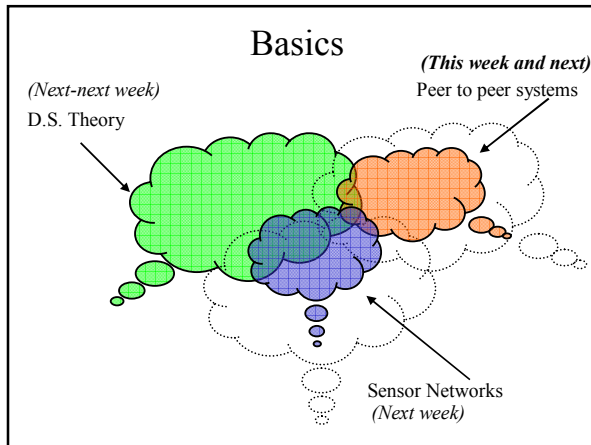
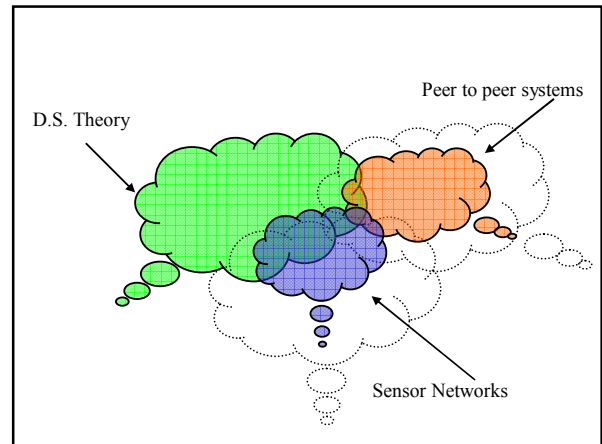


CS 525 Advanced Topics in Distributed Systems Spring 08

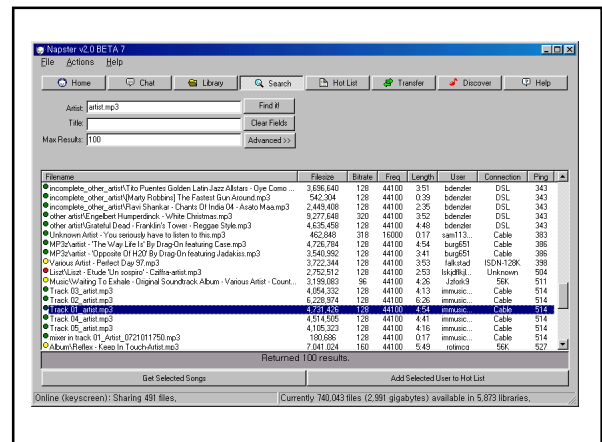
Indranil Gupta
Lecture 2
Introduction to Peer to Peer Systems
January 17, 2008



Some Questions

- Why do people get together?
 - to share information
 - to share and exchange resources they have
 - books, class notes, experiences, videos, music cd's
- How can computers help people
 - find information
 - find resources
 - exchange and share resources

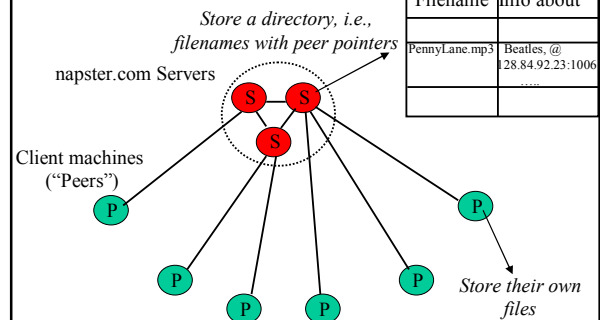
- Existing technologies: The Web!
 - Search engines
 - Forums: chat rooms, blogs, ebay
 - Online business
- But, the web is heavy weight if you want specific resources: say a Beatles' song "PennyLane"
- A google search will give you their bio, lyrics, chords, articles on them, and then perhaps the mp3
- But you want only the song, nothing else!
- **If you can find a peer who wouldn't mind exchanging her Beatles mp3 songs for your UIUC Homecoming videos, that would be great!**
 - Napster: a solution light weight that was lighter than the Web



A Brief History

- [6/99] Shawn Fanning (freshman Northeastern U.) releases Napster online music service
- [12/99] RIAA sues Napster, asking \$100K per download
- [3/00] 25% UWisc traffic Napster, many universities ban it
- [00] 60M users
- [2/01] US Federal Appeals Court: users violating copyright laws, Napster is abetting this
- [9/01] Napster decides to run paid service, pay % to songwriters and music companies
- [Today] Napster protocol is open, people free to develop `opennap` clients and servers
<http://opennap.sourceforge.net>

