Overlay Networks

Reading: 9.4

COS 461: Computer Networks

Spring 2006 (MW 1:30-2:50 in Friend 109)

Jennifer Rexford

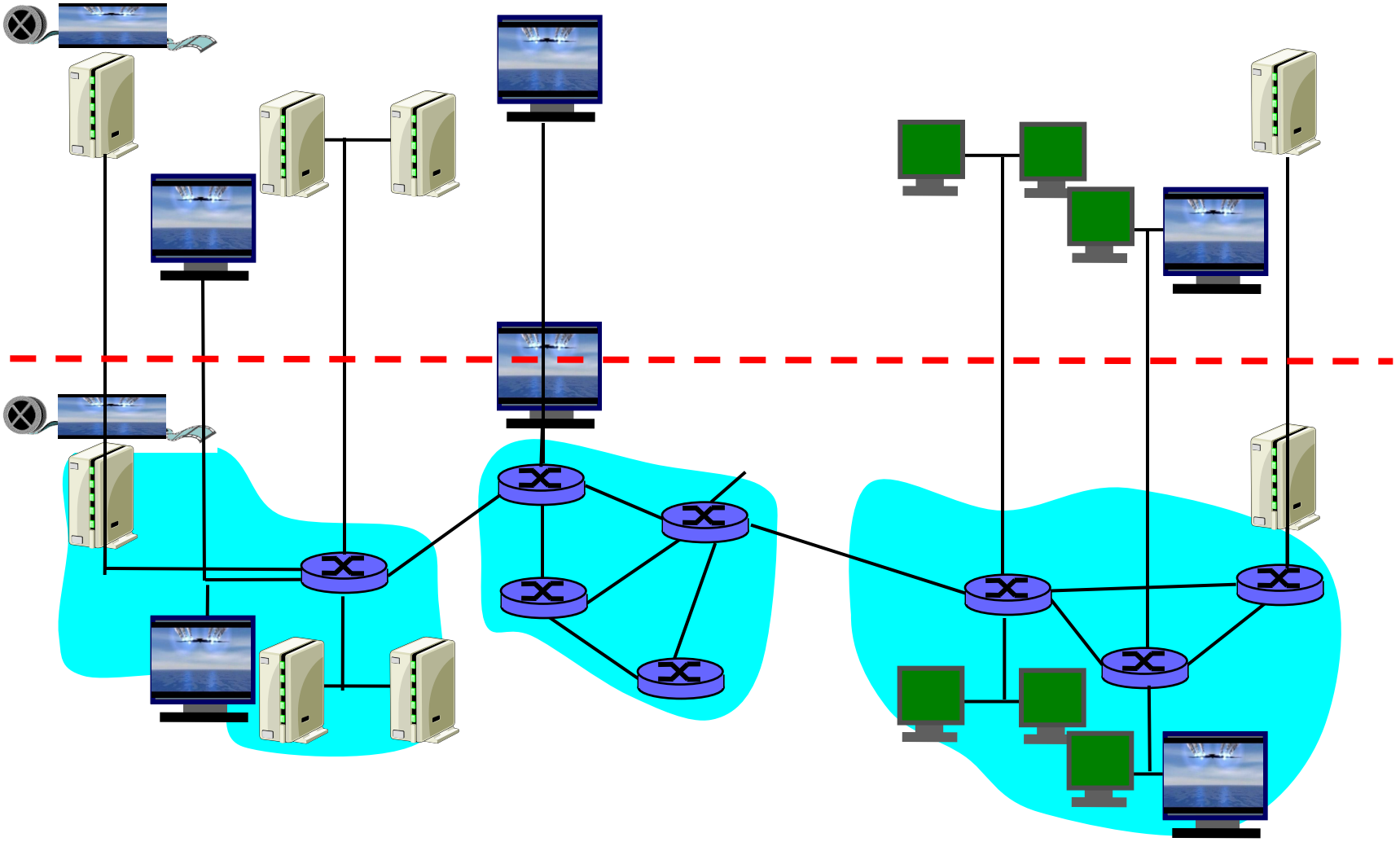
Teaching Assistant: Mike Wawrzoniak

http://www.cs.princeton.edu/courses/archive/spring06/cos461/

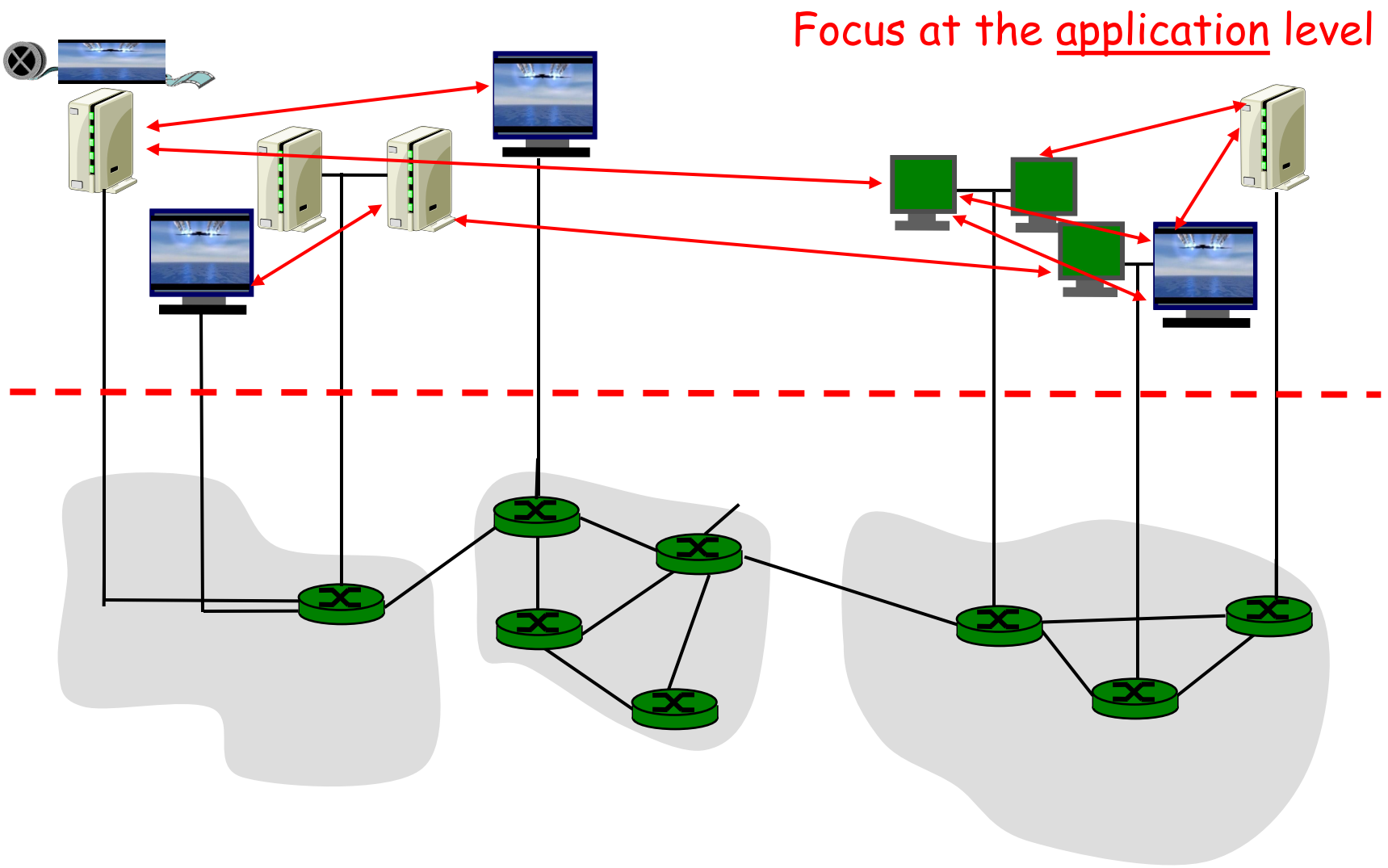
Goals of Today's Lecture

- Routing overlays
  - Experimental versions of IP (e.g., 6Bone)
  - Multicast (e.g., MBone and end-system multicast)
  - Robust routing (e.g., Resilient Overlay Networks)
- Types of peer-to-peer networks
  - Directory-based (e.g., original Napster design)
  - Unstructured (e.g., Gnutella, Kazaa, BitTorrent)
  - Structured (e.g., distributed hash tables)
- Challenges in peer-to-peer
  - Legal issues, free riding, fast response to queries, peers coming and going over time, reliability, security, …

