IP分片（gcc环境下实现IP数据报的分片功能，在终端打印）

1、实验思路

通过自己定义好的IP数据报的结构体以及linux下的recvfrom函数来接收IP数据报完成自动分片的功能并在终端打印分片的信息，如是哪个标识的第几片，是否可以分片等相关信息。

通过本机向虚拟机发送大于1500字节的IP数据报，在虚拟机下实现IP分片的功能。

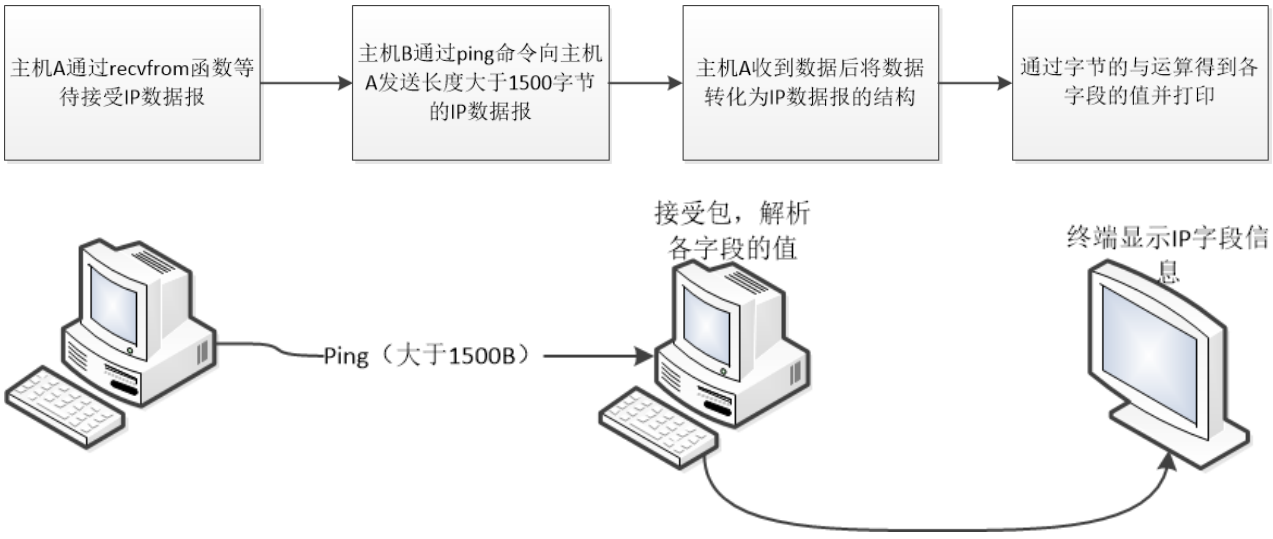
- 首先，需要了解IP结构体的定义以及其中每个成员所对应的IP字段；
- 其次，需要使用原始套接字socket来建立连接，其中第三个参数需指定为htons(ETH_P_IP);

```
fd = socket(PF_PACKET,SOCK_RAW,htons(ETH_P_IP));
```

- 然后，通过recvfrom函数来接受IP数据报；

```
struct ip_packet rcvBuffer; // 定义的buffer为IP结构体类型
recv(fd,&rcvBuffer,sizeof(rcvBuffer),0)
```

- 最后，通过字节的逻辑运算得到各字段的值并在终端打印；



2、实验步骤

- 在虚拟机运行接受IP数据报的程序等待接收IP数据报

```
./catchip    //需要在root权限下运行，否则无法创建socket
```

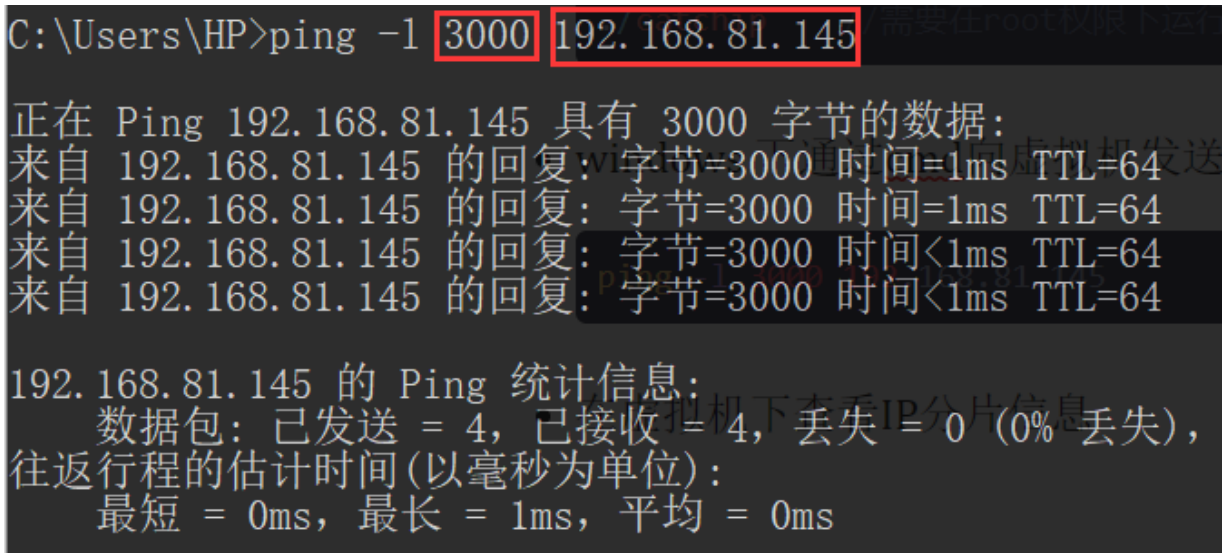
- windows 下通过cmd向虚拟机发送大的IP数据报，以自己的虚拟机IP地址为准，我的是192.168.81.145

```
ping -l 3000 192.168.81.145
```

