실습 12주차

CPS LAB

소켓 프로그래밍

목차

1. 예제풀이

2. 연습문제

3. 실습#

#include <netdb.h>

void sethostent (int stayopen);

Description:
POSIX requires the gethostent() call, which should return the next entry in the host database

if stayopen is true (1), that a connected TCP socket should be used for the name server querie and that the connection should remain open during successive queries.
Otherwise, name server queries will use UDP datagrams

예제풀이

예제 1

/etc/hosts
호스트명 읽어 오기

운영체제에 따라, 필요한 네트워크 라이브러리를 포함하지 않을 수 있음
- gcc p_1.c –o p_1.out -lnsl

```c
#include <stdio.h>
#include <netdb.h>

int main(void) {

        struct hostent *hent;

        // UDP
        sethostent(0);

        // 각 entry(구조체 내용)의 멤버인 h_name 출력
        while( (hent = gethostent()) != NULL )
                printf("Name = %s\n", hent->h_name);

        return 0;
}
```

```
(base) kyhooon@kyh:~/sysprogram_practice/p_socket$ gcc p_1.c -g -o p_1.out
(base) kyhooon@kyh:~/sysprogram_practice/p_socket$
(base) kyhooon@kyh:~/sysprogram_practice/p_socket$ cat /etc/hosts
127.0.0.1       localhost
127.0.1.1       kyh

# The following lines are desirable for IPv6 capable hosts
::1     ip6-localhost ip6-loopback
fe00::0 ip6-localnet
ff00::0 ip6-mcastprefix
ff02::1 ip6-allnodes
ff02::2 ip6-allrouters
(base) kyhooon@kyh:~/sysprogram_practice/p_socket$
(base) kyhooon@kyh:~/sysprogram_practice/p_socket$ ./p_1.out
Name = localhost
Name = kyh
Name = ip6-localhost
(base) kyhooon@kyh:~/sysprogram_practice/p_socket$
```

예제풀이

예제 2

getservent() 함수로 포트 정보 읽기

#include <netdb.h>

struct servent *getservent(void);

reads the next entry from the services database and returns a servent structure

Return Values:

NULL if an error occurs or the end of the file is reached

/etc/services

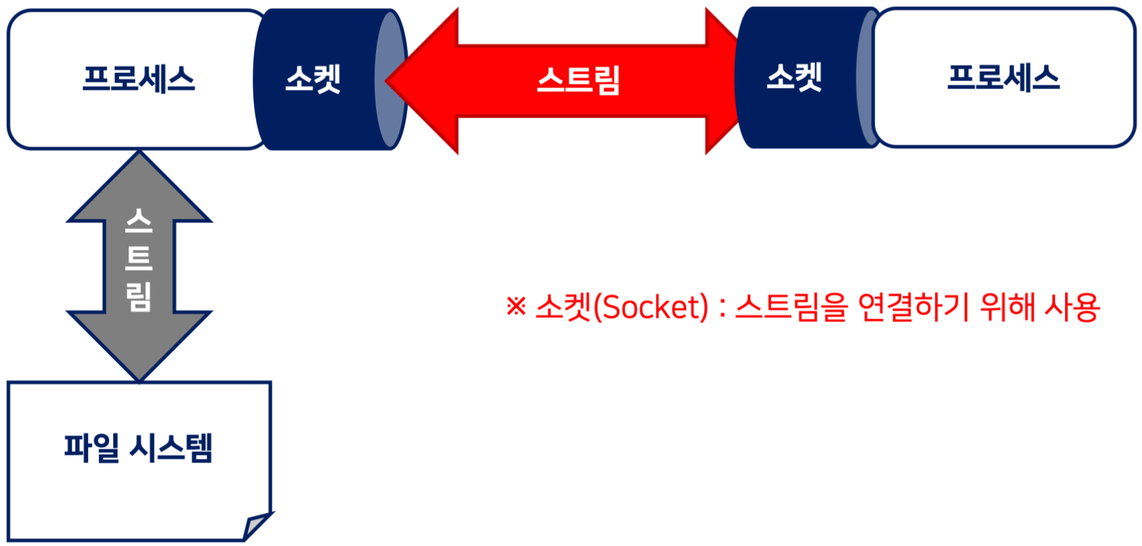
```
# port number for both TCP and UDP; hence, officially ports have two entries
# even if the protocol doesn't support UDP operations.
#
# Updated from https://www.iana.org/assignments/service-names-port-numbers/service-names-port-numbers.xhtml .
#
# New ports will be added on request if they have been officially assigned
# by IANA and used in the real-world or are needed by a debian package.
# If you need a huge list of used numbers please install the nmap package.

tcpmux          1/tcp                           # TCP port service multiplexer
echo            7/tcp
echo            7/udp
discard         9/tcp           sink null
```

```c
#include <netdb.h>
#include <stdio.h>

int main(void) {

        struct servent *port;

        int i;

        for( i = 0; i < 5; i++ ) {
                port = getservent();
                printf("Name = %s, Port = %d\n",
                        port->s_name, port->s_port);
        }

        endservent();

        return 0;
}
```

```
(base) kyhooon@kyh:~/sysprogram_practice/p_socket$ ./p_2.out
Name = tcpmux, Port = 256
Name = echo, Port = 1792
Name = echo, Port = 1792
Name = discard, Port = 2304
Name = discard, Port = 2304
(base) kyhooon@kyh:~/sysprogram_practice/p_socket$
```

