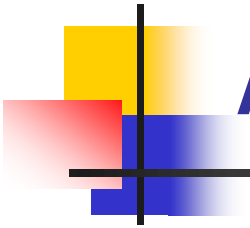
TFTP and FTP Basics

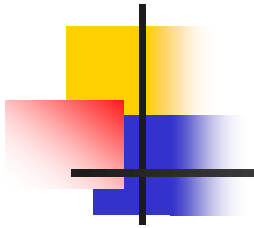
BUPT/QMUL

2019-04-22

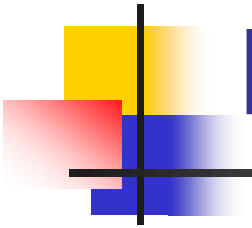


Agenda

- File transfer and access
- TFTP (Trivial File Transfer Protocol)
- FTP (File Transfer Protocol)
- NFS (Network File System)

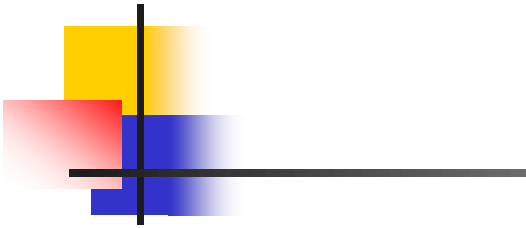


File Transfer And Access

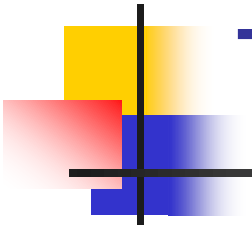


File Transfer And Access

- Providing computers with the ability to access files on remote machines
- Different goals
  - To lower overall cost
  - To archive data
  - To share data across multiple programs, multiple users, or multiple sites
- Two forms
  - On-line access: NFS
  - Whole-file copying: FTP, TFTP

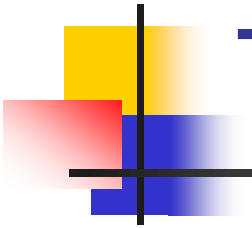


TFTP
(Trivial File Transfer Protocol)



TFTP

- TFTP Features
- TFTP Protocols
- TFTP Operations
- TFTP Example



TFTP Features

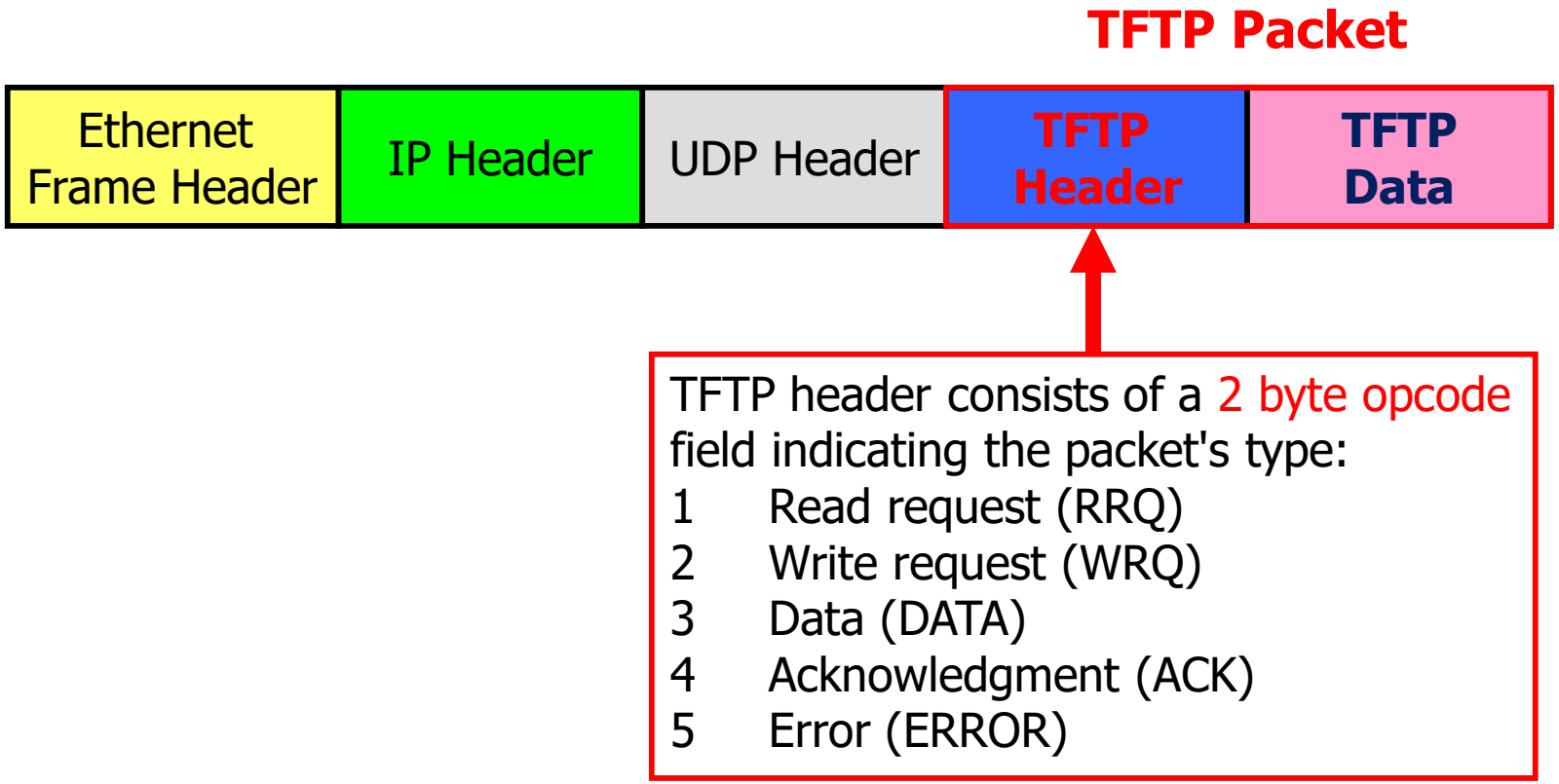
- Read and write files from / to remote computers
- Minimal overhead (no security)
- Designed for UDP, although could be used with other transport protocols
- Easy to implement
- Small – possible to include in firmware
- Often uses to bootstrap workstations and network devices
- No Access Control / No Directory Retrieval

TFTP Protocols – RFCs

- **RFC 1350** – The TFTP Protocol (Revision 2)
- **RFC 906** – Bootstrap loading using TFTP
- **RFC 1785** – TFTP Option Negotiation Analysis
- **RFC 1986** – Experiments with a Simple File Transfer Protocol for Radio Links using Enhanced Trivial File Transfer Protocol (ETFTP)
- **RFC 2090** – TFTP Multicast Option
- **RFC 2347** – TFTP Option Extension
- **RFC 2348** – TFTP Blocksize Option
- **RFC 2349** – TFTP Timeout Interval and Transfer Size Options
- **RFC 3617** – Uniform Resource Identifier (URI) Scheme and Applicability Statement for the Trivial File Transfer Protocol (TFTP)

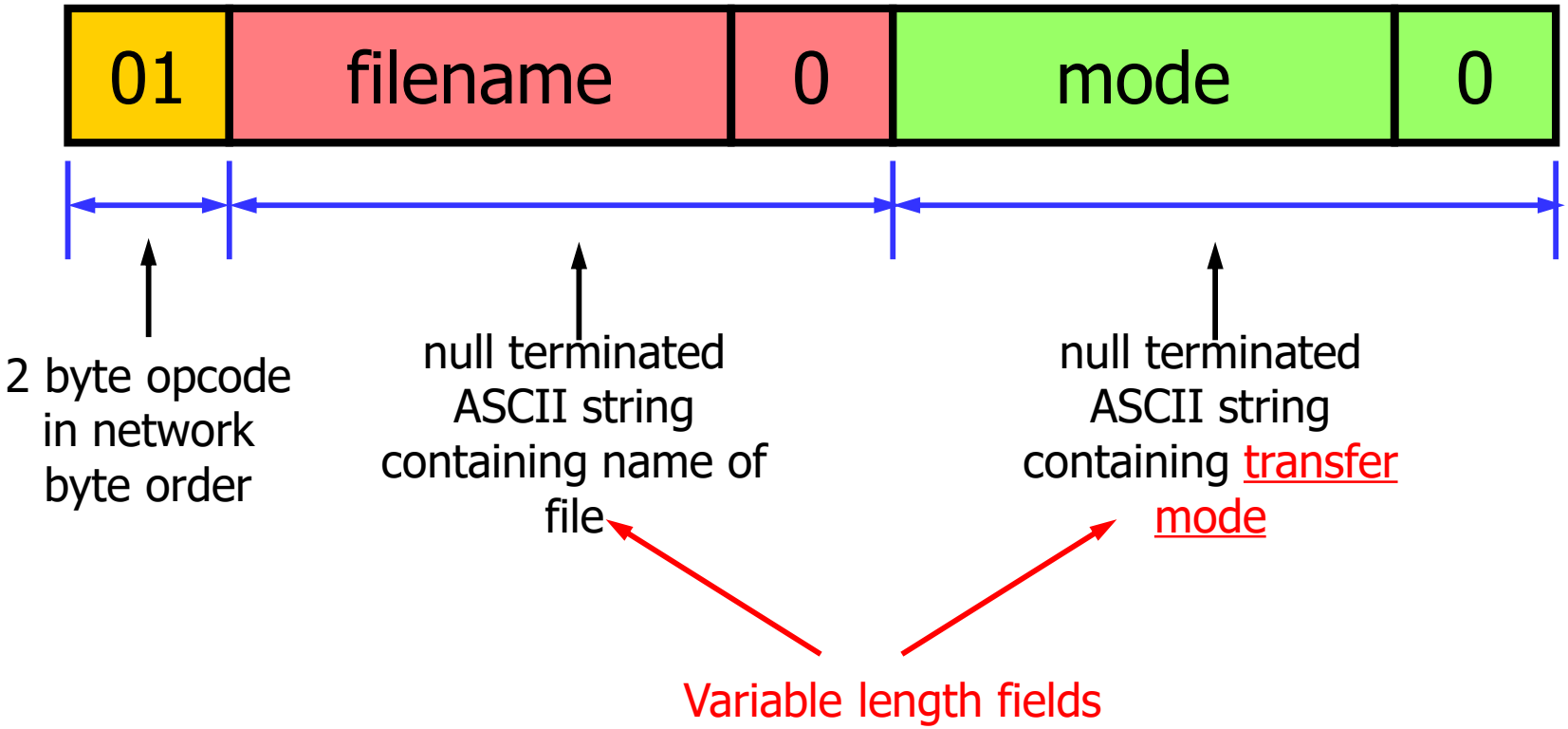
TFTP Protocols – Packet Format (1)