Overlay Networks



Overlay Networks



Overlay Networks

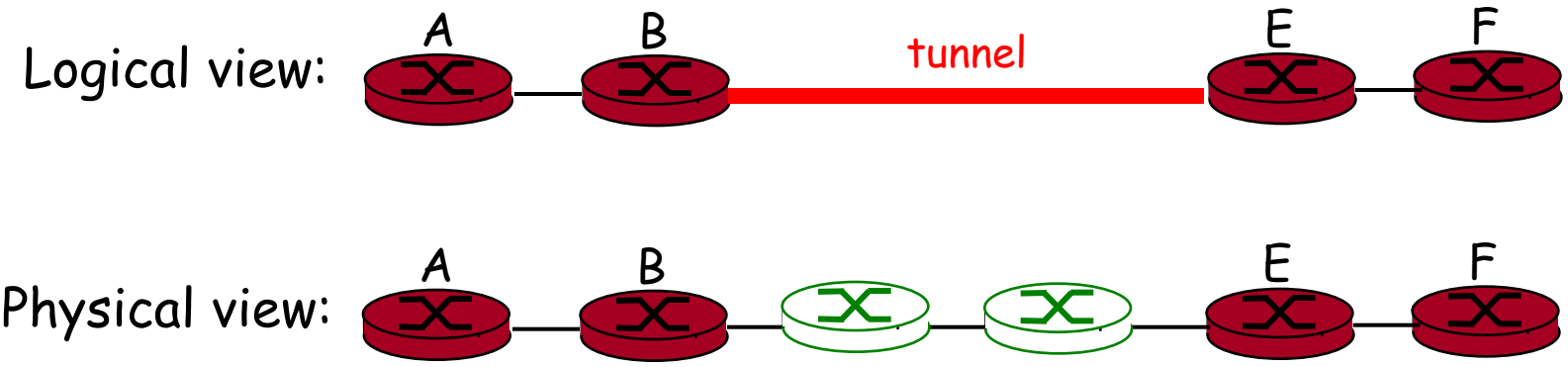
- A logical network built on top of a physical network
  - Overlay links are tunnels through the underlying network
- Many logical networks may coexist at once
  - Over the same underlying network
  - And providing its own particular service
- Nodes are often end hosts
  - Acting as intermediate nodes that forward traffic
  - Providing a service, such as access to files
- Who controls the nodes providing service?
  - The party providing the service (e.g., Akamai)
  - Distributed collection of end users (e.g., peer-to-peer)

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
  - Multicast
  - Mobility
  - Security
- Revisiting where a function belongs
  - End-system multicast: multicast distribution by end hosts
- Customized path selection
  - Resilient Overlay Networks: robust packet delivery

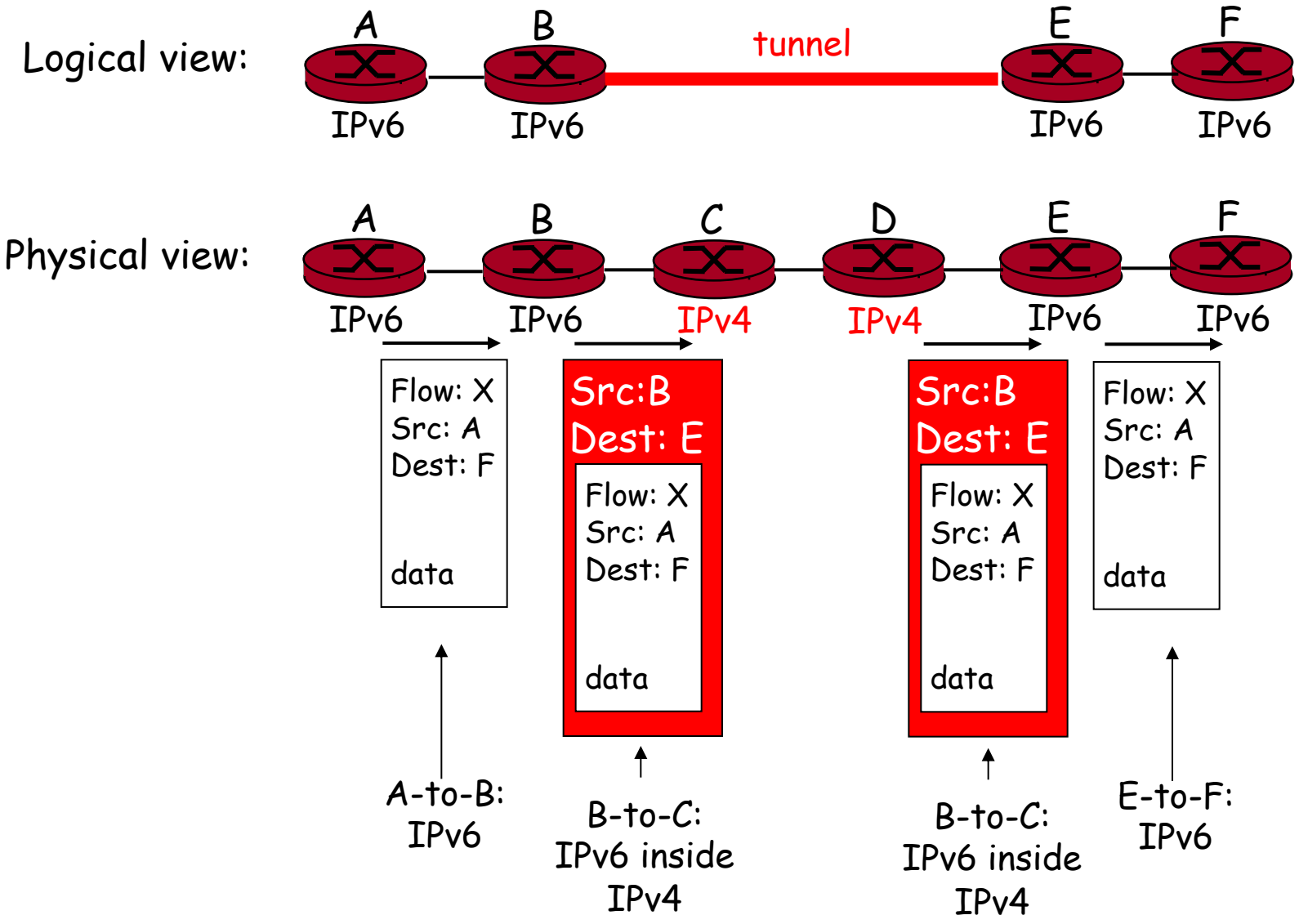
IP Tunneling

- IP tunnel is a virtual point-to-point link
  - Illusion of a direct link between two separated nodes

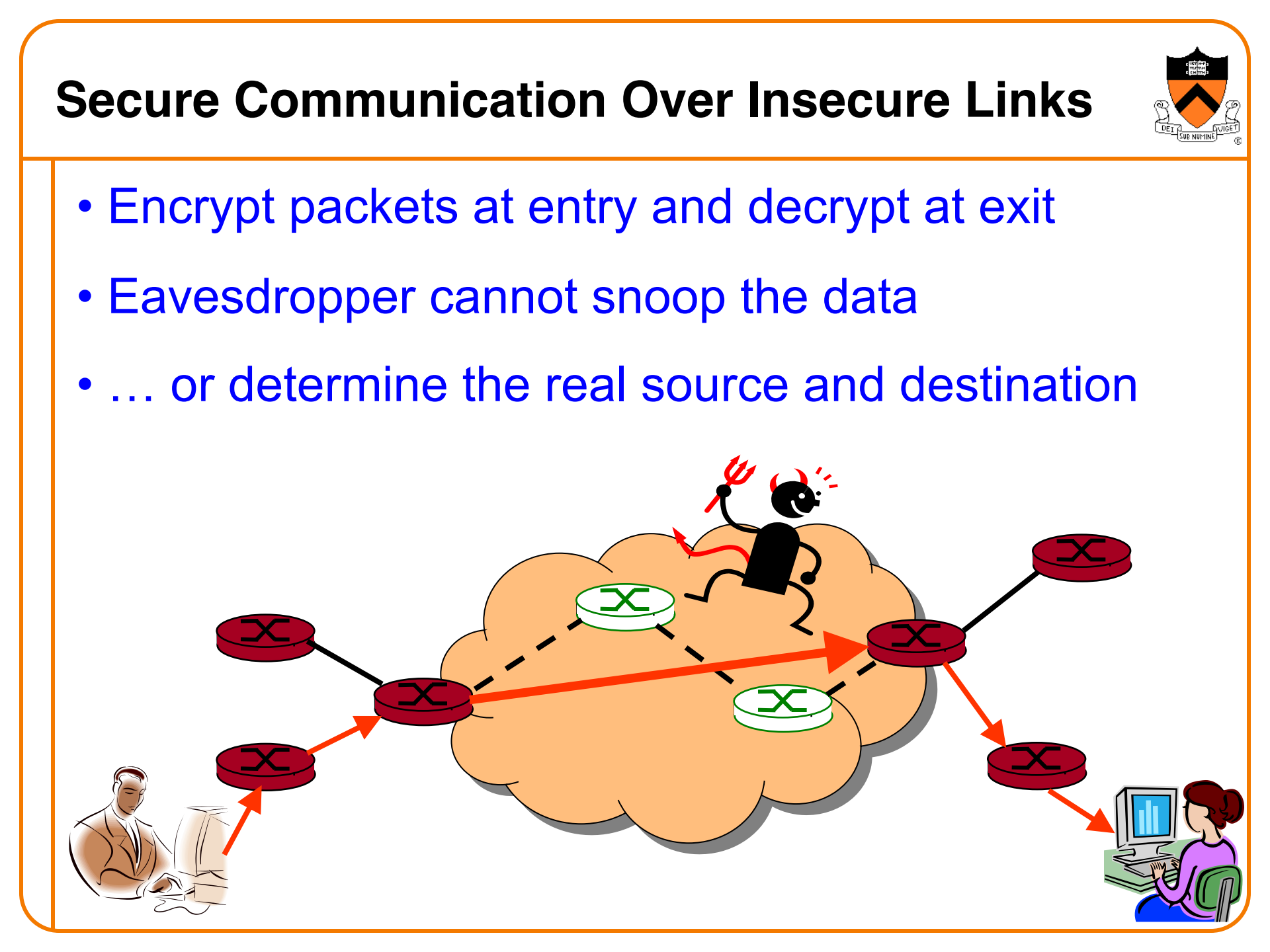


- Encapsulation of the packet inside an IP datagram
  - Node B sends a packet to node E
  - … containing another packet as the payload

