哈爾濱工業大學
Harbin Institute of Technology

计算机网络
课程实验报告

实验名称	IPv4 分组的收发和转发实验				
姓名	冯开来		院系	计算学部	
班级	1903602		学号	1190201215	
任课教师	李全龙		指导教师	李全龙	
实验地点	格物 207		实验时间	2021.11.13	
实验课表现	出勤、表现得分(10)		实验报告得分(40)	实验总分	
	操作结果得分(50)				
教师评语					

计算机科学与技术学院 SINCE 1956...
School of Computer Science and Technology

实验目的：

IPv4 分组收发实验：

1） IPv4 协议是互联网的核心协议，它保证了网络节点（包括网络设备和主机）在网络层能够按照标准协议互相通信。IPv4 地址唯一标识了网络节点和网络的连接关系。在我们日常使用的计算机的主机协议栈中，IPv4 协议必不可少，它能够接收网络中传送给本机的分组，同时也能根据上层协议的要求将报文封装为IPv4 分组发送出去。

2） 本实验通过设计实现主机协议栈中的IPv4 协议，让学生深入了解网络层协议的基本原理，学习IPv4 协议基本的分组接收和发送流程。

3） 另外，通过本实验，学生可以初步接触互联网协议栈的结构和计算机网络实验系统，为后面进行更为深入复杂的实验奠定良好的基础。

IPv4 分组转发实验：

1) 本实验需要将实验模块的角色定位从通信两端的主机转移到作为中间节点的路由器上，在 IPv4 分组收发处理的基础上，实现分组的路由转发功能。

2) 网络层协议最为关注的是如何将 IPv4 分组从源主机通过网络送达目的主机，这个任务就是由路由器中的 IPv4 协议模块所承担。路由器根据自身所获得的路由信息，将收到的 IPv4 分组转发给正确的下一跳路由器。如此逐跳地对分组进行转发，直至该分组抵达目的主机。IPv4 分组转发是路由器最为重要的功能。

3) 本实验设计模拟实现路由器中的 IPv4 协议，可以在原有 IPv4 分组收发实验的基础上，增加 IPv4 分组的转发功能。对网络的观察视角由主机转移到路由器中，了解路由器是如何为分组选择路由，并逐跳地将分组发送到目的主机。本实验中也会初步接触路由表这一重要的数据结构，认识路由器是如何根据路由表对分组进行转发的。

实验内容：

1） **实现IPv4 分组的基本接收处理功能**

对于接收到的IPv4 分组，检查目的地址是否为本地地址，并检查IPv4分组头部中其它字段的合法性。提交正确的分组给上层协议继续处理，丢弃错误的分组并说明错误类型。

2） **实现IPv4 分组的封装发送**

根据上层协议所提供的参数，封装IPv4 分组，调用系统提供的发送接口函数将分组发送出去。

3） **设计路由表数据结构。**

设计路由表所采用的数据结构。要求能够根据目的IPv4 地址来确定分组处理行为（转发情况下需获得下一跳的IPv4 地址）。路由表的数据结构和查找算法会极大的影响路由器的转发性能，有兴趣的同学可以深入思考和探索。

4） **IPv4 分组的接收和发送。**

对前面实验（IP 实验）中所完成的代码进行修改，在路由器协议栈的IPv4 模块中能够正确完成分组的接收和发送处理。具体要求不做改变，参见“IP 实验”。

5） **IPv4 分组的转发。**

对于需要转发的分组进行处理，获得下一跳的IP 地址，然后调用发送接口函数做进一步处理。

实验过程：

IPv4 分组收发实验：

首先如果收到了 ip 数据报，得到该数据报的各个信息，包括版本号，首部长，TTL，头部校验和和目的地址。然后分别比较 TTL 值，版本号，首部长目的地址，如果不一样则

