抓包神器 tcpdump

初识 tcpdump

如果你抓过 TCP 的包，你一定听说过图形化界面软件 wireshark，tcpdump 则是一个命令行的网络流量分析工具，功能非常强大。尤其是做后台开发的同学要在服务器上定位一些黑盒的应用，tcpdump 是唯一的选择。

大部分 Linux 发行包都预装了 tcpdump，如果没有预装，可以用对应操作系统的包管理命令安装，比如在 Centos 下，可以用 yum install -y tcpdump 来进行安装。

tcpdump 基础

在命令行里直接输入如下的命令，不出意外，会出现大量的输出

```
tcpdump -i any

07:02:12.195611 IP test.ya.local.59915 > c2.shared.ssh:
Flags [.], ack 1520940, win 2037, options [nop,nop,TS val 1193378555 ecr 428247729], length 0
07:02:12.195629 IP c2.shared.ssh > test.ya.local.59915:
Flags [P.], seq 1520940:1521152, ack 1009, win 315, options [nop,nop,TS val 428247729 ecr 1193378555], length 212
07:02:12.195677 IP test.ya.local.59915 > c2.shared.ssh:
Flags [.], ack 1521152, win 2044, options [nop,nop,TS val 1193378555 ecr 428247729], length 0
07:02:12.195730 IP c2.shared.ssh > test.ya.local.59915:
Flags [P.], seq 1521152:1521508, ack 1009, win 315, options [nop,nop,TS val 428247730 ecr 1193378555], length 356
```

指定网卡

`-i`表示指定哪一个网卡，any 表示任意。有哪些网卡可以用 ifconfig 来查看，在我的虚拟机上，ifconfig 输出结果如下。如果只想查看 eth0 网卡经过的数据包，就可以使用`tcpdump -i eth0`来指定。

```
ya@c2 ~$ ifconfig
eth0: flags=4163<UP,BROADCAST,RUNNING,MULTICAST>  mtu 1500
        inet 10.211.55.10  netmask 255.255.255.0  broadcast 10.211.55.255
        inet6 fdb2:2c26:f4e4:0:21c:42ff:febe:259c  prefixlen 64  scopeid 0x0<global>
        inet6 fe80::21c:42ff:febe:259c  prefixlen 64  scopeid 0x20<link>
        ether 00:1c:42:be:25:9c  txqueuelen 1000  (Ethernet)
        RX packets 163260  bytes 14754099 (14.0 MiB)
        RX errors 0  dropped 0  overruns 0  frame 0
        TX packets 138175  bytes 24488277 (23.3 MiB)
        TX errors 0  dropped 0 overruns 0  carrier 0  collisions 0

lo: flags=73<UP,LOOPBACK,RUNNING>  mtu 65536
        inet 127.0.0.1  netmask 255.0.0.0
        inet6 ::1  prefixlen 128  scopeid 0x10<host>
        loop  txqueuelen 1  (Local Loopback)
        RX packets 9500  bytes 842957 (823.2 KiB)
        RX errors 0  dropped 0  overruns 0  frame 0
        TX packets 9500  bytes 842957 (823.2 KiB)
        TX errors 0  dropped 0 overruns 0  carrier 0  collisions 0
```

过滤主机：host 选项

如果只想查看 ip 为 10.211.55.2 的网络包，这个 ip 可以是源地址也可以是目标地址

```
sudo tcpdump -i any host 10.211.55.2
```

过滤源地址、目标地址：src、dst

如果只想抓取主机 10.211.55.10 发出的包

```
sudo tcpdump -i any src 10.211.55.10
```

如果只想抓取主机 10.211.55.10 收到的包

```
sudo tcpdump -i any dst 10.211.55.1
```

过滤端口：port 选项

抓取某端口的数据包：port 选项比如查看 80 端通信的数据包

```
sudo tcpdump -i any port 80
```

如果只想抓取 80 端口**收到**的包，可以加上 dst

```
sudo tcpdump -i any dst port 80
```

过滤指定端口范围内的流量

比如抓取 21 到 23 区间所有端口的流量

```
tcpdump portrange 21-23
```

禁用主机与端口解析：-n 与 -nn 选项

如果不加-n选项，tcpdump 会显示主机名，比如下面的test.ya.local和c2.shared

```
09:04:56.821206 IP test.ya.local.59915 > c2.shared.ssh:
Flags [P.], seq 397:433, ack 579276, win 2048, options [nop,nop,TS val 1200089877 ecr 435612355], length 36
```

禁用主机与端口解析：-n 与 -nn 选项

加上-n选项以后，可以看到主机名都已经被替换成了 ip

```
sudo tcpdump -i any  -n
10:02:13.705656 IP 10.211.55.2.59915 > 10.211.55.10.ssh:
Flags [P.], seq 829:865, ack 1228756, win 2048, options [nop,nop,TS val 1203228910 ecr 439049239], length 36
```

