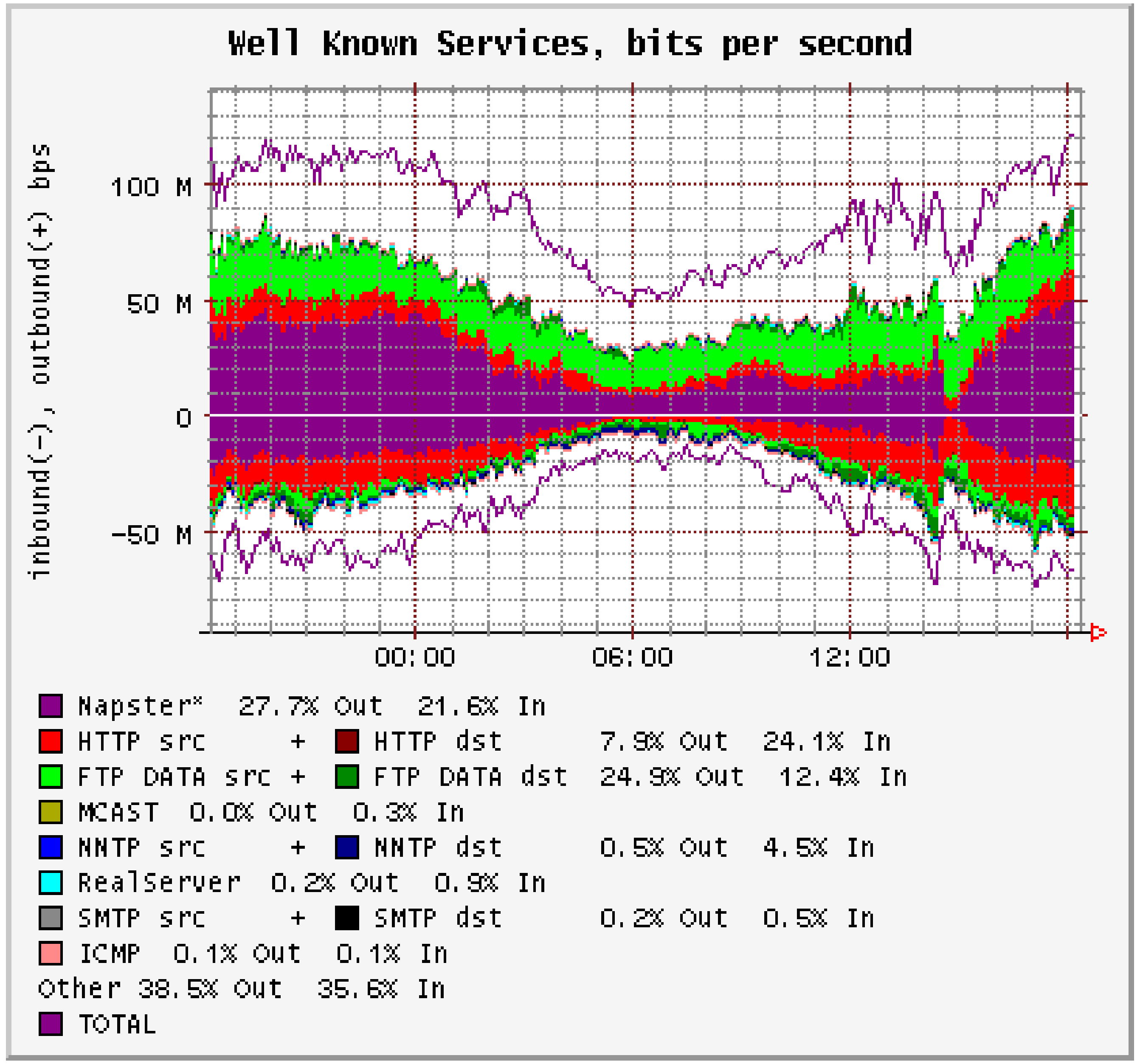
Peer-to-Peer Networks:

napster

- The rise (Shawn Fanning, Northeastern freshman)
  - January 1999: Napster version 1.0
  - May 1999: company founded
  - 2000: 80 million users
  - 2000: ~28% of Uwisc's traffic!*

- The fall
  - September 1999: first lawsuits
  - Mid 2001: out of business due to lawsuits
  - Mid 2001: dozens of P2P alternatives
  - 2003: growth of pay services like iTunes

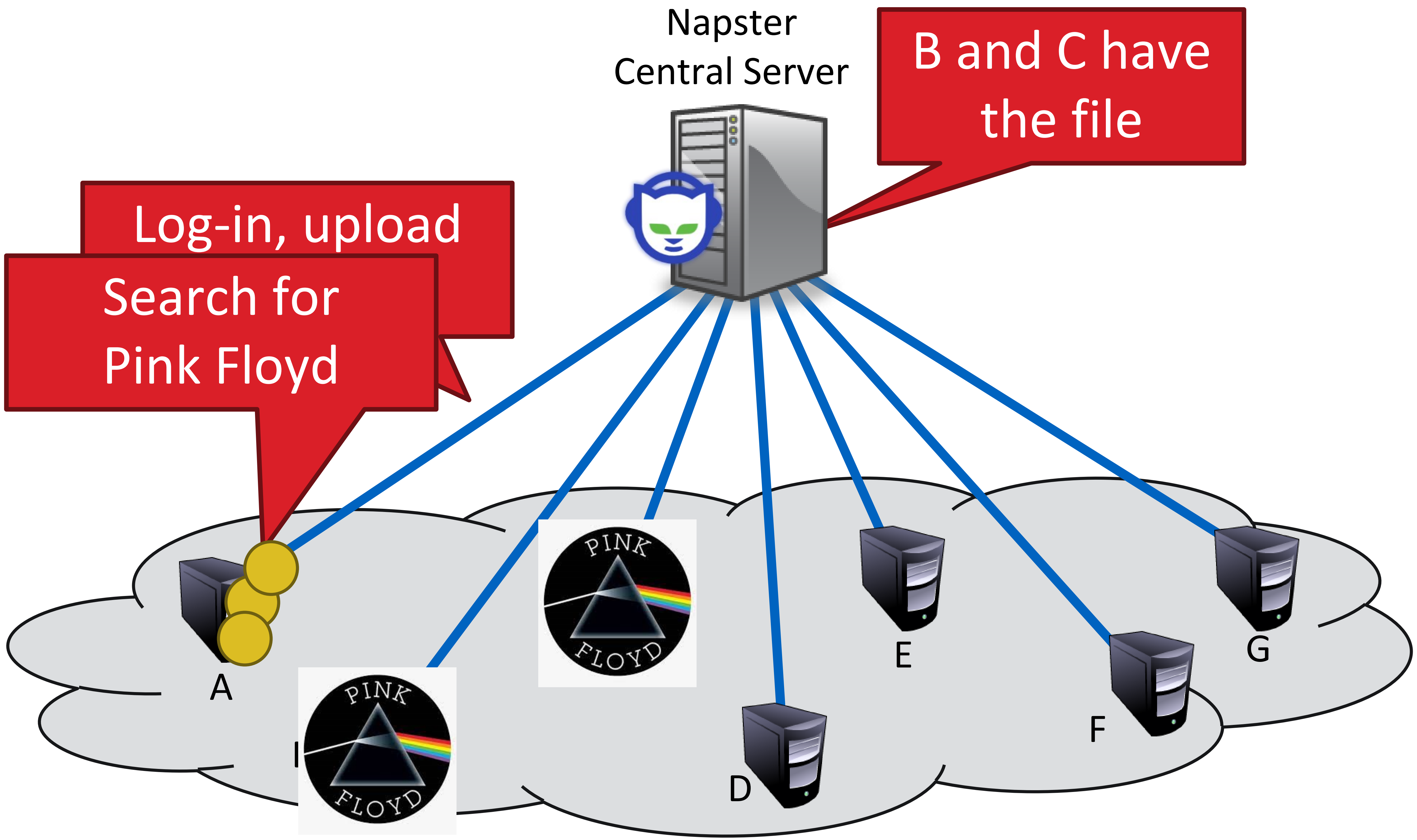
- Instructive for what it got right
- And wrong …
- Had a powerful economic message …
- And also, a legal one …



