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	利用 Wireshark 进行协议分析					
姓名			院系	计算机科学与技术		
班级			学号			
任课教师			指导教师			
实验地点			实验时间			
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。

熟悉并掌握 Wireshark 的基本操作，了解网络协议实体间进行交互以及报文交换的情况。

实验内容：

概述本次实验的主要内容，包含的实验项等。

1) 学习 Wireshark 的使用
2) 利用 Wireshark 分析 HTTP 协议
3) 利用 Wireshark 分析 TCP 协议
4) 利用 Wireshark 分析 IP 协议
5) 利用 Wireshark 分析 Ethernet 数据帧

选做内容：

a) 利用 Wireshark 分析 DNS 协议
b) 利用 Wireshark 分析 UDP 协议
c) 利用 Wireshark 分析 ARP 协议

实验过程：

学习 Wireshark 的使用

基本介绍：

Wireshark 是一种可以运行在 Windows，UNIX，Linux 等操作系统上的分组分析器。Wireshark（前称 Ethereal）是一个网络封包分析软件。网络封包分析软件的功能是撷取网络封包，并尽可能显示出最为详细的网络封包资料。Wireshark使用WinPCAP作为接口，直接与网卡进行数据报文交换。

使用简介：

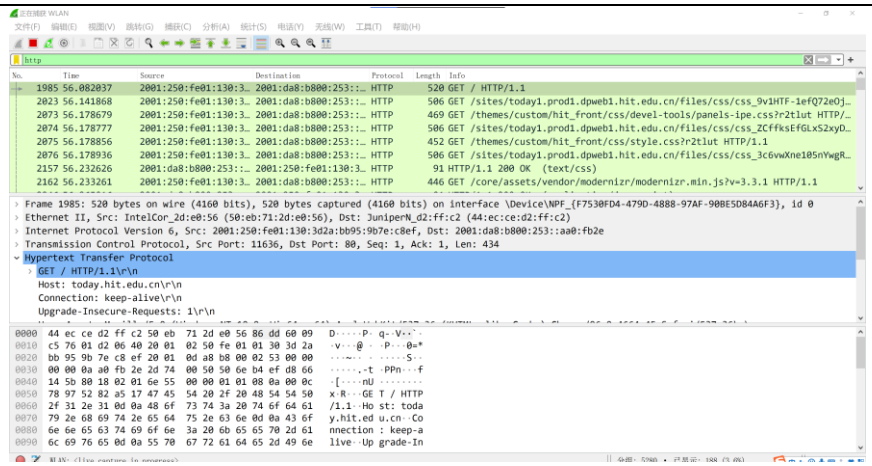
(1) 启动 Wireshark 软件，打开浏览器，选择网络接口 WLAN；

(2) 浏览器输入网址，Wireshark 抓包，并筛选 HTTP 协议报文：

(3) 详细用户界面：命令菜单、俘获分组列表、分组头部明细、分组内容窗口、筛选俘获分组等信息与实验指导书中给出的基本一致，不再展示。

利用 Wireshark 分析 HTTP 协议

浏览器中输入的网址为：http://today.hit.edu.cn， Wireshark抓包结果如下：



HTTP GET/response 交互

a)浏览器运行的是HTTP1.1，服务器运行的HTTP版本号为HTTP1.1

```
Hypertext Transfer Protocol
  GET / HTTP/1.1\r\n
  Host: today.hit.edu.cn\r\n
  Connection: keep-alive\r\n
  Upgrade-Insecure-Requests: 1\r\n
```

HTTP请求报文——浏览器的HTTP协议版本

```
Hypertext Transfer Protocol
  HTTP/1.1 200 OK\r\n
  Content-Type: text/css\r\n
  Transfer-Encoding: chunked\r\n
```

HTTP响应报文——服务器的HTTP协议版本

b)浏览器向服务器指出可接受的语言版本为zh_CN，也就是简体中文。

```
Accept-Encoding: gzip, deflate\r\n
Accept-Language: zh-CN,zh;q=0.9\r\n
```

接受的语言版本

c)本机IP地址为：172.20.171.118，服务器的IP地址为：202.118.254.117

```
Header Checksum: 0x0000 [validation disabled]
[Header checksum status: Unverified]
Source Address: 172.20.171.118
Destination Address: 202.118.254.117
```

源IP地址与目的IP地址

d)浏览器返回的状态代码为200：

```
Hypertext Transfer Protocol
  HTTP/1.1 200 OK\r\n
    [Expert Info (Chat/Sequence): HTTP/1.1 200 OK\r\n]
    Response Version: HTTP/1.1
    Status Code: 200
    [Status Code Description: OK]
    Response Phrase: OK
```

服务器返回状态码

HTTP 条件GET/response 交互

a) 第一个 GTE 请求在请求报文中无 IF-MODIFIED-SINCE。

```
Hypertext Transfer Protocol
  GET / HTTP/1.1\r\n
    [Expert Info (Chat/Sequence): GET /
    Request Method: GET
    Request URI: /
    Request Version: HTTP/1.1
  Host: today.hit.edu.cn\r\n
  Connection: keep-alive\r\n
  Cache-Control: max-age=0\r\n
  Upgrade-Insecure-Requests: 1\r\n
  User-Agent: Mozilla/5.0 (Windows NT 1(
  Accept: text/html,application/xhtml+xr
  Accept-Encoding: gzip, deflate\r\n
  Accept-Language: zh-CN,zh;q=0.9\r\n
  \r\n
  [Full request URI: http://today.hit.e(
  [HTTP request 1/2]
```

