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 收发和转发实验				
姓名	cycleke		院系	计算学部	
班级			学号		
任课教师	刘亚维		指导教师	刘亚维	
实验地点	格物 207		实验时间	2020 年 11 月 14 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					

计算学部

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。

IPv4 分组收发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。

IPv4 分组转发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。

IPv4 分组转发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：

在 IPv4 分组收发实验中，我们需要：

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。

在 IPv4 分组转发实验中，我们需要：

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”。

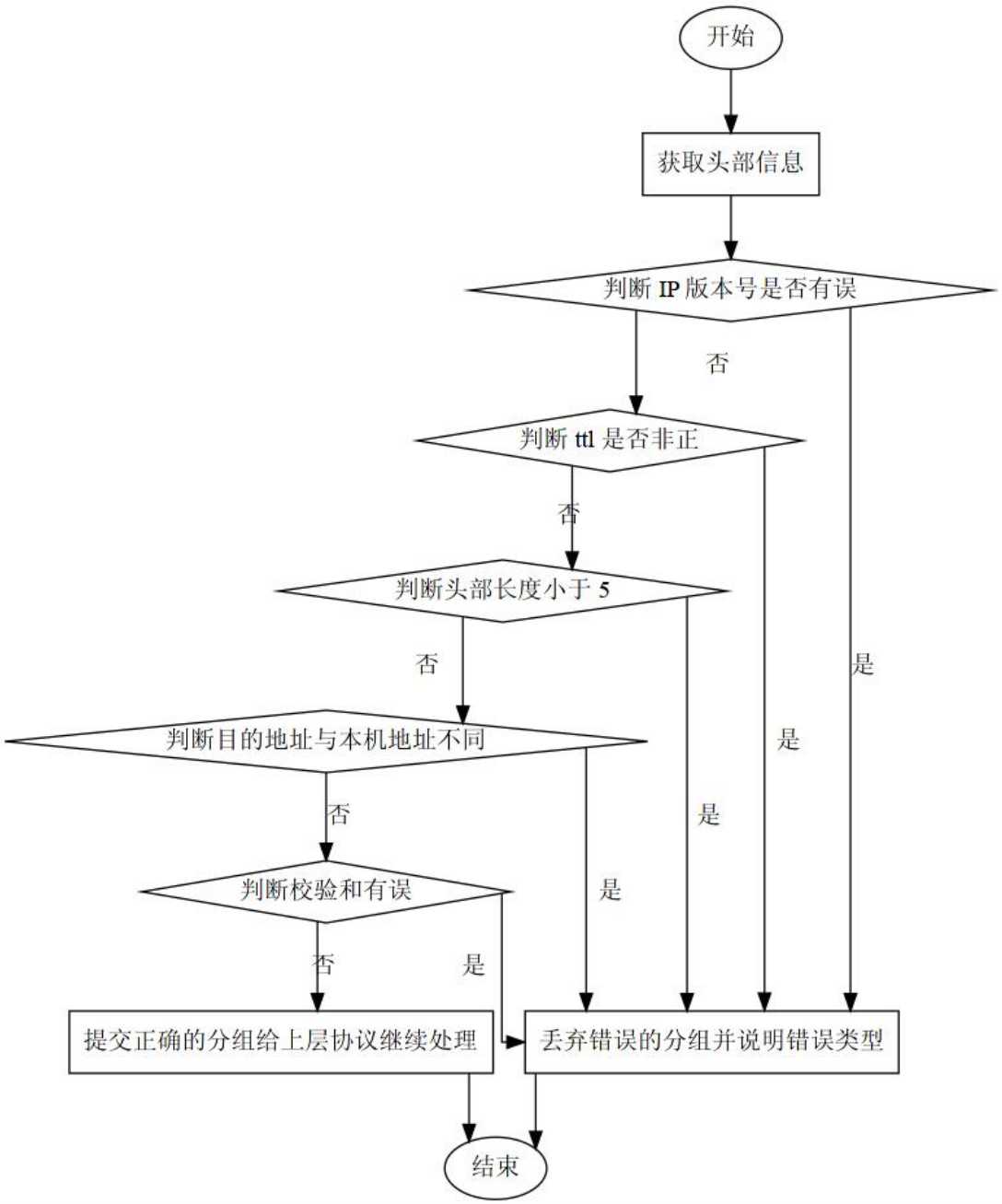
3) IPv4 分组的转发。

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。

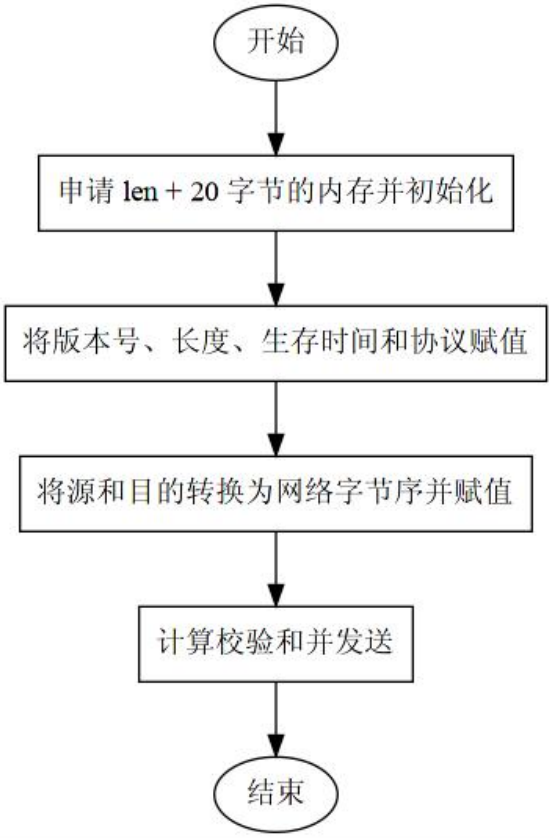
实验过程：

一、IPv4 分组收发实验

接收函数的流程图如下：

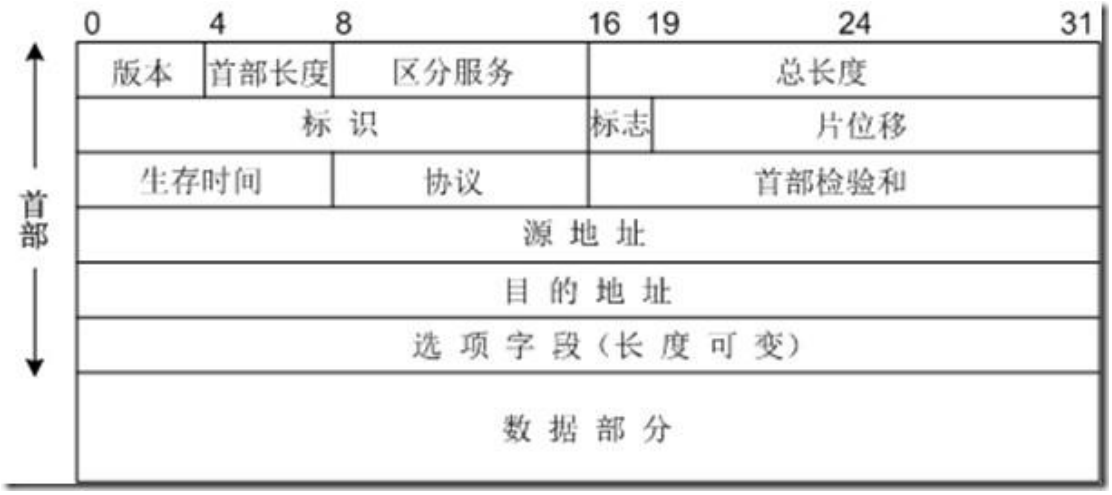


发送函数的流程图如下：



1. 获取头部信息并检验

IPV4 数据包的头部信息如下图所示：



我们可以利用位运算来获取版本等头部信息。版本号和头部长度的信息在第一个字节，然后通过位运算来分离两个信息。生存时间字段在 IPv4 数据包的头部第八个字节。源地址和目的地址在 IPv4 数据包的第十二和第十六个字节开始（获取时需要转换网络字节）。

我们需要检验版本号是否为 4，头部长度是否大于 5（因为最少的 IPv4 数据包的头部信息为 20 字节），生存时间是否为正（如果 TTL 非正则说明其已经过期），目的地址是否为本机地址。具体的代码如下：

```
unsigned short version = pBuffer[0] >> 4;       // 版本号