- Order of headers



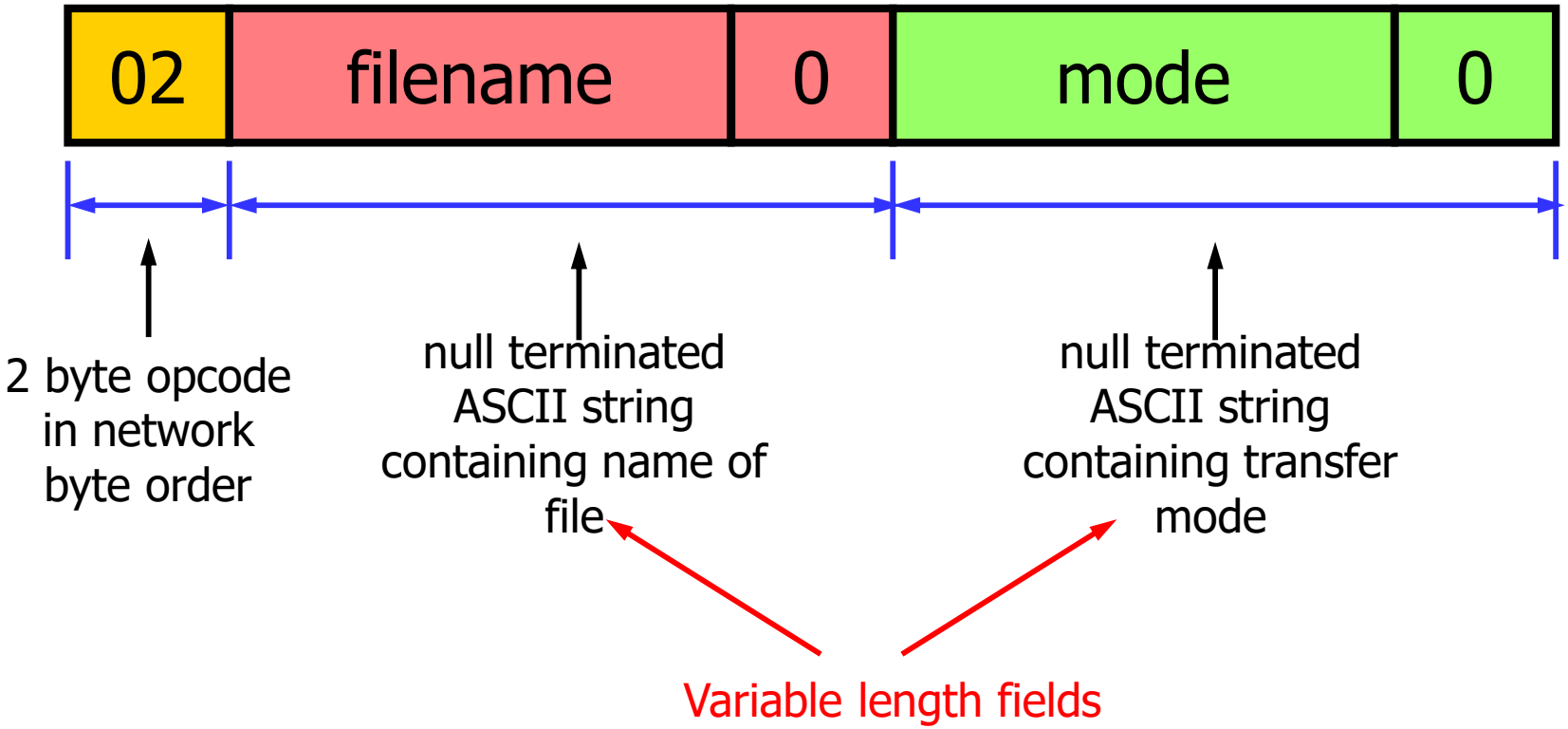
TFTP Protocols – Packet Format (2)

- Read request (RRQ)



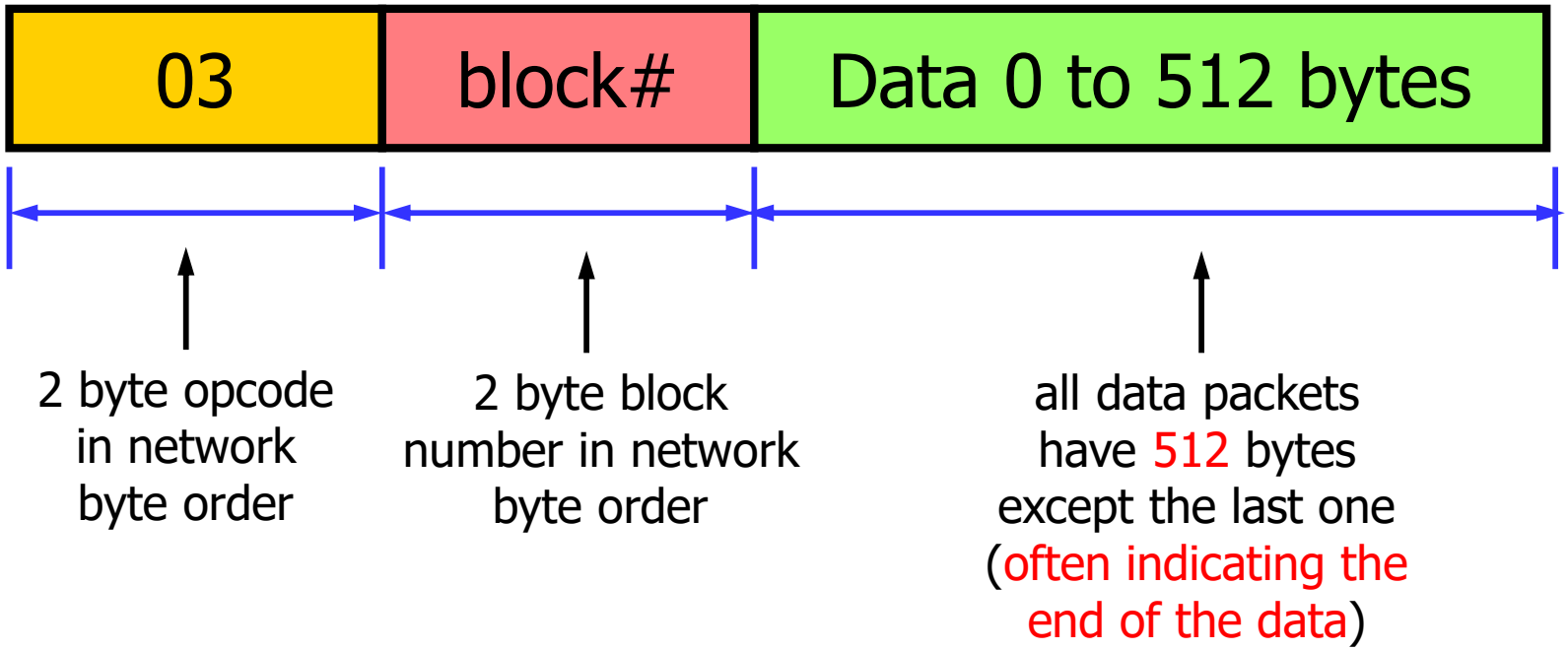
TFTP Protocols – Packet Format (3)

- Write request (WRQ)



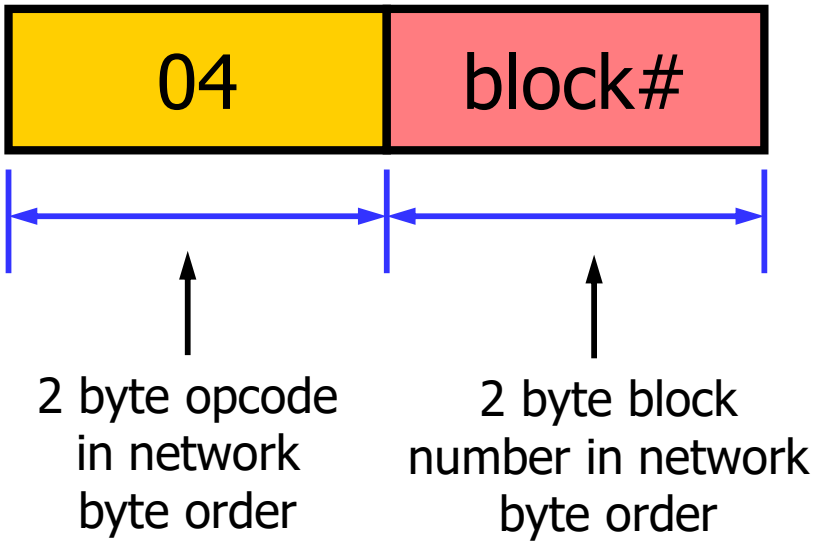
TFTP Protocols – Packet Format (4)

- Data (DATA)



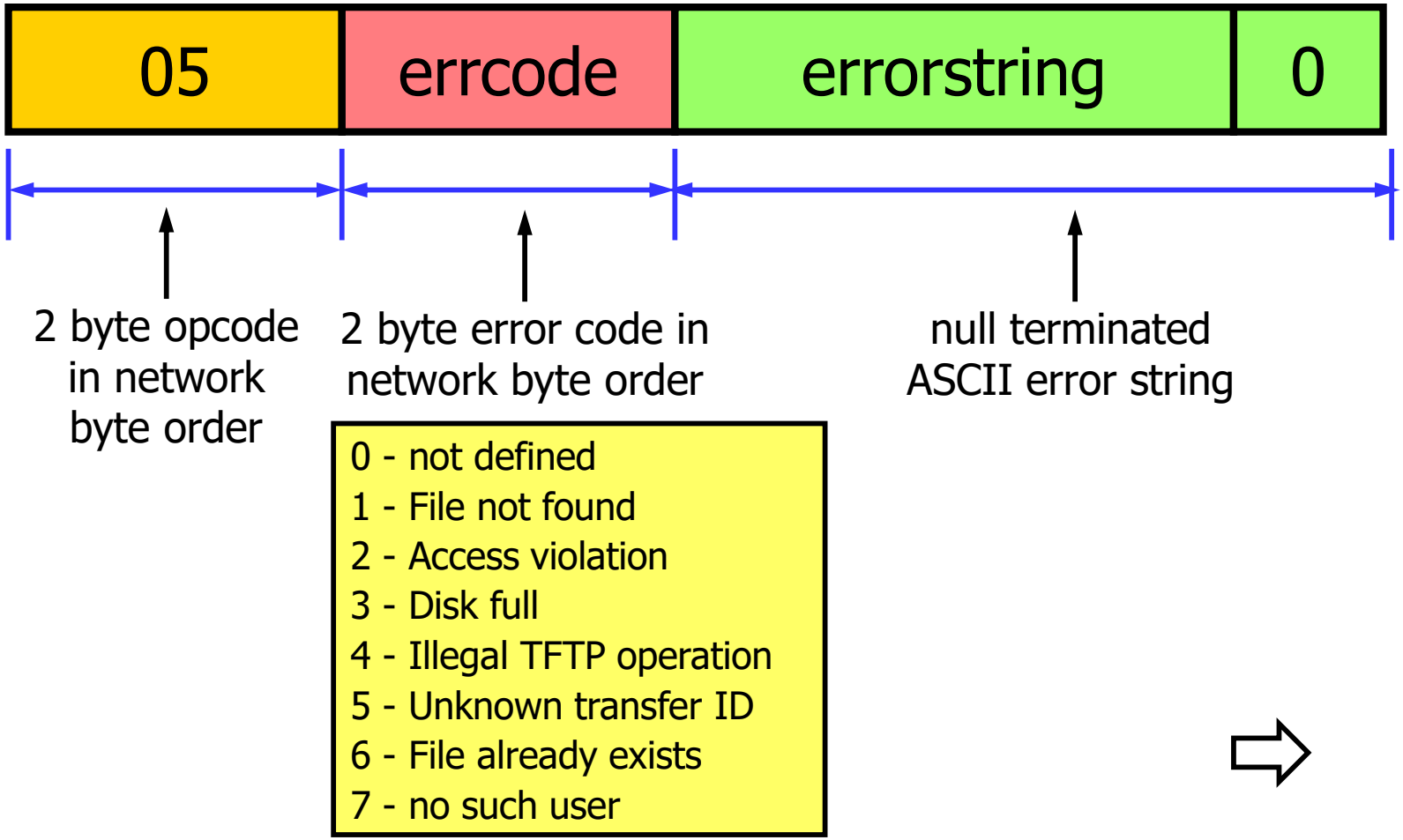
TFTP Protocols – Packet Format (5)

