CS514: Intermediate Course in Computer Systems

Lecture 33: April 11, 2003

"*Peer-to-peer protocols for file and data replication: file sharing*"

File sharing is nothing new

CS514

- Has been going on over IRC (Internet Relay Chat) for years
  - Chat groups focusing on certain artists or genres
- Upload servers were popular for a while
  - Users would upload (FTP) a song
  - This allowed them to download N songs
  - Performance typically sucked
  - The record industry shut these down
- In all the above it took user effort to find what was wanted

Napster changed everything

CS514

- Central search engine, but peer-to-peer file transfer
- 160+ search engines at peek
  - User attaches to one of them
  - Engine would index collections of its own active users
  - Search on a given engine returned results from that engine
  - Unless not enough results, then would ask other engines
    - This is what I understand from Saroiu et.al. U-Wash measurement paper
- Peer-to-peer file transfer improved scalability

Napster problem

CS514

- As we all know, the problem with Napster is that it is a single point of litigation
- Gnutella designed as a lawyer-resilient Napster
  - Nothing more, nothing less (in my opinion)

In the meanwhile…

CS514

- Ian Clarke (Edinburgh, 1999*) was thinking about P2P from an anonymity perspective
  - He was interested in free speech
  - Whistleblowers, political dissent, etc.
- Ian designed Freenet as a class project
- Freenet is not so much file sharing as it is a publishing medium

* Before Gnutella

Freenet, Gnutella, and DHTs: One out of three…

CS514

	Scalability	Keyword search	Anonymity
DHT	■		
Freenet			■
Gnutella		■	

Get-em-out-quick projects

CS514

- Both Gnutella and Freenet were quick-and-dirty prototypes
  - Neither really worked through all the issues
- As a result, both are deeply flawed
- Nevertheless, both captured the popular imagination, and so are worth talking about
  - And both have spawned new work
  - For instance, FastTrack (Kazaa, Grokster …) spawned from Gnutella

Lets look at these aspects

CS514

- System Semantics
- Bootstrap
- Neighbor discovery and selection
- Network (neighbor) maintenance
- File insertion and storage
- File search and retrieval
- File deletion

System Semantics

CS514

- DHT:
  - Hash Table (general purpose)
- Gnutella:
  - Keyword search (with wildcard) of file name and metadata
- Freenet
  - Hash of file contents
  - Hash of file-owner public key plus file name
    - Allows for owner-write, anyone-read “subspaces”

Bootstrap

CS514

- DHT:
  - Have to know at least one active member
- Gnutella:
  - Have to know at least one active member
- Freenet:
  - Have to know at least one active member
- Scaling issue for everybody
- But fundamental operational issue for Gnutella (which cannot have any central point of control)

Bootstrap

CS514

- Various possible ways to know a current member
  - Email from friends, web site with list of addresses, rendezvous server with active knowledge of P2P network
  - Pastry folks suggest using a universal DHT to bootstrap other DHTs
- Gnutella uses rendezvous server approach, called “pong server” or “host cache”!!!
  - Host cache lists are distributed with software
    - They may change, so also listed on various websites, forums, etc.
  - Single point of litigation!
  - Scaling issue also

Neighbor discovery and selection

CS514

- DHT:
  - Search network, download neighbor tables, adjust as needed
- Gnutella:
  - “Pong” message truncated flooded through network
  - “Ping” messages returned from each node via reverse path of pong
- Freenet:
  - Not sure about initial discovery and selection

Network (neighbor) maintenance

CS514

- DHT:
  - Nodes ping each other, do repair when neighbor lost (or in background)
- Gnutella:
  - Nodes only need to have enough active neighbors … no structure to maintain
- Freenet:
  - Loose structure…learns of neighbors near itself in the node ID space over time through process of insertion and search

File insertion and storage

CS514

- DHT:
  - File (or file pointer) stored at hashed node
  - may be replicated or cached
- Gnutella:
  - File stored at owner node
- Freenet:
  - File stored in “vicinity” of hashed node
  - may be cached

File search and retrieval

CS514

- DHT:
  - Search for hashed node or replica
  - Opportunistically may find cache
- Gnutella:
  - Truncated flood of search query
- Freenet:
  - “Soft” search based on “steepest-ascent hill-climbing”

File deletion

CS514

- DHT:
  - Send delete message to hashed node
- Gnutella:
  - Delete from local directory
- Freenet:
  - LRU replacement policy
  - No explicit delete
  - (No update either, because don’t know how to flush old copies)

Freenet structure

CS514

- Every node gets a GUID
  - Globally Unique ID
  - Same hash space as files
  - Assigned by some cryptographic distributed random number generation protocol run among nodes discovered with a truncated random walk
    - This is supposed to prevent a newcomer from making up its own GUID, though seems easy to break to me…

Freenet routing trains itself

CS514

- Searches route through network, and answer follows reverse path of search
  - Same search used for both insert and find
- From these answers, every node learns, over time, some keys that other nodes have
- When routing a search, pick node with key (or node?) ID closest to the key being searched
- If search reaches a dead-end or loops, back track and try next best node, etc.

