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	郑旭然		院系	软件工程	
班级	2037102		学号	120L020719	
任课教师	李全龙		指导教师	李全龙	
实验地点	格物楼 207		实验时间	2022.10.17	
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 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。

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：

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：

1) 发送和接收函数的实现程序流程图

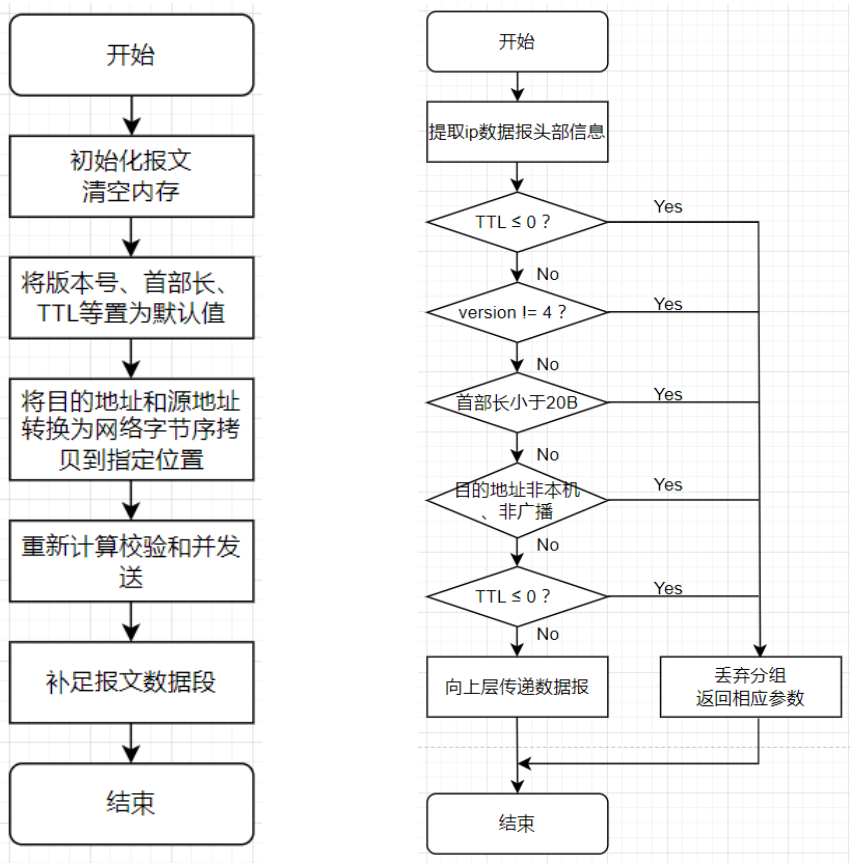


图 1 发送和接收函数的实现程序流程图

2) 新建的数据结构的说明（无）

3) 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，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

版本号：pBuffer[0]取出版本号的那四位，检测版本号是否为4。收到的错误版本号为3。

```
// 检测version错误
int version = pBuffer[0] >> 4;
if (version != 4)
{
  errorType = STUD_IP_TEST_VERSION_ERROR;
  ip_DiscardPkt(pBuffer, errorType);
  return 1;
}
```