- Acknowledgment (ACK)



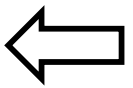
TFTP Protocols – Packet Format (6)

- Error (ERROR)



TFTP Operations – Transfer Mode

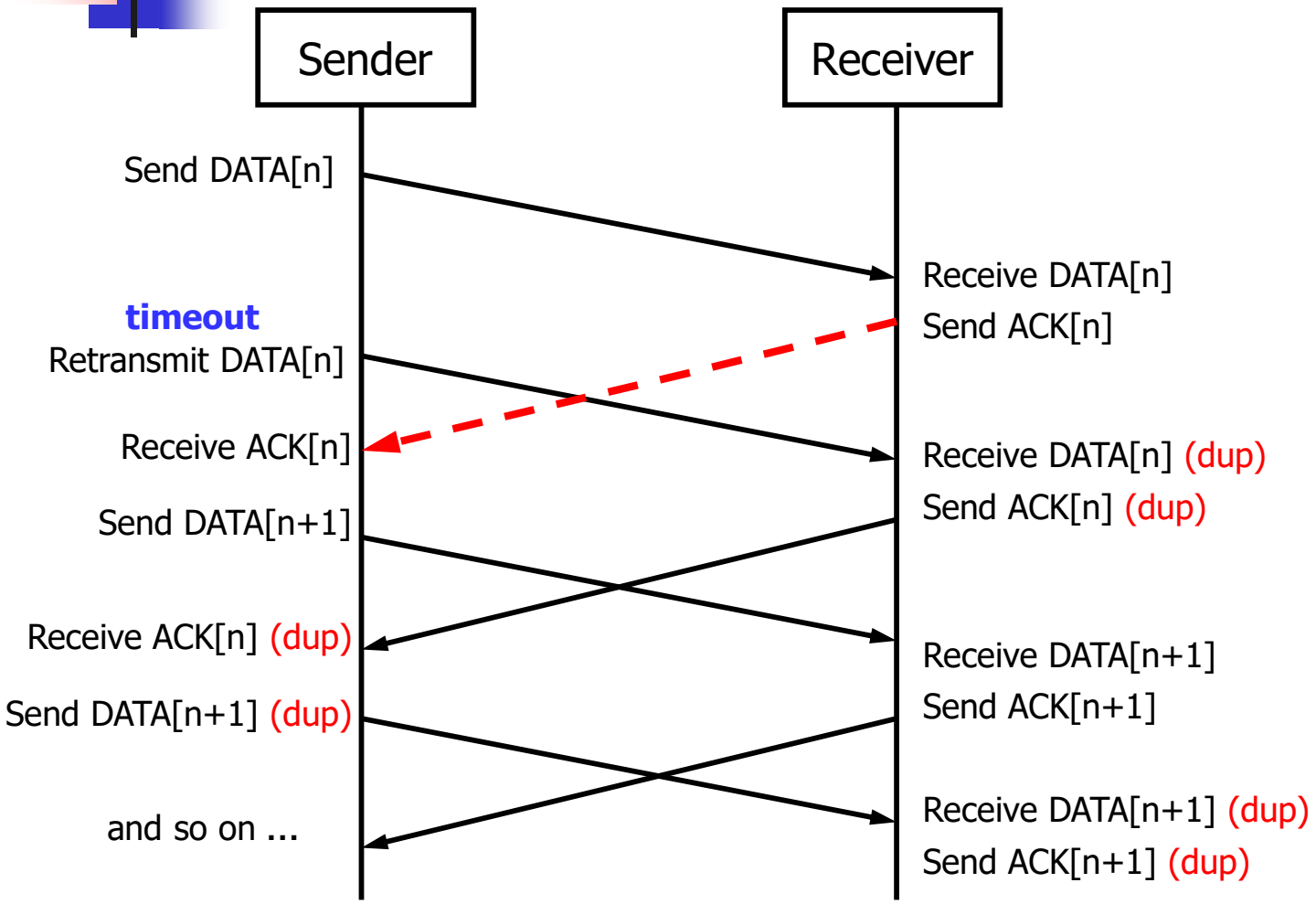
- Netascii - for transferring text files
  - all lines end with \r\n
  - provides standard format for transferring text files
  - both ends responsible for converting to/from netascii format
- Octet - for transferring binary files
  - no translation done



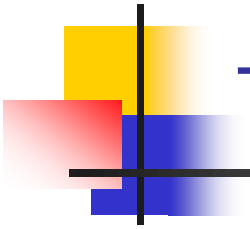
TFTP Operations – Retransmission

- Symmetric
  - Both machines involved in a transfer are considered senders and receivers.
    - One sends data and receives acknowledgments
    - The other receives data and sends acknowledgments
  - Each side implement the timeout and retransmission
    - If a data packet gets lost in the network, the data sender times out and retransmits the last data packet
    - If an acknowledgment is lost, the acknowledgment sender retransmits the last acknowledgment
- The sender has to keep just one packet on hand for retransmission, since the stop and wait mechanism guarantees that all older packets have been received
- Duplicate data packets must be recognized (ignored) and acknowledgment retransmitted
- This original protocol suffers from the *sorcerer's apprentice syndrome (SAS)*

TFTP Operations – Sorcerer's Apprentice Syndrome



- Arising when an acknowledgment for a data packet is delayed, but not lost
- Leading to excessive retransmissions
- Once started, the cycle continues indefinitely with each data packet being transmitted exactly twice

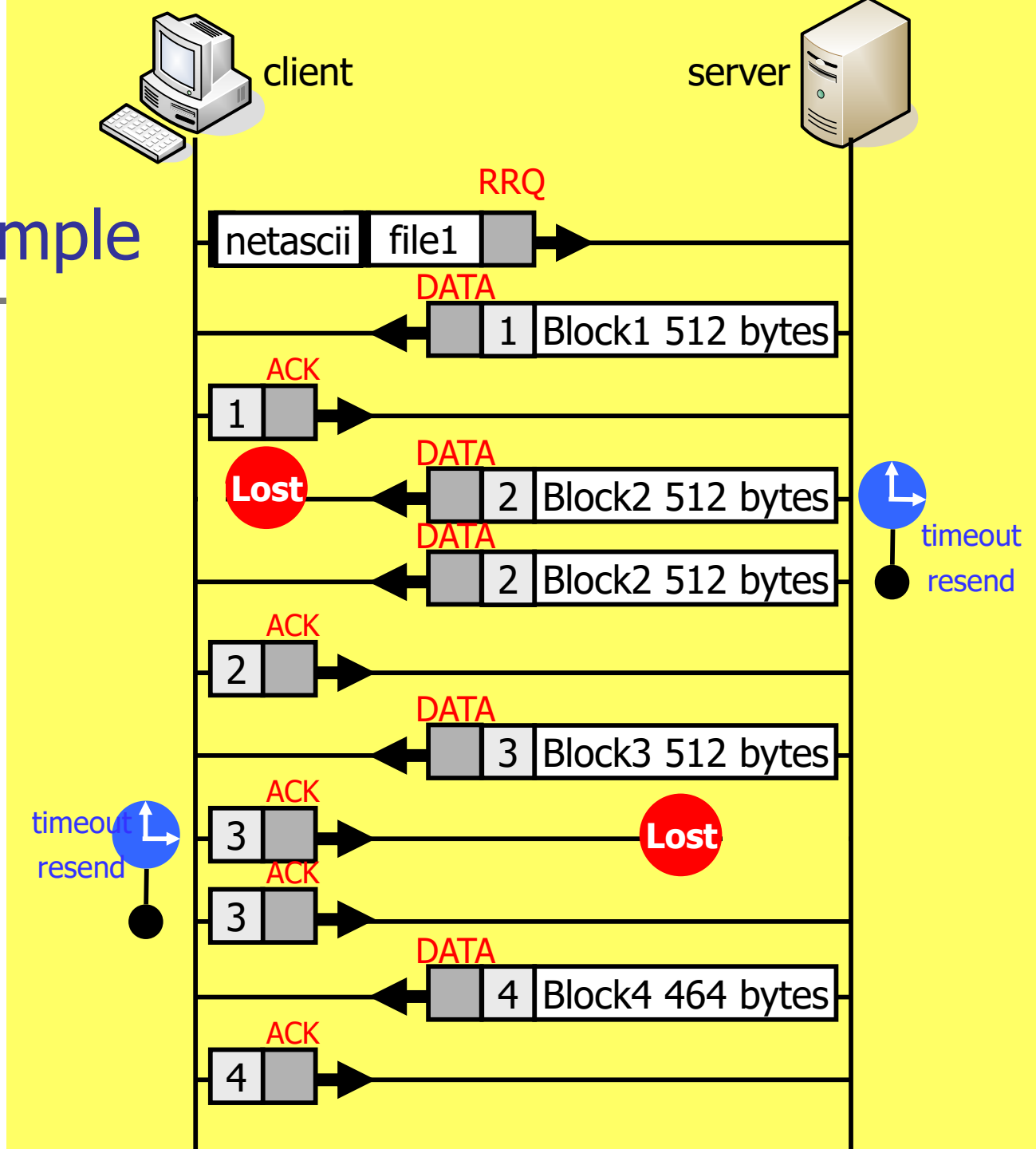


TFTP Operations – How to fix SAS

- Principle: *break the retransmission loop*
  - Sender should not resend a data packet in response to a duplicate ACK
  - If sender receives ACK[n] - don't send DATA[n+1] if the ACK was a duplicate
- See details in *RFC 1123*

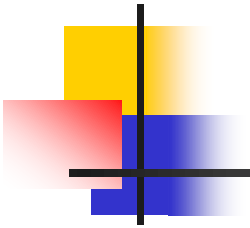
TFTP Example

- Beginning with RRQ/WRQ
- Every good block is Acknowledged
- Both sides use timers
- Simple stop and wait mechanism provides reliable delivery and flow control
- A block of less than 512 bytes signals the end of the file