全部跳转到错误。最后计算校验和。计算校验和的方法是每次在 pBuffer 取两位构成 16 进制数然后不断累加，如果超过 0xffff 了就再加 1（因为 unsigned short 是 16 位，所以超过了 0xffff 会自动舍弃第一位）。当所有和加完之后判断 sum 是不是等于 0xffff，如果不是，就跳转到错误，如果是，成功接受。

```
    int version = pBuffer[0] >> 4;
    int headLength = pBuffer[0] & 0xf;
    int TTL = (unsigned short)pBuffer[8];
    int headCheckSum = ntohs(*(unsigned short *)(pBuffer + 10));
    int dstAddr = ntohl(*(unsigned int*)(pBuffer + 16));

    //TTL值错误
    if (TTL <= 0){
        ip_DiscardPkt(pBuffer, STUD_IP_TEST_TTL_ERROR);
        return 1;
    }
```

```
    //校验和错应该最后检验错误
    unsigned short sum = 0;
    unsigned short tempNum = 0;
    for (int i = 0; i < headLength * 2; i++){
        tempNum = ((unsigned char)pBuffer[i*2]<<8) + (unsigned char)pBuffer[i*2 + 1];
        if (sum + tempNum > 0xffff)
            sum = sum + tempNum + 1;
        else
            sum = sum + tempNum;
    }
    if (sum != 0xffff){
        ip_DiscardPkt(pBuffer, STUD_IP_TEST_CHECKSUM_ERROR);
        return 1;
    }

    //成功接受
    ip_SendtoUp(pBuffer,length);
    return 0;
}
```

其次如果是发送 ip 数据报，需要定义好版本号和首部长，如 0x45 存放到 IPBuffer[0] 中，其他信息也是这样。同理，这里计算校验和的方式和接收方一样，先通过上面的方法计算出 sum（其中 headCheckSum 为 0），然后用 0xffff 减去 sum 得到我们需要的 headCheckSum。

```
    //计算checksum
    for (int i = 0; i < 10; i++){
        tempNum = ((unsigned char)IPBuffer[i*2]<<8) + (unsigned char)IPBuffer[i*2 + 1];
        if (tempNum + sum > 0xffff)
            sum = sum + tempNum + 1;
        else
            sum = sum + tempNum;
    }
    headCheckSum = htons(0xffff - sum);
```

IPv4 分组转发实验：

这部分实验首先需要初始化路由表，然后第一个要实现的函数是添加路由信息。这个就是讲数据报里面的掩码、目的 IP、下一跳分别放到路由表中。

```
void stud_route_add(stud_route_msg *proute)
{
    routeTableItem newTableItem;
    newTableItem.masklen = ntohl(proute->masklen);  //将一个无符号长整形数从网络字节顺序转换为主机字节顺序
    newTableItem.mask = (1<<31)>>(ntohl(proute->masklen)-1); //
    newTableItem.destIP = ntohl(proute->dest);
    newTableItem.nexthop = ntohl(proute->nexthop);
    table.push_back(newTableItem);
    return;
}
```

第二个实现的函数是处理IP数据报，处理步骤分别是看ip分组的地址是否为本机地址，如果是交付上层，如果不是继续判断TTL，小于等于0则丢弃。接下来的步骤是设置匹配位，这一部分有点麻烦，因为要找到最长匹配位，就要一位位的匹配。

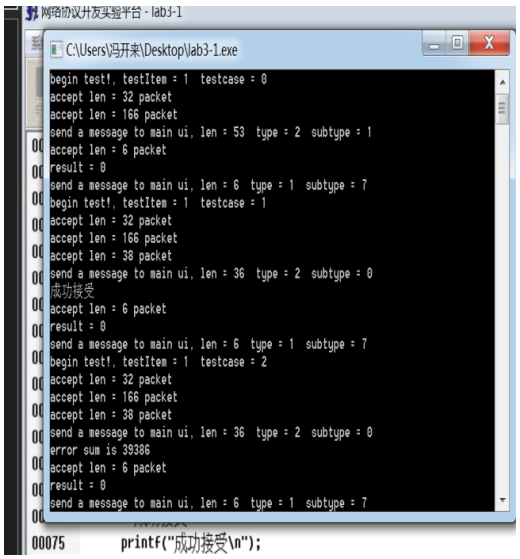
```
//设置匹配位
bool Match = false;
unsigned int longestMatchLen = 0; // 最长匹配
int bestMatch = 0;
// 判断掩码是否匹配
for(int i = 0; i < table.size(); i ++)
{
    if(table[i].masklen > longestMatchLen && table[i].destIP == (DestIP & table[i].mask)) //
    {
        bestMatch = i;
        Match = true;
        longestMatchLen = table[i].masklen;
    }
}
```

