Mininet实验环境报告

李昊宸

2017K8009929044

（一）互联网协议实验

一、实验内容

1. 在节点h1上开启wireshark抓包，用wget下载www.baidu.com页面

2. 调研说明wireshark抓到的几种协议：ARP, DNS, TCP, HTTP

3. 调研解释h1下载baidu页面的整个过程

二、实验流程

1. 搭建实验环境

$ sudo mn --nat

mininet> xterm h1

h1 # echo "nameserver 1.2.4.8" > /etc/resolv.conf

2.启动wireshark

h1 # wireshark &

3.抓取网站

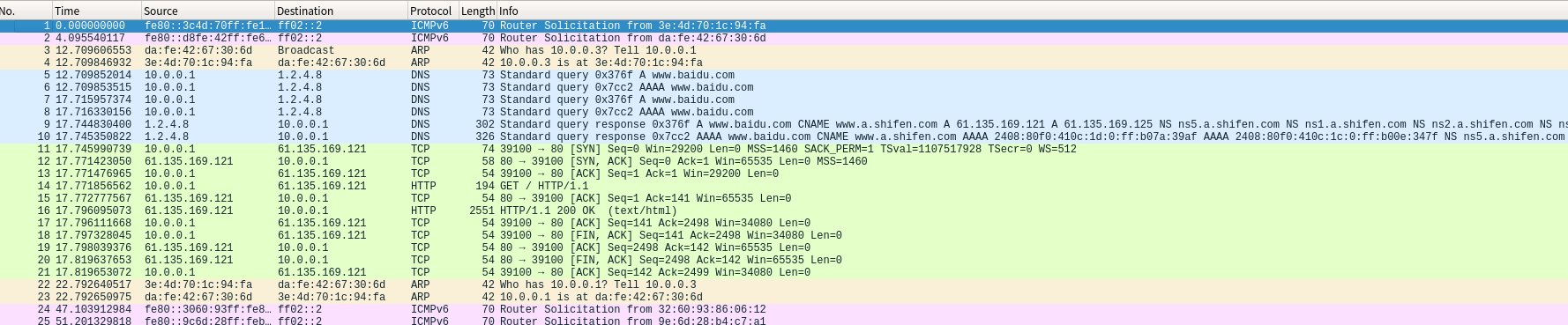
h1 # wget [www.baidu.com](http://www.baidu.com)

三、实验结果及分析

1. 实验结果



图一 xterm下h1的保存结果



图二 wireshark抓取的报文

2. 实验分析

从wireshark抓取的报文中可以看出，整个过程中共有四种类型的报文：ARP、DNS、TCP和HTTP。下面逐一阐述这四种协议。

ARP协议：Address Resolution Protocol，地址解析协议。在以太网中，一个主机要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。网络层中，实际传输的是“帧”，帧内含有传送目标主机的MAC地址，该地址通过地址解析协议获得的。也就是说，ARP协议的基本功能就是通过目标设备的IP地址（32位），查询目标设备的MAC地址（48位），以保证通信的顺利进行。

