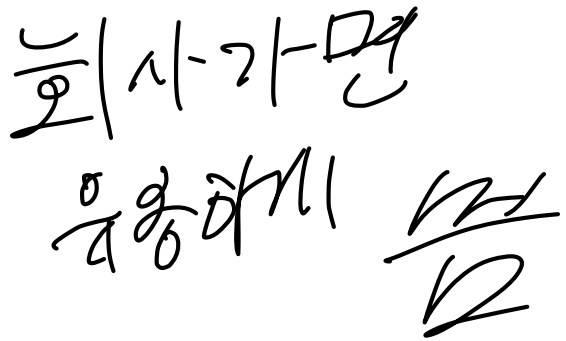
Chapter 14

멀티캐스트 & 브로드캐스트



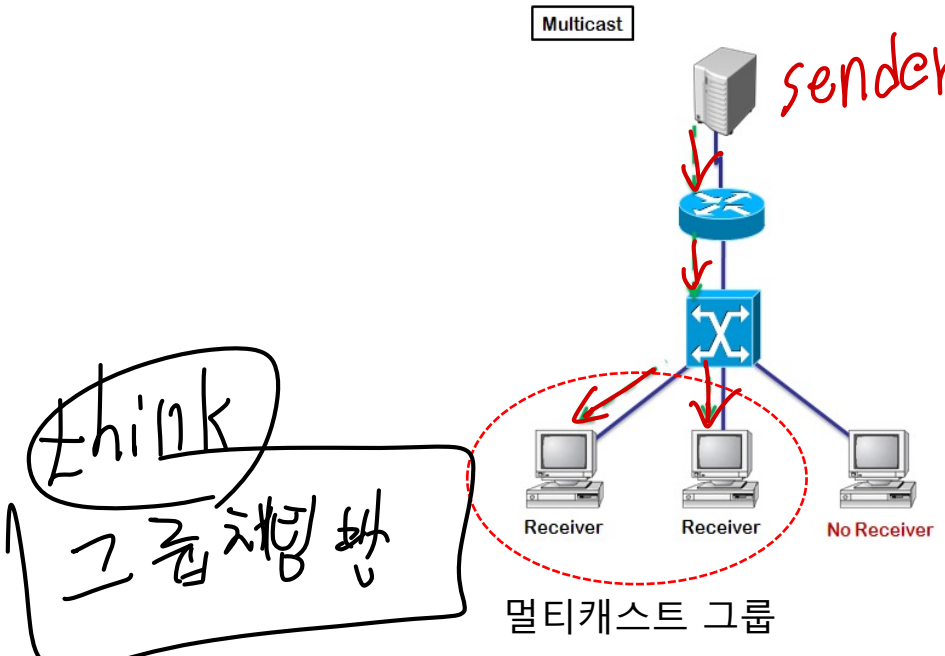
멀티캐스트의 데이터 전송 방식

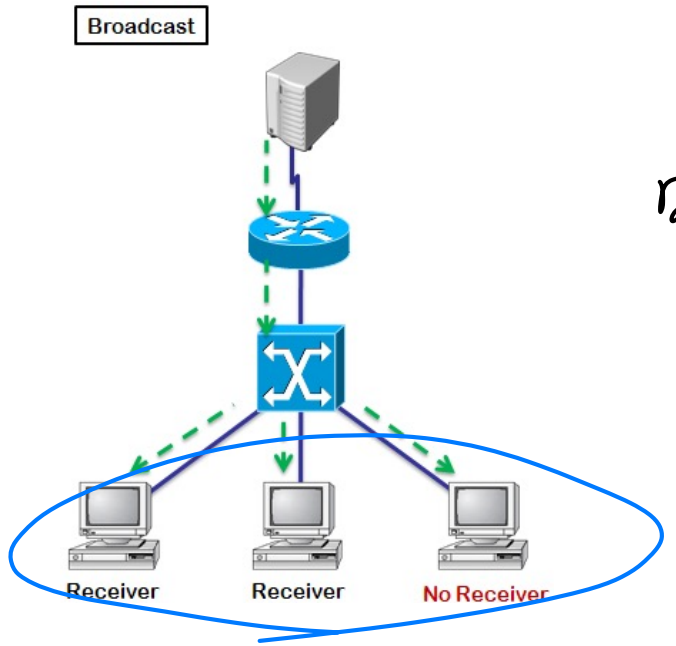
- 멀티캐스트 (Multicast)