6Bone: Deploying IPv6 over IP4



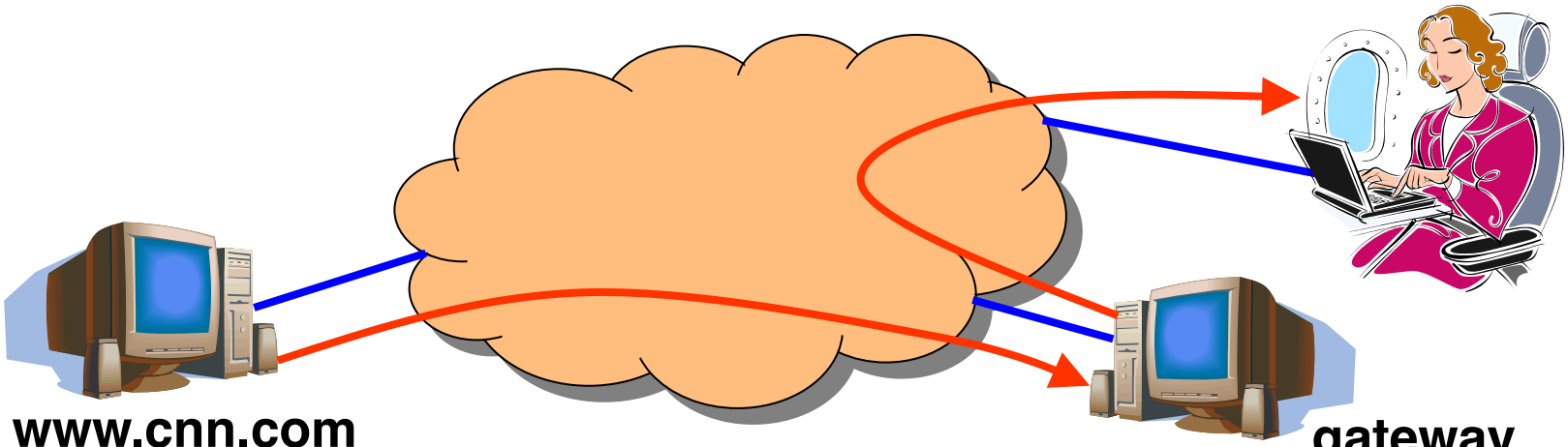
Secure Communication Over Insecure Links



- Encrypt packets at entry and decrypt at exit
- Eavesdropper cannot snoop the data
- … or determine the real source and destination

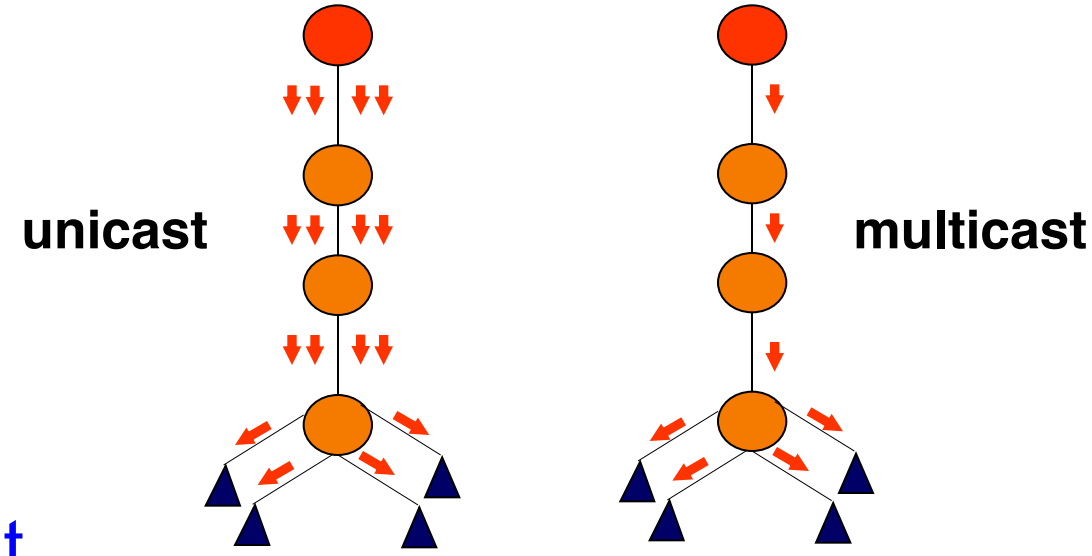
Communicating With Mobile Users

- A mobile user changes locations frequently
  - So, the IP address of the machine changes often
- The user wants applications to continue running
  - So, the change in IP address needs to be hidden
- Solution: fixed gateway forwards packets
  - Gateway has a fixed IP address
  - … and keeps track of the mobile's address changes



MBone: IP Multicast

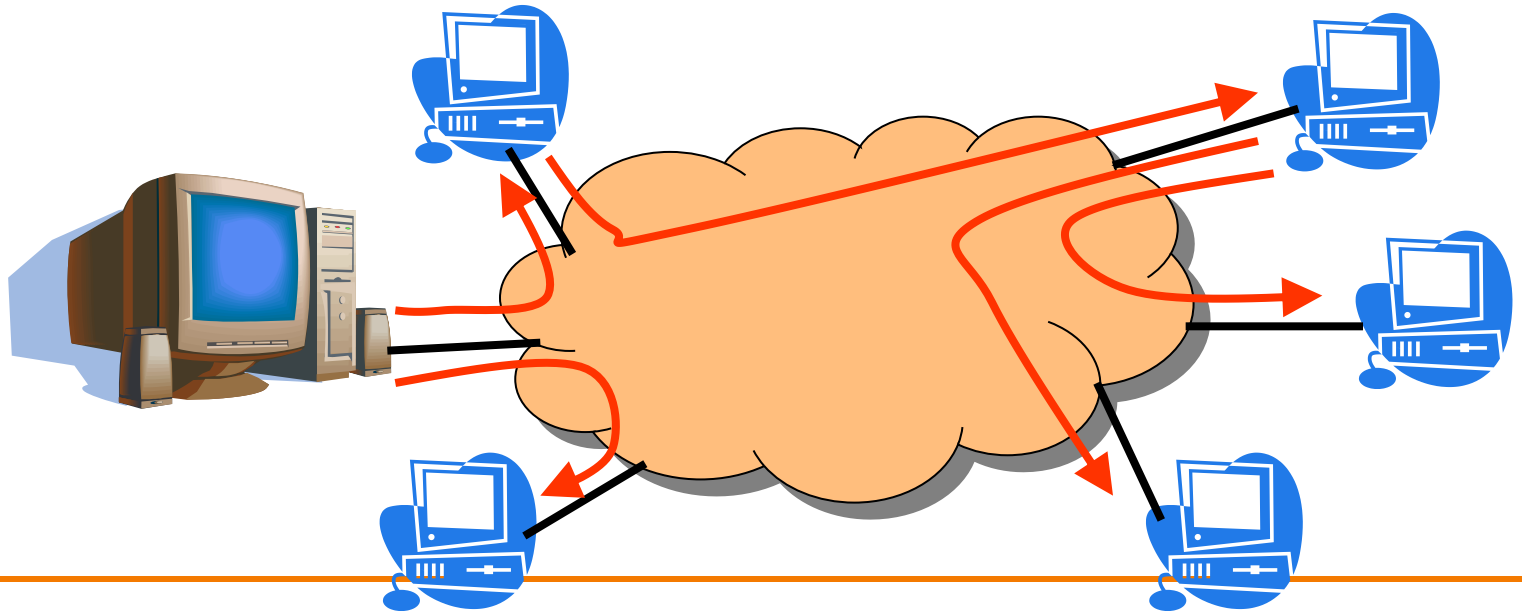
- Multicast
  - Delivering the same data to many receivers
  - Avoiding sending the same data many times