Napster Structure

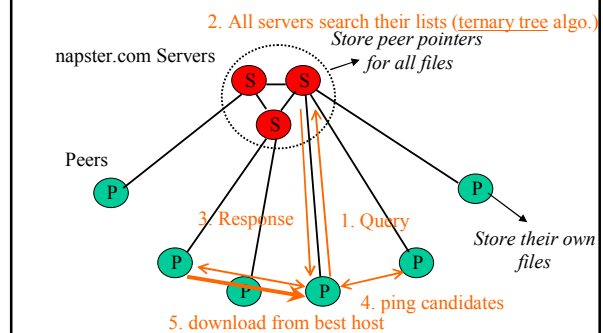


Napster Operations

Client

- Connect to a Napster server
- Upload list of music files that you want to share
 - Server maintains list of <filename, ip_address, portnum> tuples
- Search
 - Send server keywords to search with
 - (Server searches its list with the keywords)
 - Server returns a list of hosts - <ip_address, portnum> tuples - to client
 - Client pings each host in the list to find transfer rates
 - Client fetches file from best host
- All communication uses TCP

Napster Search



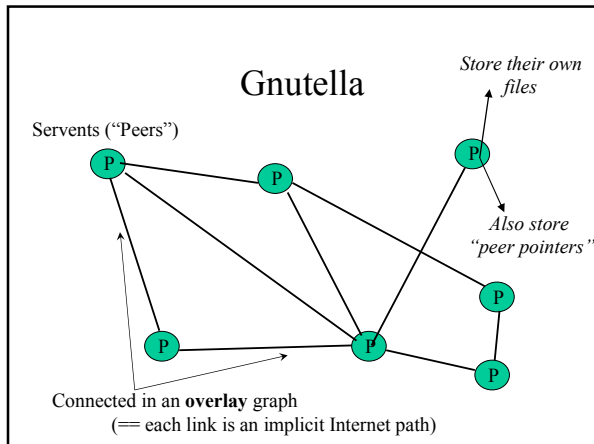
Problems

- Centralized server a source of congestion
- Centralized server single point of failure
- No security: plaintext messages and passwds
- napster.com responsible for users' copyright violation
 - "Indirect infringement"

Gnutella

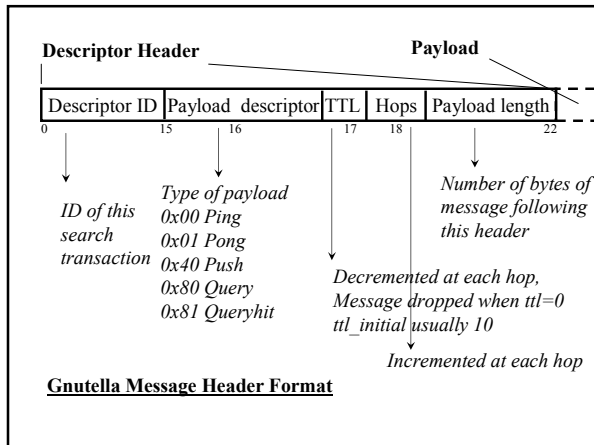
- Eliminate the servers
- Client machines search and retrieve amongst themselves
- Clients act as servers too, called **servents**
- [3/00] release by AOL, immediately withdrawn, but 88K users by 3/03
- Original design underwent several modifications

<http://www.limewire.com>

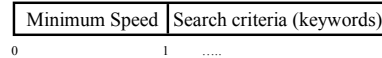


How do I search for my Beatles file?

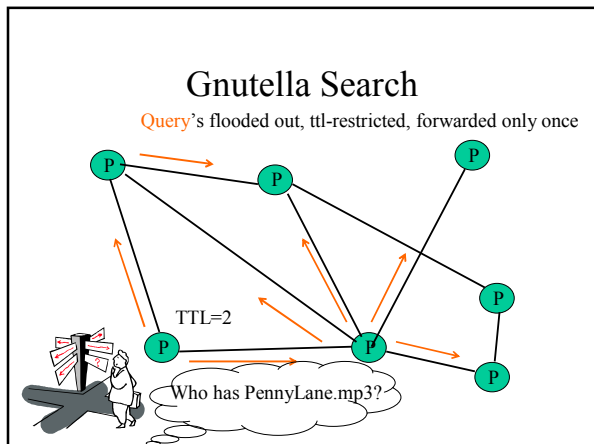
- Gnutella *routes* different messages within the overlay graph
- Gnutella protocol has 5 main message types
 - Query (search)
 - QueryHit (response to query)
 - Ping (to probe network for other peers)
 - Pong (reply to ping, contains address of another peer)
 - Push (used to initiate file transfer)
- We'll go into the message structure and protocol now (note: all fields except IP address are in little-endian format)



