

Homework 4

2017年3月8日 18:29

HW4. Using Wireshark

Description

Steps:

- Download & Install Wireshark
- Capture all packages when you access <http://www.zju.edu.cn> (using filter)
- Using Wireshark to analysis the packets
- Write a report (in pdf) to describe the procedure & results of the capture & analysis process

Deliverables

Mail the deliverables to TA, with title: SEC2017-HW4-ID-NAME

- The captured packets
- The report

Deadline

- 23:59:59, Apr. 23rd , 2017

1. 重启网卡服务

net start npf

2. ping

C:\WINDOWS\system32>ping [www.zju.edu.cn](http://www.zju.edu.cn)

正在 Ping [www.zju.edu.cn](http://www.zju.edu.cn) [10.203.5.199] 具有 32 字节的数据:  
来自 10.203.5.199 的回复: 字节=32 时间=2ms TTL=60  
来自 10.203.5.199 的回复: 字节=32 时间=4ms TTL=60  
来自 10.203.5.199 的回复: 字节=32 时间=3ms TTL=60  
来自 10.203.5.199 的回复: 字节=32 时间=1ms TTL=60

10.203.5.199 的 Ping 统计信息:  
数据包: 已发送 = 4, 已接收 = 4, 丢失 = 0 (0% 丢失),  
往返行程的估计时间 (以毫秒为单位):  
最短 = 1ms, 最长 = 4ms, 平均 = 2ms

3. host 10.203.5.199

1. 捕捉过滤器

捕捉过滤器的语法与其它使用Lipcap (Linux) 或者Winpcap (Windows) 库开发的软件一样, 比如著名的TCPdump, 捕捉过滤器必须在开始捕捉前设置完毕, 这一点跟显示过滤器是不同的。

设置捕捉过滤器的步骤是:

- 选择 capture -> options。
- 填写“capture filter”栏或者点击“capture filter”按钮为您的过滤器起一个名字并保存, 以便在今后的捕捉中继续使用这个过滤器。
- 点击开始 (Start) 进行捕捉。

tcp dst port 3128

显示目的TCP端口为3128的封包。

ip src host 10.1.1.1

显示来源IP地址为10.1.1.1的封包。

host 10.1.2.3

显示目的或来源IP地址为10.1.2.3的封包。

src portrange 2000-2500

显示来源为UDP或TCP, 并且端口号在2000至2500范围内的封包。

not icmp

显示除了icmp以外的所有封包。(icmp通常被ping工具使用)

src host 10.7.2.12 and not dst net 10.200.0.0/16

显示来源IP地址为10.7.2.12, 但目的地不是10.200.0.0/16的封包。

(src host 10.4.1.12 or src net 10.6.0.0/16) and tcp dst portrange 200-10000 and dst net 10.0.0.0/8

4. <http://skypegnu1.blog.51cto.com/8991766/1540728>

数据包分析

[http://blog.sina.com.cn/s/blog\\_c3098fa50102uxn0.html](http://blog.sina.com.cn/s/blog_c3098fa50102uxn0.html)

<https://wenku.baidu.com/view/9d8751fb700abb68a982fb88.html>

<https://wenku.baidu.com/view/ae29d033eefdc8d376ee3217.html>

网络包:

URG 紧急指针 (urgent pointer) 有效。

ACK 确认序号有效。

PSH 接收方应该尽快将这个报文段交给应用层。

RST 重建连接。

SYN 同步序号用来发起一个连接。

FIN 发端完成发送任务。

一个HTTP请求的包序列,

抓包的过滤器是 “tcp port http”

(否则还会有3个握手包:

1. client (port:53449) --[SYN]-->Server (port:80)
2. Server (port:80) --[SYN,ACK]-->client (port:53449)
3. client (port:53449) --[ACK]-->Server (port:80)

