A+

iptables详解（6）：iptables扩展匹配条件之’–tcp-flags’

在本博客中，从理论到实践，系统的介绍了iptables，如果你想要从头开始了解iptables，可以查看iptables文章列表，直达链接如下

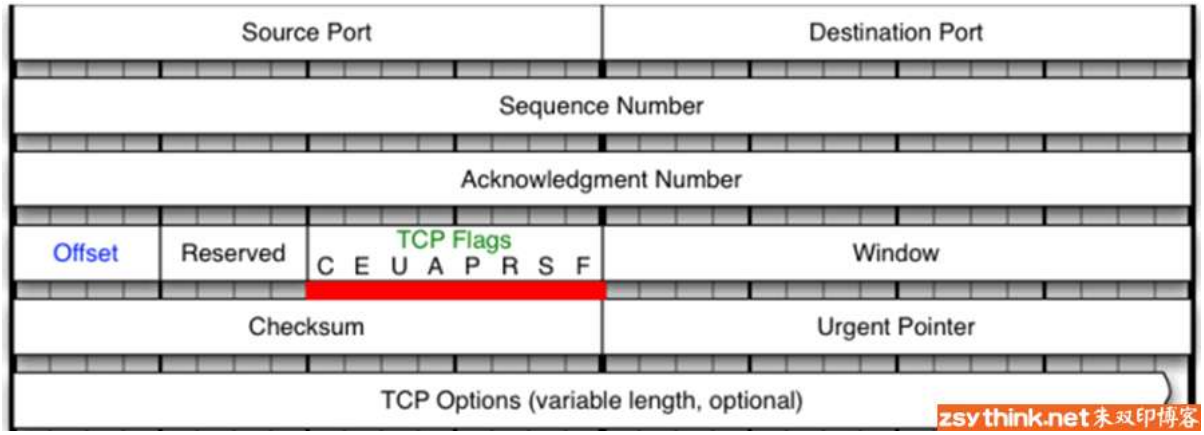
iptables零基础快速入门系列

如果你看过前文，那么你一定知道，前文已经对"tcp扩展模块"做过总结，但是只总结了tcp扩展模块中的"--sport"与"--dport"选项，并没有总结"--tcp-flags"选项
我们就来认识一下tcp扩展模块中的"--tcp-flags"。

注：阅读这篇文章之前，需要对tcp协议的基础知识有一定的了解，比如：tcp头的结构、tcp三次握手的过程。

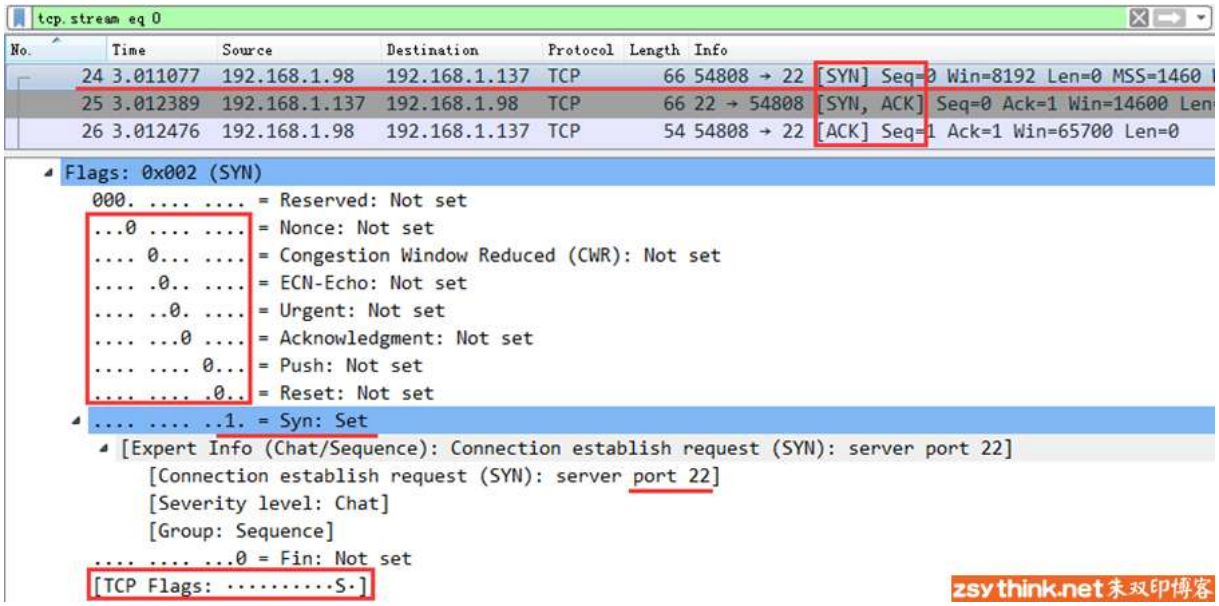
见名知义，"--tcp-flags"指的就是tcp头中的标志位，看来，在使用iptables时，我们可以通过此扩展匹配条件，去匹配tcp报文的头部的标识位，然后根据标识位的
访问控制的功能。