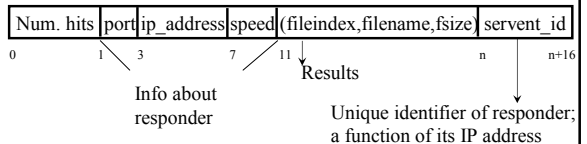
Query (0x80)



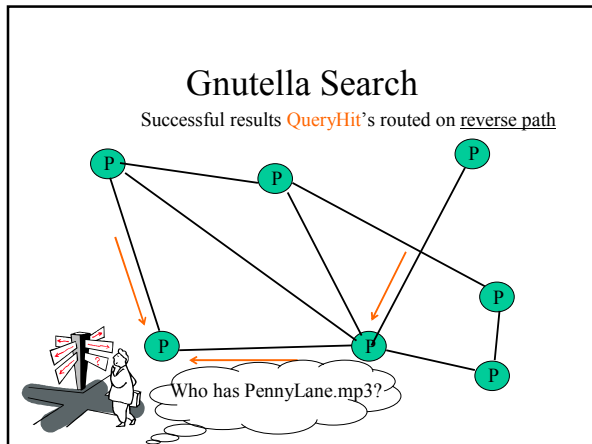
Payload Format in Gnutella Query Message



QueryHit (0x81) : successful result to a query



Payload Format in Gnutella Query Reply Message

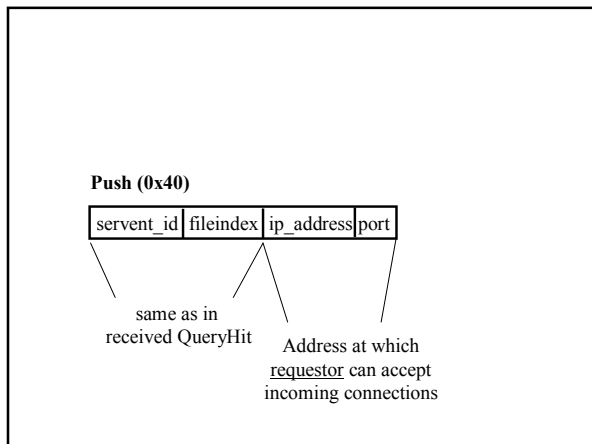
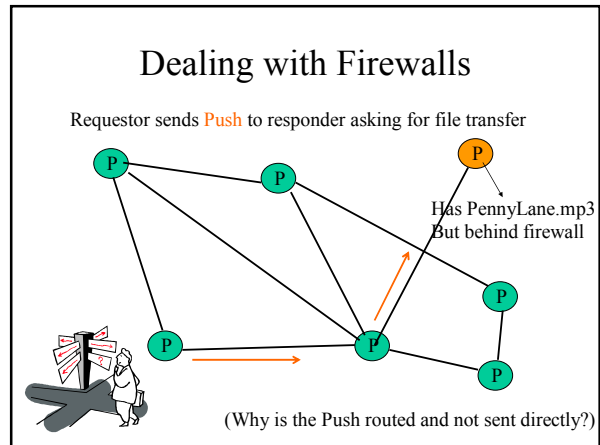


- ### Avoiding excessive traffic
- To avoid duplicate transmissions, each peer maintains a list of recently received messages
- Query forwarded to all neighbors except peer from which received
 - Each Query (identified by DescriptorID) forwarded only once
 - QueryHit routed back only to peer from which Query received with same DescriptorID
 - Duplicates with same DescriptorID and Payload descriptor (msg type) are dropped
 - QueryHit with DescriptorID for which Query not seen is dropped

- ### After receiving QueryHit messages
- Requestor chooses "best" QueryHit responder
 - Initiates HTTP request directly to responder's ip+port

```
GET /get/<File Index>/<File Name>/HTTP/1.0\r\n
Connection: Keep-Alive\r\n
Range: bytes=0-0\r\n
User-Agent: Gnutella\r\n
\r\n
```
 - Responder then replies with file packets following this message:


```
HTTP 200 OK\r\n
Server: Gnutella\r\n
Content-type: application/binary\r\n
Content-length: 1024 \r\n
\r\n
```
 - HTTP is the file transfer protocol. Why?
 - Why the "range" field in the GET request?
 - What if responder is behind firewall that disallows incoming connections?