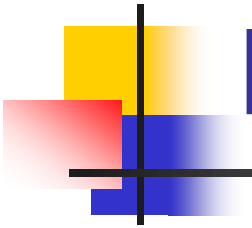


TFTP Summary

- Used to move files between machines on different networks implementing UDP, TFTP server waits for Read/Write request on port 69
- The protocol is very restrictive, in order to simplify implementation
- Used with BOOTP and DHCP Configuration applications (RFC 906, "Bootstrap loading using TFTP")
- The fixed length blocks make allocation straight forward, and the stop and wait mechanism provides reliable delivery, flow control and eliminates the need to reorder incoming data packets

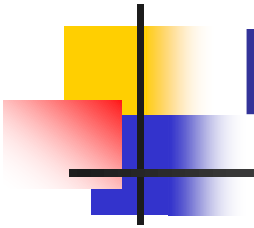


FTP
(File Transfer Protocol)



FTP

- FTP Features
- FTP Protocols
- FTP Operations
- FTP Model
- FTP Control Commands and Replies
- FTP Example



FTP Features

- Used to transfer files between hosts
- Used to manipulate files, for example:
  - List directories
  - Delete files
  - Rename files
- Uses TCP for reliable transfers
- Official Internet protocol

FTP Protocols

- **RFC 959** – File Transfer Protocol
- **RFC 5797** – FTP Command and Extension Registry
- **RFC 4823** – FTP Transport for Secure Peer-to-Peer Business Data Interchange over the Internet
- **RFC 4217** – Securing FTP with TLS
- **RFC 3659** – Extensions to FTP
- **RFC 2577** – FTP Security Considerations
- **RFC 2428** – FTP Extensions for IPv6 and NATs
- **RFC 2389** – Feature negotiation mechanism for the File Transfer Protocol
- **RFC 2228** – FTP Security Extensions
- **RFC 1635** – How to Use Anonymous FTP
- **RFC 1579** – Firewall-Friendly FTP
- **RFC 0913** – Simple File Transfer Protocol
- ……

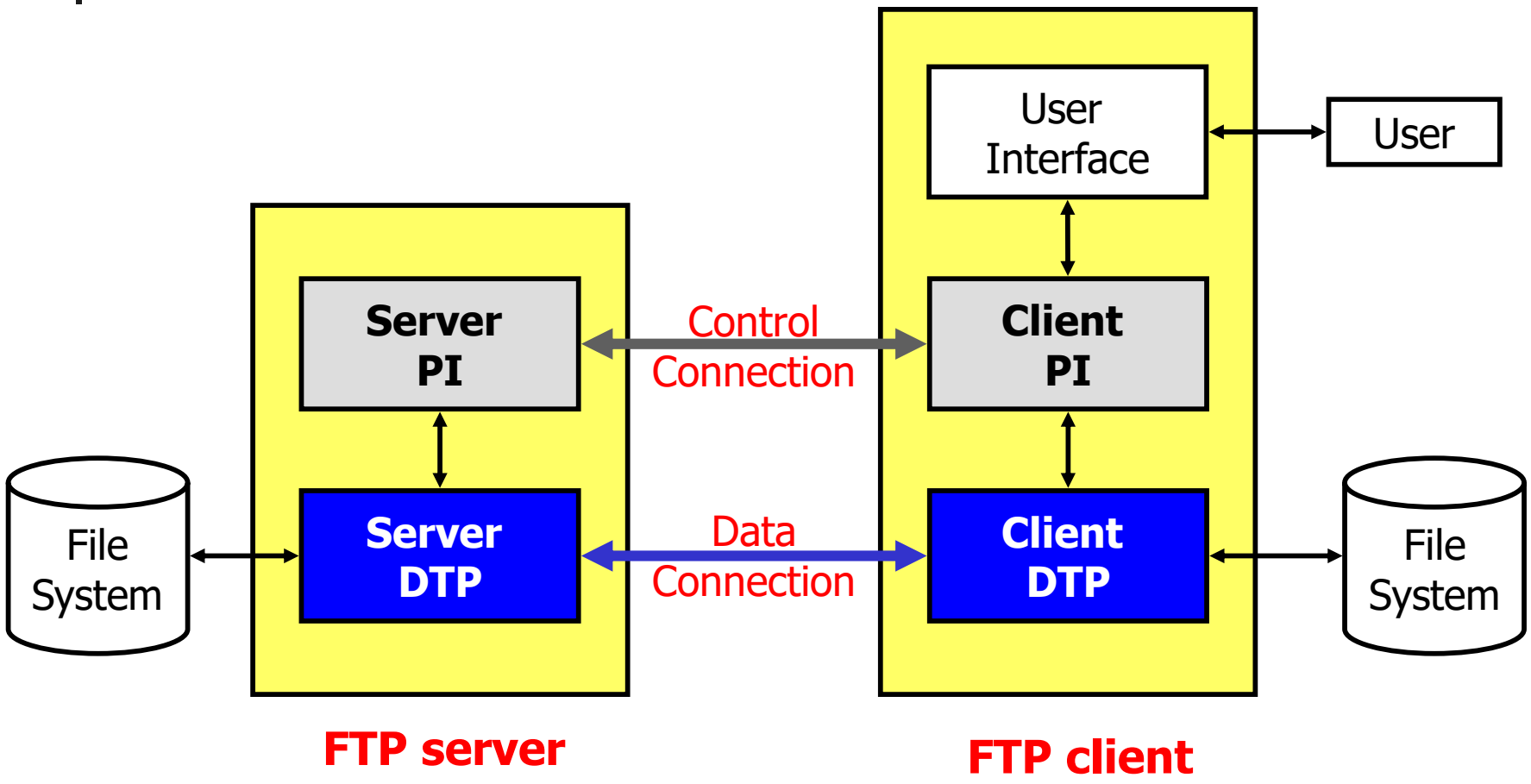
FTP Operation Sequences (user commands)

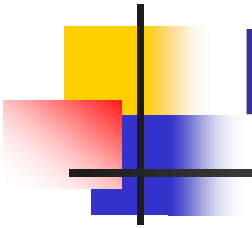
- Open connection to remote host
  - ***ftp*** hostname
  - ***open*** hostname
- Log into server (provide username and password)
  - ***user*** [username [password]]
- Set file transfer mode (such as ASCII or image)
  - ***type*** type-code
  - ***stru*** and ***mode*** commands used to alter transfer
- Transfer files (using get or put commands)
  - ***get*** remote-file [local-file]
  - ***put*** local-file [remote-file]
  - ***mget*** and ***mput*** commands used to transfer multiple files (such as * to transfer all of a directory)
- Perform other file operations
  - ***delete***, ***rename***, ***mkdir***, ***rmdir***, ***ls***, ***dir***, ...
- Exit client (***quit***) or close connection (***close***)

Example of FTP user commands

```
abc@BUPTIA:~$ ftp 192.168.56.101
Connected to 192.168.56.101.
220 (vsFTPd 3.0.2)
Name (192.168.56.101:abc): student
331 Please specify the password.
Password:
230 Login successful.
Remote system type is UNIX.
Using binary mode to transfer files.
ftp> pwd
257 "/home/student"
ftp> ls
200 PORT command successful. Consider using PASV.
150 Here comes the directory listing.
drwxrwxr-x    2 1000     1000         4096 Apr 22 12:11 lab
-rwxrwxr-x    1 1000     1000         7957 Apr 19 23:29 tcpc
-rwxrwxr-x    1 1000     1000         7860 Apr 19 23:17 tcps
-rw-r--r--    1 1000     1000          502 Apr 19 23:18 test
-rwxrwxr-x    1 1000     1000         7722 Apr 06 09:43 udpc
-rwxrwxr-x    1 1000     1000         7720 Apr 06 09:44 udps
226 Directory send OK.
ftp> quit
221 Goodbye.
```

FTP Model (1)


User Interface
User
Server PI
Control Connection
Client PI
File System
Server DTP
Data Connection
Client DTP
File System
FTP server
FTP client



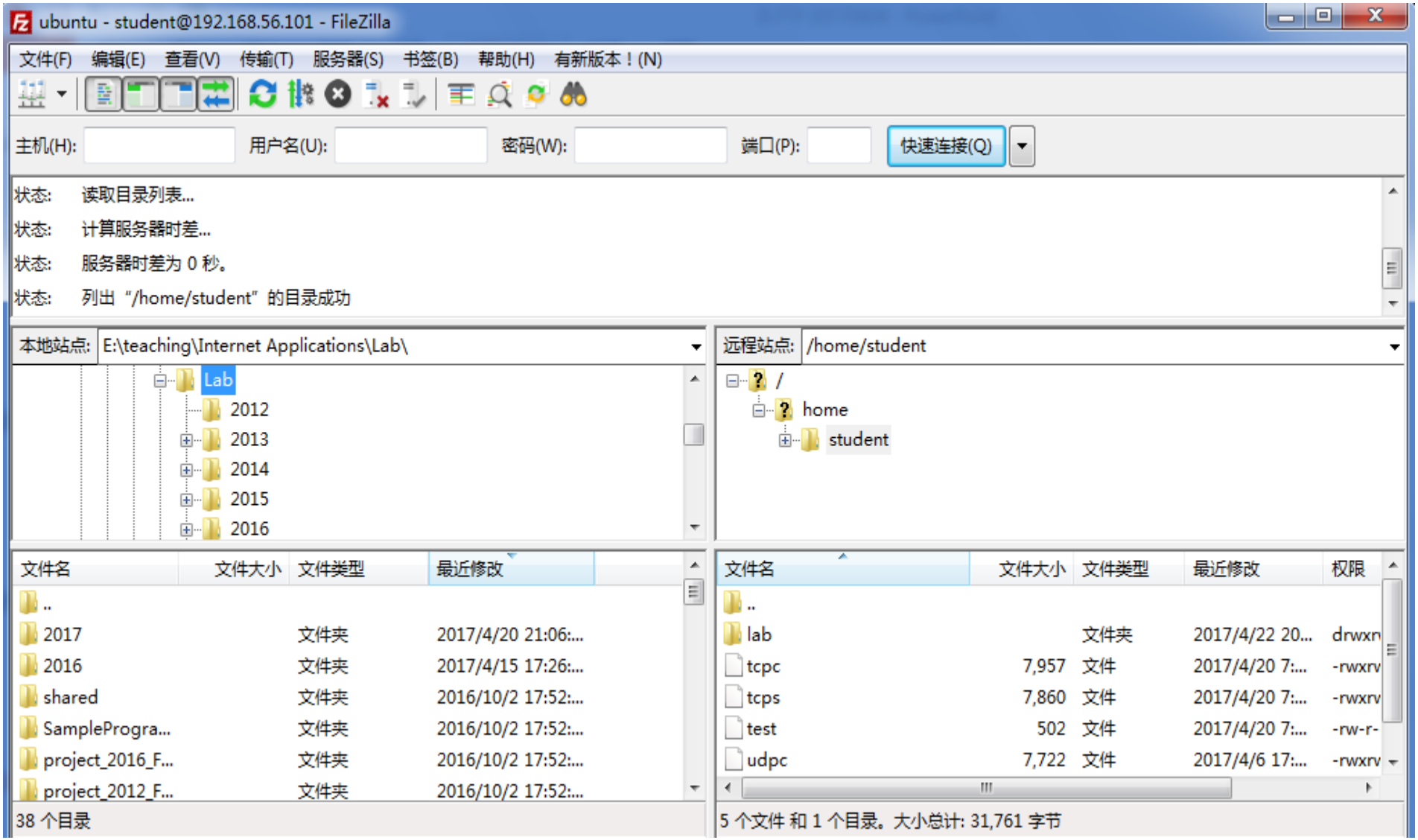
FTP Model (2)

- FTP Client
  - Users interact with Client directly
  - Active open of control connection
  - Control connection uses ASCII plain-text
  - Sends commands (over control connection)
  - Receives replies (over control connection)
  - Data connection used to transfer file data
- FTP Server
  - System process
  - “Listens” for connection on well-known port 21
  - Receives commands
  - Sends replies

FTP Model (3)

- PI (Protocol Interpreter): The user and server sides of the protocol have distinct roles implemented in a user-PI and a server-PI.
- DTP (Data Transfer Process): The data transfer process establishes and manages the data connection. The DTP can be *passive* or *active*.
- Control Connection: The communication path between the client-PI and server-PI for the exchange of commands and replies. This connection follows the Telnet Protocol.
- Data Connection: A full duplex connection over which data is transferred, in a specified mode and type. The data transferred may be a part of a file, an entire file or a number of files. The path may be between a server-DTP and a client-DTP, or between two server-DTPs.

