互联网协议实验与流完成时间实验

学号：2022K8009929011
姓名：王泽黎

实验一：互联网协议实验

一、实验一：实验任务

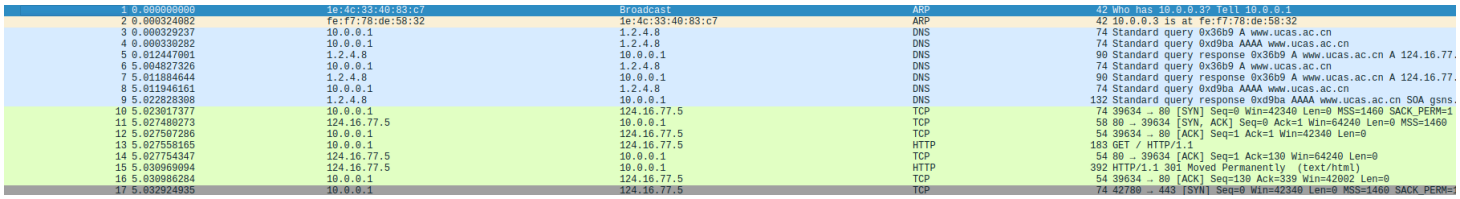
1. 利用wireshark观察wget的过程
2. 调研并解释这个过程
3. 理解相关协议的封装

二、实验一：实验流程

1. 在终端上执行sudo mn --nat，将host连接至Internet，启动mininet
2. 在mininet中输入xterm h1，打开h1的终端
3. 在h1终端中输入echo "nameserver 1.2.4.8" > /etc/resolv.conf，设置DNS服务器
4. 在h1终端中输入wireshark &，打开wireshark
5. 在wireshark中选择h1-eth0，开始抓包
6. 在h1终端中输入wget [www.ucas.ac.cn]，观察wireshark的抓包过程
7. 调研分析抓包过程以及所得的几种互联网协议

三、实验一：实验结果与分析

（一）wireshark抓包过程



1	0.000000000	1e:4c:33:40:83:c7	Broadcast	ARP	42 Who has 10.0.0.3? Tell 10.0.0.1
2	0.000324082	fe:f7:78:de:58:32	1e:4c:33:40:83:c7	ARP	42 10.0.0.3 is at fe:f7:78:de:58:32
3	0.000329237	10.0.0.1	1.2.4.8	DNS	74 Standard query 0x36b9 A www.ucas.ac.cn
4	0.000330282	10.0.0.1	1.2.4.8	DNS	74 Standard query 0xd9ba AAAA www.ucas.ac.cn
5	0.012447001	1.2.4.8	10.0.0.1	DNS	90 Standard query response 0x36b9 A www.ucas.ac.cn A 124.16.77.
6	5.004827326	10.0.0.1	1.2.4.8	DNS	74 Standard query 0x36b9 A www.ucas.ac.cn
7	5.011884644	1.2.4.8	10.0.0.1	DNS	90 Standard query response 0x36b9 A www.ucas.ac.cn A 124.16.77.
8	5.011946161	10.0.0.1	1.2.4.8	DNS	74 Standard query 0xd9ba AAAA www.ucas.ac.cn
9	5.022828308	1.2.4.8	10.0.0.1	DNS	132 Standard query response 0xd9ba AAAA www.ucas.ac.cn SOA gsns.
10	5.023017377	10.0.0.1	124.16.77.5	TCP	74 39634 → 80 [SYN] Seq=0 Win=42340 Len=0 MSS=1460 SACK_PERM=1
11	5.027480273	124.16.77.5	10.0.0.1	TCP	58 80 → 39634 [SYN, ACK] Seq=0 Ack=1 Win=64240 Len=0 MSS=1460
12	5.027507286	10.0.0.1	124.16.77.5	TCP	54 39634 → 80 [ACK] Seq=1 Ack=1 Win=42340 Len=0
13	5.027558165	10.0.0.1	124.16.77.5	HTTP	183 GET / HTTP/1.1
14	5.027754347	124.16.77.5	10.0.0.1	TCP	54 80 → 39634 [ACK] Seq=1 Ack=130 Win=64240 Len=0
15	5.030969094	124.16.77.5	10.0.0.1	HTTP	392 HTTP/1.1 301 Moved Permanently (text/html)
16	5.030986284	10.0.0.1	124.16.77.5	TCP	54 39634 → 80 [ACK] Seq=130 Ack=339 Win=42002 Len=0
17	5.032924935	10.0.0.1	124.16.77.5	TCP	74 42780 → 443 [SYN] Seq=0 Win=42340 Len=0 MSS=1460 SACK_PERM=

