

Evolution of P2P File Sharing Systems

Roman Dunaytsev

The Bonch-Bruевич Saint-Petersburg
State University of Telecommunications

roman.dunaytsev@spbgut.ru

Lecture № 2

Outline

- 1 ARPANET
- 2 Usenet
- 3 World Wide Web
- 4 Napster
- 5 Gnutella
- 6 Freenet
- 7 FastTrack
- 8 eDonkey2000
- 9 BitTorrent
- 10 F2F

Outline

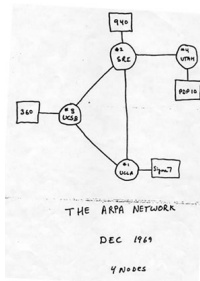
- 1 ARPANET
- 2 Usenet
- 3 World Wide Web
- 4 Napster
- 5 Gnutella
- 6 Freenet
- 7 FastTrack
- 8 eDonkey2000
- 9 BitTorrent
- 10 F2F

- Client/server vs. pure P2P systems

	Client/server	Pure P2P
Participants	Clients and servers	Equal peers
Networking software	Different for clients and servers	Similar for all
Active role (requester)	Client	Any participant
Passive role (provider)	Server	Any participant
Interaction	Clients with servers	Arbitrary
Service/content/resource provider	Servers	Active participants
Data flows (in theory)	Asymmetric	Symmetric

ARPANET (cont'd)

- The goal of the original **ARPANET** was to share computing resources around the USA
- Initially, the ARPANET consisted of 4 hosts which were already independent computing sites with **equal status**:
 - Stanford Research Institute (SRI)
 - University of Utah (UTAH)
 - University of California, Los Angeles (UCLA)
 - University of California, Santa Barbara (UCSB)



ARPANET (cont'd)

- **FTP** and **telnet** – the early Internet applications – were themselves client/server applications
- But the usage patterns as a whole were **symmetric**
- Every host could connect to any other host, and in the early days of minicomputers and mainframes servers usually acted as clients as well
- Thus, **the Internet started as a P2P system**
- In subsequent years, the Internet has become more and more restricted to client/server-type applications

Outline

- 1 ARPANET
- 2 Usenet
- 3 World Wide Web
- 4 Napster
- 5 Gnutella
- 6 Freenet
- 7 FastTrack
- 8 eDonkey2000
- 9 BitTorrent
- 10 F2F

- **Usenet** (USER + NETwork) – a worldwide system for distributing user-submitted messages on various subjects
- In Usenet, messages are posted to **newsgroups**, where any user can access the messages to read and respond to them
- Usenet was developed by Tom Truscott and Jim Ellis at the University of North Carolina and Duke University in 1979
- The initial goal was to create a system, where students could use Unix to write and read messages, and to obtain both technical help and maintain social contacts

Usenet (cont'd)

- **Key difference from Bulletin Board Systems and Web forums** – there is no central server, nor central system owner
- **Key difference from e-mail** – Usenet is oriented to public distribution, rather than private delivery to an individual user
- On August 6, 1991, Tim Berners-Lee posted on Usenet a short summary of the WWW project
 - groups.google.com/group/alt.hypertext/msg/395f282a67a1916c
- On August 25, 1991, Linus Torvalds announced on Usenet that he is doing a free operating system
 - groups.google.com/group/comp.os.minix/msg/b813d52cbc5a044b
- Software piracy has boomed with the rise of the Internet and, particularly, with Usenet
 - E.g., [alt.binaries.warez.ibm-pc](#), [alt.binaries.warez.0-day.games](#), ...
 - 0-day: www.prelist.ws/index.php?section=0DAY (via [Usenet.nl](#))

Usenet (cont'd)

- Originally, Usenet was based on the

Unix-to-Unix Copy Protocol (UUCP)

- UUCP is a protocol suite for point-to-point communication between Unix systems, typically using dial-up modems
- Using UUCP, one Unix machine was able to automatically dial another, exchange files with it over a direct computer-to-computer telephone link, and disconnect

- Nowadays, Usenet relies on the

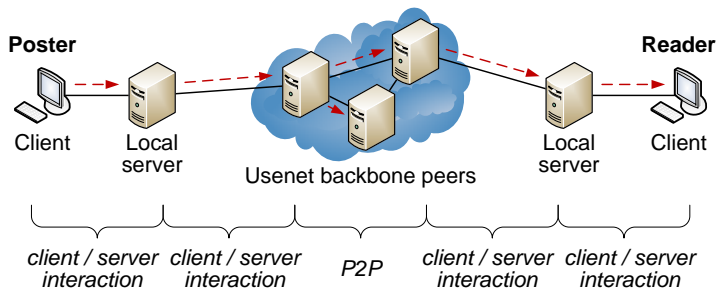
Network News Transport Protocol (NNTP)

- NNTP is an application layer protocol used for reading and posting **Usenet articles** (aka **Netnews**, **news**, or **posts**), as well as transferring them among servers
- NNTP is defined in RFC 3977

Usenet (cont'd)

- Usenet works as follows:

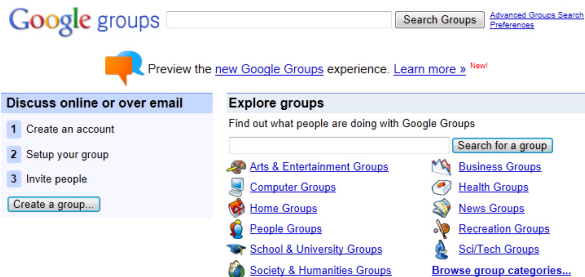
- 1 **Article composition** – a user creates a Usenet article
- 2 **Posting** – the user submits the article to a local Usenet server
- 3 **Propagation** – the article is transmitted from the local Usenet server to other servers, until all Usenet servers that want it have a copy of it
- 4 **Article access** – now other members of the newsgroup can access and read the article



- Originally, Usenet articles were carried over the P2P network known as **UUCPnet**
- Over time, it became clear that some nodes were better connected than others and these nodes went on to form **the Usenet backbone**
- Now Usenet backbone servers collect Usenet articles from local servers and forward them to other backbone servers in a P2P fashion
- But Usenet itself is a typical client/server application
 - Usenet client software (aka **newsreaders**): Usenet Explorer, Xnews, Windows Live Mail, Mozilla Thunderbird, ...
 - Usenet server software: DNews (netwin.com/dnews.htm), ...
- The history of Usenet shows that **client/server and P2P systems are not mutually exclusive**

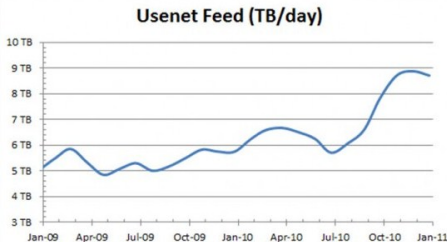
Usenet (cont'd)

- **Google Groups** is a free service from Google
 - <http://groups.google.com>
- Google provides access to 2 kinds of groups:
 - **Regular Usenet groups** – decentralized and not hosted by any single organization
 - **Regular Google groups** – hosted by Google and can be accessed using a Web browser or by subscribing to receive e-mails, but cannot be accessed using Usenet newsreaders



Usenet (cont'd)