FTP Model (4) - User interface (UI)



FTP Model (5) - Protocol Interpreter (PI)

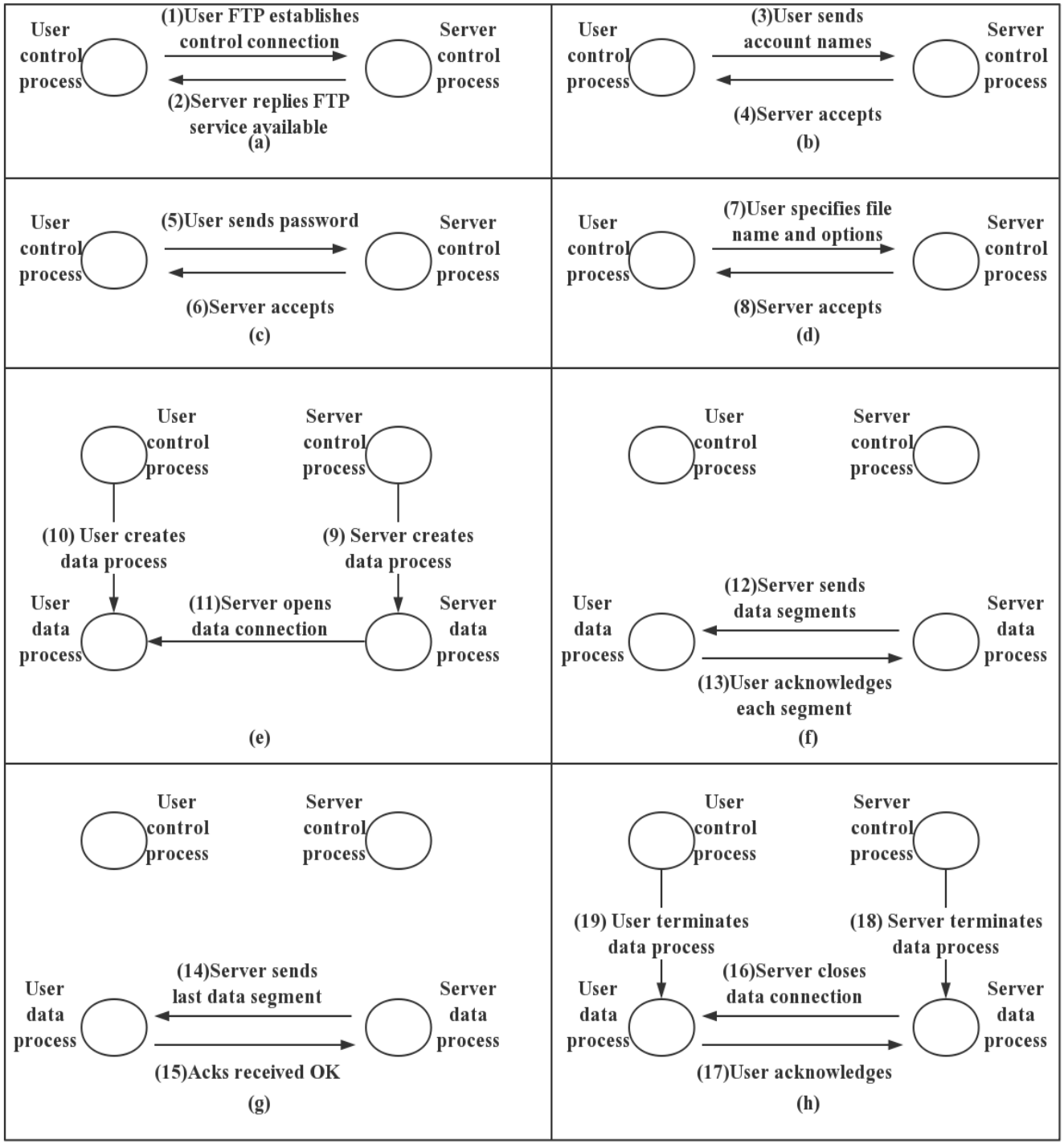
- Interprets the user's commands

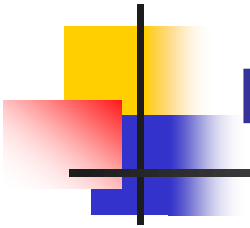
Table 6-1 FTP PI Interpretation

User Command	FTP Command
CD mystuff	CWD mystuff
GET resume.doc	RETR resume.doc
PUT timecard.doc	STOR timecard.doc
DIR	NLST

Overview of an FTP Transfer

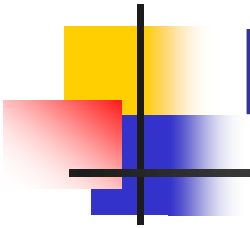
Downloading a file in Active Mode





FTP Control Connection & Data Connection

- Control connection
  - Remain alive as long as the user keeps the FTP session active
  - Passing commands and replies
  - Used to coordinate the ports used for data connection and establish data connection
- Data connection
  - Be created dynamically when needed
  - One data connection persists for one file transfer
  - Used for data transmission



FTP Control Commands

- Usage
  - Usually four (4) characters (such as RETR)
  - Arguments separated by spaces
  - Terminated by CR/LF sequence
  - Sent from client-PI to server-PI
  - The client program translates your requests into the necessary commands and responses
- Three command groups
  - Access control
  - Transfer parameter
  - Service

FTP Control Commands: Access control group

- Specify user name: *USER* username
- Specify password: *PASS* password
- Specify account: *ACCT* account
- Change directory: *CWD* directory
  - *CDUP:* Change to parent directory
- Reinitialize: *REIN*
- Terminate session: *QUIT*

FTP Control Commands: Transfer parameter group(1)

- Define data connection port:
  - *PORT* h1,h2,h3,h4,p1,p2

    --used in *active mode*, telling FTP server the port number of client to accept data connection

    ➢ Example of port number calculation

    PORT 210,25,137,230,23,189

    IP: 210.25.137.230

    Port: 23*256+189=6077

  - *PASV*

    --used in *passive mode*, informs server that client will contact to set up data connections

FTP Control Commands: Transfer parameter group(2)

- The 2 systems may use different ways to represent text and data.
- Command: *TYPE* type-code
  - typical type-code can be:
    - *A* for ASCII (initial default), used for text files
    - *I* for image, used for binary files

FTP Control Commands: Transfer parameter group(3)

- The 2 systems may store files in different directory structures.
- Define file structure: *STRU* structure-code
  - *F* for file (contiguous bytes terminated by EOF, default)
  - *R* for record (terminated by EOR)
  - *P* for page (indexed pages)
- Define file mode: *MODE* mode-code
  - *S* for stream (default)
  - *B* for block
  - *C* for compressed

FTP Control Commands: Service group

- Retrieve (get) file: *RETR* file
- Store (put) file: *STOR* file
- Append to file: *APPE* file
- Delete a file: *DELE* file
- Create a directory: *MKD* directory
- Delete a directory: *RMD* directory
- Rename a file: *RNFR* file and *RNTO* file
- List a directory: *LIST* spec and *NLST* spec
- Others such as *HELP*, *SITE*, *SYST*, ...

FTP Control Replies (1)

- Every command must generate at least one reply
- Sent from server-PI to client-PI
- 3 digit code followed by delimiter and text message
  - Delimiter is space if last line of text message
  - Delimiter is hyphen if not last line of text message

    220-******Welcome to the Network Information Center*******

    220-******Login with username *anonymous* and password *guest*

    220 And more!
  - Numeric code for client program, text for humans

FTP Control Replies (2)

