SSE2030: INTRODUCTION TO COMPUTER SYSTEMS (Fall 2013)

Programming Assignment #3

Due: October 15, 11:59:59 PM

1. Introduction

- 이번 과제를 통해 소켓 프로그래밍에 익숙해 지도록 한다.

2. Overview

- Client / Server 관계를 갖는 2개의 프로세스를 생성한다.
- Client는 여러 개의 URL을 입력 받아 해당 host, port에 접속하고 pathname를 전송한다.
- Server는 전달받은 pathname에 해당하는 파일을 read하여 Client로 헤더와 함께 전송한다.
- Client와 Server는 아래의 spec과 같이 구현한다.

3. Specification

- Client의 구현
  - ✓ `Client`는 주어진 `Skeleton` 코드의 `client.c` 파일을 구현한다.
    - ◆ main 함수를 포함하여야 하고, 그 이외의 구현에 필요한 함수들은 추가할 수 있다.
  - ✓ 프로세스의 Argument는 없음
  - ✓ URL의 입력
    - ◆ Client는 표준입력으로 아래 정의된 URL을 문자열을 입력 받는다.
    - ◆ read() 함수를 이용한다.
    - ◆ **서로 다른 URL은 '₩n'으로 구분된다.**
  - ✓ URL의 정의

◆ 형식: rfile://host[:port]/pathname

- rfle:// 는 이번 과제를 위해 새로 정의된 프로토콜 이름이다.
- host는 dotted decimal notation(예를 들어, 1.2.3.4) 혹은 domain name (예를 들어, sys.skku.edu) 또는 localhost(127.0.0.1)가 가능하다.
- port는 해당 서버에 접속할 port 번호이고 생략 가능하다. 생략된 경우에는 default port는 1234 이다.
- pathname은 일반적인 Linux 상의 파일 경로이다.

  i. pathname은 PA#1에서 설명한 것과 같이 서버상의 상대경로이고, ${HOME_DIR} 이하의 상대경로이다.

  ➔ 서버의 ${HOME_DIR}/pathname 에 파일이 위치한다.

◆ URL 형태 오류가 있는 경우는 Skeleton 코드 중 다음 함수를 호출한다.

- **print_error_code(-3); 혹은**
  print_error_code (ERR_INVALID_URL);
  // void print_error_code (int status);

◆ URL 입력 예)
rfile://1.2.3.4:9000/data.txt
rfile://localhost:1234/result.txt
rfile:///news.html
rfile://sys.skku.edu/list.dat

◆ 파싱된 각 URL이 정상적인 경우 host, port정보를 이용하여 연결하고 pathname를 data로 전송한다.

- 예) rfile://1.2.3.4:9000/mail/data.txt 인경우

  host : 1.2.3.4

  port : 9000

  pathname : /mail/data.txt

◆ **입력받은 URL이 "QUIT₩n"이면 프로그램을 종료한다.**

✓ Data전송

◆ 각 URL의 pathname 이하의 스트링을 Server로 전달하고, 마지막 문자열인 ₩n 을 함께 전송하도록 한다.

- 표준 입력에서 문자열 입력 후 엔터를 입력하면 ₩n가 포함되어 있다.

  참고) 터미널에서 'hello' 입력 후 엔터를 치면, 6번째 byte에 ₩n(line feed, ASCII - 0A)가 포함되어 있다.

- ₩n은 전송하는 Data의 끝을 알리는 용도임

- 예)
  /data.txt₩n
  /test/result.dat₩n

✓ Server 응답의 처리 (Server의 Data 전달 방법)

◆ Server는 전달한 pathname의 파일을 Client로 전송하는데, 다음과 같은 형식으로 전송된다.

◆ 헤더: status code(4bytes) + file size(4bytes)

- Status code: 파일 존재여부

  파일이 있으면: 0

  파일이 없으면: -1

  파일 접근 권한이 없으면: -2

  그 외: -4

- File size: 해당 파일의 크기

- Status Code, File Size는 숫자를 전송하는 부분이므로 Big Endian (network representation)로 사용해야 함을 주의한다.

  i. Server 구현할 때 이 점을 꼭 확인해야 한다.

Status Code(4bytes)	File Size(4bytes)
File Data(파일 내용)	

◆ File Data 저장

- Status code가 0인 경우(정상)에 File Size만큼 +8bytes 이후에 전달되는 buffer를 현재 Client가 수행중인 폴더에 저장한다.

  i. 저장할 파일의 이름은 URL의 파일이름과 동일하게 한다.

  rfile://localhost/test/data.txt 이면, data.txt 파일로 저장한다.

- 이미 파일이 존재한다면, overwrite 하도록 한다.

