2024년 1학기 **시스템프로그래밍실습** 12주차

FTP3-2

System Software Laboratory
Dept. of Computer Engineering,
Kwangwoon Univ.

과제 세부 일정

과제 차수	세부 차수	강의 및 구현 내용
1	1	String 관련 함수, getopt 함수
	2	ls 명령어 구현을 위한 file system
	3	파일 관련 명령어 구현 함수, Server, client 구현 및 FTP Simulation
2	1	소켓 프로그래밍
	2	fork 함수 및 시그널 관련 함수 (1)
	3	fork 함수 및 시그널 관련 함수 (2)
3	1	User authentication/access control
	2	**Split connection/transmission mode**
	3	Log file

Additional Operation

- **Overview**
  - Additional command
    - Send username, password with convert of command.
  - Interoperability 상호운용성
    - Synchronization 동기화
    - Reply processing 회신처리
  - Implement a new FTP command processing
    - PORT - open a data **port**
      - automatically generated command
    - RETR - **retr**ieve a remote file 원격파일검색
      - user command: **get**
    - STOR - **stor**e a file on the remote host / host에 파일저장
      - user command: **put**
    - TYPE - set transfer **type 전송타입설정**
      - user command: **type**

Additional Command

- **Client receive username and password**
  - It receive username and password from standard input

  - If it received username from user
    - then send "**USER** username" to server.
  - If it received password from user
    - then send "**PASS** password" to server.

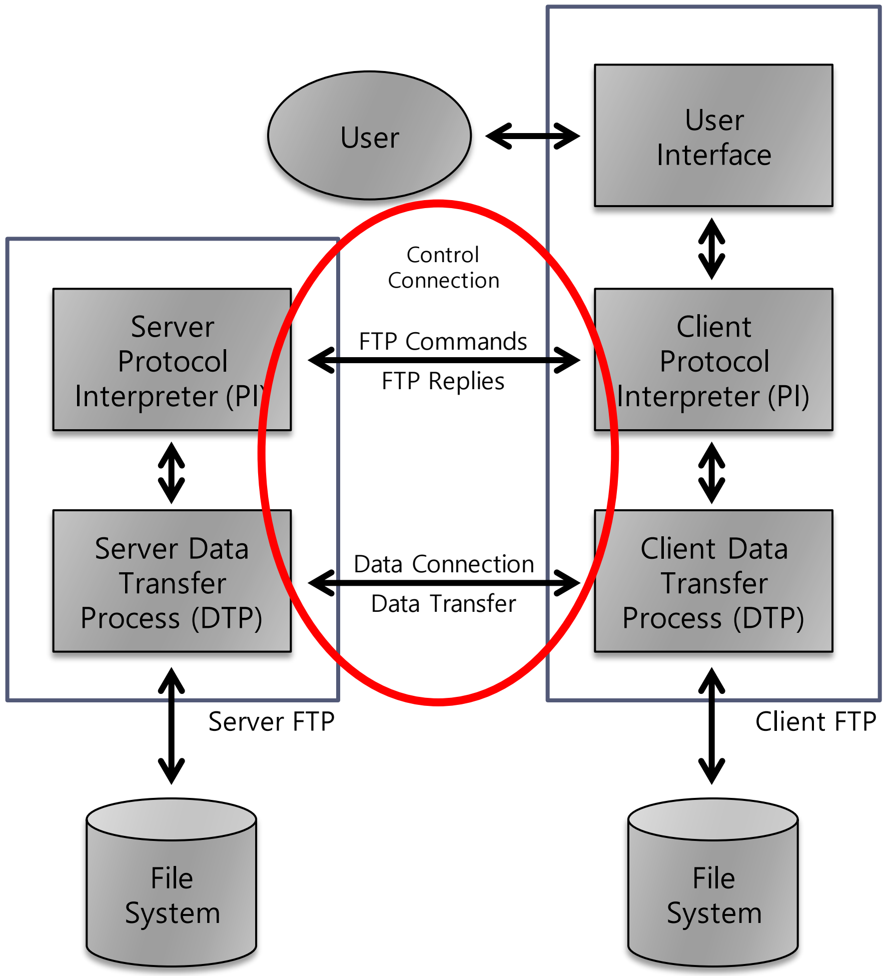
Interoperability (1/2)

- **Synchronization**
  - After the control connection is established
  - Server
    - Write welcome msg. to client via control connection
  - Client
    - Read the server's reply
    - and then start user authentication