  - 멀티캐스트 그룹에 참여하는 구성원들에게 데이터를 전송하는 기법
  - 하나의 데이터를 전송해서 모든 그룹 멤버들이 수신함 (모든 호스트에 전송하지 않음)
  - 멀티캐스트 데이터는 라우터에서 복사됨
  - 경로가 동일하면, 한 번의 데이터 전송으로 여러 호스트에 데이터를 전달할 수 있음

- 멀티캐스트 주소

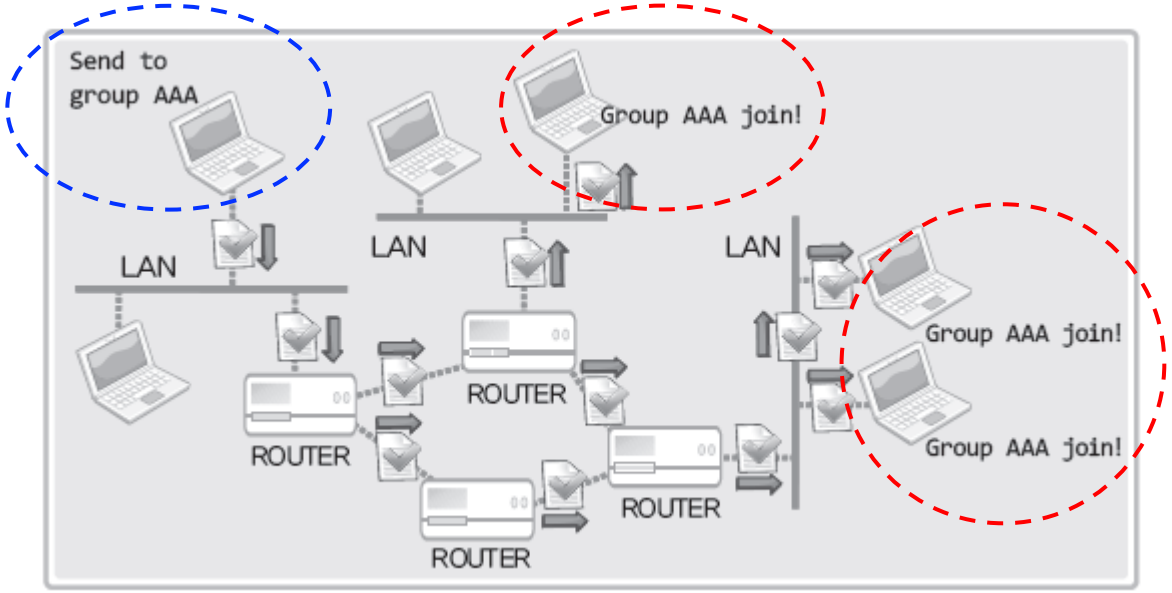
  - 224.0.0.0 ~ 239.255.255.255 범위를 갖는 Class D IP 주소를 사용함





멀티캐스트의 데이터 전송방식과 트래픽 이점

- 멀티캐스트 서버는 특정 멀티캐스트 그룹을 대상으로 데이터를 딱 한번 전송함
- 한번 전송하더라도 그룹에 속하는 클라이언트는 모두 데이터를 수신함
- 멀티캐스트 그룹의 수는 IP주소 범위 내에서 얼마든지 추가가 가능함
- 특정 멀티캐스트 그룹으로 전송되는 데이터를 수신하려면 해당 그룹에 가입하면 됨



멀티캐스트 데이터를 수신하기 위해서는 **그룹 가입의 절차**를 걸쳐야 한다.