*https://pages.cs.wisc.edu/~plonka/lisa/FlowScan/

Napster Centralized Architecture

Overview



Overview

- **Centralized:** Search & Location
  - Centralized index server
  - Centralization of liability
- **Decentralized:** Download
  - Decentralized/p2p file transfer
  - No load on the server

- Advantages:
  - Simple
- Disadvantages:
  - Single point of failure (technical and … legal!)
  - The latter is what got Napster killed

Unstructured P2P Networks

- Fully centralized systems:
  - single points of failure

- **Response:** fully unstructured P2P
  - No central server, peers only connect to each other
  - Queries sent as controlled flood

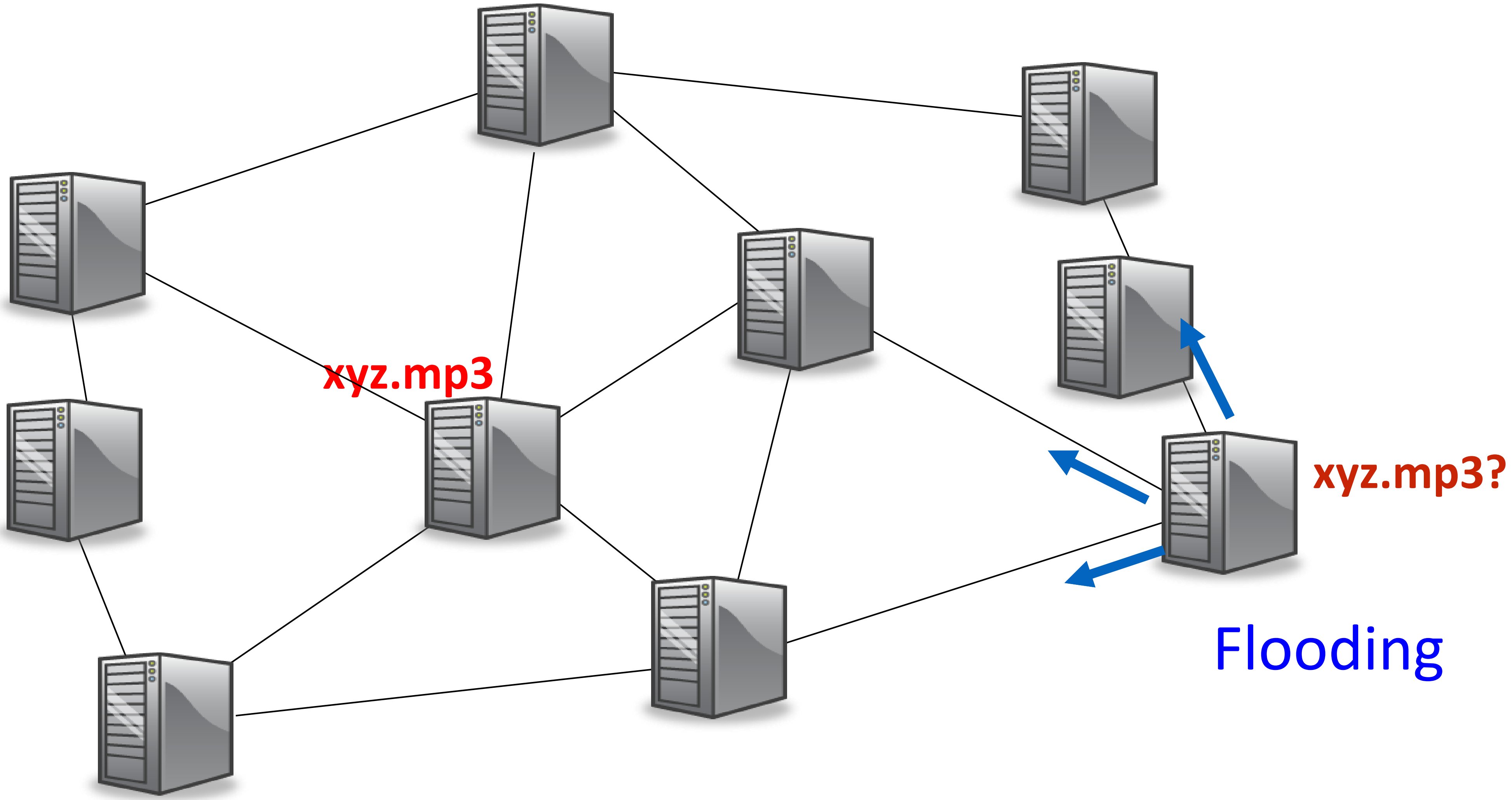
Gnutella:
First decentralized p2p

- Justin Frankel, Nullsoft, 2000
  - AOL was not happy at all!
- Original design: a flat network using **Query flooding**

- Join via bootstrap node
  - pre-existing address list, or web caches of known nodes, …
- Connect to random set of existing hosts (“neighbors”)
  - Query message sent over existing TCP connections
- Resolve queries by localized flooding
  - ask neighbors (~7), who ask their neighbors, and so on ...
  - when/if found, reply to sender
  - Time to live fields limit hops (typically 10)

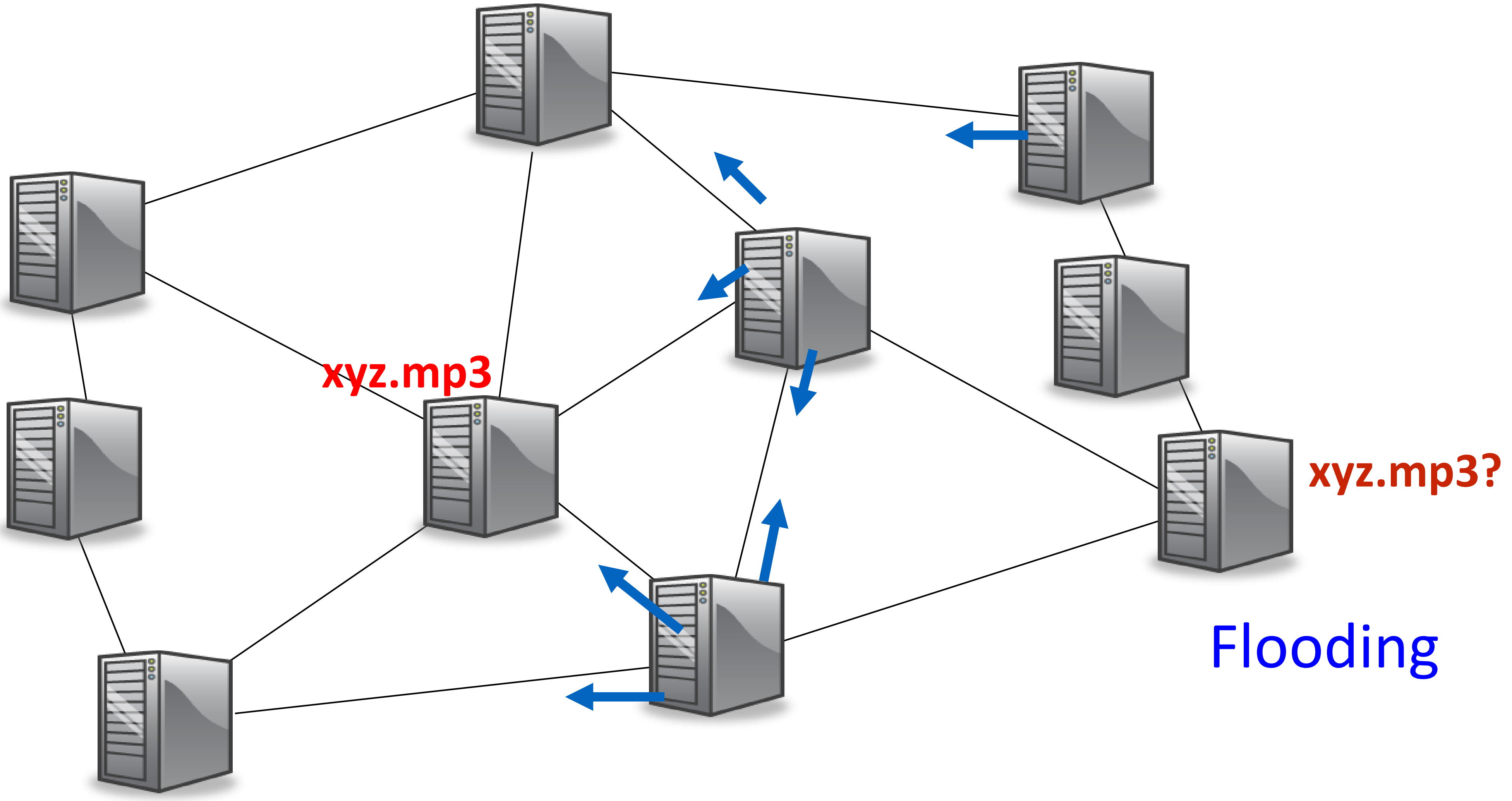
Gnutella:

Flooding on Overlays



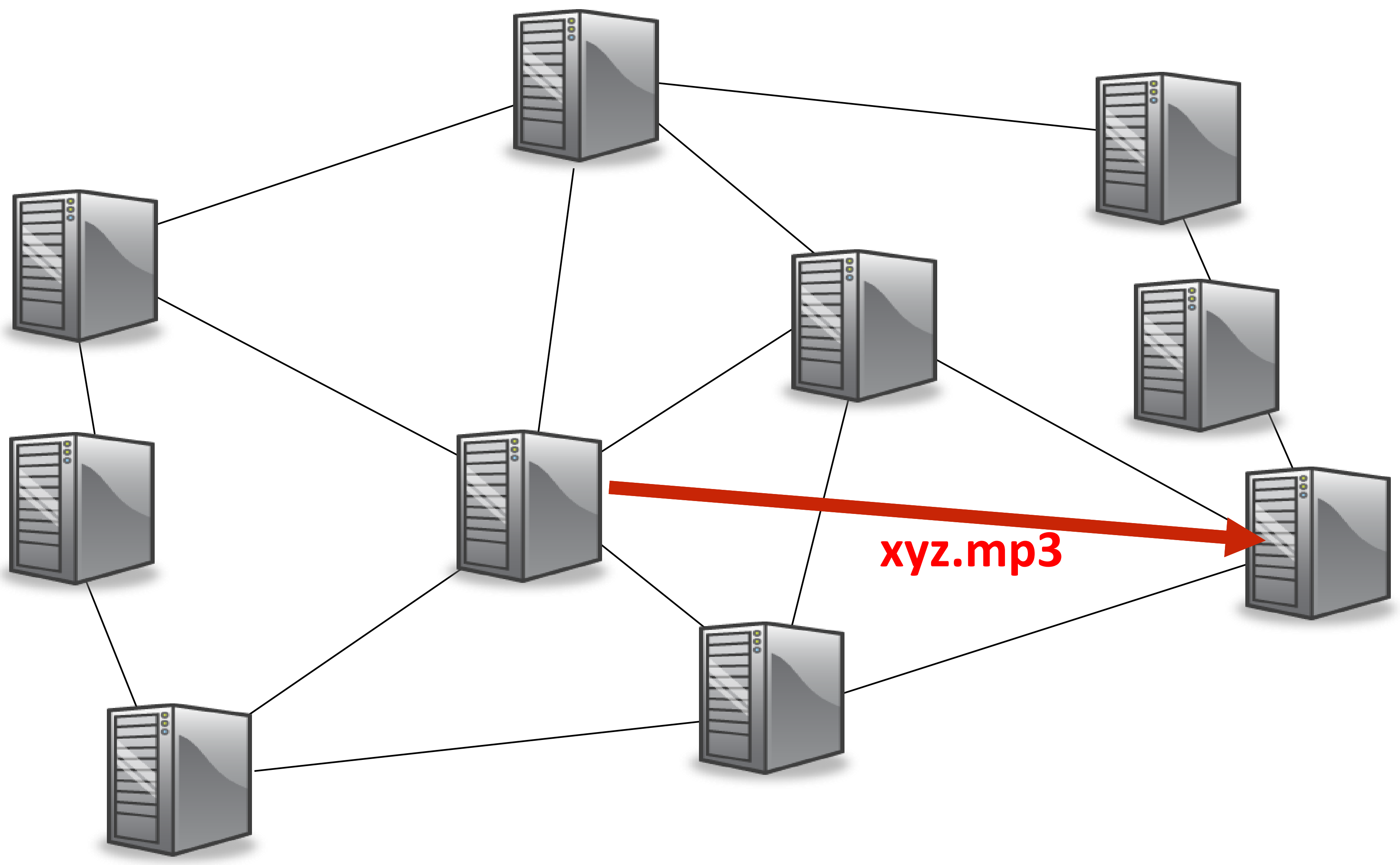
Gnutella:

Flooding on Overlays



Gnutella:

Flooding on Overlays



Gnutella: Pros and Cons

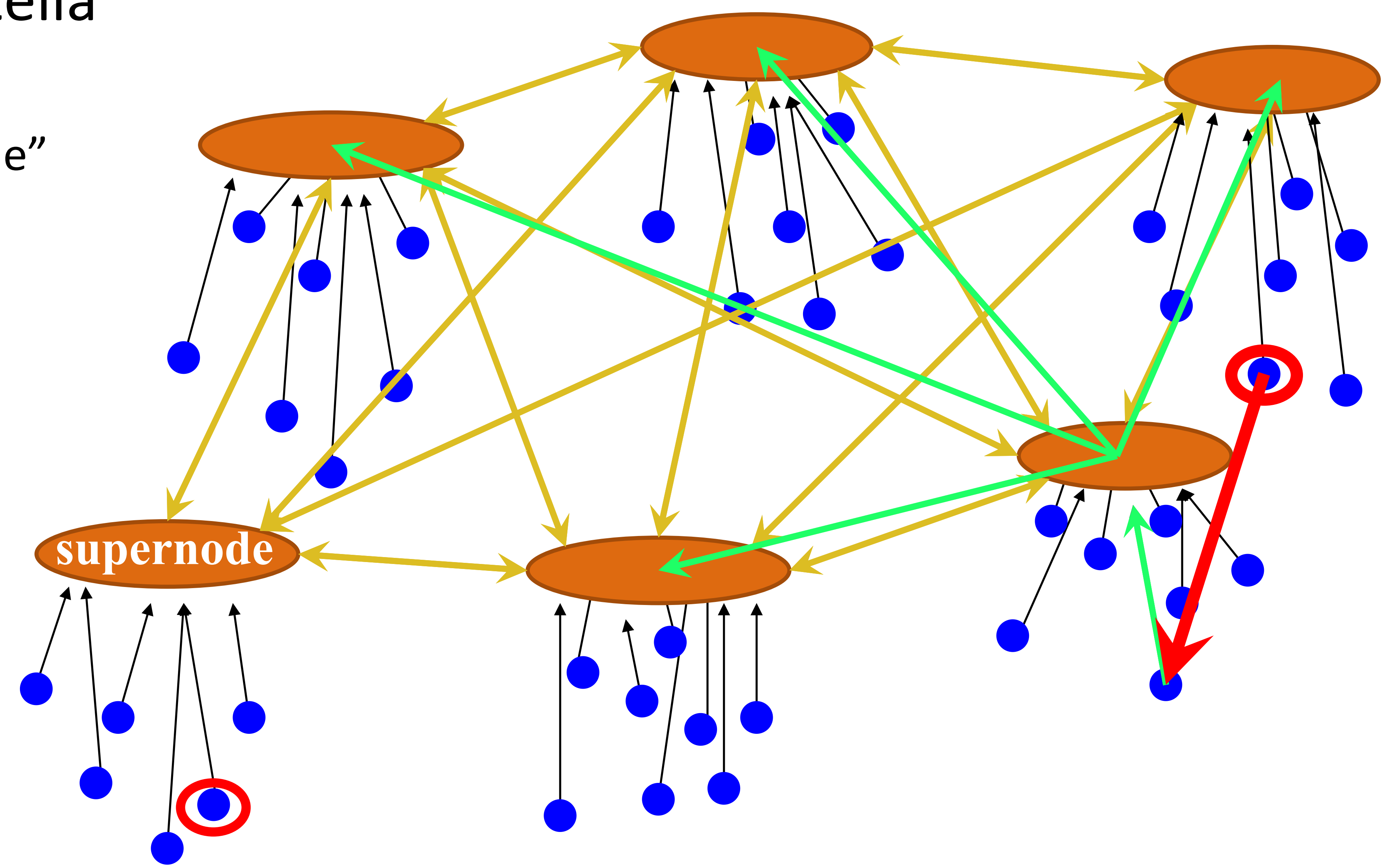
- Advantages
  - Fully decentralized
  - Search cost distributed
  - Processing per node permits powerful search semantics
- Disadvantages
  - Search scope may be quite large
  - Search time may be quite long
  - High overhead and nodes come and go often
- Users’ anecdotal comments: “It stinks”
  - Tough to find anything!
  - Downloads don’t complete!
- Response: **Hierarchy of nodes**