- Modern Usenet is very large, with 1000s of servers and terabytes of articles being transferred every day
 - ghacks.net/2011/01/26/usenet-traffic-growth-to-almost-9tb-per-day/
- But much of this traffic increase reflects not an increase in discrete users or newsgroup discussions, but instead the combination of massive automated spamming and an increase in the use of warez newsgroups in which large files are often posted publicly



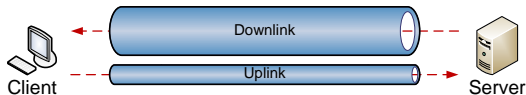
Outline

- 1 ARPANET
- 2 Usenet
- 3 World Wide Web
- 4 Napster
- 5 Gnutella
- 6 Freenet
- 7 FastTrack
- 8 eDonkey2000
- 9 BitTorrent
- 10 F2F

- In the mid-1990s, a new wave of ordinary people, not computer geeks, began to use the Internet as a way to exchange e-mails, access Web pages, etc.
- The WWW is based on the client/server model and uses **HTTP**
- The equal sharing of information over the Internet quickly shifted into **the downstream paradigm**, like TV and newspapers
- The change of the Internet to a mass media system resulted in **asymmetric bandwidth of network paths**

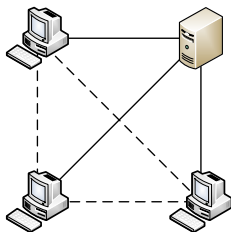


- **Bandwidth asymmetry** – the bandwidth for transmission from the server to the client is much larger than that in the opposite direction
- E.g., ADSL2+ (ITU G.992.5, 2003):
 - Downstream rate = 24 Mbit/s
 - Upstream rate = 1 Mbit/s
- Such asymmetry fits well the downstream paradigm but it places severe limitations on the use of the uplink
- The presence of bidirectional traffic can degrade TCP performance due to the adverse interaction between data packets of upstream flows and feedback messages (TCP ACKs) of downstream data flows
 - For more information, see RFC 3449

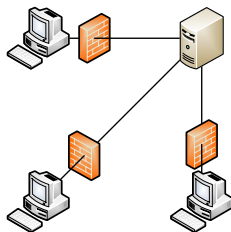


WWW (cont'd)

- In the early days of the Internet, any host that could access the Internet could also be accessed by other hosts
- As the Internet matured there came a need to secure the network and to protect individual hosts from unlimited access
- Then network administrators turned to **firewalls** as a tool to control access to their networks and hosts



Early Internet



These days

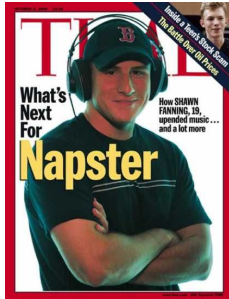
- With the rise of dialup users connecting to the Internet via the PSTN, the old practice of giving every host a fixed IP address became impractical
- **Dynamic IP address assignment** using **DHCP** is now the norm for many hosts on the Internet
 - As a result, many hosts on the Internet are not easily reachable
- Another trend is to use **Network Address Translation (NAT)**
 - NAT allows the use of a pool of private nonroutable IP addresses within a local network
 - As a result, NAT combines the problems of firewalls and dynamic IP addresses: not only the host address is dynamic, it is not even visible!
- **Serious challenges and obstacles to P2P applications are posed by bandwidth asymmetry, firewalls, dynamic IP, and NAT**

Outline

- 1 ARPANET
- 2 Usenet
- 3 World Wide Web
- 4 Napster**
- 5 Gnutella
- 6 Freenet
- 7 FastTrack
- 8 eDonkey2000
- 9 BitTorrent
- 10 F2F

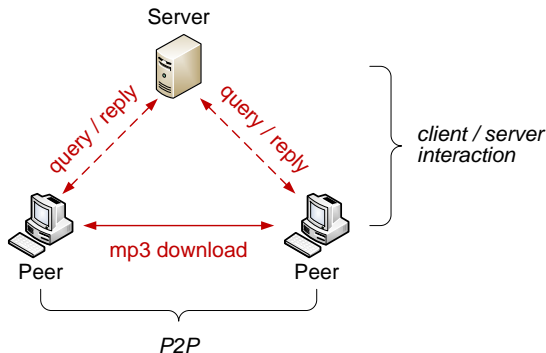
Napster

- The Internet began to shift back to the P2P model with the development, popularity, and attention given to P2P file sharing
- Shawn Fanning, an 18-year-old student, created **Napster** in 1999
- Napster allowed users to easily share mp3 files stored on their PCs
- A year later, in 2000, Napster had 20 million users



Napster (cont'd)

- Napster is an example of a hybrid P2P system:
 - Napster consisted of peers (Napster clients) and a central server
 - The search facility was implemented through the use of a central index, which had a global view of the location of all shared files
 - The actual file transfer occurred between peers



Napster (cont'd)

- Napster worked as follows:
 - ① Users download a software package from Napster and install it
 - ② When a user logs on, the program sends a list of file names that the user is sharing to the central server
 - ③ The central server maintains an index of shared files of users who are currently connected to Napster
 - ④ This index is automatically updated when users log on and log off
 - ⑤ When a user performs a search, it sends a query to the central server
 - ⑥ The query reply contains the name of the files matching the query, plus the IP addresses and port numbers of the peers hosting these files
 - ⑦ The user can then decide from which peer to download the files
 - ⑧ Finally, the file download is performed directly between the peers
- Napster ran on top of TCP for client/server and P2P communications

Napster (cont'd)

Napster v2.0 BETA 7

File Actions Help

Home Chat Library Search Hot List Transfer Discover Help

Artist: revolution Find it! Bitrate: [] []

Title: [] Clear Fields Connection: AT LEAST DSL

Max Results: 100 Advanced << Ping time: [] []