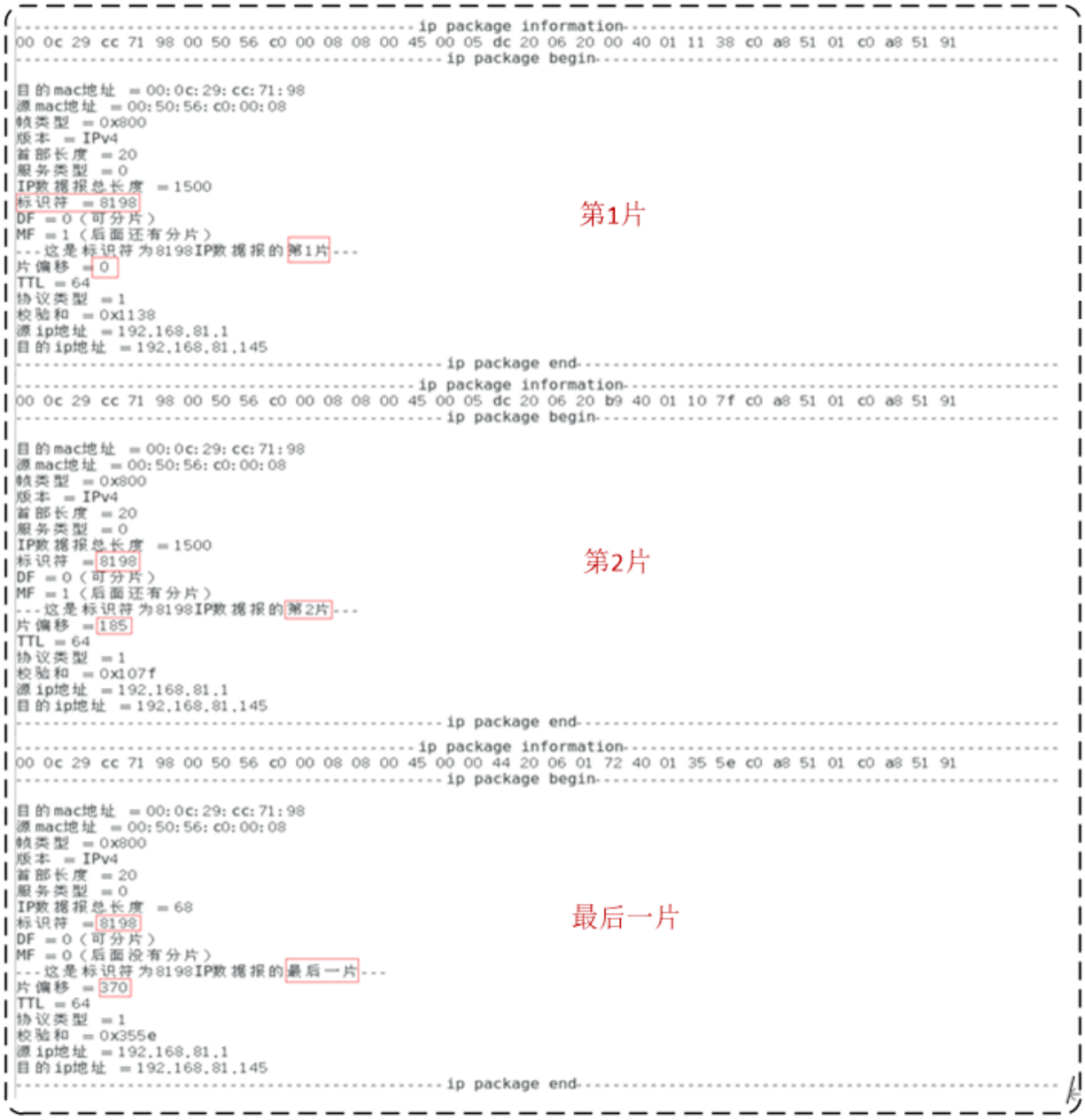
- 在虚拟机下查看IP分片信息

3、实验结果

- windows端发送3000字节IP数据报给虚拟机



- 虚拟机接收



从上图可以看出，总共接收到了3片IP数据报，标识符均为8198，表示三片IP分片来自同一个IP数据报，片偏移为370，可以算出IP数据报的总长度为3000，此时MF为0表示为IP数据报的最后一片。还有一些其他字段的信息，如MAC地址和IP地址。

