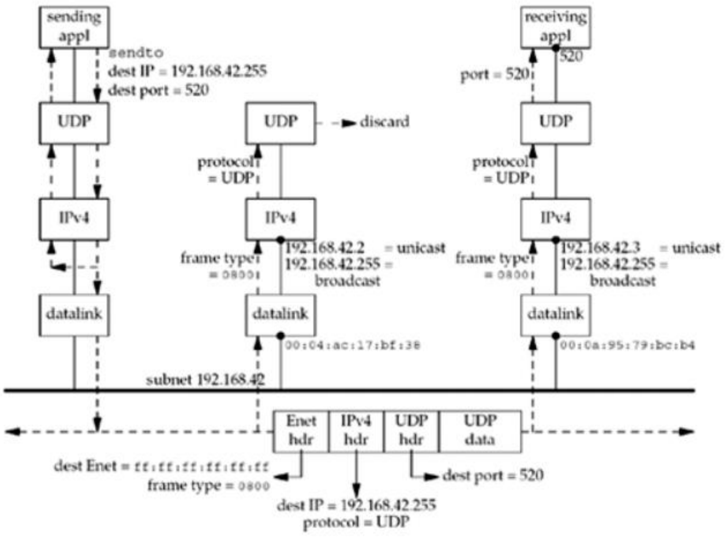
# UNIT-5

## 1. Show with a neat diagram, how UDP datagram can be applied for broadcasting

**Ans:** Block Diagram: UDP Datagram for Broadcasts

- **Step 1: Application Layer**
  - The application layer generates a UDP datagram with the desired data payload.
  - The destination address is set to the broadcast address of the interface.
- **Step 2: IP Layer**
  - The IP layer receives the UDP datagram from the application layer.
  - It checks the destination IP address and protocol field in the IPv4 header.
  - If the destination IP address is the broadcast address, the IP layer processes the datagram.
- **Step 3: UDP Layer**
  - The UDP layer receives the datagram from the IP layer.
  - It looks at the destination port and possibly the source port.
  - If the UDP socket is bound to the destination port, the datagram is placed onto the socket receive queue.
- **Step 4: Host Processing**
  - The process associated with the UDP socket is awakened, if necessary, to read the received datagram.
  - The application layer of the receiving process can then access the data payload of the broadcast UDP datagram.
- **Step 5: Broadcast Storm Prevention**
  - If a host receives a broadcast UDP datagram but does not have an application bound to the destination port, the datagram is discarded.
  - The host must not send an ICMP "port unreachable" message to prevent generating a broadcast storm.
- **Step 6: Logical Loopback**
  - By definition, a broadcast goes to every host on the subnet, including the sender.
  - The sending host may receive a copy of each broadcast datagram it sends, assuming it has bound the port it is sending to.



## 2. Explain the dg_cli function that broadcast to the standard UDP using SO_BROADCAST socket option

**Ans: SO_BROADCAST Socket Option:**

- This option enables or disables the ability of the process to send broadcast messages.
- Broadcasting is supported for only datagram sockets and only on networks that support the concept of a broadcast message.
- You cannot broadcast on a point-to-point link or any connection-based transport protocol such as SCTP or TCP.
- The POSIX specification requires the SO_BROADCAST socket option to be set to send a broadcast packet.
- Broadcasting was a privileged operation with 4.2BSD and the SO_BROADCAST socket option did not exist.
- This option was added to 4.3BSD and any process was allowed to set the option.
- We now modify our dg_cli function as shown below. This version sets the SO_BROADCAST socket option and prints all the replies received within five seconds.

```
#include "unp.h"

static void recvfrom_alarm(int);

void dg_cli(FILE *fp, int sockfd, const SA *pservaddr, socklen_t servlen){

        int n;

        const int on = 1;

        char sendline[MAXLINE], recvline[MAXLINE + 1];

        socklen_t len;

        struct sockaddr *preply_addr;

        preply_addr = Malloc(servlen);

        Setsockopt(sockfd, SOL_SOCKET, SO_BROADCAST, &on, sizeof(on));

        Signal(SIGALRM, recvfrom_alarm);

        while (Fgets(sendline, MAXLINE, fp) != NULL){

                Sendto(sockfd, sendline, strlen(sendline), 0, pservaddr, servlen);

                len = servlen;
```