☒ Ping search results

Filename	Filesize	Bitrate	Freq	Length	User	Connection	Ping
White Album\be11b - 12 - revolution 9.mp3	12,053,653	192	44100	8:16	therealh...	DSL	50
White Album\be11b - 08 - revolution 1.mp3	6,139,531	192	44100	4:15	therealh...	DSL	50
Past Masters Volume Two\be15 - 08 - revolution.mp3	4,912,821	192	44100	3:25	therealh...	DSL	50
Beatles - Revolution 1.mp3	3,058,032	96	44100	4:15	rgann1c	T3	90
The Beatles_1 Revolution 9 master version, RS1(mono).mp3	7,489,536	128	44100	7:42	rgann1c	T3	90
nbs\the_revolution-rbd.mp3	2,589,304	160	44100	2:12	Wisema...	DSL	100
Napster\Talkin' Bout A Revolution.mp3	3,196,928	160	44100	2:41	watflakes	T1	110
Music\Revolution 1993.mp3	24,672,131	320	44100	10:08	yoshizumi	DSL	120
Music\Purple Rain - Prince And The Revolution - The Beautiful Ones[1].mp3	3,772,105	96	44100	5:12	mjking46	T1	131
Music\The Beatles - Revolution.mp3	3,282,624	128	44100	3:26	donnie2...	T3	131
Music\Queensryche\Operation Mindcimes=Revolution Calling.mp3	2,473,200	128	44100	2:36	frankief...	DSL	140
Prince and the Revolution - Raspberry Beret.mp3	3,402,470	128	44100	3:33	xmurdoc	DSL	141
Sway and King Tech feat. Rza, Eminem, KRS-One, Ezbit and DJ Revolution - The Anthem.mp3	5,439,488	160	44100	4:31	JtheLover	T1	160
Coalesce 012 Revolution in Just Listening\09-They Always Come In Fall.mp3	2,118,272	128	44100	2:15	youasth...	T1	160
Coalesce 012 Revolution in Just Listening\08-Counting Murders and Drinking Beer (the \$46000 escap...	2,454,384	128	44100	2:35	youasth...	T1	160
Coalesce 012 Revolution in Just Listening\07-Jesus In The Year 2000 Next On the Sht List.mp3	2,839,324	128	44100	2:59	youasth...	T1	160
Coalesce 012 Revolution in Just Listening\06-Where the Hell Is Rick Thorne These Days.mp3	1,852,523	128	44100	1:59	youasth...	T1	160
Coalesce 012 Revolution in Just Listening\05-Sometimes Selling Out Is Waking Up.mp3	3,219,249	128	44100	3:22	youasth...	T1	160
Coalesce 012 Revolution in Just Listening\04-While the Jackass Operation Spins Its Wheels.mp3	2,275,915	128	44100	2:24	youasth...	T1	160
Coalesce 012 Revolution in Just Listening\03-Burn Everything That Bears Our Name.mp3	2,300,575	128	44100	2:26	youasth...	T1	160
Coalesce 012 Revolution in Just Listening\02-cowards.com.mp3	2,330,250	128	44100	2:28	youasth...	T1	160
Coalesce 012 Revolution in Just Listening\001-What Happens On the Road Always Comes Home.mp3	2,965,548	128	44100	3:07	youasth...	T1	160
and everything else\weakeners - ringing of revolution.mp3	3,312,328	128	44100	3:28	youasth...	T1	160
Beatles\Beatles - Revolution.mp3	4,874,385	192	44100	3:24	goenir	T1	180
WorldWideMessage\The Revolution.mp3	3,377,663	128	44100	3:32	sh9453	DSL	190
Bad Religion\Victims of the Revolution.mp3	3,154,337	128	44100	3:18	robertje...	T1	240
(Dance Dance Revolution) Get Up 'n Move.mp3	1,272,740	128	44100	1:23	pnckini	DSL	240
Music\Pantera - Revolution is my name.mp3	5,103,616	128	44100	5:17	P0D48i...	DSL	261
Music\Pantera - Revolution is my name.mp3	5,103,616	128	44100	5:17	P0D48i...	DSL	261
Music\Pantera - Revolution is my name.mp3	5,103,616	128	44100	5:17	P0D48i...	DSL	261
Music\Pantera - Revolution is my name.mp3	5,103,616	128	44100	5:17	P0D48i...	DSL	261

Returned 82 results.

Get Selected Songs Add Selected User to Hot List

Online (dwiner): Sharing 368 files. Currently 871,800 files (3,524 gigabytes) available in 7,119 libraries.

Napster (cont'd)

- Napster also had a method to allow clients behind firewalls to share their files:
 - In order to get a file, a downloader sends a `DOWNLOAD_REQUEST` message to the server
 - The server answers with a `DOWNLOAD_ACK` message containing more information about the file and the uploader (IP address, port number, etc.)
 - If the port number is 0, it means that the uploader is behind a firewall and can only push files outward
 - In this case, the downloader sends an `ALTERNATE_DOWNLOAD_REQUEST` message to the server
 - The server sends an `ALTERNATE_DOWNLOAD_ACK` message to the uploader
 - Once the uploader receives this message from the server, it should establish a TCP connection to the downloader's data port (given in the `ALTERNATE_DOWNLOAD_ACK` message)

Napster (cont'd)

- Napster focused exclusively on mp3-encoded music files
- When Napster was first introduced, there was only one client implementation, which was called Napster
 - The service and software program were initially Windows-only, but in 2000 Black Hole Media wrote a Mac client called Macster
 - Macster was later used by Napster as the official Napster client for Mac
- The Napster protocol is a **closed-source** (proprietary) protocol
 - I.e., no one knows for sure how file searching and transfer is done
- It was only possible to build up a similar application to reveal the Napster protocol through **reverse engineering**
 - Project OpenNap: <http://opennap.sourceforge.net>

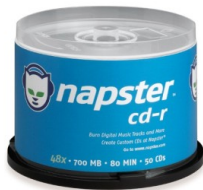
Napster (cont'd)

- The reason for Napster's failure was not technical, but legal:
 - As the Napster network grew to a size of millions of users, the **Recording Industry Association of America (RIAA)** started pressuring Napster to ban the exchange of copyrighted material or to shut down the network
 - Napster deployed various filters to reduce the number of files available for exchange
 - In 2001, Napster was forced to shut down due to the inefficiency of its filtering
- In 2002, Napster's brand and logos were purchased by Roxio, which used them to rebrand the online music service as Napster 2.0
- In 2008, Napster was purchased by Best Buy
- In 2011, Napster merged with Rhapsody

Napster (cont'd)

- **Shortcomings of Napster:**

- **Reliability** – the reliance on a central index server made the system vulnerable to DoS attacks and legal prosecution
- **Security** – the security of the system was weak, since the authenticity of mp3 files was based on file names only and, therefore, could not be verified or guaranteed



Outline

- 1 ARPANET
- 2 Usenet
- 3 World Wide Web
- 4 Napster
- 5 Gnutella**
- 6 Freenet
- 7 FastTrack
- 8 eDonkey2000
- 9 BitTorrent
- 10 F2F

Gnutella

- **Gnutella** was developed by Justin Frankel and Tom Pepper, employees of Nullsoft, in 2000
 - The idea was to design a P2P system without a central point of control
 - This way, the network would be completely decentralized and autonomous, and would be much more robust and harder to shut down
- The authors used the **open-source** model and released the source code and protocol specification under the General Public License (GPL) to allow the system to evolve
- In contrast to Napster, Gnutella has more than 20 different clients from different vendors
- Moreover, Gnutella enables sharing any type of files, not just mp3



Gnutella (cont'd)

The screenshot displays the LimeWire application window. The top menu bar includes File, View, Friends, Tools, and Help. Below the menu is a toolbar with icons for My Files, Friends, and a search bar containing the text 'revolution'. The main window shows search results for 'revolution' (237 results). On the left, there are sections for Categories (Audio, Other, Videos), Artists (Miranda Lambert, Rise Against, The Beatles), Albums (Revolution, Revolutions Per Minute, Purple Rain), and Genres (Country, Rock, Other). A 'Refine results...' search bar is also present. The main content area displays a list of search results, including 'Miranda Lambert - Revolution', 'The Beatles - Revolution', 'The Veronicas - Revolution', 'Lenny Kravitz - It's Time For A...', 'Tracy Chapman - Superchick', 'The Hills Sound - 212 The B...', 'Miranda Lambert - Revolution', 'Miranda Lambert - Revolution', 'Kreator - V...', and 'Starfield - Revolution'. A detailed view of 'The Veronicas - Revolution.mp3' is shown, including its properties (4.83 MB, 3.03, 212 kbps), a 'General' tab, and a 'Details' section with fields for Title, Artist, Album, Genre, Year, and Track. A 'Location' table lists the file's path and filename. A 'Hash' section shows the file's SHA1 hash. On the right, there is a 'Switch to Classic View' button and a list of users who have the file, including 'P2P Users'.

