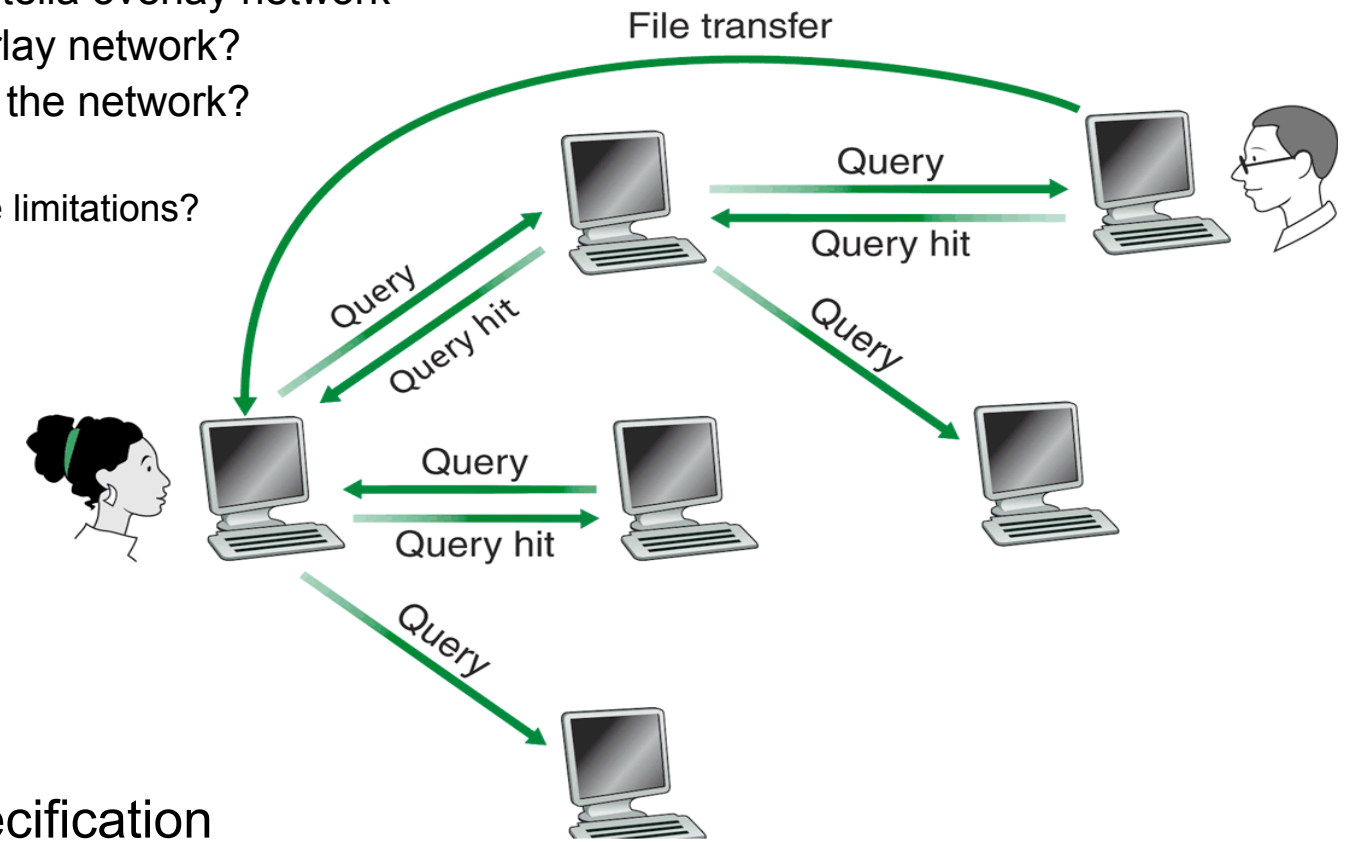
Examples Peer-to-Peer Applications

(Gnutella, Kazaa, BitTorrent, Skype)

Second generation approach

- Gnutella
  - Fully distributed approach
  - Constructs Gnutella overlay network
  - What is an overlay network?
  - How do we join the network?
  - Query flooding
    - Performance limitations?



