

Socket

네트워크의 연결 도구

- Socket API 서로 다른/같은 호스트 상에서 실행되는 두개 이상의 프로세스들이 컴퓨터 네트워크를 사용하여 통신하기 위해 제공되는 도구
- 소켓과 입출력 파일 디스크립터를 이용해 소켓 입출력 가능 소켓 외의 입출력 - stdin/out, file, terminal...
- 파일 디스크립터 열린 파일들을 식별하기 위한 정보 (int)
- 기본 입출력 함수

1. open

```
open(path, oflag)
```

- oflag O_RDONLY, O_APPEND, O_CREATE, O_TRUNC.. 등

2. read

```
read(fd, buf, count)
```

fd에서 count만큼 buf에 읽는다

3. write

```
write(fd, buf, size)
```

fd에서 size만큼 buf에 쓴다

4. close

```
close(fd)
```

열려있는 fd 닫기

▶ 실습 01:io-syscall.c

```
1 #include <stdio.h>
2 #include <fcntl.h>
3 // #include <stdlib.h>
4
5 int main(int argc, char** argv) {
6     int fd, writeLen, readLen;
7     char rBuff[BUFSIZ];
```

```

8   if(argc != 2) {
9       fprintf(stderr, "Usage: %s [Filename] \n ",argv[0]);
10      return 0;
11  }
12
13  //fd=0: stdin
14  readLen = read(0, rBuff, BUFSIZ-1);
15  //readLen 리턴값 -> 읽은 만큼의 사이즈
16  if(readLen == -1)
17  {
18      fprintf(stderr, "Read Error \n");
19      return 0;
20  }
21
22  printf("Total reading data: %d\n", readLen);
23
24  //개행 추가
25  rBuff[readLen] = '\0';
26
27  //해당 파일을 read, right 권한으로 열고 / 없으면 새 파일 생성 / 있으면 초기화 후 다시
28  fd = open(argv[1],O_WRONLY|O_CREAT|O_TRUNC);
29  if(fd == -1)
30  {
31      fprintf(stderr, "Open Error \n");
32      return 0;
33  }
34
35  //null 포함 +1
36  writeLen = write(fd, rBuff, readLen+1);
37  if(writeLen == -1)
38  {
39      fprintf(stderr, "Write Error \n");
40      return 0;
41  }
42  printf("Total writing data: %d\n", readLen);
43
44  close(fd);
45  return 0;
46  }

```

Colored by Color

```

socket(int domain, type, protocol)
// 프로토콜 체계(IPv4/6), 전송방식 (SOCK_STREAM / SOCK_DGRAM), 프로토콜
(IPPROTO_TCP/UDP)

```

▶ 실습 02: socket-syscall.c

```

1  #include <sys/socket.h>
2  #include <stdio.h>
3  #include <fcntl.h>
4
5  int main()
6  {
7      int sD1, fD1, sD2, fD2;
8

```

CS

```
9 //IPv4, TCP
10 sD1 = socket(PF_INET, SOCK_STREAM, 0);
11 fD1 = open("test",O_RDONLY);
12 sD2 = socket(PF_INET, SOCK_STREAM, 0);
13 fD2 = open("test",O_RDONLY);
14
15 printf("sD1:%d, fD1:%d, sD2:%d, fD2:%d \n",sD1, fD1, sD2, fD2);
16
17 close(sD1);
18 close(fD1);
19 close(sD2);
20 close(fD1);
21
22 return 0;
23 }
24
```

Colored by Colon Scripter

통신 프로토콜

pid - 프로세스 식별할 수 있지만 Host 내에서만 가능함 port - 서로다른 호스트 사이에서 프로세스 식별, app과 전송 계층 사이에 존재

프로토콜 체계

PF_INET - Protocol Family AF_INET - Address Family

PF_INET = AF_INET //ipv4 PF_INET6 = AF_INET6 //ipv6

--> 요즘에는 동일하게 사용하는중..

Socket API

- 호스트 식별용 IP주소, 프로세스 식별용 port 필요가 필요함

IP

IPv4 - 32bit / IPv6 - 128bit

IP address : classful addressing

A class : 8
B class : 16
C class : 24

→ subnet?
 2^n (2)
 xx..0 → 도메인 주소
 xx..255 → 브로드캐스트 주소
 → 호스트 비트 전부 1

→ 지정된 값이기에 낭비 심할 때도

⇒ CIDR (Classless Inter Domain Routing) ex) 300개의 SubNet + 50개의 호스트

필요한 만큼만 subnet part로 할당.