The Veronicas - Revolution.mp3 Properties

Revolution
The Veronicas - Revolution.mp3
4.83 MB (4,858,748 bytes), 3.03, 212 kbps (Excellent Quality)
[Lookup on Bitz](#)
[Copy Link](#)

General

Details

Title: Revolution Artist: The Veronicas

Album: Genre: Year: Track:

Location

Name	Address	Filename
PurpleSquirrel-68-107	88.115.68.107	The Veronicas - Revolution...
ClumayEel-99-174	72.224.99.174	The Veronicas - Revolution...
GlowLobster-153-204	99.237.153.204	The Veronicas - Revolution...
LongRay-49-183	98.246.49.183	The Veronicas - Revolution...
BraveSun-52-120	189.160.52.120	The Veronicas - Revolution...

Hash

urn:sha1:SQEWJ762HM2ZDWWKXQ2FCRH77L2S03R

OK Cancel

Switch to Classic View

19 P2P Users + i

22 P2P Users + i

16 P2P Users + i

6 P2P Users + i

5 P2P Users + i

10 P2P Users + i

4 P2P Users + i

5 P2P Users + i

3 P2P Users + i

2 P2P Users + i

1 P2P User + i

1 P2P User + i

1 P2P User + i

Gnutella (cont'd)

- The idea of Gnutella is similar to the search strategy employed by humans:
 - If you want to get a particular thing, you can ask one of your friends nearby
 - If he does not have the thing, he can ask his friends
 - If everyone is eager to help, this request will be conveyed from one person to another until it reaches someone who has it
 - The information about that person will be routed to you according to the original path
- This search strategy is known as **query flooding**



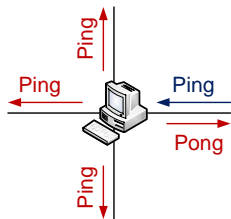
Gnutella (cont'd)

- **Gnutella v0.4** employs the following messages (aka **descriptors**):
- **Ping** – used to actively discover peers (aka **servents**) on the network
 - A peer receiving this descriptor is expected to respond with a Pong
- **Pong** – the response to a Ping descriptor
 - This descriptor includes the IP address and port number of the peer and the number of files and kilobytes of data that the peer is sharing on the network
- **Query** – used to search files on the network
 - A peer receiving a Query descriptor will respond with a QueryHit if a match is found against its local data set
- **QueryHit** – the response to a Query descriptor
 - This descriptor provides the recipient with enough information to retrieve the data matching the corresponding Query

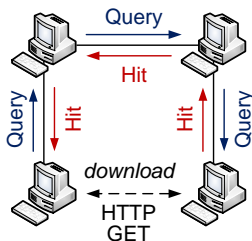
- **Push** – a peer may send a Push descriptor if it receives a QueryHit descriptor from a peer that does not support incoming connections
 - E.g., the peer may be behind a firewall that does not permit incoming connections to its Gnutella port
 - Then the peer attempting the file download may request that the peer sharing the file 'push' the file instead
 - Thus, a peer can request a file push by routing a Push descriptor back to the peer that sent the QueryHit descriptor describing the target file
 - The peer that is the target of the Push descriptor should, upon receipt of the Push descriptor, attempt to establish a new TCP connection to the requesting peer

Gnutella (cont'd)

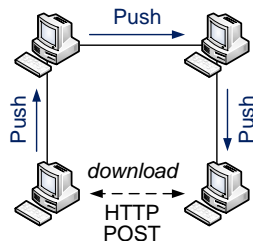
- The Gnutella protocol just provides search functionality
- The file download protocol is HTTP



Ping / Pong routing



Query / QueryHit routing



Push routing

Gnutella (cont'd)

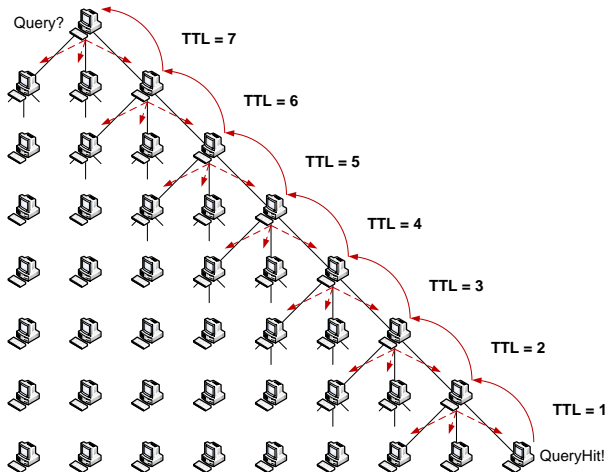
- **Gnutella v0.4** worked as follows:

- ① A given peer, upon launching, must connect to other peers via a list (generated by a variety of means) of available other peers
- ② This new peer pings its known peers and active peers respond with information about the files they are sharing
- ③ A query can then be sent to the other peers and propagated as needed until the query's condition is met and a reply is returned or the descriptor is destroyed after a preset **Time To Live (TTL)**
- ④ Each peer will decrement the TTL before passing it to another peer
- ⑤ When the TTL reaches 0, the descriptor will no longer be forwarded
- ⑥ Once the file is found, a simple HTTP GET request is initiated and the file is on its way

- **Gnutella TTL** specifies the number of times the descriptor will be forwarded by Gnutella peers before it is removed from the network
 - Do not confuse with (although very similar) TTL in IP!

Gnutella (cont'd)

- The original Gnutella used $TTL = 7$



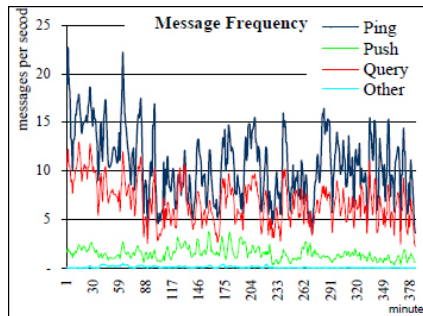
- **Avoiding uncontrolled flooding:**

- All peers must memorize the unique 128-bit descriptor ID every time a descriptor is delivered or originated; if this memorized descriptor is received again, it will not be forwarded
- Pong, QueryHit, and Push descriptors may only be sent along the same path that carried the incoming Ping, Query, and QueryHit descriptors, respectively
- A peer forwards incoming Ping and Query descriptor to all of its directly connected peers, except for the one that sent the incoming Ping or Query
- A peer decrements the TTL field, before it forwards a received descriptor to any directly connected peer; if after decrementing, the TTL field is found to be 0, the descriptor is discarded

- **Performance and scalability issues of Gnutella v0.4:**
- Peers with low-speed connections
 - These peers are usually scattered all over the network and they get overloaded with queries until they become unavailable
 - This causes the network to be highly fragmented
- Signaling traffic
 - Since every peer has to constantly ping other peers in order to obtain their addresses, most of the network bandwidth is used to send Ping/Pong descriptors
 - This prevents delivery of the other more important descriptors

Gnutella (cont'd)

- TR-2001-26 'Peer-to-Peer Architecture Case Study: Gnutella Network' by M. Ripeanu
- Gnutella v0.4 traffic (in bytes), for November 2000:
 - Query/QueryHit descriptors (user-generated traffic) $\approx 36\%$
 - Ping/Pong descriptors (overhead traffic) $\approx 55\%$
 - Push and non-standard descriptors $\approx 9\%$