모든 호스트에 데이터 전송을 해야 할 때, TCP 또는 UDP보다 발생하는 트래픽의 양이 적다

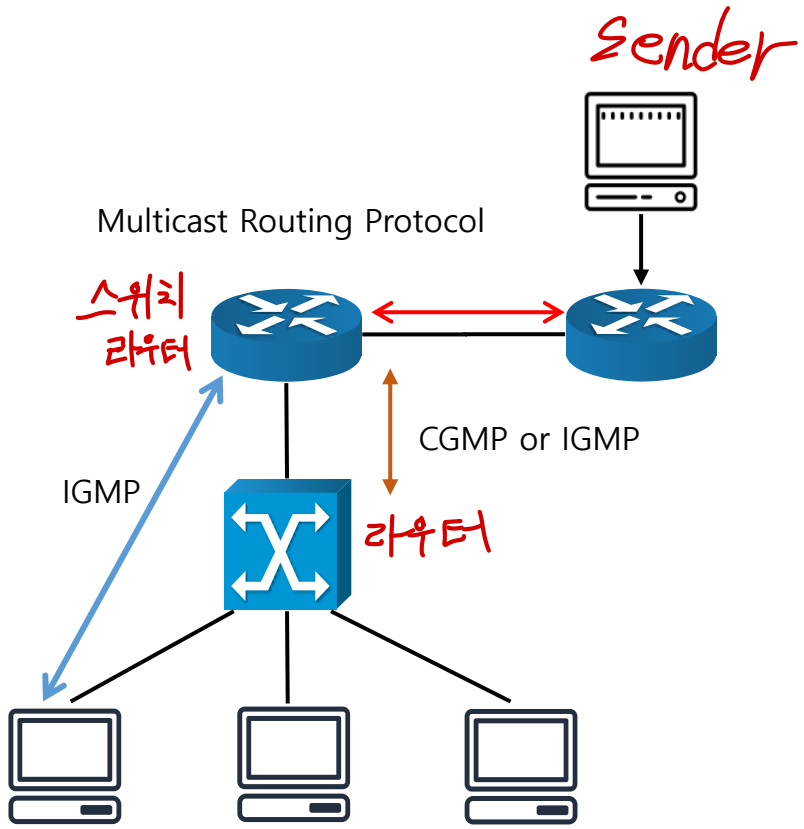
- 멀티캐스트는 연결의 개념이 존재하지 않음
- 따라서 UDP 소켓을 기반으로 전송됨
- 라우터들이 패킷을 복사해서 다수의 호스트에게 전달함

멀티캐스트 과정

- IGMP (Internet Group Management Protocol)
  - 호스트와 라우터 간의 multicast 정보를 교환
  - 데이터 전송 용도가 아닌 그룹 관리용 신호 프로토콜
  - Membership Query (라우터 -> 호스트)
  - Membership Report (호스트 -> 라우터)
  - Leave Report (호스트 -> 라우터)

- IGMP, CGMP(Cisco Group Management Protocol)
  - 라우터와 스위치 간 multicast 정보 교환

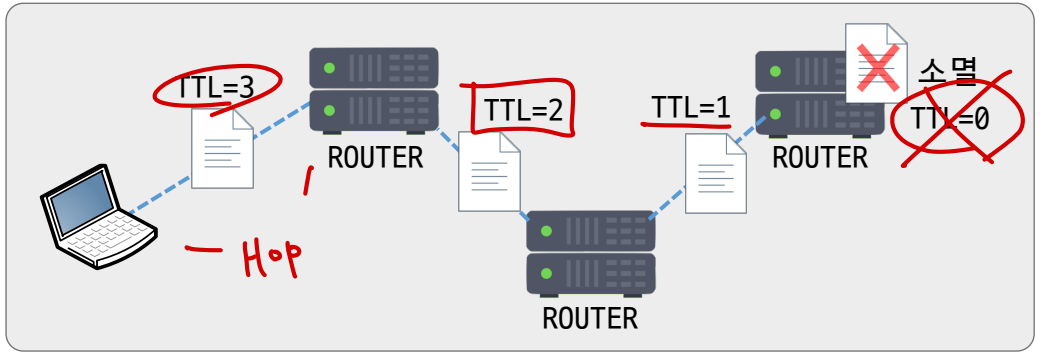
- Multicast Routing Protocol
  - 라우터와 라우터 간 multicast 정보 교환