无该请求行

b) 服务器明确返回了文件内容，可以通过状态码和数据段感知。

```
Hypertext Transfer Protocol
  HTTP/1.1 200 OK\r\n
    [Expert Info (Chat/Sequence): HTTP/1.1 200 OK\r\n]
    Response Version: HTTP/1.1
    Status Code: 200
    [Status Code Description: OK]
    Response Phrase: OK
  Content-Type: text/html\r\n
  Content-Length: 405758\r\n
  Connection: keep-alive\r\n
  Date: Sun, 21 Nov 2021 13:17:56 GMT\r\n
  Server: Server\r\n
  X-Content-Type-Options: nosniff\r\n
  X-Frame-Options: SAMEORIGIN\r\n
  ETag: "630fe-5d14be14a40dd"\r\n
  Accept-Ranges: bytes\r\n
  Vary: Accept-Encoding\r\n
```

服务器返回的状态码

```
File Data: 1248 bytes
JavaScript Object Notation: application/json
  Array
    Object
      Member Key: command
      Member Key: method
      Member Key: selector
      Member Key: data
      Member Key: settings
```

服务器返回的数据

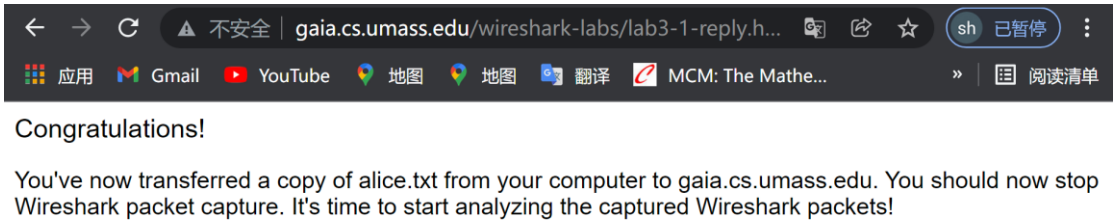
c) 向发出的较晚 GET 请求中，有该行：IF-MODIFIED-SINCE，且该行后的信息是本地缓存文件中 Last-Modified 字段的最后修改时间；

```
Content-Type: image/png\r\n
Content-Length: 479\r\n
Connection: keep-alive\r\n
Date: Sun, 21 Nov 2021 13:39:39 GMT\r\n
Server: Server\r\n
X-Content-Type-Options: nosniff\r\n
X-Frame-Options: SAMEORIGIN\r\n
Last-Modified: Fri, 18 Oct 2019 13:02:20 GMT\r\n
ETag: "1df-5952ef1bd380e"\r\n
X-Varnish: 428994548 434750890\r\n
Age: 124\r\n
Via: 1.1 varnish-v4\r\n
X-Varnish-Cache: HIT\r\n
Accept-Ranges: bytes\r\n
```

响应报文中出现IF-MODIFIED-SINCE字段

d) 服务器对较晚的 HTTP GET 请求的响应中的 HTTP 状态代码是304。服务器并为返回了文件的内容。这是因为客户端在找到本地缓存之后，经过请求报文向服务器端确定这一份缓存是最新的，那么服务器端就不再向客户端发送这一份数据，客户端直接使用缓存的数据段。

利用 Wireshark 分析 TCP 协议



上传数据

浏览追踪信息：

a)向 gaia.cs.umass.edu 服务器传送文件的客户端主机的 IP 地址是172.20.171.118， TCP 端口号是3144。

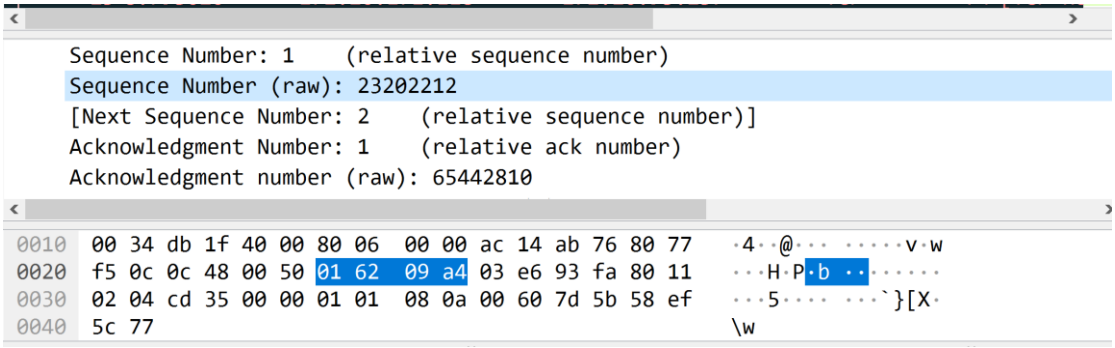
b) Gaia.cs.umass.edu 服务器的 IP 地址是128.119.245.12。对这一连接，它用来发送和接收 TCP 报文的端口号是80。

```
Internet Protocol Version 4, Src: 172.20.171.118, Dst: 128.119.245.12
Transmission Control Protocol, Src Port: 3144, Dst Port: 80, Seq: 1, Ack: 1, Len: 0
```