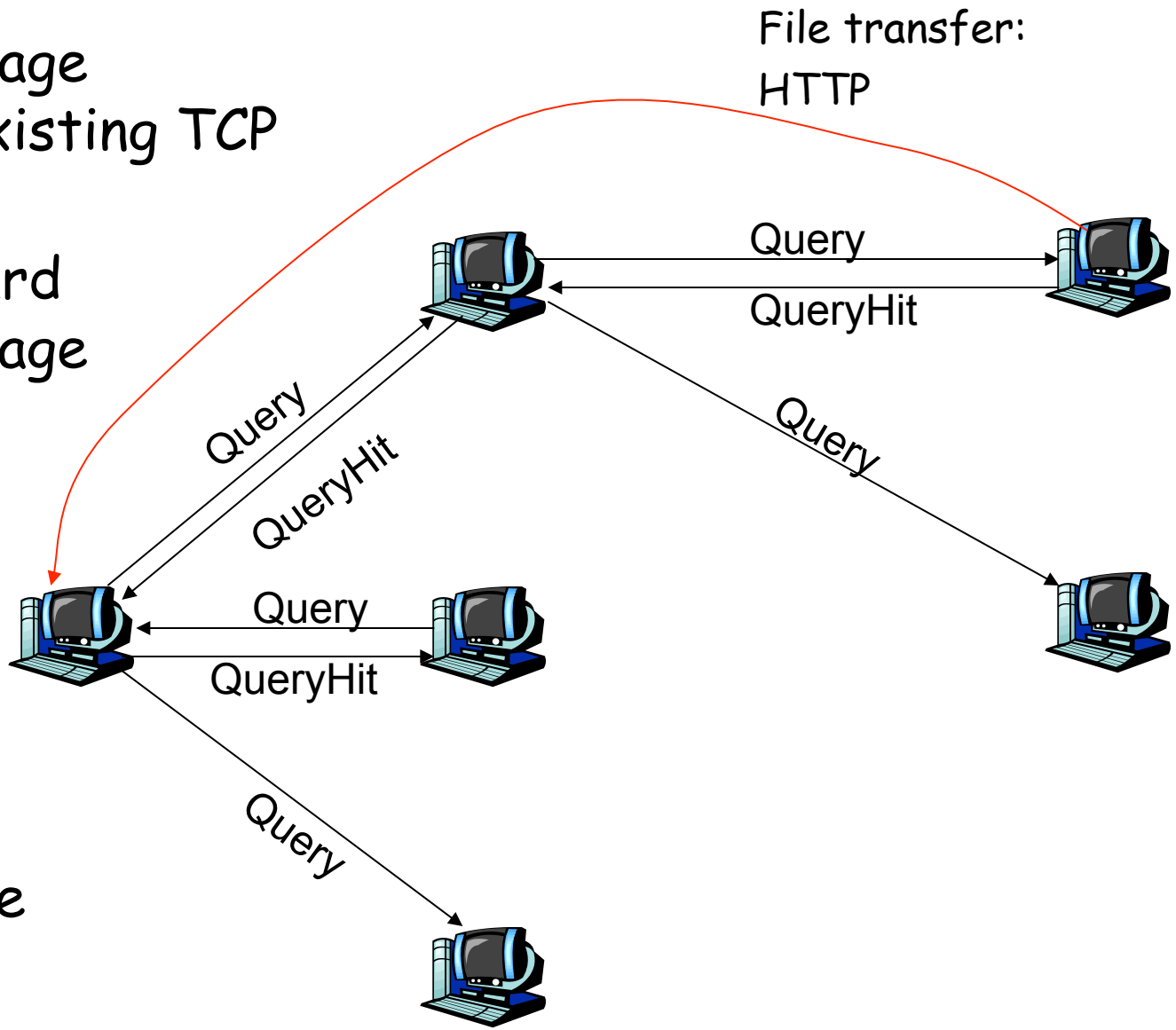
- Easy to read specification
  - http://rfc-gnutella.sourceforge.net/src/rfc-0_6-draft.html

Gnutella: protocol

Query message sent over existing TCP connections

peers forward Query message

QueryHit sent over reverse path

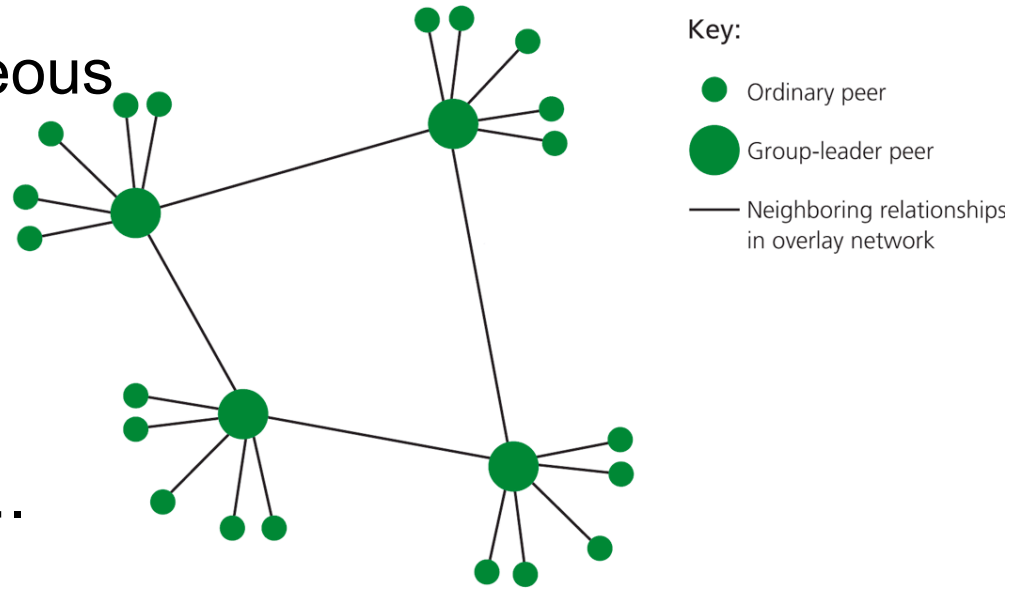


Scalability:

limited scope flooding

Kazaa: Not all peers are equal

- Basic idea
  - Peers and group leaders
    - 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
- Speed ups
  - Limitations on simultaneous uploads
  - Request queuing
  - Incentive priorities
  - Parallel downloading
- Leads to next generation….



BitTorrent

Please read this short but in my estimation brilliant paper:

http://www.bittorrent.org/bittorrentecon.pdf

The cool idea behind BitTorrent

- Make downloading as fast as possible and ignore search
- Solve the freeloader problem and therefore make downloading as fast as possible
- Tit for tat protocol makes sure that downloaders have to be uploaders too
- Hot content gets quickly distributed in the swarm

