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 | 傅浩东 | | 院系 | 软件工程 | |
| 班级 | 1937102 | | 学号 | 1190202105 | |
| 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 | 2021.11.12 | |
| 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：

本次实验的主要目的：

**1. IPv4 分组收发实验**

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。

**2. IPv4 分组转发实验**

通过前面的实验，我们已经深入了解了 IPv4 协议的分组接收和发送处理流程。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。

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。

实验内容：

概述本次实验的主要内容，包含的实验项等。

**1. IPv4 分组收发实验**

根据计算机网络实验系统所提供的上下层接口函数和协议中分组收发的主要流程，独立设计实现一个简单的 IPv4 分组收发模块。要求实现的主要功能包括：

(1) IPv4 分组的基本接收处理，能够检测出接收到的 IP 分组是否存在如下错误：校验和错、TTL 错、版本号错、头部长度错、错误目标地址；

**(2)** IPv4 分组的封装发送；（不要求实现 IPv4 协议中的选项和分片处理功能）

**2. IPv4 分组转发实验**

在前面 IPv4 分组收发实验的基础上，增加分组转发功能。具体来说，对于每一个到达本机的 IPv4 分组，根据其目的 IPv4 地址决定分组的处理行为，对该分组进行如下的几类操作：

(1) 向上层协议上交目的地址为本机地址的分组；

(2) 根据路由查找结果，丢弃查不到路由的分组；

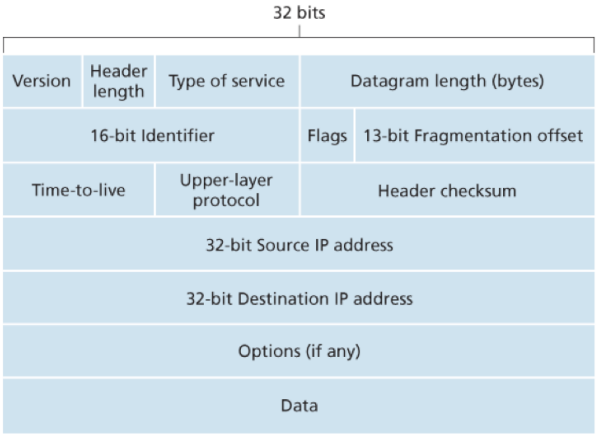
**(3)** 根据路由查找结果，向相应接口转发不是本机接收的分组。

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。

1. **IPv4分组收发实验**

(1) IPv4的头部解析




无论是IPv4的接收还是发送都是基于对IP数据报的读取、改变以及路由等操作，那么实验的第一部分就是清楚地了解到IPv4头部所包含的信息，以及信息所占的位数。

- ✓ 版本号，4比特规定了数据报的IP协议版本；
- ✓ 首部长度。因为一个IPv4数据报首部可能包含一些可变数量的选项，故需要用这4比特来确定1P数据报中载荷，一般的IP数据报具有20字节的首部；
- ✓ 服务类型；
- ✓ 数据报长度。IP数据报的总长度（首部加上数据），以字节计。该字段长为16比特，所以IP数据报的理论最大长度为65535字节；
- ✓ 标识、标志、片偏移。这三个字段与所谓IP分片有关，本实验不做考虑；
- ✓ TTL寿命，若TTL字段减为0，则该数据报必须丢弃，8bits；
- ✓ 协议，该字段值指示了IP数据报的数据部分应交给哪个特定的运输层协议；
- ✓ 首部检验和 checksum，16bits；
- ✓ 源和目的IP地址，各32bits；
- ✓ 选项；
- ✓ 数据（有效载荷）。