-nn 选项

但是常用端口还是会被转换成协议名，比如 ssh 协议的 22 端口。如果不想 tcpdump 做转换，可以加上 -nn，这样就不会解析端口了，输出中的 ssh 变为了 22

```
sudo tcpdump -i any  -nn

10:07:37.598725 IP 10.211.55.2.59915 > 10.211.55.10.22:
Flags [P.], seq 685:721, ack 1006224, win 2048, options [nop,nop,TS val 1203524536 ecr 439373132], length 36
```

过滤协议

如果只想查看 udp 协议，可以直接使用下面的命令

```
sudo tcpdump -i any -nn udp

10:25:31.457517 IP 10.211.55.10.51516 > 10.211.55.1.53: 23956+ A? www.baidu.com. (31)
10:25:31.490843 IP 10.211.55.1.53 > 10.211.55.10.51516: 23956 3/13/9 CNAME www.a.shifen.com., A 14.215.177.38, A 14.215.177.39 (506)
```

用 ASCII 格式查看包体内容：-A 选项

使用 -A 可以用 ASCII 打印报文内容，比如常用的 HTTP 协议传输 json 、html 文件等都可以用这个选项

```
sudo tcpdump -i any -nn port 80 -A

11:04:25.793298 IP 183.57.82.231.80 > 10.211.55.10.40842: Flags [P.], seq 1:1461, ack 151, win 16384, length 1460
HTTP/1.1 200 OK
Server: Tengine
Content-Type: application/javascript
Content-Length: 63522
Connection: keep-alive
Vary: Accept-Encoding
Date: Wed, 13 Mar 2019 11:49:35 GMT
Expires: Mon, 02 Mar 2020 11:49:35 GMT
Last-Modified: Tue, 05 Mar 2019 23:30:55 GMT
ETag: W/"5c7f06af-f822"
Cache-Control: public, max-age=30672000
Access-Control-Allow-Origin: *
Served-In-Seconds: 0.002
```

-X 选项

与 -A 对应的还有一个 -X 命令，用来同时用 HEX 和 ASCII 显示报文内容。

```
sudo tcpdump -i any -nn port 80 -X

11:33:53.945089 IP 36.158.217.225.80 > 10.211.55.10.45436: Flags [P.], seq 1:1461, ack 151, win 16384, length 1460
    0x0000:  4500 05dc b1c4 0000 8006 42fb 249e d9e1  E.........B.$...
    0x0010:  0ad3 370a 0050 b17c 3b79 032b 8ffb cf66  ..7..P.|;y.+...f
    0x0020:  5018 4000 9e9e 0000 4854 5450 2f31 2e31  P.@.....HTTP/1.1
    0x0030:  2032 3030 204f 4b0d 0a53 6572 7665 723a  .200.OK..Server:
    0x0040:  2054 656e 6769 6e65 0d0a 436f 6e74 656e  .Tengine..Conten
    0x0050:  742d 5479 7065 3a20 6170 706c 6963 6174  t-Type:.applicat
    0x0060:  696f 6e2f 6a61 7661 7363 7269 7074 0d0a  ion/javascript..
    0x0070:  436f 6e74 656e 742d 4c65 6e67 7468 3a20  Content-Length:.
    0x0080:  3633 3532 320d 0a43 6f6e 6e65 6374 696f  63522..Connectio
    0x0090:  6e3a 206b 6565 702d 616c 6976 650d 0a56  n:.keep-alive..V
    0x00a0:  6172 793a 2041 6363 6570 742d 456e 636f  ary:.Accept-Enco
    0x00b0:  6469 6e67 0d0a 4461 7465 3a20 5765 642c  ding..Date:.Wed,
    0x00c0:  2031 3320 4d61 7220 3230 3139 2031 313a  .13.Mar.2019.11:
    0x00d0:  3439 3a33 3520 474d 540d 0a45 7870 6972  49:35.GMT..Expir
```

限制包大小：-s 选项

当包体很大，可以用 -s 选项截取部分报文内容，一般都跟 -A 一起使用。查看每个包体前 500 字节可以用下面的命令

```
sudo tcpdump -i any -nn port 80 -A -s 500
```

如果想显示包体所有内容，可以加上-s 0

只抓取 5 个报文： -c 选项

使用 -c number命令可以抓取 number 个报文后退出。在网络包交互非常频繁的服务器上抓包比较有用，可能运维人员只想抓取1000 个包来分析一些网络问题，就比较有用了。

```
sudo tcpdump -i any -nn port 80  -c 5
```

数据报文输出到文件：-w 选项

-w 选项用来把数据报文输出到文件，比如下面的命令就是把所有 80 端口的数据输出到文件

```
sudo tcpdump -i any port 80 -w test.pcap
```

生成的 pcap 文件就可以用 wireshark 打开进行更详细的分析了

也可以加上-U强制立即写到本地磁盘，性能稍差

显示绝对的序号：-S 选项

默认情况下，tcpdump显示的是从 0 开始的相对序号。如果想查看真正的绝对序号，可以用 -S 选项。

没有 -S 时的输出，seq 和 ACK 都是从 0 开始

```
sudo tcpdump -i any port 80 -nn

12:12:37.832165 IP 10.211.55.10.46102 > 36.158.217.230.80:
Flags [P.], seq 1:151, ack 1, win 229, length 150
12:12:37.832272 IP 36.158.217.230.80 > 10.211.55.10.46102:
Flags [.], ack 151, win 16384, length 0
```

