2주차_UDP_ 소켓프로그래밍

데이터 네트워크연구실
이현호
leeo75@cs-cnu.org

Goals

- C언어의 기본, 컴파일
- UDP 통신 과정
- UDP 소켓 프로그래밍 (ECHO SERVER)

GCC

- GNU 컴파일러 모음(GNU Compiler Collection, 줄여서 GCC)
- GCC는 원래 C만을 지원했던 컴파일러

Back to the basic

- hello.c
- 생성 : vi hello.c
- 저장 : wq

```c
#include <stdio.h>

void main(){
    printf("Hello World Again!\n");
}
```

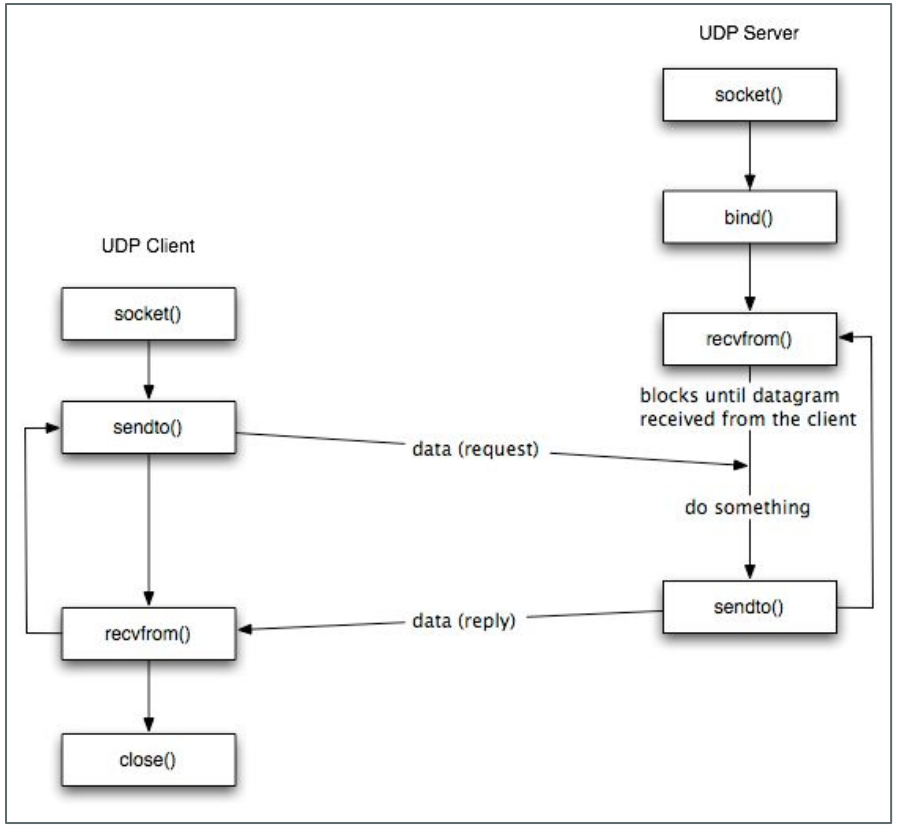
Back to the basic

- 컴파일 : gcc filename
- 옵션 : -o outfile 이름 설정
- man gcc로 다양한 옵션 확인 가능

```
hyunholee@DNLAB:~/temp/UDP_socket$ gcc hello.c -o hello
hyunholee@DNLAB:~/temp/UDP_socket$ ./hello
Hello World Again!
```

UDP 통신 과정

- UDP 서버에서 소켓 생성
- socket() -> bind()의 과정으로 연결
- recvfrom() 으로 데이터를 읽어옴
- sendto()로 데이터를 전송



UDP_socket 예제

- 클라이언트에서 2개의 숫자를 전송
- 서버에서 계산해서 결과를 전송 받음
- 결과 출력

Server

서버

- 선언부
- 필요한 라이브라리 Include

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

#define BUFSIZE 1024
```

서버

- 에러 처리 부분

```
/*
 * error - wrapper for perror
 */
void error(char *msg) {
  perror(msg);
  exit(1);
}
```

서버

- Main 변수 선언 부분
- 서버 구성에 필요한 변수 선언