(2) IPv4分组数据报的接收

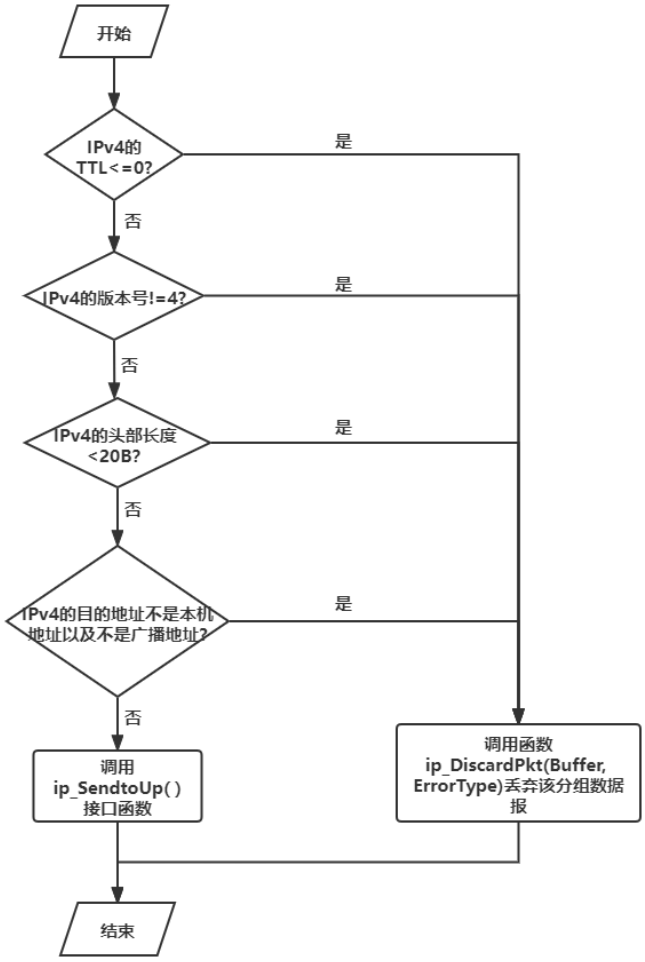
接收数据报过程：

A. 首先，提取IPv4数据报头部的某些字段，包括版本号（Version）、头部长度（IP Head length）、生存时间（TTL）以及头校验和（Header checksum）等字段；

B. 检查接收到的 IPv4 分组头部字段的正确性（版本号、头部长度、生存时间、目的地址是否为本机地址或者广播地址以及头校验和）。对于出错的分组调用函数ip_DiscardPkt(Buffer, ErrorType)丢弃，并且说明错误类型；

C. 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。

那么给出如下流程图：




代码结构：

```
if (ttl <= 0) {
    ip_DiscardPkt(pBuffer, STUD_IP_TEST_TTL_ERROR);
    return 1;
}
if (ver != 4) {
    ip_DiscardPkt(pBuffer, STUD_IP_TEST_VERSION_ERROR);
    return 1;
}
if (head_length < 5) {
    ip_DiscardPkt(pBuffer, STUD_IP_TEST_HEADLEN_ERROR);
    return 1;
}
if (dest != getIpv4Address() && dest != 0xffff) {
    ip_DiscardPkt(pBuffer, STUD_IP_TEST_DESTINATION_ERROR);
    return 1;
}
unsigned short sum = 0;
unsigned short temp = 0;
```

```
        for (int i = 0; i < head_length * 2; i++) {
            temp = 0;
            temp += ((unsigned char)pBuffer[i * 2] << 8);
            temp += (unsigned char)pBuffer[i * 2 + 1];
            if (0xFFFF - sum < temp)
                sum = sum + temp + 1;
            else
                sum = sum + temp;
        }
        if (sum != 0xffff) {
            ip_DiscardPkt(pBuffer, STUD_IP_TEST_CHECKSUM_ERROR);
            return 1;
        }
```