Hierarchical P2P Networks

Example: KazaA 2001

- Some nodes better and longer connected than others
  - Use them more heavily
- Cross between Napster and Gnutella

- **Join**: on startup, client contacts a “supernode”
  - may at some point become one itself
- **Publish**: send list of files to supernode
- **Search**: send query to supernode,
  - supernodes flood query amongst themselves.
- **Fetch**: get the file directly from peer(s)



- Improves scalability
- Limits flooding
- What if a supernode leaves the network?
  - Still no guarantees of performance

Motivation for Super-Nodes

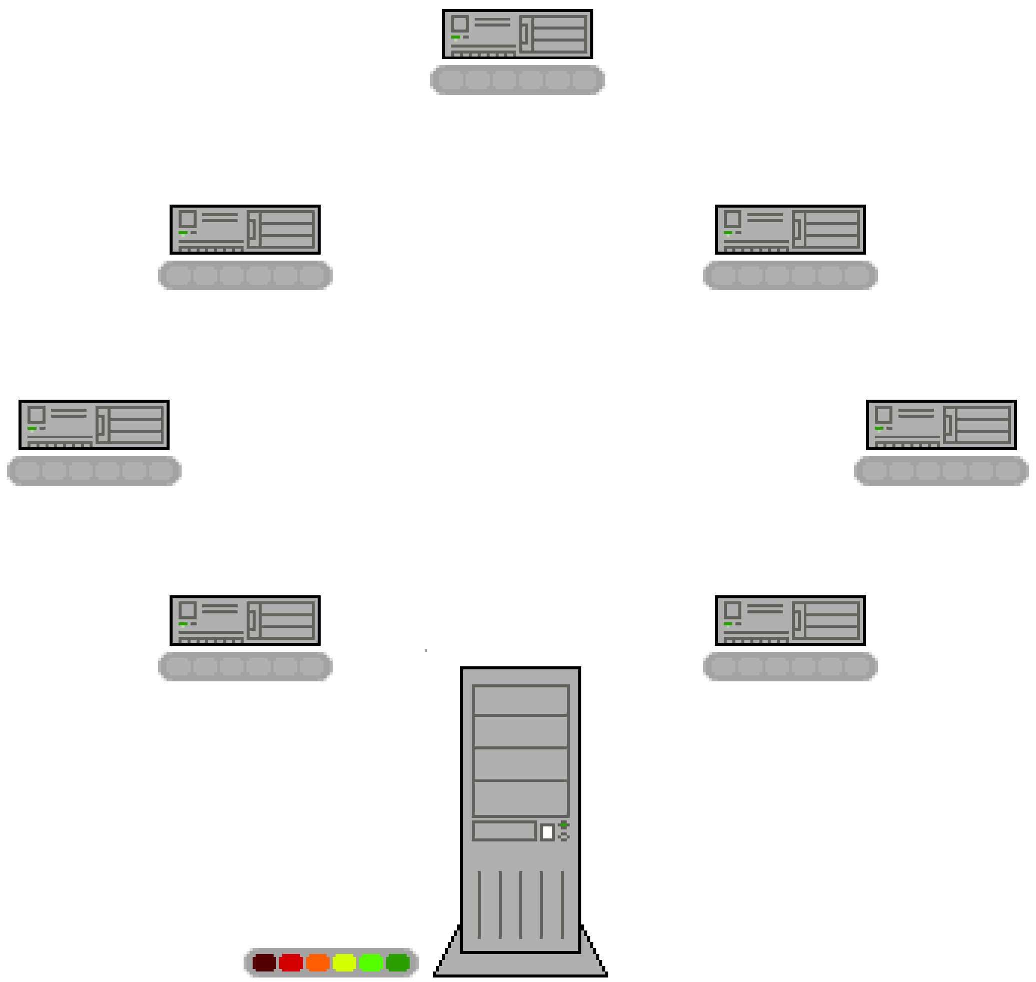
- Query consolidation
  - Many connected nodes may have only a few files
  - Propagating query to a sub-node may take more time than for the super-node to answer itself
- Stability
  - Super-node selection favors nodes with high up-time
  - How long you’ve been on is a good predictor of how long you’ll be around in the future

BitTorrent™

- 2002: B. Cohen debuted BitTorrent
- Key motivation: **popular content**
  - Popularity exhibits temporal locality (Flash Crowds)
  - E.g., release of a new movie or game, Slashdot effect, …
- Focused on **efficient fetching**, not searching
  - Distribute same file to many peers
- “Swarming”
  - Download from others downloading same object at same time

Overview

- Swarming:
  - Join: contact centralized “tracker” server, get a list of peers
  - Search: Out-of-band
    - E.g., use Google to find a tracker for the file you want.
  - Fetch: Download chunks of the file from your peers.
    - Upload chunks you have to them.



- Big difference compared to Napster:
  - Chunk based downloading
  - “Few large files” focus