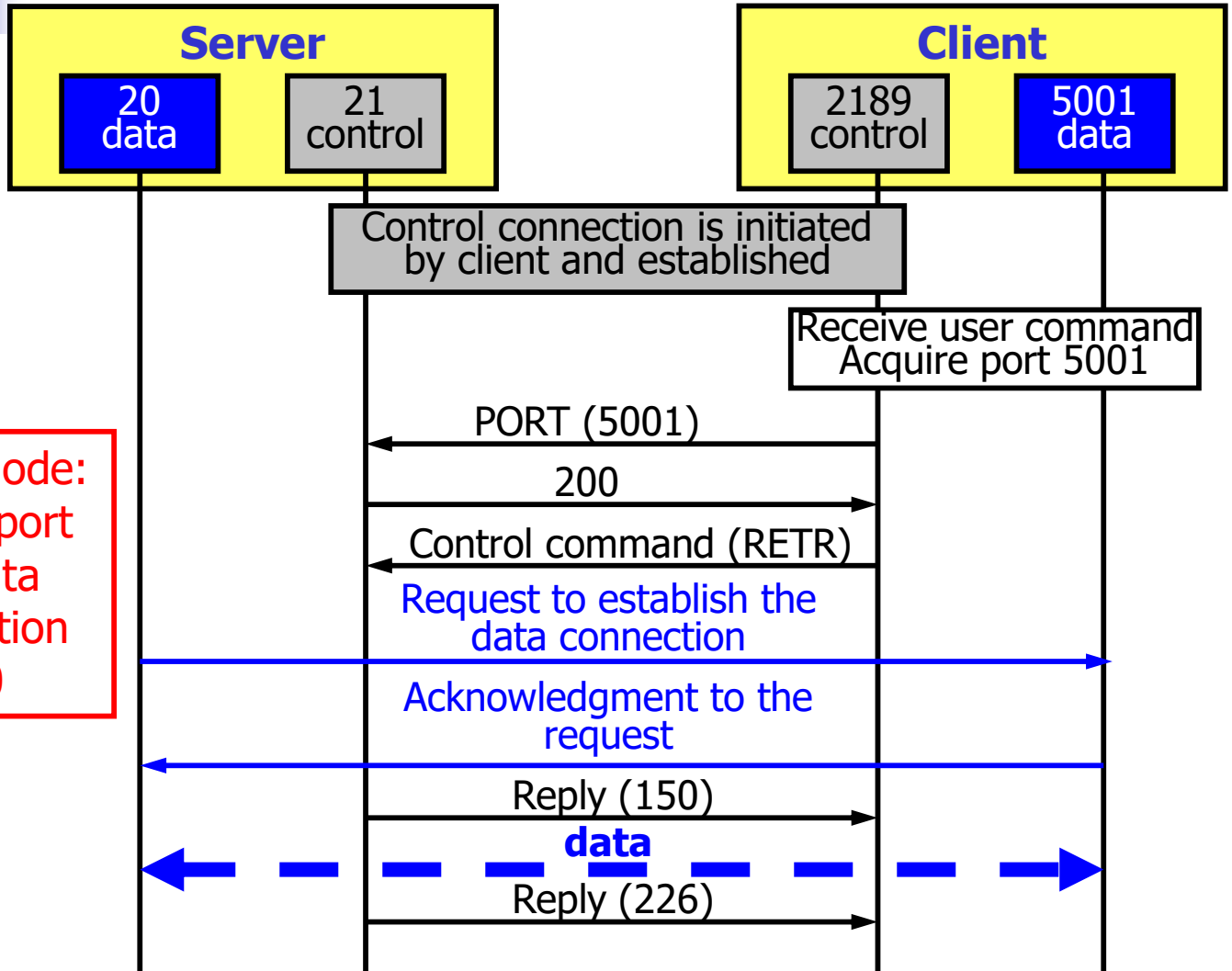
- Reply code meanings

Reply Code	Meaning	Reply Code	Meaning
1nn	Positive preliminary reply	n0n	Syntax
2nn	Positive completion	n1n	Information
3nn	Positive intermediate	n2n	Connection information
4nn	Transient negative (try again)	n3n	Authentication / accounting
5nn	Permanent negative (no such file)	n5n	File System

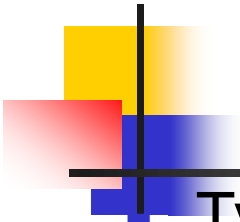
FTP Control Replies (3)

- Reply code examples (see RFC959 for more details)

FTP Control Connection & Data Connection (1)



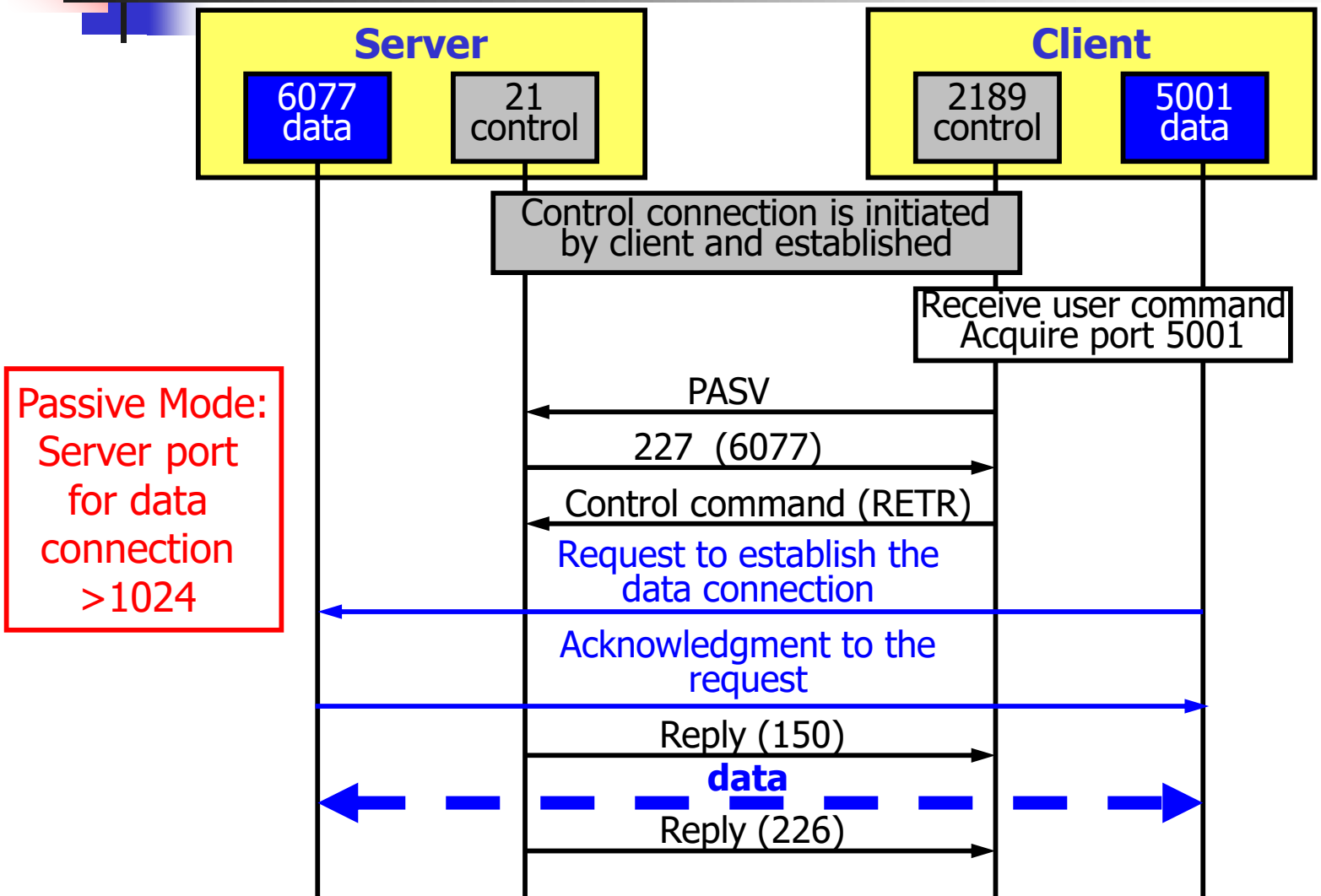
Active Mode: Server port for data connection =20



FTP Control Connection & Data Connection (2)

- Typical data connection handling sequence (in active mode)
  - Client sets up to “listen” on a unique port
  - Client uses local socket information to send PORT command to server
  - Server responds with “200” reply to acknowledge the port number
  - Client sends RETR, STOR, or other transfer command
  - Server sends preliminary reply
  - Server does active open (“connect”)
  - File data sent over connection
  - Server sends “226” or other reply
  - Server/client closes data connection
- Another mode: passive mode
  - Client sends command *PASV*
  - server listens to a specific port and client should access that port

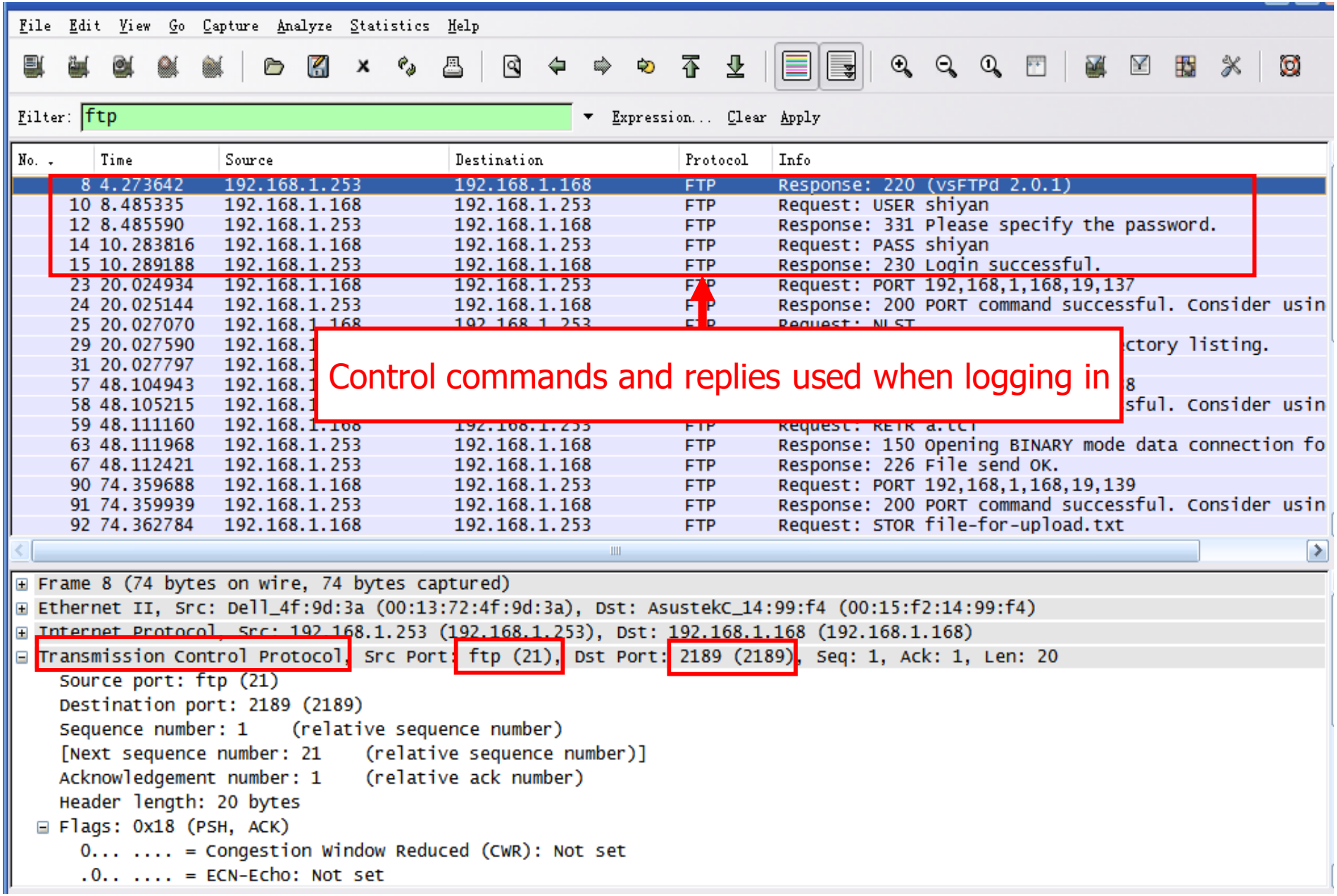
FTP Control Connection & Data Connection (3)