```
int sockfd; /* socket */
int portno; /* port to listen on */
int clientlen; /* byte size of client's address */
struct sockaddr_in serveraddr; /* server's addr */
struct sockaddr_in clientaddr; /* client addr */
struct hostent *hostp; /* client host info */
char buf[BUFSIZE]; /* message buf */
char *hostaddrp; /* dotted decimal host addr string */
int optval; /* flag value for setsockopt */
int n; /* message byte size */
```

서버

- 소켓 생성 부분
- 주소 체계 (AF_INET)
- DataGram : UDP 소켓

```
/*
 * socket: create the parent socket
 */
sockfd = socket(AF_INET, SOCK_DGRAM, 0);
if (sockfd < 0)
  error("ERROR opening socket");
```

서버

- 주소 체계 구성하기
- 포트번호와 주소 입력 후 바인딩 하기

```
/*
 * build the server's Internet address
 */
bzero((char *) &serveraddr, sizeof(serveraddr)); /*buffer initalizer*/
serveraddr.sin_family = AF_INET;
serveraddr.sin_addr.s_addr = htonl(INADDR_ANY);
serveraddr.sin_port = htons((unsigned short)portno);

/*
 * bind: associate the parent socket with a port
 */
if (bind(sockfd, (struct sockaddr *) &serveraddr,
         sizeof(serveraddr)) < 0)
  error("ERROR on binding");
```

서버

- 메일 루프의 시작
- recvfrom() : 데이터를 클라이언트로 부터 받음

```
/*
 * main loop: wait for a datagram, then echo it
 */
clientlen = sizeof(clientaddr);
while (1) {

  /*
   * recvfrom: receive a UDP datagram from a client
   */
  bzero(buf, BUFSIZE);
  n = recvfrom(sockfd, buf, BUFSIZE, 0,
               (struct sockaddr *) &clientaddr, &clientlen);
  if (n < 0)
    error("ERROR in recvfrom");
```

서버

- 누구로 부터 데이터를 받았는지 데이터 추출

```
/*
 * gethostbyaddr: determine who sent the datagram
 */
hostp = gethostbyaddr((const char *)&clientaddr.sin_addr.s_addr,
                      sizeof(clientaddr.sin_addr.s_addr), AF_INET);
if (hostp == NULL)
  error("ERROR on gethostbyaddr");
hostaddrp = inet_ntoa(clientaddr.sin_addr);
if (hostaddrp == NULL)
  error("ERROR on inet_ntoa\n");
printf("server received datagram from %s (%s)\n",
       hostp->h_name, hostaddrp);
printf("server received %d/%d bytes: %s\n", strlen(buf), n, buf);
```

서버

- 받은 메시지를 보낸 상대에게 보내줌
- 루프의 끝

```
    n = sendto(sockfd, buf, strlen(buf), 0,
               (struct sockaddr *) &clientaddr, clientlen);
    if (n < 0)
      error("ERROR in sendto");
  }
```

```c
/*
 * udpserver.c - A simple UDP echo server
 * usage: udpserver <port>
 */

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

#define BUFSIZE 1024

/*
 * error - wrapper for perror
 */
```

```
21 void error(char *msg) {
22   perror(msg);
23   exit(1);
24 }
25
26 int main(int argc, char **argv) {
27   int sockfd; /* socket */
28   int portno; /* port to listen on */
29   int clientlen; /* byte size of client's address */
30   struct sockaddr_in serveraddr; /* server's addr */
31   struct sockaddr_in clientaddr; /* client addr */
32   struct hostent *hostp; /* client host info */
33   char buf[BUFSIZE]; /* message buf */
34   char *hostaddrp; /* dotted decimal host addr string */
35   int optval; /* flag value for setsockopt */
36   int n; /* message byte size */
37
38   /*
39    * check command line arguments
40    */
```