$2^k 2300 \&\& 2^{16-k} 50$

xx..xx / n

$2^{32-k} - 2$: 호스트 주소 개수

172.25.171.182 / B class / 255.255.224.0

10101100.00011001.10101011.10101110

11111111.11111111.11110000.00000000

host 범위
 → 255.255.224.0

네트워크 대역 서브넷 → 255.255.224.0

The Subnet mask is not the default Class B subnet mask.
 The Subnet mask can also be represented in CIDR format.

IP address : how get?

- 수동 - 핸드코딩
- 동적 - DHCP

: 서버로부터 동적으로 IP 얻어냄.

class 단위 말고 CIDR로 비트단위로 할당함 10.10.1.32/27: 상위 27비트는 네트워크 주소

- Loopback 인터페이스 127.0.0.1 주소 - 호스트 사용 (localhost)

소켓 주소

- 구조체

```
sockaddr(AF_INET, port, internet address)
```

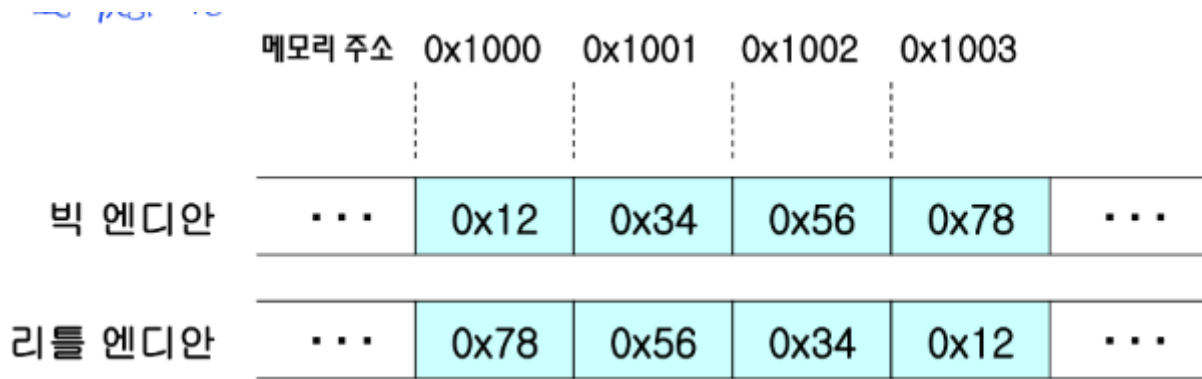
- bind 소켓에 주소를 설정하는 함수

```
bind (fd, const struct sockaddr *addr, len)
```

바이트 순서

메모리에 데이터를 저장할 때 바이트 순서

- 빅엔디언: 상위 바이트 값이 메모리의 작은 주소 (네트워크 바이트)
- 리틀 엔디언: 하위 바이트 값이 메모리의 작은 주소 (호스트 바이트, 보통 사용)



-> 바이트 순서를 고려하지 않으면 오류가 나도 메모리의 어디를 봐야할지 모르게 때문에..

- IP -> 빅엔디안 변경
- Prot -> 빅엔디안 변경
- app에서 주고받는 데이터 -> 빅엔디안 / 리틀엔디안 통일 필요

1. 엔디안 변경

```
htonl //host to network long
htons //host to network short
ntohs //network to host long
ntohl //network to host short
```

2. IPv4의 점 형식.. 192.168.137.1

```
inet_addr //IP to network
inet_network //IP to host
int inet_aton //IPv4 (196.168.137.2) -> network
char* inet_ntoa //network -> IPv4

//차이점
ip = 127.0.0.1
inet_addr(ip) = inet_aton(ip) 까지와 기능은 동일하지만
inet_addr은 변환해서 직접 넣어줘야하고
inet_aton은 파라미터에 소켓의 주소를 넣을 수 있음
```

3. IPv6를 지원하기 위해 새롭게 바이트 변환 함수..

```
inet_pton //ASCII string to numeric form
inet_ntop //network to string
```

▶ 실습 03:ip-uint.c

```
1 #include <netinet/in.h>
2 #include <stdio.h>
3 #include <arpa/inet.h>
```

```

4
5 int main(int argc, char** argv)
6 {
7     uint32_t ipInInt;
8     uint32_t ipInInt2;
9
10    char *ipInStr = "127.0.0.1";
11    //IPv4 to host
12    ipInInt = inet_network(ipInStr);
13
14    printf("String: %s, Decimal: %u, Hex: %X \n",
15          ipInStr, ipInInt, ipInInt);
16
17    //host(little) to network(big)
18    ipInInt = htonl(ipInInt);
19    printf("After htonl - Decimal: %u, Hex: %X \n",
20          ipInInt, ipInInt);
21
22    //IP to network
23    ipInInt2 = inet_addr(ipInStr);
24    printf("after inet_addr: %u, Hex: %x \n", ipInInt2, ipInInt2);
25
26    return 0;
27 }