라우팅과 TTL: 데이터 전송 부분

- TTL (Time to Live)
  - 패킷을 얼마나 멀리 보낼 것인가를 결정하는 요소 (IP header에 포함)
  - TTL은 정수로 표현되며, 라우터를 하나 거칠 때마다 1씩 감소
  - TTL이 0이 되면, 해당 패킷은 소멸됨
  - 멀티캐스트 전송을 위해 TTL 설정

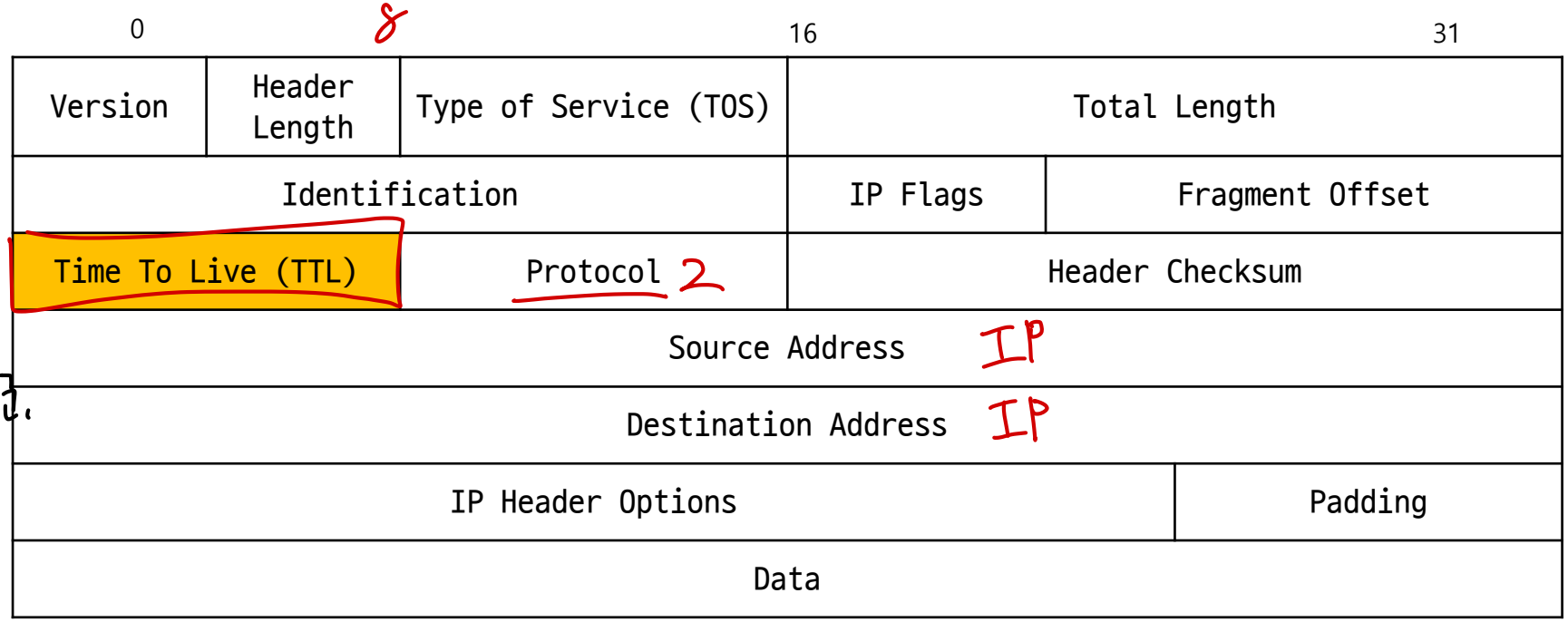
- TTL 설정
  - 프로토콜 레벨: IPPROTO_IP
  - 옵션 이름: IP_MULTICAST_TTL




```
int send_sock;
int time_live = 64;
. . .
send_sock = socket(PF_INET, SOCK_DGRAM, 0);
setsockopt(send_sock, IPPROTO_IP, IP_MULTICAST_TTL, (void*)&time_live, sizeof(time_live));
```