（二）ARP协议层次

```
▸ Frame 1: 42 bytes on wire (336 bits), 42 bytes captured (336 bits) on interface h1-eth0, id 0
▾ Ethernet II, Src: 5a:f2:ba:3b:c4:ec (5a:f2:ba:3b:c4:ec), Dst: Broadcast (ff:ff:ff:ff:ff:ff)
    ▸ Destination: Broadcast (ff:ff:ff:ff:ff:ff)
    ▸ Source: 5a:f2:ba:3b:c4:ec (5a:f2:ba:3b:c4:ec)
      Type: ARP (0x0806)
▾ Address Resolution Protocol (request)
      Hardware type: Ethernet (1)
      Protocol type: IPv4 (0x0800)
      Hardware size: 6
      Protocol size: 4
      Opcode: request (1)
      Sender MAC address: 5a:f2:ba:3b:c4:ec (5a:f2:ba:3b:c4:ec)
      Sender IP address: 10.0.0.1
      Target MAC address: 00:00:00_00:00:00 (00:00:00:00:00:00)
      Target IP address: 10.0.0.3
```

分析结果为：Ethernet < ARP

（三）DNS协议层次

```
▸ Frame 3: 74 bytes on wire (592 bits), 74 bytes captured (592 bits) on interface h1-eth0, id 0
▾ Ethernet II, Src: 5a:f2:ba:3b:c4:ec (5a:f2:ba:3b:c4:ec), Dst: a2:0f:ba:91:9e:a9 (a2:0f:ba:91:9e:a9)
    ▸ Destination: a2:0f:ba:91:9e:a9 (a2:0f:ba:91:9e:a9)
    ▸ Source: 5a:f2:ba:3b:c4:ec (5a:f2:ba:3b:c4:ec)
      Type: IPv4 (0x0800)
▸ Internet Protocol Version 4, Src: 10.0.0.1, Dst: 1.2.4.8
▸ User Datagram Protocol, Src Port: 38375, Dst Port: 53
▾ Domain Name System (query)
      Transaction ID: 0x67b1
    ▸ Flags: 0x0100 Standard query
      Questions: 1
      Answer RRs: 0
      Authority RRs: 0
      Additional RRs: 0
    ▸ Queries
      [Response In: 5]
```