假设我们的计算机IP地址是10.0.0.1，要执行这个命令：ping 10.0.0.2。该命令会通过ICMP协议发送ICMP数据包。该过程需要经过下面的步骤：

1. 应用程序构造数据包，该示例是产生ICMP包，被提交给内核（网络驱动程序）

2. 内核检查是否能够转化该IP地址为MAC地址，也就是在本地的ARP缓存中查看IP-MAC对应表

　　3. 如果存在该IP-MAC对应关系，那么跳到步骤7；如果不存在该IP-MAC对应关系，那么接续下面的步骤

　　4. 内核进行ARP广播，目的地的MAC地址是FF-FF-FF-FF-FF-FF，ARP命令类型为REQUEST（1），其中包含有自己的MAC地址

　　5. 当10.0.0.2主机接收到该ARP请求后，就发送一个ARP的REPLY（2）命令，其中包含自己的MAC地址

　　6. 本地获得10.0.0.2主机的IP-MAC地址对应关系，并保存到ARP缓存中

7. 内核将把IP转化为MAC地址，然后封装在以太网头结构中，再把数据发送出去

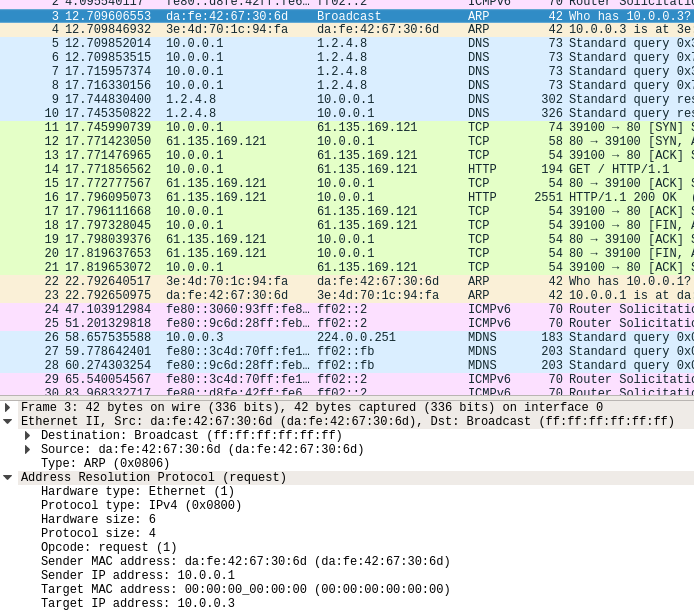
在实验中，我们需要获得一个网站的内容，所以该网站的MAC地址起初并不在主机的ARP缓存中，所以抓取到两个ARP报文，一个为发送报文，一个为回应报文。

发送报文：ARP请求包：

opcode为request（1），发送方为h1（IP地址10.0.0.1），传输方式为广播Broadcast。

ARP回应包：

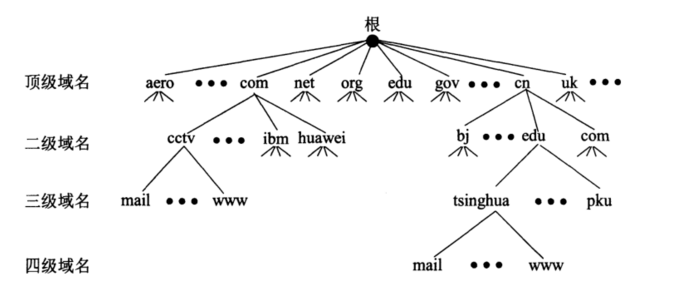
opcode为request（2），发送方的mac地址为3e:4d:70:1c:94:fa,表明h1想询问的mac地址。目标方IP地址为10.0.0.1，即发送请求的主机h1，传输方式为单播，



图三 ARP报文 Ethernet < ARP

DNS协议：Domain Name System，域名系统。该系统用于命名组织到域层次结构中的计算机和网络服务。域名是由圆点分开一串单词或缩写组成的，每一个域名都对应一个惟一的IP地址，在Internet上域名与IP地址之间是一一对应的，DNS就是进行域名解析的服务

器。



图四 域名分级

根域名服务器：最高层次的域名服务器，本地域名服务器解析不了的域名就会向其求助

顶级域名服务器：负责管理在该顶级域名服务器下注册的二级域名

权限域名服务器：负责一个区域的域名解析工作

本地域名服务器：当一个主机发出 DNS 查询请求时，这个查询请求首先发给本地域名服务器

下面以一个例子说明在无缓存的情况下，通过DNS来查询域名的步骤。假设域名为m.xyz.com的主机想要查询y.abc.com这个域名对应的IP地址，那么它就会按照以下方式方式进行查询：

1. 主机m.xyz.com先向本地域名服务器dns.xyz.com进行递归查询。

2. 本地域名服务器无法给出IP地址，所以本地域名服务器向（离自己最近的）根域名服务器查询，这是的查询已经变为了迭代查询。

3. 根域名服务器根据本地域名服务器发送的报文，知道了下一步应该查询的是哪个顶级域名服务器，这时根域名服务器告诉本地域名服务器，下一步应该查询的顶级域名服务器dns.com的IP地址。