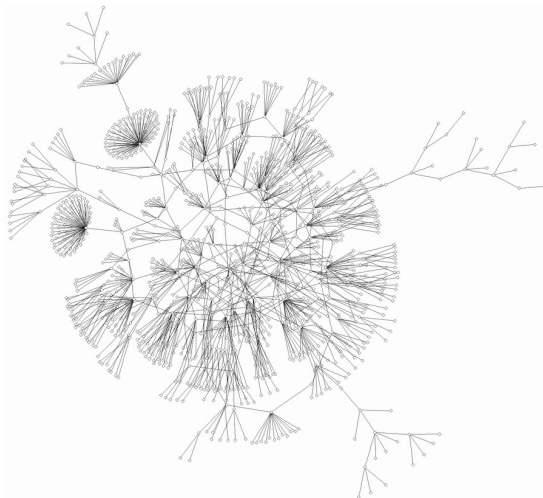
Gnutella (cont'd)

- Some engineering problems have been solved with the arrival of **Gnutella v0.6** in 2001
- To avoid a huge amount of signaling traffic, Gnutella v0.6 introduced a hierarchy by defining **ultrapeers**, which store the information about the content available at the connected peers (aka **leaves**) together with their IP addresses
- TR-2001-26 'Peer-to-Peer Architecture Case Study: Gnutella Network' by M. Ripeanu
- Gnutella v0.6 traffic (in bytes), for June 2001:
 - Query/QueryHit descriptors (user-generated traffic) $\approx 92\%$
 - Ping/Pong descriptors (overhead traffic) $\approx 8\%$ (55% in v0.4)

- **2 modes of Gnutella v0.6 peers :**
 - Leaf
 - Ultrapeer
- A peer can switch from one mode to the other, but it cannot be in both modes at the same time
- A leaf maintains only a single connection to an ultrapeer
- An ultrapeer maintains many (~ 100) leaf connections, as well as a small (~ 10) number of connections to other ultrapeers
- When a leaf connects to an ultrapeer, it sends information about what it is sharing
- Ultrapeers use this information to only forward search queries to leaves that could possibly have hits for them
 - E.g., a leaf that tells its ultrapeer it is sharing nothing will not get any search descriptors at all

Gnutella (cont'd)

- An ultrapeer network
 - <http://home.comcast.net/~gregory.bray/>



- **New features of Gnutella v0.6:**

- http://limewire.negatis.com/index.php?title=How_Gnutella_Works

- **Handshaking**

- A Gnutella connection begins with a handshake
 - It lets 2 programs advertise which advanced features they support
 - It also lets a Gnutella program disconnect from an incompatible peer

- **Pong caching**

- Ultrapeers keep a Pong cache
 - Leaves can ping their ultrapeers to get the Pongs they are currently caching

- **Bye descriptor**

- An optional descriptor used to inform the remote host that the peer is closing the connection

- **Dynamic querying**

Gnutella (cont'd)

- **Gnutella2** (aka **G2** or **Mike's Protocol, MP**) – a P2P protocol developed mainly by Michael Stokes and released in 2002
- G2 is a fork (offshoot) of the Gnutella protocol
- G2 divides nodes into 2 groups:
 - **Leaves** – maintain 1 or 2 connections to hubs, while hubs accept 100s of leaves and many connections to other hubs
 - **Hubs** – index what files their leaves have
- In essence, **Gnutella v0.6** and **G2** are very similar
- However, G2 is not supported by many Gnutella clients, while many G2 clients can also connect to the Gnutella network

Gnutella (cont'd)

- G2 network statistics
 - <http://crawler.trillinux.org>
- Geographical distribution (for April 2019)
 - France: 60 hubs (48.0%); USA: 18 (14.4%); Canada: 6 (4.8%)
- Network size (for April 2019)

Continent	Count
Europe	127
North America	27
Africa	9
Asia	7
Australia	7
Unknown	5
South America	5

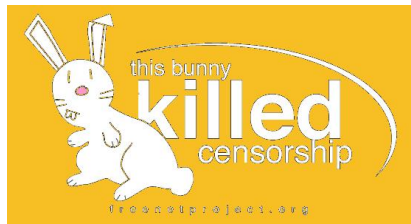
Outline

- 1 ARPANET
- 2 Usenet
- 3 World Wide Web
- 4 Napster
- 5 Gnutella
- 6 Freenet**
- 7 FastTrack
- 8 eDonkey2000
- 9 BitTorrent
- 10 F2F

- The shutdown of the original Napster inspired the creation of pure P2P systems such as Gnutella v0.4, which unlike Napster have **no central control**
- **Freenet** was proposed by Ian Clarke in 1999 as a distributed P2P file sharing and data storage system
 - MSc thesis 'A Distributed Decentralized Information Storage and Retrieval System', University of Edinburgh, 1999
- The Freenet protocol is open-source and has been under continuous development since 2000
 - <http://freenetproject.org>



- **Motivation** – Governments around the world undertake efforts to force ISPs to block access to content deemed unsuitable or subversive, or to make them liable for such material hosted on their servers
- **Objective** – Privacy for information producers/consumers/holders and resistance to information censorship
- **Key features** – Internet-wide information **storage** and **anonymous** information publication and retrieval



- Freenet works as follows:
 - ① Each Freenet participant runs a node (peer) that provides the network with some storage space
 - ② A file is inserted into the network with an associated **key**
 - ③ After insertion is finished, the publisher is free to shut down his node, since the file is stored in the network
 - ④ During a file's lifetime, it can migrate to or be replicated on other nodes
 - ⑤ To retrieve a file, a user sends out a request message containing the key
 - ⑥ When the request finds a node containing a copy of the file, the file is returned through the search path

- **Keys:**

- Each file that exists on Freenet has a **key** associated with it
 - Freenet keys are somewhat analogous to URLs on the WWW, except unlike URLs, they do not point to the physical location of the data
 - Freenet uses **semantic-free** references to make the keys independent of the file content; this is achieved by using **hash-based** keys
 - Keys are created using SHA (Secure Hash Algorithm)
- In Freenet, shared files are encrypted and the encryption keys are separated from the actual data

- Freenet basic messages:
 - **Insert** – allows a node to insert new data into the network; the message includes the data file and the key
 - **Request** – a request for a certain file; the request contains the key of the file
 - **Reply** – sent by the node that has the requested file; the actual file is included in the reply message
 - **Failed** – denotes a failure to insert or locate a file; the message contains the location of the node where the failure occurred and the reason
- For more information, see FCPv2 (Freenet Client Protocol version 2):
 - <https://wiki.freenetproject.org/FCPv2>

• Bootstrapping in Freenet:

- The process starts when a new node is announced in the network
- A node new in Freenet needs to obtain an **identifier (ID)** for itself
- This ID is a number that is derived through the announcement process
- The announcement message contains the public key and an address of some existing node
- This announcement message is propagated by Freenet nodes
- Every message in Freenet has a **Time To Live (TTL)** value, which determines when the message propagation is stopped
- When the message propagation stops, the nodes in the chain collectively assign a new ID for the new node and some subspace of the keyspace

- **Inserting data:**

- For every piece of data, there is an associated key
- Each node has a routing table, which with respect to the key, gives an ordering of neighbors after their closeness as the destination of the query
- The route is then constructed by going from node to node and selecting the most suitable neighbor
- When this is not possible, the request backtracks and the route is restarted from the previous node
- The routing process is terminated either due to the request achieving its purpose or the TTL field is found to be 0