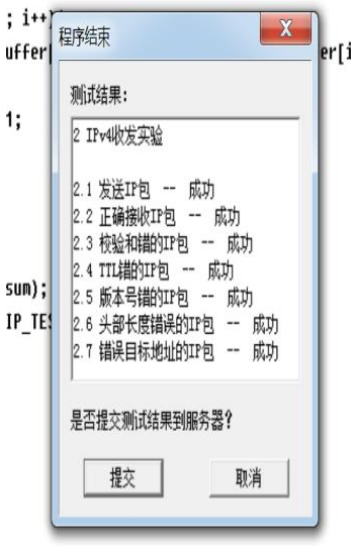
最后如果匹配成功了，就重新计算校验和并发送给下一层协议。如果匹配失败了，则舍弃该IP分组。

```
82
83      if(Match) //匹配成功
84      {
85          char *buffer = new char[length];
86          memcpy(buffer,pBuffer,length);
87          buffer[8]--; //TTL - 1
88          int sum = 0;
89          unsigned short int localCheckSum = 0;
90          for(int j = 1; j < 2 * headsum +1; j ++)
91          {
92              if (j != 6){
93                  sum = sum + (buffer[(j-1)*2]<<8)+(buffer[(j-1)*2+1])
94                  sum %= 65535;
95              }
96          }
97          //重新计算校验和
98          localCheckSum = htons(~(unsigned short int)sum);
99          memcpy(buffer+10, &localCheckSum, sizeof(unsigned short));
100         // 发给下一层协议
101         fwd_SendtoLower(buffer, length, table[bestMatch].nexthop);
102         return 0;
103     }
```