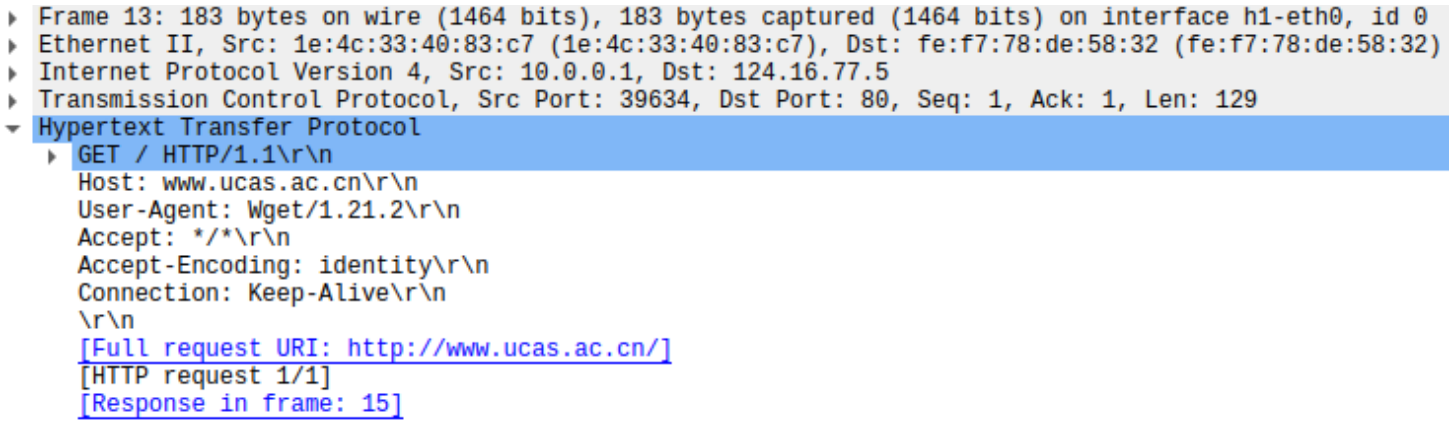
分析结果为：Ethernet < IP < UDP < DNS

（四）TCP协议层次

```
▸ Frame 10: 74 bytes on wire (592 bits), 74 bytes captured (592 bits) on interface h1-eth0, id 0
▾ Ethernet II, Src: 5a:f2:ba:3b:c4:ec (5a:f2:ba:3b:c4:ec), Dst: a2:0f:ba:91:9e:a9 (a2:0f:ba:91:9e:a9)
    ▸ Destination: a2:0f:ba:91:9e:a9 (a2:0f:ba:91:9e:a9)
    ▸ Source: 5a:f2:ba:3b:c4:ec (5a:f2:ba:3b:c4:ec)
      Type: IPv4 (0x0800)
▸ Internet Protocol Version 4, Src: 10.0.0.1, Dst: 124.16.77.5
▾ Transmission Control Protocol, Src Port: 46758, Dst Port: 80, Seq: 0, Len: 0
      Source Port: 46758
      Destination Port: 80
      [Stream index: 0]
      [Conversation completeness: Complete, WITH_DATA (31)]
      [TCP Segment Len: 0]
      Sequence Number: 0    (relative sequence number)
      Sequence Number (raw): 1382592474
      [Next Sequence Number: 1    (relative sequence number)]
      Acknowledgment Number: 0
      Acknowledgment number (raw): 0
      1010 .... = Header Length: 40 bytes (10)
    ▸ Flags: 0x002 (SYN)
```

分析结果为：Ethernet < IP < TCP

（五）HTTP协议层次



分析结果为：Ethernet < IP < TCP < HTTP

（六）结果分析

在获取UCAS官网的过程中，wireshark抓包得到了ARP、DNS、TCP、HTTP等协议的封装层次：

1. ARP协议层次为Ethernet < ARP
2. DNS协议层次为Ethernet < IP < UDP < DNS
3. TCP协议层次为Ethernet < IP < TCP
4. HTTP协议层次为Ethernet < IP < TCP < HTTP

从wireshark抓包结果看出TCP承载HTTP协议

Wireshark · Follow TCP Stream (tcp.stream eq 0) · h1-eth0

```
GET / HTTP/1.1
Host: www.ucas.ac.cn
User-Agent: Wget/1.21.2
Accept: */*
Accept-Encoding: identity
Connection: Keep-Alive

HTTP/1.1 301 Moved Permanently
Date: Thu, 05 Sep 2024 13:53:09 GMT
Content-Type: text/html
Content-Length: 162
Connection: keep-alive
Location: https://www.ucas.ac.cn/

<html>
<head><title>301 Moved Permanently</title></head>
<body>
<center><h1>301 Moved Permanently</h1></center>
<hr><center>nginx</center>
</body>
</html>
```

四、实验一：调研分析

（一）ARP（地址解析协议）

ARP协议“Address Resolution Protocol”(地址解析协议)的缩写。其作用是在以太网环境中，数据的传输所依懒的是MAC地址而非IP地址，而将已知IP地址转换为MAC地址的工作是由ARP协议来完成的。

在局域网中，网络中实际传输的是“帧”，帧里面是有目标主机的MAC地址的。在以太网中，一个主机和另一个主机进行直接通信，必须要知道目标主机的MAC地址。目标MAC地址是通过地址解析协议获得的。所谓“地址解析”就是主机在发送帧前将目标IP地址转换成目标 MAC 地址的过程。ARP 协议的基本功能就是通过目标设备的 IP 地址，查询目标设备的MAC地址，以保证通信的顺利进行。ARP通过发送一个ARP请求帧到局域网中的所有设备来查找目标设备的MAC地址。这个请求包含源设备的IP地址和MAC地址。目标设备收到请求后，会回复一个包含其IP地址和MAC地址的ARP响应帧。

（二）DNS（域名系统）

DNS (Domain Name System)是一个应用层协议，域名系统(DNS)的作用是将人类可读的域名(如[www.example.com])转换为机器可读的IP地址(如192.0.2.44)。 DNS系统使用树状层次结构，包括多个DNS 服务器，它们负责不同的域名解析。当用户输入一个域名时，客户端的DNS解析器将向根DNS服务器发送查询，然后逐级查询更低级别的DNS服务器，直到找到与域名相关的IP地址。