unsigned short head_length = pBuffer[0] & 0×f; // 头部长度
unsigned short ttl = pBuffer[8];                // 生存时间
unsigned short checksum = ntohs(*(unsigned short *)(pBuffer + 10)); // 校验和
unsigned int dest = ntohl(*(unsigned int *)(pBuffer + 16)); // 目的地址

if (version ≠ 4) {
  // IP 版本号错
  ip_DiscardPkt(pBuffer, STUD_IP_TEST_VERSION_ERROR);
  return 1;
} else if (ttl ⩽ 0) {
  // TTL 值出错
  ip_DiscardPkt(pBuffer, STUD_IP_TEST_TTL_ERROR);
  return 1;
} else if (head_length < 5) {
  //头部长度错
  ip_DiscardPkt(pBuffer, STUD_IP_TEST_HEADLEN_ERROR);
  return 1;
} else if (dest ≠ getIpv4Address()) {
  //目的地址错
  ip_DiscardPkt(pBuffer, STUD_IP_TEST_DESTINATION_ERROR);
  return 1;
}
```

2. 头部校验和字段检验原理

头部校验和字段在 IPv4 数据包的头部第十个字节，并且根据之前进行计算时取反的性质，将所有的头部信息进行与 checksum 生成时相同的计算步骤，得到的结果应该为全 1；否则说明头部校验和错误。以接收函数的代码为例，具体的实现如下：

```
unsigned long sum = 0;
for (int i = 0; i < head_length * 2; ++i) {
  sum += (unsigned char)pBuffer[i * 2] << 8;
  sum += (unsigned char)pBuffer[i * 2 + 1];
}
unsigned short l_word = sum & 0×ffff;
unsigned short h_word = sum >> 16;
if (l_word + h_word ≠ 0×ffff) {
  // IP 校验和出错
  ip_DiscardPkt(pBuffer, STUD_IP_TEST_CHECKSUM_ERROR);
  return 1;
}
```

