Lecture 14
Plan: FTP and wired LAN

FTP protocol
ftp (File Transfer Protocol) is one of the oldest application-level protocols on the Internet (first appear in 1971 at MIT).

Detailed specifications for various network protocols are published in documents called RFCs (Request for Comment). The RFC for ftp is 69 pages long. It contains many details which will not be necessary for us, but I'm putting a link to the document here: rfc959.txt Principle of internet design (RFC 1958) as after class reading material

The purpose of ftp is to facilitate the transfer of entire files between the file system on a local machine and the file system on a remote machine.
Characteristics:
1. Arbitrary file content
2. Bidirectional transfer
3. Support for authentication
4. Accommodates hetergeniety

An ftp client is an interactive program controlled by a human user. The user enters commands to the client program, which carries them out by communicating with the ftp server on the remote machine. Users never interact with the FTP interface.

rfc 959 defines 33 commands which can be sent from a client to a server. Each one is designated by a 3- or 4-letter code. Commands are transmitted over a tcp connection as a string of characters, the command code followed (optionally) by arguments. The commands for file transfer are

RETR filename -- retrieve the specified file from the server and store it on the client
STOR filename -- send the specified file from the client and store it on the server

Some other commands of interest:

USER username -- log in to the server as a particular user
PASS password -- supply a password as part of the login sequence
LIST -- get a listing of the contents of the current directory (on the server) from the server
CWD dirname -- change the working directory on the server to the specified directory
PWD -- get the name of the server's working directory from the server
QUIT -- close the connection to the server

Basics of the protocol

More detail can be found by studying the rfc.

1. A client begins a session by opening a tcp connection with a server on port 21. This is called the *control connection*, used for sending commands to the server. It is also used by the server to send reply messages to the client. (The actual data transfers are made through a separate connection, as we shall see shortly.)
2. Commands are sent as strings of ASCII text. The first word of the string indicates the name of the command. Additional words (if there are any) are arguments. The command names are 3- and 4- letter codes and are typically *not* the same as the command codes used in the user-client interaction. For example, most client programs use the commands *get* and *put* to retrieve and send files instead of the actual protocol codes RETR and STOR.
3. The server responds to every command by sending a reply message on the control connection. Reply messages are also ASCII text strings. Every reply message begins with a three digit numeric code. (A list of all of the codes and their meanings is found in the rfc.) If the reply is a single line, the code is followed by a space and then (optionally) some additional text. If the reply will carry over to more lines, the code is followed by a hyphen ("-"). In that case, the end of the reply is indicated by a line containing the same three-digit code followed by a space.
4. Some commands, like PWD, can be handled using only the control connection. If a data transfer is required (e.g., RETR, STOR), a second tcp connection called the data connection is made. The data transfer is carried out using the data connection.
5. The data connection is set up in one of two ways, *active mode* or *passive mode*, depending whether the connection is initiated by the client (passive mode) or the server (active mode).
6. In active mode, the client sends a PORT command to the server, indicating that it will accept an incoming tcp connection from the server. (The port number on which the client is listening is given as an argument to the PORT command.) The server makes the connection and then transfers the data through the data connection.
7. In passive mode, the client sends a PASV command to the server. This command instructs the server to create a listening socket. The server sends a reply to the client. The client then makes a tcp connection to the server's listening socket. The data transfer is performed on this connection.
8. In both active and passive modes, a new data connection is set up for each data transfer command.

Common commands

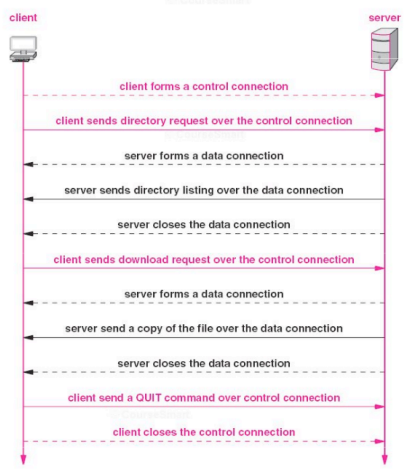
- ABOR - **abor**t a file transfer
- CWD - **c**hange **w**orking **d**irectory
- DELE - **dele**te a remote file
- LIST - **list** remote files
- MDTM - return the **mo**d**i**fication **ti**m**e** of a file
- MKD - **m**a**k**e a remote **d**irectory
- NLST - **n**ame **l**i**st** of remote directory
- PASS - send **pass**word
- PASV - enter **pas**si**v**e mode
- PORT - open a data **port**
- PWD - **p**rint **w**orking **d**irectory

- QUIT - terminate the connection
- RETR - **retr**ieve a remote file
- RMD - **rm**ove a remote **d**irectory
- RNFR - **rn**ame **fr**om
- RNTO - **rn**ame **to**
- SITE - **site**-specific commands
- SIZE - return the **size** of a file
- STOR - **sto**re a file on the remote host
- TYPE - set transfer **type**
- USER - send **user**name