4、实验代码

```
// 必要的头文件
#include <stdio.h>
#include <stdlib.h>
#include <string.h>
#include <sys/types.h>
#include <sys/socket.h>
#include <net/if.h>
#include <arpa/inet.h>
#include <netpacket/packet.h>
#include <net/ethernet.h>
#include<netinet/in.h>
```

```c
// 定义IP结构体
struct ip_packet {
    unsigned char dst_mac[6];   // 目的MAC
    unsigned char soc_mac[6];   // 源MAC
    unsigned short ethertype;   // 硬件类型
    unsigned char ip_v:4;     // 版本
    unsigned char ip_hl:4;    // 首部长度
    unsigned char ip_tos;    // 服务类型
    unsigned short tot_len;  // 总长度
    unsigned short id;    // 标志
    unsigned short off;      // 分片偏移
    unsigned char ttl;    // 生存时间
    unsigned char pro;  // 协议类型
    unsigned short chk_sum;  // 校验和
    unsigned char soc_ip[4];  // 源IP
    unsigned char dst_ip[4];  // 目的IP
};

void print_ip_packet(struct ip_packet ip);

int main() {
    int fd,num;
    struct ip_packet rcvBuffer;
    fd = socket(PF_PACKET,SOCK_RAW,htons(ETH_P_IP));  // 创建socket
    if(fd == -1) {
        perror("创建套接字失败! ");
    }
    while(1) {
        if(recv(fd,&rcvBuffer,sizeof(rcvBuffer),0) == -1)  // 持续接受IP数据报
            continue;
        print_ip_packet(rcvBuffer);  // 打印IP字段的信息
    }
    return 0;
}
void print_ip_packet(struct ip_packet ip) {
    int N,i,num;
    unsigned char *b;
    N = sizeof(struct ip_packet);
    b = (unsigned char *)malloc(N);
    num = ((ntohs(ip.off) & 0x1fff) / 185 ) + 1;
    memcpy(b,&ip,sizeof(struct ip_packet));
    printf("-----------------------------------------ip package information-----------
-----------------------------------\n");
    for(i=0; i<N; i++) {
        printf("%02x ",b[i]);
    }
    printf("\n-----------------------------------------ip package begin-------------
-----------------------------------\n");
    printf("目的mac地址 = ");
    for(i=0; i<6; i++)
        printf(i > 0 ? ":%.2x" : "%.2x", ip.dst_mac[i]);
```

```
    printf("\n源mac地址 = ");
    for(i=0; i<6; i++)
        printf(i > 0 ? ":%.2x" : "%.2x", ip.soc_mac[i]);
    printf("\n帧类型 = 0x%x",ntohs(ip.ethertype));
    printf("\n版本 = IPv%d",ip.ip_hl); //版本和首部长度换以下
    printf("\n首部长度 = %d",ip.ip_v*4);
    printf("\n服务类型 = %d",ntohs(ip.ip_tos) >> 8);
    printf("\nIP数据报总长度 = %d",ntohs(ip.tot_len));
    printf("\n标识符 = %d",ntohs(ip.id));

    if((ntohs(ip.off)>>14) & 1) // 取第2位
        printf("\nDF = 1 (不可分片) ");
    else
        printf("\nDF = 0 (可分片) ");
    if((ntohs(ip.off)>>13) & 1) {
        printf("\nMF = 1 (后面还有分片) ");
        printf("\n---这是标识符为%dIP数据报的第%d片---",ntohs(ip.id),num);
    } // 取第3位
    else {
        printf("\nMF = 0 (}后面没有分片) ");
        printf("\n---这是标识符为%dIP数据报的最后一片---",ntohs(ip.id));
    }
    printf("\n片偏移 = %d",ntohs(ip.off) & 0x1fff); // 取后13位得到片偏移
    printf("\nTTL = %d",ntohs(ip.ttl) >> 8);
    printf("\n协议类型 = %d",ntohs(ip.pro) >> 8);
    printf("\n校验和 = 0x%x",ntohs(ip.chk_sum));
    printf("\n源ip地址 = ");
    for(i=0; i<4; i++)
        printf(i > 0 ? ".%d" : "%d", ip.soc_ip[i]);
    printf("\n目的ip地址 = ");
    for(i=0; i<4; i++)
        printf(i > 0 ? ".%d" : "%d", ip.dst_ip[i]);
    printf("\n----------------------------------------------ip package end---------------
-----------------------------------\n");
    printf("\n\n");
}
```