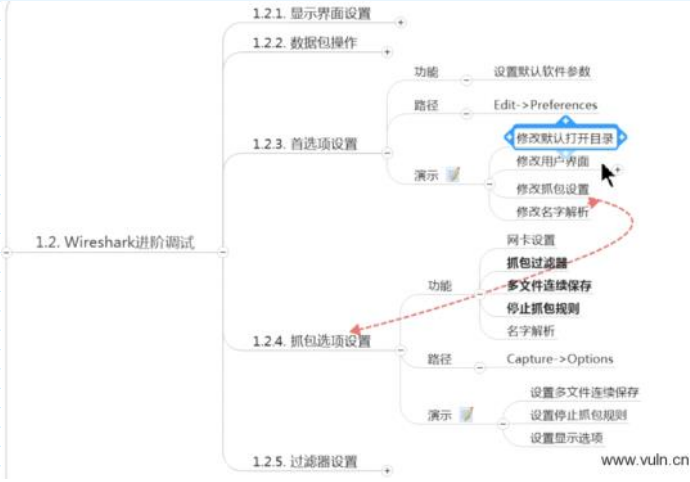
<http://www.vuln.cn/2103>

[http://www.cr173.com/html/20128\\_all.html](http://www.cr173.com/html/20128_all.html)

<http://www.cnblogs.com/imwtr/articles/4356016.html>

<http://www.cnblogs.com/TankXiao/archive/2012/10/10/2711777.html#savefilter>

[http://blog.csdn.net/liu\\_vanzhao/article/details/6593478](http://blog.csdn.net/liu_vanzhao/article/details/6593478)  
[http://www.vuln.cn/?utm\\_source=itdadao&utm\\_medium=referral](http://www.vuln.cn/?utm_source=itdadao&utm_medium=referral)



ip.dst==10.202.78.11&&http

来自 <[http://blog.sina.com.cn/s/blog\\_c3098fa50102uxn0.html](http://blog.sina.com.cn/s/blog_c3098fa50102uxn0.html)>

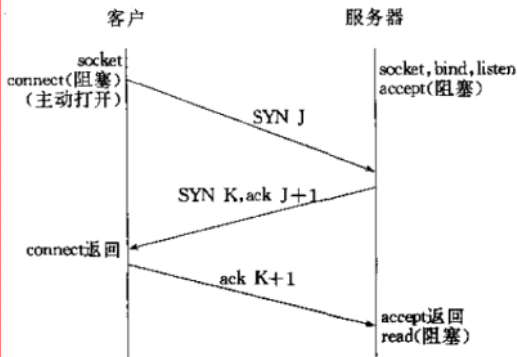


图 2.2 TCP 的三路握手

ACK==对方Seq+1

11.0.75413	18.68.226.7	13.2.15.60	TCP	HTTP 2.255 [FIN, ACK] Seq=1431, Win=1460, Len=0
12.0.75413	13.2.15.60	18.68.226.7	TCP	2773 -> 2872 [ACK] Seq=1431, Win=0, Len=0
13.0.75413	13.2.15.60	18.68.226.7	HTTP	GET /c.php?u=186e4d0c005c534c220&u=020e5c2f8202002223

1. client (port:53449) -->Server (port:80) 是基于应用级协议HTTP传送的
2. Server (port:80) -->client (port:53449) 是TCP传送的
3. 由于页面比较大, 故Server端的页面会分解成多个包传过去。  
而client端收到了会返回ACK, 其中NO11中的ACK 的Ack: 13140; 13140刚好是No10的[Next sequence number]和No12的[sequence number]
4. 所有的页面片段传送完, HTTP协议会组装起来, 弄成独立的包, 即No37. No37的sequence number是42460

