ARP（Address Resolution Protocol）和NTP（Network Time Protocol）都属于广播通信。

ARP是局域网中的地址解析协议，利用这个协议，可以找出IP地址到MAC地址的映射关系。当主机A准备与主机B通信时，如果只知道主机B的IP地址，则主机A向整个全网发送一个ARP请求，询问IP地址为XXXX的主机，如果主机B收到就会产生回应。

NTP是网络时间协议。在支持广播的局域网中设置NTP协议，可以使NTP服务器每隔一个固定的时间间隔，就向全网发送时间信息，客户端在收到时间信息后进行更新处理。

原理解析：

要进行广播通信，首先要理解广播地址。在IP地址中，如果最后一个数字是255，则一定是一个广播地址。

- 网络广播地址：网络广播地址在没有进行子网划分的网络内广播，由于当强的网络均涉及子网划分，故此种地址很少存在
- 受限广播地址：以255.255.255.255组成的广播地址，在当前路由器均不转发此类广播
- 子网广播地址：子网广播地址是一种常用的广播方式，它是指在一个具体的子网内进行广播，比如192.168是网络ID，那么192.168.1.255就是子网192.168.1的广播
- 全部子网广播地址：是指所有子网络的广播，以上一个为例，全部子网广播地址是192.168.255.255

广播要采用UDP的方式，具体流程如下：

1. 创建UDP套接字
2. 设置套接字属性为SO_BROADCAST，设置为广播地址
3. 设置广播地址为INADDR_BROADCAST，同时也要指定发送端口
4. 进行数据收发操作

例子：


```
//bserver.c
#include <sys/types.h>
#include <stdio.h>
#include <sys/socket.h>
#include <stdlib.h>
#include <string.h>
#include <netdb.h>
#include <errno.h>
```

```c
#define BUFFSIZE 200
#define PORT 5050

int main(int argc, char **argv)
{
    int serversocket;
    struct sockaddr_in serveraddress,clientaddress;

    int so_broadcast=1;


    if((serversocket=socket(AF_INET,SOCK_DGRAM,0))<0){
        perror("socket");
        return 0;
    }

    if(setsockopt(serversocket,SOL_SOCKET,SO_BROADCAST,&so_broadcast,sizeof(so_broadcast))
<0){
        perror("setsockopt");
        return 0;
    }

    serveraddress.sin_family=AF_INET;
    serveraddress.sin_port=htons(INADDR_ANY);
    serveraddress.sin_addr.s_addr=htonl(INADDR_BROADCAST);

    if(bind(serversocket,(struct sockaddr*)&serveraddress,sizeof(struct sockaddr))<0){
        perror("bind");
        return 0;
    }

    clientaddress.sin_family=AF_INET;
    clientaddress.sin_port=htons(PORT);
    clientaddress.sin_addr.s_addr=htonl(INADDR_BROADCAST);

    while(1){
        char buf[BUFFSIZE];
        printf("please input your word:");
        scanf("%s",buf);
        if(sendto(serversocket,buf,strlen(buf),0,
(struct sockaddr*)&clientaddress,sizeof(clientaddress))<0){
            perror("sendto");
            return 0;
        }
        else
```

```
            printf("send msg: %s\n",buf);
    }

    return 0;
}
```

```
//bclient.c
#include <sys/types.h>
#include <stdio.h>
#include <sys/socket.h>
#include <stdlib.h>
#include <string.h>
#include <netdb.h>
#include <errno.h>

int main(int argc, char **argv)
{
    int clientsocket;
    struct sockaddr_in serveraddress,clientaddress;

    clientsocket=socket(AF_INET,SOCK_DGRAM,0);

    serveraddress.sin_family=AF_INET;
    serveraddress.sin_port=htons(5050);
    serveraddress.sin_addr.s_addr=htonl(INADDR_ANY);

    int opt=1;
    if(setsockopt(clientsocket,SOL_SOCKET,SO_REUSEADDR,&opt,sizeof(opt))<0){
        perror("setsockopt");
        return 0;
    }

    if(bind(clientsocket,(struct sockaddr*)&serveraddress,sizeof(struct sockaddr))!=0){
        perror("bind");
        return 0;
    }

    char buf[200];

    while(1){
        memset(buf,0,200);
        int size=0;
        size=recvfrom(clientsocket,buf,200,0,
(struct sockaddr*)&serveraddress,sizeof(serveraddress));
```

```
        buf[size]='\0';
        printf("IP:%s msg:%s\n",inet_ntoa(clientaddress.sin_addr),buf);

        if(strcmp(buf,"quit")==0){
            printf("system quit!\n");
            close(clientsocket);
            return 0;
        }
    }

    return 0;
}
```

（我在测试的时候只能发送不收接收，折磨了我半天，后来才想到是Linux防火墙的问题。。）