Freenet routing trains itself

CS514

- Idea is this:
  - Files with certain keys will tend to exist on nodes with nearby GUID
  - Nodes will tend to learn of other nodes with nearby keys and GUIDs
  - These two trends reinforce each other
  - Searches get more and more efficient

How Freenet supports anonymity

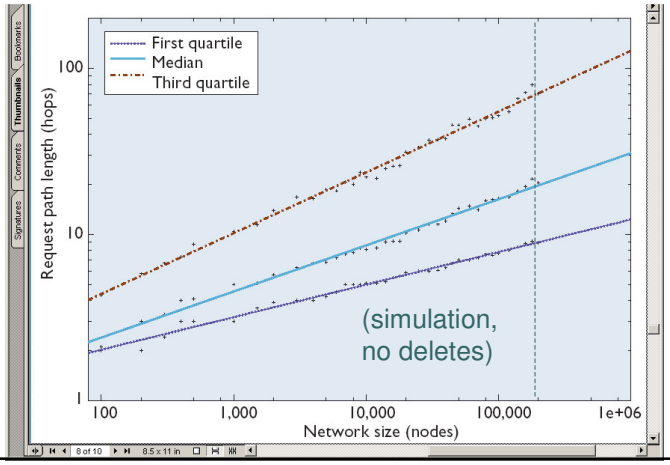
CS514

- All messages are encrypted (hop-by-hop)
- Source of search (find and insert) is hidden
  - Each node remembers previous hop in chain back to source
- Originator of file does not store file in network
- Nodes in search path occasionally (randomly) claim to be the holder of a file
- Search can start with initial random walk to hide “location” of searcher (partly deducible through TTL value)

Freenet scalability: Fits curve of $N^{0.28}$

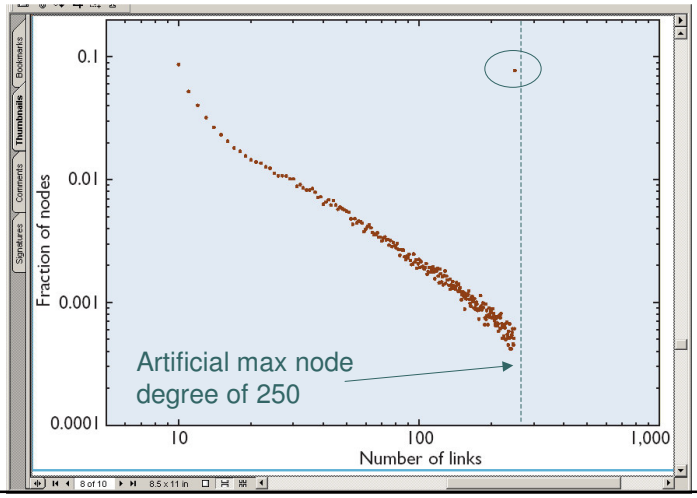
CS514

That's pretty good! But . . .



Distribution of node degree

CS514



Distribution of node degree

CS514

- Reinforcing nature of learning tends to concentrate knowledge on a small percentage of nodes
- Freenet inventors consider this a feature, but . . .
  - "Small world network" (power-law distribution of node degree)
- Terrible load balance!!!
- In essence like centralizing the search!!!

Freenet's anonymity is weak at best

CS514

- Attacker can attach itself all over the network
  - Pretend to have lots of different keys so attach to lots of nodes
  - Now can see a lot of the traffic
- Attacker can push files out of the system by authoring lots of files of similar ID, and by suppressing requests for the file
  - Cache file itself and answer queries from cache
  - Other nodes won't see queries, so won't refresh cache

Freenet's anonymity is weak at best

CS514

- Originator has to repeatedly refresh the file
  - Look for these refreshes, and deduce location of originator
- Of course, can similarly find out who is searching for certain files
- In fact, repressive government's best strategy might be to encourage Freenet, in order to encourage usage by subversives and find them!
  - Weak anonymity worse than none at all!