编号	时间	源地址	目的地址	协议	数据包描述	实验描述
1	Sat Oct 15 23:56:34.359 2022	10.0.255.243	10.0.255.241	IP	Version 4, ...	2.1 发送IP包
2	Sat Oct 15 23:56:36.359 2022	10.0.0.1	10.0.0.3	TCP	Bogus TCP h...	2.2 正确接收IP包
3	Sat Oct 15 23:56:38.359 2022	10.0.0.1	10.0.0.3	TCP	Bogus TCP h...	2.3 校验和错的IP包
4	Sat Oct 15 23:56:40.421 2022	10.0.0.1	10.0.0.3	TCP	Bogus TCP h...	2.4 TTL错的IP包
5	Sat Oct 15 23:56:42.390 2022	10.0.0.1	10.0.0.3	TCP	Bogus TCP h...	2.5 版本号错的IP包
6	Sat Oct 15 23:56:44.421 2022	10.0.0.1	10.0.0.3	TCP	Bogus TCP h...	2.6 头部长度错误的IP包
7	Sat Oct 15 23:56:46.421 2022	10.0.0.1	192.167.173.8	TCP	Bogus TCP h...	2.7 错误目标地址的IP包

```
Ethernet II, Src: 00:0D:01:00:00:0A , Dst: 00:0D:03:00:00:0A
    Destination : 00:0D:03:00:00:0A
    Source : 00:0D:01:00:00:0A
    TYPE : IP (0x0800)
Version :3, Src: 10.0.0.1 ,  Dst: 10.0.0.3
    Version :3 (Unknown Version)
    Header length: 20 bytes
    Type of service: 0x00
    Total length: 20 bytes
    Identification: 0x0 (0)
    Flags: 0
    Fragment offset: 0
    Time to live: 64
    Protocol: TCP (0x06)
    Header checksum: 0x76E1 [correct]
```

```
0000  00 0D 03 00 00 0A 00 0D 01 00 00 0A 08 00 35 00
0010  00 14 00 00 00 00 40 06 76 E1 0A 00 00 01 0A 00
0020  00 03
```

头部长度：与0xF进行与运算得到首部长。如果首部长小于5字节，则丢弃该分组。收到的错误头部长度为2。

```
// 检测headerLength错误
int headerLength = pBuffer[0] & 0xF;
if (headerLength < 5)
{
  errorType = STUD_IP_TEST_HEADLEN_ERROR;
  ip_DiscardPkt(pBuffer, errorType);
  return 1;
}
```

、

编号	时间	源地址	目的地址	协议	数据包描述	实验描述
1	Sat Oct 15 23:56:34.359 2022	10.0.255.243	10.0.255.241	IP	Version 4, ...	2.1 发送IP包
2	Sat Oct 15 23:56:36.359 2022	10.0.0.1	10.0.0.3	TCP	Bogus TCP h...	2.2 正确接收IP包
3	Sat Oct 15 23:56:38.359 2022	10.0.0.1	10.0.0.3	TCP	Bogus TCP h...	2.3 校验和错的IP包
4	Sat Oct 15 23:56:40.421 2022	10.0.0.1	10.0.0.3	TCP	Bogus TCP h...	2.4 TTL错的IP包
5	Sat Oct 15 23:56:42.390 2022	10.0.0.1	10.0.0.3	TCP	Bogus TCP h...	2.5 版本号错的IP包
6	Sat Oct 15 23:56:44.421 2022	10.0.0.1	10.0.0.3	TCP	Bogus TCP h...	2.6 头部长度错误的IP包
7	Sat Oct 15 23:56:46.421 2022	10.0.0.1	192.167.173.8	TCP	Bogus TCP h...	2.7 错误目标地址的IP包

```
Ethernet II, Src: 00:0D:01:00:00:0A , Dst: 00:0D:03:00:00:0A
Version :4, Src: 10.0.0.1 ,  Dst: 10.0.0.3
    Version :4
    Header length: 8 bytes (bogus, must be at least 20)
    Type of service: 0x00
    Total length: 20 bytes
    Identification: 0x0 (0)
    Flags: 0
    Fragment offset: 0
    Time to live: 64
    Protocol: TCP (0x06)
    Header checksum: 0x69E1 [correct]
    Source: 10.0.0.1
    Destination : 10.0.0.3
Data(12 bytes)(invalid TCP header)
```

```
0000  00 0D 03 00 00 0A 00 0D 01 00 00 0A 08 00 42 00
0010  00 14 00 00 00 00 40 06 69 E1 0A 00 00 01 0A 00
0020  00 03
```

生存时间：生存时间如果为负数或0则丢弃该分组。收到的错误生存时间为0。

```
// 检测TTL错误
int ttl = (unsigned short)pBuffer[8];
if (ttl <= 0)
{
  errorType = STUD_IP_TEST_TTL_ERROR;
  ip_DiscardPkt(pBuffer, errorType);
  return 1;
}
```