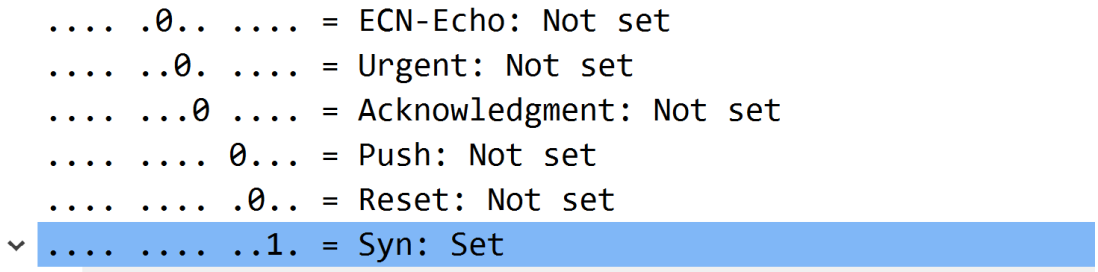
服务器端与客户端IP地址与端口号

TCP 基础：

a)客户服务器之间用于初始化 TCP 连接的 TCP SYN 报文段的序号是0x01 62 09 a4，在该报文段中，是用SYN标志位是否为1标示该报文段是 SYN 报文段的。

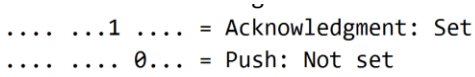


SYN报文端序号



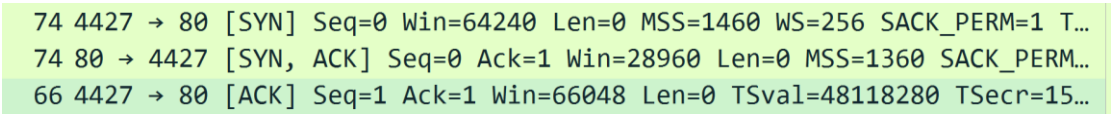
标志此报文段为SYN报文段

b)服务器向客户端发送的 SYNACK 报文段序号是0x03 e6 93 fa;该报文段中， Acknowledgement 字段的值是0x01 62 09 a5;Gaia.cs.umass.edu 服务器是将SYN报文段序号+1确定的这个值？在该报文段中，可以通过 SYN 和 ACK标志位都为1标识该报文段：



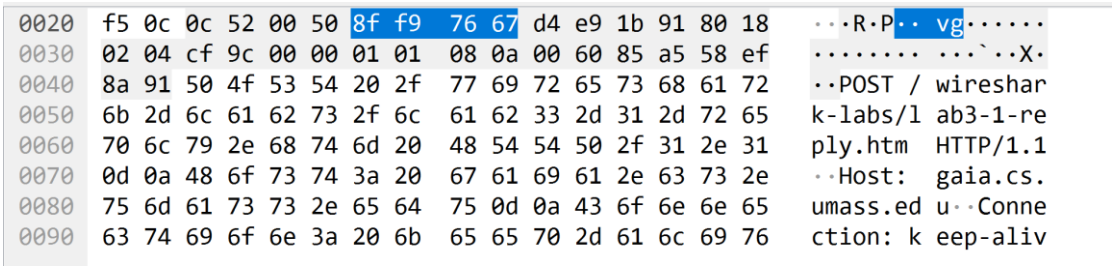
设置ACK以及SYN字段为1

c)可以分析出三次握手，具体信息如下：



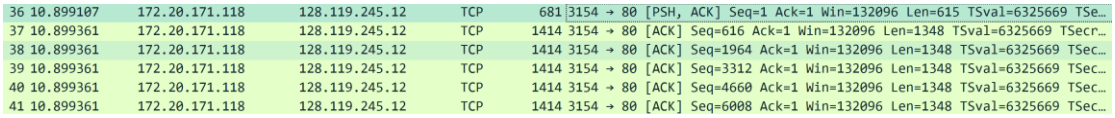
三次握手具体信息

d)包含 POST 命令的 TCP 报文段序号为8f f9 76 67：

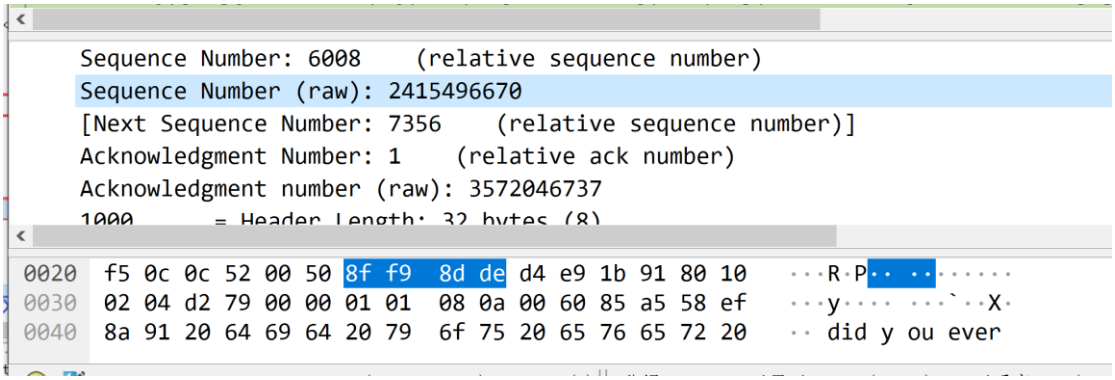


包含POST命令的TCP报文段

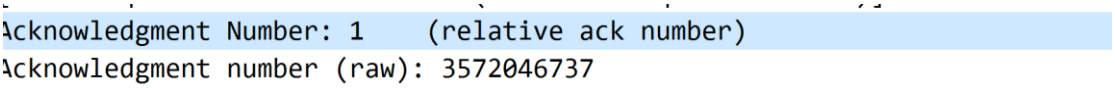
e)那么该 TCP 连接上的第六个报文段的序号是8f f9 8d de;发送时间：该报文段于 TCP 三次握手之后（作为第 9 个 TCP 报文段发送），四次挥手之前发送的；该报文段所对应的ACK是在第三次握手的时候接收的。