Fundamental problem with Gnutella

CS514

- Scalability of broadcast search
  - Low bandwidth nodes essentially unusable because of search load
- Solution is to elect "super-nodes" (FastTrack does this) among stable, well-connected participants
- Clients upload index to super-nodes
- Two positive effects:
  - Search sent to nodes better equipped to handle them
  - Searches sent to fewer nodes total

Gnutella load balancing is very bad

CS514

- Suffers from same "small world network" phenomenon as Freenet
  - Saroiu et.al. measurement study
- Broadcast discovery (ping) tends to discover already well-connected nodes
  - Connecting to them makes them more well-connected
- Well-connected nodes will get disproportionate share of searches
- Like Freenet, this makes it easy for an attacker to be everywhere in the network

Gnutella file transfer performed poorly

CS514

- >50% of downloads failed (in 2001)
  - Bad client software
  - Thinly connected clients
- Improvements:
  - Persistent automatic attempts by client (acts like pub/sub in a way)
  - Chunked data
    - Allows interrupted downloads to continue later
    - Allows "striped" parallel download

P2P measurement study

CS514

- Saroiu, Gummadi, Gribble, UWash
- Crawled Gnutella and Napster
- Measured:
  - Bottleneck bandwidths
  - IP-level latencies
  - Connect/disconnect patterns
  - Degree of sharing and downlaod
  - Degree of cooperation
  - Correlations of above

P2P measurement study major results

CS514

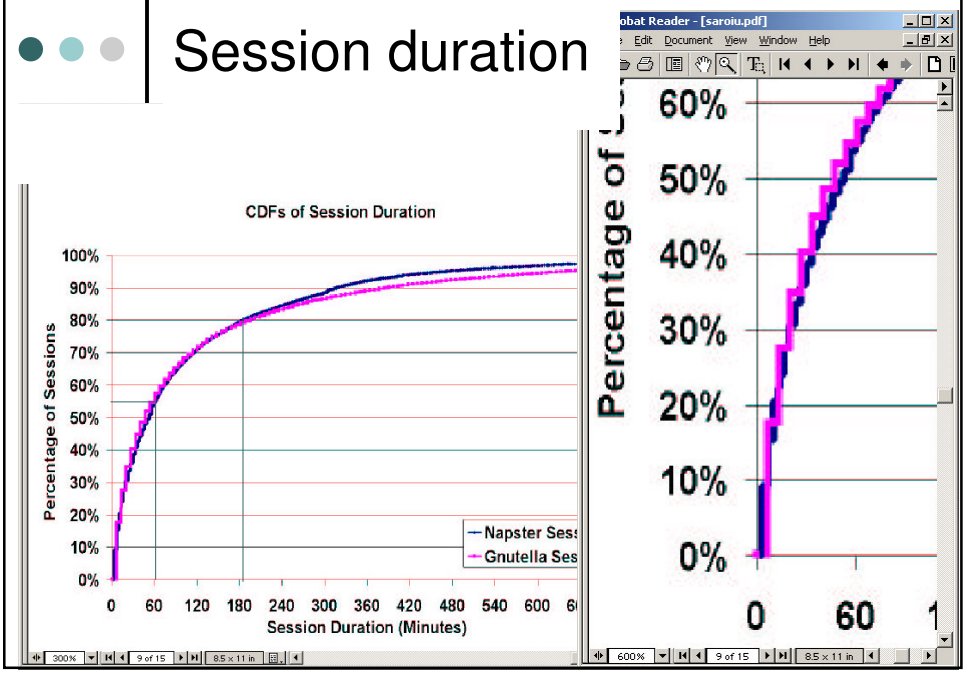
- Extreme heterogeneity
  - 7% of users offer more than 50% of files
  - 20% longest sessions an order of magnitude longer than 20% shortest sessions
  - *Design for extreme heterogeneity*
- Gnutella overlays often disjoint

P2P measurement study major results

CS514

- Latency: Closest 20% 4 times closer than furthest 20%
  - *Have techniques to find low-latency neighbors*
- Peers deliberately misrepresent themselves
  - 30% advertise lower than actual bandwidth, presumably to discourage upload
  - 25% don't advertise bandwidth
  - *Don't trust clients to be honest, measure their capabilities in the system*

Session duration



Status of distributed file sharing

CS514

- FastTrack/Kazaa the big player today
- RIAA is suing them, claiming:
  - They control the network
  - They are aware of and encourage pirating
  - They profit from it
- I believe it is true that they control (or could control) the network
  - They offer a separate for-pay service (though couldn't find this recently)
  - They were able to kick off a whole set of users (Morpheus)