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 分组收发实验&&转发实验					
姓名	孙月晴		院系	计算机科学与技术		
班级	1603104		学号	1160300901		
任课教师	刘亚维		指导教师	刘亚维		
实验地点	格物 213		实验时间	2018 年 10 月 31 日		
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 1：

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。

实验目的 2:

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：

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。

实验内容2:

1) 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。

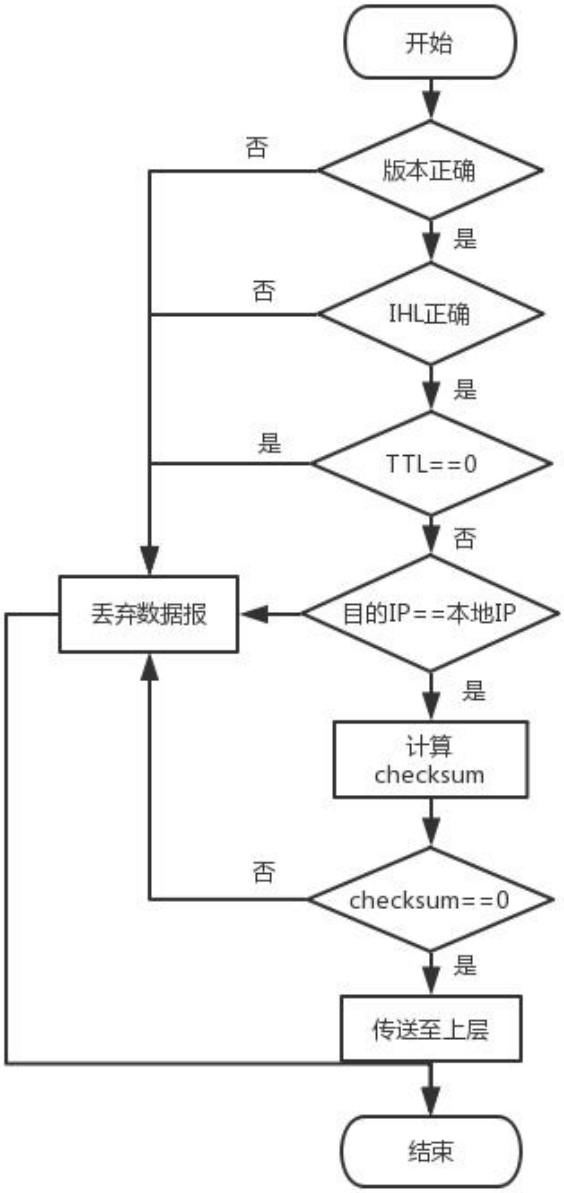
2) IPv4 分组的接收和发送。对前面实验（IP实验）中所完成的代码进行修改，在路由器协议栈的IPv4模块中能够正确完成分组的接收和发送处理。具体要求不做改变，参见“IP实验”。

3) 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。

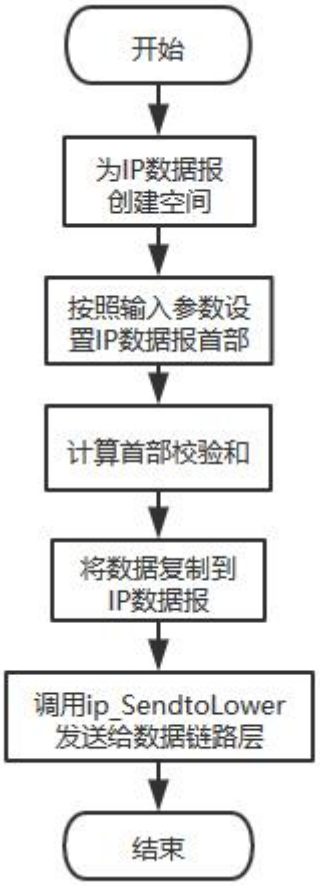
实验过程 1：

1) 发送和接收函数的实现程序流程图；

接收函数stud_ip_recv( ):