- IP multicast
  - Special addressing, forwarding, and routing schemes
  - Not widely deployed, so MBone tunneled between nodes

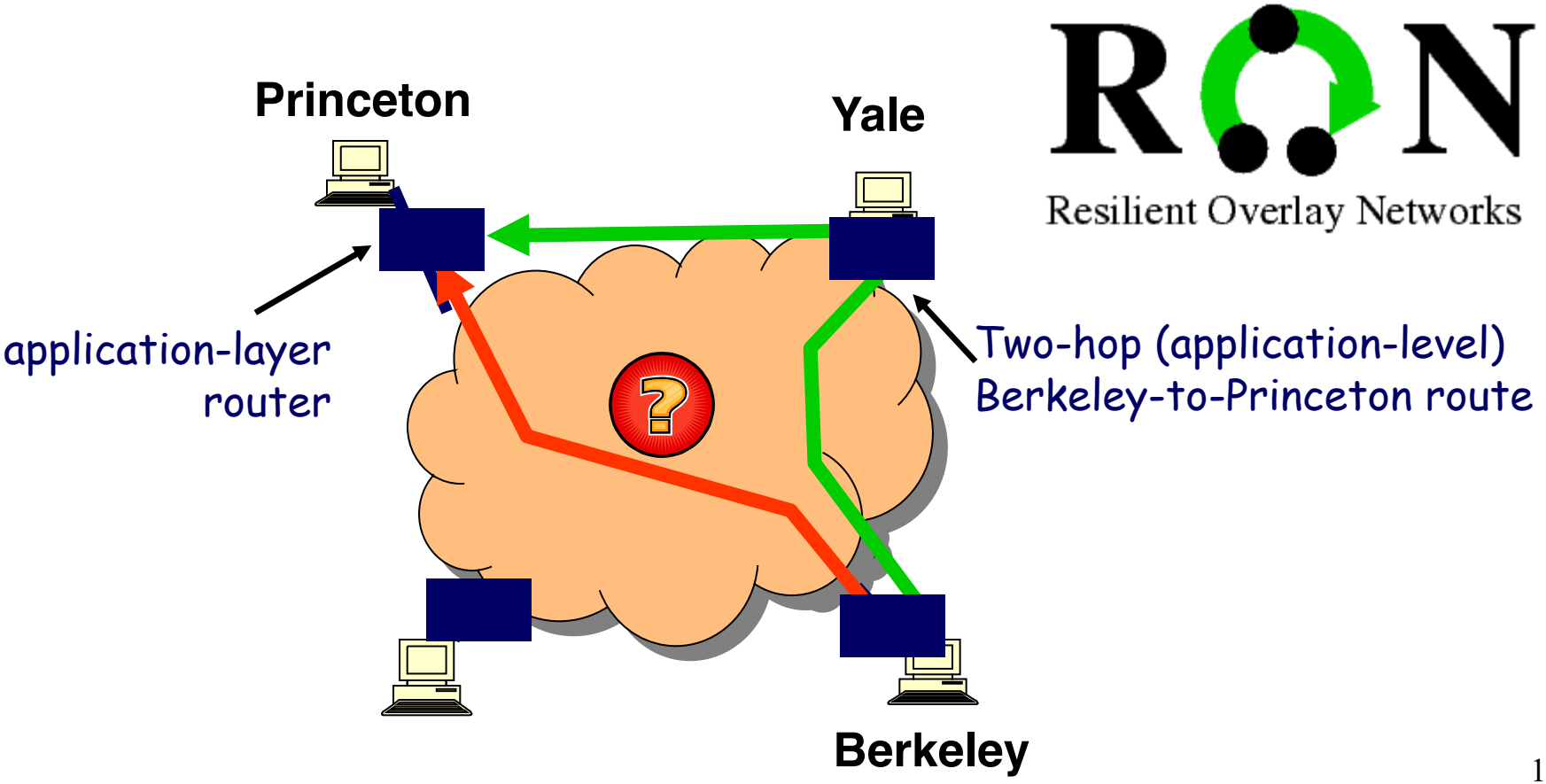
End-System Multicast

- IP multicast still is not widely deployed
  - Technical and business challenges
  - Should multicast be a *network*-layer service?
- Multicast tree of end hosts
  - Allow end hosts to form their own multicast tree
  - Hosts receiving the data help forward to others



RON: Resilient Overlay Networks

Premise: by building application overlay network, can increase performance and reliability of routing



RON Can Outperform IP Routing

- IP routing does not adapt to congestion
  - But RON can reroute when the direct path is congested
- IP routing is sometimes slow to converge
  - But RON can quickly direct traffic through intermediary
- IP routing depends on AS routing policies
  - But RON may pick paths that circumvent policies
- Then again, RON has its own overheads
  - Packets go in and out at intermediate nodes
    - Performance degradation, load on hosts, and financial cost
  - Probing overhead to monitor the virtual links
    - Limits RON to deployments with a small number of nodes

Peer-to-Peer Networks: Napster

Shawn Fanning, Northeastern freshman

- Napster history: the rise
  - January 1999: Napster version 1.0
  - May 1999: company founded
  - September 1999: first lawsuits
  - 2000: 80 million users
- Napster history: the fall
  - Mid 2001: out of business due to lawsuits
  - Mid 2001: dozens of P2P alternatives that were harder to touch, though these have gradually been constrained
  - 2003: growth of pay services like iTunes
- Napster history: the resurrection
  - 2003: Napster reconstituted as a pay service
  - 2006: still lots of file sharing going on

Napster Technology: Directory Service

- User installing the software
  - Download the client program
  - Register name, password, local directory, etc.
- Client contacts Napster (via TCP)
  - Provides a list of music files it will share
  - … and Napster's central server updates the directory
- Client searches on a title or performer
  - Napster identifies online clients with the file
  - … and provides IP addresses
- Client requests the file from the chosen supplier
  - Supplier transmits the file to the client
  - Both client and supplier report status to Napster

Napster Technology: Properties

- Server's directory continually updated
  - Always know what music is currently available
  - Point of vulnerability for legal action
- Peer-to-peer file transfer
  - No load on the server
  - Plausible deniability for legal action (but not enough)
- Proprietary protocol
  - Login, search, upload, download, and status operations
  - No security: cleartext passwords and other vulnerability
- Bandwidth issues
  - Suppliers ranked by apparent bandwidth & response time

Napster: Limitations of Central Directory

- Single point of failure
- Performance bottleneck
- Copyright infringement

File transfer is decentralized, but locating content is highly centralized

- So, later P2P systems were more distributed

Peer-to-Peer Networks: Gnutella

- Gnutella history
  - 2000: J. Frankel & T. Pepper released Gnutella
  - Soon after: many other clients (e.g., Morpheus, Limewire, Bearshare)
  - 2001: protocol enhancements, e.g., “ultrapeers”
- Query flooding
  - Join: contact a few nodes to become neighbors
  - Publish: no need!
  - Search: ask neighbors, who ask their neighbors
  - Fetch: get file directly from another node