编号	时间	源地址	目的地址	协议	数据包描述	实验描述
1	Sat Oct 15 23:56:34.359 2022	10.0.255.243	10.0.255.241	IP	Version 4, ...	2.1 发送IP包
2	Sat Oct 15 23:56:36.359 2022	10.0.0.1	10.0.0.3	TCP	Bogus TCP h...	2.2 正确接收IP包
3	Sat Oct 15 23:56:38.359 2022	10.0.0.1	10.0.0.3	TCP	Bogus TCP h...	2.3 校验和错的IP包
4	Sat Oct 15 23:56:40.421 2022	10.0.0.1	10.0.0.3	TCP	Bogus TCP h...	2.4 TTL错的IP包
5	Sat Oct 15 23:56:42.390 2022	10.0.0.1	10.0.0.3	TCP	Bogus TCP h...	2.5 版本号错的IP包
6	Sat Oct 15 23:56:44.421 2022	10.0.0.1	10.0.0.3	TCP	Bogus TCP h...	2.6 头部长度错误的IP包
7	Sat Oct 15 23:56:46.421 2022	10.0.0.1	192.167.173.8	TCP	Bogus TCP h...	2.7 错误目标地址的IP包

```
Ethernet II, Src: 00:0D:01:00:00:0A , Dst: 00:0D:03:00:00:0A
Version :4, Src: 10.0.0.1 ,  Dst: 10.0.0.3
    Version :4
    Header length: 20 bytes
    Type of service: 0x00
    Total length: 20 bytes
    Identification: 0x0(0)
    Flags: 0
    Fragment offset: 0
    Time to live: 0
    Protocol: TCP (0x06)
    Header checksum: 0xA6E1 [correct]
    Source: 10.0.0.1
    Destination : 10.0.0.3
```

```
0000  00 0D 03 00 00 0A 00 0D 01 00 00 0A 08 00 45 00
0010  00 14 00 00 00 00 00 06 A6 E1 0A 00 00 01 0A 00
0020  00 03
```

首部校验和：16进制反码求和，和的反码即为首部校验和。收到的错误首部校验和为0x029A。

```
// 检测checkSum错误
unsigned short sum = 0;
unsigned short temp = 0;
for (int i = 0; i < headerLength * 2; i++)
{
  // <<8表示其做高2位
  temp = ((unsigned char)pBuffer[i * 2] << 8) + (unsigned char)pBuffer[i * 2 + 1];
  // 若结果>0xFFFF，则将高16位加在低16位上
  if (sum + temp > 0xFFFF)
    sum += 1;
  sum += temp;
}
if (sum != 0xFFFF) // 计算结果 ≠ FFFF，说明数据报发生错误
{
  errorType = STUD_IP_TEST_CHECKSUM_ERROR;
  ip_DiscardPkt(pBuffer, errorType);
  return 1;
}
```

编号	时间	源地址	目的地址	协议	数据包描述	实验描述
1	Sat Oct 15 23:56:34.359 2022	10.0.255.243	10.0.255.241	IP	Version 4, ...	2.1 发送IP包
2	Sat Oct 15 23:56:36.359 2022	10.0.0.1	10.0.0.3	TCP	Bogus TCP h...	2.2 正确接收IP包
3	Sat Oct 15 23:56:38.359 2022	10.0.0.1	10.0.0.3	TCP	Bogus TCP h...	2.3 校验和错的IP包
4	Sat Oct 15 23:56:40.421 2022	10.0.0.1	10.0.0.3	TCP	Bogus TCP h...	2.4 TTL错的IP包
5	Sat Oct 15 23:56:42.390 2022	10.0.0.1	10.0.0.3	TCP	Bogus TCP h...	2.5 版本号错的IP包
6	Sat Oct 15 23:56:44.421 2022	10.0.0.1	10.0.0.3	TCP	Bogus TCP h...	2.6 头部长度错误的IP包
7	Sat Oct 15 23:56:46.421 2022	10.0.0.1	192.167.173.8	TCP	Bogus TCP h...	2.7 错误目标地址的IP包

```
Ethernet II, Src: 00:0D:01:00:00:0A , Dst: 00:0D:03:00:00:0A
Version :4, Src: 10.0.0.1 ,  Dst: 10.0.0.3
    Version :4
    Header length: 20 bytes
    Type of service: 0x00
    Total length: 20 bytes
    Identification: 0x0 (0)
    Flags: 0
    Fragment offset: 0
    Time to live: 64
    Protocol: TCP (0x06)
    Header checksum: 0x029A[incorrect, should be 0xFE49]
    Source: 10.0.0.1
    Destination : 10.0.0.3
```

```
0000   00 0D 03 00 00 0A 00 0D 01 00 00 0A 08 00 45 00
0010   00 14 00 00 00 00 40 06 02 9A 0A 00 00 01 0A 00
0020   00 03
```