(3) IPv4分组数据报的发送

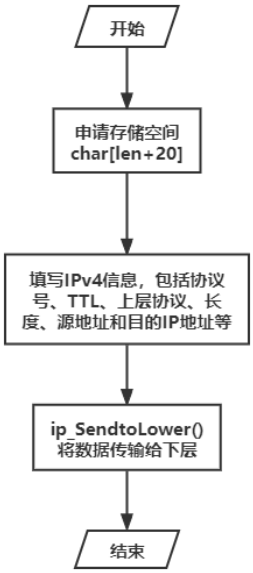
发送数据报过程：

A. 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；

B. 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）；

C. 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。

给出流程图：




关键代码：

```
    memcpy(buffer + 2, &total_length, 2);
    unsigned int src = htonl(srcAddr);
    unsigned int dst = htonl(dstAddr);
    memcpy(buffer + 12, &src, 4);
    memcpy(buffer + 16, &dst, 4);
    unsigned short head_checksum = htons(0xffff - sum);
```

```
    memcpy(buffer + 10, &head_checksum, 2);
    memcpy(buffer + 20, pBuffer, len);
```

## 2. IPv4 分组转发实验

### (1) 首先定义路由条目

首先对于路由表的每一个条目，都有目的 IP 地址、掩码（掩码长度）以及下一跳，路由就是根据目的 IP 地址和掩码的最长匹配结果来选择下一跳，因此我们定义路由表的每一条有如下：

```
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
};
```