TTL을 64로 설정

IP Header 구조



- Protocol 필드
  - ✓ 1: ICMP, 2: IGMP
  - ✓ 6: TCP, 17: UDP 등

멀티캐스트 Sender 소스 (news_sender.c 일부)

```c
send_sock = socket(PF_INET, SOCK_DGRAM, 0);

memset(&mul_adr, 0, sizeof(mul_adr));
mul_adr.sin_family = AF_INET;
mul_adr.sin_addr.s_addr = inet_addr(argv[1]); // Multicast IP
mul_adr.sin_port=htons(atoi(argv[2])); // Multicast Port

setsockopt(send_sock, IPPROTO_IP, IP_MULTICAST_TTL, (void*)&time_live, sizeof(time_live));

if((fp = fopen("news.txt", "r")) == NULL)
    error_handling("fopen() error");

while(!feof(fp))
{
    fgets(buf, BUF_SIZE, fp);
    sendto(send_sock, buf, strlen(buf), 0, (struct sockaddr*)&mul_adr, sizeof(mul_adr));
    sleep(1);
}
```

- news_sender는 특정 멀티캐스트 그룹을 향해 파일에 저장된 데이터를 전송
  - ✓ IP_MULTICAST_TTL 설정
- 이 데이터는 라디오 방송처럼 receiver가 그룹에 가입하는 순간부터 수신을 시작

실행 결과

```
$ gcc news_sender.c -o news_sender
$ ./news_sender
Usage : ./news_sender <GroupIP> <PORT>
$ ./news_sender 224.1.1.2 9190
```

↳ 멀티캐스트 IP

멀티캐스트 데이터 송신: news_sender.c

TTL 설정.

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

#define TTL 64
#define BUF_SIZE 60
void error_handling(char *message);

int main(int argc, char *argv[])
{
    int send_sock;
    struct sockaddr_in mul_adr;
    int time_live = TTL;
    FILE *fp;
    char buf[BUF_SIZE];

    if(argc != 3) {
        printf("Usage : %s <GroupIP> <PORT>\n", argv[0]);
        exit(1);
    }

    send_sock = socket(PF_INET, SOCK_DGRAM, 0);
    memset(&mul_adr, 0, sizeof(mul_adr));
    mul_adr.sin_family = AF_INET;
    mul_adr.sin_addr.s_addr = inet_addr(argv[1]); // Multicast IP
    mul_adr.sin_port=htons(atoi(argv[2])); // Multicast Port

    setsockopt(send_sock, IPPROTO_IP, IP_MULTICAST_TTL,
                (void*)&time_live, sizeof(time_live));

    if((fp = fopen("news.txt", "r")) == NULL)
        error_handling("fopen() error");

    while(!feof(fp))
    {
        fgets(buf, BUF_SIZE, fp);
        sendto(send_sock, buf, strlen(buf), 0,
                (struct sockaddr*)&mul_adr, sizeof(mul_adr));
        sleep(1);
    }
    fclose(fp);
    close(send_sock);
    return 0;
}

void error_handling(char *message)
{
    fputs(message, stderr);
    fputc('\n', stderr);
    exit(1);
}
```

멀티캐스트 그룹 주소와 포트번호 설정

224.1.1.2 / 9190

Multicast TTL 설정

UDP 소켓을 이용하여 멀티캐스트 주소(mul_adr)로 데이터 전송

UDP 소켓

멀티캐스트 그룹 가입: 데이터 수신

- 멀티캐스트 그룹 가입은 소켓의 옵션 정보를 활용하여 그룹 가입이 이루어짐
- 그룹 가입 방법
  - ✓ 프로토콜 레벨: IPPROTO_IP
  - ✓ 옵션 이름: IP_ADD_MEMBERSHIP
- ip_mreq 구조체에 그룹 정보를 저장

```
struct ip_mreq
{
    struct in_addr imr_multiaddr;
    struct in_addr imr_interface;
}
```

멀티 캐스트 그룹주소

자신의 IP 주소

```
int recv_sock;
struct ip_mreq join_adr;
. . .
recv_sock = socket(PF_INET, SOCK_DGRAM, 0);
. . .
join_adr.imr_multiaddr.s_addr = "가입할 멀티캐스트 그룹 주소";
join_adr.imr_interface.s_addr = "멀티캐스트 그룹에 가입할 자신의 IP 주소";