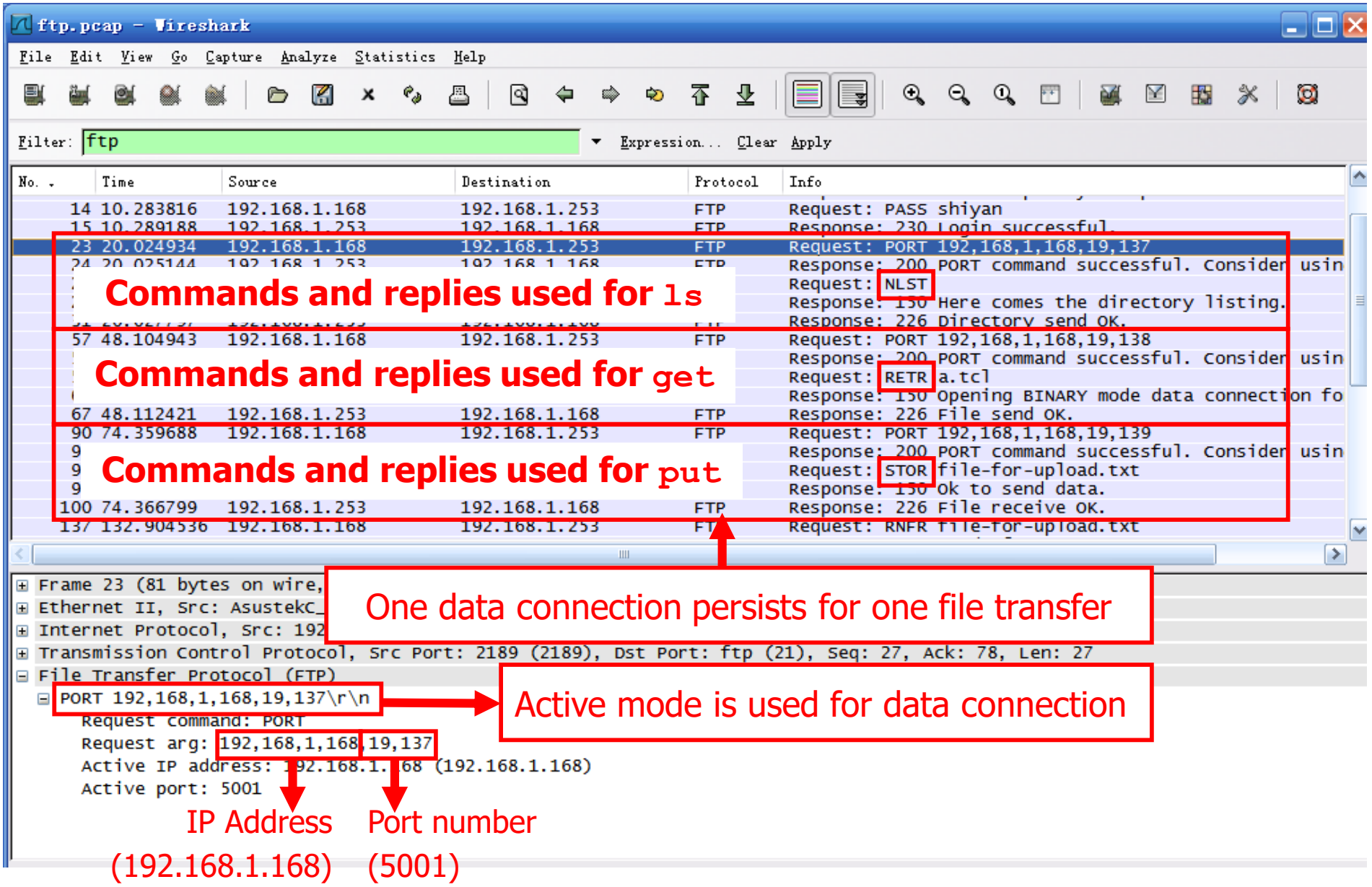


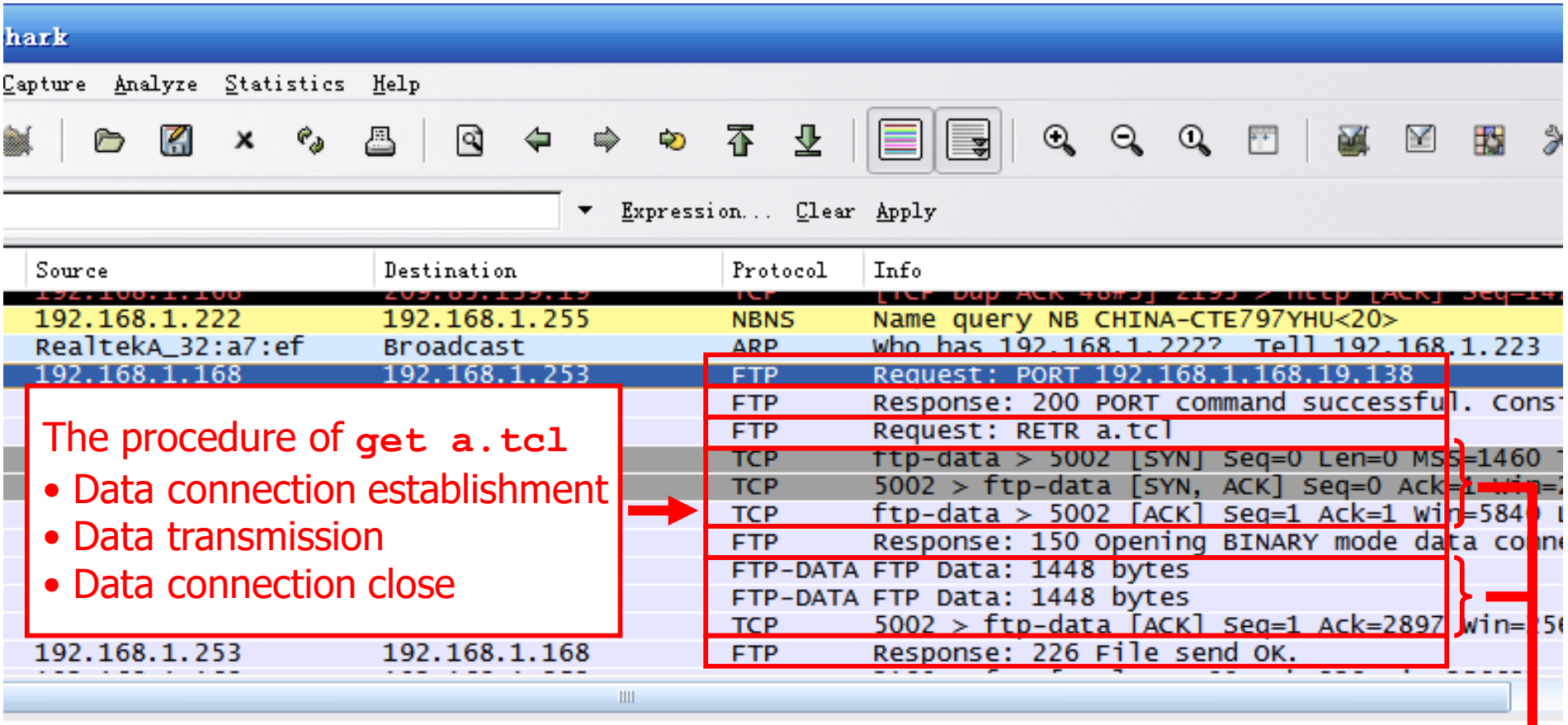
FTP Session Example

- User command sequence

```
d:\temp> ftp 192.168.1.253
ftp> ls
ftp> get a.tcl
ftp> put file-for-upload.txt
ftp> rename file-for-upload.txt mykey.txt
ftp> ls
ftp> delete mykey.txt
ftp> binary
ftp> ascii
```


File Edit View Go Capture Analyze Statistics Help
Filter: ftp
Expression... Clear Apply
No. Time Source Destination Protocol Info
8 4.273642 192.168.1.253 192.168.1.168 FTP Response: 220 (vsFTPd 2.0.1)
10 8.485335 192.168.1.168 192.168.1.253 FTP Request: USER shiyan
12 8.485590 192.168.1.253 192.168.1.168 FTP Response: 331 Please specify the password.
14 10.283816 192.168.1.168 192.168.1.253 FTP Request: PASS shiyan
15 10.289188 192.168.1.253 192.168.1.168 FTP Response: 230 Login successful.
23 20.024934 192.168.1.168 192.168.1.253 FTP Request: PORT 192,168,1,168,19,137
24 20.025144 192.168.1.253 192.168.1.168 FTP Response: 200 PORT command successful. Consider usin
25 20.027070
29 20.027590
31 20.027797
57 48.104943
58 48.105215
59 48.111160
63 48.111968 192.168.1.253 192.168.1.168 FTP Response: 150 Opening BINARY mode data connection fo
67 48.112421 192.168.1.253 192.168.1.168 FTP Response: 226 File send OK.
90 74.359688 192.168.1.168 192.168.1.253 FTP Request: PORT 192,168,1,168,19,139
91 74.359939 192.168.1.253 192.168.1.168 FTP Response: 200 PORT command successful. Consider usin
92 74.362784 192.168.1.168 192.168.1.253 FTP Request: STOR file-for-upload.txt
Control commands and replies used when logging in
Frame 8 (74 bytes on wire, 74 bytes captured)
Ethernet II, Src: Dell_4f:9d:3a (00:13:72:4f:9d:3a), Dst: AsustekC_14:99:f4 (00:15:f2:14:99:f4)
Internet Protocol, Src: 192.168.1.253 (192.168.1.253), Dst: 192.168.1.168 (192.168.1.168)
Transmission Control Protocol, Src Port: ftp (21), Dst Port: 2189 (2189), Seq: 1, Ack: 1, Len: 20
Source port: ftp (21)
Destination port: 2189 (2189)
Sequence number: 1 (relative sequence number)
[Next sequence number: 21 (relative sequence number)]
Acknowledgement number: 1 (relative ack number)
Header length: 20 bytes
Flags: 0x18 (PSH, ACK)
0... .... = Congestion Window Reduced (CWR): Not set
.0.. .... = ECN-Echo: Not set