◆ 결과 출력

- 저장이 완료되면 다음 함수를 이용하여 결과를 출력한다.

i. 예를 들어 File size가 1200bytes이고, 파일이름이 data.txt인 경우

// void print_msg(char* file_name, int file_size);

print_msg("data.txt", 1200);

➔ OK:data.txt:1200 가 출력됨

◆ Status code가 0이 아닌 경우(비정상) Skeleton 코드 중 다음 함수를 호출

- **void print_error_code (int status);**

✓ Client의 종료

◆ **앞서 설명한 바와 같이 표준입력으로 QUIT₩n이 입력되면 프로그램을 종료한다.**

◆ Client는 QUIT₩n이 입력되기 전까지 URL입력 → Server연결 → pathname전달 → 파일저장 → URL 입력을 반복한다.

- Server의 구현

✓ Server는 주어진 Skeleton 코드의 server.c 파일을 구현한다.

◆ main 함수를 포함하여야 하고, 그 이외의 구현에 필요한 함수들은 추가할 수 있다.

✓ 프로세스 Argument 설명

◆ Argument 개수는 1개이다.

◆ 서버가 사용할 기본 포트이다.

- 인자 값이 없을 경우 1234포트를 사용한다.
- 채점 서버에서 랜덤하게 주어질 수 있다. (중복 방지)

✓ Client 요청 대기

◆ 인자로 주어진 port를 사용한다.

◆ listen() → accept() 해서 연결 요청을 대기한다.

- ◆ Client와 연결되면 read() 하여 ₩n 이전 문자열까지 읽는다.

✓ Data 처리

- ◆ 읽어온 pathname의 내용을 환경변수 ${HOME_DIR} + pathname 형태로 전체 경로를 만든다.
  - ➢ 예)

    ${HOME_DIR}이 /home/test 이고, 읽어온 pathname가 /data.txt 이면 /home/test/data.txt 이 전체 경로가 된다.
- ◆ 전체 경로에 해당하는 파일이 존재하면 헤더와 함께 해당 파일을 읽어 내용을 전송한다.
  - ➢ 전송할 내용의 형식은 Client절에 나와 있는 구성방법을 참고한다.
    - i. Status Code(4bytes) + File Size(4Bytes) + File Data(파일 내용)
  - ➢ 파일의 크기는 stat 함수를 이용하여 알 수 있다.
- ◆ Data를 Client로 전송한 다음 연결을 close하고 다시 accept()로 요청을 대기한다.

✓ Server의 종료

- ◆ Server는 종료하지 않고, 계속해서 동작을 유지시킨다.