소켓

- 소켓(socket)는 네트워크 상에서 서로 다른 시스템 간에 통신을 가능하게 해 줌
- 프로세스 간에 데이터를 주고받기 위한 인터페이스로 사용되며, IP 주소와 포트 번호를 기반으로 연결을 수립



※ 소켓(Socket) : 스트림을 연결하기 위해 사용

소켓(Socket) 의 종류

- AF_UNIX : 유닉스 도메인 소켓
  - 같은 호스트에서 프로세스 사이에 통신할 때 사용
- AF_INET : 인터넷 소켓
  - 인터넷을 통해 다른 호스트와 통신할 때 사용

소켓 API 정리

클라이언트 측 소켓 API

- socket(2)
- connect(2)

서버 측 소켓 API

- socket(2)
- bind(2)
- listen(2)
- accept(2)

소켓 API 정리

클라이언트 측 소켓 API

socket (2)

connect (2)

```
#include <sys/types.h>
#include <sys/socket.h>

int socket(int domain, int type, int protocol);
```

[소켓 종류]
AF_UNIX
AF_INET (IPv4)
AF_INET6 (IPv6)

[통신 방식]
SOCK_STREAM (TCP)
SOCK_DGRAM (UDP)

- 소켓에서 이용할 프로토콜
- 보통 "0" 지정 : 시스템(OS)이 프로토콜 값을 지정해 줌

소켓 API 정리

클라이언트 측 소켓 API

socket (2)

connect (2)

```
#include <sys/types.h>
#include <sys/socket.h>

int connect(int sockfd, const struct sockaddr *addr, socklen_t addrlen);
```

소켓 함수가 생성한 socket descriptor

- 접속하려는 서버의 IP 주소
- open() 함수의 *path 와 비슷

*addr 의 크기를 지정

소켓 API 정리

서버 측 소켓 API

socket(2)

bind(2)

listen(2)

accept(2)

```
#include <sys/socket.h>
#include <sys/types.h>

int bind(int sockfd, struct sockaddr *addr, socklen_t addrlen);
```

sockfd에 대응되는 소켓에 접속을 기다리는 주소 addr을 할당

소켓 API 정리

서버 측 소켓 API

socket(2)

bind(2)

listen(2)

accept(2)

```
#include <sys/socket.h>

int listen(int sockfd, int backlog);
```

동시에 받아들일 수 있는 커넥션 최대 수

sockfd에 대응되는 소켓이 접속을 기다리는 서버용 소켓임을 커널에 알림
(sockfd에 대응되는 소켓이 passive 소켓임을 marking)