4. 本地域名服务器向顶级域名服务器dns.com发送请求查询。

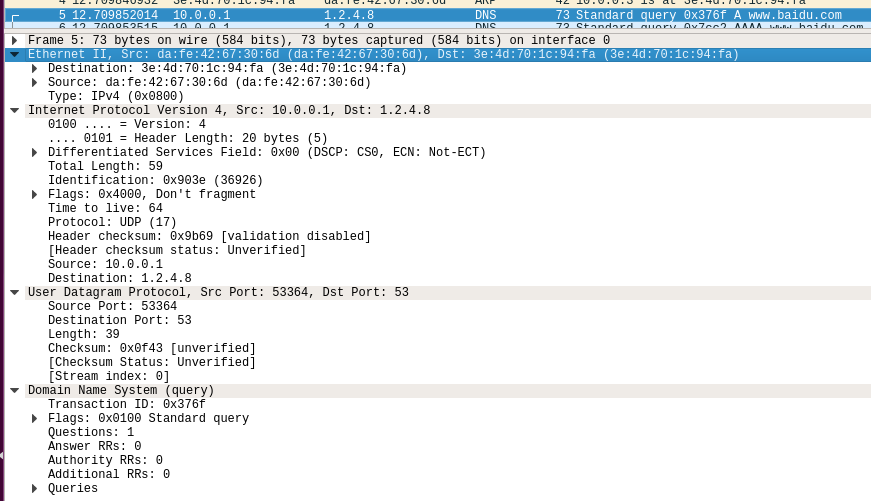
5. 顶级域名服务器dns.com告诉本地域名服务器，下一步应该查询的权限域名服务器dns.abc.com的IP地址。

6. 本地域名服务器向权限域名服务器dns.abc.com发送请求查询。

7. 权限域名服务器dns.abc.com告诉本地域名服务器想要查询的域名y.abc.com的IP地址。

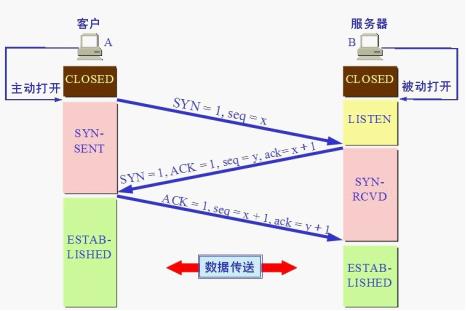
8. 本地域名服务器在拿到IP地址后，将IP地址返回给主机m.xyz.com。

但是，通常情况下，本地域名服务器中会有常用域名的缓存，比如本例中的www.baidu.com，所以可以直接将IP地址返回。



图五 DNS报文 Ethernet < IP < UDP < DNS

TCP协议：Transmission Control Protocol，传输控制协议。TCP层是位于IP层之上，应用层之下的传输层。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把数据流分割成适当长度的报文段。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为了保证不发生丢包，就给每个字节一个序号，同时序号也保证了传送到接收端实体的包的按序接收。然后接收端实体对已成功收到的字节发回一个相应的确认（ACK）；如果发送端实体在合理的往返时延（RTT）内未收到确认，那么对应的数据（假设丢失了）将会被重传。TCP用一个校验和函数来检验数据是否有错误；在发送和接收时都要计算和校验。



图六 TCP建立连接的三次握手

TCP建立连接：三次握手

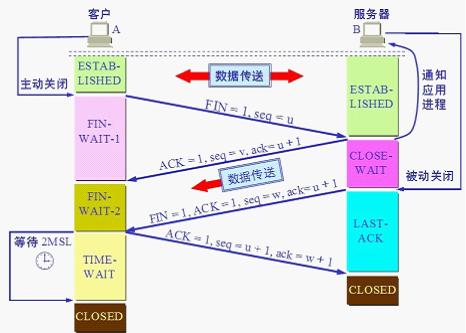
1. A 的 TCP 向 B 发出连接请求报文段,其首部中的同步位 SYN = 1,并选择序号 seq = x,表明传送数据时的第一个数据字节的序号是 x