No.	Seq	Source	Destination	Protocol	Len
1	0.000000	18.68.226.7	13.2.15.60	TCP	60
2	0.000000	13.2.15.60	18.68.226.7	TCP	60
3	0.000000	13.2.15.60	18.68.226.7	TCP	60
4	0.000000	13.2.15.60	18.68.226.7	TCP	60
5	0.000000	13.2.15.60	18.68.226.7	TCP	60
6	0.000000	13.2.15.60	18.68.226.7	TCP	60
7	0.000000	13.2.15.60	18.68.226.7	TCP	60
8	0.000000	13.2.15.60	18.68.226.7	TCP	60
9	0.000000	13.2.15.60	18.68.226.7	TCP	60
10	0.000000	13.2.15.60	18.68.226.7	TCP	60
11	0.000000	13.2.15.60	18.68.226.7	TCP	60
12	0.000000	13.2.15.60	18.68.226.7	TCP	60
13	0.000000	13.2.15.60	18.68.226.7	TCP	60
14	0.000000	13.2.15.60	18.68.226.7	TCP	60
15	0.000000	13.2.15.60	18.68.226.7	TCP	60
16	0.000000	13.2.15.60	18.68.226.7	TCP	60
17	0.000000	13.2.15.60	18.68.226.7	TCP	60
18	0.000000	13.2.15.60	18.68.226.7	TCP	60
19	0.000000	13.2.15.60	18.68.226.7	TCP	60
20	0.000000	13.2.15.60	18.68.226.7	TCP	60
21	0.000000	13.2.15.60	18.68.226.7	TCP	60
22	0.000000	13.2.15.60	18.68.226.7	TCP	60
23	0.000000	13.2.15.60	18.68.226.7	TCP	60
24	0.000000	13.2.15.60	18.68.226.7	TCP	60
25	0.000000	13.2.15.60	18.68.226.7	TCP	60
26	0.000000	13.2.15.60	18.68.226.7	TCP	60
27	0.000000	13.2.15.60	18.68.226.7	TCP	60
28	0.000000	13.2.15.60	18.68.226.7	TCP	60
29	0.000000	13.2.15.60	18.68.226.7	TCP	60
30	0.000000	13.2.15.60	18.68.226.7	TCP	60
31	0.000000	13.2.15.60	18.68.226.7	TCP	60
32	0.000000	13.2.15.60	18.68.226.7	TCP	60
33	0.000000	13.2.15.60	18.68.226.7	TCP	60
34	0.000000	13.2.15.60	18.68.226.7	TCP	60
35	0.000000	13.2.15.60	18.68.226.7	TCP	60
36	0.000000	13.2.15.60	18.68.226.7	TCP	60
37	0.000000	13.2.15.60	18.68.226.7	TCP	60
38	0.000000	13.2.15.60	18.68.226.7	TCP	60
39	0.000000	13.2.15.60	18.68.226.7	TCP	60
40	0.000000	13.2.15.60	18.68.226.7	TCP	60
41	0.000000	13.2.15.60	18.68.226.7	TCP	60
42	0.000000	13.2.15.60	18.68.226.7	TCP	60
43	0.000000	13.2.15.60	18.68.226.7	TCP	60
44	0.000000	13.2.15.60	18.68.226.7	TCP	60
45	0.000000	13.2.15.60	18.68.226.7	TCP	60
46	0.000000	13.2.15.60	18.68.226.7	TCP	60
47	0.000000	13.2.15.60	18.68.226.7	TCP	60
48	0.000000	13.2.15.60	18.68.226.7	TCP	60
49	0.000000	13.2.15.60	18.68.226.7	TCP	60
50	0.000000	13.2.15.60	18.68.226.7	TCP	60
51	0.000000	13.2.15.60	18.68.226.7	TCP	60
52	0.000000	13.2.15.60	18.68.226.7	TCP	60
53	0.000000	13.2.15.60	18.68.226.7	TCP	60
54	0.000000	13.2.15.60	18.68.226.7	TCP	60
55	0.000000	13.2.15.60	18.68.226.7	TCP	60
56	0.000000	13.2.15.60	18.68.226.7	TCP	60
57	0.000000	13.2.15.60	18.68.226.7	TCP	60
58	0.000000	13.2.15.60	18.68.226.7	TCP	60
59	0.000000	13.2.15.60	18.68.226.7	TCP	60
60	0.000000	13.2.15.60	18.68.226.7	TCP	60
61	0.000000	13.2.15.60	18.68.226.7	TCP	60
62	0.000000	13.2.15.60	18.68.226.7	TCP	60
63	0.000000	13.2.15.60	18.68.226.7	TCP	60
64	0.000000	13.2.15.60	18.68.226.7	TCP	60
65	0.000000	13.2.15.60	18.68.226.7	TCP	60
66	0.000000	13.2.15.60	18.68.226.7	TCP	60
67	0.000000	13.2.15.60	18.68.226.7	TCP	60
68	0.000000	13.2.15.60	18.68.226.7	TCP	60
69	0.000000	13.2.15.60	18.68.226.7	TCP	60
70	0.000000	13.2.15.60	18.68.226.7	TCP	60
71	0.000000	13.2.15.60	18.68.226.7	TCP	60
72	0.000000	13.2.15.60	18.68.226.7	TCP	60
73	0.000000	13.2.15.60	18.68.226.7	TCP	60
74	0.000000	13.2.15.60	18.68.226.7	TCP	60
75	0.000000	13.2.15.60	18.68.226.7	TCP	60
76	0.000000	13.2.15.60	18.68.226.7	TCP	60
77	0.000000	13.2.15.60	18.68.226.7	TCP	60
78	0.000000	13.2.15.60	18.68.226.7	TCP	60
79	0.000000	13.2.15.60	18.68.226.7	TCP	60
80	0.000000	13.2.15.60	18.68.226.7	TCP	60
81	0.000000	13.2.15.60	18.68.226.7	TCP	60
82	0.000000	13.2.15.60	18.68.226.7	TCP	60
83	0.000000	13.2.15.60	18.68.226.7	TCP	60
84	0.000000	13.2.15.60	18.68.226.7	TCP	60
85	0.000000	13.2.15.60	18.68.226.7	TCP	60
86	0.000000	13.2.15.60	18.68.226.7	TCP	60
87	0.000000	13.2.15.60	18.68.226.7	TCP	60
88	0.000000	13.2.15.60	18.68.226.7	TCP	60
89	0.000000	13.2.15.60	18.68.226.7	TCP	60
90	0.000000	13.2.15.60	18.68.226.7	TCP	60
91	0.000000	13.2.15.60	18.68.226.7	TCP	60
92	0.000000	13.2.15.60	18.68.226.7	TCP	60
93	0.000000	13.2.15.60	18.68.226.7	TCP	60
94	0.000000	13.2.15.60	18.68.226.7	TCP	60
95	0.000000	13.2.15.60	18.68.226.7	TCP	60
96	0.000000	13.2.15.60	18.68.226.7	TCP	60
97	0.000000	13.2.15.60	18.68.226.7	TCP	60
98	0.000000	13.2.15.60	18.68.226.7	TCP	60
99	0.000000	13.2.15.60	18.68.226.7	TCP	60
100	0.000000	13.2.15.60	18.68.226.7	TCP	60