소켓 API 정리

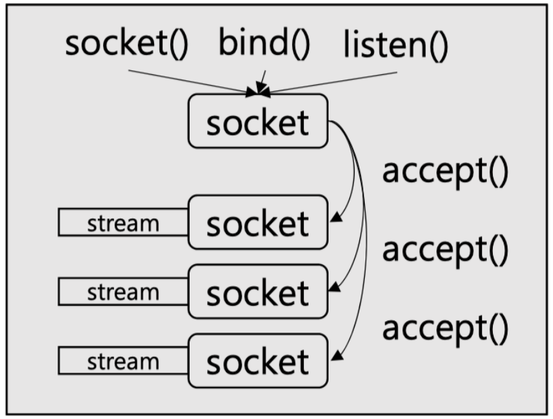
서버 측 소켓 API

socket(2)

bind(2)

listen(2)

accept(2)




```
#include <sys/socket.h>
#include <sys/types.h>

int accept(int sockfd, struct sockaddr *addr, socklen_t *addrlen);
```

클라이언트의 주소 정보

- sockfd에 대응되는 listening 소켓으로부터 클라이언트의 연결 요청을 처리함
- 연결 시 새로운 소켓을 생성하고 파일 디스크립터를 반환

예제풀이

예제 3

소켓 API로 두 프로세스 통신하기 (서버)

```c
#include <sys/socket.h>
#include <sys/un.h>
#include <unistd.h>
#include <stdlib.h>
#include <stdio.h>
#include <string.h>

#define SOCKET_NAME "kyh_socket"

int main(void) {

        char buf[256];
        struct sockaddr_un ser, cli;
        // sd: 서버가 클라이언트 요청을 기다리는 식별자
        // nsd: sd가 클라이언트 요청을 받았으며, 새 sd를 생성함
        int sd, nsd;
        int len, nlen;

        if( ( sd = socket(AF_UNIX, SOCK_STREAM, 0)) == -1 ) {
                perror("socket");
                exit(1);
        }

        memset((char*)&ser, 0, sizeof(struct sockaddr_un));
        ser.sun_family = AF_UNIX;
        strcpy(ser.sun_path, SOCKET_NAME);
        len = sizeof(ser.sun_family) + strlen(ser.sun_path);

        if( bind(sd, (struct sockaddr *)&ser, len) ) {
                perror("bind");
        }

        if( listen(sd, 5) < 0 ) {
                perror("listen");
                exit(1);
        }

        printf("Waiting..\n");
        if( (nsd = accept(sd, (struct sockaddr *)&cli, &nlen)) == -1 ) {
                perror("accept");
                exit(1);
        }
```

Error:
"bind: bind: Address already in use"

Reason:
비정상적으로 종류(shutdown .etc)
빈번 주소 binding 경우

Solution:
unlink [정의된 소켓 이름]

```c
        printf("Waiting..\n");
        if( (nsd = accept(sd, (struct sockaddr *)&cli, &nlen)) == -1 ) {
                perror("accept");
                exit(1);
        }

        if( recv(nsd, buf, sizeof(buf), 0) == -1 ) {
                perror("recv");
                exit(1);
        }

        printf("Recevied Message: %s\n", buf);
        // 주의:
        // nsd 먼저 지워야 함
        close(nsd);
        close(sd);

        return 0;
}
```

예제풀이

예제 3

소켓 API로 두 프로세스 통신하기 (클라이언트)

```c
#include <stdio.h>
#include <sys/un.h>
#include <sys/socket.h>
#include <sys/types.h>
#include <stdlib.h>
#include <unistd.h>
#include <stdio.h>
#include <string.h>

#define SOCKET_NAME "kyh_socket"

int main(void) {

        int sd, len;
        char buf[256];
        struct sockaddr_un ser;

        if( (sd = socket(AF_UNIX, SOCK_STREAM, 0)) == -1) {
                perror("socket");
                exit(1);
        }

        memset((char *)&ser, '\0', sizeof(ser));
        ser.sun_family = AF_UNIX;
        strcpy(ser.sun_path, SOCKET_NAME);
        len = sizeof(ser.sun_family) + strlen(ser.sun_path);

        if( connect(sd, (struct sockaddr *)&ser, len) < 0 ) {
                perror("connect");
                exit(1);
        }

        strcpy(buf, "Unix Domain Socket Test Message");
        if( send(sd, buf, sizeof(buf), 0) == -1 ) {
                perror("send");
                exit(1);
        }

        close(sd);

        return 0;
}
```