Interoperability (2/2)

- **Reply processing**
  - Server
    - not only string type reply
    - but also supply a **numerical reply** to client
  - Client
    - handle a numerical reply

Split Connection


User
User Interface
Control Connection
FTP Commands
FTP Replies
Server Protocol Interpreter (PI)
Client Protocol Interpreter (PI)
Server Data Transfer Process (DTP)
Data Connection
Data Transfer
Client Data Transfer Process (DTP)
Server FTP
Client FTP
File System
File System

Control Connection (1/2)

- **Client**
  - **send FTP command**
  - **print a numerical reply from server**
    - 1xx: Positive Preliminary reply 긍정적 예비 응답
    - 2xx: Positive Completion reply 긍정적 완료 응답
    - 3xx: Positive Intermediate reply 긍정적 중간 응답
    - 4xx: Transient Negative Completion reply 일시적인 부정적 완료 회신
    - 5xx: Permanent Negative Completion reply 영구적인 부정적 완료 회신

Control Connection (2/2)

- **Server**
  - Server replied Keyword like "REJECTION", "OK", "FAIL" in FTP#3-1.
  - Now Server supply **a numerical reply** to client
    - **[ex]**
    - 150: File status okay; about to open data connection.
    - 200: Command okay
    - 220: Service ready for new user
    - 221: Service closing control connection.
    - 226: Closing data connection. Requested file action successful (for example, file transfer or file abort).
    - 230: User logged in, proceed
    - 331: User name okay, need password
    - 350: Requested file action pending further information
    - 500: Syntax error, command unrecognized
    - 501: Syntax error in parameters or arguments
    - 530: Not logged in.
    - 550: Requested action not taken. File unavailable (e.g., file not found, no access).
  - https://en.wikipedia.org/wiki/List_of_FTP_server_return_codes 참고

Data Connection

- **Make data connection**
  - control connection과는 다르게, server가 client에 접속
  - client는 데이터 전송을 위해, 임의의 포트번호를 생성하고 **PORT** command를 이용하여 server에게 알림
    - E.g. **PORT** 128,134,54,83,**150,48 (150,48 = 38448)**
      - First four numbers are IP address
      - Last two number port number
      - Separated by commas
      - Port number
        - First number is High 8bit code for port number
          ex) 150 → 10010110
        - Last number is lower 8bit code for port number
          ex) 48 → 110000
        - Convert 16bit code to decimal number.
          ex) 150,48 → 10010110 00110000 → **38448**
  - Server는 client가 알려준 포트번호로 접속

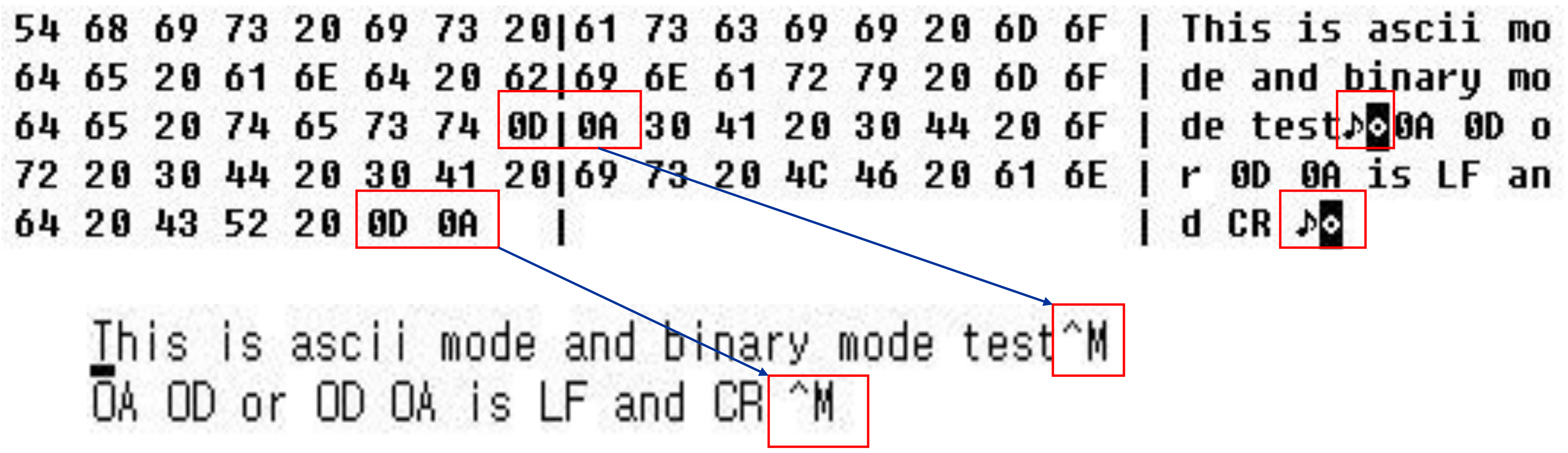
File Type (1/3)