DNS协议建立在UDP或TCP协议之上，默认使用53号端口。客户端默认通过UDP协议进行通讯，但是由于广域网中不适合传输过大的 UDP 数据包，因此规定当报文长度超过了512 字节时，应转换为使用TCP协议进行数据传输。DNS是一种可以将域名和IP地址相互映射的以层次结构分布的数据库系统。

（三）TCP（传输控制协议）

TCP (Transmission Control Protocol 传输控制协议)是一种面向连接的、可靠的、基于字节流的传输层通信协议，由IETF的RFC 793定义。在简化的计算机网络OSI模型中，它完成第四层传输层所指定的功能。

应用层向TCP层发送用于网间传输的、用8位字节表示的数据流，然后TCP把数据流分区成适当长度的报文段(通常受该计算机连接的网络的数据链路层的最大传输单元(MTU)的限制) 。之后TCP把结果包传给IP层，由它来通过网络将包传送给接收端实体的TCP层。TCP将用户数据打包构成报文段，它发送数据时启动一个定时器，另一端收到数据进行确认，对失序的数据重新排序，丢弃重复的数据。简单说，TCP协议的作用是，保证数据通信的完整性和可靠性，防止丢包。

（四）HTTP（超文本传输协议）

HTTP协议(超文本传输协议HyperText Transfer Protocol)，它是基于TCP协议的应用层传输协议，用于从WWW服务器传输超文本到本地浏览器的传输协议， HTTP是一个应用层协议，由请求和响应构成，是一个标准的客户端和服务器模型，简单来说就是客户端和服务端进行数据传输的一种规则。它指定了客户端可能发送给服务器什么样的消息以及得到什么样的响应。客户端发送HTTP请求到服务器，请求特定资源（如网页或图像）。服务器收到请求后，会发送HTTP响应，包含请求的资源以及相关信息。HTTP通信通常是无状态的，每个请求和响应都独立于之前的请求和响应。

五、实验一：实验总结

通过本次实验，我了解了互联网协议的封装过程，掌握了wireshark的使用方法，对ARP、DNS、TCP、HTTP等协议的封装层次有了更深入的了解。同时，我还调研了这几种协议的作用和功能，对互联网协议的工作原理有了更深入的认识。

实验二：流完成时间实验

一、实验二：实验任务

1. 利用fct_exp.py脚本复现讲义上的图，每个数据点做5次实验，取均值
2. 调研解释图中的现象。

二、实验二：实验流程

1. 在Python脚本中设定带宽及延迟
2. 在终端中输入 sudo python fct_exp.py

3. 在终端中输入 xterm h1 h2 中启动h1、h2两个host
4. 在h2 终端输入 dd if=/dev/zero of=file_sizeMB.dat bs=file_sizeM count=1，其中file_size 分别设置为 1，10， 100进行不同大小的实验
5. 在h1终端中输入wget [http://10.0.0.2/file_sizeMB.dat] 获取主机 h2 上对应大小的文件
6. 记录每次完成传输的时间和速度，每个数据点做五次实验，取均值
7. 根据结果绘图

三、实验二：实验结果与分析

（一）带宽为10Mbps下的实验结果（延迟为100ms）

（1）1MB数据包传输结果

时间(s)	1.5	1.5	1.5	1.5	1.5	1.5
速度(MB/s)	0.671	0.681	0.675	0.672	0.674	0.6746
序号	1	2	3	4	5	均值

（2）10MB数据包传输结果

时间(s)	9.5	9.5	9.5	9.5	9.5	9.5
速度(MB/s)	1.15	1.14	1.14	1.12	1.14	1.138
序号	1	2	3	4	5	均值

（3）100MB数据包传输结果

序号	1	2	3	4	5	均值
时间(s)	89	89	89	89	89	89
速度(MB/s)	1.13	1.12	1.13	1.13	1.14	1.13

（二）带宽为50Mbps下的实验结果（延迟为100ms）

（1）1MB数据包传输结果

序号	1	2	3	4	5	均值
时间(s)	1.2	1.2	1.2	1.2	1.2	1.2
速度(MB/s)	0.829	0.828	0.83	0.829	0.829	0.829

（2）10MB数据包传输结果

序号	1	2	3	4	5	均值
时间(s)	2.9	2.9	2.9	2.9	2.9	2.9
速度(MB/s)	3.41	3.42	3.4	3.42	3.41	3.412

（3）100MB数据包传输结果

序号	1	2	3	4	5	均值
时间(s)	19	19	19	19	19	19
速度(MB/s)	5.61	5.63	5.57	5.62	5.62	5.61

（三）带宽为100Mbps下的实验结果（延迟为100ms）

（1）1MB数据包传输结果

序号	1	2	3	4	5	均值
时间(s)	1.2	1.2	1.2	1.2	1.2	1.2
速度(MB/s)	0.835	0.836	0.833	0.834	0.836	0.8348