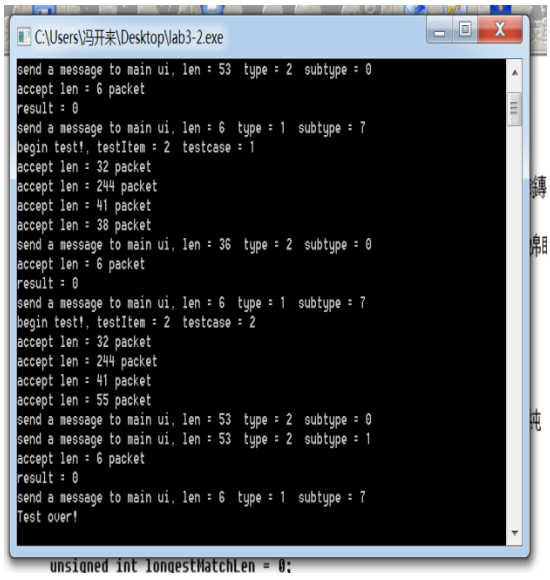
实验结果：

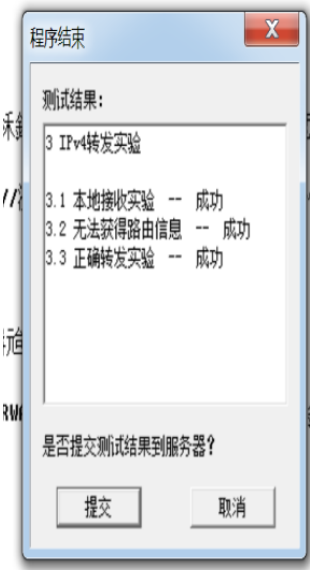
IPv4分组收发实验：





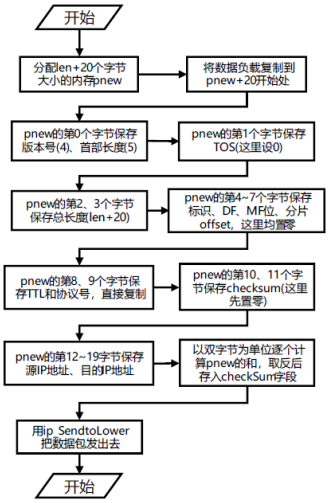
IPv4分组转发实验：



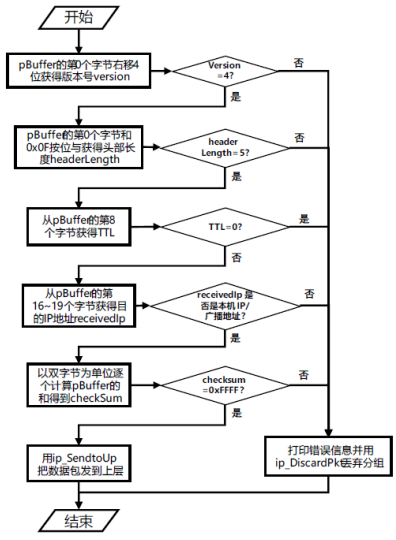


问题讨论：

1） 要求给出发送和接收函数的实现程序流程图；



2） 给出自己所新建的数据结构的说明（如果有）：



3) 给出版本号（Version）、头部长度（IP Head length）、生存时间（Time to live）以及头校验和（Header checksum）字段的错误检测原理，并根据实验具体情况给出错误的具体数据，例如如果为头部长度错，请给出收到的错误的IP 分组头部长度字段值为多少。

a) 版本号错误匹配

对于版本号，由于我们所用的IP版本都是IPv4，因此版本号都应该保存的是4. 接受的pBuffer的第0个字节的高4位即为版本号。

如下图所示，我们的代码可以检测出版本号检测，并输出版本号错误的值。

```
begin test!, testItem = 1  testcase = 4
accept len = 32 packet
accept len = 166 packet
accept len = 38 packet
send a message to main ui, len = 36  type = 2  subtype = 0
ERROR: version is 3
accept len = 6 packet
result = 0
send a message to main ui, len = 6  type = 1  subtype = 7
```

b) 首部长度错误

pBuffer第0个字节的低4位表示首部长度，用pBuffer[0]&0xf即可获得。由于头部长度的单位是4字节，而典型的IP分组头部长度是20字节，因此headLength一般是5。

如下图所示，我们代码可以检测出首部长度错误，并打印接受到的首部长度。

```
begin test!, testItem = 1  testcase = 5
accept len = 32 packet
accept len = 166 packet
accept len = 38 packet
send a message to main ui, len = 36  type = 2  subtype = 0
ERROR: headlength is 3
accept len = 6 packet
result = 0
send a message to main ui, len = 6  type = 1  subtype = 7
```

c) 生存时间错误

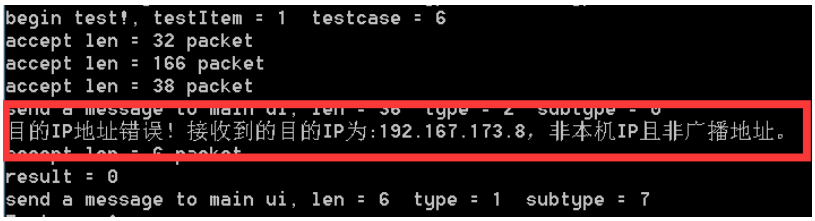
TTL每经过一个路由减1，当TTL减为0的时候，则不应该转发这个分组。直接通过pBuffer第8个字节获得TTL

图所示，TTL小于等于0的时候丢弃分组。

```
begin test!, testItem = 1  testcase = 3
accept len = 32 packet
accept len = 166 packet
accept len = 38 packet
send a message to main ui, len = 36  type = 2  subtype = 0
ERROR: ttl <= 0accept len = 6 packet
result = 0
send a message to main ui, len = 6  type = 1  subtype = 7
```

d) 目的IP地址

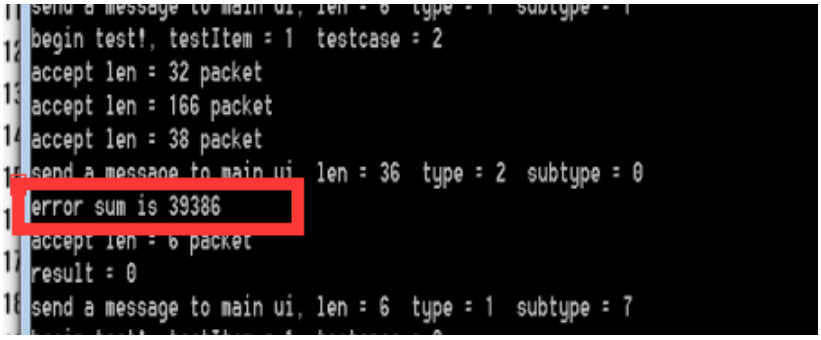
目的IP地址保存再pBuffer第16-19个字节，用一个unsigned int类型的数来保存。将pBuffer+16开始的4个字节用unsigned int类型表示，与Ipv4Address比较是否相等。如下图，若IP地址既不是广播地址也不是目的地址，则丢弃该分组。



e) 检验和错误

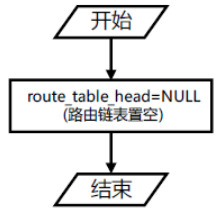
Checksum计算原理通过头部20个字节，每两个字节放在一起求和（进位移到末尾相加），如果求得的和等于0xffff(二进制全1)则验证通过。