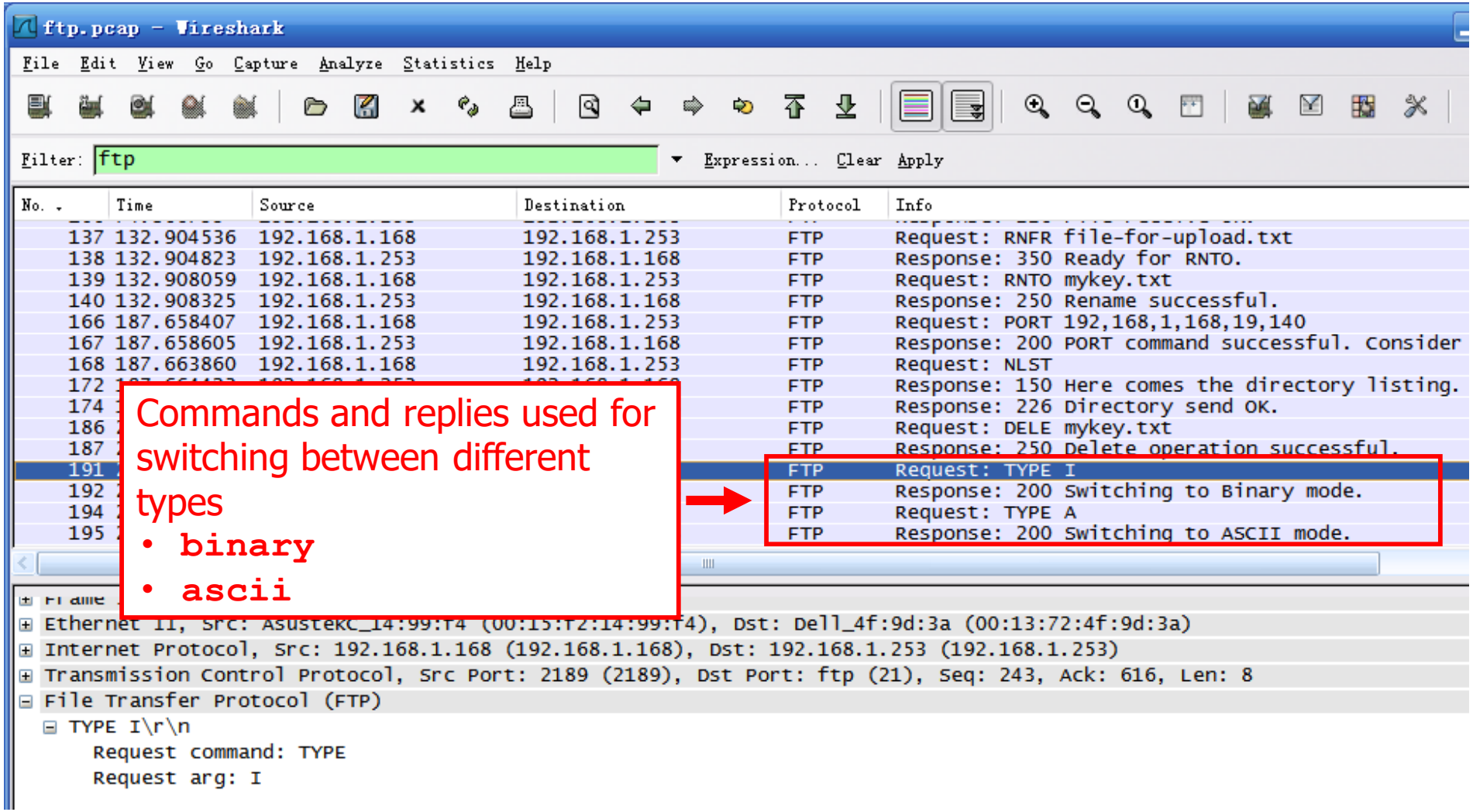

ftp.pcap - Wireshark
File Edit View Go Capture Analyze Statistics Help
Filter: ftp
Expression... Clear Apply
No. Time Source Destination Protocol Info
14 10.283816 192.168.1.168 192.168.1.253 FTP Request: PASS shiyan
15 10.289188 192.168.1.253 192.168.1.168 FTP Response: 230 Login successful.
23 20.024934 192.168.1.168 192.168.1.253 FTP Request: PORT 192,168,1,168,19,137
Response: 200 PORT command successful. Consider usin
Request: NLST
Response: 150 Here comes the directory listing.
Response: 226 Directory send OK.
57 48.104943 192.168.1.168 192.168.1.253 FTP Request: PORT 192,168,1,168,19,138
Response: 200 PORT command successful. Consider usin
Request: RETR a.tcl
Response: 150 Opening BINARY mode data connection fo
67 48.112421 192.168.1.253 192.168.1.168 FTP Response: 226 File send OK.
90 74.359688 192.168.1.168 192.168.1.253 FTP Request: PORT 192,168,1,168,19,139
Response: 200 PORT command successful. Consider usin
Request: STOR file-for-upload.txt
Response: 150 Ok to send data.
100 74.366799 192.168.1.253 192.168.1.168 FTP Response: 226 File receive OK.
137 132.904536 192.168.1.168 192.168.1.253 FTP Request: RNFR file-for-upload.txt
Commands and replies used for ls
Commands and replies used for get
Commands and replies used for put
One data connection persists for one file transfer
Frame 23 (81 bytes on wire,
Ethernet II, Src: AsustekC_
Internet Protocol, Src: 192
Transmission Control Protocol, Src Port: 2189 (2189), Dst Port: ftp (21), Seq: 27, Ack: 78, Len: 27
File Transfer Protocol (FTP)
PORT 192,168,1,168,19,137\r\n
Request command: PORT
Request arg: 192,168,1,168,19,137
Active IP address: 192.168.1.168 (192.168.1.168)
Active port: 5001
Active mode is used for data connection
IP Address
(192.168.1.168)
Port number
(5001)



The procedure of **`get a.tcl`**

- Data connection establishment
- Data transmission
- Data connection close

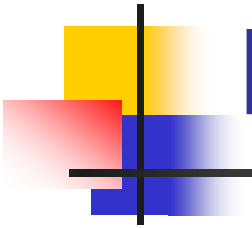
Between server port 20 and client port 5002, which is indicated in PORT command


ftp.pcap – Wireshark
File Edit View Go Capture Analyze Statistics Help
Filter: ftp
Expression... Clear Apply
No. Time Source Destination Protocol Info
137 132.904536 192.168.1.168 192.168.1.253 FTP Request: RNFR file-for-upload.txt
138 132.904823 192.168.1.253 192.168.1.168 FTP Response: 350 Ready for RNTO.
139 132.908059 192.168.1.168 192.168.1.253 FTP Request: RNTO mykey.txt
140 132.908325 192.168.1.253 192.168.1.168 FTP Response: 250 Rename successful.
166 187.658407 192.168.1.168 192.168.1.253 FTP Request: PORT 192,168,1,168,19,140
167 187.658605 192.168.1.253 192.168.1.168 FTP Response: 200 PORT command successful. Consider
168 187.663860 192.168.1.168 192.168.1.253 FTP Request: NLST
FTP Response: 150 Here comes the directory listing.
FTP Response: 226 Directory send OK.
FTP Request: DELE mykey.txt
FTP Response: 250 Delete operation successful.
191 FTP Request: TYPE I
192 FTP Response: 200 Switching to Binary mode.
194 FTP Request: TYPE A
195 FTP Response: 200 Switching to ASCII mode.
Commands and replies used for switching between different types
• binary
• ascii
Ethernet II, Src: AsustekC_14:99:f4 (00:15:f2:14:99:f4), Dst: Dell_4f:9d:3a (00:13:72:4f:9d:3a)
Internet Protocol, Src: 192.168.1.168 (192.168.1.168), Dst: 192.168.1.253 (192.168.1.253)
Transmission Control Protocol, Src Port: 2189 (2189), Dst Port: ftp (21), Seq: 243, Ack: 616, Len: 8
File Transfer Protocol (FTP)
TYPE I\r\n
Request command: TYPE
Request arg: I

Traditional FTP vs. Anonymous FTP

- Traditional FTP
  - Must log in to FTP server before transfers
  - You log into a specific account with a password
    - i.e. your own user account
  - You can transfer to and from directories accessible to that account

- Anonymous FTP
  - You log in with "anonymous" as the account name
  - Give your e-mail address as password
  - Host gives you access to public directories
  - Usually for downloading only
  - Not truly anonymous: your computer's IP address is known

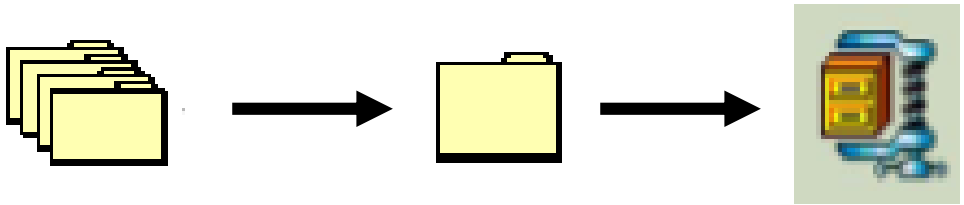


FTP via the Web

- Downloading Files
  - In the browser, type the URL: ftp://hostname
    - Note that the URL begins with “ftp”, not “http”
  - This will log you into that host (anonymous FTP) and show you a list of files and directories
  - Go down the directory path to the directory you want
  - Click on the filename to start the downloading
  - Uses binary file transfer

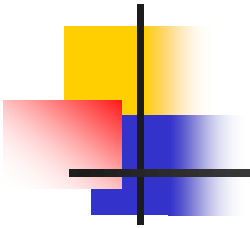
FTP Archiving

- Many FTP files are archived
- Two-step process
  - First, several files are combined into one archive to avoid having to make multiple downloads
  - Second, the combined files are compressed to reduce download times
  - Receiver must unarchive the files to read them

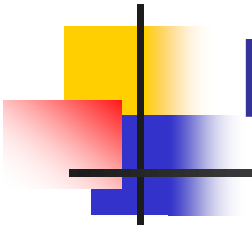


1. Combine (tar)

2. Compress (zip)



NFS
(Network File System)

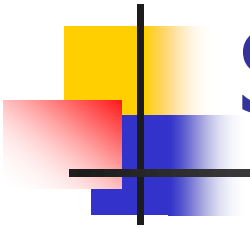


Brief Introduction To NFS

- The Network File System (NFS) is a client/server application that lets a computer user view and optionally store and update file on a remote computer as though they were on the user's own computer
- The user's system needs to have an NFS client and the other computer needs the NFS server
- Earlier versions of NFS use UDP
- NFS was developed by Sun Microsystems

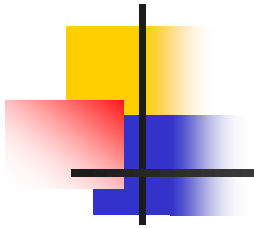
Comparing FTP/NFS

FTP	NFS
FTP just provides file transfer	NFS provides transparent file access for clients to files and file-systems on a server
With FTP a complete copy of the file is made	NFS accesses portions of a file
FTP uses TCP	NFS usually uses UDP on port 2049

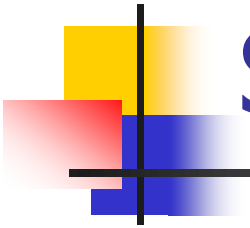


Secure FTP

- SSL-FTP (Secure Socket Layer FTP)
- SFTP (Secure File Transfer Program)
- SCP (Secure Copy)

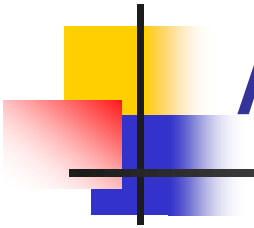


Summary



Summary Of Key Points

- TFTP
  - TFTP packet format
  - TFTP transfer mode
  - Sorcerer's Apprentice Syndrome
  - Typical communication procedure
- FTP
  - FTP model
  - FTP control commands and replies
  - Control connection vs. Data connection
  - Active FTP vs. Passive FTP
  - Traditional FTP vs. Anonymous FTP



Abbreviations

EOF	End Of File
EOR	End Of Record
FTP	File Transfer Protocol
NFS	Network File System
SAS	Sorcerer's Apprentice Syndrome
TFTP	Trivial File Transfer Protocol