(2) 路由表初始化

实验中，一个路由表是用 vector<routeTableItem>来表示的，因此初始化路由表就是将其清空，在这里就不再赘述。

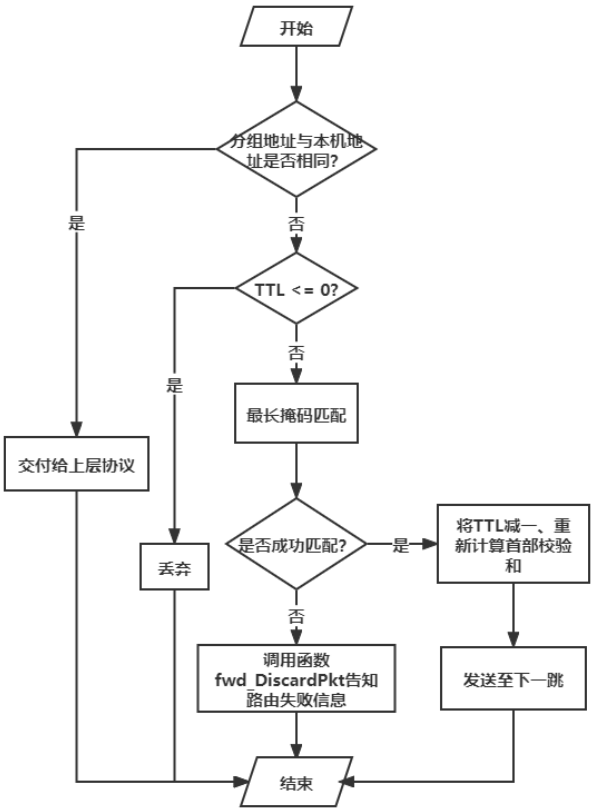
(3) 根据路由分组转发

A. 首先判断该分组地址与本机是否相同，如果相同则将其向上层交付；

B. 接着判断存活时间是否还能支持继续转发，如果不能则直接丢弃该分组数据报；

C. 根据最长匹配在路由表中查找是否有与该数据报目的地址匹配的路由条目，如果有则向相应接口转发不是本机接收的分组，否则丢弃查不到路由的分组。

这里给出该过程的流程图：



关键代码：

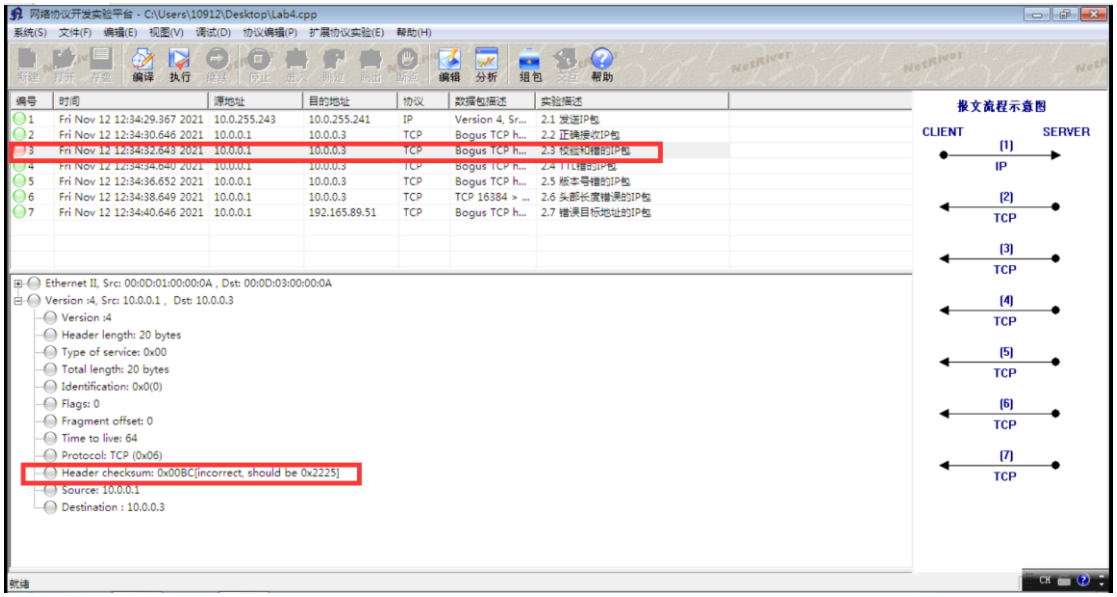
```
    for(int i = 0; i < m_table.size(); i ++) {
        if(m_table[i].masklen > longestMatchLen && m_table[i].destIP == (DestIP &
m_table[i].mask)) {
            bestMatch = i;
            Match = true;
            longestMatchLen = m_table[i].masklen;
        }
    }
    if(Match) {
        char *buffer = new char[length];
        memcpy(buffer,pBuffer,length);
        // TTL - 1
        buffer[8]--;
        int sum = 0;
        unsigned short int localCheckSum = 0;
        for(int j = 1; j < 2 * headsum +1; j ++) {
            if (j != 6) {
                sum = sum + (buffer[(j-1)*2]<<8)+(buffer[(j-1)*2+1]);
                sum %= 65535;
            }
        }
        // checksum
        localCheckSum = htons(~(unsigned short int)sum);
        memcpy(buffer+10, &localCheckSum, sizeof(unsigned short));
        // Send to Lower protocol
        fwd_SendtoLower(buffer, length, m_table[bestMatch].nexthop);
        return 0;
    }
    else {
        fwd_DiscardPkt(pBuffer, STUD_FORWARD_TEST_NOROUTE);
        return 1;
    }
```