如下图，求和结果不是0xffff，则丢弃该分组并答应算出来的sum。

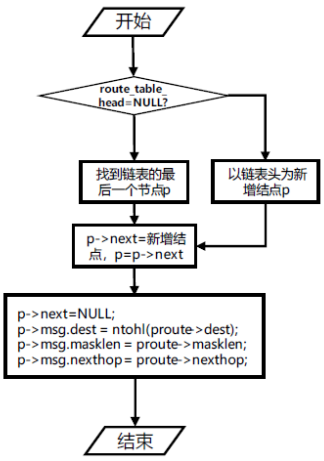


4) 要求给出路由表初始化、路由增加、路由转发三个函数的实现流程图；

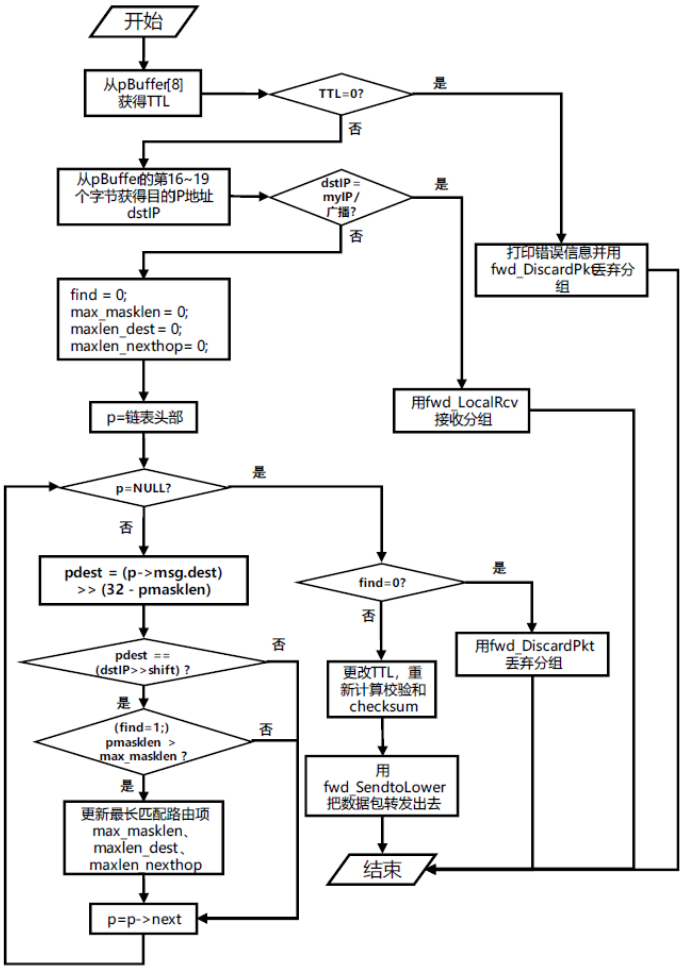
a) 路由表初始化



b) 路由增加函数



c） 路由转发函数



5） 要求给出所新建数据结构的说明；

```
typedef struct stud_toute_msg
{
    unsigned int dest;      //目的IP
    unsigned int mask;      //掩码
    unsigned int masklen;   //掩码长度
    unsigned int nexthop;   //下一跳
}stud_route_msg;
```

6） 请分析在存在大量分组的情况下如何提高转发效率，如果代码中有相关功能实现，请给出具体原理说明。

a） 对路由表建立红黑树数据结构用于查询路由项提高检索速度

b） 采用路由聚合技术，将统一子网内的路由信息合并以提高检索速度

心得体会：

1. 了解了IPv4分组收发原理
2. 了解了IPv4分组转发原理
3. 了解IP数据报的结构
4. 了解了提高转发效率的方法
5. 对计算机网络有了进一步认识