Mininet 实验环境

中国科学院大学

蔡润泽 2017K8009908018

2020 年 9 月 19 日

1. 互联网协议实验

（1） 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

- 几种协议的运行机制

（2） 实验流程

1） 安装 wireshark，并搭建 mininet 实验环境

2） 在 xterm 中通过 wget www.baidu.com 实现抓包，并通过 wireshark 了解相关信息

（3） 实验结果和分析

1） Wireshark 抓包结果如下：

No.	Time	Source	Destination	Protocol	Length	Info
1	0.000000000	10.0.0.1	1.2.4.8	DNS	73	Standard query 0x393a A www.baidu.com
2	0.000158024	10.0.0.1	1.2.4.8	DNS	73	Standard query 0xd759 AAAA www.baidu.com
3	0.057719127	1.2.4.8	10.0.0.1	DNS	302	Standard query response 0x393a A www.baidu.com CNAME www.a.shifen…
4	0.059931259	1.2.4.8	10.0.0.1	DNS	157	Standard query response 0xd759 AAAA www.baidu.com CNAME www.a.shi…
5	0.060459632	10.0.0.1	180.101.49.12	TCP	74	46286 → 80 [SYN] Seq=0 Win=42340 Len=0 MSS=1460 SACK_PERM=1 TSval…
6	0.113426727	180.101.49.12	10.0.0.1	TCP	62	80 → 46286 [SYN, ACK] Seq=0 Ack=1 Win=32768 Len=0 MSS=1460 WS=2
7	0.113508702	10.0.0.1	180.101.49.12	TCP	54	46286 → 80 [ACK] Seq=1 Ack=1 Win=42496 Len=0
8	0.113768855	10.0.0.1	180.101.49.12	HTTP	194	GET / HTTP/1.1
9	0.114091810	180.101.49.12	10.0.0.1	TCP	54	80 → 46286 [ACK] Seq=1 Ack=141 Win=32768 Len=0
10	0.164985867	180.101.49.12	10.0.0.1	TCP	1414	80 → 46286 [PSH, ACK] Seq=1 Ack=141 Win=32768 Len=1360 [TCP segme…
11	0.165057772	10.0.0.1	180.101.49.12	TCP	54	46286 → 80 [ACK] Seq=141 Ack=1361 Win=42496 Len=0
12	0.164987330	180.101.49.12	10.0.0.1	HTTP	1191	HTTP/1.1 200 OK (text/html)
13	0.165064905	10.0.0.1	180.101.49.12	TCP	54	46286 → 80 [ACK] Seq=141 Ack=2498 Win=41984 Len=0
14	0.166570372	10.0.0.1	180.101.49.12	TCP	54	46286 → 80 [FIN, ACK] Seq=141 Ack=2498 Win=42496 Len=0
15	0.166780003	180.101.49.12	10.0.0.1	TCP	54	80 → 46286 [ACK] Seq=2498 Ack=142 Win=32768 Len=0
16	0.208214185	180.101.49.12	10.0.0.1	TCP	54	80 → 46286 [FIN, ACK] Seq=2498 Ack=142 Win=32768 Len=0
17	0.208240887	10.0.0.1	180.101.49.12	TCP	54	46286 → 80 [ACK] Seq=142 Ack=2499 Win=42496 Len=0
18	5.150342759	96:16:06:88:d9:d6	fe:61:b0:85:06:ad	ARP	42	Who has 10.0.0.1? Tell 10.0.0.3
19	5.150361141	fe:61:b0:85:06:ad	96:16:06:88:d9:d6	ARP	42	10.0.0.1 is at fe:61:b0:85:06:ad
20	9.501787793	fe80::9416:6ff:fe88…	ff02::2	ICMPv6	70	Router Solicitation from 96:16:06:88:d9:d6
21	42.269822123	fe80::fc61:b0ff:fe8…	ff02::2	ICMPv6	70	Router Solicitation from fe:61:b0:85:06:ad
22	58.653981268	fe80::10c4:2aff:fe0…	ff02::2	ICMPv6	70	Router Solicitation from 12:c4:2a:0f:ee:6c
23	67.069485990	10.0.0.3	224.0.0.251	MDNS	87	Standard query 0x0000 PTR _ipps._tcp.local, "QM" question PTR _ip…
24	68.302434914	fe80::9416:6ff:fe88…	ff02::fb	MDNS	107	Standard query 0x0000 PTR _ipps._tcp.local, "QM" question PTR _ip…
25	68.940340990	fe80::1c21:abff:fe3…	ff02::fb	MDNS	107	Standard query 0x0000 PTR _ipps._tcp.local, "QM" question PTR _ip…
26	75.036920650	fe80::1c21:abff:fe3…	ff02::2	ICMPv6	70	Router Solicitation from 1e:21:ab:31:40:4f