实验结果：

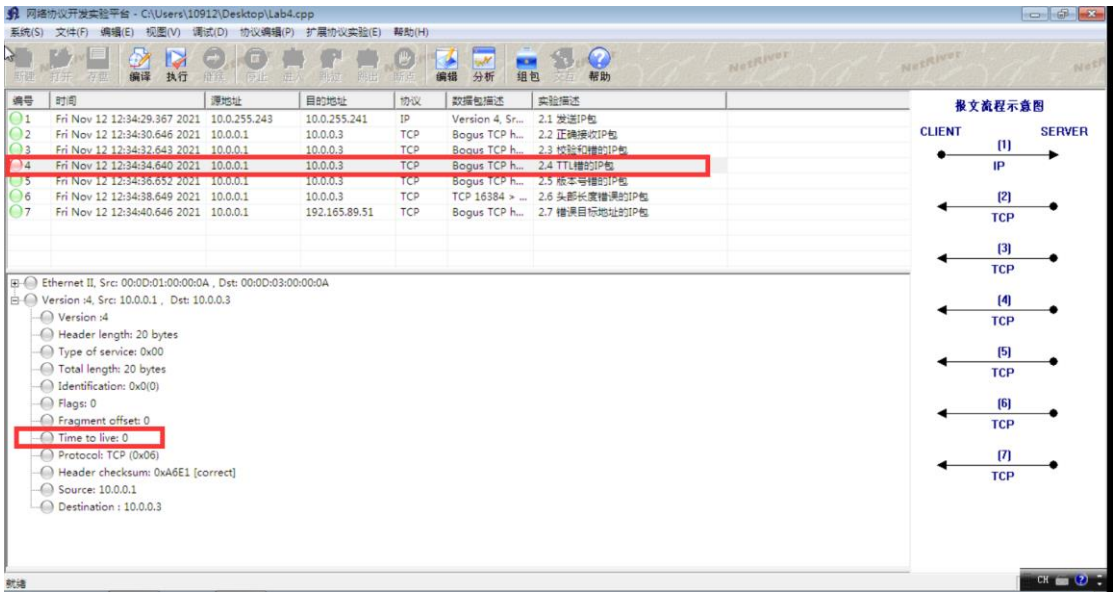
采用演示截图、文字说明等方式，给出本次实验的实验结果。

1. IPv4分组收发实验

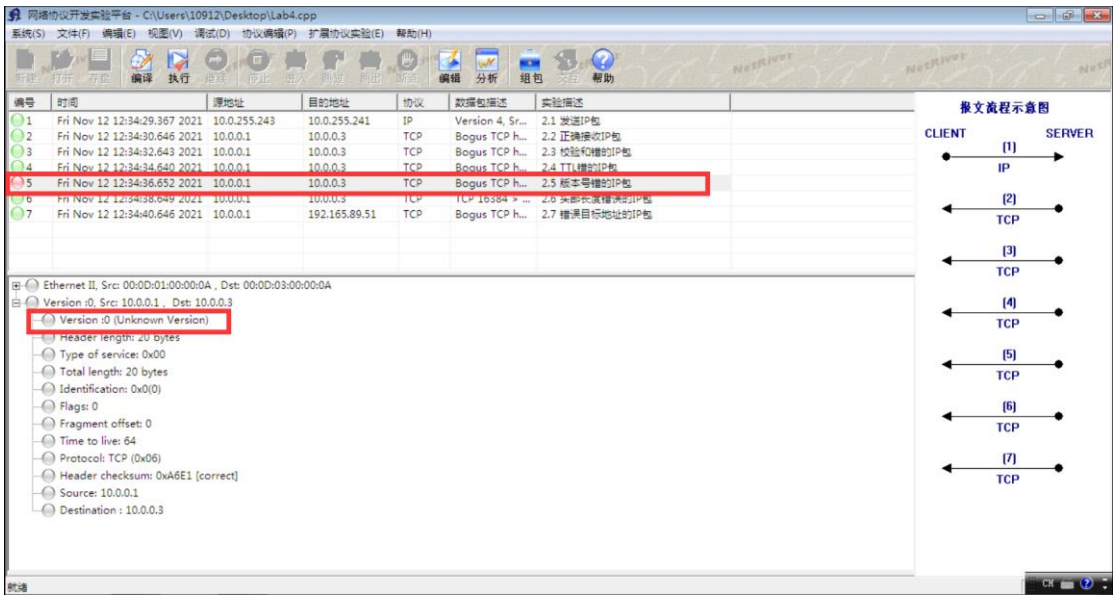