- **A request is routed to the node that has the numerically closest ID value to the key**

- **Retrieving data:**

- To search for a file in the network, matching a key, one establishes a route for the key
- At each step, the node checks its cache to see if a file associated with that key is present
- If such a file is found, the search terminates and the file is returned to the previous node in the route, which relays it back towards the node which initiated the request
- Otherwise, the request is forwarded to the node with the closest matching ID
- This routing process is repeated until either the file is found or the TTL field becomes 0

- **Storing data:**

- During the relaying data at both insert and reply steps, nodes store the data in their cache
 - Each node has 2 separate caches
 - One is a **short-term cache** where all data that the node transfers are stored temporarily until they are pushed out by other data
 - The other is a **long-term cache** for storing only inserted data that match the node ID
- The network can be viewed as a large grid of caching proxy hosts, each relaying and caching for one another

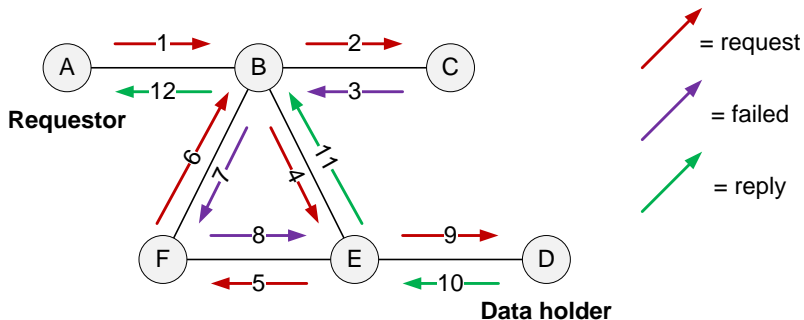
Freenet (cont'd)

- The Freenet routing algorithm is **steepest ascent hill climbing with backtracking**
 - Limited by a preset TTL value
- **Simple hill climbing** – the first **closer** node is chosen
- **Steepest ascent hill climbing** – all available nodes are compared and **the closest** to the solution is chosen
- **Backtracking** – when the search reaches a dead end ('plateau'), it simply returns back to the previous node and tries alternative paths



- **Key-based routing:**

- The request moves through the network from node to node, backing out of a dead end (Step 3) and a loop (Step 7) before locating the desired file



Freenet (cont'd)

Statistics for Node id|7733353574668240287

[Browsing](#) [Filesharing](#) [Friends](#) [Discussion](#) [Status](#) [Configuration](#) [Key Utils](#)

Node Version Information Freenet 0.7.5 Build #1277 build01277 Freenet-ext Build #26 r23771	Current Activity Inserts: 6 CHK handlers, 0 SSK handlers (0/0 local) Requests: 29 CHK handlers, 35 SSK handlers (9/6 local) Transferring Requests: sending 14, receiving 13, receiving turtles 1
JVM Info Used Java memory: 54.8 MiB Allocated Java memory: 82.1 MiB Maximum Java memory: 185 MiB Running threads: 146/500 Available CPUs: 1 Java Version: 1.6.0_21 JVM Vendor: Sun Microsystems Inc. JVM Name: Java HotSpot(TM) Client VM JVM Version: 17.0-b17 OS Name: Windows XP OS Version: 5.1 OS Architecture: x86	Peer statistics CONNECTED: 6 BUSY: 8 TOO OLD: 2 Max peers: 14 Max strangers: 14
Statistics Gathering Generate a Thread Dump Get latest node's logfile Download your translations file	Bandwidth Input Rate: 17.6 KiB/s (of 64.0 KiB/s) Output Rate: 9.63 KiB/s (of 16.0 KiB/s) Session Total Input: 104 MiB (7.53 KiB/s average) Session Total Output: 64.6 MiB (4.65 KiB/s average) Payload Output: 40.3 MiB (2.89 KiB/sec)(62%) Global Total Input: 104 MiB Global Total Output: 64.6 MiB

Messages: 6 Minor English Switch to advanced mode Security levels: LOW NORMAL LOW 14 / 14

Outline

- 1 ARPANET
- 2 Usenet
- 3 World Wide Web
- 4 Napster
- 5 Gnutella
- 6 Freenet
- 7 FastTrack**
- 8 eDonkey2000
- 9 BitTorrent
- 10 F2F

- **FastTrack** – a P2P protocol developed by Niklas Zennström and Janus Friis in 2001
- In 2003, FastTrack became very popular, having at one point more users than that Napster on its peak
- Similar to Gnutella v0.6, highly connected peers with sufficient resources perform the search on behalf of more resource constraint regular peers
- FastTrack is proprietary software
- In 2003, an open-source alternative, called **OpenFT**, was developed by the giFT project through reverse engineering
 - <http://gift.sourceforge.net>

FastTrack (cont'd)

- In FastTrack, peers with the fastest Internet connections and the most powerful computers become **supernodes**
- A supernode maintains information about shared resource as well as connections with other supernodes
- When a peer performs search, it first searches for the closest supernode, which returns immediate results (if any) or refers the search to other supernodes (if needed)
- However, supernodes themselves are just ordinary nodes that can join or leave the network as they please
- Therefore, **the network is dynamic and always changing**

FastTrack (cont'd)

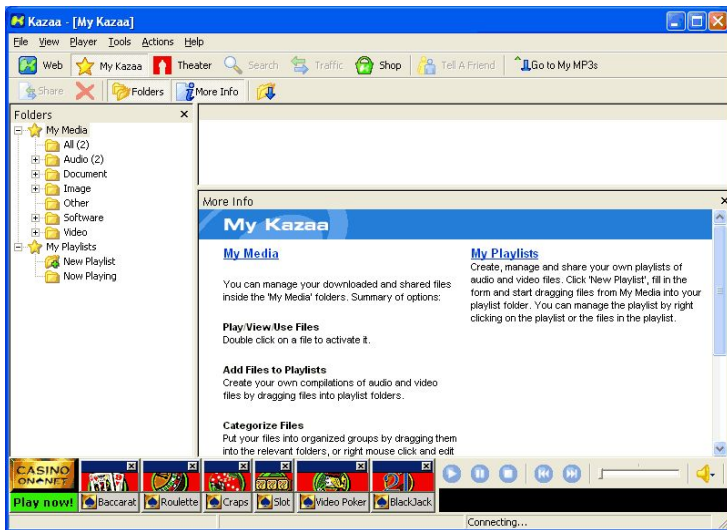
- In order to ensure the constant availability of the network, there exists a need for dedicated peers that will monitor and keep track of the network
- Such peers are called **bootstrapping nodes**
- **Bootstrapping in FastTrack:**
 - ① When a peer joins the network, it will first contact the bootstrapping node
 - ② The bootstrapping node will then determine if that particular peer qualifies to be a supernode
 - ③ If it does, then it will be provided with some IP addresses of other supernodes
 - ④ Otherwise, the bootstrapping node will respond by providing the IP address of one of the supernodes

FastTrack (cont'd)

- The P2P model heavily depends on the concept of **cooperation**
- 'Free Riding on Gnutella' by E. Adar, B. Huberman:
 - ~ 50% of all files for sharing are stored on only 1% of peers
 - ~ 70% of Gnutella users share no files (they are referred to as **freeloaders** or **free riders**)
- Some FastTrack clients, such as Kazaa, used a method known as **reputation**, where the reputation of a certain user is reflected by its **participation level**
 - If you have a high participation level, you will get priority over people with a lower participation level when you download files
 - You can increase your participation level by uploading files
 - When you download files, your participation level will go down
 - When your participation level is low, downloading will be slower