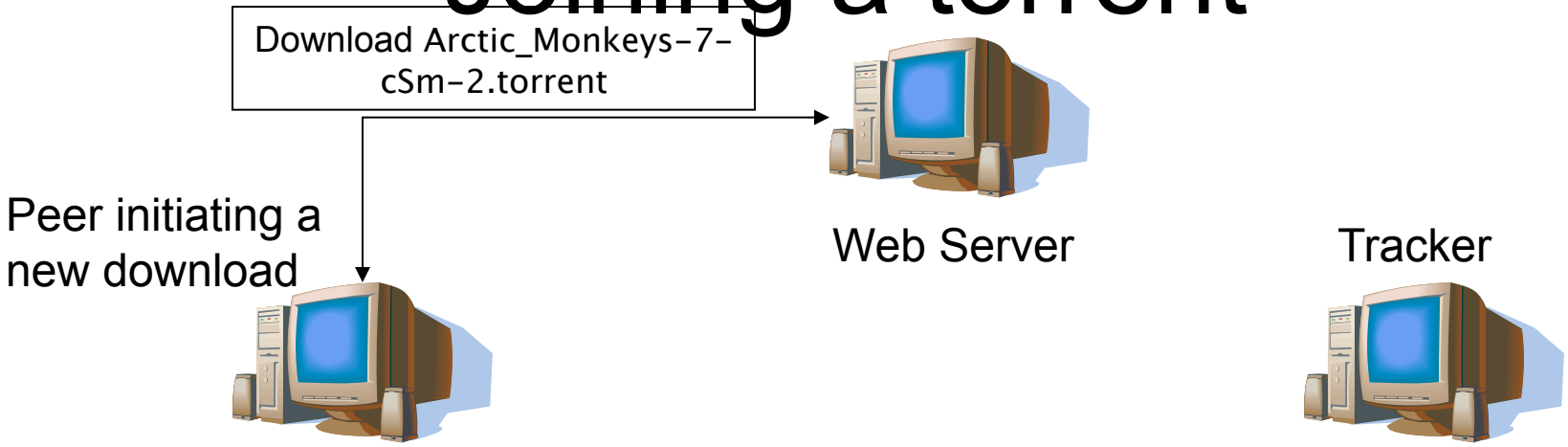
Piece Selection Strategies

- Strict priority
- Rarest first piece
- Random (really) first piece
- Endgame

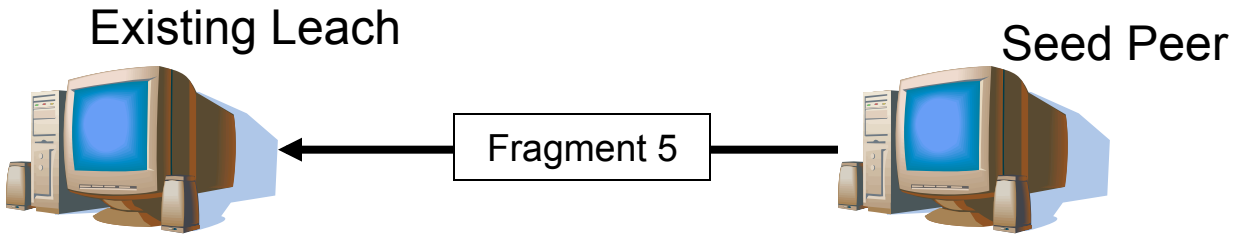
Choking Peers

- Choking algorithms
- Opportunistic Unchoking
- Anti snubbing
- Upload only

