System Programming Report

Assignment 3-3 – Advanced Web Server

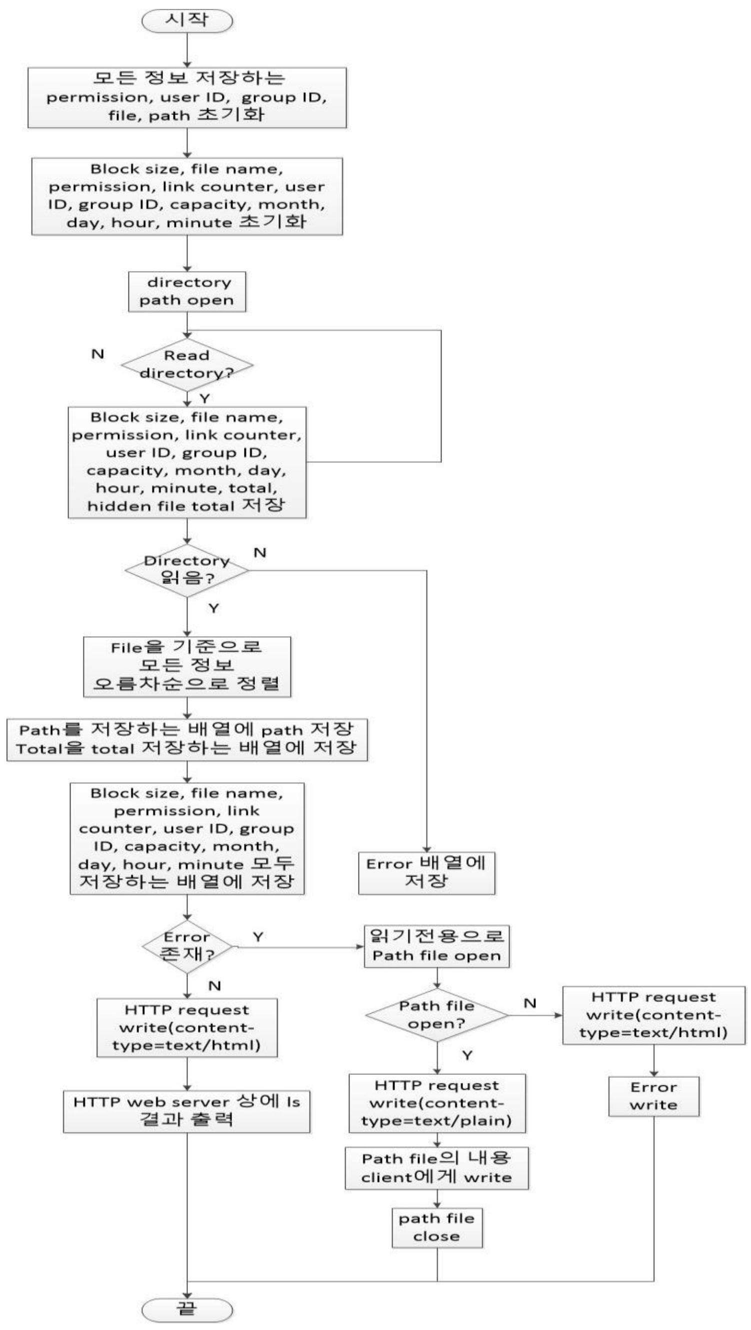
Professor	황호영 교수님
Department	Computer engineering
Student ID	2014722057
Name	김 진아
Class	설계 (화6 목4) / 실습 (금 56)
Date	2016. 5. 20

◆ Introduction

이번 과제는 다중 접속과 접근 제어를 지원하는 웹 서버 프로그램을 구현하는 것이다. 다중 접속을 하기 위해 fork함수를 이용해 child process을 만들어 여기서 ls의 결과가 출력하도록 한다. 또한 좀비 process를 방지하기 위해 wait함수, signal함수, alarm함수, exit함수를 사용하다. 접근 제어를 하기 위해 접근할 수 있는 서버의 주소가 저장된 파일을 읽어와 fnmatch함수를 통해 비교를 해 이에 따라 접근할 수 있도록 구현한다.

