



哈尔滨工业大学
Harbin Institute of Technology

计算机网络 课程实验报告

| | | | | | | |
|-------|-------------|--|------------|------------|------|--|
| 实验名称 | IP 分组的收发与转发 | | | | | |
| 姓名 | 马旭 | | 院系 | 计算科学与技术学院 | | |
| 班级 | 1603106 | | 学号 | 1160300601 | | |
| 任课教师 | 聂兰顺 | | 指导教师 | 聂兰顺 | | |
| 实验地点 | 格物楼 207 | | 实验时间 | 2018.11.14 | | |
| 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 | | | | | | |
| | | | | | | |

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

目录

| | |
|---------------------------|----|
| 1. 实验目的 | 3 |
| 2. 实验内容 | 3 |
| 3. 实验过程 | 4 |
| 3.1. 使用的数据结构 | 4 |
| 3.2. 错误检测原理 | 4 |
| 3.3 流程图 | 5 |
| 3.4 大量分组下提高转发效率 | 8 |
| 4. 实验结果 | 9 |
| 4.1. IPv4 分组收发实验结果: | 9 |
| 4.2. IPv4 分组转发实验结果: | 9 |
| 4.3. 在线实验系统成绩 | 10 |
| 5. 问题讨论 | 10 |
| 6. 心得体会 | 10 |

1. 实验目的

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。

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。

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。

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。

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。

2. 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。

3.实验过程

3.1.使用的数据结构

1. 在构建路由表时使用 C++中自带的容器类 `vector` 类，构成了链表
2. 路由结构如下：

```
struct table
{
    int dest;
    int nexthop;
};
vector<table> routerTable;
```

3.2.错误检测原理

1. 版本号错误检测原理

从 IPv4 报文段中提取出来版本号，然后判断是否为 4，不为 4 就抛出错误。

```
if(version!=4){
    ip_DiscardPkt(pBuffer,STUD_IP_TEST_VERSION_ERROR);
    return 1;
}
```

2. 头部长度的错误检测原理从 IPv4 报文段中提取出来头部长度的，然后判断是否大于等于 5，如果小于 5 就抛出错误。

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

3. 生存时间错误检测原理从 IPv4 报文段中提取出来生存时间，然后判断是否小于等于 0，如果小于等于 0 就抛出错误。

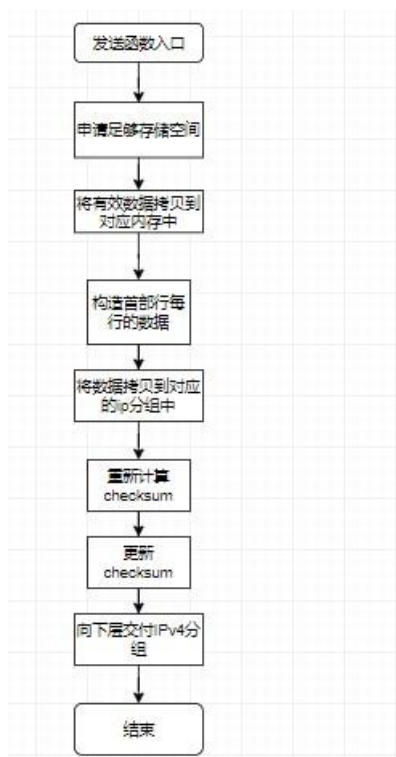
```
if(ttl <= 0 ){
    ip_DiscardPkt(pBuffer,STUD_IP_TEST_TTL_ERROR);
    return 1;
}
```

4. 头校验和错误检测原理提取出来头部校验和，然后重新计算头部校验和，判断两者是否相等，如果不相等就抛出错误。

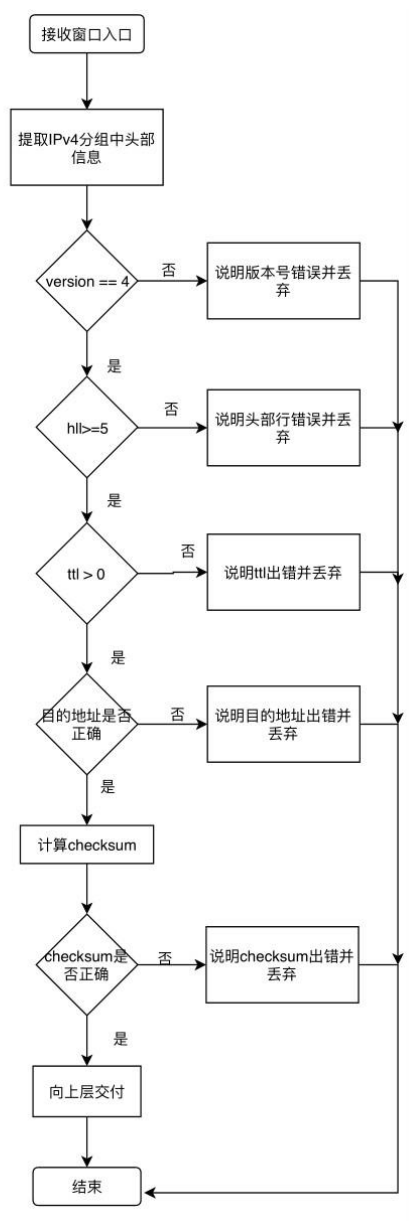
```
if(checksum != headerChecksum) {  
    ip_DiscardPkt(pBuffer,STUD_IP_TEST_CHECKSUM_ERROR);  
    return 1;  
}
```