Less common commands

- ACCT* - send **acc**oun**t** information
- APPE - **appe**nd to a remote file
- CDUP - CWD to the parent of the current directory
- HELP - return **help** on using the server
- MODE - set transfer **mode**
- NOOP - do nothing
- REIN* - **rein**itialize the connection
- STAT - return server **stat**us
- STOU - **sto**re a file **u**niquely
- STRU - set file transfer **stru**cture
- SYST - return **syst**em type

Example transfer procedure



Algorithm to transmit a data item over an FTP data connection

Given an FTP control connection

1. Client sends request for a specific file over control connection

2. server receives the request

3. client allocates a local protocol port, call it X

4. client binds to port X and prepares to accept a connection

5. client sends “PORT X” to server over control connection

6. Server receives PORT command and request for data item.

7. Client waits for a data connection at port X and accept

8. Server creates a data connection to port X on client's computer

9. Server sends the requested file over the data connection

10. Server closes the data connection.

Some problems with ftp:

1. Security. Usernames and passwords are sent without encryption on the command connection. Data is transferred without encryption on the data connection.
2. Firewalls. Because of security concerns, many firewalls will only permit incoming tcp connections on ports that it knows about. This may cause a problem with active mode ftp. See rfc 1579 for further discussion.

Demonstrate FTP communications using wireshark with port 433 as a filter.

2. Wired LANS and Token Ring
The real world realization of the algorithms we learned.

Wired LAN
IEEE 802 standard is for local area networks
802.3: Ethernet
802.4: Token Bus
802.5: Token Ring
802.11: Wireless (wi-fi)
802.15: Bluetooth

The standards are concerned with

- addressing
- Frame structure
- Transmission and coding
- Cabling/ topology (physical aspects)

Ethernet
Developed in 1970’s in Xerox PARK
Uses Bus topology

- Hardware Technology

Name	Cable	Max. seg.	Nodes/seg.	Advantages
10Base5	Thick coax	500 m	100	Original cable; now obsolete
10Base2	Thin coax	185 m	30	No hub needed
10Base-T	Twisted pair	100 m	1024	Cheapest system
10Base-F	Fiber optics	2000 m	1024	Best between buildings

10 Base 5
* thick coax (the original -- 1978)
* the "10" indicates 10 Mbps transmission rate
* "5" indicates 500 m(eters) the maximum distance between nodes without repeaters.
* Base = Baseband signaling (only 3 states on the cable: 0,1 and idle → digital signal)

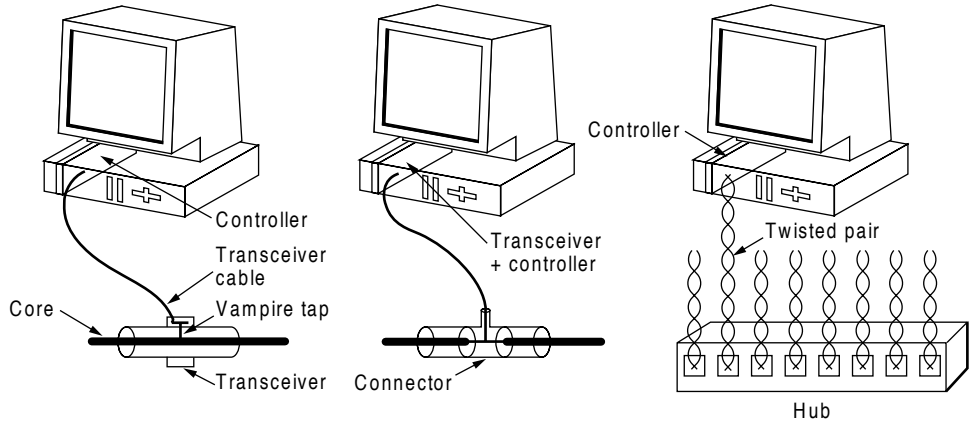
10 Base 2
* thin coax -- use T connector at each adapter to tap into signal, terminator at each end.
* maximum distance between nodes is 200 m
* can use up to 4 repeaters, extending length to 985 meters

10 Base T
* twisted pair -- use hub, individual connections (a star topology at the physical level)
* limited to 100 m with twisted pair

100 Base T
* Fast Ethernet
* uses higher quality shielded twisted pair for higher data rate

10 Base F, 100 Base F
* fiber

Gigabit Ethernet and 10 Gigabit Ethernet



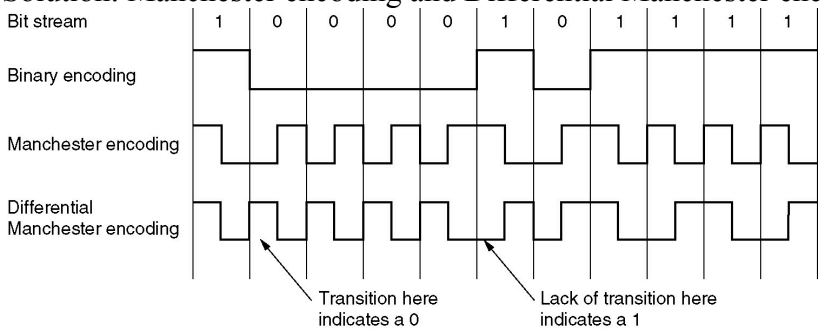
from left to right we have 10base5, 10 base2 and 10baseT

- Encoding

Problem: if we use +5V as 1 and 0V as 0, then it is hard to tell idle from 0.