2. B 的 TCP 收到连接请求报文段后,如同意,则发回确认(B 在确认报文段中应使 SYN = 1,使 ACK = 1,其确认号ack = x﹢1,自己选择的序号 seq = y)

3. A 收到此报文段后向 B 给出确认,其 ACK = 1,确认号 ack = y﹢1(A 的 TCP 通知上层应用进程,连接已经建立,B 的 TCP 收到主机 A 的确认后,也通知其上层应用进程：TCP 连接已经建立)

该过程对应图三的11-13号报文。

TCP协议的数据传送使用滑动窗口方式进行传送。在此不再赘述。



图七 TCP释放连接的四次握手

TCP释放连接：四次握手

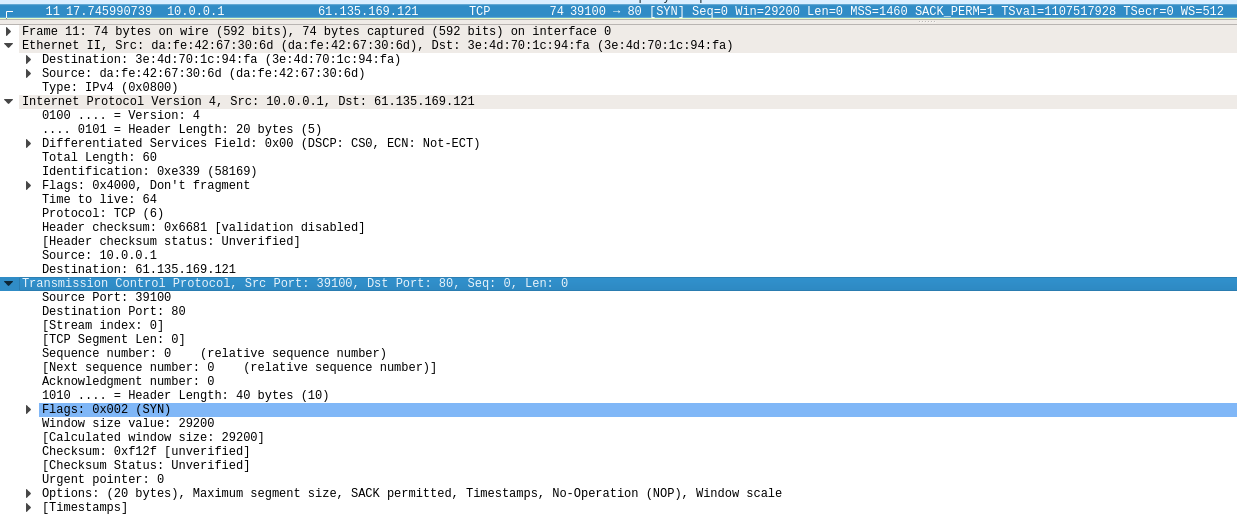
1. 数据传输结束后,通信的双方都可释放连接.现在 A 的应用进程先向其 TCP 发出连接释放报文段,并停止再发送数据,主动关闭 TCP 连接(A 把连接释放报文段首部的 FIN = 1,其序号seq = u,等待 B 的确认)