- **mode bin**
  - It doesn’t matter, Just open file
    - Copy to buffer and write socket descriptor

- **mode ascii**
  - Some problem with this mode
    - Different operating systems(window, unix, …) use different EOL characters.
    - ASCII mode automatically converts EOL characters.
  - CR & LF : HEX decimal value is 0D 0A
    - 0D == Char ‘₩r’
    - 0A == Char ‘₩n’

File Type (2/3)

- **Original text**
  - This is ascii mode and binary mode test
  - 0A 0D or 0D 0A is LF and CR
- **HEX format**



```
54 68 69 73 20 69 73 20|61 73 63 69 69 20 6D 6F | This is ascii mo
64 65 20 61 6E 64 20 62|69 6E 61 72 79 20 6D 6F | de and binary mo
64 65 20 74 65 73 74 0D|0A 30 41 20 30 44 20 6F | de test♪◙0A 0D o
72 20 30 44 20 30 41 20|69 73 20 4C 46 20 61 6E | r 0D 0A is LF an
64 20 43 52 20 0D 0A    |                        | d CR ♪◙
```

```
This is ascii mode and binary mode test^M
0A 0D or 0D 0A is LF and CR ^M
```

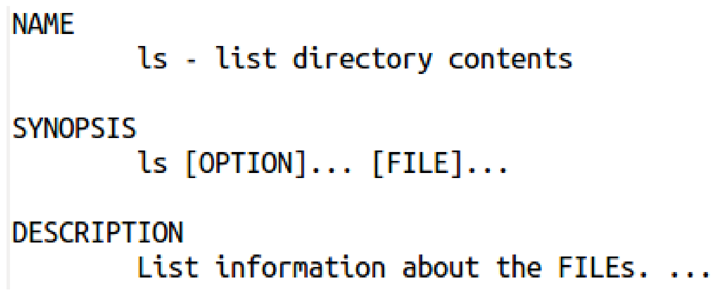
File Type (3/3)

- **When the client set mode ASCII,**
  - 파일 read 후 0D 0A 또는 0A 0D를 0A로 변환 후 전송
    - ₩r₩n or ₩n₩r -> ₩n
    - client OS는 무조건 unix라고 가정함

File Type example (1/3)

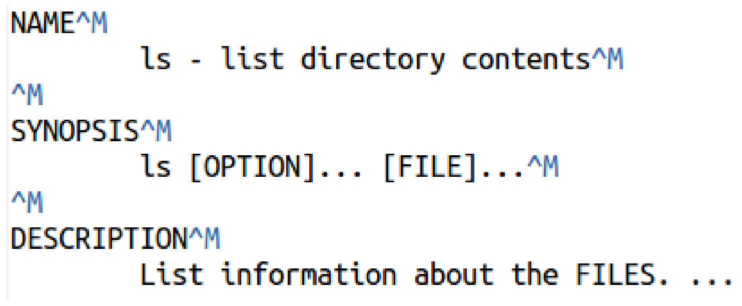
- **vi –b ls_unix.txt**
  - 리눅스에서 작성한 파일

※ vi -b option: Binary mode



```
NAME
        ls - list directory contents

SYNOPSIS
        ls [OPTION]... [FILE]...

DESCRIPTION
        List information about the FILEs. ...
```

- **vi –b ls_win.txt**
  - 윈도우에서 작성 후 linux로 옮긴 파일



```
NAME^M
        ls - list directory contents^M
^M
SYNOPSIS^M
        ls [OPTION]... [FILE]...^M
^M
DESCRIPTION^M
        List information about the FILES. ...
```

File Type example (2/3)