二、IPv4 分组转发实验

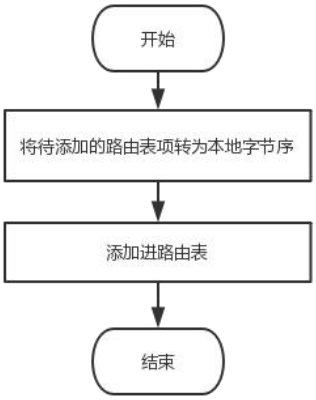
1. stud_route_init 函数

在 stud_route_init 函数中，我们需要初始化路由表。由于我们使用 STL 中的 vector 类维护路由表，所以我们只需要一条语句即可完成。

```
void stud_Route_Init() {
  routes.clear(); // 清空路由
}
```

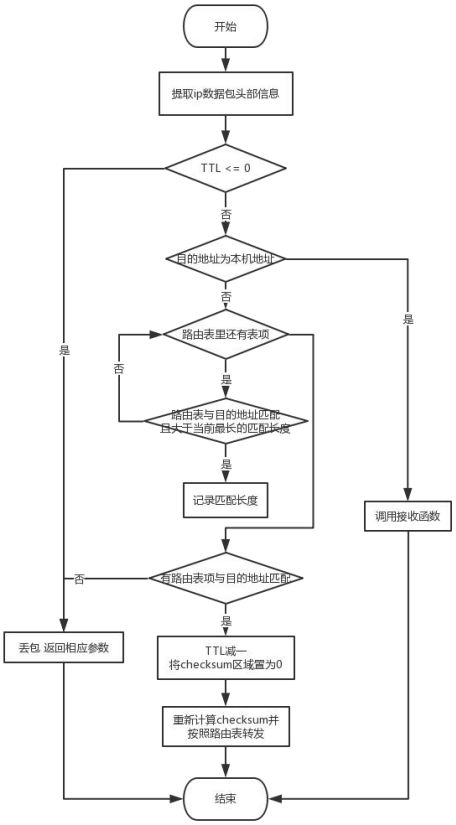
2. stud_route_add 函数

在 stud_route_add 函数中，我们只需要将网络字节转化为本地字节之后赋值即可。其流程图如下：



3. stud_fwd_deal 函数

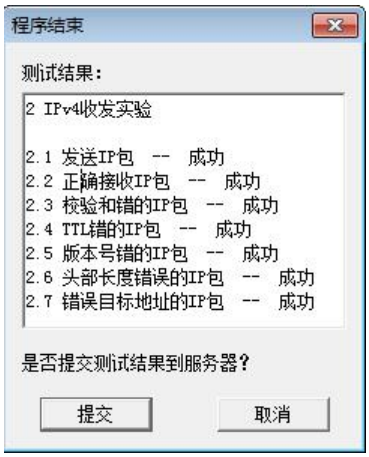
在 stud_fwd_deal 函数中，我们需要查找路由表并处理转发流程。首先我们需要判断生存时间，之后根据目的地址来判断是需要转发还是接收。其流程图如下：



实验结果：

1. IPv4 收发实验的测试

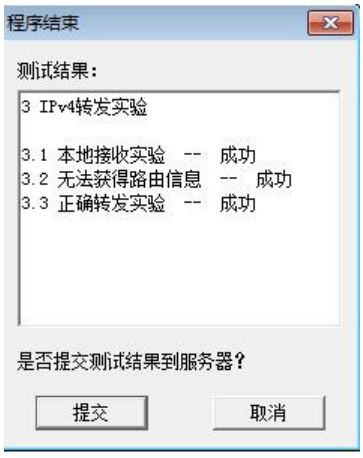
在 Windows 7 客户端下测试的截图为：



2020年秋	1	1180100217	郭炼	计算机科学与技术学院	数据科学与大数据技术	IPv4收发实验	2020-11-14		10		

2. IPv4 转发实验的测试

在 Windows 7 客户端下测试的截图为：



2020年秋	2	1180100217	郭炼	计算机科学与技术学院	数据科学与大数据技术	IPv4转发实验	2020-11-14		10		

问题讨论：

如何提高路由表的查找效率？

在本次实验中，我使用了 STL 库中的 vector 类，方便动态在路由表中添加路由项。但是这样在搜索时必须使用线性搜索 O(n) 的时间代价，在路由表极大时搜索的效率较低。

基于最大匹配原则，我们可以使用字典树来存储路由表，以 IP 地址为单词来建立字典，进而进行匹配。这样，每次添加路由表项和搜索路由表项的时间代价均为 O(1) 的，极大地提高了路由表的效率。

心得体会：

通过本次实验，我对于 IPv4 协议的理解有了极大的提高，学习了 IPv4 协议基本的分组接收和发送流程，初步接触了路由表这一重要的数据结构，认识了路由器是如何根据路由表对分组进行转发的。

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，为后面进行更为深入复杂的实验奠定良好的基础。