TCP

对方发送数据, 套接口变为可读且read返回大于0

对方发送FIN (对方进程终止), 套接口就变成可读且read返回0 (文件结束)

对方发送RST (对方主机崩溃并重启), 套接口就变为可读且返回-1, errno则含有明确的错误码

我们对返回为0的关闭, 对返回-1的关闭且视其情况是否记日志

close终止了数据传送的方向是读和写。由于TCP连接时全双工的, 有很多时候我们要通知另一端我们已完成了数据发送, 那一端仍有很多数据要发送也是如此。

来自 <[http://blog.sina.com.cn/s/blog\\_c3098fa50102uxn0.html](http://blog.sina.com.cn/s/blog_c3098fa50102uxn0.html)>

1. 过滤IP, 如来源IP或者目标IP等于某个IP

例子:

ip.src eq 192.168.1.107 or ip.dst eq 192.168.1.107

或者

ip.addr eq 192.168.1.107 // 都能显示来源IP和目标IP

来自 <<http://www.cnblogs.com/imwtr/articles/4356016.html>>

来自 <<http://www.cnblogs.com/imwtr/articles/4356016.html>>

来自 <<http://www.cnblogs.com/imwtr/articles/4356016.html>>

捕捉过滤器：用于决定将什么样的信息记录在捕捉结果中。需要在开始捕捉前设置。

显示过滤器：在捕捉结果中进行详细查找。他们可以在得到捕捉结果后随意修改。

捕捉过滤器是数据经过的第一层过滤器，它用于控制捕捉数据的数量，以避免产生过大的日志文件。

显示过滤器是一种更为强大（复杂）的过滤器。它允许您在日志文件中迅速准确地找到所需要的记录。

两种过滤器使用的语法是完全不同的。