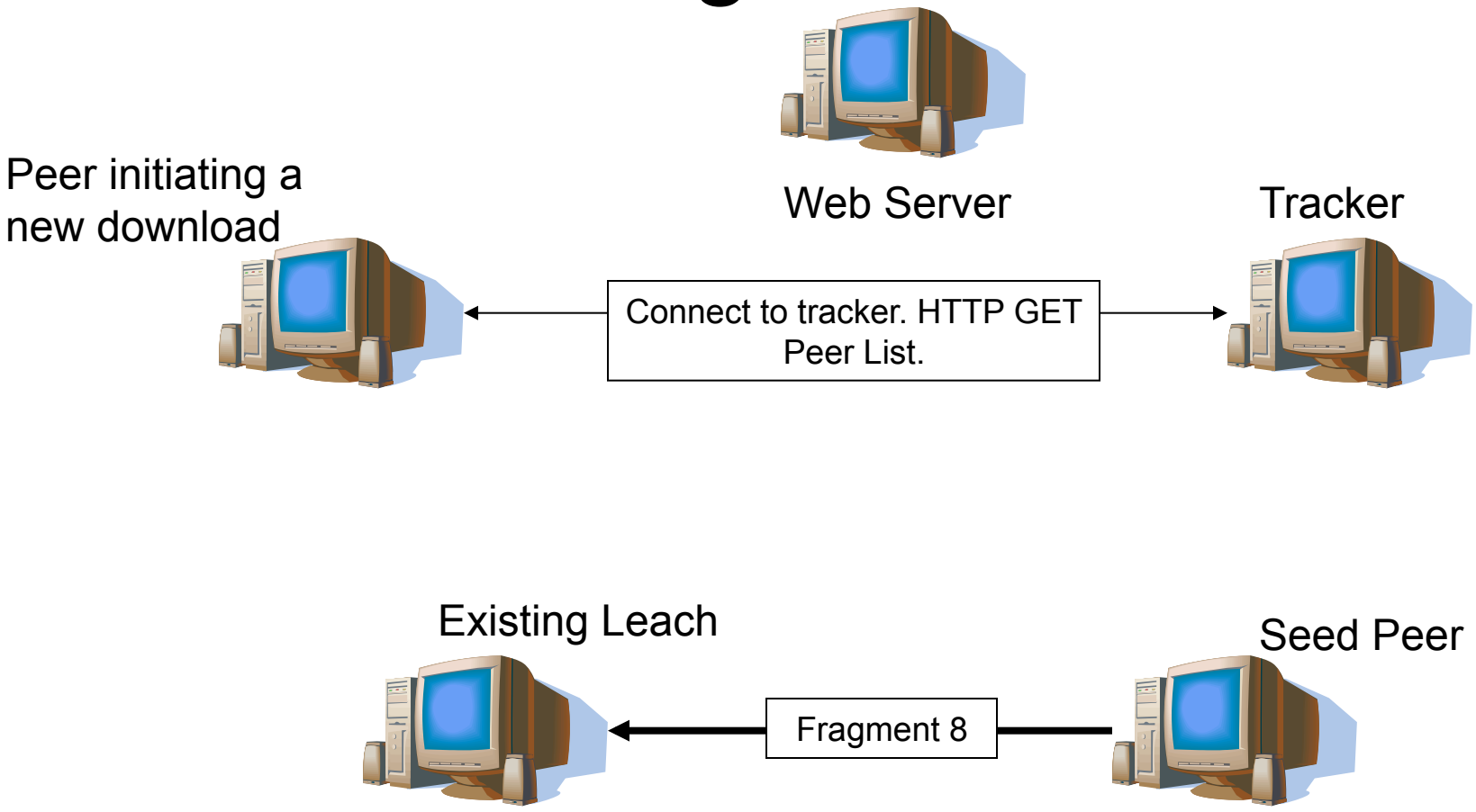
Joining a torrent



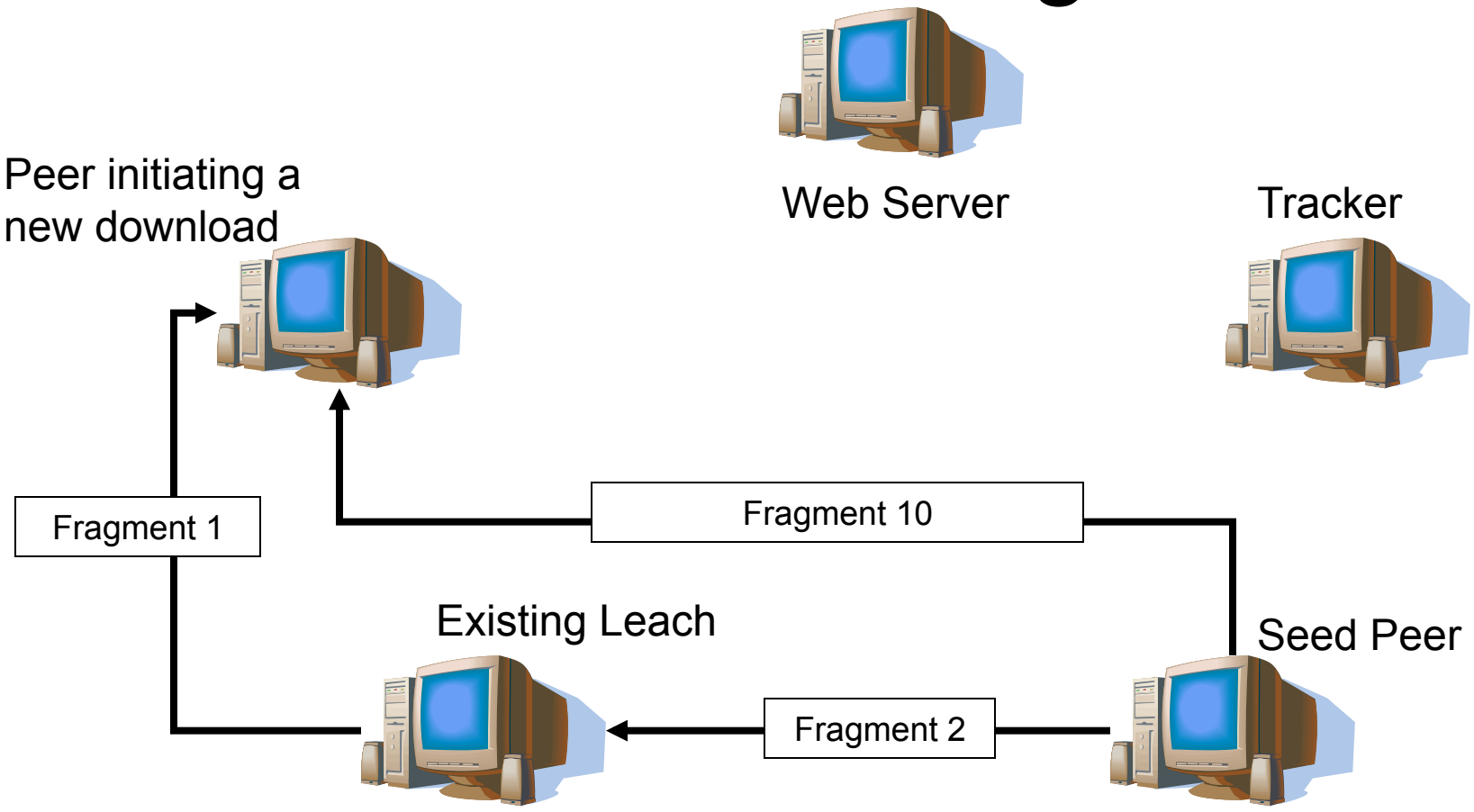
.torrent:
d8:announce30:http://tracker.prq.to/announce13:announce-listll48:http://open.tracker.thepiratebay.org:80/
:name48:Arctic_Monkeys-cSm12:piece lengthi65536e6:pieces13280



