KWANGWOON UNIVERSITY

2022년 1학기 **시스템프로그래밍실습** 10주차

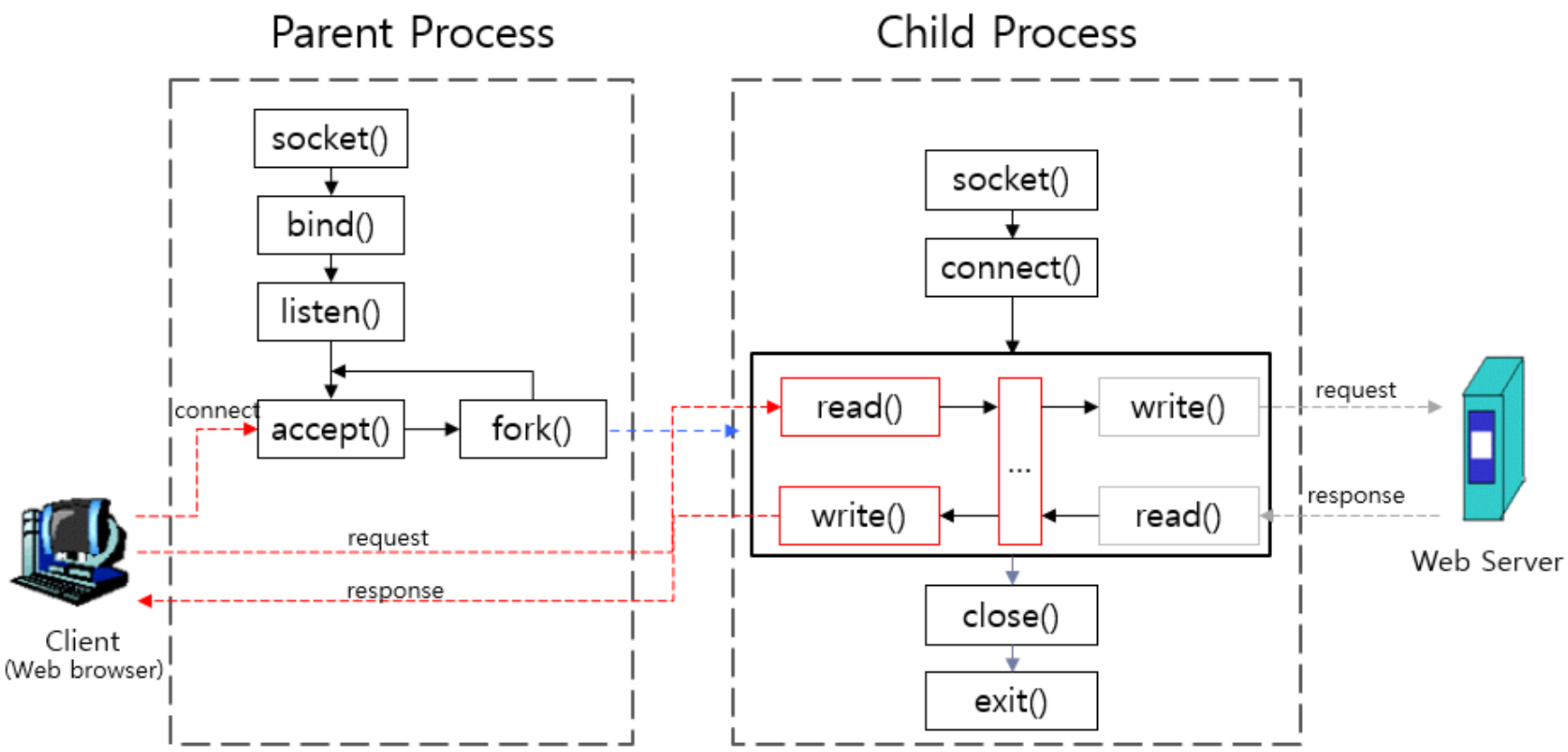
Construction Proxy Connection

System Software Laboratory
College of Software and Convergence
Kwangwoon Univ.