6个报文段

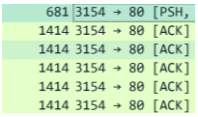


该报文段序号



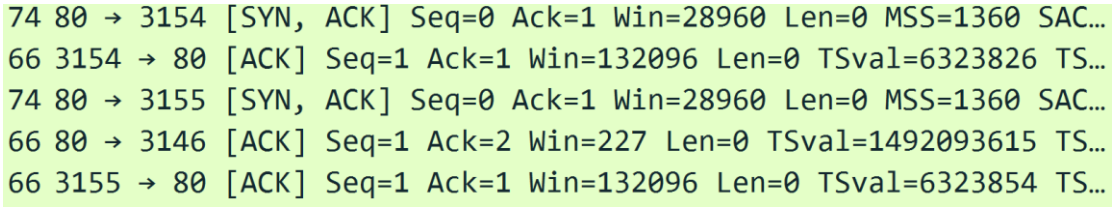
该报文段发送时间

f)前六个 TCP 报文段的长度各是681,1414,1414,1414,1414,1414。

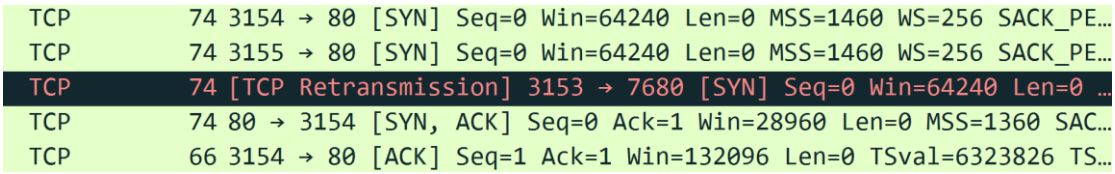


六个报文段长度

g)在整个跟踪过程中，接收端公示的最小的可用缓存空间是28960;限制发送端的传输以后，可以发现接收端的可用缓存空间在很多时候都是在递增的，最终可用缓存为132096。可以发现当限制了发送方的传输之后，接收端的缓存空间不断增大到132096。由此可见，接收端的缓存是足够的。

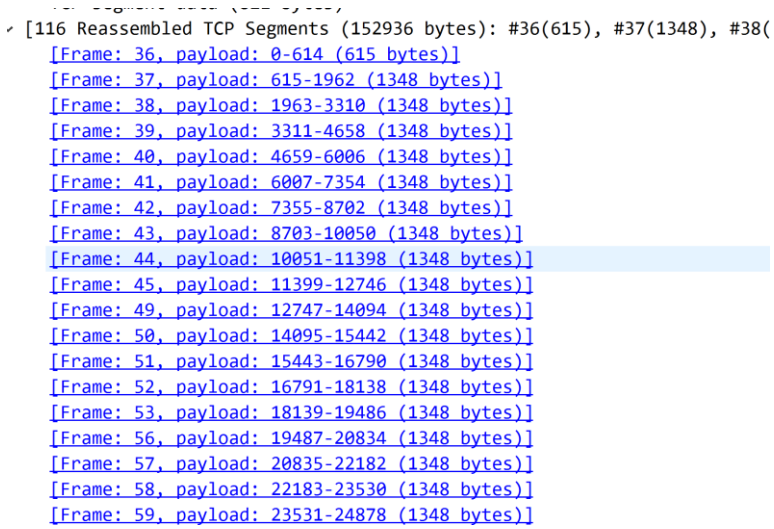


接收端公示的最小缓存空间为28960



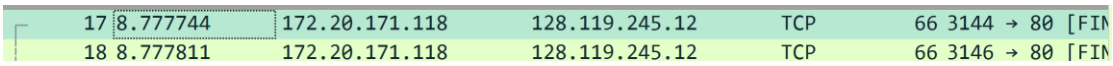
最终可用缓存为132096

h)在跟踪文件中没有重传的报文段，进行判断的依据是通过观察客户端的分组序号，可以发现分组序号是一直在增长，没有出现过重复的序号的，因此可以判断没有重传的报文段。

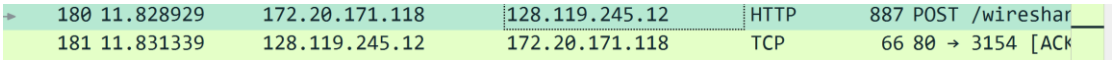


部分序列号

i)TCP 连接的 throughput是：50083.59Bps;请写出你的计算过程：152935B/3.053595=50083.59Bps



连接开始创建时间



连接结束时间

利用 Wireshark 分析 IP 协议

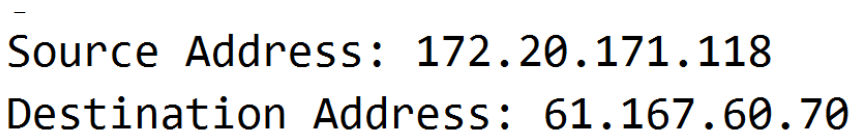
通过执行traceroute执行捕获数据包：

使用www.hit.edu.cn作为分析的网站。

对捕获的数据包进行分析：

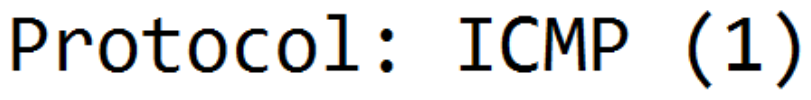
分析主机发出的第一个ICMP echo Request请求：

1)主机的IP地址是172.20.171.118



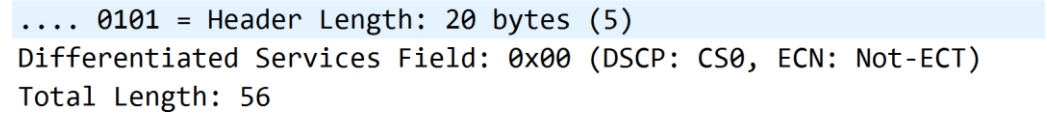
本主机IP地址

2)在IP数据包头中，上层协议字段的值是1，表示ICMP协议。



上层协议字段

3) IP头有20字节；该IP数据包的净载为36字节；确定方式为：IP 分组总长度-IP首部长度。



头部长度与总长度

4) 该IP数据包未分片；由于标志位全 0，表示允许分片但是未分片；

```
Flags: 0x00
    0... .... = Reserved bit: Not set
    .0.. .... = Don't fragment: Not set
    ..0. .... = More fragments: Not set
```