## 3. Demonstrate with code, the use of dg_cli function that broadcast

**Ans:**

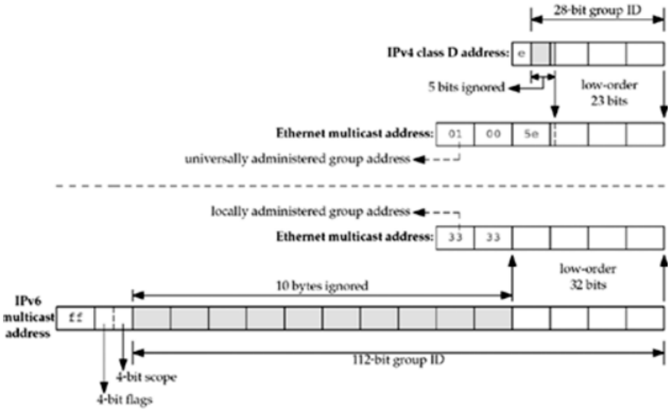
To demonstrate the use of the dg_cli function for broadcasting, we can refer to the code snippets provided in the document. The dg_cli function is defined in the code and it allows us to send broadcast datagrams to a specific destination address.

```
#include "unp.h"

static void recvfrom_alarm(int);

void dg_cli(FILE *fp, int sockfd, const SA *pservaddr, socklen_t servlen){

    int n;

    const int on = 1;

    char sendline[MAXLINE], recvline[MAXLINE + 1];

    socklen_t len;

    struct sockaddr *preply_addr;

    preply_addr = Malloc(servlen);

    Setsockopt(sockfd, SOL_SOCKET, SO_BROADCAST, &on, sizeof(on));

    Signal(SIGALRM, recvfrom_alarm);

    while (Fgets(sendline, MAXLINE, fp) != NULL){

        Sendto(sockfd, sendline, strlen(sendline), 0, pservaddr, servlen);

        alarm(5);

        for ( ; ; ){

            len = servlen;

            n = recvfrom(sockfd, recvline, MAXLINE, 0, preply_addr, &len);

            if (n < 0){

                if (errno == EINTR)

                    break;

                else

                err_sys("recvfrom error");

            }

            else {

                recvline[n] = 0;

                printf("from %s: %s", Sock_ntop_host(preply_addr, len), recvline);

            }
```

```
                    }
          }
          free(preply_addr);
}
static void recvfrom_alarm(int signo) {
          return;
}
```

## 4. Explain with a neat block diagram how IPV4 and IPV6 multicast addresses are mapped to Ethernet addresses

**Ans:**



IPv4 and IPv6 multicast addresses are mapped to Ethernet addresses using a process called multicast address resolution protocol (MAR) mapping.

The mapping process is slightly different for IPv4 and IPv6.

### For IPv4:

- It uses IGMP (Internet group Management Protocol)
- The least significant 23 bits of the IPv4 multicast address are used to form the low-order 23 bits of the Ethernet address.
- The most significant bit of the Ethernet address is set to 1.

### For IPv6:

- It uses MLD protocol (Multicast Listen Discovers)
- The low-order 32 bits of the IPv6 multicast address are used to form the low-order 32 bits of the Ethernet address.
- The next bit is set to 1.
- The most significant bit of the Ethernet address is set to 1.

The resulting Ethernet address is called the multicast MAC address, which is used to identify the specific multicast group on the local network segment.

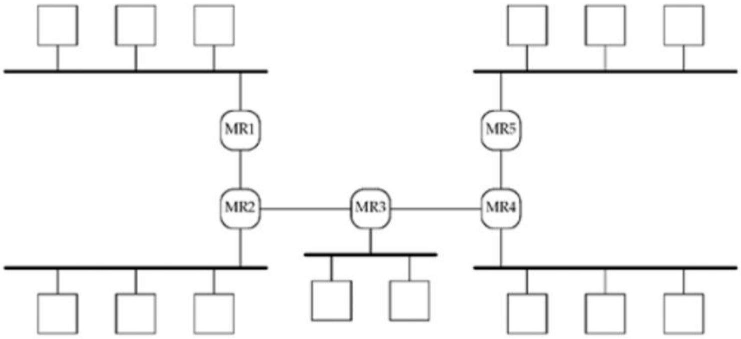
**5. Make use of UDP datagram to understand unicasting.**

**Ans:**

## 6. Justify how beneficial is Multicasting on a WAN network.

**Ans:**

- Multicasting on a WAN network is beneficial for efficiently transmitting data to multiple hosts interested in a specific multicast group.
- Unlike broadcasting, multicasting reduces unnecessary load on uninterested hosts.
- In a WAN setup with multicast routers and hosts joining a multicast group, multicast packets are selectively forwarded.
- This targeted transmission minimizes network congestion and optimizes bandwidth usage compared to less efficient alternatives like broadcast flooding or sending individual copies to each receiver.
- Multicasting is particularly advantageous in scenarios with numerous receivers, as it streamlines data delivery in a scalable and resource-effective manner.


MR1
MR5
MR2
MR3
MR4