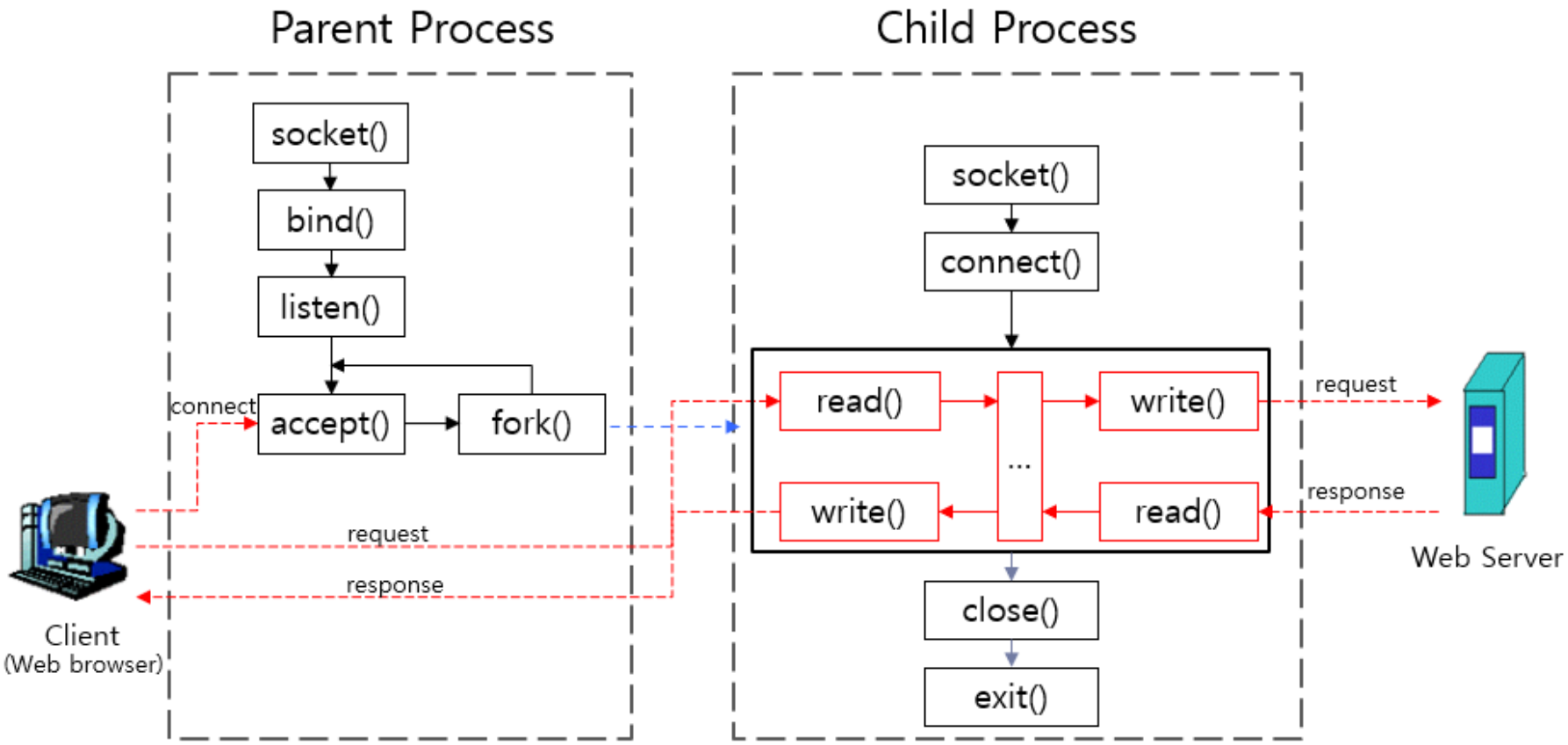
2st Assignment's Descriptions

- **Assignment 2-1**
  - Implement server/client
- **Assignment 2-2**
  - HTTP request handling in proxy server
- **Assignment 2-3 & 2-4**
  - Forward HTTP request to web server and signal handling

Previous Assignment



Proxy Server의 동작



gethostbyname() Function

```
#include <netdb.h>
extern int h_errno;

struct hostent *gethostbyname (const char *name);
```

- **Host name으로부터 network host entry를 가져옴**
- **Return**
  - hostent 구조체 if OK, NULL on error
- **Parameter**
  - Name
    - Host name

hostent Structure

```
struct hostent{
        char *h_name;           /* 호스트의 공식 이름 */
        char **h_aliases;       /* 별칭 리스트 */
        int h_addrtype;         /* 호스트의 주소 타입 */
        int h_length;           /* 주소의 길이 */
        char **h_addr_list;     /* 주소 리스트 */
}
#define h_addr h_addr_list[0]   /* 구 버전과의 호환 */
```