发送函数stud_ip_Upsend( ):



2) 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

1. 版本号校验

版本号在第一个字节的前 4 位，取 pBuffer[0]和 0xF0 做与运算，如果结果依旧是 0x40，则代表版本号正确

```
if ((pBuffer[0] & 0xf0) != 0x40) {
    ip_DiscardPkt(pBuffer, STUD_IP_TEST_VERSION_ERROR);
    return 1;
}
```

2. 头部长度校验

头部长度在第一个字节的后 4 位里面，则 pBuffer[0]和 0x0F 做与运算，如果结果为 0x05，则代表头部长度没有问题

```
if ((pBuffer[0] & 0x0f) != 0x05) {
    ip_DiscardPkt(pBuffer, STUD_IP_TEST_HEADLEN_ERROR);
    return 1;
}
```

3. 生存时间校验

TTL 存在于第 9 个字节里面，按照规定，如果 ttl 的值是 0，则代表生命周期结束，要抛弃这个包

```
if (pBuffer[8] == 0x00) {
```

```
        ip_DiscardPkt(pBuffer, STUD_IP_TEST_TTL_ERROR);
        return 1;
    }
```

4. 目标地址校验

目标地址存在于首部的第 17 -20 个字节中，和本地 ip 地址做比较，如果不等于本地地址，则表示目标地址出错。

```
    unsigned int address = getIpv4Address();
    unsigned int *intAddress = (unsigned int *)(pBuffer + 16);
    if (address != ntohl(*intAddress)) {
        ip_DiscardPkt(pBuffer, STUD_IP_TEST_DESTINATION_ERROR);
        return 1;
    }
```

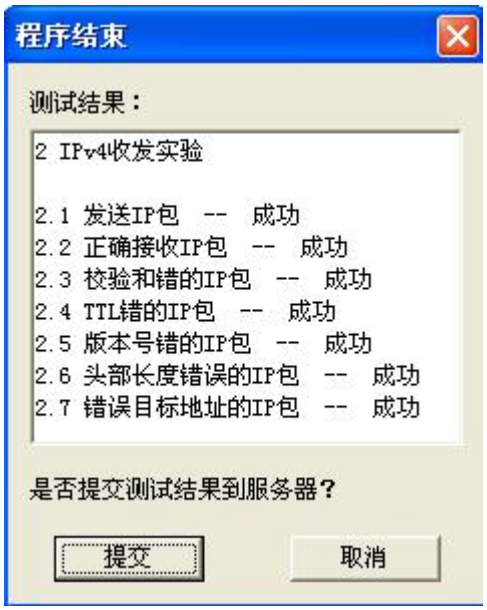
5. 校验和

校验和存在于 11-12 个字节，校验和检测的规则如下：16 进制反码求和，也就是说是将所有的字节加起来（校验和部分忽略，即为 0），然后用 0xFFFF 减去。

```
unsigned short checksum(unsigned short *buffer, int len) {
    unsigned long cksum = 0;
    while (len > 1) {
        cksum += *buffer++;
        len -= sizeof(unsigned short);
    }
    if (len) {
        cksum += *(unsigned char *)buffer;
    }
    cksum = (cksum >> 16) + (cksum & 0xffff);
    cksum += (cksum >> 16);
    return (unsigned short)(~cksum);
}
```

实验结果 1：

1. 程序运行结果:



2. 错误的版本号:

编号	时间	源地址	目的地址	协议	数据包描述	实验描述
1	Wed Nov 07 11:54:54.875 2018	10.0.255.243	10.0.255.241	IP	Version 4, S...	2.1 发送IP包
2	Wed Nov 07 11:54:56.046 2018	10.0.0.1	10.0.0.4	TCP	Bogus TCP he...	2.2 正确接收IP包
3	Wed Nov 07 11:54:58.046 2018	10.0.0.1	10.0.0.4	TCP	Bogus TCP he...	2.3 校验和错的IP包
4	Wed Nov 07 11:55:00.046 2018	10.0.0.1	10.0.0.4	TCP	Bogus TCP he...	2.4 TTL错的IP包
5	Wed Nov 07 11:55:02.046 2018	10.0.0.1	10.0.0.4	TCP	Bogus TCP he...	2.5 版本号错的IP包
6	Wed Nov 07 11:55:04.046 2018	10.0.0.1	10.0.0.4	TCP	Bogus TCP he...	2.6 头部长度错误的IP包
7	Wed Nov 07 11:55:06.046 2018	10.0.0.1	192.166.77.9	TCP	Bogus TCP he...	2.7 错误目标地址的IP包

3. 错误的头部长度

编号	时间	源地址	目的地址	协议	数据包描述	实验描述
1	Wed Nov 07 11:54:54.875 2018	10.0.255.243	10.0.255.241	IP	Version 4,...	2.1 发送IP包
2	Wed Nov 07 11:54:56.046 2018	10.0.0.1	10.0.0.4	TCP	Bogus TCP ...	2.2 正确接收IP包
3	Wed Nov 07 11:54:58.046 2018	10.0.0.1	10.0.0.4	TCP	Bogus TCP ...	2.3 校验和错的IP包
4	Wed Nov 07 11:55:00.046 2018	10.0.0.1	10.0.0.4	TCP	Bogus TCP ...	2.4 TTL错的IP包
5	Wed Nov 07 11:55:02.046 2018	10.0.0.1	10.0.0.4	TCP	Bogus TCP ...	2.5 版本号错的IP包
6	Wed Nov 07 11:55:04.046 2018	10.0.0.1	10.0.0.4	TCP	Bogus TCP ...	2.6 头部长度错误的IP包
7	Wed Nov 07 11:55:06.046 2018	10.0.0.1	192.166.77.9	TCP	Bogus TCP ...	2.7 错误目标地址的IP包

```
Version :4, Src: 10.0.0.1 ,  Dst: 10.0.0.4
    Version :4
    Header length: 4 bytes (bogus, must be at least 20)
    Type of service: 0x00
    Total length: 20 bytes
    Identification: 0x0(0)
    Flags: 0
    Fragment offset: 0
```

```
0000  00 0D 04 00 00 0A 00 0D 01 00 00 0A 08 00 41 00
0010  00 14 00 00 00 00 40 06 6A E0 0A 00 00 01 0A 00
0020  00 04
```

4. 错误的 TTL，例如 0，即“00000000”

编号	时间	源地址	目的地址	协议	数据包描述	实验描述
1	Wed Nov 07 11:54:54.875 2018	10.0.255.243	10.0.255.241	IP	Version 4,...	2.1 发送IP包
2	Wed Nov 07 11:54:56.046 2018	10.0.0.1	10.0.0.4	TCP	Bogus TCP ...	2.2 正确接收IP包
3	Wed Nov 07 11:54:58.046 2018	10.0.0.1	10.0.0.4	TCP	Bogus TCP ...	2.3 校验和错的IP包
4	Wed Nov 07 11:55:00.046 2018	10.0.0.1	10.0.0.4	TCP	Bogus TCP ...	2.4 TTL错的IP包
5	Wed Nov 07 11:55:02.046 2018	10.0.0.1	10.0.0.4	TCP	Bogus TCP ...	2.5 版本号错的IP包
6	Wed Nov 07 11:55:04.046 2018	10.0.0.1	10.0.0.4	TCP	Bogus TCP ...	2.6 头部长度错误的IP包
7	Wed Nov 07 11:55:06.046 2018	10.0.0.1	192.166.77.9	TCP	Bogus TCP ...	2.7 错误目标地址的IP包

```
    Identification: 0x0(0)
    Flags: 0
    Fragment offset: 0
    Time to live: 0
    Protocol: TCP (0x06)
    Header checksum: 0xA6E0 [correct]
    Source: 10.0.0.1
    Destination : 10.0.0.4
```

```
0000  00 0D 04 00 00 0A 00 0D 01 00 00 0A 08 00 45 00
0010  00 14 00 00 00 00 00 06 A6 E0 0A 00 00 01 0A 00
0020  00 04
```