Gnutella: Query Flooding

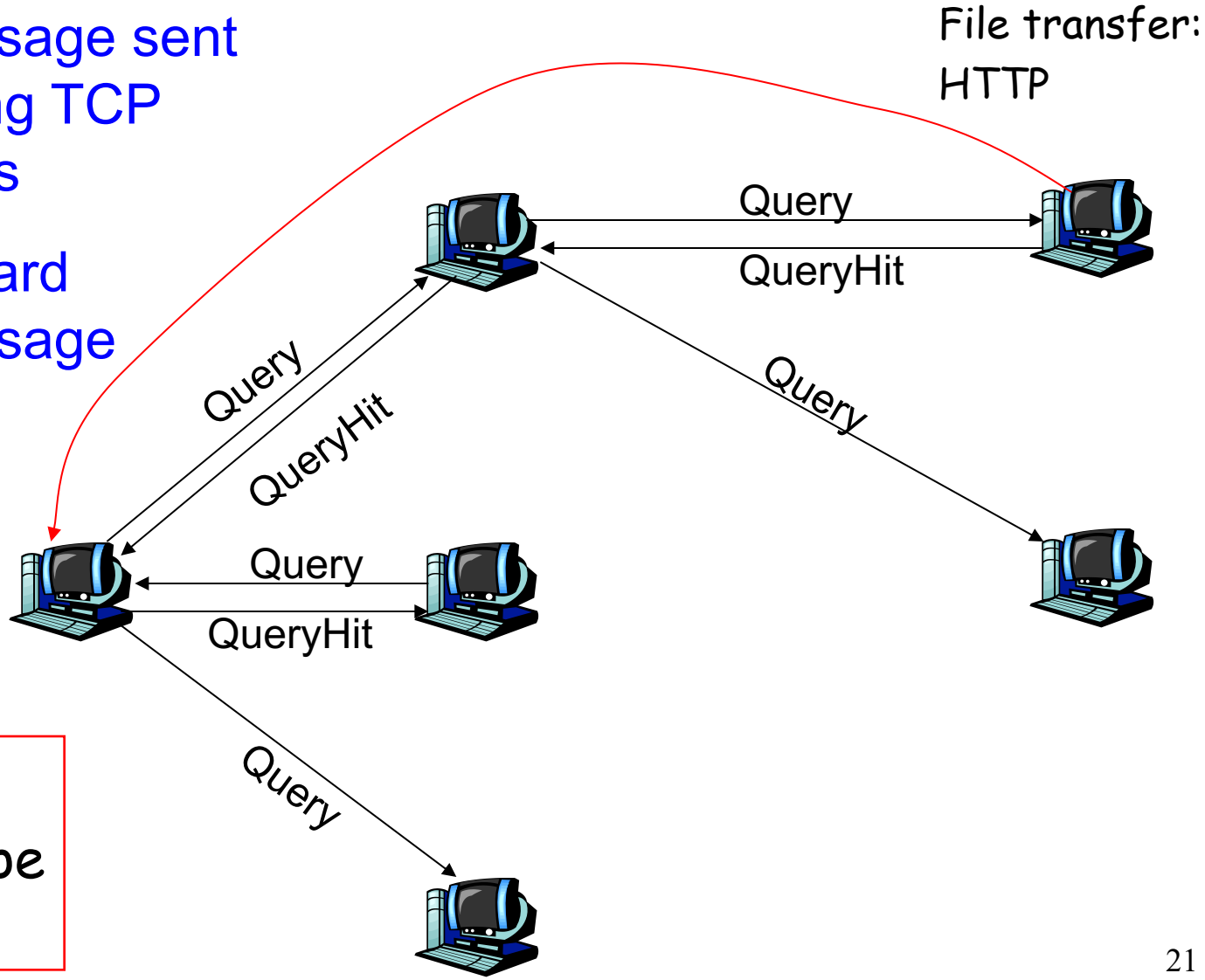
- Fully distributed
  - No central server
- Public domain protocol
- Many Gnutella clients implementing protocol

Overlay network: graph

- Edge between peer X and Y if there's a TCP connection
- All active peers and edges is overlay net
- Given peer will typically be connected with < 10 overlay neighbors

Gnutella: Protocol

- Query message sent over existing TCP connections
- Peers forward Query message
- QueryHit sent over reverse path



Scalability:
limited scope flooding

Gnutella: Peer Joining

- Joining peer X must find some other peer in Gnutella network: use list of candidate peers
- X sequentially attempts to make TCP with peers on list until connection setup with Y
- X sends Ping message to Y; Y forwards Ping message.
- All peers receiving Ping message respond with Pong message
- X receives many Pong messages. It can then setup additional TCP connections

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

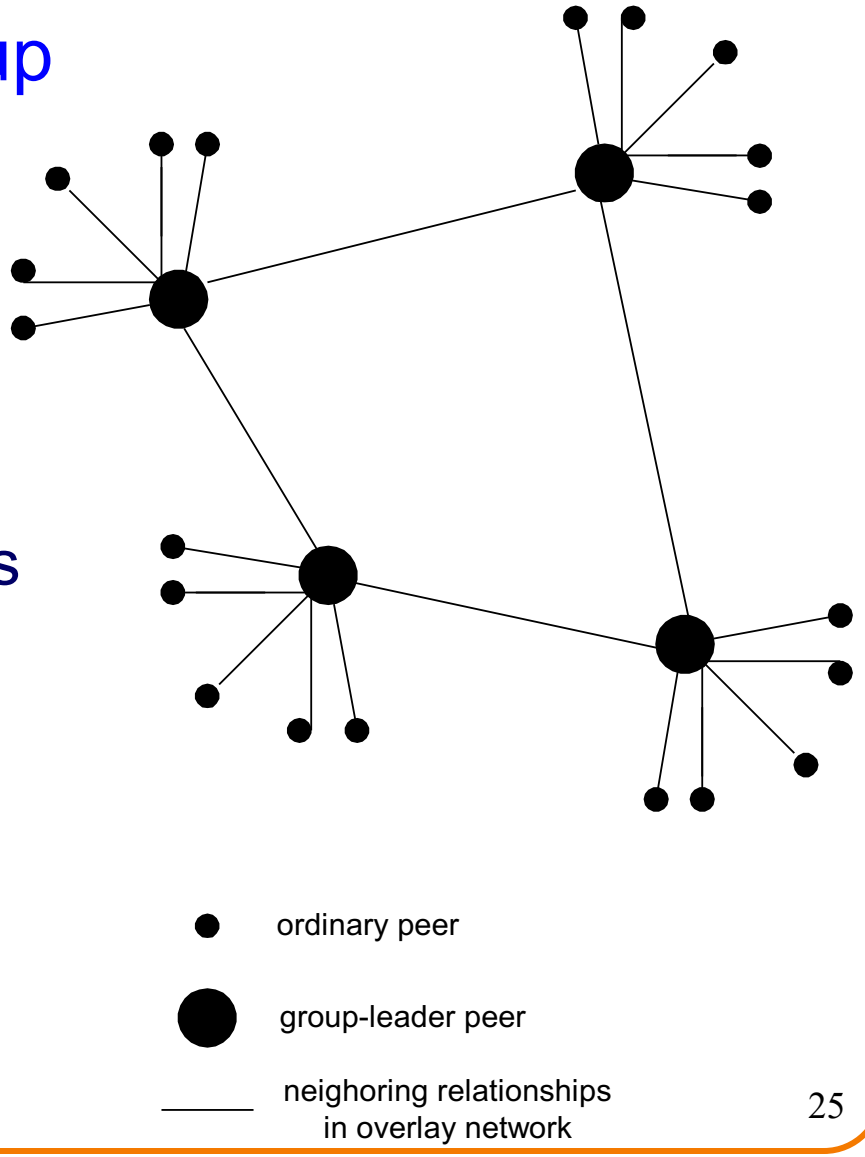
Peer-to-Peer Networks: KaAzA

- KaZaA history
  - 2001: created by Dutch company (Kazaa BV)
  - Single network called FastTrack used by other clients as well
  - Eventually the protocol changed so other clients could no longer talk to it

- Smart query flooding
  - Join: on start, the client contacts a super-node (and may later become one)
  - Publish: client sends list of files to its super-node
  - Search: send query to super-node, and the super-nodes flood queries among themselves
  - Fetch: get file directly from peer(s); can fetch from multiple peers at once

KaZaA: Exploiting Heterogeneity

- Each peer is either a group leader or assigned to a group leader
  - TCP connection between peer and its group leader
  - TCP connections between some pairs of group leaders
- Group leader tracks the content in all its children



KaZaA: 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

Peer-to-Peer Networks: BitTorrent

- BitTorrent history and motivation
  - 2002: B. Cohen debuted BitTorrent
  - Key motivation: popular content
    - Popularity exhibits temporal locality (Flash Crowds)
    - E.g., Slashdot effect, CNN Web site on 9/11, release of a new movie or game
  - Focused on efficient *fetching*, not searching
    - Distribute same file to many peers
    - Single publisher, many downloaders
  - Preventing free-loading