// 멀티캐스트 그룹 가입
setsockopt(recv_sock, IPPROTO_IP, IP_ADD_MEMBERSHIP, (void*)&join_adr, sizeof(join_adr));
```

멀티캐스트 Receiver 소스 (news_receiver.c 일부)

```c
recv_sock = socket(PF_INET, SOCK_DGRAM, 0);
memset(&adr, 0, sizeof(adr));
adr.sin_family = AF_INET;
adr.sin_addr.s_addr = htonl(INADDR_ANY);
adr.sin_port = htons(atoi(argv[2]));

if(bind(recv_sock, (struct sockaddr*)&adr, sizeof(adr)) == -1)
    error_handling("bind() error");

// 멀티캐스트 그룹 주소
join_adr.imr_multiaddr.s_addr = inet_addr(argv[1]);
// 멀티캐스트 그룹에 가입할 자신의 IP주소
join_adr.imr_interface.s_addr = htonl(INADDR_ANY);

// 멀티캐스트 그룹 가입
setsockopt(recv_sock, IPPROTO_IP, IP_ADD_MEMBERSHIP,
           (void*)&join_adr, sizeof(join_adr));

while(1)
{
    str_len = recvfrom(recv_sock, buf, BUF_SIZE-1, 0, NULL, 0);
    if(str_len < 0)
        break;
    buf[str_len] = 0;
    fputs(buf, stdout);
}
```

- 멀티캐스트 그룹 가입
  - ✓ IP_ADD_MEMBERSHIP
- receiver가 멀티캐스트 그룹에 가입하는 순간부터 데이터를 수신

- imr_multiaddr.s_addr: 멀티캐스트 그룹 IP 주소
- imr_interface.s_addr: 자신의 IP 주소

실행 결과

```
$ gcc news_receiver.c -o news_receiver
$ ./news_receiver 224.1.1.2 9190
```

멀티캐스트 데이터 수신: news_receiver.c

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

#define BUF_SIZE 30
void error_handling(char *message);
int main(int argc, char *argv[])
{
    int recv_sock;
    int str_len;
    char buf[BUF_SIZE];
    struct sockaddr_in adr;
    struct ip_mreq join_adr;

    if(argc != 3) {
        printf("Usage : %s <GroupIP> <PORT>\n", argv[0]);
        exit(1);
    }

    recv_sock = socket(PF_INET, SOCK_DGRAM, 0);
    memset(&adr, 0, sizeof(adr));
    adr.sin_family = AF_INET;
    adr.sin_addr.s_addr = htonl(INADDR_ANY);
    adr.sin_port = htons(atoi(argv[2]));
    if(bind(recv_sock, (struct sockaddr*)&adr, sizeof(adr)) == -1)
        error_handling("bind() error");
    // 가입할. 멀티캐스트 그룹 주소 및 자신의 IP 주소 설정
    join_adr.imr_multiaddr.s_addr = inet_addr(argv[1]);
    join_adr.imr_interface.s_addr = htonl(INADDR_ANY);
    // 멀티캐스트 그룹 가입.
    setsockopt(recv_sock, IPPROTO_IP, IP_ADD_MEMBERSHIP,
                          (void*)&join_adr, sizeof(join_adr));

    while(1)
    {
        str_len = recvfrom(recv_sock, buf, BUF_SIZE-1, 0, NULL, 0);
        if(str_len < 0)
            break;
        buf[str_len] = 0;
        fputs(buf, stdout);
    }
    close(recv_sock);
    return 0;
}

void error_handling(char *message)
{
    fputs(message, stderr);
    fputc('\n', stderr);
    exit(1);
}
```