(1) 首部校验和，应该为0x2225，但计算得到的是0x00BC



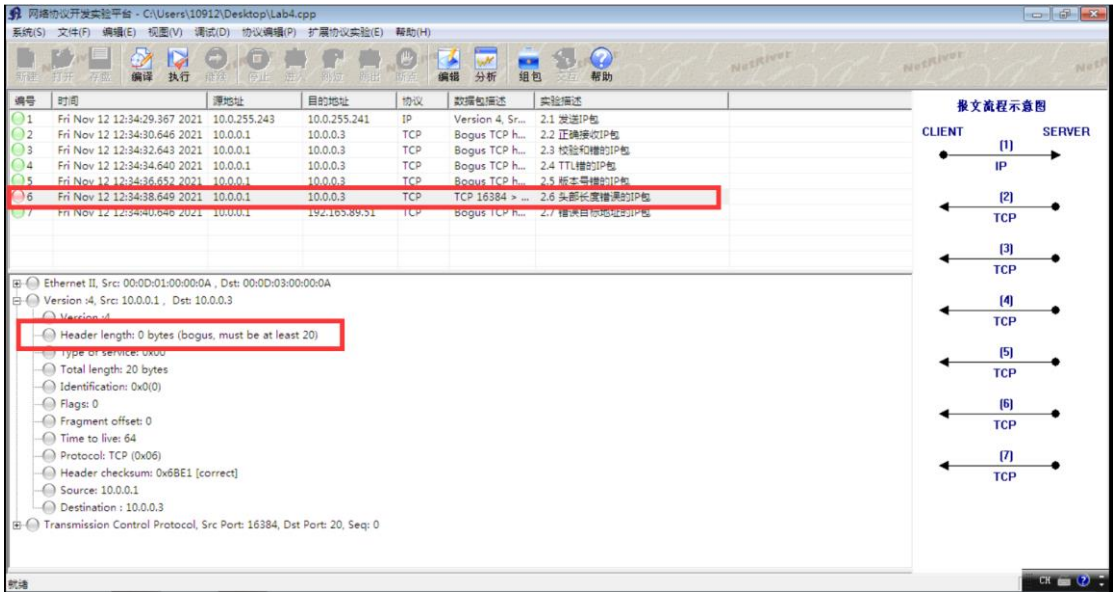

(2) TTL生存时间错误，生存时间变为0，IP包被丢弃



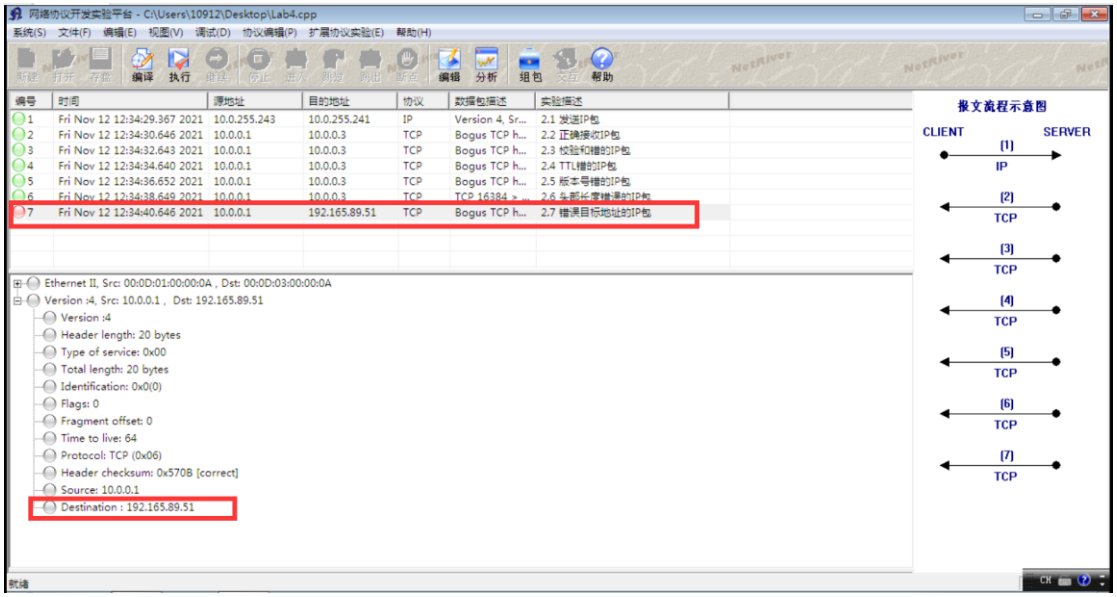