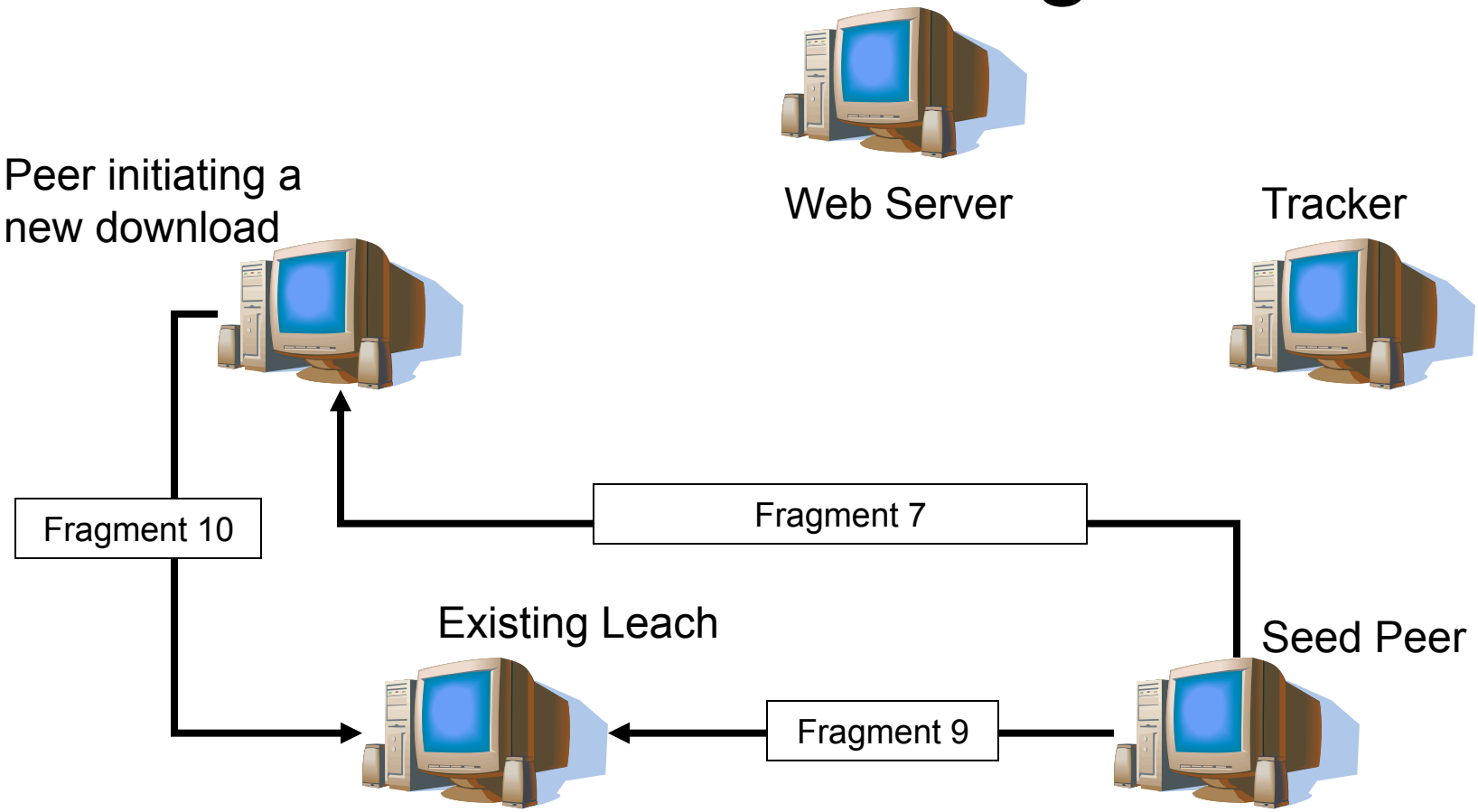
Joining a torrent



Data exchange



Data exchange



Algorithm: use hashes to verify pieces; download sub-pieces (fragments) in parallel; look for rarest pieces first; advertise received pieces to the entire peer list

Peer-to-Peer Internet Telephony Network

“Hello Mum!”