```

Colored by Color Scripter

inet_network() + htonl() == inet_addr() // IP to host / host to network -> IP to network

▶ 실습 04: address-resolution.c

```

1  #include <stdio.h>
2  #include <sys/socket.h>
3  #include <netinet/in.h>
4  #include <arpa/inet.h>
5
6  int printAddr(struct sockaddr_in *);
7  int main(char argc, char** argv)
8  {
9      char *sampleIP = "127.0.0.1";
10     int port = 9002;
11
12     struct sockaddr_in sockAddr1, sockAddr2, sockAddr3;
13
14     sockAddr1.sin_family = AF_INET; //IPv4
15     sockAddr1.sin_addr.s_addr = inet_addr(sampleIP); //IP to network
16     sockAddr1.sin_port = htons(port); //host to network
17
18     sockAddr2.sin_family = AF_INET;
19     inet_aton(sampleIP, &(sockAddr2.sin_addr)); //IP to network 후 파라미터로 저장
20     sockAddr2.sin_port = htons(port);
21
22     sockAddr3.sin_family = AF_INET;
23     inet_pton(AF_INET, sampleIP, &(sockAddr3.sin_addr)); //IPv4를 선택하고 IP to network
24     sockAddr3.sin_port = htons(port);
25
26     //3개 모두 동일한 결과
27     printAddr(&sockAddr1);
28     printAddr(&sockAddr2);

```

```

29     printAddr(&sockAddr3);
30
31     //network to IPv4
32     printf("=====ntoa=====\\n");
33     printf("IP:%s \\n",inet_ntoa(sockAddr1.sin_addr));
34     printf("IP:%s \\n",inet_ntoa(sockAddr2.sin_addr));
35     printf("IP:%s \\n",inet_ntoa(sockAddr3.sin_addr));
36
37     return 0;
38 }
39 int printAddr(struct sockaddr_in *myAddr)
40 {
41     int port;
42     char txt[INET_ADDRSTRLEN];
43
44     //network to host (이미 myAddr에 network 형태로 저장되어있기 때문)
45     port = ntohs(myAddr->sin_port);
46
47     //network to IP
48     //IPv4, socket의 IP, 저장할 공간, 사이즈
49     inet_ntop(AF_INET,&((struct sockaddr_in *)myAddr)->sin_addr),
50         txt,sizeof(struct sockaddr_in));
51     printf("IP:%s, Port:%d \\n",txt, myAddr->sin_port);
52     return 0;
53 }
54

```

호스트 이름과 IP 주소

- gethostbyname() 호스트 이름을 이용해 IP주소 조회
- hostent 구조체 name, aliases, addrtype, len, addr_list; //도메인 이름, 별칭, AF_INET., address list

▶ 실습 05: gethostbyname.c

```

1  #include <stdio.h>
2  #include <netdb.h>
3  #include <errno.h>
4  #include <sys/types.h>
5  #include <sys/socket.h>
6  #include <netinet/in.h>
7  #include <stdlib.h>
8  #include <arpa/inet.h>
9  #include <string.h>
10 void errProc(const char *);
11 int main(int argc, char **argv)
12 {
13     //호스트 구조체
14     struct hostent *ent;
15     //주소 구조체
16     struct in_addr **res;
17     int i = 0;
18     if(argc != 2)
19     {
20         fprintf(stderr,"Usage: %s <hostname> \\n", argv[0]);
21         return -1;

```

```

22     }
23
24     //호스트 이름을 이용해 IP 주소 조회
25     ent = gethostbyname(argv[1]);
26     if(ent == NULL) errProc("gethostbyname");
27
28     //host의 addr_list 가져오기
29     res = (struct in_addr **)ent->h_addr_list;
30
31     //호스트 이름
32     printf("hostname: %s \n",ent->h_name);
33
34     //호스트의 addr_list 모두 출력
35     while(res[i] != NULL)
36     {
37         //network to IPv4
38         printf("%s ", inet_ntoa(*res[i]));
39         i++;
40     }
41     printf("\n");
42
43 }
44
45 void errProc(const char *str)
46 {
47     fprintf(stderr,"%s: %s\n",str,strerror(errno));
48     exit(errno);
49 }
50
51

```

Colored by Color Scripter

```

ssoxong@DESKTOP-1CG42BG:/mnt/c/users/leeso/NP
/week03$ ./gethostbyname www.naver.com
hostname: www.naver.com.nheos.com
223.130.195.200 223.130.200.107

```

해당 domain(host)에 접근할 수 있는 모든 ip 출력