Q:/ How to solve it?
A:/ What if we use +1V as 1 and -1V as 0
Q:/ What is wrong with +1V and -1V?
A:/ Needs synchronization to identify boundaries.

Idea: need to identify the start, end, and the middle of each bit
Solution: Manchester encoding and Differential Manchester encoding



1 –high low
0 – low high

For differential, if there is a change then it means 0. if there isn't a change then it is 1.

- Addressing

Q:/ Why?
A:/ In Ethernet, signals are broadcasting. So each machine needs an identifier

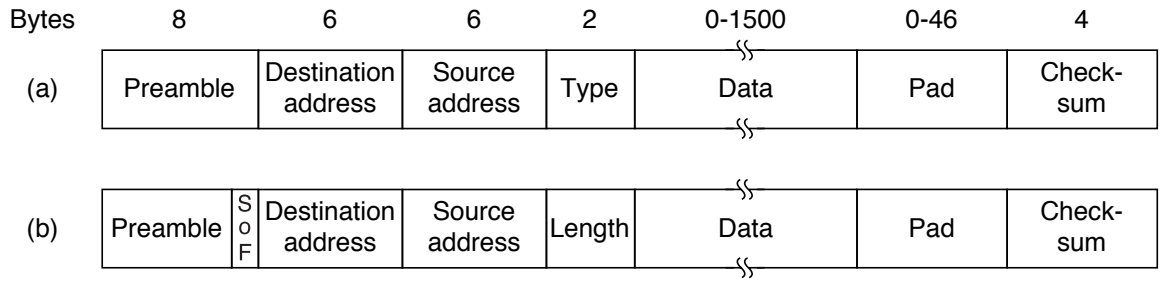
Unique 48-bit address is burned in the Ethernet adapter's ROM. → Won't change

Normally, it is written in 6 octets, each of which is two hex. → MAC address (same format in wireless)
My apple wireless card: 00:23:12:00:85:e4
my apple Ethernet card: 00:22:41:32:bc:a5

Address space is managed by IEEE.
The first 24bits are a block number. → a manufacturer can buy a block for a nominal fee
my laptop's Ethernet addresses can be expressed as apple_32:32:bc:a5 (ethernet)

An address of 111…1 (all 1's) indicates a broadcast message.
ARP (address resolution protocol) from the network layer maps Ethernet address with ip address

- Frame Structure

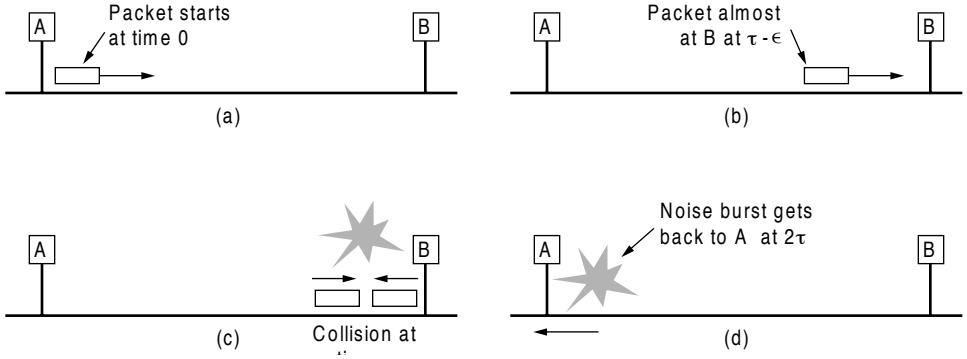


a is DIX Ethernet frame and b is IEEE 802.3 frame

Preamble: 8 bytes of 10101010.
In IEEE 802.3, the last byte is 10101011 to mark the start of the frame

Type: Tells what network protocol is used (IP, IPX). → support a host with more than one network protocols

Data and Pad: 46-1500 bytes
Q:/ Why 46?
A:/ If a frame is too short, then it is possible that sender may finish sending the frame before it detects a collision. Thus it won't resend.



The requirement of 64 comes from the following specification. The propogation delay in Ethernet network (max length is 2500 meters with four repeaters). The round trip time is determined to be 50 μs. With a speed of 10Mbps, each bit takes about 1/10M=100ns. So to make sure that the sender won't finish sending the frame before the jam signal is received, we need at least 500 bits per frame. To be on the safe side, 64 bytes are used as the minimum size.

If the data length is less than 46, then pad will be used to make sure that the length from destination address to checksum is at least 64 bytes.

Checksum: CRC 4bytes.

No ACK or NAK → bad frames are discarded.