(3) 版本号错误，版本号应该是4，但这里是0（未知版本号）

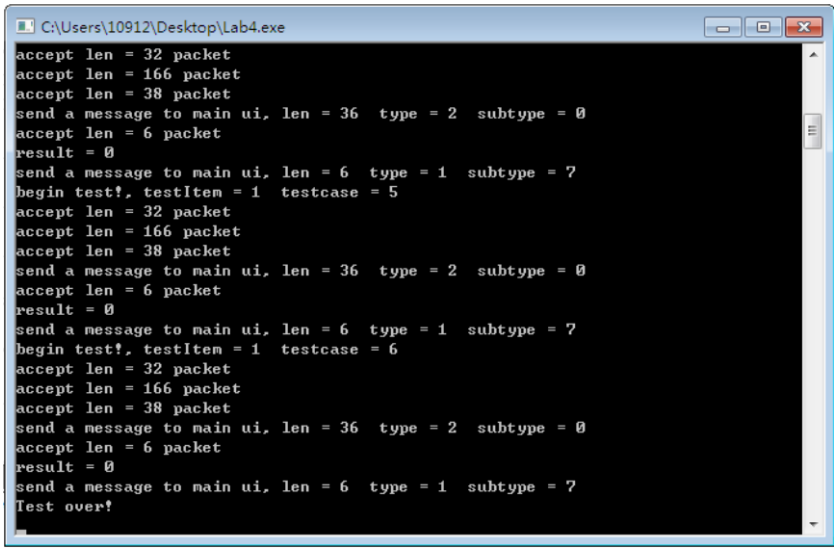


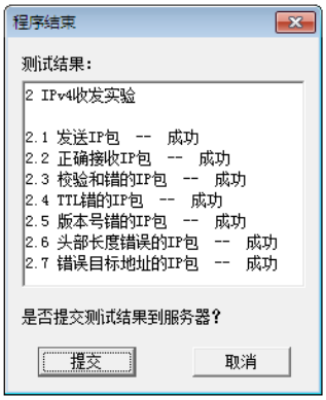

(4) 头部长度错误，头部长度至少为20Bytes，但这里头部长度标志位为0.