예제풀이

예제 3

소켓 API로 두 프로세스 통신하기 (실행)

```
(base) kyhooon@kyh:~/sysprogram_practice/p_socket$ ./p_3_server.out
Waiting..
Recevied Message: Unix Domain Socket Test Message
(base) kyhooon@kyh:~/sysprogram_practice/p_socket$
```

주의: 서버 먼저 실행
이유는?

```
(base) kyhooon@kyh:~/sysprogram_practice/p_socket$ ./p_3_client.out
(base) kyhooon@kyh:~/sysprogram_practice/p_socket$
```

예제풀이

예제 4

소켓 API로 지정된 서버 통신하기 (서버) - Demo

```c
#include <stdio.h>
#include <stdlib.h>
#include <string.h>
#include <sys/socket.h>
#include <arpa/inet.h>
#include <sys/un.h>
#include <unistd.h>

#define PORTNUM 7500

int main(void) {

        char buf[256];
        struct sockaddr_in ser, cli;
        int sd, nsd;
        int len = sizeof(cli);

        if( (sd = socket(AF_INET, SOCK_STREAM, 0)) == -1 ) {
                perror("socket");
                exit(1);
        }

        memset((char *)&ser, '\0', sizeof(ser));
        ser.sin_family = AF_INET;
        ser.sin_port = htons(PORTNUM);
        // 서버 주소 설정해 주기
        // 여기서 192.168.0.1로 가정
        ser.sin_addr.s_addr = inet_addr("192.168.0.1");

        if( bind(sd, (struct sockaddr *)&ser, sizeof(ser)) ) {
                perror("bind");
                exit(1);
        }

        if( listen(sd, 5) ) {
                perror("listen");
                exit(1);
        }

        if( (nsd = accept(sd, (struct sockaddr *)&cli, &len)) ) {
                perror("accept");
                exit(1);
        }
```

주의:

서버 주소에 접근 권한, 주소 맞는지 등 체크해야 함

```c
        if( (nsd = accept(sd, (struct sockaddr *)&cli, &len)) ) {
                perror("accept");
                exit(1);
        }

        // cli 구조체 받은 클라이언트 주소 출력하기
        sprintf(buf, "Your IP address is %s\n", inet_ntoa(cli.sin_addr));
        if( send(nsd, buf, strlen(buf) + 1, 0) == -1 ) {
                perror("send");
                exit(1);
        }

        close(nsd);
        close(sd);

        return 0;
}
```

예제풀이

예제 4

소켓 API로 지정된 서버 통신하기 (클라이언트)

Demo

주의:

서버 주소에 접근 권한, 주소 맞는지 등 체크해야 함

```c
#include <stdio.h>
#include <stdlib.h>
#include <unistd.h>
#include <sys/un.h>
#include <sys/socket.h>
#include <arpa/inet.h>
#include <string.h>

// 포트 지정
#define PORTNUM 7500

int main(void) {

        int sd;
        char buf[256];
        struct sockaddr_in cli;

        if( (sd = socket(AF_INET, SOCK_STREAM, 0)) == -1 ) {
                perror("socket");
                exit(1);
        }

        memset((char *)&cli, '\0', sizeof(cli));
        cli.sin_family = AF_INET;
        cli.sin_port = htons(PORTNUM);
        cli.sin_addr.s_addr = inet_addr("192.168.147.129");

        if( connect(sd, (struct sockaddr *)&cli, sizeof(cli)) ) {
                perror("connect");
                exit(1);
        }

        if( recv(sd, buf, sizeof(buf), 0) == -1 ) {
                perror("recv");
                exit(1);
        }
        close(sd);

        return 0;
}
```