```
41   if (argc != 2) {
42     fprintf(stderr, "usage: %s <port>\n", argv[0]);
43     exit(1);
44   }
45   portno = atoi(argv[1]);
46
47   /*
48    * socket: create the parent socket
49    */
50   sockfd = socket(AF_INET, SOCK_DGRAM, 0);
51   if (sockfd < 0)
52     error("ERROR opening socket");
53
54   /* setsockopt: Handy debugging trick that lets
55    * us rerun the server immediately after we kill it;
56    * otherwise we have to wait about 20 secs.
57    * Eliminates "ERROR on binding: Address already in use" error.
58    */
59   optval = 1;
60   setsockopt(sockfd, SOL_SOCKET, SO_REUSEADDR,
```

```
61          (const void *)&optval , sizeof(int));
62
63  /*
64   * build the server's Internet address
65   */
66  bzero((char *) &serveraddr, sizeof(serveraddr));
67  serveraddr.sin_family = AF_INET;
68  serveraddr.sin_addr.s_addr = htonl(INADDR_ANY);
69  serveraddr.sin_port = htons((unsigned short)portno);
70
71  /*
72   * bind: associate the parent socket with a port
73   */
74  if (bind(sockfd, (struct sockaddr *) &serveraddr,
75           sizeof(serveraddr)) < 0)
76    error("ERROR on binding");
77
78  /*
79   * main loop: wait for a datagram, then echo it
80   */
```

```
81    clientlen = sizeof(clientaddr);
82    while (1) {
83
84      /*
85       * recvfrom: receive a UDP datagram from a client
86       */
87      bzero(buf, BUFSIZE);
88      n = recvfrom(sockfd, buf, BUFSIZE, 0,
89                   (struct sockaddr *) &clientaddr, &clientlen);
90      if (n < 0)
91        error("ERROR in recvfrom");
92
93      /*
94       * gethostbyaddr: determine who sent the datagram
95       */
96      hostp = gethostbyaddr((const char *)&clientaddr.sin_addr.s_addr,
97                            sizeof(clientaddr.sin_addr.s_addr), AF_INET);
98      if (hostp == NULL)
99        error("ERROR on gethostbyaddr");
100     hostaddrp = inet_ntoa(clientaddr.sin_addr);
```

```
101     if (hostaddrp == NULL)
102       error("ERROR on inet_ntoa\n");
103     printf("server received datagram from %s (%s)\n",
104            hostp->h_name, hostaddrp);
105     printf("server received %d/%d bytes: %s\n", strlen(buf), n, buf);
106
107     /*
108      * sendto: echo the input back to the client
109      */
110     n = sendto(sockfd, buf, strlen(buf), 0,
111                (struct sockaddr *) &clientaddr, clientlen);
112     if (n < 0)
113       error("ERROR in sendto");
114   }
115 }
```

Client

클라이언트

- 도입부
- 필요한 라이브러리 Include

```
/*
 * udpclient.c - A simple UDP client
 * usage: udpclient <host> <port>
 */
#include <stdio.h>
#include <stdlib.h>
#include <string.h>
#include <unistd.h>
#include <sys/types.h>
#include <sys/socket.h>
#include <netinet/in.h>
#include <netdb.h>

#define BUFSIZE 1024
```

클라이언트

- 에러 처리부분

```
/*
 * error - wrapper for perror
 */
void error(char *msg) {
    perror(msg);
    exit(0);
}
```

클라이언트

- 필요변수 선언부분
- 변수의 갯수를 세어주는 부분

```
int main(int argc, char **argv) {
    int sockfd, portno, n;
    int serverlen;
    struct sockaddr_in serveraddr;
    struct hostent *server;
    char *hostname;
    char buf[BUFSIZE];

    /* check command line arguments */
    if (argc != 3) {
       fprintf(stderr,"usage: %s <hostname> <port>\n", argv[0]);
       exit(0);
    }
    hostname = argv[1];
    portno = atoi(argv[2]);
```

클라이언트

- UDP 소켓을 생성하는 부분
- host 이름을 받아오는 부분