- **type binary인 경우**
  - 파일 read 후 그대로 write

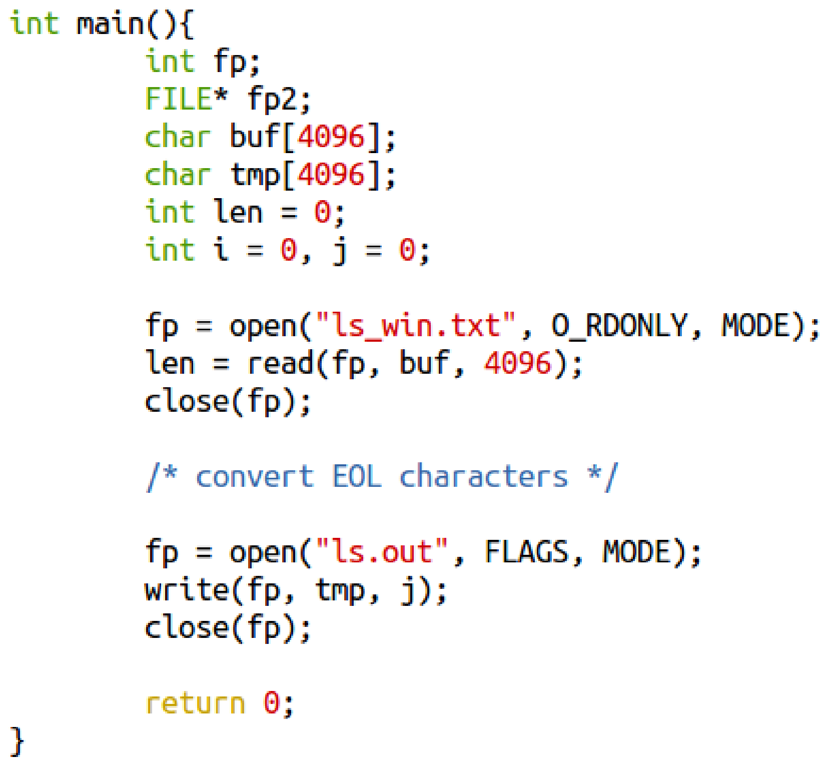
```c
#include <stdio.h>
#include <stdlib.h>
#include <fcntl.h>
#include <unistd.h>
#include <string.h>

#define FLAGS (O_RDWR | O_CREAT | O_TRUNC)
#define MODE (S_IRUSR | S_IWUSR | S_IRGRP | S_IROTH)

int main(){
        int fp;
        char buf[4096];
        int len = 0;
        int i = 0;

        fp = open("ls_win.txt", O_RDONLY, MODE);
        len = read(fp, buf, 4096);
        close(fp);

        fp = open("ls.out", FLAGS, MODE);
        write(fp, buf, len);
        close(fp);

        return 0;
}
```

vi -b ls.out

```
NAME^M
        ls - list directory contents^M
^M
SYNOPSIS^M
        ls [OPTION]... [FILE]...^M
^M
DESCRIPTION^M
        List information about the FILES. ...
```

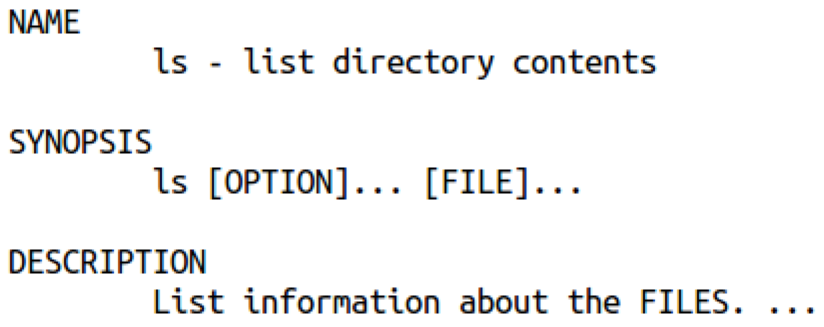
File Type example (3/3)

- **type ascii인 경우**
  - 파일 read 후 ₩r₩n or ₩n₩r => ₩n 변경 후 전송



```
int main(){
        int fp;
        FILE* fp2;
        char buf[4096];
        char tmp[4096];
        int len = 0;
        int i = 0, j = 0;

        fp = open("ls_win.txt", O_RDONLY, MODE);
        len = read(fp, buf, 4096);
        close(fp);

        /* convert EOL characters */

        fp = open("ls.out", FLAGS, MODE);
        write(fp, tmp, j);
        close(fp);

        return 0;
}
```

vi -b ls.out



```
NAME
        ls - list directory contents

SYNOPSIS
        ls [OPTION]... [FILE]...

DESCRIPTION
        List information about the FILES. ...
```