在 h1 下载 baidu 页面的整个过程主要依次出现了四种协议：
DNS、TCP、HTTP 和 ARP

2） DNS 协议是一种用来将域名转化为 IP 地址的协议。在上图中首先由本地 IP 地址 10.0.0.1 作为 source，向目的地址 1.2.4.8（DNS 服务器 IP）发送查询的请求。然后 1.2.4.8 返回给本地 IP 其解析结果(其中提到 www.baidu.com 的 CNAME 为 www.a.shifen.com)。 其解析的 IP 地址根据上图可以看出为 180.101.49.12。

3） TCP 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。

上图中出现的 SYN 用于建立 TCP/IP 时的握手连接。由客户端向服务端发送 SYN。服务端向客户端发送 SYN+ACK 的响应报文，再由客户端向服务端发送一个 ACK 响应报文，通过三次握手建立一个完整的连接。

相应的，FIN 表示关闭连接，PSH 代表着存在数据传输。

4） HTTP 是一个的请求 - 响应协议，通常运行在 TCP 之上。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。请求和响应消息的头以 ASCII 码形式给出；而消息内容则具有一个类似 MIME 的格式。

在上图中出现了 GET 请求，即从指定资源请求数据。（由于在 xterm h1 中输入了 wget）。

5） ARP 协议是一种根据 IP 地址来获得 MAC 地址的一个 TCP/IP 协议。主机发送信息时，将包含目标 IP 地址的 ARP 请求广播到局域网络上的所有主机，并接收返回消息，以此来确定目标的 MAC 地址。收到返回消息后将该 IP 地址和 MAC 地址存入本机 ARP，这样可以在下次请求查询时节约资源。

图上出现了 ARP 协议。 96:16:06:88:d9:d6 为对应本地 IP 为 10.0.0.1 的 MAC 地址，而 fe:61:b0:85:06:ad 为对应本地 IP 为 10.0.0.3 的 MAC 地址。一开始 10.0.0.3 询问哪个 MAC 地址对应的 IP 为 10.0.01，然后 10.0.0.1 对应的 MAC 地址再回复给 10.0.0.3。

6） 另外，不同层次之间存在着封装。Ethernet 解决了子网内部的点对点通信，而上层的 IP 解决了多个局域网之间的通讯。而 UDP<DNS 和 TCP<HTTP 又是对 IP 的封装

```
▸ Ethernet II, Src: fe:61:b0:85:06:ad (fe:61:b0:85:06:ad), Dst: 96:16:06:88:d9:d6 (96:16:06:8
▸ Internet Protocol Version 4, Src: 10.0.0.1, Dst: 1.2.4.8
▸ User Datagram Protocol, Src Port: 57110, Dst Port: 53
▸ Domain Name System (query)
```

该图展现了 Ethernet < IP <UDP<DNS

```
▸ Ethernet II, Src: fe:61:b0:85:06:ad (fe:61:b0:85:06:ad), Dst: 96:16:06:88:d9:d6 (96:16:06
▸ Internet Protocol Version 4, Src: 10.0.0.1, Dst: 180.101.49.12
▸ Transmission Control Protocol, Src Port: 46286, Dst Port: 80, Seq: 1, Ack: 1, Len: 140
▸ Hypertext Transfer Protocol
```