3.3 流程图

1. 发送函数的流程图



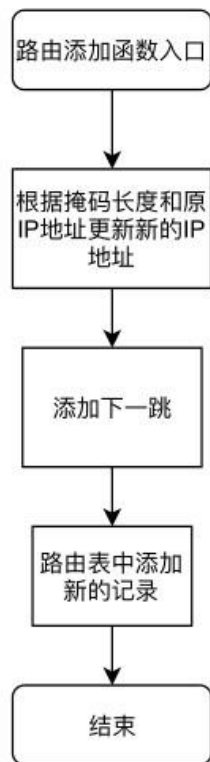
2. 接收函数的流程图



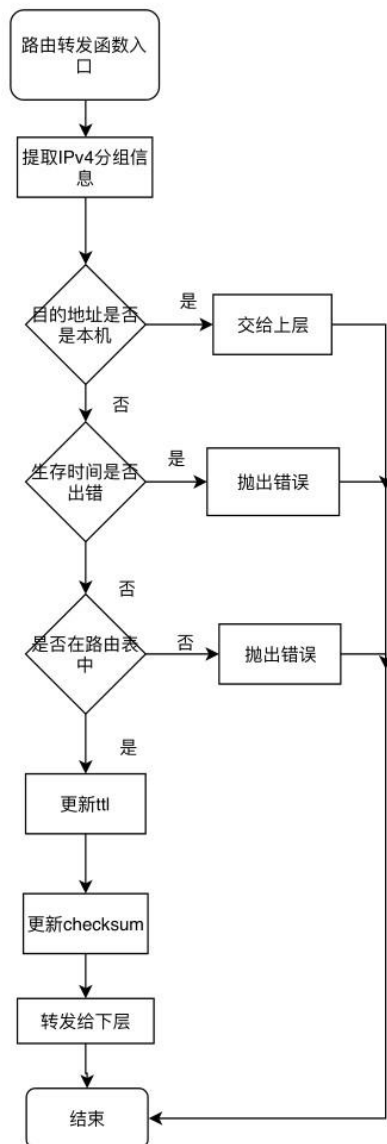
5. 路由表初始化函数流程图



6. 路由增加函数流程图



7. 路由转发函数流程图

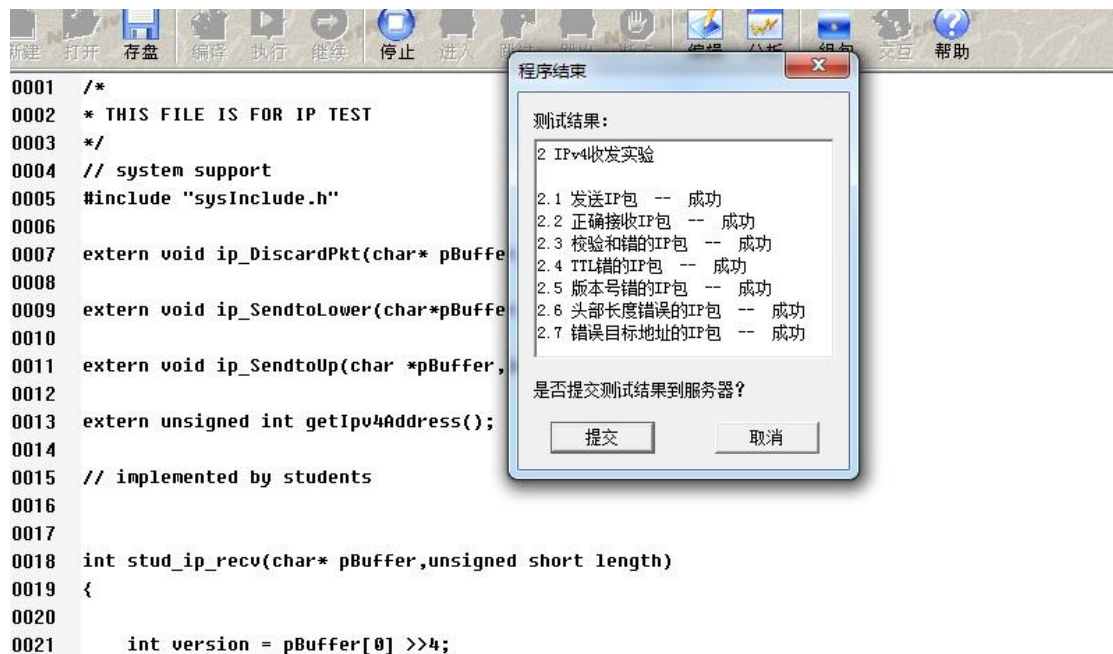


3.4 大量分组下提高转发效率

1. 使用哈希表存储路由表，这样的话可以极快的查找对应的下一跳，时间复杂度为 $O(1)$
2. 在实现时使用最长优先匹配原则，可以加快匹配的速度。
3. 对存储的路由数据进行排序，这样可以加快查找的速度。