有 -S 时的输出，可以看到 seq 不是从 0 开始

```
sudo tcpdump -i any port 80 -nn -S

12:13:21.863918 IP 10.211.55.10.46074 > 36.158.217.223.80:
Flags [P.], seq 4277123624:4277123774, ack 3358116659, win 229, length 150
12:13:21.864091 IP 36.158.217.223.80 > 10.211.55.10.46074:
Flags [.], ack 4277123774, win 16384, length 0
```

高级技巧

tcpdump 真正强大的是可以用布尔运算符and（或&&）、or（或||）、not（或!）来组合出任意复杂的过滤器

抓取 ip 为 10.211.55.10 到端口 3306 的数据包

```
sudo tcpdump -i any host 10.211.55.10 and dst port 3306
```

抓取源 ip 为 10.211.55.10，目标端口除了22 以外所有的流量

```
sudo tcpdump -i any src 10.211.55.10 and not dst port 22
```

复杂的分组

如果要抓取：来源 ip 为 10.211.55.10 且目标端口为 3306 或 6379 的包，按照前面的描述，我们会写出下面的语句

```
sudo tcpdump -i any src 10.211.55.10 and (dst port 3306 or 6379)
```

如果运行一下，就会发现执行报错了，因为包含了特殊字符( )，解决的办法是用单引号把复杂的组合条件包起来。

```
sudo tcpdump -i any 'src 10.211.55.10 and (dst port 3306 or 6379)'
```

如果想显示所有的 RST 包，要如何来写 tcpdump 的语句呢

```
tcpdump 'tcp[13] & 4 != 0'
```

要弄懂这个语句，必须要清楚 TCP 首部中 offset 为 13 的字节的第 3 比特位就是 RST

TCP Header

Offsets Octet		0								1								2								3							
Octet	Bit	0	1	2	3	4	5	6	7	8	9	10	11	12	13	14	15	16	17	18	19	20	21	22	23	24	25	26	27	28	29	30	31
0	0	Source port																Destination port														
4	32	Sequence number																														
8	64	Acknowledgment number (if ACK set)																														
12	96	Data offset				Reserved 0 0 0				NS	CWR	ECE	URG	ACK	PSH	RST	SYN	FIN	Window Size													
16	128	Checksum																Urgent pointer (if URG set)														
20 ...	160 ...	Options (if *data offset* > 5. Padded at the end with "0" bytes if necessary.) ...																														

tcp[13] 表示 tcp 头部中偏移量为 13 字节，如上图中红色框的部分，

!=0 表示当前 bit 置 1，即存在此标记位，跟 4 做与运算是因为 RST 在 TCP 的标记位的位置在第 3 位(00000100)

如果想过滤 SYN + ACK 包，那就是 SYN 和 ACK 包同时置位（00010010），写成 tcpdump 语句就是

```
tcpdump 'tcp[13] & 18 != 0'
```

TCPDump 输出解读

```
1 16:46:22.722865 IP 10.211.55.5.45424 > 10.211.55.10.8080:
Flags [S], seq 3782956689, win 29200, options [mss 1460,sackOK,TS val 463670960 ecr 0,nop,wscale 7], length 0

2 16:46:22.722903 IP 10.211.55.10.8080 > 10.211.55.5.45424:
Flags [S.], seq 3722022028, ack 3782956690, win 28960, options [mss 1460,sackOK,TS val 463298257 ecr 463670960,nop,wscale 7], length 0

3 16:46:22.723068 IP 10.211.55.5.45424 > 10.211.55.10.8080:
Flags [.], ack 1, win 229, options [nop,nop,TS val 463670960 ecr 463298257], length 0

4 16:46:25.947217 IP 10.211.55.5.45424 > 10.211.55.10.8080:
Flags [P.], seq 1:13, ack 1, win 229, options [nop,nop,TS val 463674184 ecr 463298257], length 12
hello world

5 16:46:25.947261 IP 10.211.55.10.8080 > 10.211.55.5.45424:
Flags [.], ack 13, win 227, options [nop,nop,TS val 463301481 ecr 463674184], length 0

6 16:46:28.011057 IP 10.211.55.5.45424 > 10.211.55.10.8080:
Flags [F.], seq 13, ack 1, win 229, options [nop,nop,TS val 463676248 ecr 463301481], length 0

7 16:46:28.011153 IP 10.211.55.10.8080 > 10.211.55.5.45424:
Flags [F.], seq 1, ack 14, win 227, options [nop,nop,TS val 463303545 ecr 463676248], length 0

8 16:46:28.011263 IP 10.211.55.5.45424 > 10.211.55.10.8080:
Flags [.], ack 2, win 229, options [nop,nop,TS val 463676248 ecr 463303545], length 0
```