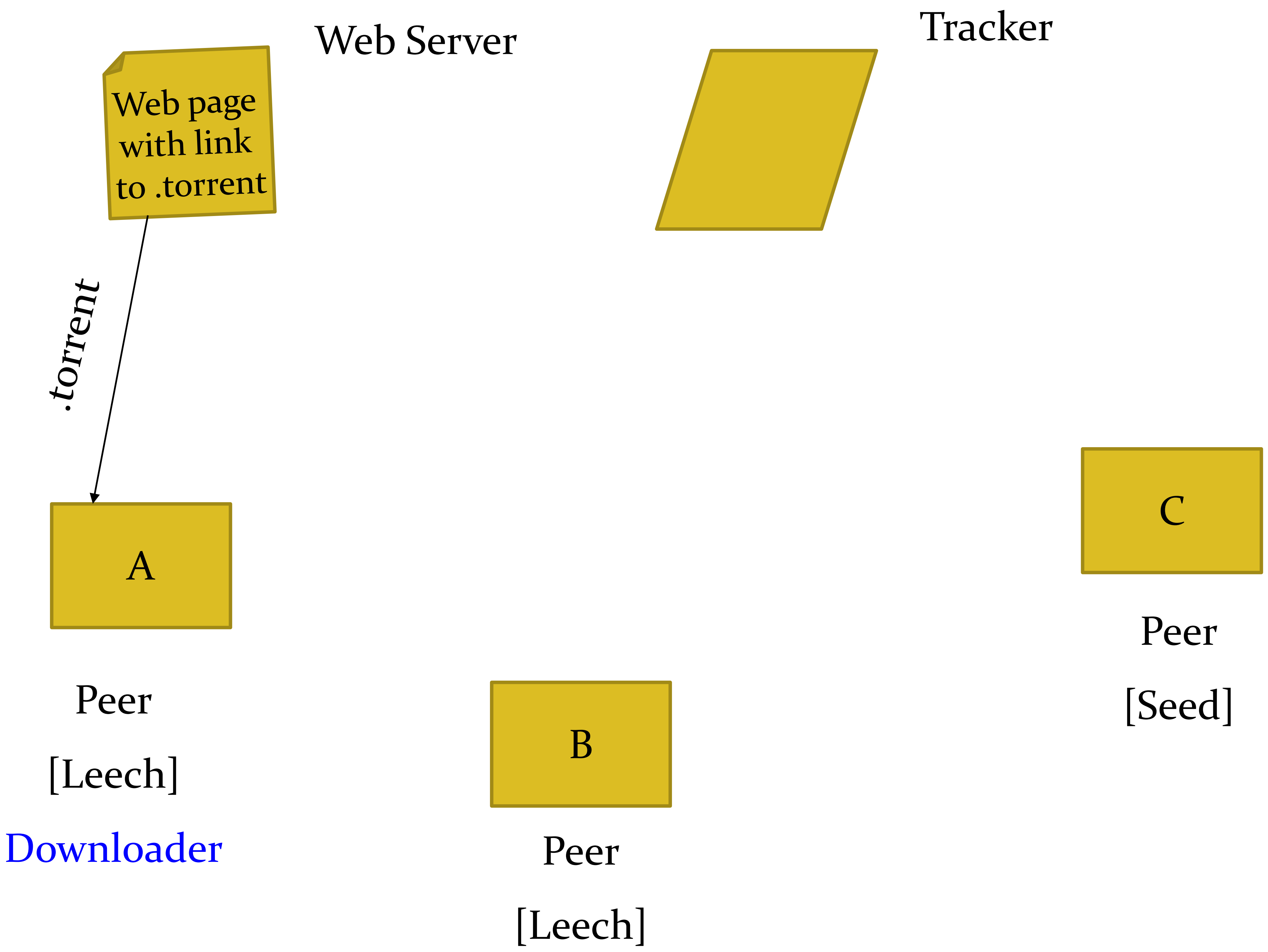
*BitTorrent - Wikipedia

Components

- Seed
  - Peer with entire file (Fragmented in small pieces ~16KB)
- Leacher
  - Peer with an incomplete copy of the file
- Torrent file
  - Passive component
  - Contains all meta-data related to a torrent
    - File name(s), sizes
    - Torrent hash: hash of the whole file, hash of each piece
    - URL of tracker(s)
- Tracker
  - Allows peers to find each other
  - Returns a list of random peers
  - Later, the creation of Distributed Hash Table (DHT) method led to trackerless torrents
    - We will leave DHT for an advanced distributed course!

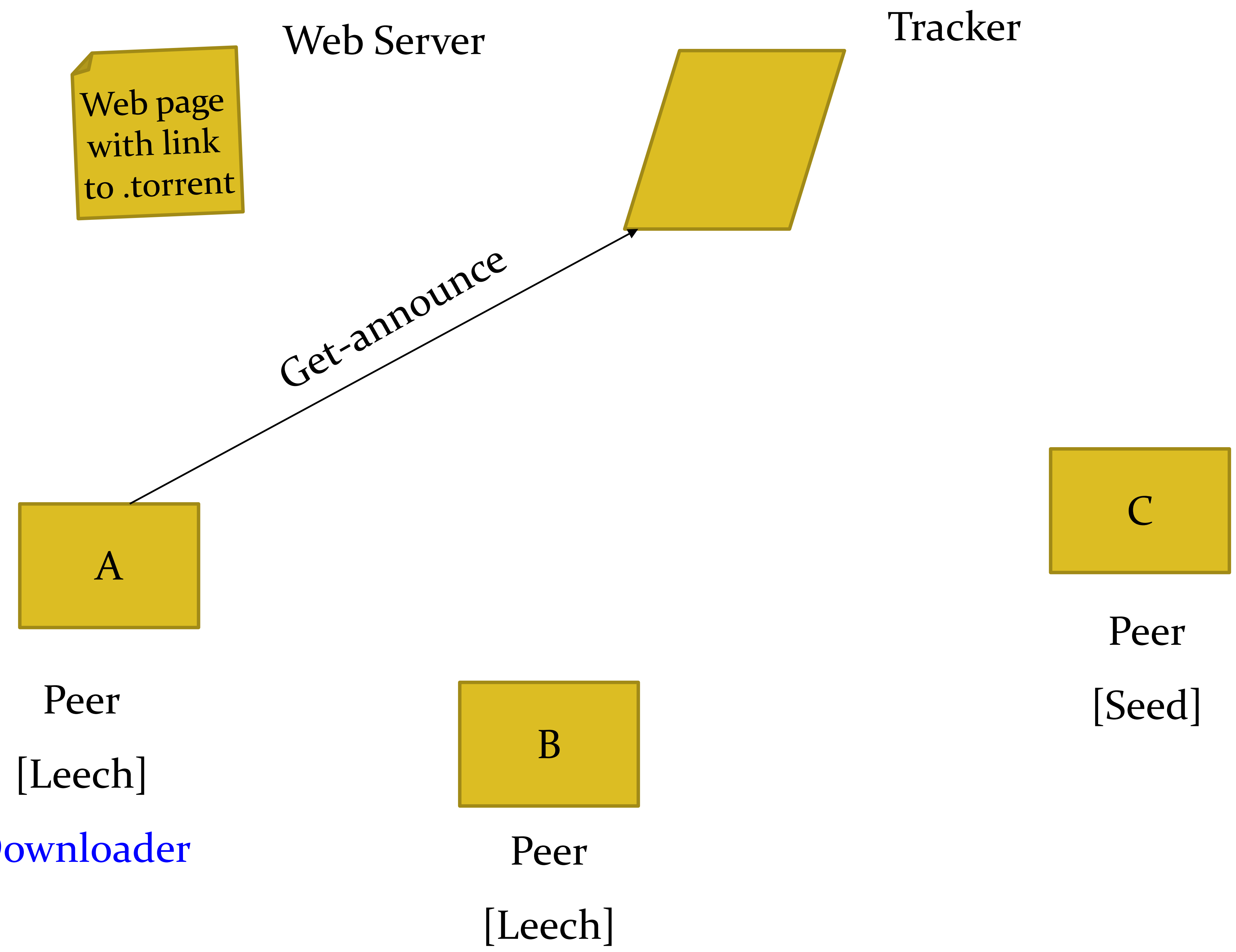
BitTorrent:

Overall Architecture



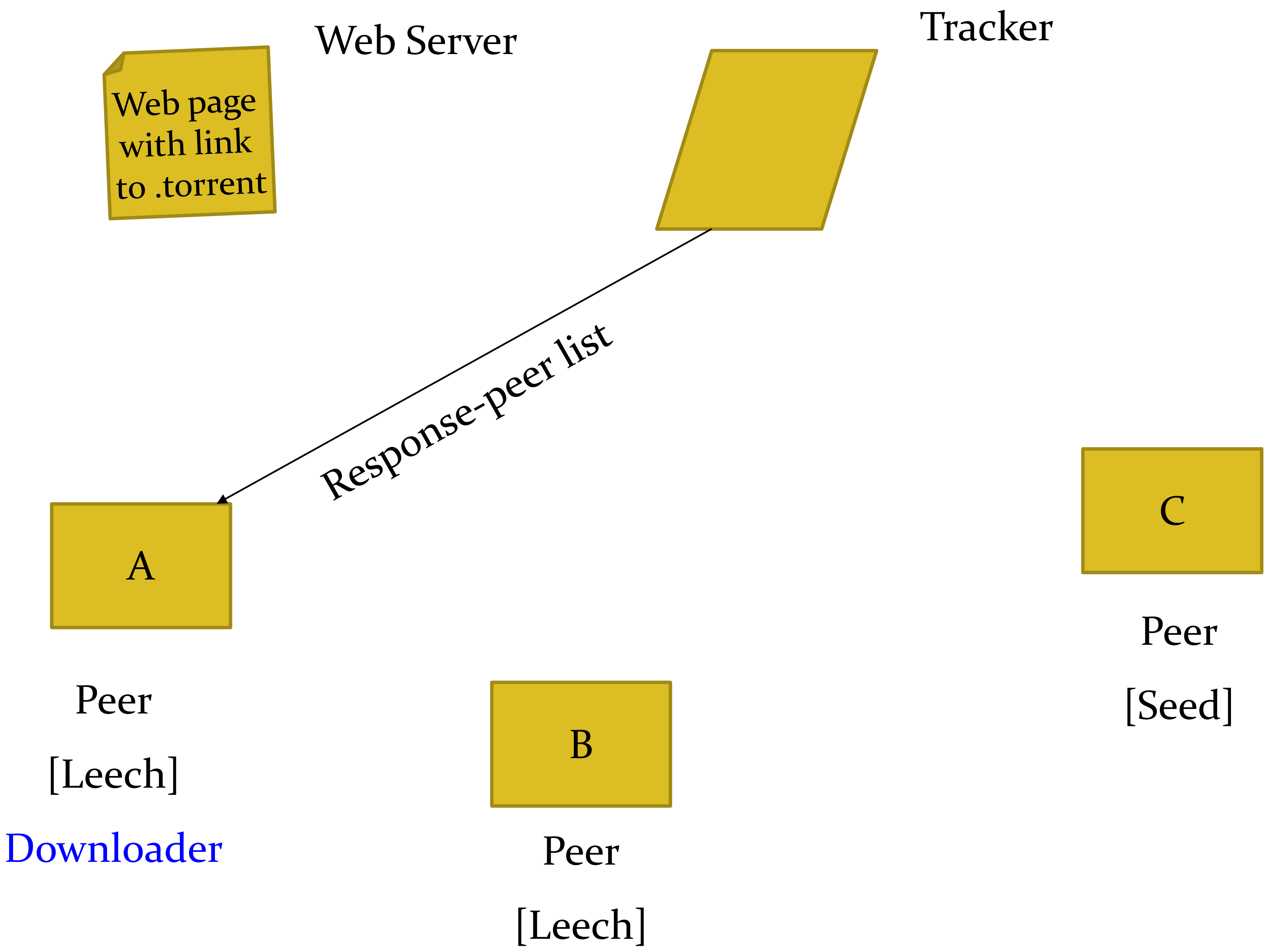
BitTorrent:

Overall Architecture

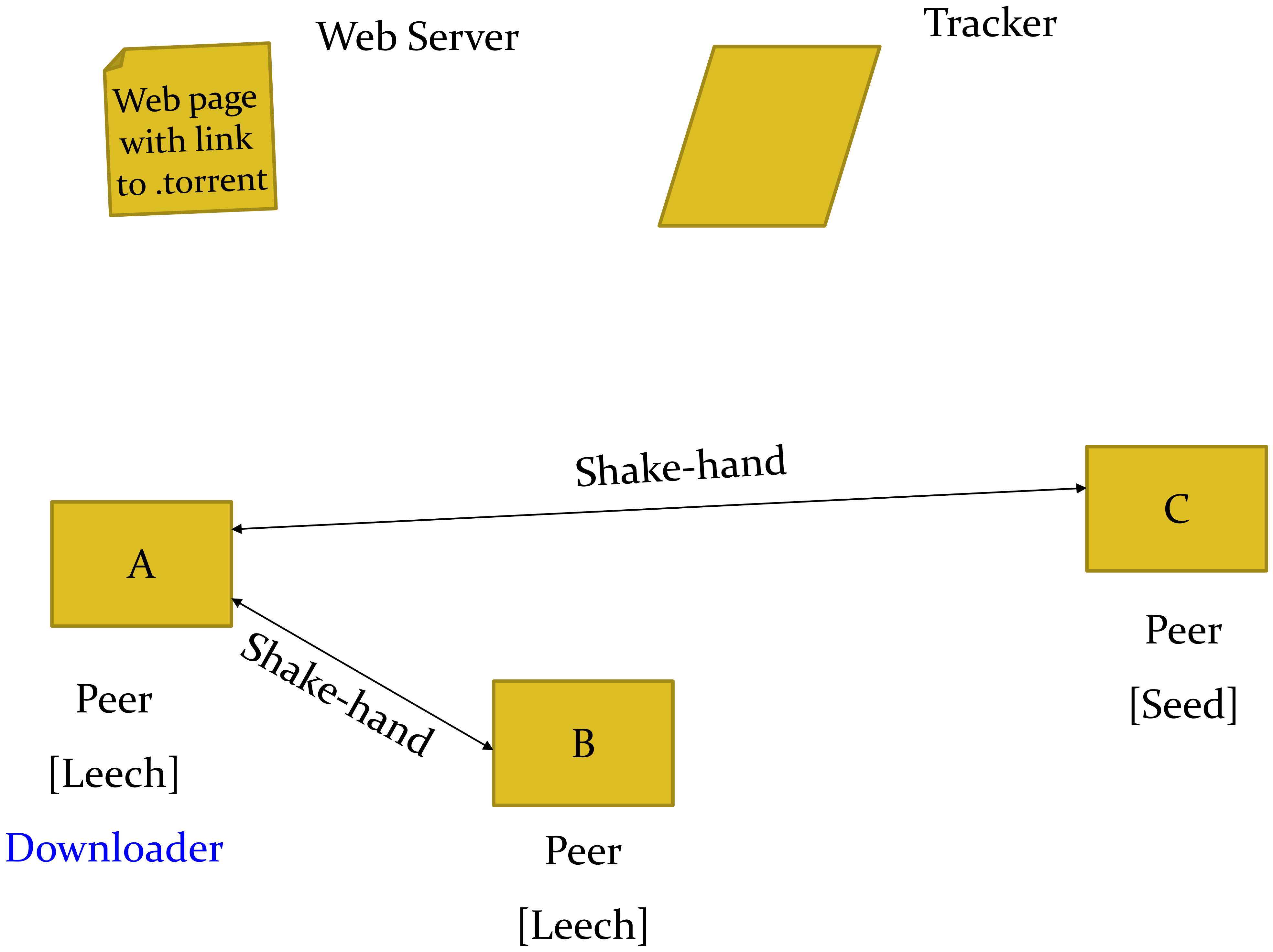


BitTorrent:

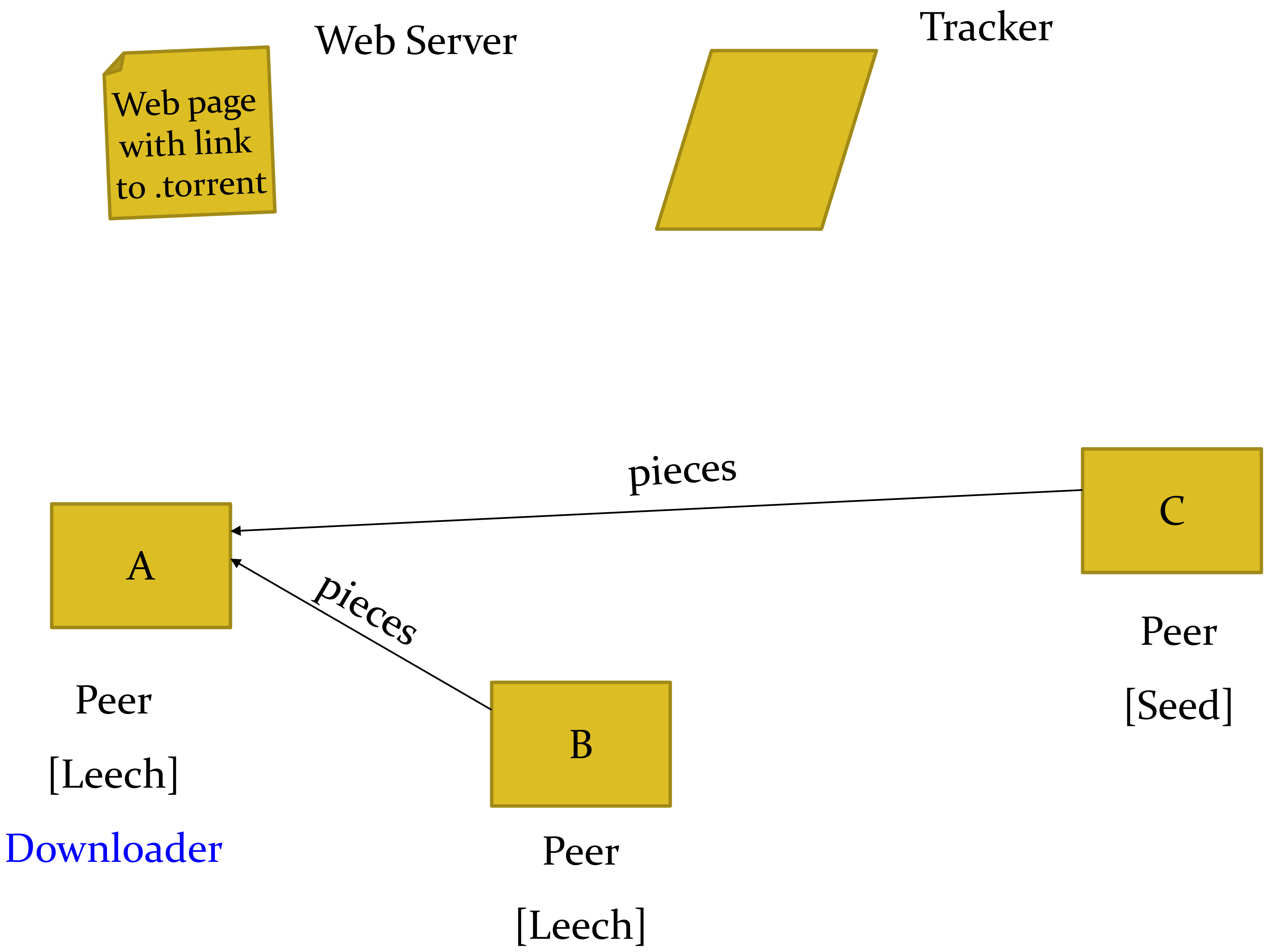
Overall Architecture



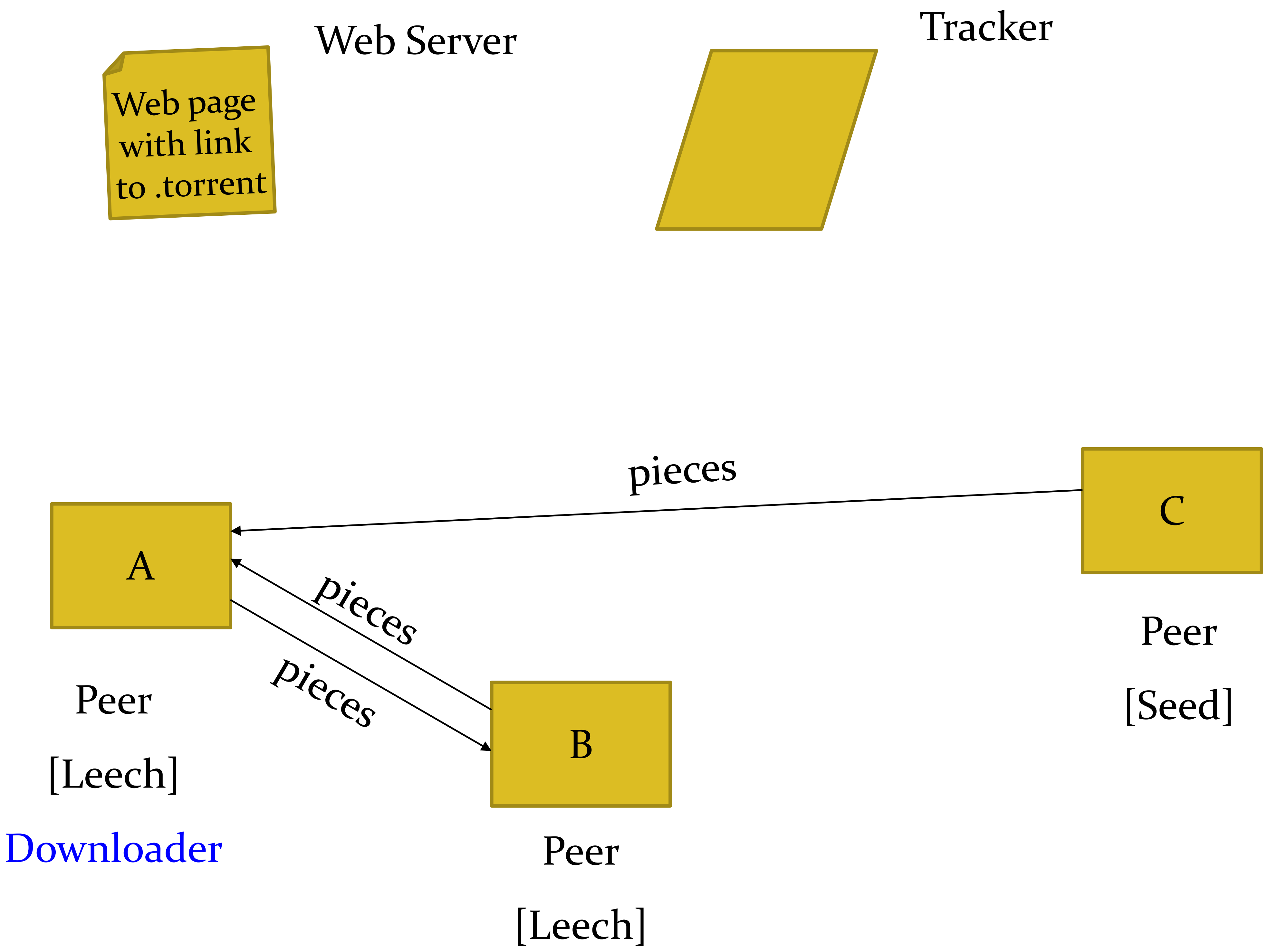
BitTorrent:
Overall Architecture



Overall Architecture



Overall Architecture



The Beauty of BitTorrent!

- More leechers = more replicas of pieces
- More replicas = faster downloads
  - Multiple, redundant sources for each piece
- Even while downloading, leechers take load off the seed(s)
  - Great for content distribution
  - Cost is shared among the swarm

Free-Riding Problem in P2P Networks

- Vast majority of users are free-riders
  - Most share no files and answer no queries
  - Others limit # of connections or upload speed

- A few “peers” essentially act as servers
  - A few individuals contributing to the public good
  - Making them hubs that basically act as a server

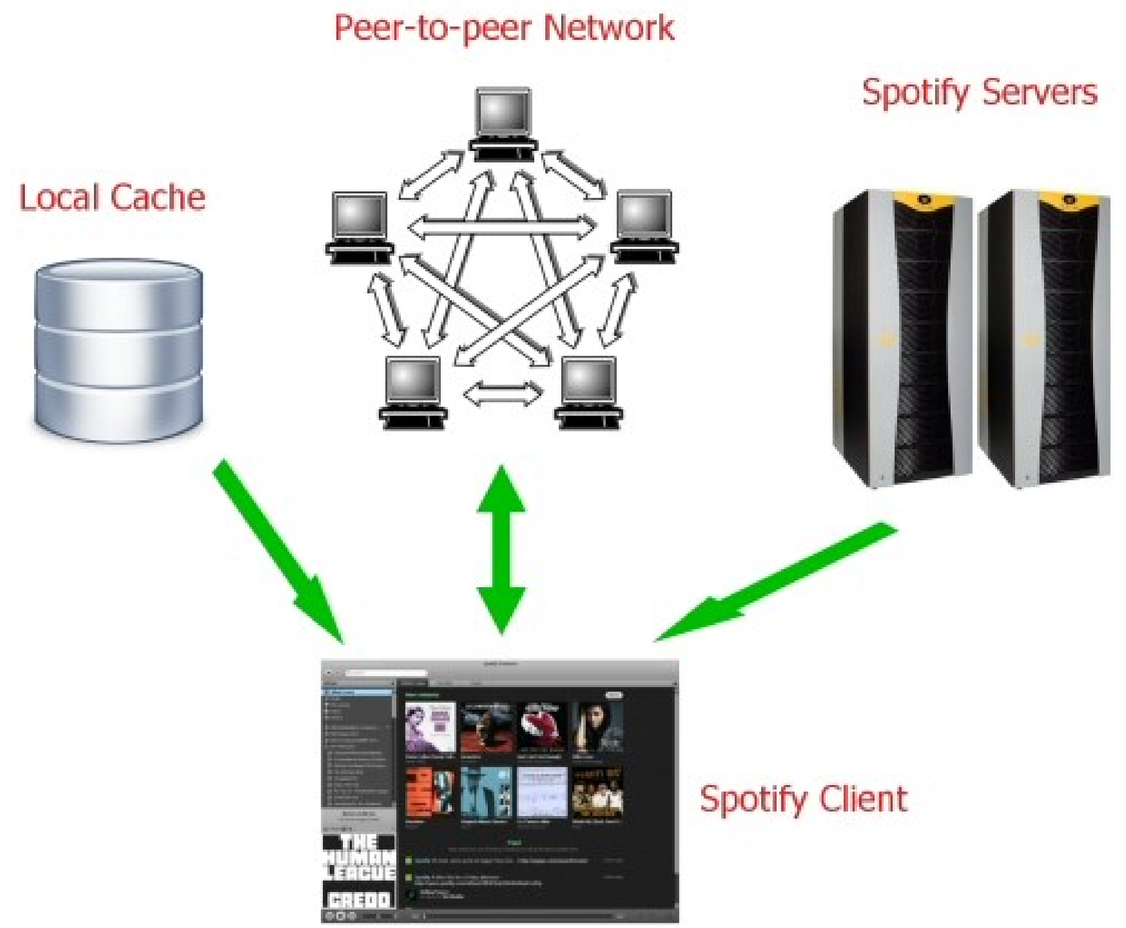
- BitTorrent prevents free riding: A “Tit-for-tat” sharing strategy
  - Yang is downloading from some other people
    - Yang will let the fastest N of those download from him
  - Be optimistic: occasionally let freeloaders download
    - Otherwise, no one would ever start!
    - Also allows you to discover better peers to download from when they reciprocate