4.实验结果

4.1.IPv4 分组收发实验结果：



```
0001  /*
0002  * THIS FILE IS FOR IP TEST
0003  */
0004  // system support
0005  #include "sysInclude.h"
0006
0007  extern void ip_DiscardPkt(char* pBuffer)
0008
0009  extern void ip_SendtoLower(char* pBuffer, unsigned short length)
0010
0011  extern void ip_SendtoUp(char* pBuffer, unsigned short length)
0012
0013  extern unsigned int getIpv4Address();
0014
0015  // implemented by students
0016
0017
0018  int stud_ip_rcv(char* pBuffer, unsigned short length)
0019  {
0020
0021      int version = pBuffer[0] >> 4;
```

程序结束

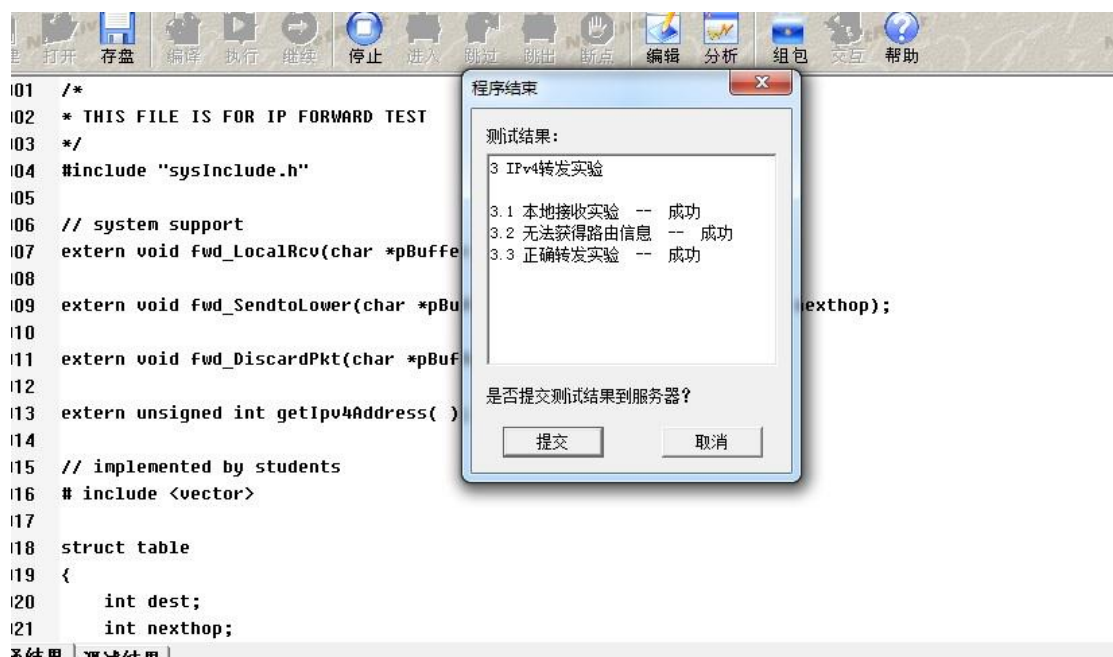
测试结果:

- 2 IPv4收发实验
- 2.1 发送IP包 -- 成功
- 2.2 正确接收IP包 -- 成功
- 2.3 校验和错误的IP包 -- 成功
- 2.4 TTL错误的IP包 -- 成功
- 2.5 版本号错误的IP包 -- 成功
- 2.6 头部长度错误的IP包 -- 成功
- 2.7 错误目标地址的IP包 -- 成功

是否提交测试结果到服务器?

提交 取消

4.2.IPv4 分组转发实验结果：



```
001  /*
002  * THIS FILE IS FOR IP FORWARD TEST
003  */
004  #include "sysInclude.h"
005
006  // system support
007  extern void fwd_LocalRcv(char* pBuffer, unsigned short length)
008
009  extern void fwd_SendtoLower(char* pBuffer, unsigned short length)
010
011  extern void fwd_DiscardPkt(char* pBuffer)
012
013  extern unsigned int getIpv4Address();
014
015  // implemented by students
016  #include <vector>
017
018  struct table
019  {
020      int dest;
021      int nexthop;
```

程序结束

测试结果:

- 3 IPv4转发实验
- 3.1 本地接收实验 -- 成功
- 3.2 无法获得路由信息 -- 成功
- 3.3 正确转发实验 -- 成功

是否提交测试结果到服务器?

提交 取消