3) 路由表初始化、路由增加、路由转发三个函数的实现流程图

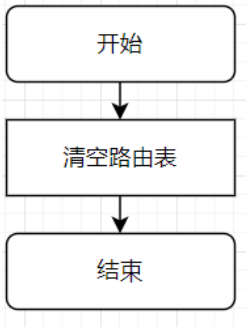


图 2 路由表初始化函数实现流程图

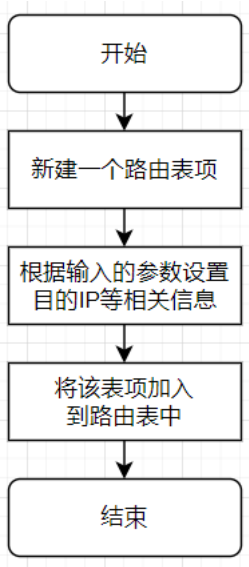


图 3 路由增加函数实现流程图

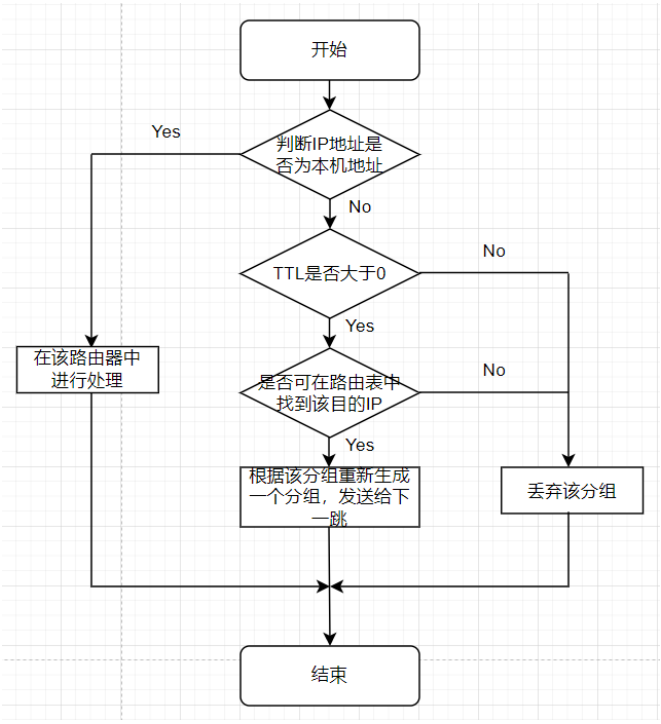


图 4 路由转发函数实现流程图

5) 所新建数据结构的说明

新建了路由表结构体，存储了四个字段。

```
// 构造路由表结构体
struct routingTable
{
    unsigned int dstIP;   // 目的IP
    unsigned int mask;    // 掩码
    unsigned int masklen; // 掩码长度
    unsigned int nexthop; // 下一跳
};
```