(2) 10MB数据包传输结果

序号	1	2	3	4	5	均值
时间(s)	2.2	2.2	2.2	2.2	2.2	2.2
速度(MB/s)	4.53	4.5	4.52	4.51	4.52	4.516

(3) 100MB数据包传输结果

序号	1	2	3	4	5	均值
时间(s)	11	11	11	11	11	11
速度(MB/s)	10.8	10.9	10.9	10.7	10.8	10.82

（四）带宽为500Mbps下的实验结果（延迟为100ms）

（1）1MB数据包传输结果

序号	1	2	3	4	5	均值
时间(s)	1.2	1.2	1.2	1.2	1.2	1.2
速度(MB/s)	0.842	0.841	0.842	0.84	0.84	0.841

（2）10MB数据包传输结果

序号	1	2	3	4	5	均值
时间(s)	1.9	1.9	1.9	1.9	1.9	1.9
速度(MB/s)	5.34	5.35	5.33	5.33	5.33	5.336

（3）100MB数据包传输结果

序号	1	2	3	4	5	均值
时间(s)	4.9	4.9	4.9	4.9	4.9	4.9
速度(MB/s)	25.6	26.8	26.3	26.9	26.2	26.36

（五）带宽为1GBps下的实验结果（延迟为100ms）

（1）1MB数据包传输结果

序号	1	2	3	4	5	均值
时间(s)	1.2	1.2	1.2	1.2	1.2	1.2
速度(MB/s)	0.846	0.843	0.845	0.848	0.848	0.846

（2）10MB数据包传输结果

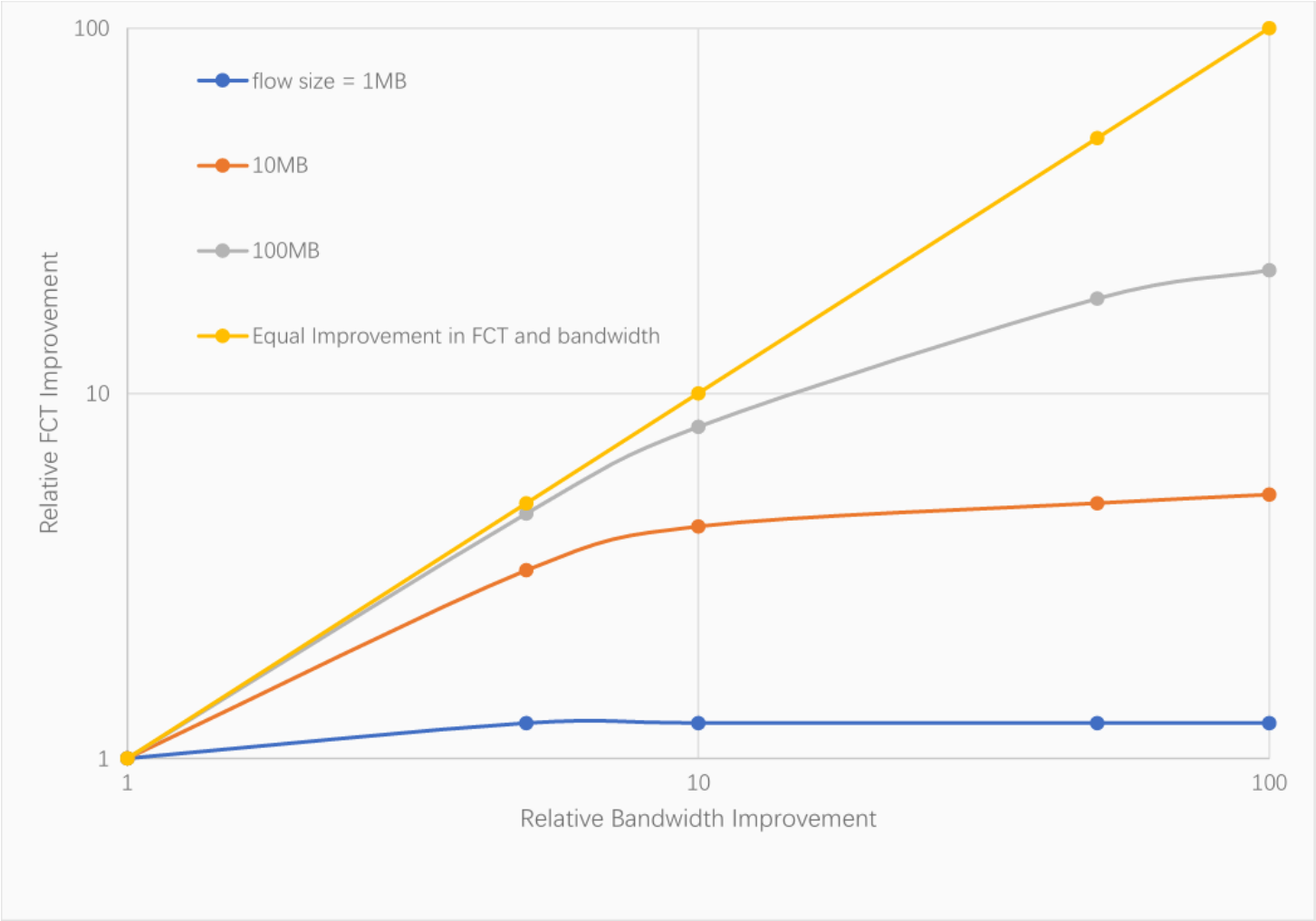
序号	1	2	3	4	5	均值
时间(s)	1.8	1.8	1.8	1.8	1.8	1.8
速度(MB/s)	5.42	5.42	5.42	5.42	5.42	5.42