FastTrack (cont'd)

- Kazaa could be downloaded free of charge, but was full of **adware**



FastTrack (cont'd)

- Sharing copyrighted content using the original FastTrack protocol:

- **Kazaa**

- Owned: Consumer Empowerment BV, Sharman Networks, Ltd.
- Lifetime: 2001 - 2005 (reason - lawsuits from copyright owners)
- Today: legal music download service, www.kazaa.com

- **Grokster**

- Owned: Grokster, Ltd.
- Lifetime: 2001 - 2005 (reason - lawsuits from copyright owners)
- Today: closed

- **iMesh**

- Owned: iMesh, Inc.
- Lifetime: 2001 - 2005 (reason - lawsuits from copyright owners)
- Today: legal music download service, www.imesh.com

- **Morpheus**

- Owned: StreamCast Networks, Inc.
- Lifetime: 2001 - 2002 (reason - failed to pay FastTrack license fees)
- Today: closed

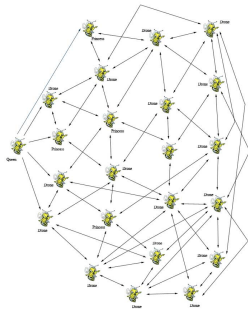
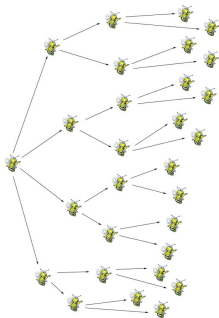
Outline

- 1 ARPANET
- 2 Usenet
- 3 World Wide Web
- 4 Napster
- 5 Gnutella
- 6 Freenet
- 7 FastTrack
- 8 eDonkey2000**
- 9 BitTorrent
- 10 F2F

- How to increase the throughput of P2P networks?
 - With the size of the files being transferred, it might take a long time before a new peer would be able to share a new file it is acquiring
- Use **swarming** !
 - If a file can be downloaded in small pieces (aka **chunks**) and then immediately be shared by new peers, this bidirectional transfer reduces the load on peers that have the complete file, thereby distributing not only the file but its access as well
- iMesh was the first company to introduce swarming
- Nowadays, swarming is a must for P2P file sharing (eD2k, BitTorrent)



- Without swarming, the upload bandwidth goes wasted until that peer has the whole file
- With swarming, all peers work bidirectionally with each other
- But if the originator leaves the swarm too early, no one will be able to use the file



- **eDonkey** (aka **eDonkey2000** or **eD2k**) was developed by Jed McCaleb (MetaMachine Inc.) and was first released in 2000
- Currently, eD2k is not supported by any company and works by being supported by its users alone
- eD2k is a hybrid P2P file sharing network with client applications running on PCs that are connected to a distributed network of **dedicated servers**
 - The servers are slightly similar to the Gnutella v0.6 ultrapeers and FastTrack supernodes
 - However, they do not share any files, only manage the information distribution
- In 2004, eD2k overtook FastTrack to become the most widely used P2P file sharing system on the Internet

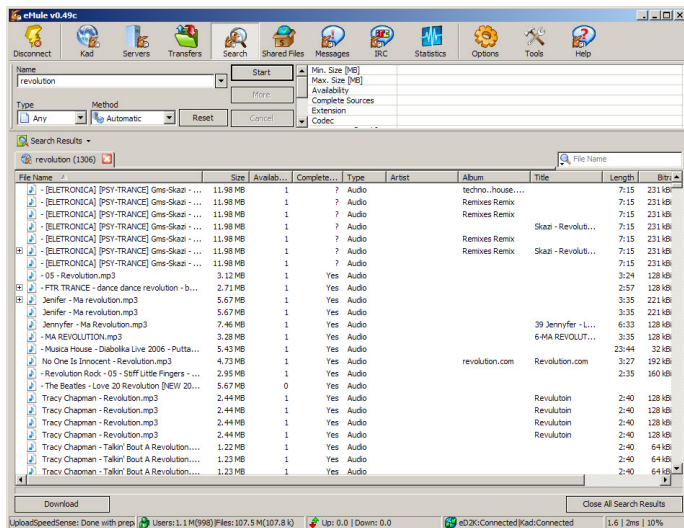
- eD2k works as follows:

- ➊ A client connects to a server and stays connected
- ➋ The client sends information about itself (username, IP address, port number)
- ➌ The client sends a list of the files it offers to the server
- ➍ This information is added to the server's database
- ➎ A list of other known servers is transferred from the server to the client
- ➏ Now the client can use the eD2k network to search and download files, while itself shares its files by making them available for download by other users
- ➐ Between clients there is no communication except file downloads
- ➑ Servers communicate with each other to keep their databases up-to-date

- In eD2k, files are divided in chunks of 9500 KB plus a remainder chunk, and a separate 128-bit MD4 checksum is computed for each
 - A transmission error corrupts only a chunk instead of the whole file
 - Valid downloaded chunks are available for sharing before the rest of the file is downloaded, speeding up the distribution of large files throughout the network
- eD2k uses MD4 (Message Digest version 4) hashes to uniquely identify a file independent of its filename
 - Hashing allows to treat files with identical content but different names as same, and files with different contents but the same name as different

eDonkey2000 (cont'd)

- eD2k allows to search for content within the application



Outline

- 1 ARPANET
- 2 Usenet
- 3 World Wide Web
- 4 Napster
- 5 Gnutella
- 6 Freenet
- 7 FastTrack
- 8 eDonkey2000
- 9 BitTorrent**
- 10 F2F

- **BitTorrent** was developed by Bram Cohen in 2001
 - Nowadays, it is maintained by Cohen's company, BitTorrent Inc.
- Unlike other P2P networks, BitTorrent does not offer the capability to search for content within the application
- Instead, users must have prior knowledge of tracker sites and know where to look for the **torrents** they want to download
- **BitTorrent tracker** – a server which assists in the communication between peers using the BitTorrent protocol

BitTorrent (cont'd)

- BitTorrent works as follows:

- 1 A user contacts a tracker to find and download a torrent file for the data file he wants
- 2 Using this torrent file, the BitTorrent client software communicates with the tracker to find other peers running BitTorrent that have the complete file (aka **seeds**) and those with a portion of the file (i.e., peers that are in the process of downloading the file)
- 3 The tracker identifies the **swarm**, which is a set of connected peers that have the complete file or a portion of it and are in the process of sending/receiving it
- 4 The tracker helps the client software trade pieces of the file with other peers in the swarm
- 5 BitTorrent uses the SHA-1 hash function to determine which pieces of the file are good and which are bad
- 6 If the user continues to run the BitTorrent client software after the download is complete, this peer becomes a seed