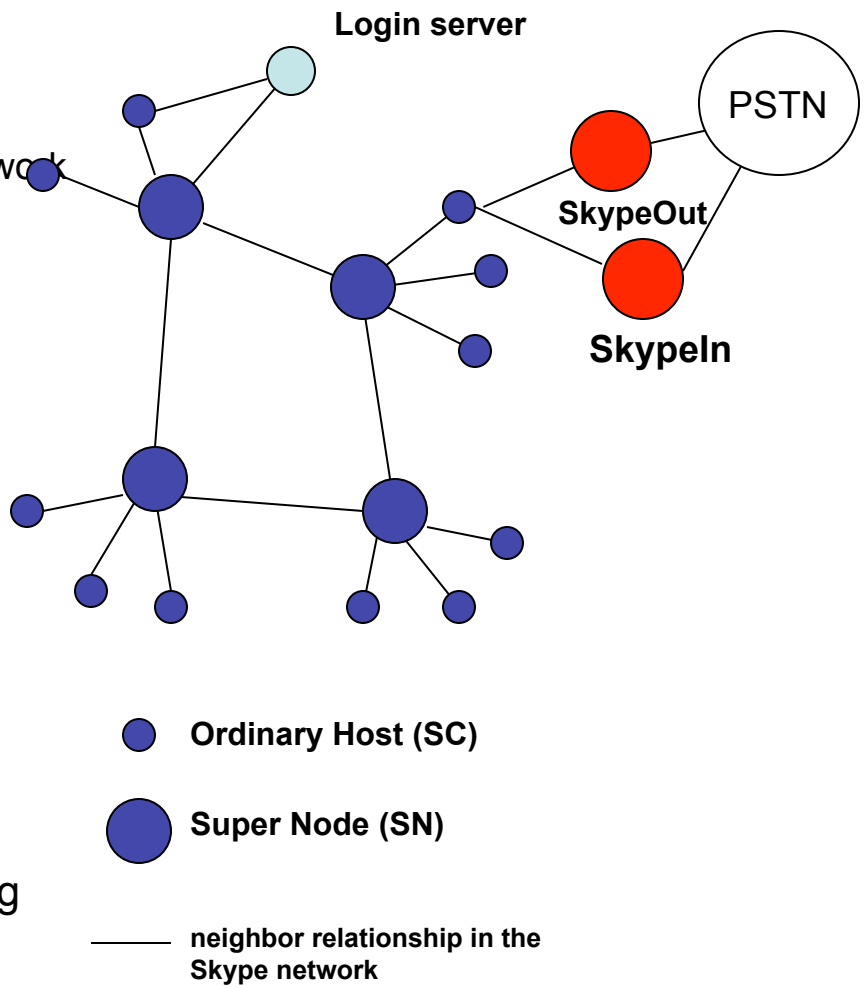
www.skype.com/

Client-side operations

- Login
- User search
- Start and end calls
- Media transfer
- Presence messages

Skype architecture

- Skype client (SC)
  - which is an application to start/receive calls, IM, send/receive files
- Super nodes (SN)
  - any node with sufficient CPU, memory, and network bandwidth can become a SN.
  - A SC can become a SN - and can't prevent this.
- Login server (LS)
  - stores user's profile and performs users authentication
  - Implements a public key infrastructure (PKI)
  - Infrastructure with clients
- Gateways
  - SkypeIn and SkypeOut servers provide PC-to-PSTN and PSTN-to-PC bridging, respectively
- Tricks
  - Network Address Translation (NAT) and firewall traversals are important Skype functions
- TCP for signaling, and both UDP and TCP for transporting media - different ports for signaling and data transfer
- Wideband codec for 5-32 kbps
- Unique skype user names across system

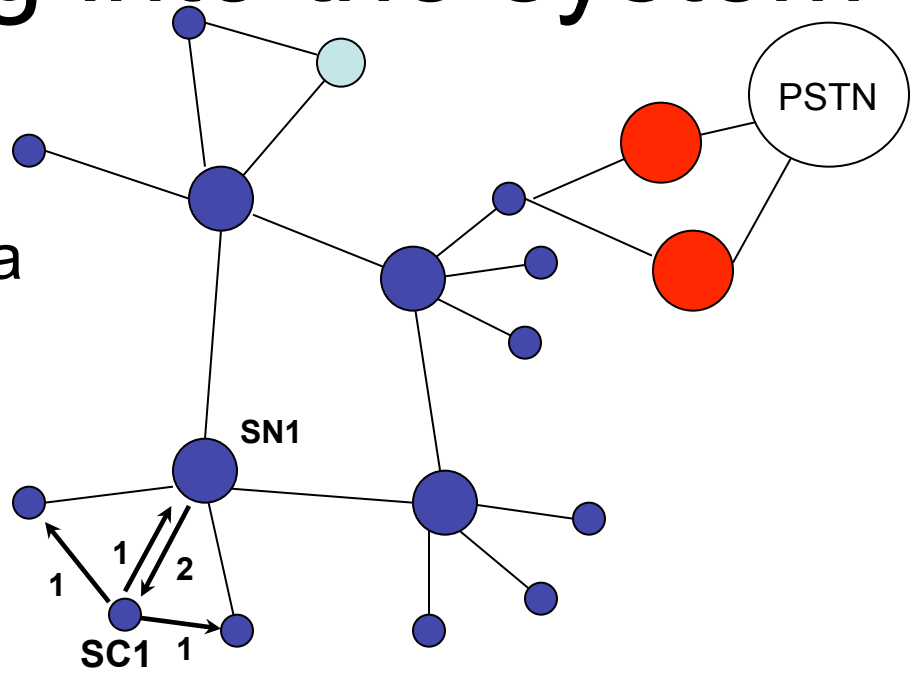


Observations

- The problem is that there are a lot of different configurations. Some routers allow outgoing connections but not incoming. Some others allow UDP (User Datagram Protocol) connections. Others allow TCP (Transmission Control Protocol). Most existing Internet telephony applications don't work that well in consumer environments.
- **How does Skype get around that?**
  When both parties are behind NATs (Network Address Translation), they can't actually set up a connection between each other.
- Use the overlay for connecting these peers
- Setting up hot standby connections. Set up four, maybe five standby paths.
- There are a few hundred clients per supernode.

Bootstrapping into the system

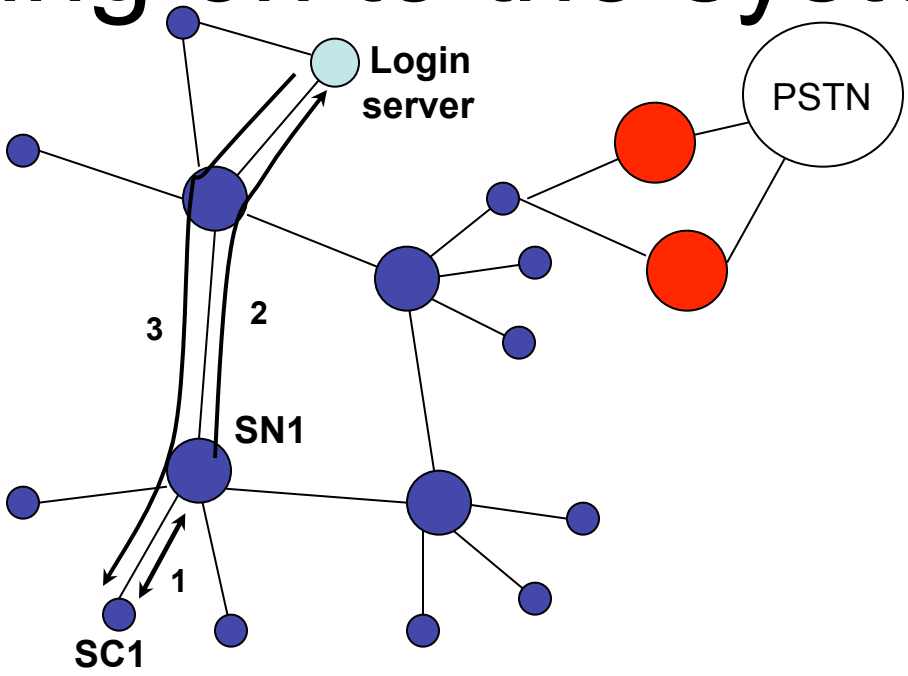
A SC builds and refreshes a host cache (HC) which is a list containing a set of SN's IP addresses-port number tuples.