该图展现了 Ethernet < IP < TCP<HTTP

2. 流完成时间实验

（1） 实验内容

1） 利用 fct_exp.py 脚本复现 fct 和带宽增长之间的关系图

2） 调研解释上图中的现象：

- TCP 传输、慢启动机制

（2） 实验流程

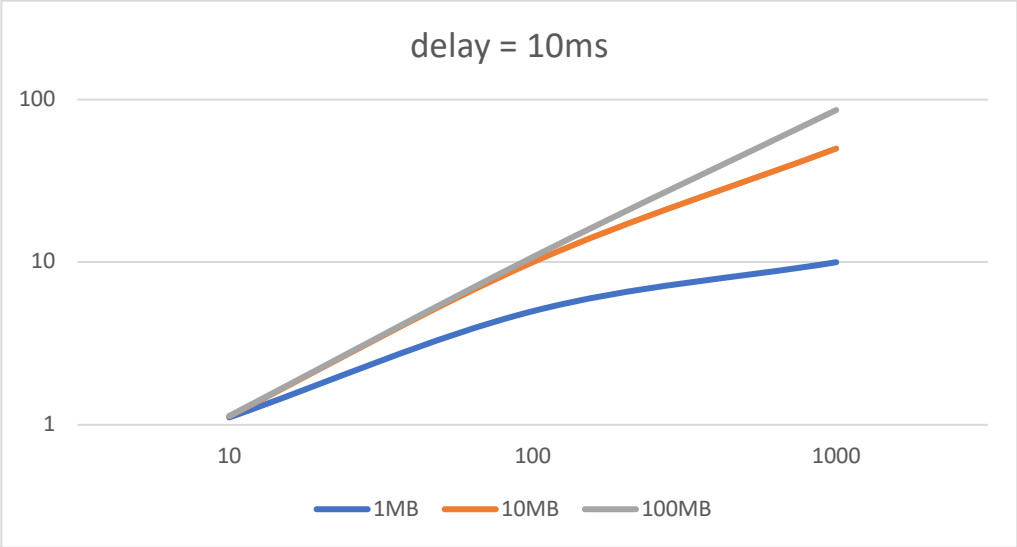
1） mininet 实验环境

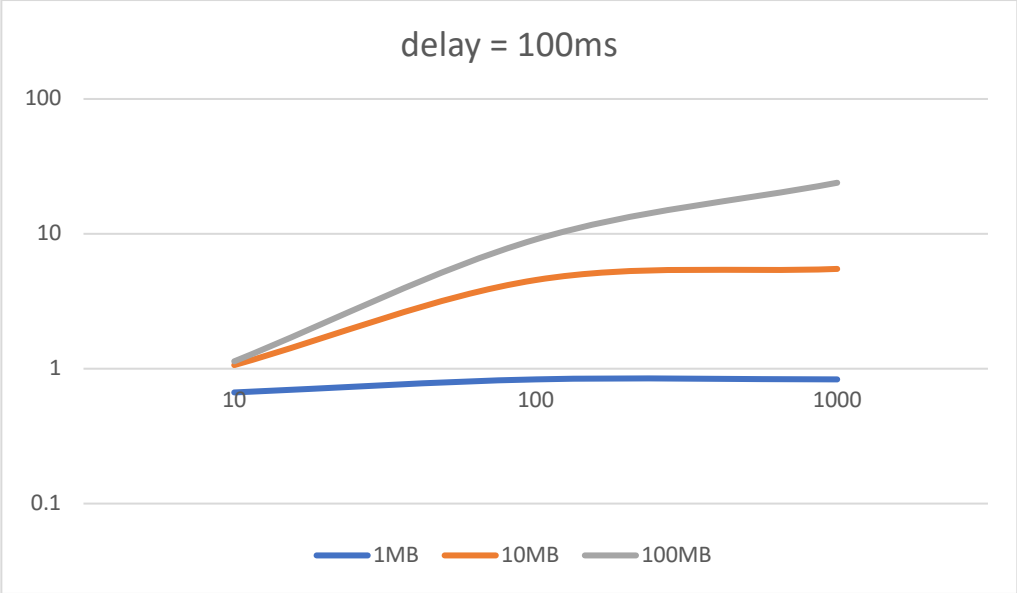
2） h2 分配一个数据包空间，在 xterm h1 中通过 wget http://10.0.0.2/1MB.dat 实现抓包，并统计传输时间

3） 修改数据包大小、延迟和带宽等参数，得出结果。

（3） 实验结果和分析

通过分析作图，可以得出以下两张图（分别是延迟为 10ms 和延迟为 100ms 的图）





原因分析：

TCP 协议利用慢启动和拥塞避免机制来避免传输拥塞。

慢启动算法在主机刚开始发送数据报的时先探测网络的状况。如果网络状况良好，发送方每发送一次文段都能正确的接受确认报文段。那么就从小到大的增加拥塞窗口的大小，即增加发送窗口的大小。而正常情况下拥塞窗口的大小随着一次往返而加倍。这解释了上图中增长非线性的问题。

另外拥塞窗口的增长存在阈值，其不会无限制的增长下去。因此对于小数据包而言，其受慢启动机制影响会比大数据包更显著。

参考文献或网站

[1] https://baike.baidu.com/item/ARP/609343?fromtitle=ARP 协议&fromid=1742212&fr=aladdin
[2] https://baike.baidu.com/item/TCP/33012?fromtitle=TCP 协议&fromid=8988699&fr=aladdin