실습1. hostent Example (1/2)

```
#include <unistd.h>
#include <netinet/in.h>
#include <arpa/inet.h>
#include <string.h>
#include <stdio.h>
#include <netdb.h>

char *getIPAddr(char *addr);

int main()
{
    int fd;
    int len;
    char* host = "www.google.com";
    char* IPAddr;

    printf("* Hostname : %s₩n",host);
    IPAddr = getIPAddr(host);
    printf("* IP address : %s₩n",IPAddr);
    return 1;
}
```

Text-URL → dotted IPv4 address로 변환 (ex."www.google.com" → "142.250.199.100")

실습1. hostent Example (2/2)

```
char *getIPAddr(char *addr)
{
    struct hostent* hent;
    char * haddr;
    int len = strlen(addr);

    if ( (hent = (struct  hostent*)gethostbyname(addr)) != NULL)
    {
        haddr=inet_ntoa(*((struct in_addr*)hent->h_addr_list[0]));
    }
    return haddr;
}
```

```
sslab@ubuntu:~$ ./a.out
* Hostname : www.google.com
* IP address : 142.250.199.100
sslab@ubuntu:~$
```

32bit big-endian IP address를 dotted decimal string으로 변환

hostent 구조체의 IP 주소를 network byte order (32bit big-endian)으로 변환

Signal handling (1/2)

- **Signals**
  - Software interrupt
  - provide a way of handling asynchronous events
    - users press certain terminal key
    - hardware exceptions
    - kill() function
    - software conditions
  - When a signal occurs
    - ignore the signal
    - catch the signal
    - let the default action apply

Signal handling (2/2)

- **SIGCHLD**
  - change in status of child
  - default action : ignore
  - normally one of the wait() functions is called to fetch the child's process ID and termination status

- **SIGALRM**
  - time out
  - default action : abnormal terminate
  - normally alarm() function is used

- **SIGINT**
  - users press certain terminal key (ctrl+c, ^C)
  - Use to shutdown the server
  - default action : abnormal terminate

signal() function

```
#include <signal.h>

void (*signal(int signo, void (*func) (int))) (int) ;
```

- **특정 시그널에 대한 새로운 시그널 핸들러 설치**

- **Returns**
  - previous disposition of signal (see following) if OK, SIG_ERR on error

- **Parameters**
  - signo
    - 시그널의 이름
    - Ex) SIGALRM, SIGCHLD, SIGINT, SIGFPE
  - void (*func)(int)
    - 시그널 처리 함수, signal handler 또는 signal-catching function이라고 한다

alarm() function

```
#include <unistd.h>

unsigned int alarm(unsigned int seconds);
```

- **Returns**
  - 이전에 설정된 alarm이 signal을 전달할 때까지 남은 시간을 초 단위로 반환
  - 이전에 설정된 alarm이 없을 경우 0 return

- **Parameters**
  - seconds
    - seconds 초 후에 프로세스에 SIGALRM을 전달

실습2. Signal Example : SIGINT

```
#include <sys/types.h>
#include <sys/wait.h>
#include <signal.h>
#include <stdio.h>
#include <string.h>
#include <stdlib.h>

static void sig_int(int);

int main()
{
        char buf[255];
        pid_t pid;
        int     status;

        if (signal(SIGINT, sig_int) == SIG_ERR)
                fprintf(stderr, "signal error");

        printf("%% ");
        while (strcmp(fgets(buf, 255, stdin),"q₩n"))
        {
                buf[strlen(buf) - 1] = 0;
                if ( (pid = fork()) < 0 )
                        fprintf(stderr, "fork error");
                else if (pid == 0) {    /* child */
                        printf("%s₩n", buf);
                        sleep(3);
                        exit(0);
                }
                /* parent */
                if ( (pid = waitpid(pid, &status, 0)) < 0 )
                        fprintf(stderr, "waitpid error₩n");
                printf("%% ");
        }
        exit(0);
}

void sig_int(int signo)
{
        printf("interrupt₩n");
}
```

```
sslab@ubuntu:~$ ./a.out
% aaaa
aaaa
% bbbb
bbbb
% ^Cinterrupt
```

KWANGWOON UNIVERSITY

2022년 1학기 **시스템프로그래밍실습**

Proxy #2-3

System Software Laboratory
College of Software and Convergence
Kwangwoon Univ.