Spotify 2006

- uTorrent (a minimalistic version of BT) lunched in Sep 2005
- Spotify bought uTorrent in 2006, before releasing its service (in 2008)

- Spotify uses BT as basic protocol
  - Uses server for first 15s
  - Tries to find peers and download from them
  - Only 8.8% of bytes come from servers (@ 2010)

- When 30s left
  - Starts searching for next track
  - Uses sever with 10s to go if no peers found



Spotify

Discussion

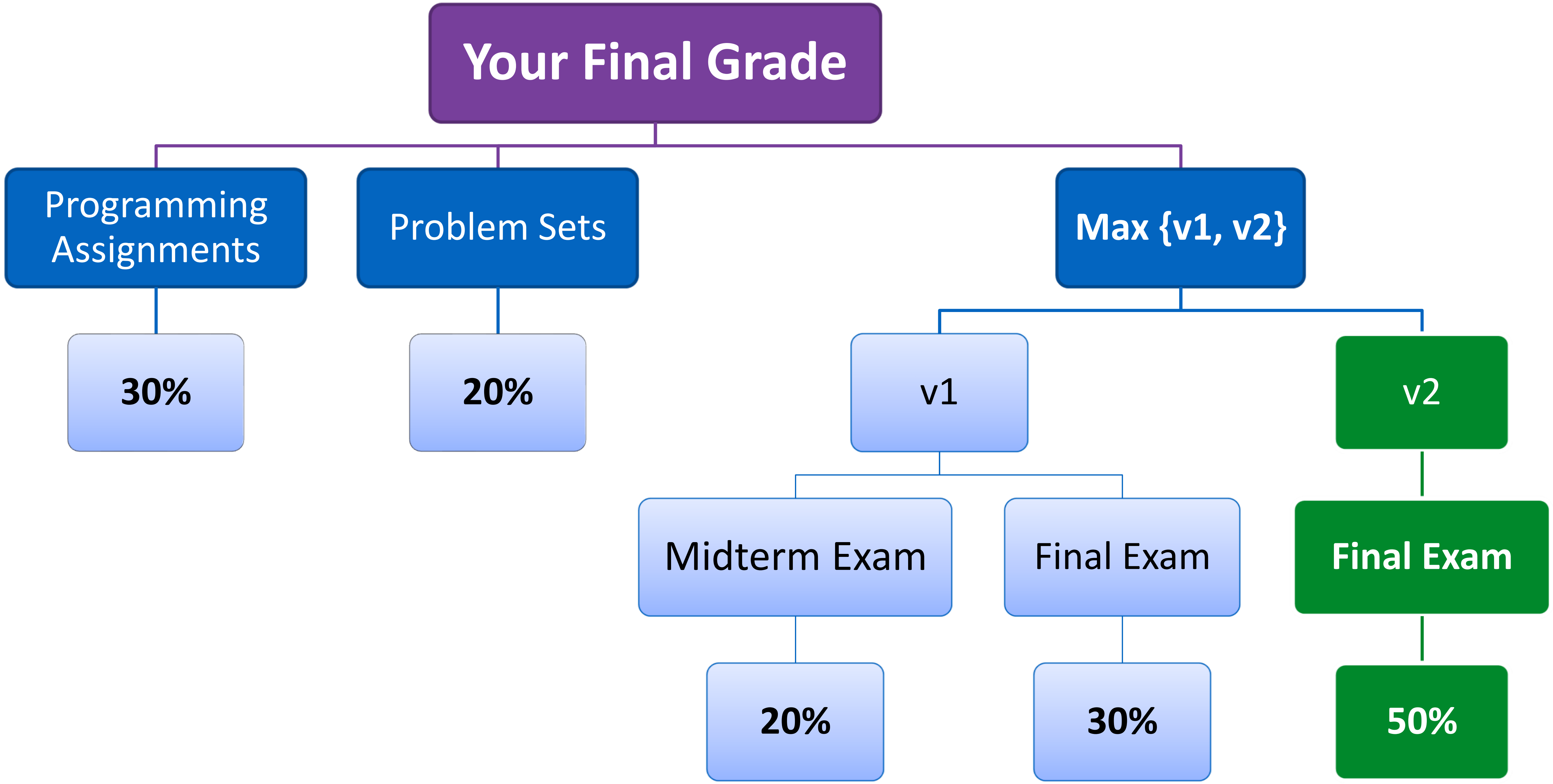
- Does Spotify use BitTorrent anymore? Why?
  - In 2014, they moved away from p2p
- What drove P2P initially?
  - Renting servers was expensive
  - Content was not otherwise available
- What about now?
  - AWS, Azure, Google Cloud, …
- In 2016 Spotify migrated to Google Cloud

- **Is P2P dead?**
  - Still lives on: Bitcoin/Ethereum/…

Wrap Up

- Overlay networks
  - Tunnels between host computers
  - Hosts implement new protocols and services
  - Effective way to build **logical** networks on top of the Internet
- P2P networks
  - Nodes are end hosts
  - Primarily for file sharing, and recently telephony
  - E.g., Centralized directory (Napster)
  - Query flooding (Gnutella)
  - Super-nodes (KaZaA)
  - Distributed downloading and anti-free-loading (BitTorrent)
- Great examples of how fast changes can happen in application-level protocols

Final Exam: the Logistics


Your Final Grade
Programming Assignments
30%
Problem Sets
20%
Max {v1, v2}
v1
Midterm Exam
20%
Final Exam
30%
v2
Final Exam
50%

Final Exam Logistics

- Time:
  - December 14^{th}
  - For exact location and time, check this:
    - https://www.artsci.utoronto.ca/current/faculty-registrar/exams-assessments/exam-assessment-schedule#exam-assessments-schedule-accordion-1
- Extra office hours on Monday Dec 12 (3-5pm)

- Closed book, closed notes, etc.
- Single two-sided "cheat sheet", handwritten
  - Not bigger than an A4 paper!

- **No** calculators, electronic devices, etc.
  - Test does not require any complicated calculations

General Guidelines (1)
Similar to the Midterm!

- Test only assumes material covered in our lectures
  - Text: only to clarify details and context for the above
    The test doesn't require you to do complicated calculations
  - Use this as a hint to whether you are on the right track!
- Everything covered
  - With more focus on lectures 6-11
- You do need to understand **how** things work
  - not for the sake of knowing gory details but to understand **pros/cons**, when a solution is applicable/useful/useless, etc.

General Guidelines (2)

Similar to the Midterm!

Be prepared to:

- Weigh design options outside of the context we studied them in
  - *e.g., if TCP connections didn't have three-way handshake, then …"*
- Contemplate new designs we haven't talked about
  - *e.g., I introduce a new congestion control; how does this affect …"*
- Don't let this daunt you. Reason from what you know about the pros/cons of solutions we did study
  - *e.g., circuit switching is inefficient when …*

General Guidelines (3)
Exam format

Q1- Multiple-choice questions (~26 questions)

Q2- Design questions

- Similar to the midterm:
  - A set of “here’s a scenario, tell me if the following is true/false”-style questions

Q3- Short answer questions

Q4+ more traditional long questions

Sample questions

- Some sample questions for final + Midterm and solutions:

  https://t.ly/mxJBo

Questions?