UDP 소켓을 이용하여 데이터 수신
recvfrom()의 수신 주소는 NULL 설정

실행 결과

news_receiver.c

```
$ gcc news_receiver.c –o receiver
$ ./receiver 224.1.1.2 9190

Ministry to form panel to remedy CPA exam
[????????] 2002?? 11?? 19?? (õ) 18:18
The Ministry of Finance and Economy announced its plans yesterday to create a committee to
reform the controversial CPA (certified public accountant) examination system.
```

news_sender.c

```
$ gcc news_sender.c –o sender
$ ./sender 224.1.1.2 9190
```

브로드캐스트의 이해와 구현의 방법

- 브로드캐스트 (Broadcast)
  - 동일한 네트워크 내에 존재하는 호스트에게 데이터를 전송하는 방법
  - 데이터 전송의 대상이 호스트가 아닌 네트워크
  - 멀티캐스트와 마찬가지로 UDP 소켓을 기반으로 함

1. Directed broadcast
   - IP 주소에서 네트워크 주소를 제외한 호스트 주소를 모두 1로 설정
     - 해당 네트워크로 데이터가 전송
   - 155.230.120.xxx 인 네트워크에 연결된 모든 호스트에게 전송
     - 155.230.120.255로 전송

2. Local broadcast
   - 255.255.255.255로 데이터 전송
     - 전송한 호스트가 속한 네트워크로 데이터가 전송

```
(base) [redacted]:~$ ifconfig
eno1: flags=4163<UP,BROADCAST,RUNNING,MULTICAST>  mtu 1500
        inet 155.230.120.235  netmask 255.255.255.0  broadcast 155.230.120.255
        inet6 fe80::7fd1:8e5b:2d1f:9558  prefixlen 64  scopeid 0x20<link>
        ether 34:17:eb:c1:ce:cf  txqueuelen 1000  (Ethernet)
        RX packets 104807952  bytes 9643333709 (9.6 GB)
        RX errors 0  dropped 7659204  overruns 0  frame 0
        TX packets 1824026  bytes 425295982 (425.2 MB)
        TX errors 0  dropped 0 overruns 0  carrier 0  collisions 0
        device interrupt 20  memory 0xf7800000-f7820000
```

SO_BROADCAST

- SO_BROADCAST 옵션 설정

```
int so_broad = 1; // SO_BROADCAST의 옵션정보를 1로 변경하기 위한 변수

setsockopt(send_sock, SOL_SOCKET, SO_BROADCAST, (void*)&so_broad, sizeof(so_brd));
```

– SO_BROADCAST
  ➢ 브로드캐스드를 설정하려면, 소켓의 SO_BROADCAST 정보를 1로 설정

- 예제 소스

```
int send_sock;
int so_broad = 1; // SO_BROADCAST의 옵션정보를 1로 변경하기 위한 변수

send_sock = socket(PF_INET, SOCK_DGRAM, 0);
. . .
setsockopt(send_sock, SOL_SOCKET, SO_BROADCAST, (void*)&so_broad, sizeof(so_broad));
```

브로드캐스트 Sender와 Receiver

브로드캐스트 Sender

```
send_sock = socket(PF_INET, SOCK_DGRAM, 0);
memset(&broad_adr, 0, sizeof(broad_adr));
broad_adr.sin_family = AF_INET;
broad_adr.sin_addr.s_addr = inet_addr(argv[1]);
broad_adr.sin_port = htons(atoi(argv[2]));

setsockopt(send_sock, SOL_SOCKET, SO_BROADCAST,
                    (void*)&so_brd, sizeof(so_brd));

if((fp = fopen("news.txt", "r")) == NULL)
    error_handling("fopen() error");

while(!feof(fp))
{
    fgets(buf, BUF_SIZE, fp);
    sendto(send_sock, buf, strlen(buf), 0,
        (struct sockaddr*)&broad_adr, sizeof(broad_adr));
sleep(1);
}
```

broadcast IP 주소

브로드캐스트 Receiver

```
recv_sock=socket(PF_INET, SOCK_DGRAM, 0);
memset(&adr, 0, sizeof(adr));
adr.sin_family=AF_INET;
adr.sin_addr.s_addr=htonl(INADDR_ANY);
adr.sin_port=htons(atoi(argv[1]));

if(bind(recv_sock, (struct sockaddr*)&adr, sizeof(adr))==-1)
    error_handling("bind() error");

while(1)
{
    str_len=recvfrom(recv_sock, buf, BUF_SIZE-1, 0, NULL, 0);
    if(str_len<0)
        break;
    buf[str_len]=0;
    fputs(buf, stdout);
}
```

from 주소는 NULL

- Broadcast sender는 255.255.255.255로 데이터 전송
- Broadcast receiver는 recvfrom(sock, buf, BUF_SIZE, 0, NULL, 0)
  - ✓ from 주소: NULL, 주소 길이(addrlen): 0
- 옵션 설정, 전송에 사용되는 IP 주소외에는 일반적인 UDP프로그램과 차이가 없음

