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 分组收发/转发实验				
姓名	卢兑玳		院系	计算机科学与技术学院	
班级	1803501		学号	L170300901	
任课教师			指导教师		
实验地点			实验时间		
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					

计算学部

实验目的：

（注：实验报告模板中的各项内容仅供参考，可依照实际实验情况进行修改。）

本次实验的主要目的。

1. 设计实现主机协议栈中的 IPv4 协议
2. 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. 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。
4. 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。
5. 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

实验内容：

概述本次实验的主要内容，包含的实验项等。

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。

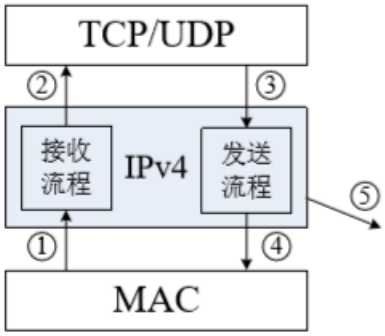
4）IPv4 分组的接收和发送。对前面实验（IP 实验）中所完成的代码进行修改，在路由器协议栈的IPv4模块中能够正确完成分组的接收和发送处理。具体要求不做改变，参见“IP 实验”。

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：

以文字描述、实验结果截图等形式阐述实验过程，必要时可附相应的代码截图或以附件形式提交。

一、 流程图：



二、发送函数的实现程序流程：

1. 根据所传参数（如数据大小），来确定分配的存储空间的大小并申请分组的存储空间。
2. 按照 IPv4 协议标准填写 IPv4 分组头部各字段，标识符（Identification）字段可以使用一个随机数来填写。（注意：部分字段内容需要转换成网络字节序）
3. 完成 IPv4 分组的封装后，调用 ip_SendtoLower( )接口函数完成后续的发送处理工作，最

终将分组发送到网络中。

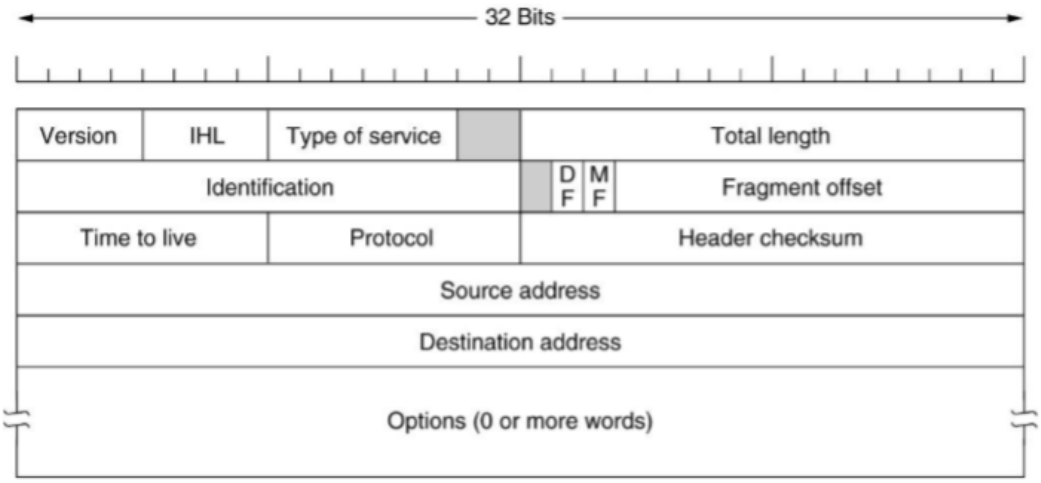
三、接收函数的实现程序流程：

1. 检查接收到的 IPv4 分组头部的字段，包括版本号（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。对于出错的分组调用 ip_DiscardPkt( ) 丢弃，并说明错误类型。

2. 检查 IPv4 分组是否应该由本机接收。如果分组的目的地址是本机地址或广播地址，则说明此分组是发送给本机的；否则调用 ip_DiscardPkt( )丢弃，并说明错误类型。

3. 如果 IPV4 分组应该由本机接收，则提取得到上层协议类型，调用 ip_SendtoUp( )接口函数，交给系统进行后续接收处理。

分组头部格式：



字段的错误检测原理：

1. 版本号（Version）：

版本号在第一个字节的前 4 位里面，pBuffer[0]是第一个字节，因此可以用 pBuffer[0] >> 4来取得版本号，检测是否等于4，如果是 4，则版本号正确，代码如下：

```
//IP版本号错
if (version != 4) {
    ip_DiscardPkt(pBuffer, STUD_IP_TEST_VERSION_ERROR);
    return 1;
}
```

2. 头部长度（IP Head length）：

头部长度是紧跟在版本号后的四位，pBuffer[0]是第一个字节，因此可以用pBuffer[0] & 0xf来表示头部长度。如果小于 5，则代表出错。代码如下：

```
if (headLength < 5) {
    ip_DiscardPkt(pBuffer, STUD_IP_TEST_HEADLEN_ERROR);
    return 1;
}
```

3. 生存时间（Time to live）：

生存时间就是TTL，在第 9 个字节里，根据上面两个取值的方法，很容易就可以找到该值，为TTL = (unsigned short)pBuffer[8]。如果TTL的值小于等于 0，则这个包不能再被转发了，需要抛弃这个包。因此，检测代码如下：

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

4. 头校验和（Header checksum）：

头部校验和校验和在第 11到12 个字节，校验和计算方法：首先头部除去校验和部分，所有字段进行求和，然后用FFFF减，检测得到的值是否正确。

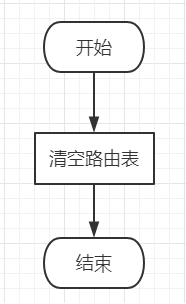
代码如下：

```
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 (0xffff - sum < tempNum)
        sum = sum + tempNum + 1;
    else
        sum = sum + tempNum;
}
if (sum != 0xffff) {
    ip_DiscardPkt(pBuffer, STUD_IP_TEST_CHECKSUM_ERROR);
    return 1;
}
```