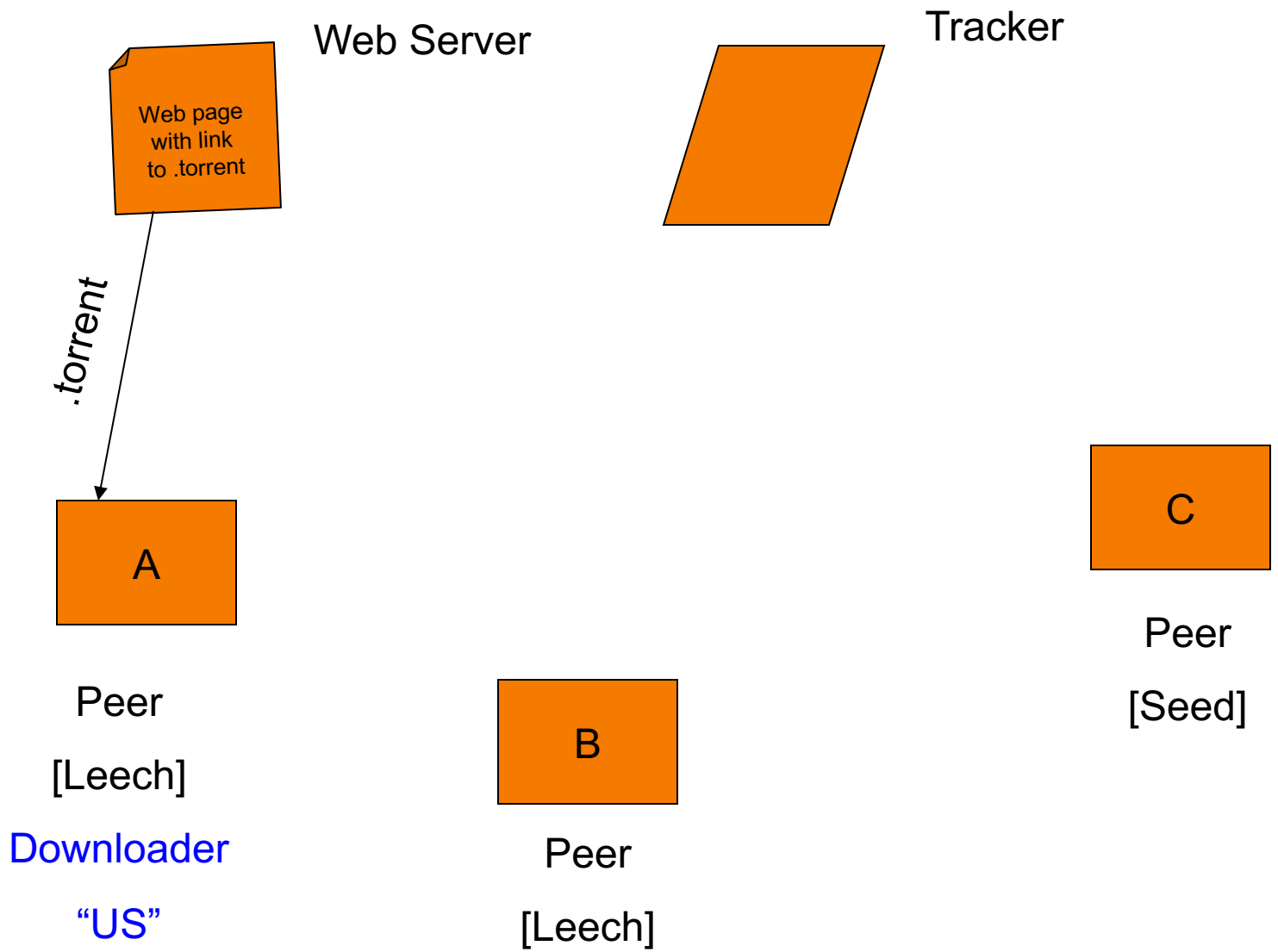
BitTorrent: Simultaneous Downloading

- Divide large file into many pieces
  - Replicate different pieces on different peers
  - A peer with a complete piece can trade with other peers
  - Peer can (hopefully) assemble the entire file
- Allows simultaneous downloading
  - Retrieving different parts of the file from different peers at the same time

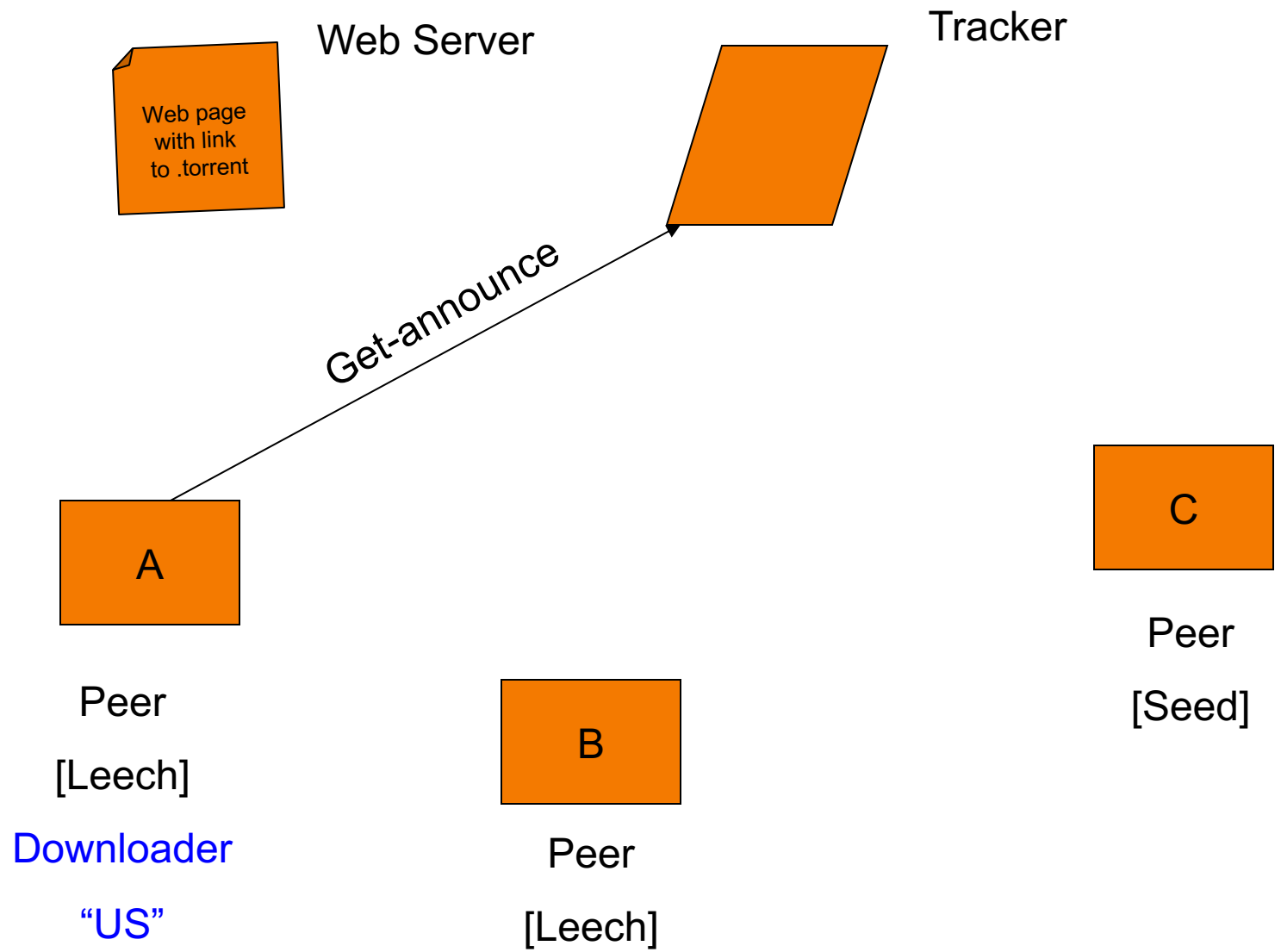
BitTorrent Components

- Seed
  - Peer with entire file
  - Fragmented in pieces
- Leacher
  - Peer with an incomplete copy of the file
- Torrent file
  - Passive component
  - Stores summaries of the pieces to allow peers to verify their integrity
- Tracker
  - Allows peers to find each other
  - Returns a list of random peers