2. B 发出确认,确认号 ack = u＋1,而这个报文段自己的序号 seq = v(TCP 服务器进程通知高层应用进程.从 A 到 B 这个方向的连接就释放了,TCP 连接处于半关闭状态.B 若发送数据,A 仍要接收)

3. 若 B 已经没有要向 A 发送的数据,其应用进程就通知 TCP 释放连接

4. A 收到连接释放报文段后,必须发出确认,在确认报文段中 ACK = 1,确认号 ack=w﹢1,自己的序号 seq = u + 1

该过程对应图三的18-21号报文。



图八 TCP报文 Ethernet < IP < TCP

HTTP协议：Hyper Text Transfer Protocol，超文本传输协议。是用于从WWW服务器传输超文本到本地浏览器的传送协议。

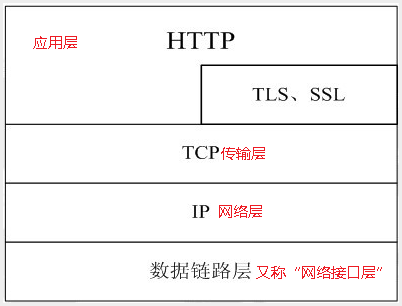
一次HTTP操作称为一个事务，其工作过程可分为四步：

1. 首先客户机与服务器需要建立连接。只要单击某个超级链接，HTTP的工作开始。

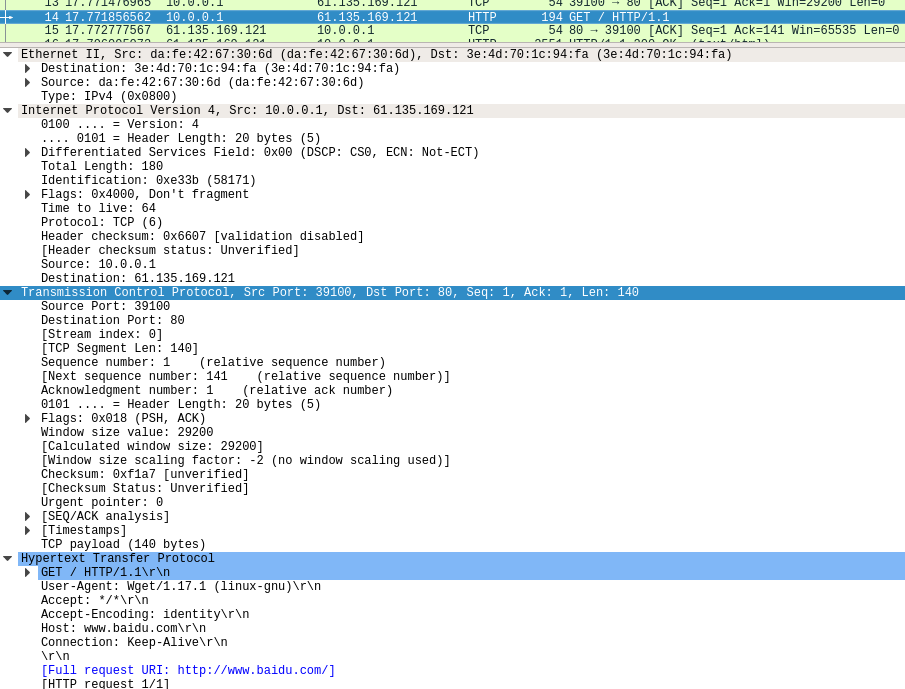
2. 建立连接后，客户机发送一个请求给服务器，请求方式的格式为：统一资源标识符（URL）、协议版本号，后边是MIME信息包括请求修饰符、客户机信息和可能的内容。

3. 服务器接到请求后，给予相应的响应信息，其格式为一个状态行，包括信息的协议版本号、一个成功或错误的代码，后边是MIME信息包括服务器信息、实体信息和可能的内容。

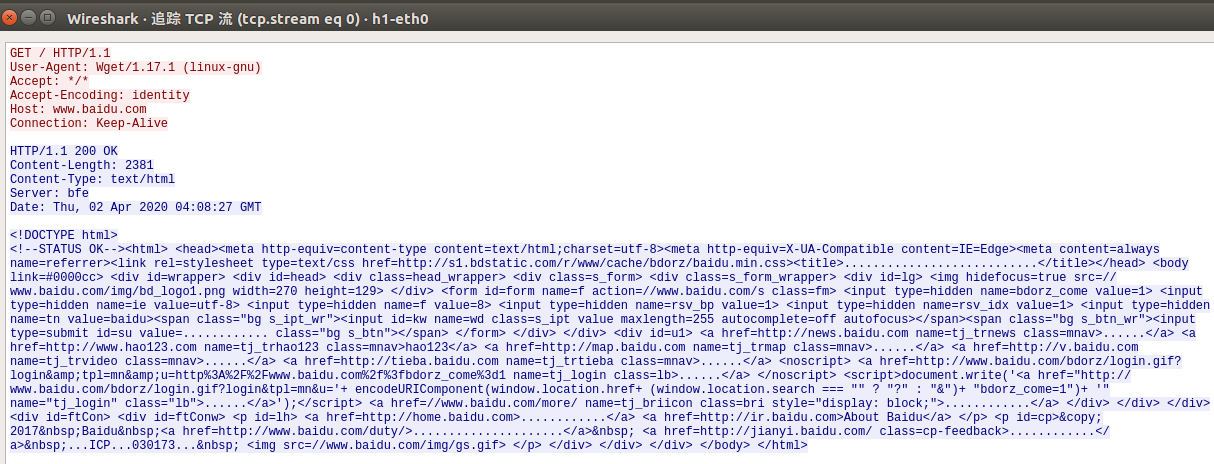
4. 客户端接收服务器所返回的信息通过浏览器显示在用户的显示屏上，然后客户机与服务器断开连接。