（3）100MB数据包传输结果

序号	1	2	3	4	5	均值
时间(s)	4.1	4.1	4.1	4.1	4.1	4.1
速度(MB/s)	24.6	24.5	24.6	24.5	24.4	24.52

（六）结果绘图

我们使用对数坐标系，横纵坐标是带宽和 FCT 的改进，对于纵坐标，我们将最长的耗时作为分子，每个耗时的数据作为分母，以此作为纵坐标来体现FCT的改进横坐标则是每个带宽和最小带宽的比值，这样定义的横纵坐标可以用来表示带宽的改进和FCT的改进，绘图如下：



由图可得：

1. 当带宽一定时，数据包越大，网络传输速率越高，数据包越小，网络传输速率越低
2. 当带宽到达50Mbps时，1MB的包速率不再发生变化（时间耗时均为1.2s）
3. 当数据包大小不变时，网络传输速率不会随着带宽线性增加

此外，根据改变延迟实验结果（未列出），我们可以发现，减小延迟可以显著增加高带宽、大数据包的传输速率。

四、实验二：调研分析

（一）TCP传输

TCP 协议会将应用层的数据流分割成适当长度的报文段，最大传输段大小（MSS）通常受该计算机连接的网络的数据链路层的最大传送单元（MTU）限制。而 TCP 的传输速率是由其阻塞算法决定的，TCP 拥塞算法缓慢地探测网络的可用带宽，增加传输速率直到检测到分组丢失，然后指数地降低传输速率。

当数据包大小不变带宽增加时，该算法会增加传输速率直至分组丢失，而降低传输速率时指数级的，因此速率并不会随着带宽的增加而线性增加。

同时，对数据进行分组也会造成丢包、排队、阻塞等问题，这也会影响到传输速率的增长。

（二）慢启动机制

慢启动是TCP使用的一种阻塞控制机制。慢启动也叫做指数增长期。慢启动是指每次TCP接收窗口收到确认时都会增长。增加的大小就是已确认段的数目。这种情况一直保持到要么没有收到一些段，要么窗口大小到达预先定义的阈值。如果发生丢失事件， TCP就认为这是网络阻塞，就会采取措施减轻网络拥挤。一旦发生丢失事件或者到达阈值， TCP就会进入线性增长阶段。这时，每经过一个RTT窗口增长一个段。

由于TCP连接会随着时间进行自我调谐，起初会限制连接的最大速度，如果数据传输成功，会随着时间的推移提高传输速度，这就是TCP的慢启动机制。

这样就解释了在带宽较高时，小数据包没有达到期待的网速的问题。在慢启动阶段， TCP 预留的窗口大小会随着每接受到一个段而指数级增长，对于数据包大小较小的包，在窗口还没有达到带宽的阈值时可能传输就已经结束了，因此此时测得的传输速率会明显小于对应带宽的最大速率。

五、实验二：实验总结

这次实验使用了实际的数据，给我带来的直观且深刻的印象。加深了我对TCP协议的认识，也让我进一步熟悉了实验环境的使用，同时对影响网速的因素有了浅显的认识。