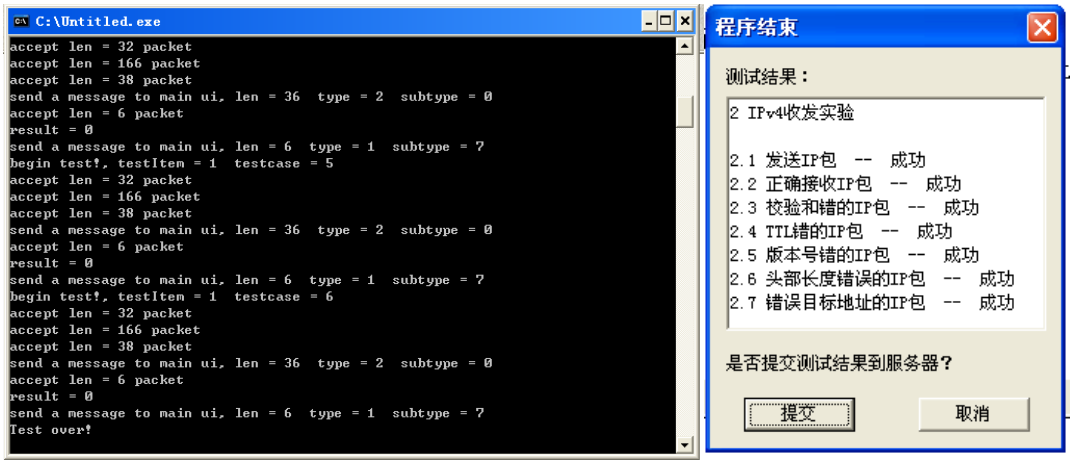
6) 请分析在存在大量分组的情况下如何提高转发效率，如果代码中有相关功能实现，请给出具体原理说明

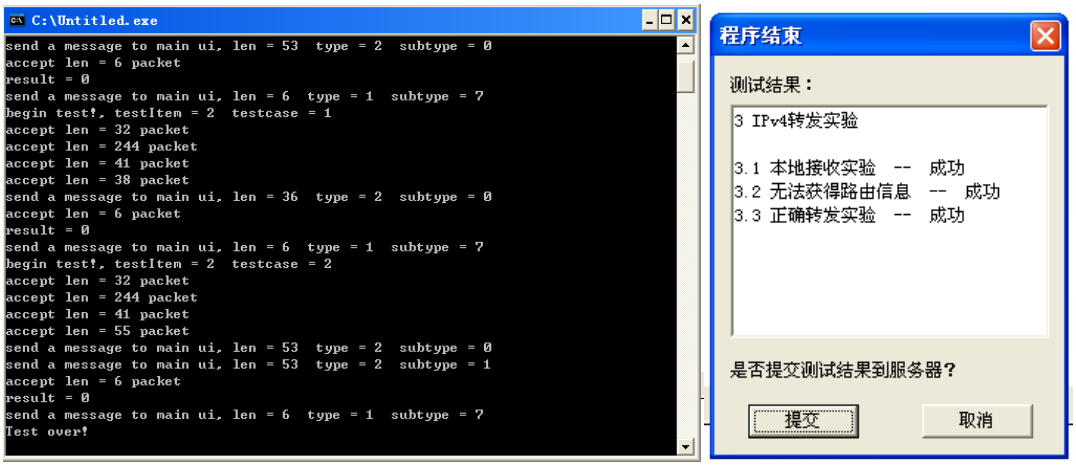
1. 树形结构匹配路由表项：路由表存储结构由线性结构改为树形结构，提高匹配效率。

2. 并行检查：每次在转发分组时，都要检测数据合法性，计算校验和等操作，我们可以并行操作，多个转发同时进行，就能提高转发效率，由硬件来实现。

3. 缓存分组：经过路由器的前后分组间的相关性很大，具有相同目的地址和源地址的分组往往连续到达，快速转发过程中，缓存分组，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，减轻了路由器的负担，提高路由器吞吐量。

实验结果：





问题讨论：

本次实验，在转发的时候，不仅需要确定目的主机的 IP 地址，还需要根据子网掩码进行进一步判断。

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：（展开讨论）

1.首先最直接的一点就是改进存储的结构，在本实验中使用的是线性结构，因此查询的时间复杂度就是O(n)。如果想把查询时间优化到O(logn)，可以根据IP目的地址进行有序存储，在查询的时候使用二分查找，这样查询时间就可以优化到O(logn)，如果希望查询时间复杂度为O(1)，可以将路由表设置为一个哈希表，使用空间换取时间。

2.如果不考虑从查询时间入手，那么由于存在大量分组，因此可以并行转发，所有的检测、匹配等过程都可以并行处理，这可以从硬件层面入手处理。

心得体会：

通过本次实验，我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，了解 IP 报文的结构。了解了路由器是如何为分组选择路由，并逐跳地发送到目的主机。深入理解了路由表的数据结构， 理解路由器是如何根据路由表对分组进行转发的。