```
/* socket: create the socket */
sockfd = socket(AF_INET, SOCK_DGRAM, 0);
if (sockfd < 0)
    error("ERROR opening socket");

/* gethostbyname: get the server's DNS entry */
server = gethostbyname(hostname);
if (server == NULL) {
    fprintf(stderr,"ERROR, no such host as %s\n", hostname);
    exit(0);
}
```

클라이언트

- 연결 할 서버의 주소를 만드는 부분

```
/* build the server's Internet address */
bzero((char *) &serveraddr, sizeof(serveraddr));
serveraddr.sin_family = AF_INET;
bcopy((char *)server->h_addr,
      (char *)&serveraddr.sin_addr.s_addr, server->h_length);
serveraddr.sin_port = htons(portno);
```

클라이언트

- 서버로 보낼 메시지
- 서버에서 보낸 메시지
- 출력 부분

```
/* get a message from the user */
bzero(buf, BUFSIZE);
printf("Please enter msg: ");
fgets(buf, BUFSIZE, stdin);

/* send the message to the server */
serverlen = sizeof(serveraddr);
n = sendto(sockfd, buf, strlen(buf), 0, &serveraddr, serverlen);
if (n < 0)
  error("ERROR in sendto");

/* print the server's reply */
n = recvfrom(sockfd, buf, strlen(buf), 0, &serveraddr, &serverlen);
if (n < 0)
  error("ERROR in recvfrom");
printf("Echo from server: %s", buf);
return 0;
}
```

```c
/*
 * udpclient.c - A simple UDP client
 * usage: udpclient <host> <port>
 */
#include <stdio.h>
#include <stdlib.h>
#include <string.h>
#include <unistd.h>
#include <sys/types.h>
#include <sys/socket.h>
#include <netinet/in.h>
#include <netdb.h>

#define BUFSIZE 1024

/*
 * error - wrapper for perror
 */
void error(char *msg) {
    perror(msg);
```

```c
    exit(0);
}

int main(int argc, char **argv) {
    int sockfd, portno, n;
    int serverlen;
    struct sockaddr_in serveraddr;
    struct hostent *server;
    char *hostname;
    char buf[BUFSIZE];

    /* check command line arguments */
    if (argc != 3) {
       fprintf(stderr,"usage: %s <hostname> <port>\n", argv[0]);
       exit(0);
    }
    hostname = argv[1];
    portno = atoi(argv[2]);

    /* socket: create the socket */
```

```
41      sockfd = socket(AF_INET, SOCK_DGRAM, 0);
42      if (sockfd < 0)
43          error("ERROR opening socket");
44
45      /* gethostbyname: get the server's DNS entry */
46      server = gethostbyname(hostname);
47      if (server == NULL) {
48          fprintf(stderr,"ERROR, no such host as %s\n", hostname);
49          exit(0);
50      }
51
52      /* build the server's Internet address */
53      bzero((char *) &serveraddr, sizeof(serveraddr));
54      serveraddr.sin_family = AF_INET;
55      bcopy((char *)server->h_addr,
56            (char *)&serveraddr.sin_addr.s_addr, server->h_length);
57      serveraddr.sin_port = htons(portno);
58
59      /* get a message from the user */
60      bzero(buf, BUFSIZE);
```

```
61      printf("Please enter msg: ");
62      fgets(buf, BUFSIZE, stdin);
63
64      /* send the message to the server */
65      serverlen = sizeof(serveraddr);
66      n = sendto(sockfd, buf, strlen(buf), 0, &serveraddr, serverlen);
67      if (n < 0)
68        error("ERROR in sendto");
69
70      /* print the server's reply */
71      n = recvfrom(sockfd, buf, strlen(buf), 0, &serveraddr, &serverlen);
72      if (n < 0)
73        error("ERROR in recvfrom");
74      printf("Echo from server: %s", buf);
75      return 0;
76  }
```

Result

결과 화면

- Client