图九 HTTP模型



图十 HTTP报文 Ethernet < IP < TCP < HTTP



图十一 追踪到的TCP流

红色区域为源到目的地，本次实验中为主机h1，使用http1.1协议，目的主机为百度。

蓝色区域为目的地到源，本次实验中目的主机为百度的服务器，连接方式为text/html，接下来的蓝色区域就是将百度页面的html编码发送给h1主机。

整个过程：

1. h1发现www.baidu.com不在本地局域网内，于是通过ARP协议获取路由器的MAC地址。h1发送广播，询问路由器的MAC地址，路由器接收到包后将其MAC地址返回h1。

2. h1通过DNS协议将www.baidu.com该域名转化为IP地址。

3. h1将MAC地址和IP地址封装到帧中，通过TCP协议与百度服务器建立连接。

4. h1通过http协议获取数据

（二）流完成时间实验

一、实验内容

1. 在给定带宽、延迟和文件大小前提下，查看流完成时间

2. 变化文件大小(10MB, 100MB)、带宽(10Mbps, 100Mbps, 1Gbps)、延迟(10ms, 100ms)，查看不同条件下的流完成时间

二、实验流程

1. 搭建实验环境

$ sudo python fct\_exp.py

mininet> xterm h1 h2

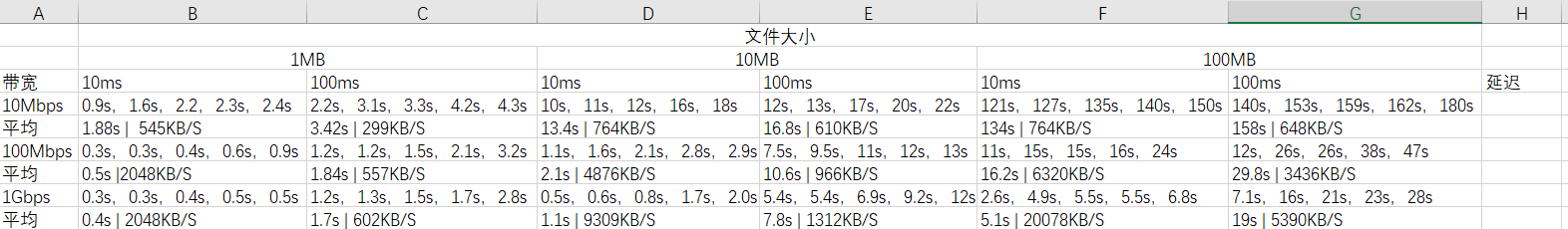
2. 生成传输文件

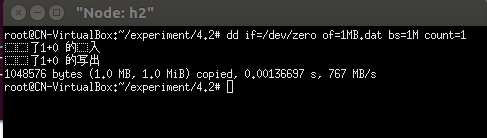
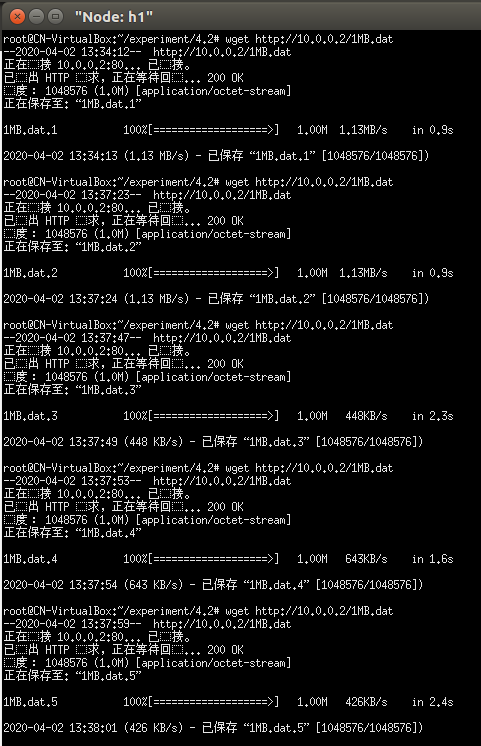
h2 # dd if=/dev/zero of=1MB.dat bs=1M count=1

3. 发起传输

h1 # wget http://10.0.0.2/1MB.dat

三、实验结果及分析

1. 实验结果

图十二 流传输时间记录

图十三 生成传输文件示例

图十四 发起传输示例

2. 实验分析

1. 10ms延迟

|  |  |  |  |
| --- | --- | --- | --- |
| 10ms延迟 | 文件大小 | | |
| 带宽 | 1MB | 10MB | 100MB |
| 10Mbps | 545KB/s | 764 KB/s | 764 KB/s |
| 100Mbps | 2048 KB/s | 4876 KB/s | 6320 KB/s |
| 1Gbps | 2048 KB/s | 9309 KB/s | 20078 KB/s |