数据包未分片

分析源主机发出的一系列报文：

1)本主机发出的一系列ICMP消息中IP数据报中以下字段总在发生改变：标识ID，TTL，首部校验和，数据域。

2)除了上述四个数据段以外的数据必须保持常值。上述四个字段总要发生改变的原因：标识ID对于每个数据包来说唯一，因此每个数据包的这个字段都不一样；由于是ICMP的ping探测，因此TTL在不断变大；由于上述两个字段不断变化，因此首部校验和也需要变化；由于数据域中封装有ICMP的报文，而ICMP的头部信息不断变化，因此IP数据报的数据域也需要不断变化。

3)我看到的IP数据包Identification字段值的形式：每个报文有一个唯一的16字节的数值，且不断+1递增。

```
Identification: 0x1a5d (6749)

Identification: 0x1a5e (6750)
```

Identification字段

分析第一跳返回的ICMP消息：

1)Identification字段值为0x6a7a，TTL字段为124。

2)最近的路由器（第一跳）返回给你主机的ICMP Time-to-live exceeded消息中TTL保持不点，ID字段不断改变，原因是：第一跳路由器设置TTL字段为RFC指定的值，因此始终保持不变；而ID值标识每一个IP字段，是唯一的，因此不断改变。

分析将包改为2000字节之后主机发送的第一个ICMP Echo Request消息：

1)该消息被分解为2个数据包

2)标志位MF被置为1标识后面还有分片，该分片的数据域大小为1450B，IP总长度为1500B。

```
ICMP     1514 Echo (ping) reply    id=0x0001, seq=736/57346, ttl=124 (request in 38…
ICMP      534 Echo (ping) request  id=0x0001, seq=737/57602, ttl=1 (no response fou…
```

分为两个数据包

```
Flags: 0x20, More fragments
    0... .... = Reserved bit: Not set
    .0.. .... = Don't fragment: Not set
    ..1. .... = More fragments: Set
```

标志位MF被置为1

```
.... 0101 = Header Length: 20 bytes (5)
> Differentiated Services Field: 0x00 (DSCP: CS0, ECN: Not-ECT)
Total Length: 1500
```

总长度1500B，头部长度20B，数据长度1480B

分析将包改为3500字节之后主机发送的第一个ICMP Echo Request消息：

1)原始数据包被分成了3片。

2)这些分片中IP数据报头部标志位MF变化、片偏移变化。第一个和第二个分片标志位MF为1标识后面还有分片，第一个分片的片偏移为0，第二个为185，第三个是370。

利用 Wireshark 分析 Ethernet 数据帧

1)访问的网页为www.hit.edu.cn

2)本主机IP：172.20.171.118，目的主机IP：61.167.60.70

```
Source Address: 172.20.171.118
Destination Address: 61.167.60.70
```

本主机IP与目的主机IP

3)本主机发送的第一条HTTP报文的以太网帧结构等装了上层的IP数据，IP封装了上层的TCP数据报，TCP数据报封装了上层的HTTP数据。

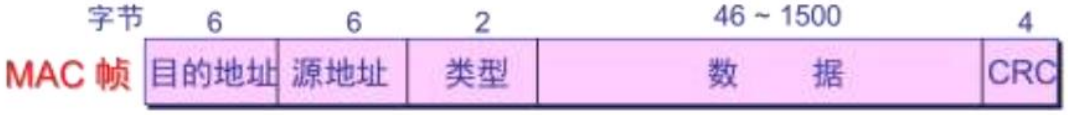
4) 以太网帧结构如下：

目的MAC、源MAC地址(各6B):若网卡的MAC地址与收到的帧的目的MAC地址匹配，或者帧的目的 MAC 地址为广播地址(FF-FF-FF-FF-FF-FF)，则网卡接收该帧，并将其封装的网络层分组交给相应的网络层协议；否则，网卡丢弃(不接收)该帧；

类型 Type2B：指示帧中封装的是哪种高层协议的分组 (如，IP 数据报、Novell IPX 数据报、AppleTalk 数据报等)；

数据(Data)(46-1500B): 指上层协议载荷；

CRC(4B): 循环冗余校验码，丢弃差错帧



以太网帧结构

5)本主机MAC地址：50:eb:71:2d:e0:56，目的主机MAC地址：44:ec:ce:d2:ff:c2；类型是 IPv4

```
> Destination: JuniperN_d2:ff:c2 (44:ec:ce:d2:ff:c2)
> Source: IntelCor_2d:e0:56 (50:eb:71:2d:e0:56)
  Type: IPv4 (0x0800)
```

主机MAC地址

6)发送报文的数据域长度范围为46B-1500B，以太网帧MTU为1500B，所以数据域最大为1500B；数据域最小值计算过程：R=10Mbps，RTTmax=512us，Lmin/R=RTTmax，则Lmin=64B，则Datamin=Lmin-18=46B。

选做内容：

利用 Wireshark 分析 ARP 协议

(1)说明 ARP 缓存中每一列的含义是什么?