```
hyunholee@DNLAB:~/temp/UDP_socket/origin_code$ ./client
usage: ./client <hostname> <port>
hyunholee@DNLAB:~/temp/UDP_socket/origin_code$ ./client 127.0.0.1 6292
Please enter msg: Hello world
Echo from server: Hello world
```

- Server

```
hyunholee@DNLAB:~/temp/UDP_socket/origin_code$ ./server
usage: ./server <port>
hyunholee@DNLAB:~/temp/UDP_socket/origin_code$ ./server 6292
Start to run server!
server received datagram from localhost (127.0.0.1)
server received 12/12 bytes: Hello world
```

Assignment

과제

- 두 숫자를 서버에 보내서 계산 결과 보내기
- Switch 를 사용하여 계산하는 프로그램
- 구조체를 전달하고 분석하는 프로그램

과제

- 구조체
- 변수와 연산자 결과를 저장

```
struct cal_data
{
        int left_num;
        int right_num;
        char op;
        int result;
        short int error;
};
```

과제

- 클라이언트 전송 부분

```
fgets(msg, MAXLEN-1, stdin);
if(strncmp(msg, "quit\n",5) == 0)
{
        break;
}
sscanf(msg, "%d%[^0-9]%d", &left_num, op, &right_num);
memset((void *)&sdata, 0x00, sizeof(sdata));
sdata.left_num = htonl(left_num);
sdata.right_num = htonl(right_num);
sdata.op = op[0];

addrlen = sizeof(addr);
sendto(sockfd, (void *)&sdata, sizeof(sdata), 0,
        (struct sockaddr *)&addr, addrlen);
```

과제

- 서버에서 전송 받는 부분

```
while(1)
{
        addrlen = sizeof(cliaddr);
        recvfrom(sockfd, (void *)&rdata, sizeof(rdata), 0,
                        (struct sockaddr *)&cliaddr, &addrlen);
EBUG
printf("Client Info : %s (%d)\n", inet_ntoa(cliaddr.sin_addr), ntohs(cliaddr.sin_port));
printf("Input : %d %c %d\n", ntohl(rdata.left_num), rdata.op, ntohl(rdata.right_num));
F

        left_num  = ntohl(rdata.left_num);
        right_num  = ntohl(rdata.right_num);
        switch(rdata.op)
        {
```

과제

- 서버에서 계산 해주는 부분

```
switch(rdata.op)
{
        case '+':
                cal_result = left_num + right_num;
                break;
        case '-':
                cal_result = left_num - right_num;
                break;
        case '*':
                cal_result = left_num * right_num;
                break;
        case '/':
                if(right_num == 0)
                {
                        rdata.error = htons(2);
                        break;
                }
                cal_result = left_num / right_num;
                break;
}
```

결과 화면

- Server

```
hyunholee@DNLAB:~/temp/UDP_socket$ ./server_cal
Start to run server!
Client Info : 127.0.0.1 (47293)
Input : 1 + 2
Result : 3
Client Info : 127.0.0.1 (47293)
Input : 3 * 3
Result : 9
Client Info : 127.0.0.1 (47293)
Input : 4 / 2
Result : 2
Client Info : 127.0.0.1 (47293)
Input : 5 - 7
Result : -2
```

결과 화면

- Client

```
hyunholee@DNLAB:~/temp/UDP_socket$ ./client_cal
Usage : ./client_cal [ipaddress]
hyunholee@DNLAB:~/temp/UDP_socket$ ./client_cal 127.0.0.1
> 1+2
1 + 2 = 3
> 3*3
3 * 3 = 9
> 4/2
4 / 2 = 2
> 5-7
5 - 7 = -2
```

과제

1. Linux 실습환경 구축한 후 결과화면
2. dissector.lua 내부에 구현되어있는 프로토콜 분석 레포트

과제 제출

- 과제 제출 기한:
  - 실습 하루 전 18시
- e-learing 페이지에 제출
- 보고서 제목 : NW_학번_이름_실습번호.pdf
- 추가 첨부파일 : NW_학번_이름_실습번호.zip
  - 추가 첨부파일은 본인이 작성한 파일로 제한합니다