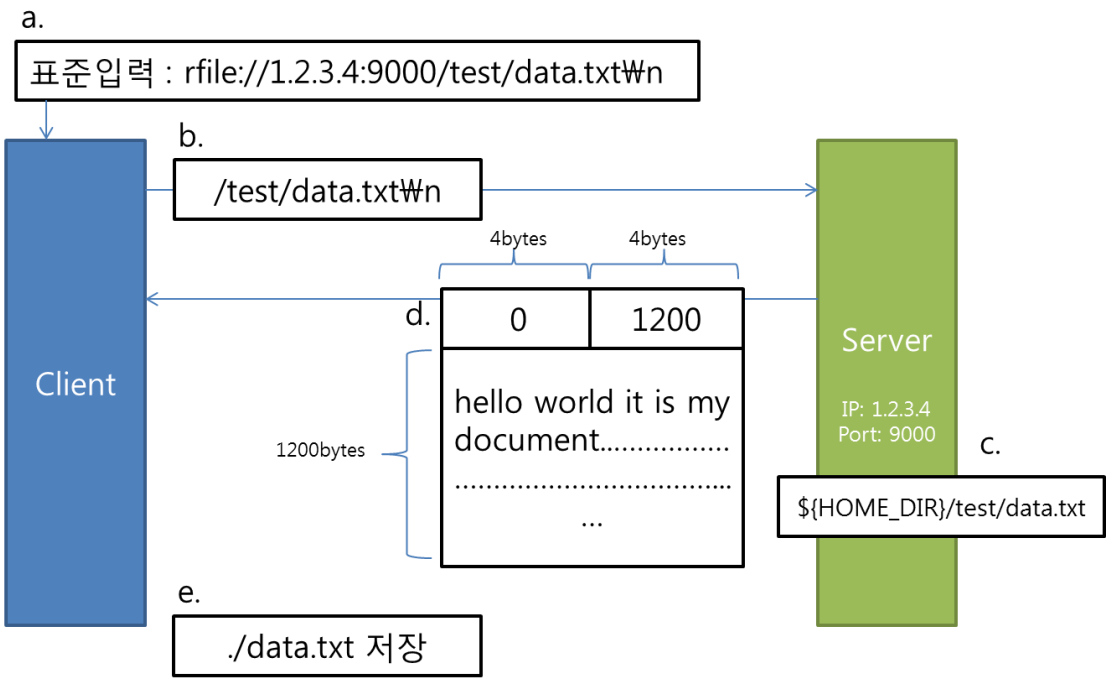
● Status 출력

✓ **print_error_code () 함수를 호출하고, 다음의 define을 참고한다.**

```
#define ERR_NO_FILE         (-1) // 파일이 없는 경우(Client)
#define ERR_NOT_ALLOWED     (-2) // 파일 권한이 없는 경우(Both)
#define ERR_INVALID_URL     (-3) // URL 형식이 잘못된 경우(Client)
#define ERR_CONNECTION_FAIL (-4) // 서버 연결 실패(Client)
#define ERR_INVALID_DOMAIN  (-5) // 도메인이 존재하지 않는 경우(Client)
```

- ◆ Server의 Status Code가 0이 아닌 경우 출력한다. (common.h 정의)

2. Simple Control Flow



3. Background and Additional Information

- 리눅스는 Man-page (manual-page)를 통해 상세한 메뉴얼을 제공한다.
- 잘 모르는 사항은 TA에게 질문하거나, 구글을 통한 검색을 권장한다.
- 소켓 관련 강의 교안을 참고한다.

4. Restriction

- 과제는 본인이 직접 설치한 리눅스 환경에서 구현한다.
  - ✓ 테스트 서버에서 컴파일, 실행시킬 수 없는 코드는 과제 제출물로 인정하지 않는다.
  - ✓ 테스트 서버는 Linux kernel 3.0-32bit, gcc 4.6.1 를 이용한다.
- 구현을 완료하면 http://sys.skku.edu에 과제를 제출하고, 수행하여 결과를 확인한다.
- 과제 점수는 컴파일/실행 가능 여부, 작성한 함수의 완성도, 출력 결과 및 작성한 문서에 의하여 평가된다.
- 시스템 콜을 사용하여 프로그램을 작성하며, Standard C library의 `malloc()`, `free()`, `getenv()` 이외 함수는 사용하지 않는다.
- 파일 입출력의 경우 `open()`, `read()`, `write()`, `close()` 시스템 콜 함수 중 필요한 것을 선택하여 사용한다.
- 프로세스 생성의 경우 `fork()`, `exec*()` 계열, `wait*()` 계열 시스템 콜 함수와 관련된 매크로 함수를 선택하여 사용한다.

- ✓ 추가적으로 `stat()`, `fstat()` 시스템 콜 함수를 사용할 수 있다.

- `Socket API`는 수업시간에 배운 것으로 활용한다.
- 필요한 경우 함수를 직접 client.c, server.c 안에 구현하여 추가할 수 있다.
- client.c, server.c 이외의 파일은 수정할 수 없고, 설사 수정해서 제출하더라도 채점 서버에서 자동으로 제외하여 컴파일한다.

5. Skeleton Codes

- 본 과제 수행을 위하여 아래와 같이 4개의 파일이 주어진다.
  - ✓ `Makefile` GNU make utility를 위해 사용되는 파일
  - ✓ common.h Client/Server가 함께 사용하는 헤더 정보(오류 코드)
  - ✓ `client.c` Client 구현할 파일
  - ✓ `server.c` Server 구현할 파일

6. Hand in instruction

- 작성한 프로그램 코드 상단에 이름과 학번을 주석으로 표기한다.
- 작성한 과제 코드는 "**학번.tar.gz**" 형태로 압축해 http://sys.skku.edu에 제출한다.
  - ✓ 작업 디렉토리에서 "`$ make tar 학번`" 명령을 이용해 파일을 "학번.tar.gz" 으로 압축할 수 있다.
- 작성한 과제 코드의 디자인와 별도로 구현에 관한 내용을 담은 보고서를 "**학번.pdf**" 파일로 채점 서버에 제출한다.

7. Logistics

- 과제 제출 결과는 http://csl.skku.edu/SSE2030F13/Assignments 에서 확인할 수 있다.
- 과제 제출 기한은 http://sys.skku.edu 서버시간을 기준으로 하며, 기한 이후 24시간 내에 제출할 경우 30%, 48시간 내에 제출할 경우 60% 감점한다. 그 이후 제출이 불가능하며, 0점 처리한다.
- 과제에 대한 의논은 함께 할 수 있으나, 프로그램 소스코드 작성은 스스로 해야 한다.
  - ✓ 다른 사람의 과제를 copy 한 경우, 두 사람 모두 0점 처리하며 **학점상의 불이익이 있다**. 인터넷 등에서 찾은 소스 코드를 그대로 copy 한 경우에도 0점처리한다. **두 번 이상 적발되면 F 학점**을 받는다.