表一 10ms延迟下的平均传输速率表

文件大小 MB （对数正则化）

传输速率 KB/S (对数正则化)

图十五 10ms延迟下的传输速率-文件大小双对数图

分析：

作图使用的数据为表格中的数据。首先，使用了传输速率作为流完成时间的替代，这样一来数据更加直观。其次，为了放大数据之间的直观程度，对横轴和纵轴均取了对数坐标，纵坐标的底数为545，即针对第一个点的传输速率进行正则化；横坐标以10为底数，即针对第一个点的文件大小进行正则化；随后移动坐标轴，使原点位于第一个点处，做出传输速率-文件大小的双对数正则化折线图。需要说明的是，为了数据的直观性，坐标轴上标注的值并非取对数之后的值，而为取对数之前的值。

现象复现了课件中要求的目标，即在延迟固定、带宽固定的情况下，增大文件大小，传输速率呈现出增长速率逐渐减小的增长，最终会逐渐逼近带宽的理论值。

解释：

一部分原因是TCP传输需要经历建立传输连接的三次握手阶段和释放连接的四次握手阶段，该部分需要占用一定固定的时长，文件越小，该部分时间占整个传输时间的比重越大，导致直观上的传输速率的下降。

另一个原因原因是TCP的慢启动机制。TCP 协议为了做到效率与可靠性的统一，设计了慢启动（slow start）机制。开始的时候，发送得较慢，然后根据丢包的情况，调整速率：如果不丢包，就加快发送速度；如果丢包，就降低发送速度。

Linux 内核里面设定了（常量TCP\_INIT\_CWND），刚开始通信的时候，发送方一次性发送10个数据包，即"发送窗口"的大小为10。然后停下来，等待接收方的确认，再继续发送。默认情况下，接收方每收到两个 TCP 数据包，就要发送一个确认消息。发送方有了这两个信息，再加上自己已经发出的数据包的最新编号，就会推测出接收方大概的接收速度，从而降低或增加发送速率。在实际操作中，可以明显的观察到刚启动时的速率往往比较低下，随着发送时间的延长，速率才会被提升到逐渐逼近带宽的水平。对于较小的文件，带宽往往还没来得及拉满传输就结束了，这直接导致了直观上的小文件传输速率远低于大文件的传输速率。另一方面，固定带宽的文件传输速率并不伴随传输文件变大而线性增长，是因为文件在变得足够大后，TCP有充足的时间测试满带宽，理想情况下之后的瞬时传输速率与带宽相同而并不会继续增长，所以传输一个无穷大的文件的传输速率就是带宽完全拉满的速率。

2. 100ms延迟

|  |  |  |  |
| --- | --- | --- | --- |
| 100ms延迟 | 文件大小 | | |
| 带宽 | 1 | 10 | 100 |
| 10Mbps | 299 | 610 | 648 |
| 100Mbps | 557 | 966 | 3436 |
| 1Gbps | 602 | 1312 | 5390 |

表二 100ms延迟下的平均传输速率表

图十六 100ms延迟下的传输速率-文件大小双对数图

分析：

发现对于较大带宽（100Mbps和1Gbps）而言，在100ms延迟下，增大文件大小，传输速率提升的速率反而增加，看似与10ms延迟时出现了相矛盾的现象。但是低带宽（10Mbps）表现出的行为却与10ms延迟时更为相似。

解释：

延迟增大到100ms，对于低带宽而言，延迟的量级与传输速率的量级较为接近，也就是延迟并未超过传输时间，一定程度上可以视作略微降低了带宽，于是低带宽表现出和低延迟相近的曲线。

但是对于高带宽而言，在较小文件的传输上，100ms的延迟甚至要超过传输的时间，占到整个传输过程的大部分，在这个意义上，慢启动带来的影响远远不及降低延迟所占时间比重带来的速率提升，于是在起初的这段曲线呈现出加速上升的趋势。但在延迟的影响降到较低的比重时，慢启动带来的影响开始呈现，以及速率逐渐可以拉满，随后依旧会出现和10ms延迟时类似的现象。