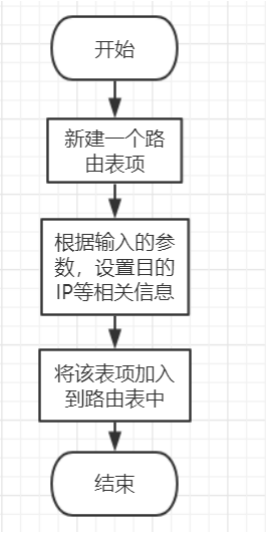
分组转发实验：

一、流程图：

1. 路由表初始化函数的实现流程图：

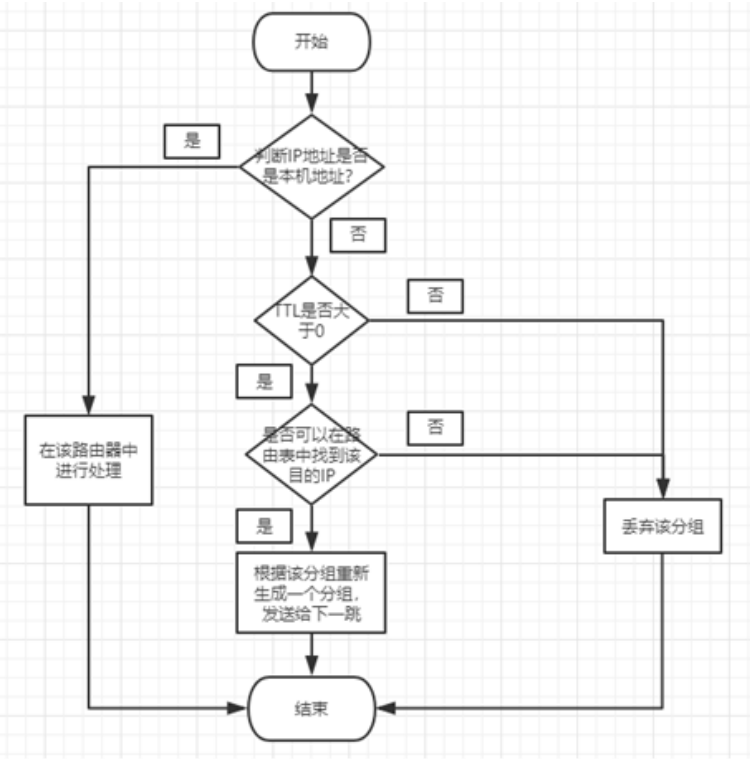


2. 路由增加函数的实现流程图：



代码中，需要设置目的地址、子网掩码的长度、子网掩码的值以及下一跳的位置。

3.路由转发函数的实现流程图：



二、新建的数据结构的说明：

新建的数据结构中，主要有四项：目的地址、子网掩码的长度、子网掩码的值以及下一跳的位置，代码如下：

```
struct routeTableItem
{
    unsigned int destIP;      //目的IP
    unsigned int mask;        // 掩码
    unsigned int masklen;     // 掩码长度
    unsigned int nexthop;     // 下一跳
};
```

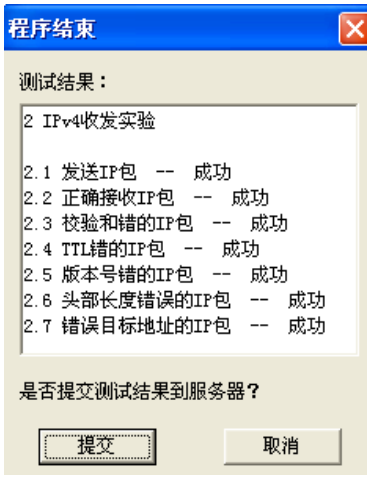
三、在存在大量分组的情况下如何提高转发效率：

1. 可以改进数据结构，利用堆查询的方法进行路由转发查找，提高查找效率，间接提高转发效率。

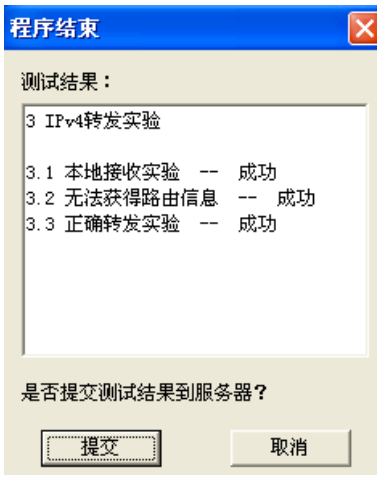
2. 可以使用多线程进行，多个转发同时进行，可以直接提高转发效率。

实验结果：

分组收发：



分组转发：



问题讨论：

对实验过程中的思考问题进行讨论或回答。

在IP分组转发实验中，如果存在大量的分组的情况下，如何提高转发效率：

1. 如果继续使用本次实验中所使用的vector进行效率的提高，最简单的方式就是将路由表进行有序的存储（如按照IP目的地址的大小存储），在vector进行二分查找，将每次转发搜索路由表的时间由O(n)变为O(logn)。

2. 如果改进路由表的数据结构，可以使用平衡树之类数据结构，将查询的时间变为稳定O(logn)，并且缩小新增表项重新整理成有序的时间。

心得体会：

结合实验过程和结果给出实验的体会和收获。

本次实验，让我更加了解了 IPv4 的分组收发和转发原理，体验了分组收发转发的真实情况，对网络有了进一步的认识。