- Responder establishes a TCP connection at ip_address, port specified. Sends


```
GIV <File Index>:<Servent Identifier>/<File Name>\n\n
```
- Requestor then sends GET to responder (as before) and file is transferred
- What if requestor is behind firewall too?
 - Gnutella gives up
 - Can you think of an alternative solution?

Ping-Pong

Ping (0x00)
no payload

Pong (0x01)

Port	ip_address	Num. files shared	Num. KB shared
------	------------	-------------------	----------------

- Peers initiate Ping's periodically
- Ping's flooded out like Query's, Pong's routed along reverse path like QueryHit's
- Pong replies used to update set of neighboring peers
 - to keep neighbor lists fresh in spite of peers joining, leaving and failing

Gnutella Summary

- No servers
- Peers/servents maintain "neighbors", this forms an overlay graph
- Peers store their own files
- Queries flooded out, ttl restricted
- Query Replies reverse path routed
- Supports file transfer through firewalls
- Periodic Ping-pong to continuously refresh neighbor lists
 - List size specified by user at peer : heterogeneity means some peers may have more neighbors
 - Gnutella found to follow **power law** distribution:
 $P(\#links = L) \sim L^{-k}$ (k is a constant)

Problems

- Ping/Pong constituted 50% traffic
 - Solution: Multiplex, *cache* and reduce frequency of pings/pongs
- Repeated searches with same keywords
 - Solution: *Cache* Query, QueryHit messages
- Modem-connected hosts do not have enough bandwidth for passing Gnutella traffic
 - Solution: use a central server to act as proxy for such peers
 - Another solution:
 - FastTrack System (in a few slides)

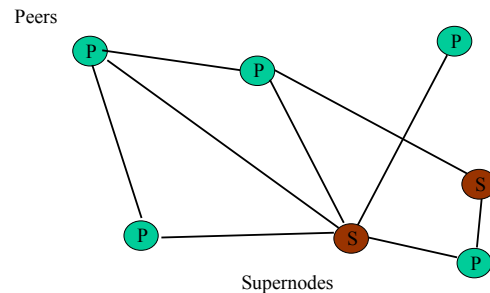
Problems (contd.)

- Large number of *freeloaders*
 - 70% of users in 2000 were freeloaders
- Flooding causes excessive traffic
 - Is there some way of maintaining meta-information about peers that leads to more intelligent routing?
 - Structured Peer-to-peer systems
e.g., Chord System (next lecture)

FastTrack

- Hybrid between Gnutella and Napster
- Takes advantage of "healthier" participants in the system
- Underlying technology in Kazaa, KazaaLite, Grokster
- Proprietary protocol, but some details available
- Like Gnutella, but with some peers designated as *supernodes*

A FastTrack-like System



FastTrack (contd.)

- A supernode stores a directory listing (<filename,peer pointer>), similar to Napster servers
- Supernode membership changes over time
- Any peer can become (and stay) a supernode, provided it has earned enough *reputation*
 - Kazaalite: participation level of a user between 0 and 1000, initially 10, then affected by length of periods of connectivity and total number of uploads
 - More sophisticated Reputation schemes now a research area. Use of economic theory in distributed systems!
- A peer searches by contacting a nearby supernode

Wrap-up Notes

Applies to all p2p systems

- How does a peer join the system
 - Send an http request to well known url
<http://www.myp2pservice.com>
 - Message routed (after DNS lookup) to a well known server that then initializes new peers' neighbor table
- Lookups can be speeded up by having each peer cache:
 - Queries and their results that it sees
 - All directory entries (filename,host) mappings that it sees
 - The files from the above

Summary

- Napster: protocol overview, more details available on webpage
- Gnutella protocol
- FastTrack protocol
- Protocols continually evolving, software for new clients and servers conforming to respective protocols: developer forums at
 - Napster: <http://opennap.sourceforge.net>
 - Gnutella: <http://www.limewire.com>
- Others
 - Peer to peer working groups: <http://www.p2pwg.com>

DHT=Distributed Hash Table

- A hash table allows you to insert, lookup and delete objects with keys
- A *distributed* hash table allows you to do the same in a distributed setting (objects=files)
- Napster, Gnutella, FastTrack are all DHTs (sort of)
- So is Chord, a structured peer to peer system that we study next

As you leave Today's Lecture...

What new *design techniques* have you learnt during this lecture?

Think about at least 3 new techniques you have learnt in this class.

For this Weekend

- 1/22: Read "Chord" paper from website
- 1/24: Browse "Sensor Networks" links from website
- Look for prospective presentation sessions – sign up early and you'll get your top preferences
- No reviews required yet