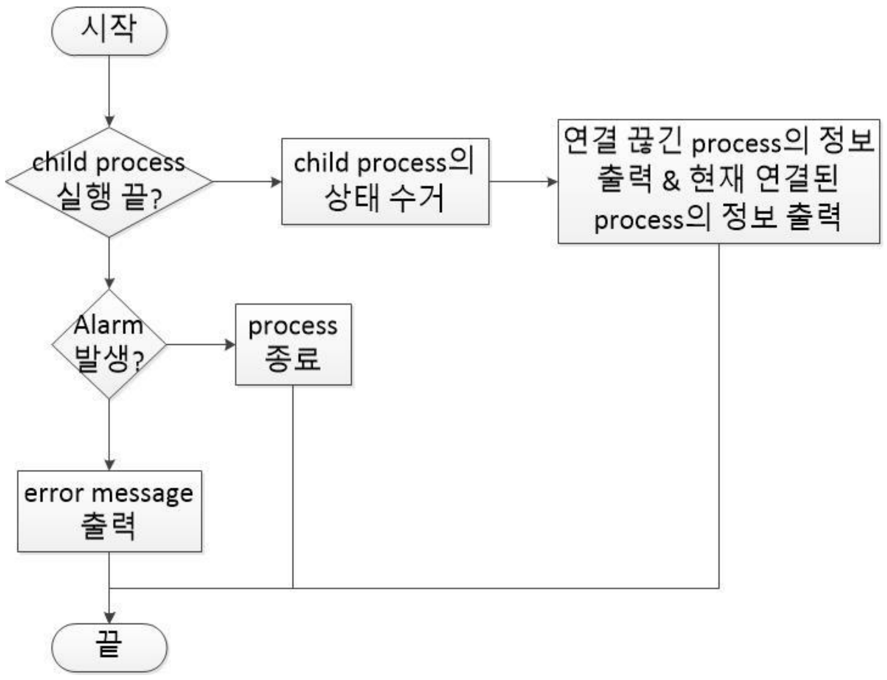
◆ Flowchart

- ls_print 함수



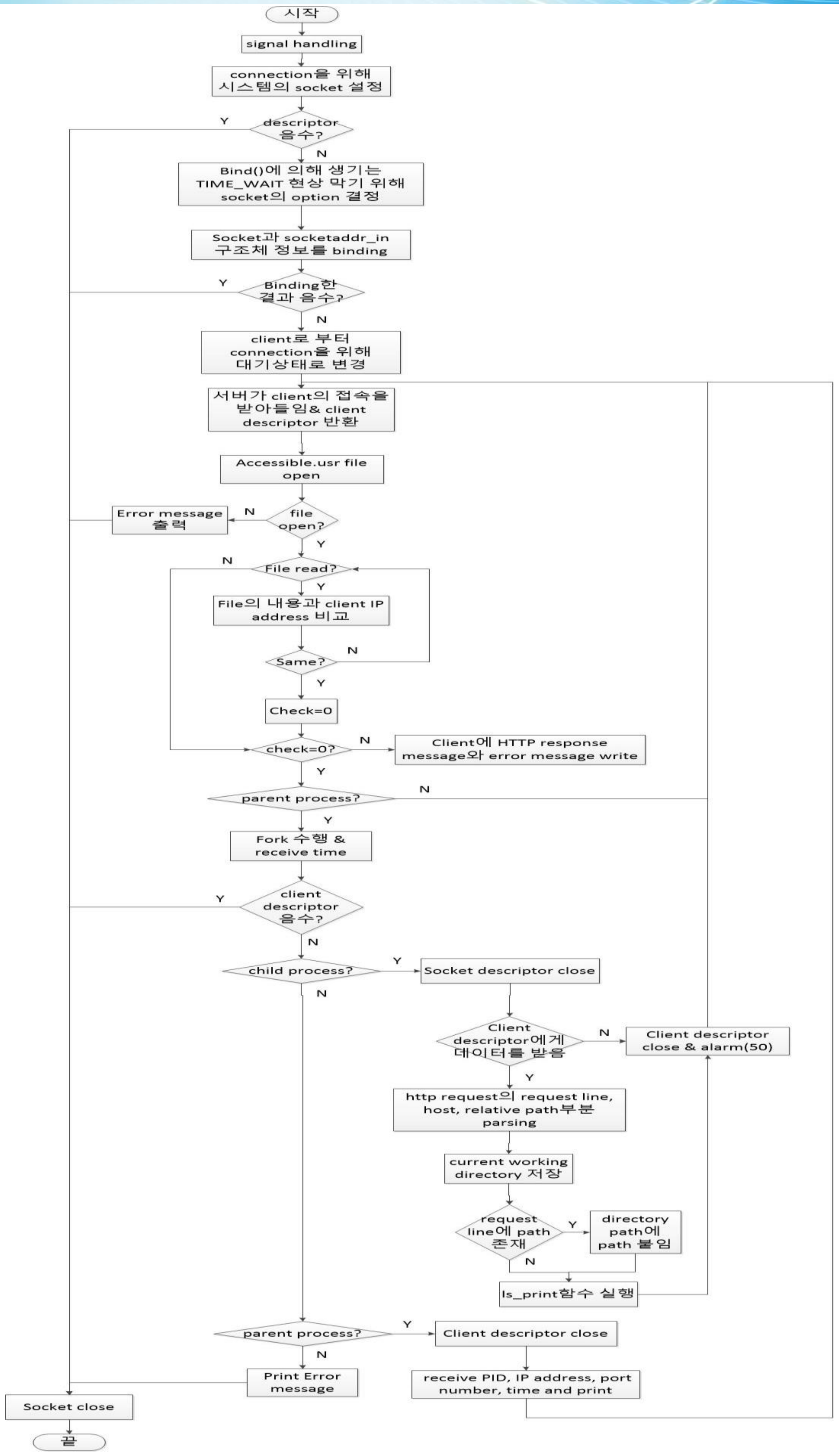
시작
모든 정보 저장하는 permission, user ID, group ID, file, path 초기화
Block size, file name, permission, link counter, user ID, group ID, capacity, month, day, hour, minute 초기화
directory path open
Read directory?
N
Y
Block size, file name, permission, link counter, user ID, group ID, capacity, month, day, hour, minute, total, hidden file total 저장
Directory 읽음?
N
Y
File을 기준으로 모든 정보 오름차순으로 정렬
Path를 저장하는 배열에 path 저장 Total을 total 저장하는 배열에 저장
Block size, file name, permission, link counter, user ID, group ID, capacity, month, day, hour, minute 모두 저장하는 배열에 저장
Error 배열에 저장
Error 존재?
Y
N
읽기전용으로 Path file open
HTTP request write(content-type=text/html)
Path file open?
N
Y
HTTP request write(content-type=text/html)
HTTP web server 상에 ls 결과 출력
HTTP request write(content-type=text/plain)
Error write
Path file의 내용 client에게 write
path file close
끝