5. 错误的目标地址，例如“192.166.77.9”

编号	时间	源地址	目的地址	协议	数据包描述	实验描述
1	Wed Nov 07 11:54:54.875 2018	10.0.255.243	10.0.255.241	IP	Version 4,...	2.1 发送IP包
2	Wed Nov 07 11:54:56.046 2018	10.0.0.1	10.0.0.4	TCP	Bogus TCP ...	2.2 正确接收IP包
3	Wed Nov 07 11:54:58.046 2018	10.0.0.1	10.0.0.4	TCP	Bogus TCP ...	2.3 校验和错的IP包
4	Wed Nov 07 11:55:00.046 2018	10.0.0.1	10.0.0.4	TCP	Bogus TCP ...	2.4 TTL错的IP包
5	Wed Nov 07 11:55:02.046 2018	10.0.0.1	10.0.0.4	TCP	Bogus TCP ...	2.5 版本号错的IP包
6	Wed Nov 07 11:55:04.046 2018	10.0.0.1	10.0.0.4	TCP	Bogus TCP ...	2.6 头部长度错误的IP包
7	Wed Nov 07 11:55:06.046 2018	10.0.0.1	192.166.77.9	TCP	Bogus TCP ...	2.7 错误目标地址的IP包

```
Identification: 0x0 (0)
Flags: 0
Fragment offset: 0
Time to live: 64
Protocol: TCP (0x06)
Header checksum: 0x6334 [correct]
Source: 10.0.0.1
Destination : 192.166.77.9
```

```
0000  00 0D 04 00 00 0A 00 0D 01 00 00 0A 08 00 45 00
0010  00 14 00 00 00 00 40 06 63 34 0A 00 00 01 C0 A6
0020  4D 09
```

6. 错误的校验和

编号	时间	源地址	目的地址	协议	数据包描述	实验描述
1	Wed Nov 07 11:54:54.875 2018	10.0.255.243	10.0.255.241	IP	Version 4,...	2.1 发送IP包
2	Wed Nov 07 11:54:56.046 2018	10.0.0.1	10.0.0.4	TCP	Bogus TCP ...	2.2 正确接收IP包
3	Wed Nov 07 11:54:58.046 2018	10.0.0.1	10.0.0.4	TCP	Bogus TCP ...	2.3 校验和错的IP包
4	Wed Nov 07 11:55:00.046 2018	10.0.0.1	10.0.0.4	TCP	Bogus TCP ...	2.4 TTL错的IP包
5	Wed Nov 07 11:55:02.046 2018	10.0.0.1	10.0.0.4	TCP	Bogus TCP ...	2.5 版本号错的IP包
6	Wed Nov 07 11:55:04.046 2018	10.0.0.1	10.0.0.4	TCP	Bogus TCP ...	2.6 头部长度错误的IP包
7	Wed Nov 07 11:55:06.046 2018	10.0.0.1	192.166.77.9	TCP	Bogus TCP ...	2.7 错误目标地址的IP包

```
Identification: 0x0 (0)
Flags: 0
Fragment offset: 0
Time to live: 64
Protocol: TCP (0x06)
Header checksum: 0x03E8[incorrect, should be 0x4AFB]
Source: 10.0.0.1
Destination : 10.0.0.4
```

```
0000  00 0D 04 00 00 0A 00 0D 01 00 00 0A 08 00 45 00
0010  00 14 00 00 00 00 40 06 03 E8 0A 00 00 01 0A 00
0020  00 04
```

实验过程2：

1) 路由表初始化、路由增加、路由转发三个函数的实现流程图

1. 路由表初始化 stud_Route_Init ( )