Proxy #2-3 (1/3)

- **Preview**
  - **Multiple process**
    - **Parent process**
      - Web browser 로부터의 연결 대기
      - 프록시 서버가 Web browser와 연결되면, **fork()**를 이용하여 새로운 child process를 생성한 후, 웹 브라우저로부터 연결을 대기
    - **Child process**
      - Web browser로부터 HTTP request 받음
      - Request header로부터 web server의 주소 추출
      - 추출 된 URL을 이용한 HIT/MISS 판별
        - HIT
          - 저장되어 있는 데이터를 web browse에 전송
        - MISS
          - Web browser로부터 받은 HTTP request를 web server에 전달
          - Web server로부터 HTTP response 수신
          - Web browser에 HTTP response 전달, Cache file에 response data 저장
      - Child process 종료

Proxy #2-3 (2/3)

- **Requirements**
  - **signal()** 함수를 사용하여 SIGCHLD, SIGALRM 시그널 처리
    - Web server에 **HTTP** request 전송 후 10초 동안 HTTP response를 받지 못할 경우 화면에 **"응답 없음" 메시지 출력 후 child process 종료**
    - 10초 이전에 Web server로부터 response를 받으면 alarm 해제
    - HTTP request 10초 동안 받지 못하는 방법
      - **인터넷 연결이 원활하지 않을 때**
        - ex) 프로그램 실행 후 컴퓨터의 인터넷을 끊기 / 가상머신에서 인터넷 연결 끊기
      - **"응답 없음" 메시지**
        - 터미널에 출력
        - 메시지 형식 자율
        - [예시]

```
=========No Response=========
```

Report Requirements

- **Ubuntu 16.04.5 Desktop 64bits 환경에서 채점**
- **보고서 구성**
  - **보고서 표지**
    - 수업 명, 과제 이름, 담당 교수님, 학번, 이름, 강의 시간 필히 명시
      - 과제 이름 ➔ Proxy 2-3
      - 강의 시간 ➔ 목요일 7,8교시인 경우 : 목7,8
  - 아래의 내용은 보고서에 필히 포함
    - Introduction
      - 과제 소개 – 4줄 이상(background 제외) 작성
    - Flow Chart
      - 코드 작성 순서도
    - Pseudo code
      - 알고리즘
    - 결과화면
      - 수행한 내용을 캡처 및 설명
    - 고찰
      - 과제를 수행하면서 느낀점 작성
    - Reference
      - 과제를 수행하면서 참고한 내용을 구체적으로 기록
      - 강의자료만 이용한 경우 생략 가능

Report Requirements

- **소프트카피만 작성**
- **제출 파일**
  - **보고서(.pdf 파일) + Source Code + Makefile**
  - 보고서 이름:
    - **실습번호_학번_수업시간.pdf**
    - 목 7,8 → 2-3_2022722000_thu78.pdf
    - 금 1,2 → 2-3_2022722000_fri12.pdf
    - 금 5,6 → 2-3_2022722000_fri56.pdf
  - C 파일 명:
    - *.h, *.c (자유롭게 구성 가능)
    - **Comment 작성**
  - Makefile:
    - 실행파일명: **proxy_cache**
    - 실행파일명 지정한 이름 외 다른 명으로 작성 시 감점

- **위 파일들을 압축해서 제출 (파일명:실습번호_학번_수업시간.tar.gz)**
  - 목 7,8 → 2-3_2022722000_thu78.tar.gz
  - 금 1,2 → 2-3_2022722000_fri12.tar.gz
  - 금 5,6 → 2-3_2022722000_fri56.tar.gz

  - **tar 압축 방법:**
    - 압축 시 : tar –zcvf [압축 파일명].tar.gz [폴더 명]
    - 해제 시 : tar -zxvf 파일명.tar.gz

  - 컴파일은 무조건 Makefile(makefile)을 이용한 make로 함.
    - Makefile(makefile) 없거나 실행 불가시 0점
    - 파일 압축 오류 시, 0점 처리

Report Requirements

- **과제 제출**
  - On-line 제출
    - **KLAS - 강의 과제 제출**
    - **2022년 5월 18일 23:59:59 까지**
      - **딜레이 받지 않음**
        - 제출 마감 시간 내 미제출시 해당과제 **0점 처리**
      - 교내 서버 문제 발생시, 메일로 과제 제출 허용

- **수업시간 외 과제 질문**
  - 수업시간 외 과제 질문은 **"강의 묻고 답하기"** 게시판을 통해 진행
  - **과제 제출 마감날 전날까지 업로드 된 질문에만 답변**

- **이론 과목에 과제 제출 시**
  - 간단한 txt 파일로 제출

    Ex.) 실습수업 때 과제 제출했습니다.

  - **이론 과목에 간단한 txt 파일 미 제출 시 감점**