- signal handling 함수



시작
child process 실행 끝?
child process의 상태 수거
연결 끊긴 process의 정보 출력 & 현재 연결된 process의 정보 출력
Alarm 발생?
process 종료
error message 출력
끝

- main 함수


시작
signal handling
connection을 위해 시스템의 socket 설정
descriptor 음수?
Y
N
Bind()에 의해 생기는 TIME_WAIT 현상 막기 위해 socket의 option 결정
Socket과 socketaddr_in 구조체 정보를 binding
Binding한 결과 음수?
Y
N
client로 부터 connection을 위해 대기상태로 변경
서버가 client의 접속을 받아들임& client descriptor 반환
Accessible.usr file open
Error message 출력
N
file open?
Y
N
File read?
Y
File의 내용과 client IP address 비교
N
Same?
Y
Check=0
check=0?
N
Client에 HTTP response message와 error message write
Y
parent process?
N
Y
Fork 수행 & receive time
Y
client descriptor 음수?
N
child process?
Y
Socket descriptor close
N
Client descriptor에게 데이터를 받음
N
Client descriptor close & alarm(50)
Y
http request의 request line, host, relative path부분 parsing
current working directory 저장
request line에 path 존재
Y
directory path에 path 붙임
N
ls_print함수 실행
parent process?
Y
Client descriptor close
N
Print Error message
receive PID, IP address, port number, time and print
Socket close
끝