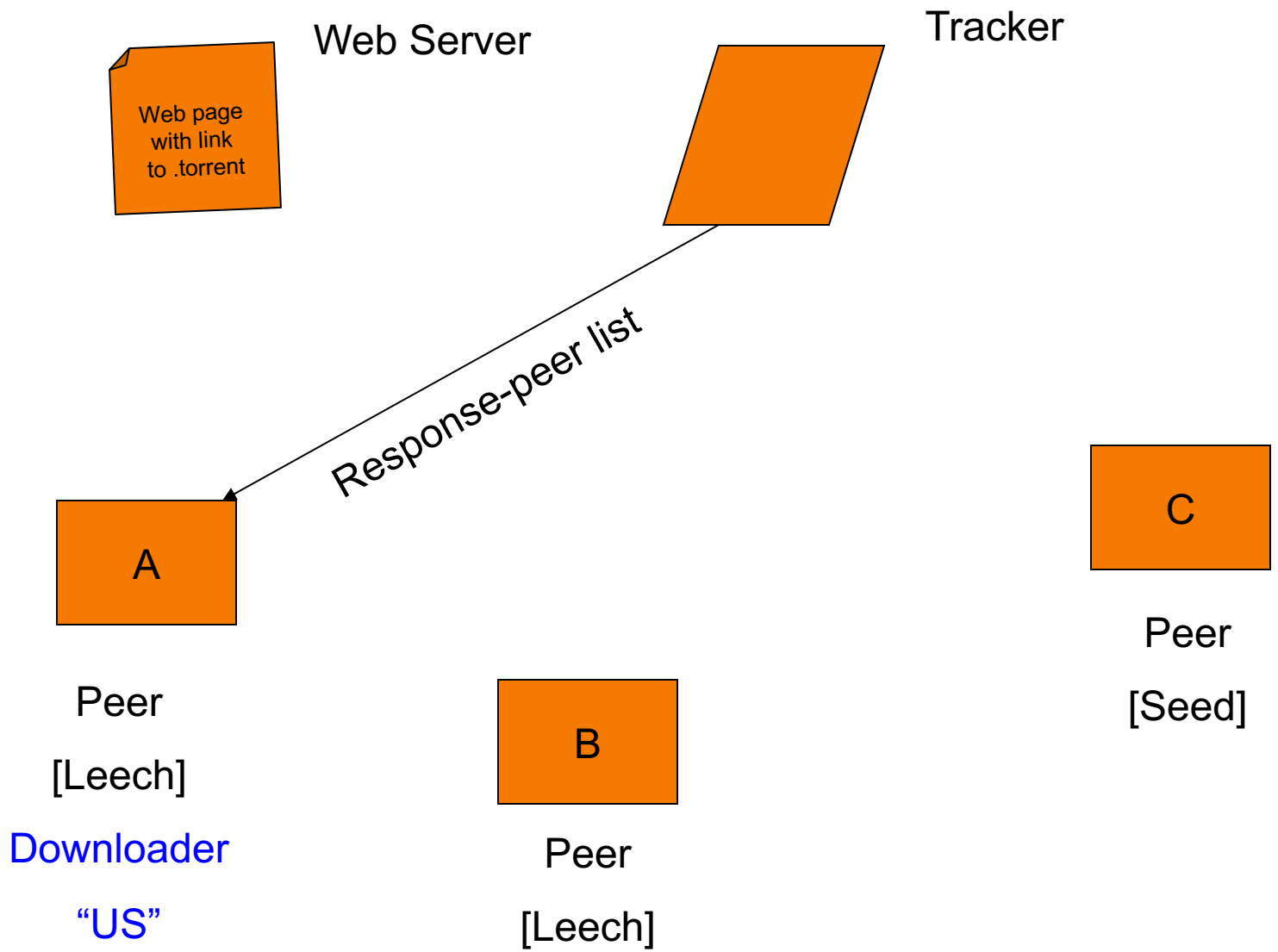
BitTorrent: Overall Architecture



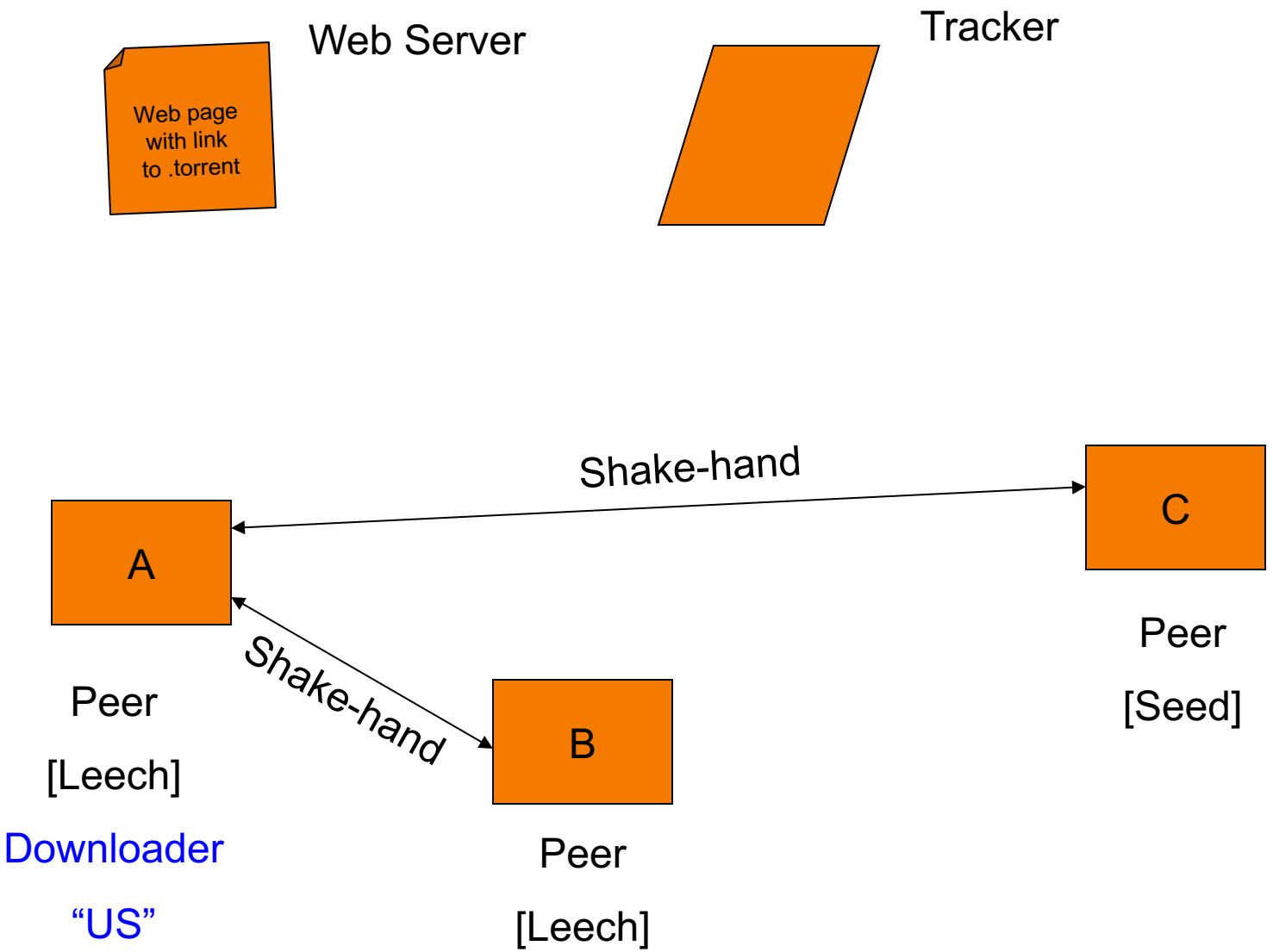
BitTorrent: Overall Architecture



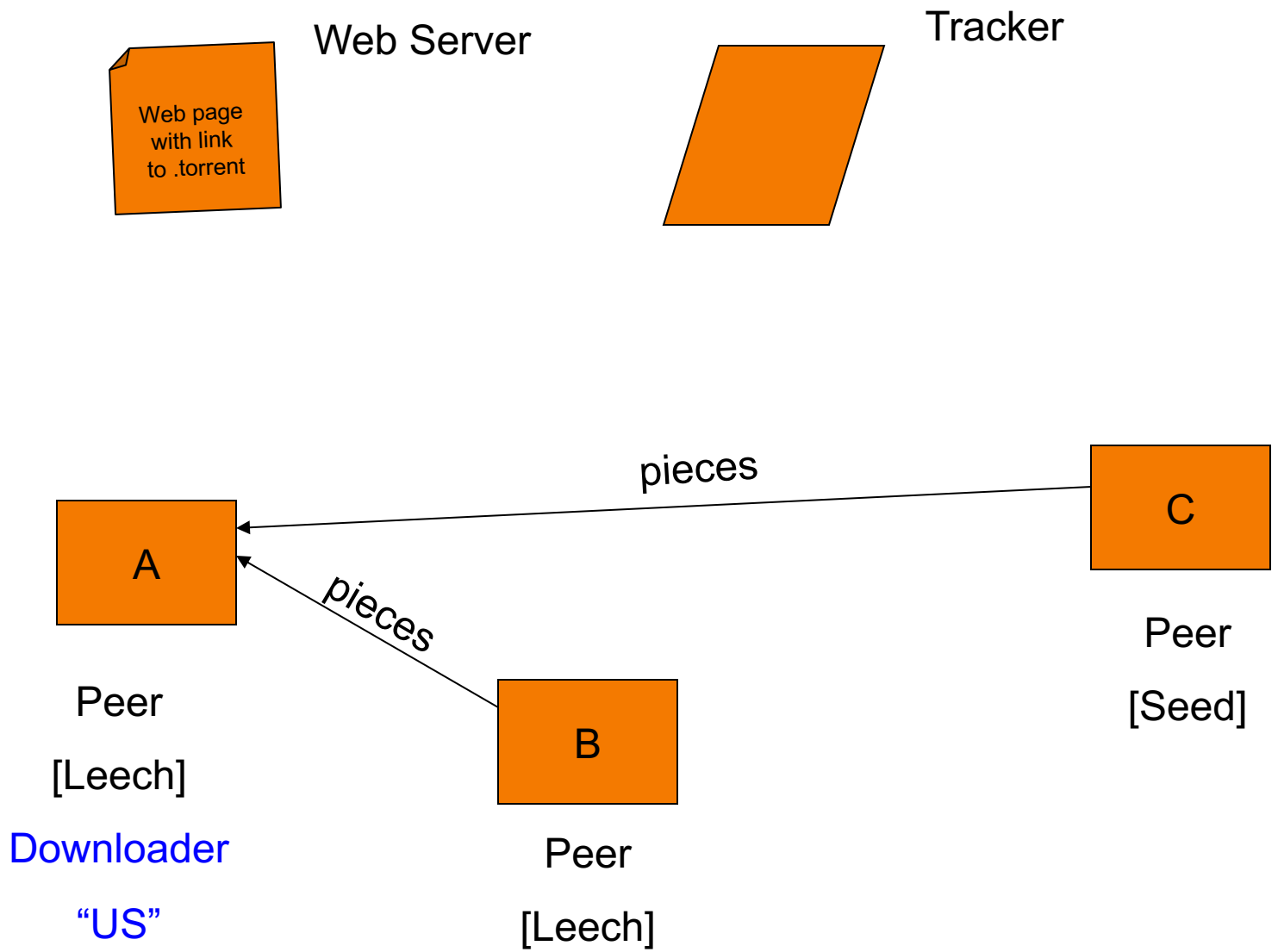
BitTorrent: Overall Architecture



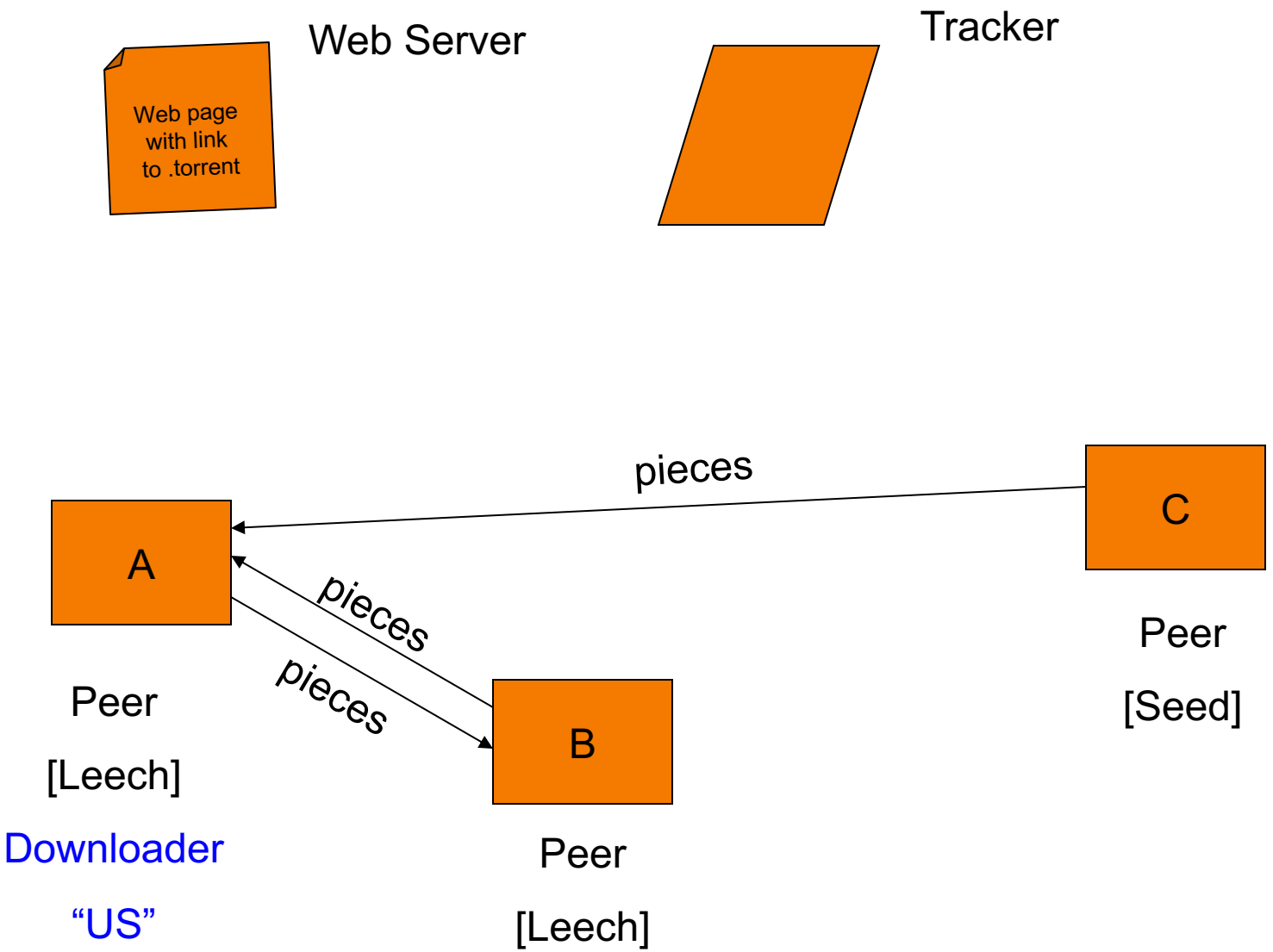
BitTorrent: Overall Architecture



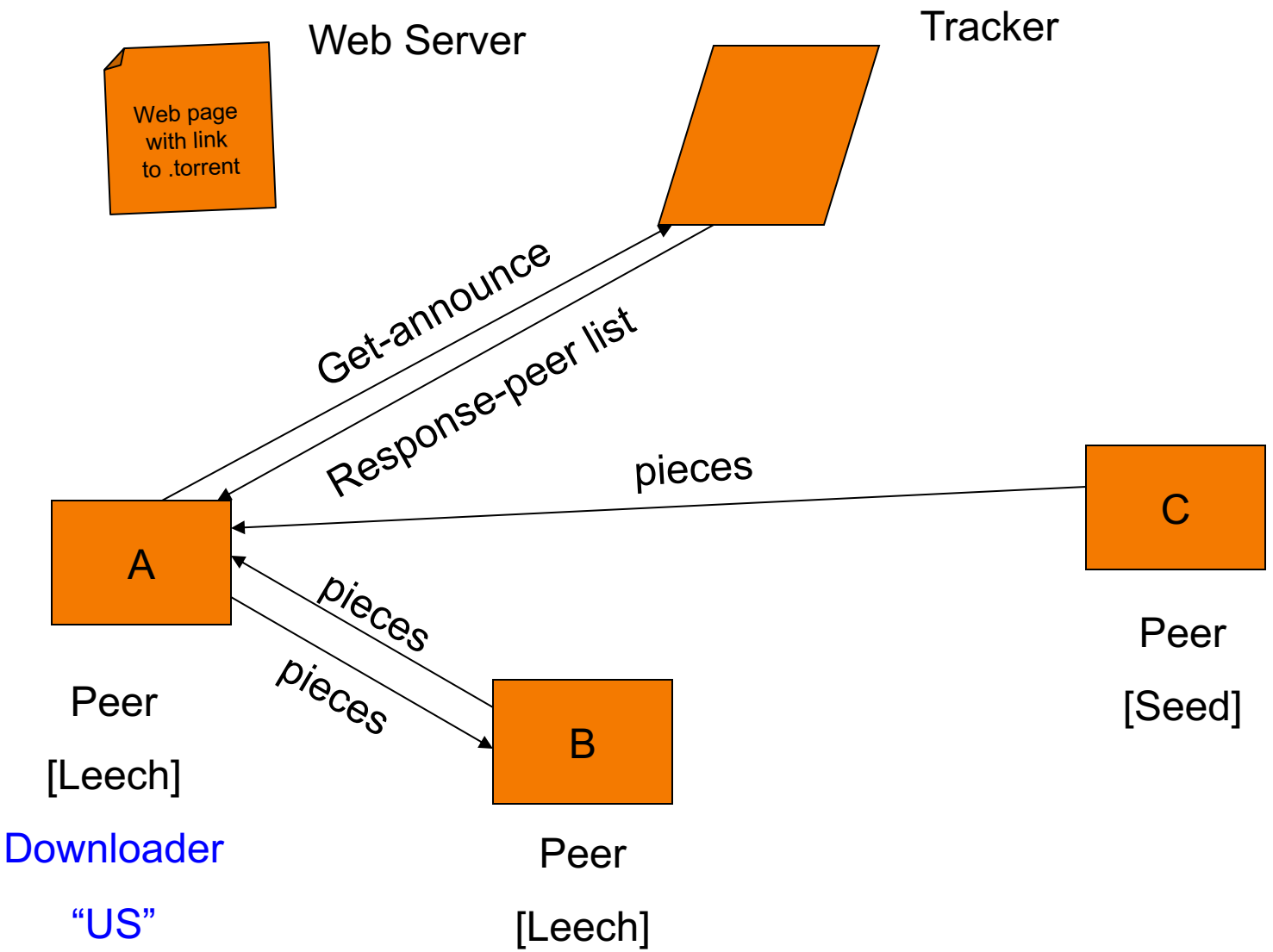
BitTorrent: Overall Architecture



BitTorrent: Overall Architecture



BitTorrent: Overall Architecture



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 prevent free riding
  - Allow the fastest peers to download from you
  - Occasionally let some free loaders download

Conclusions

- Overlay networks
  - Tunnels between host computers
  - Hosts implement new protocols and services
  - Effective way to build networks on top of the Internet
- Peer-to-peer networks
  - Nodes are end hosts
  - Primarily for file sharing, and recently telephony
  - Centralized directory (Napster), query flooding (Gnutella), super-nodes (KaZaA), and distributed downloading and anti-free-loading (BitTorrent)
- Great example of how change can happen so quickly in application-level protocols