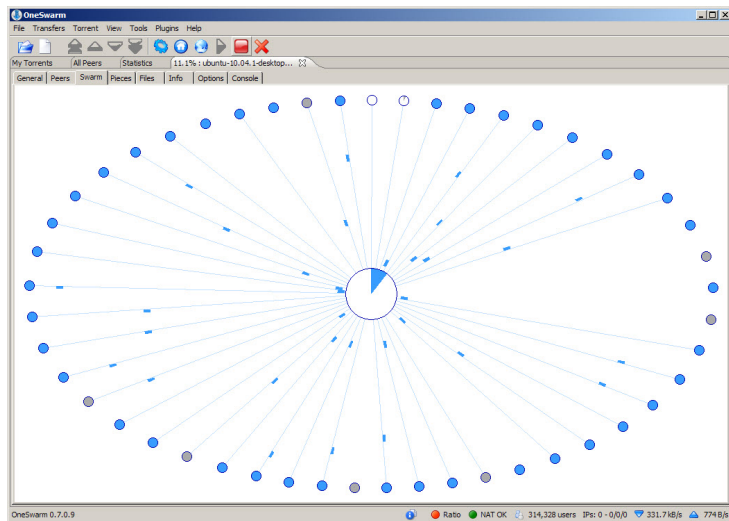
Outline

- 1 ARPANET
- 2 Usenet
- 3 World Wide Web
- 4 Napster
- 5 Gnutella
- 6 Freenet
- 7 FastTrack
- 8 eDonkey2000
- 9 BitTorrent
- 10 F2F**

- **Friend-to-friend (F2F) file sharing** – a new P2P paradigm that provides users with explicit control over their privacy by letting them determine how files are shared
 - Aka **private P2P networks**
- Instead of sharing data indiscriminately, data shared with F2F can be made public, can be shared with friends, shared with some friends but not others, etc.
 - E.g., using IP addresses or digital signatures for authentication
- A node in a F2F network requires more effort to set up and maintain, because all peers must be connected manually
- F2F software:
 - OneSwarm
 - Freenet
 - LimeWire
 - etc.

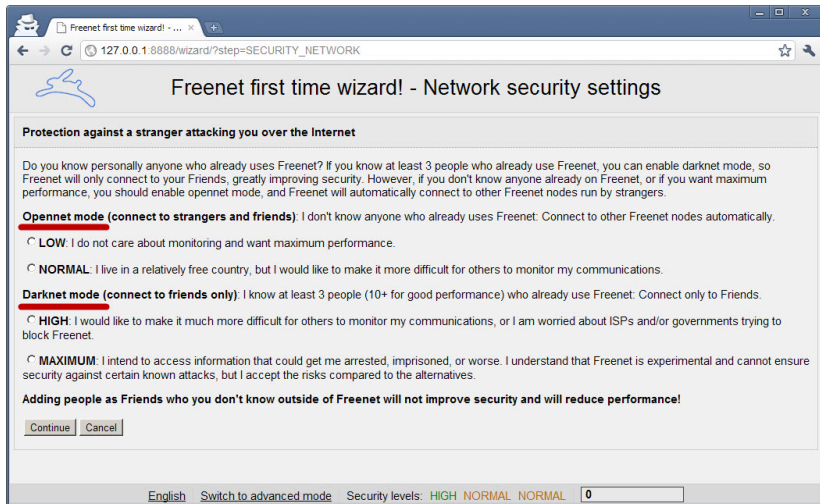
Friend-to-Friend (cont'd)

- OneSwarm's interface is browser-based



Friend-to-Friend (cont'd)

- F2F in Freenet (aka Darknet)



The screenshot shows a web browser window titled "Freenet first time wizard! - Network security settings". The address bar shows the URL "127.0.0.1:8888/wizard/?step=SECURITY_NETWORK". The page content includes a section titled "Protection against a stranger attacking you over the Internet". It explains that if the user knows at least 3 people who already use Freenet, they can enable darknet mode. It then presents three options: "Opennet mode (connect to strangers and friends)", "Darknet mode (connect to friends only)", and "MAXIMUM". Each option has a radio button and a description. At the bottom, there are "Continue" and "Cancel" buttons, and a footer with "English", "Switch to advanced mode", "Security levels: HIGH NORMAL NORMAL", and a numeric input field set to "0".

Freenet first time wizard! - Network security settings

Protection against a stranger attacking you over the Internet

Do you know personally anyone who already uses Freenet? If you know at least 3 people who already use Freenet, you can enable darknet mode, so Freenet will only connect to your Friends, greatly improving security. However, if you don't know anyone already on Freenet, or if you want maximum performance, you should enable opennet mode, and Freenet will automatically connect to other Freenet nodes run by strangers.

Opennet mode (connect to strangers and friends): I don't know anyone who already uses Freenet. Connect to other Freenet nodes automatically.

☐ LOW: I do not care about monitoring and want maximum performance.

☐ NORMAL: I live in a relatively free country, but I would like to make it more difficult for others to monitor my communications.

Darknet mode (connect to friends only): I know at least 3 people (10+ for good performance) who already use Freenet. Connect only to Friends.

☐ HIGH: I would like to make it much more difficult for others to monitor my communications, or I am worried about ISPs and/or governments trying to block Freenet.

☐ MAXIMUM: I intend to access information that could get me arrested, imprisoned, or worse. I understand that Freenet is experimental and cannot ensure security against certain known attacks, but I accept the risks compared to the alternatives.

Adding people as Friends who you don't know outside of Freenet will not improve security and will reduce performance!

English Switch to advanced mode Security levels: HIGH NORMAL NORMAL 0

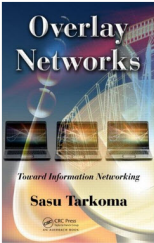
Friend-to-Friend (cont'd)

File Edit View History Bookmarks Tools Help

Library Genesis: Sasu Tarkoma x +

libgen.io/book/index.php?md5=46607F01EBF75EBF300089A3BACF81C

RU FORUM DOWNLOAD UPLOAD LAST OTHERS TOPICS DONATE



Overlay Networks
Toward Information Networking
Sasu Tarkoma

Title: Overlay Networks: Toward Information Networking.
Author(s): Sasu Tarkoma
Series:
Publisher: Auerbach Publications
Year: 2010
Language: English
ISBN: 143981371X, 9781439813713
Time added: 2011-04-20 14:05:05
Library:
Size: 4 MB (4213124 bytes)
Worse versions:
Desr. old vers.: 2016-04-03 09:51:08
Commentary:
Topic: Computers\\Networking
Identifiers: ISSN: UDC: LBC: LCC: TK5105.5.T366 2010 DDC: 004.6/52 DOI: OpenLibraryID: OL23954066M GoogleID: ASIN:

Book attributes: DPI: OCR: Bookmarked: Scanned: Orientation: Paginated: Color: Clean:

Mirrors: Libgen.io Libgen.pw Bookfi.net Bookzz.org One-file Torrent Sasu Tarkon Copy filename Gnutella EdzK DC++ Torrent per 1000 books

Hashes:
A1CH:COLXHOZAB3s1WZ2Q6RZ8Y0LAQ7gHYKFG
CBC3smAqF4k4
eDns8ep12AgBB6gBF63sA1v7E6D7k445DzDg6D
MD5:46607F01EBF75EBF300089A3BACF81C
SHA1:c4A0C513ZNYGVYEMCWB3MFUW4g3Y03F
TTH:YHtDQwADOEgWZ24CP3B3g3y4d07m6HCUYSC0A
SHA256:335F8D51993CF0413F5AAM034F8aD83aFD4
En37B9x7379aD0g993B4q9970BF3s

A recent Cisco traffic forecast indicates that annual global IP traffic will reach two-thirds of a zettabyte by 2013. With their ability to solve problems in massive information distribution and processing, while keeping scaling costs low, overlay systems represent a rapidly growing area of R&D with important implications for