◆ **Pseudo code**

- ls_print 함수

```
open directory
if directory can read
        while(read information in opened directory){
                receive file name, permission, link counter, user ID, group ID, capacity, month, day, hour, minute, the number of 1K blocks, total
        }
}
if directory read
        for(k1=0;k1<index1;k1++){
                for(k2=0;k2<index1-1;k2++){
                        receive two files’ name
                        if files’ name are same
                                continue for statement
                        calculate files' length
                        receive letters of files’ name while two letter are different
                        if first file's character > second file's character
                                change file's position, permission's position, linkcounter's position, user ID's position, group ID's position, capacity's position, month's position, day's position, hour's position, minute's position, block's position
                }
        }
        save directory path and total
        save beginning information
        for(i=0;i<index1;i++,s_index++)
```

```
                    save file, permission, user ID, group ID, block size, linkcounter, capacity, month,
day, hour, minute
                save ending information
        }
        if directory path doesn't exist
                save current path into error array
        close directory
        if error exists{
                path file open for read only
                if path file doesn't open{
                        write HTTP response message at client descriptor(content type is text/html)
                        write error message at client descriptor
                }
                if path file opens{
                        write HTTP response message at client descriptor(content type is text/plain)
                        read path file's content and write file's content at client descriptor
                        close path file
                }
                end of function
        }
        write HTTP response message at client descriptor(content type is text/html)
        write title and head at client descriptor
        for(i=0;i<s_index;i++)
                write result of 'ls –al' at client descriptor
```

- **signal handling 함수**

```
if child process is done
```

```
        PID is child process's status by using wait function
        for(i=0;i<index_info;i++){
            if ith pid is PID{
                receive current time
                print disconnected client's information(IP address, port number
                for(j=i;j<index_info;j++){
                    ith arrays(IP, port number, pid, time) has next array's data
                }
                decrease process count
                print process count and pid, port number, time
                stop for statement
            }
        }
    }
    if alarm operates
        exit process
    if others
        print default signal message
```

- main 함수

```
    signal handling
    create a socket
    if socket doesn't create{
        print "Server: Can't open stream socket."
        end of program
    }
    receive address family, IPv4 address, port number
```

```
use setsockopt function to block bind error
associate an address with a socket
if socket doesn’t bind{
    print “Server: Can't bind local address.
    end of program
}
announce that server is willing to accept connect request
while(1){
    save client_address's size into len
    accept a connect request from client
    accessible.usr file open
    if file doesn't exist{
        print no file message
        end of program
    }
    if file exists{
        while(read file){
            for(i=0;f_str[i]!='₩0';i++){  // change new line character to null
                if new line character exists{
                    change new line character to null
                    stop for statement
                }
            }
            compare client’s IP address and IP address in file
            if client’s IP address and IP address in file are same{
                stop for statement
                }
            }
            close file
        }
        if client’s IP address and IP address in file are different{
            write HTTP response message and error message at client descriptor(content
type is text/html)
            close client descriptor
            continue
        }
        if PID is parent process{
            make child process
            receive time
            if it isn't accept{
                print "Server: accept failed.
                end of program
            }
        }
        if others
            continue
        if it cannot make child process{
            print error message
            end of program
        }
        if PID is child process{
            if it can read HTTP request message{
                    close socket descriptor
                    if favicon.ico message operates{
                              close client file descriptor
                              continue
                    }
                    initialize host, version, temp
                    write HTTP request message
                    find GET / HTTP/1.1 in HTTP request message
                    find Host in HTTP request message
                    find temp in HTTP request message
                    if temp's last letter is '/'
                              temp's last letter is NULL
                    get current working directory path
                    if exist relative path
                              add relative path to current working directory path
                    go to ls_print function
          }
          use alarm function to stop process after 50 seconds
          close client descriptor
}
if PID is parent process{
          close client descriptor
          get PID, IP address, port number, time and print new client's information
          print process count and pid, port number, time
          increase process counter
}
```

```
    }

    close socket descriptor
```

◆ **Reference**

이번 과제에서 fork을 사용해 다중 process를 만들어야 하는데 이때 child process가 종료될 때 종료 상태를 wait 함수로 수거하지 못해 좀비 process가 발생했다. 그래서 처음에는 exit함수를 이용해 종료시켰더니 발생하지 않았다. 하지만 이렇게 하면 다중 process가 안 되므로 강의자료를 보면서 signal함수와 alarm을 사용하여 좀비 process가 발생하지 못하도록 구현하였다. 이때 alarm의 변수를 50으로 하여 충분히 다중 process가 발생하도록 하였다. 또한 child process는 parent process에서 생성해야 되는데 이를 구분하지 않아 child process에서도 child process를 생성하도록 했다. 그래서 처음에 우선 부모인지 아닌지 확인하여 부모일 때만 child process가 발생하도록 하였다.