(1) SC1 broadcasts a UDP lookup message to discover the closest SNs

(2) SN1 responds back to SC1 and SC1 puts SN1's IP address and TCP port number in the HC list (if SC1 does not get any response back it tries to associate with one of the seven IP address port pairs of SN provided at logon

Logging on to the system



(1) SC1 must establish a TCP connection and exchange information with SN1 in order to join the network. If the connection attempt fails the system returns "Login failure"

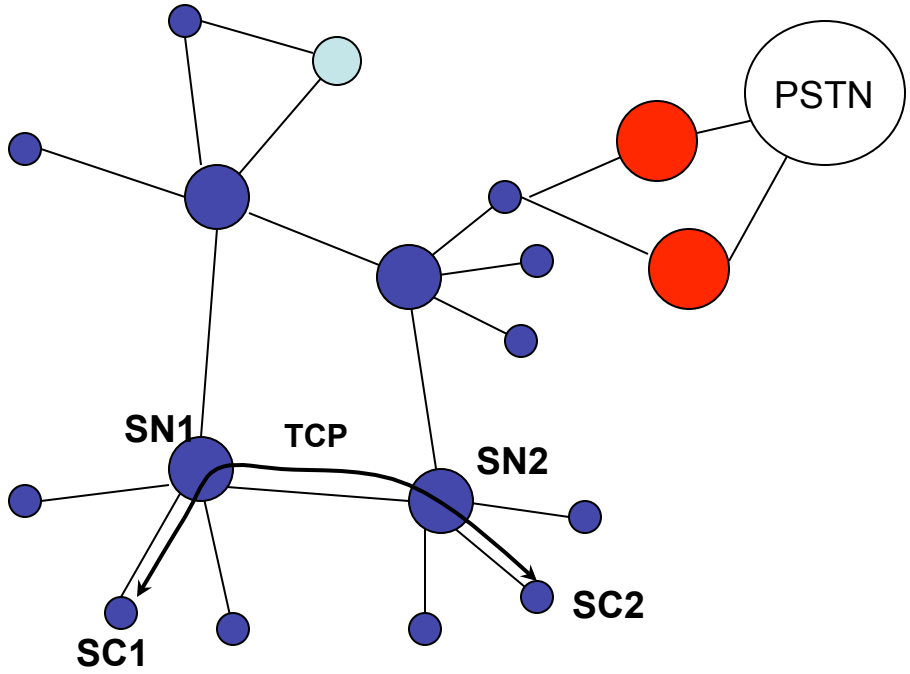
(2) SN1 forwards the authentication request to the Login server

(3) The login server authenticates SC1 and responds to SC1

User search

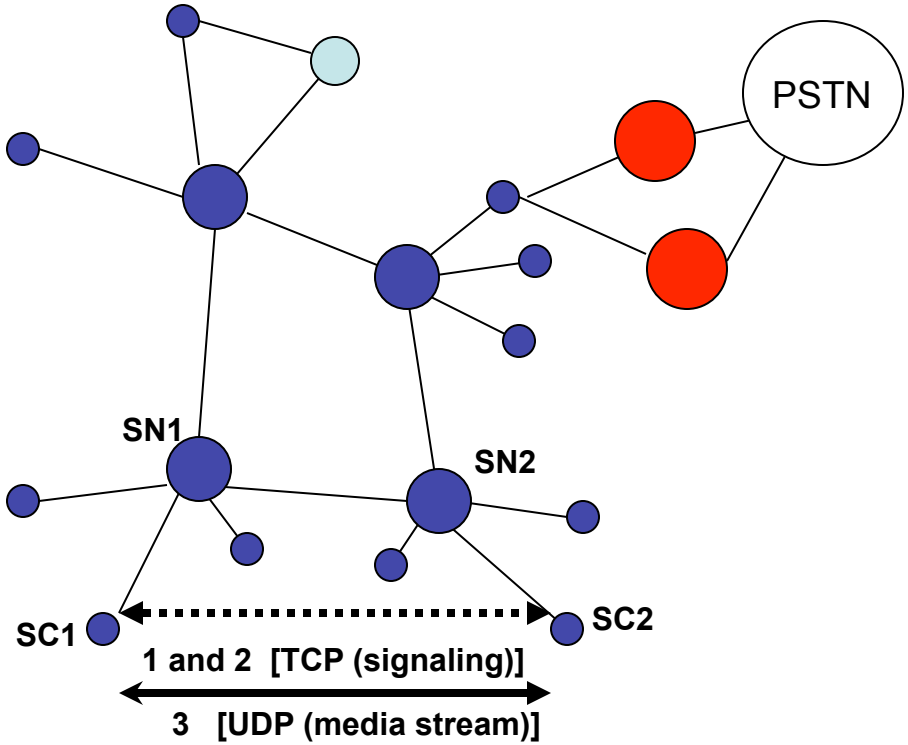
- Skype uses its proprietary Global Index technology to performs users search across the network in a scalable and distributes fashion

Control messaging over the P2P network



- SCs sends control traffic including availability information, user search, instant messages, and requests for VoIP and file transfer session over the skype P2P network. The control messages are sent over TCP

Call setup with public IP addresses



(1) Assume SC2 was in the buddy list of SC1 (if not run the search over TCP). The caller (SC1) presses dial and attempts to establish a TCP connection with the buddy SC2 to exchange signaling information