ARP缓存中第一列指的是ARP协议的缓存的IP地址，第二列是MAC地址，第三列是类型，即表示是动态类型还是静态类型。

```
C:\Users\11903>arp -a

接口: 192.168.32.1 --- 0xb
  Internet 地址         物理地址              类型
  192.168.32.254        00-50-56-fc-67-dd     动态
  192.168.32.255        ff-ff-ff-ff-ff-ff     静态
  224.0.0.22            01-00-5e-00-00-16     静态
  224.0.0.251           01-00-5e-00-00-fb     静态
  224.0.0.252           01-00-5e-00-00-fc     静态
  239.255.255.250       01-00-5e-7f-ff-fa     静态
  255.255.255.255       ff-ff-ff-ff-ff-ff     静态

接口: 192.168.153.1 --- 0xe
  Internet 地址         物理地址              类型
  192.168.153.254       00-50-56-f4-0d-e5     动态
  192.168.153.255       ff-ff-ff-ff-ff-ff     静态
  224.0.0.22            01-00-5e-00-00-16     静态
  224.0.0.251           01-00-5e-00-00-fb     静态
  224.0.0.252           01-00-5e-00-00-fc     静态
  239.255.255.250       01-00-5e-7f-ff-fa     静态
  255.255.255.255       ff-ff-ff-ff-ff-ff     静态
```

ARP缓存内容

(2) ARP数据包的格式是怎样的？由几部分构成，各个部分所占的字节数是多少？

格式如下，9 部分组成，分别是：硬件类型 2B、协议类型 2B、硬件地址长度 1B、协议地址长度 1B、OP2B、源 MAC 地址 6B、源 IP 地址 4B、目的 MAC 地址 6B、目的 IP 地址 4B；

ARP请求和应答的分组格式

(3)如何判断一个ARP数据是请求包还是应答包？

当 OP 值为 1 时是请求包，当 OP 值为 2 时是应答包。

```
Opcode: request (1)
```

请求ARP

```
Opcode: reply (2)
```

响应ARP

(4)为什么ARP查询要在广播帧中传送，而ARP响应要在一个有着明确目的局域网地址的帧中传送？

查询ARP不知道目的IP对应的MAC地址，因此需要广播查询；ARP响应的时候已经从查询ARP中找到了源MAC地址，因此ARP响应可以有一个明确的目的地址。

利用 Wireshark 分析 UDP 协议

1)消息是基于UDP的还是TCP的？UDP

2)你的主机ip地址是什么？目的主机ip地址是什么？

本主机IP：172.20.171.118，目的主机IP：202.118.224.100。

```
Source Address: 172.20.171.118
Destination Address: 202.118.224.100
```

IP地址

3)你的主机发送QQ消息的端口号和QQ服务器的端口号分别是多少？

我主机发送消息的端口号为 4003，服务器端口号为 8000。

```
User Datagram Protocol, Src Port: 4003, Dst Port: 8000
    Source Port: 4003
    Destination Port: 8000
```

端口号信息

4)数据报的格式是什么样的？都包含哪些字段，分别占多少字节？

数据报格式：源端口号2B，目的端口号2B，UDP 段长度2B，校验和2B

5) 为什么你发送一个ICQ数据包后，服务器又返回给你的主机一个ICQ数据包？这UDP的不可靠数据传输有什么联系？对比前面的TCP协议分析，你能看出UDP是无连接的吗？

服务器返回ICQ用于确认。这是由于UDP提供的是不可靠的无连接的传输服务，客户端无法确认服务器是否接收到信息，因此需要一个ICQ报文表示收到。可以看出UDP是无连接的。这是因为TCP需要三次握手来建立连接，而UDP没有这个过程。同时UDP首部也没有标志位用于客户端与服务器端之间互相确认传输情况。

利用 Wireshark 分析 DNS 协议

1)本主机IP：172.20.171.118，目的主机IP：202.118.224.100。

```
Source Address: 172.20.171.118
Destination Address: 202.118.224.100
```

IP地址

2)DNS消息包括消息头部中的ID，flags等和消息体，具体格式如下：

0	1	2	3	4	5	6	7	8	9	10	11	12	13	14	15
Message ID															
QR	OPCODE				AA	TC	RD	RA	res1	res2	res3	RCODE			
QDCOUNT															
ANCOUNT															
NSCOUNT															
ARCOUNT															

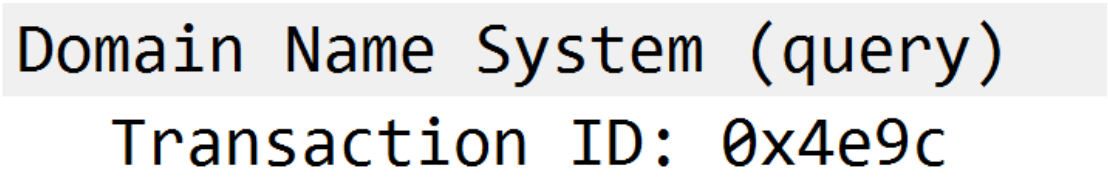
DNS消息格式

3) DNS 的下层协议是 UDP 协议，是不可靠无连接的传输服务

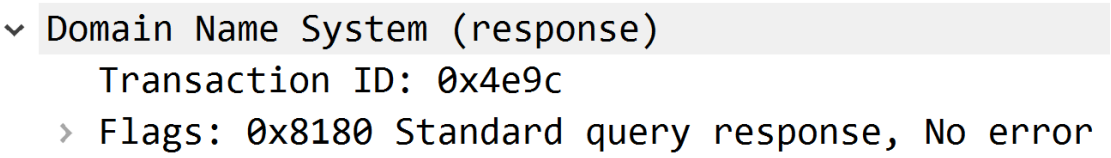
```
Time to Live: 128
Protocol: UDP (17)
Header Checksum: 0x0000 [validation disabled]
```