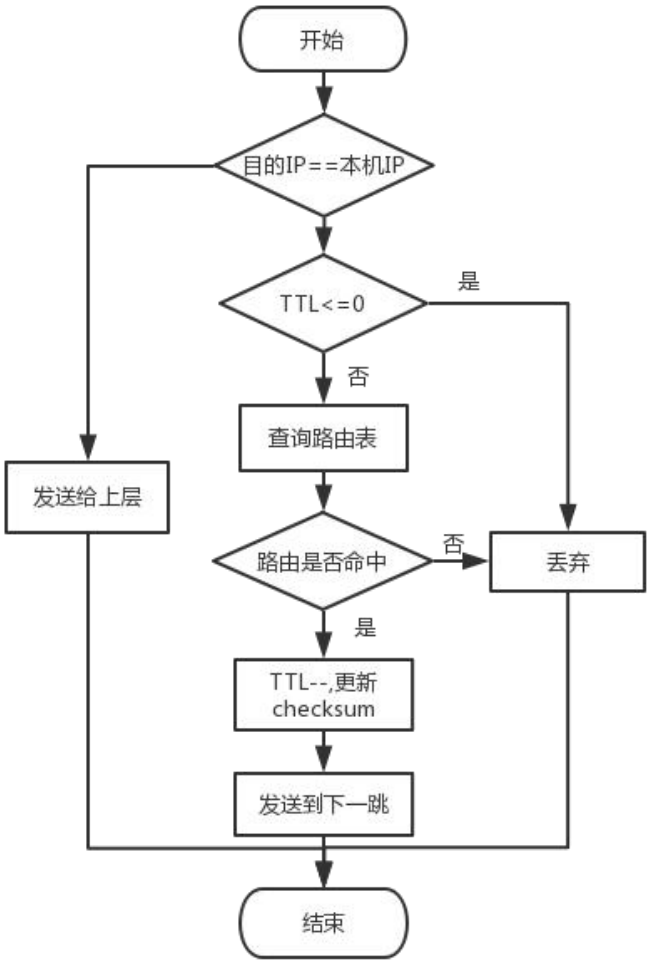
2. 路由增加 stud_route_add ( )

从 stud_route_msg 结构中取得 dest, masklen, nexthop，转为网络字节序之后经过处理，构建结构体 route，并添加到 vector 中。

3. 路由转发 stud_fwd_deal ( )

查找路由表。根据相应路由表项的类型来确定下一步操作，错误分组调用函数 fwd_DiscardPkt ( )进行丢弃，上交分组调用接口函数 fwd_LocalRcv ( )提交给上层协议继续处理，转发分组进行转发处理。注意，转发分组还要从路由表项中获取下一跳的 IPv4 地址。

转发处理流程。对 IPv4 头部中的 TTL 字段减 1，重新计算校验和，然后调用下层接口 fwd_SendtoLower ( )进行发送处理。



2) 新建数据结构说明

```
struct routeTable
{
    int dest;
    int masklen;
    int nexthop;
    routeTable(int d = 0, int m = 0, int n = 0) :
        dest(d), masklen(m), nexthop(n)
    {}
};

vector<routeTable> routeTable; //路由器转发表
```

3) 存在大量分组的情况下如何提高转发效率

1. 树形结构匹配路由表项：路由表存储结构由线性结构改为树形结构，如红黑树，将查询时间从 O(n)降低到 O(logn)，提高匹配效率。

2. 缓存分组：设置路由表缓存，利用 LRU 算法，缓存最近使用的路由表项，如果该分组的目的地址和源地址与转发缓存中的匹配，则直接根据转发缓存中的下一网关地址进行转发，提高路由器吞吐量。

实验结果 2:

心得体会：

通过设计实现主机协议栈中的IPv4协议，我更加深入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。在IPv4分组转发实验中,设计模拟实现了路由器中的 IPv4 协议，在原有IPv4分组收发实验的基础上，增加了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，了解了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的。同时也意识到路由表的结构对转发效率也有很大的影响。