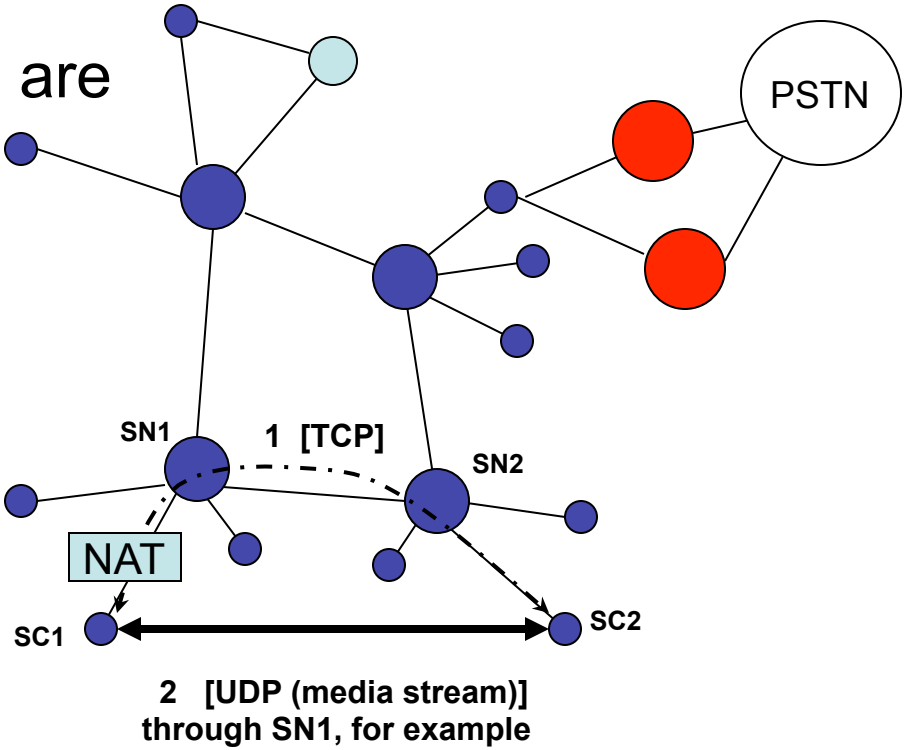
(2) SC2 responds back and a TCP connection is established

(3) Media stream flows over UDP being encrypted

Another call setup scenario

The “caller” (who initiates the call) is behind NAT (network address translator) and the “callee” (the party called) has public IP addresses

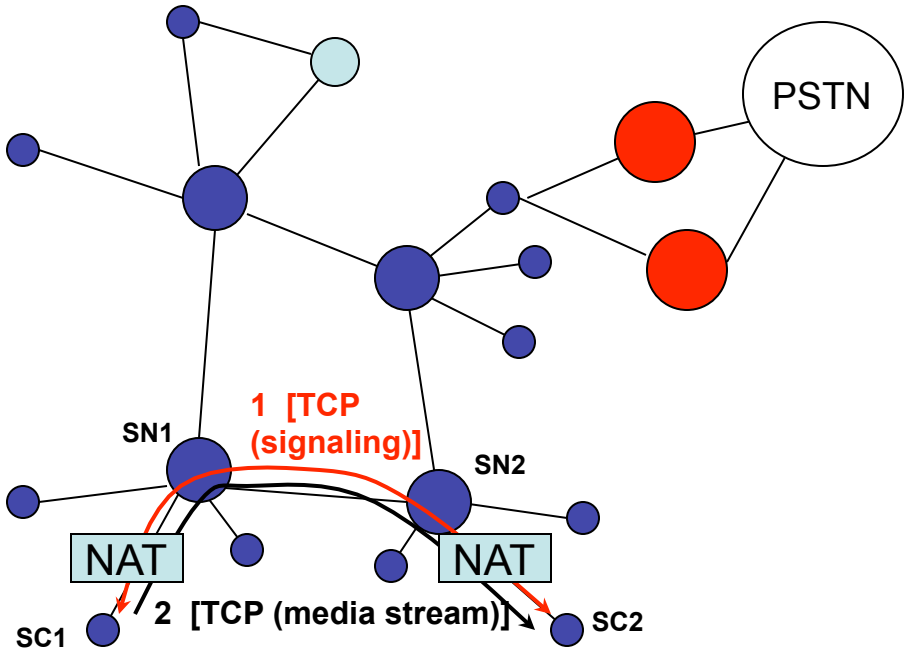
Note, that all broadband users are behind NATs and/or firewalls.



(1) Similar to the previous case but the TCP signaling exchange does not flow directly between the two SC 1 and SC 2 but rather through another intermediate node (SN1, for example)

(2) The media stream is transferred over UDP being encrypted but again through SN1

Call setup when both caller and callee are behind a NATs



(1) Similar to the previous case but the TCP signaling exchange does not flow directly between the two SCs

(2) The media stream is transferred over TCP being encrypted

Call setup and teardown

- For the call teardown the signaling flows over TCP connections either directly (both caller and callee having public IP addresses) or indirectly if either of them is behind a NAT (Network Address Translation - we will talk about this later in the course).

Silence suppression

- Skype uses efficient codec to encode voice → 5 KBps total uplink and bandwidth usage
- Skype doesn't perform silence suppression (interesting)
  - *Cons*: increase bandwidth usage
  - *Pros*: silence packets maintain UDP bindings at NAT and when the media traffic is sent over TCP silence packets avoid the drop in TCP congestion window size (which otherwise would take several RTT to reach the maximum level again)

Reverse engineering …

- Skype is propriety system not fully understood or documented, start here
  - http://en.wikipedia.org/wiki/Skype
  - there are a few papers that have reserved engineered the signaling system; this is a good one to read:
    - http://www1.cs.columbia.edu/~library/TR-repository/reports/reports-2004/cucs-039-04.pdf

Be entrepreneurial – take risks

- Niklas Zennstrom very cool entrepreneur and co-inventor of kazaa, skype, joost
  - http://news.com.com/2008-7352-5112783.html
    - Like this Q&A snippet:
      - **Are you afraid of intruders targeting your server that signs user ID keys?**
        The signing server is like Fort Knox.
      - **Where's it located?**
        I won't tell you. That's kind of a sensitive part of (the company). It's very, very secure.
  - Message: while young be entrepreneurial, be a risk taker!