브로드캐스트 데이터 전송: news_sender_brd.c

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

#define BUF_SIZE 30
void error_handling(char *message);
int main(int argc, char *argv[])
{
    int send_sock;
    struct sockaddr_in broad_adr;
    FILE *fp;
    char buf[BUF_SIZE];
    int so_brd=1; // SO_BROADCAST를  1로 변경하기 위한 변수
    if(argc!=3) {
        printf("Usage : %s <Boradcast IP> <PORT>\n", argv[0]);
        exit(1);
    }

    send_sock = socket(PF_INET, SOCK_DGRAM, 0);
    memset(&broad_adr, 0, sizeof(broad_adr));
    broad_adr.sin_family = AF_INET;
    broad_adr.sin_addr.s_addr = inet_addr(argv[1]);
    broad_adr.sin_port = htons(atoi(argv[2]));
```

SO_BROADCAST 속성 설정

```c
    setsockopt(send_sock, SOL_SOCKET, SO_BROADCAST,
               (void*)&so_brd, sizeof(so_brd));

    if((fp = fopen("news.txt", "r")) == NULL)
        error_handling("fopen() error");

    while(!feof(fp))
    {
        fgets(buf, BUF_SIZE, fp);

        sendto(send_sock, buf, strlen(buf), 0,
               (struct sockaddr*)&broad_adr, sizeof(broad_adr));
        sleep(1);
    }
    fclose(fp);
    close(send_sock);
    return 0;
}

void error_handling(char *message)
{
    fputs(message, stderr);
    fputc('\n', stderr);
    exit(1);
}
```

브로드캐스트 주소로 데이터 전송
(UDP 사용)

브로드캐스트 데이터 수신: news_receiver_brd.c

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

#define BUF_SIZE 30
void error_handling(char *message);

int main(int argc, char *argv[])
{
    int recv_sock;
    struct sockaddr_in adr;
    int str_len;
    char buf[BUF_SIZE];
    if(argc!=2) {
        printf("Usage : %s <PORT>\n", argv[0]);
        exit(1);
    }
    recv_sock=socket(PF_INET, SOCK_DGRAM, 0);

    memset(&adr, 0, sizeof(adr));
    adr.sin_family=AF_INET;
    adr.sin_addr.s_addr=htonl(INADDR_ANY);
    adr.sin_port=htons(atoi(argv[1]));

    if(bind(recv_sock, (struct sockaddr*)&adr, sizeof(adr))==-1)
        error_handling("bind() error");

    while(1)
    {
        str_len=recvfrom(recv_sock, buf, BUF_SIZE-1, 0, NULL, 0);

        if(str_len<0)
            break;
        buf[str_len]=0;
        fputs(buf, stdout);
    }
    close(recv_sock);
    return 0;
}

void error_handling(char *message)
{
    fputs(message, stderr);
    fputc('\n', stderr);
    exit(1);
}
```

UDP

브로드캐스트 통신에서
recvfrom()의 수신 주소는 NULL

실행 결과

news_receiver_brd.c

```
$ gcc news_receiver_brd.c -o receiver
$ ./receiver 9190

Ministry to form panel to remedy CPA exam
[????????] 2002?? 11?? 19?? (õ) 18:18
The Ministry of Finance and Economy announced its plans yesterday to create a committee to
reform the controversial CPA (certified public accountant) examination system.
Although CPAs will not be among the committee`s members, accountants say that the committee
will in all likelihood tackle issues that have been previously raised by them with the
ministry.
```

news_sender_brd.c

```
$ gcc news_sender_brd.c -o sender
$ ./sender 255.255.255.255 9190
```

Local Broadcast 주소

Questions?