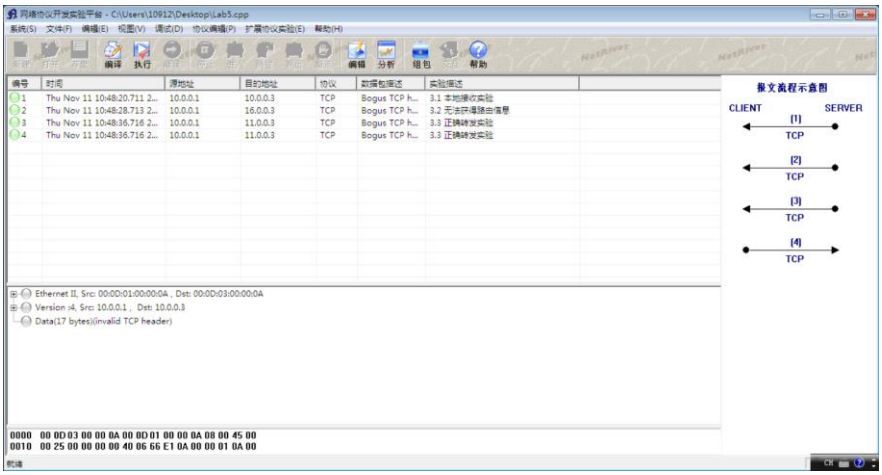

(5) 目标地址错误，目标地址错误地记为192.165.89.51

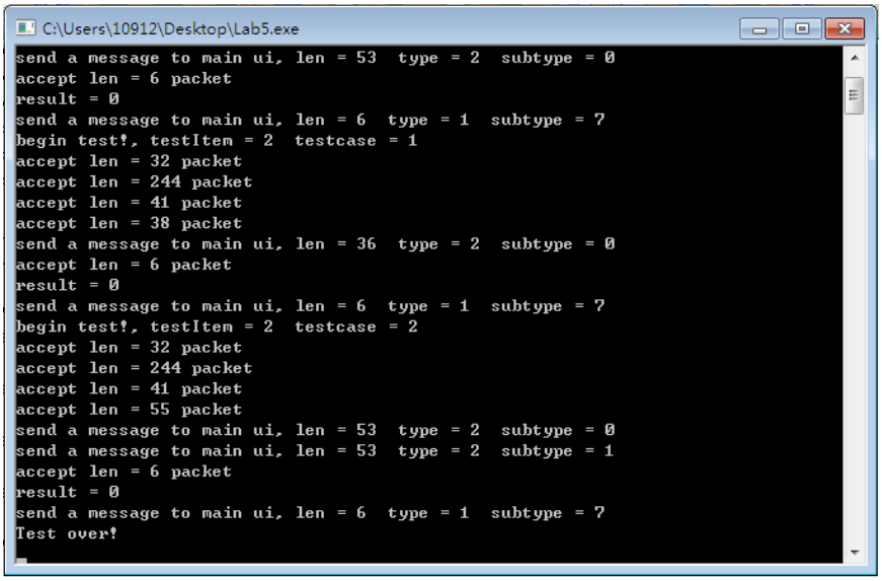


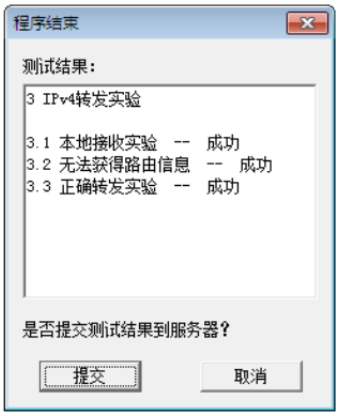




2. IPv4分组转发实验




3. 提交结果

| 学期 | 序号 | 学号 | 姓名 | 院系 | 班级 | 实验名称 | 实验日期 | 实验结果 | 总成绩 | 程序 | 报告 |
|---|---|---|---|---|---|---|---|---|---|---|---|
| 2021年秋季 | 1 | 1190202105 | 傅浩东 | 软件工程 | 1937102 | IPv4收发实验 | 2021-11-09 | ■■■■■■■■■■■■■■■■ | 10 | | |
| 2021年秋季 | 2 | 1190202105 | 傅浩东 | 软件工程 | 1937102 | IPv4转发实验 | 2021-11-09 | ■■■■■■■■■■■■■■■■ | 10 | | |

问题讨论：

- ✓ 路由表最长匹配查询，最简单的方式就是将路由表进行随机的存储，并依次查询每一个表；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)；若是按照掩码长度降序排列，则第一次匹配到的路由条目则是最后所求的路由项。
- ✓ 各个判断条件的优先级问题：先判定TTL还是先判定IP地址是否符合等问题，这些判定顺序有的会直接影响接收转发结果，而有的则会降低效率，例如checksum放在太靠前了则对于TTL<=0等情况，效率就会降低。
- ✓ 数据溢出：由于数据报头部每个内容长度都有不同，而有的内容例如checksum会有溢出处理操作，所以对于数据类型的选择也是需要深思熟虑。

心得体会：

结合实验过程和结果给出实验的体会和收获。

本实验通过设计实现主机协议栈中的 IPv4 协议，了解了网络层协议的基本原理，学习了 IPv4 协议基本的分组接收和发送流程。另外，通过本实验，我还初步接触互联网协议栈的结构和计算机网络实验系统。我也初步接触了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。