DNS基于UDP协议

4)DNS使用Transaction ID来标识一次查询和响应报文，长度是2B，可以发现请求和相应的响应报文的ID是一致的。

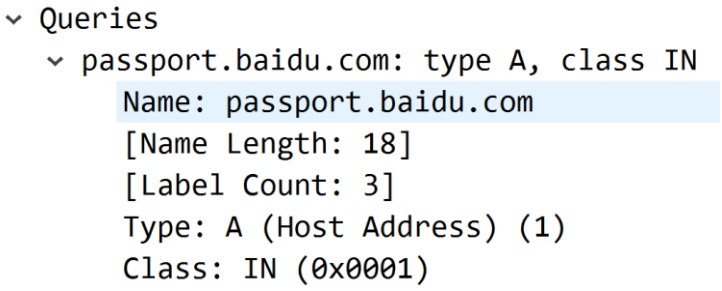


请求报文ID



响应报文ID

5)请求体内容：Name表示请求域名，Type表示请求类型，Class一般为IN。



请求体内容

6)DNS记录的不同形式如下图所示：

Name	TTL	Class	Type	Value
主机域名	60	IN	A	IP 地址
域(edu.cn)	86400	IN	NS	该权威域名解析服务器的主机域名
真实域名别名	60	IN	CNAME	真实域名
收件地址后缀	86400	IN	MX	与 Name 相对应的邮件服务器

DNS记录类型

7) DNS查询分为递归查询与迭代查询。

实验结果：

实验结果在上一部分基本展示，这一部分展示对于上述实验中出现的协议的认识。

HTTP协议：

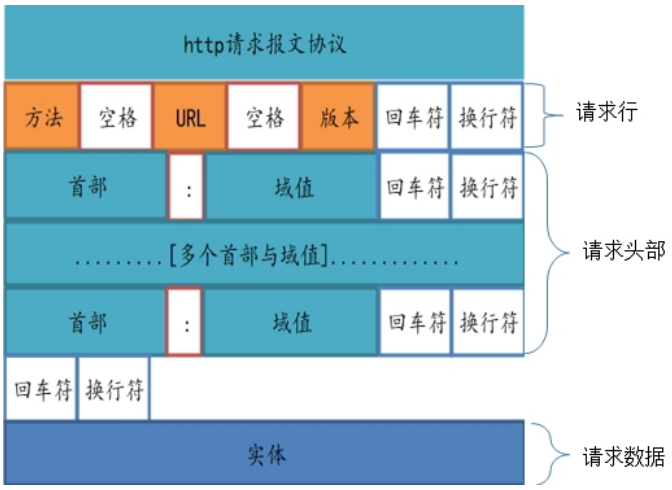
1. 简介：HTTP协议(超文本传输协议)，它是基于TCP协议的应用层传输协议，简单来说就是客户端和服务端进行数据传输的一种规则。

2. HTTP特点：a)无连接：每进行一次HTTP通信，都要断开一次TCP连接。b)无状态：HTTP 协议无法根据之前的状态进行本次的请求处理。c)灵活性：HTTP 允许传输任意类型的数据对象。正在

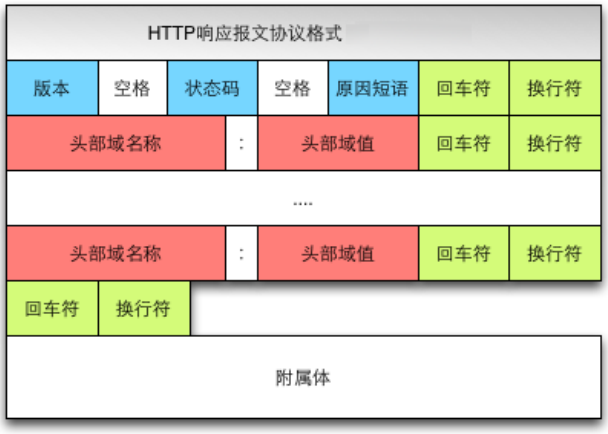
传输的类型由 Content-Type加以标记d)简单快速：客户向服务器请求服务时，只需传送请求方法和路径。请求方法常用的有 GET、HEAD、POST。每种方法规定了客户与服务器联系的类型不同。由于 HTTP 协议简单，使得 HTTP 服务器的程序规模小，因而通信速度很快。

3.HTTP请求状态行：请求行由请求Method, URL 字段和HTTP Version三部分构成, 总的来说请求行就是定义了本次请求的请求方式, 请求的地址, 以及所遵循的HTTP协议版本。

4.HTTP响应状态行：状态行由三部分组成，包括HTTP协议的版本，状态码，以及对状态码的文本描述。



HTTP请求报文



HTTP响应报文

TCP协议：

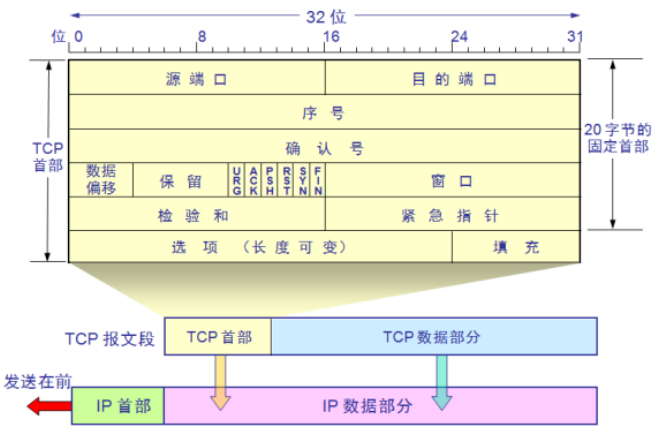
1.TCP 是面向连接的运输层协议。应用程序在使用 TCP 协议之前，必须先建立 TCP 连接。在传送数据完毕后，必须释放已经建立的 TCP 连接。