예제풀이

예제 5

무한 접속 받을 수 있는 서버 구현하기 (서버) - Demo

```c
#include <stdio.h>
#include <stdlib.h>
#include <unistd.h>
#include <string.h>
#include <sys/socket.h>
#include <netinet/in.h>
#include <arpa/inet.h>

#define PORTNUM 7500

int main(void) {

        char buf[256];
        struct sockaddr_in ser, cli;
        int sd, nsd;
        int len = sizeof(cli);

        memset((char *)&ser, '\0', sizeof(ser));
        ser.sin_family = AF_INET;
        ser.sin_port = htons(PORTNUM);
        ser.sin_addr.s_addr = inet_addr("192.168.0.1");

        if( (sd = socket(AF_INET, SOCK_STREAM, 0)) == -1 ) {
                perror("socket");
                exit(1);
        }

        if( bind(sd, (struct sockaddr *)&ser, sizeof(ser)) ) {
                perror("bind");
                exit(1);
        }

        if( listen(sd, 5) ) {
                perror("listen");
                exit(1);
        }

        while(1) {

                if( (nsd = accept(sd, (struct sockaddr *)&cli, &len)) == -1 ) {
                        perror("accept");
                        exit(1);
                }

                sprintf(buf, "%s", inet_ntoa(cli.sin_addr));
                printf("*** Send a Message to Client(%s)\n", buf);

                strcpy(buf, "Welcome to Network Server !");
                if( send(nsd, buf, strlen(buf) + 1, 0) == -1 ) {
                        perror("send");
                        exit(1);
                }

                if( recv(nsd, buf, sizeof(buf), 0) == -1 ) {
                        perror("recv");
                        exit(1);
                }

                printf("** From Client : %s\n", buf);
                close(nsd);
        }
        close(sd);

        return 0;
}
```

예제풀이

예제 6

UDP 프로그래밍 (서버) - Demo

```c
#include <stdio.h>
#include <stdlib.h>
#include <unistd.h>
#include <string.h>
#include <sys/socket.h>
#include <netinet/in.h>
#include <arpa/inet.h>

#define PORTNUM 7500

int main(void) {

        char buf[256];
        struct sockaddr_in ser, cli;
        int sd;
        int len = sizeof(cli);

        if( (sd = socket(AF_INET, SOCK_DGRAM, 0)) == -1 ) {
                perror("socket");
                exit(1);
        }

        memset((char *)&ser, '\0', sizeof(ser));
        ser.sin_family = AF_INET;
        ser.sin_port = htons(PORTNUM);
        ser.sin_addr.s_addr = inet_addr("192.168.0.1");

        if( bind(sd, (struct sockaddr *)&ser, sizeof(ser)) ) {
                perror("ind");
                exit(1);
        }

```

질문:

왜 새로운 소켓 받아들이지 않을까?

```c
        if( bind(sd, (struct sockaddr *)&ser, sizeof(ser)) ) {
                perror("ind");
                exit(1);
        }

        while(1) {
                if( (recvfrom(sd, buf, 255, 0, (struct sockaddr *)&cli, &len)) == -1 ) {
                        perror("recvfrom");
                        exit(1);
                }
                printf("** From Client: %s\n", buf);
                strcpy(buf, "Hello Client");
                if( (sendto(sd, buf, strlen(buf) + 1, 0, (struct sockaddr *)&cli, sizeof(cli))) == -1 ) {
                        perror("sendto");
                        exit(1);
                }
        }

        return 0;
```

연습 문제

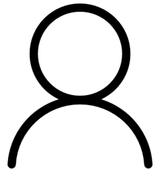
실습#

프로그램
은행에서 사용하는 보안 카드와 유사하게,
클라이언트가 접속하면 서버는 임의의 번호를 전송하고
클라이언트에서 이에 해당하는 보안 번호를 입력하는 프로그램을 작성하시오!

(22p 그림으로 해설하기)

실습#

그림으로 해설



사용자

은행 서버

데이터 베이스
(user)

pwnum[5] – 저장된 보안번호

"은행 어플 열기" socket()

connect()

socket()

bind()

listen()

accept()

Connection Establishment

"다른 사용자 접근 방해하지 않기 위해
새 socket(nsd)를 생성.."

"이체하려고 하니까,
보안 번호 줘 !"

recv()

srand((unsigned int)time(NULL)로
랜덤 번호를 전송함

send()

"사용자에게서 입력한
보안번호를 기다림"

recv()

"은행 어플에서 받아온
랜덤번호를 확인해,
보안번호를 입력한다"

fgets()

"사용자 보낸 보안번호가 맞는지,
pwnum에서 iteration하여 확인한다"

send()

send()

"성공적으로 찾으면, 사용자에게,
Valid Security Number이라는 메시지 보냄"

"계좌이체 성공 !"

"어플 닫기" ...

... "다른 사용자 접근 기다림 ..."

감사합니다.

CPS LAB