2.每一条 TCP 连接只能有两个端点，每一条 TCP 连接只能是点对点的(一对一)。

3.TCP 提供可靠交付的服务。通过 TCP 连接传送的数据，无差错、不丢失、不重复，并且按序到达。

4.TCP 提供全双工通信。TCP 允许通信双方的应用进程在任何时候都能发送数据。TCP 连接的两端都设有发送缓存和接受缓存，用来临时存放双向通信的数据。

5.面向字节流。TCP 中的“流”指的是流入到进程或从进程流出的字节序列。



TCP报文格式

IP协议：

1.主要内容：IP协议是TCP/IP协议族的核心协议，其主要包含两个方面：a)IP头部信息。IP头部信息出现在每个IP数据报中，用于指定IP通信的源端IP地址、目的端IP地址，指导IP分片和重组，以及指定部分通信行为。b)IP数据报的路由和转发。IP数据报的路由和转发发生在除目标机器之外的所有主机和路由器上。它们决定数据报是否应该转发以及如何转发。

2.特点：IP协议是TCP/IP协议族的动力，它为上层协议提供无状态、无连接、不可靠的服务。

3.任务：负责对数据包进行路由选择和存储转发。

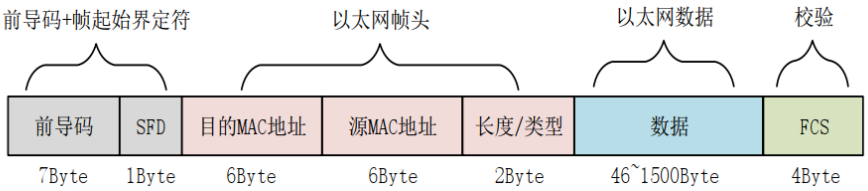
4.IP协议：逐跳发送模式；根据数据包的目的地 IP 地址决定数据如何发送；如果数据包不能直接发送至目的地，IP 协议负责寻找一个合适的下一跳路由器，并将数据包交付给该路由器转发；

IP协议

Ethernet协议：

1.定义：Ethernet以太网协议，用于实现链路层的数据传输和地址封装（MAC），由DIX联盟（Digital、Intel、Xero）开发

2.任务：两个相邻节点之间传送数据时，数据链路层将网络层交下来的 IP 数据报组装成帧，在两个相邻的链路上传送帧（frame)。每一帧包括数据和必要的控制信息。



以太网帧格式

ARP协议：

简介：ARP协议是地址解析协议（Address Resolution Protocol）是通过解析IP地址得到MAC地址的，是一个在网络协议包中极其重要的网络传输协议，它与网卡有着极其密切的关系，在TCP/IP分层结构中，把ARP划分为网络层，为什么呢，因为在网络层看来，源主机与目标主机是通过IP地址进行识别的，而所有的数据传输又依赖网卡底层硬件，即链路层，那么就需要将这些IP地址转换为链路层可以识别的东西，在所有的链路中都有着自己的一套寻址机制，如在以太网中使用MAC地址进行寻址，以标识不同的主机，那么就需要有一个协议将IP地址转换为MAC地址，由此就出现了ARP协议。

ARP报文格式

UDP协议：

简介：UDP是一个简单的面向消息的传输层协议，尽管UDP提供标头和有效负载的完整性验证（通过校验和），但它不保证向上层协议提供消息传递，并且UDP层在发送后不会保留UDP 消息的状态。因此，UDP有时被称为不可靠的数据报协议。如果需要传输可靠性，则必须在用户应用程序中实现。UDP是基于IP的简单协议，不可靠的协议。

UDP的优点：简单，轻量化。

UDP的缺点：没有流控制，没有应答确认机制，不能解决丢包、重发、错序问题。

UDP报文格式

DNS协议：

简介：DNS 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 协议建立在 UDP 或 TCP 协议之上，默认使用 53 号端口。当前，对于每一级域名长度的限制是63个字符，域名总长度则不能超过253个字符。DNS协议是用来将域名转换为IP地址（也可以将IP地址转换为相应的域名地址）。

DNS报文格式

问题讨论：

对实验过程中的思考问题进行讨论或回答。

实验过程中遇到了关于ICMP协议的相关内容，在此对ICMP协议进行一定的展示。ICMP（Internet Control Message Protocol）Internet控制报文协议。它是TCP/IP协议簇的一个子协议，用于在IP主机、路由器之间传递控制消息。控制消息是指网络通不通、主机是否可达、路由是否可用等网络本身的消息。这些控制消息虽然并不传输用户数据，但是对于用户数据的传递起着重要的作用。

事实上ICMP是IP的一个组成部分，与 IP 协议、ARP 协议、RARP 协议及 IGMP 协议共同构成 TCP/IP 模型中的网络层。

心得体会：

结合实验过程和结果给出实验的体会和收获。

1. 对于计算机网络模型有了更加深入的认识。
2. 尤其是对于应用层、传输层、网络层、链路层的一些协议有了更加深入的了解。通过对这些协议报文的抓包分析，对于其结构与工作原理的认识更加深刻。