既然说到了tcp头中的标志位，那么我们就来回顾一下tcp头的结构，如下图所示。



在使用iptables时，使用tcp扩展模块的"--tcp-flags"选项，即可对上图中的标志位进行匹配，判断指定的标志位的值是否为"1"，而tcp header的结构不是我们今天
我们继续聊tcp的标识位，在tcp协议建立连接的过程中，需要先进行三次握手，而三次握手就要依靠tcp头中的标志位进行。

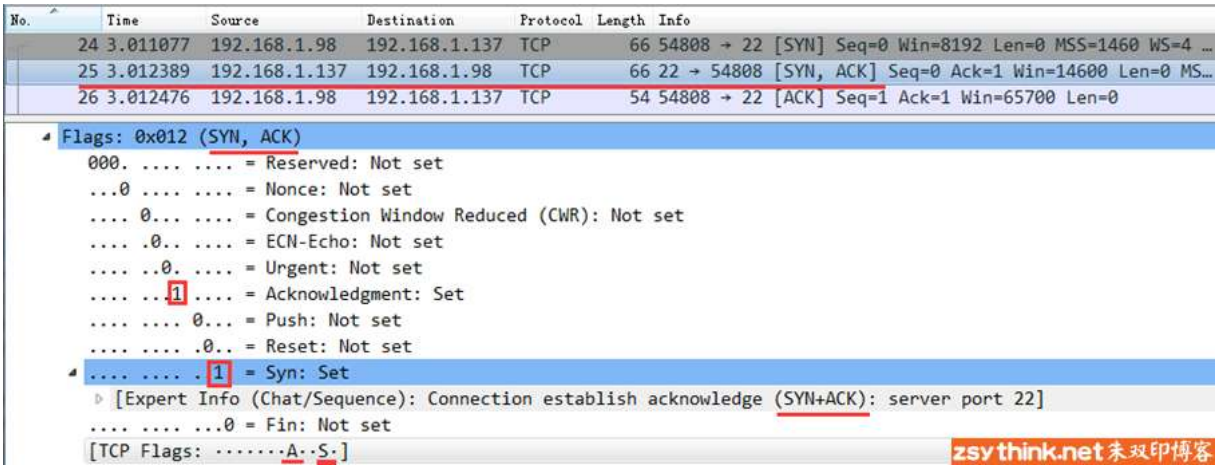
为了更加具象化的描述这个过程，我们可以抓包查看ssh建立连接的过程，如下图所示（使用wireshark在ssh客户端抓包，跟踪对应的tcp流）：



上图为tcp三次握手中的第一次握手，客户端（IP为98）使用本地的随机端口54808向服务端（IP为137）发起连接请求，tcp头的标志位中，只有SYN位被标识为1
均为0。

在上图的下方可以看到"[TCP Flags: ··········S·]"，其中的"S"就表示SYN位，整体表示只有SYN位为1。

上图为tcp三次握手中第一次握手的tcp头中的标志位，下图是第二次握手的，服务端回应刚才的请求，将自己的tcp头的SYN标志位也设置为1，同时将ACK标志位
下图所示。



上图中的下方显示的标志位列表也变成了，[TCP Flags: ·····A··S·]，表示只有ACK标志位与SYN标志位为1，如上图所示，第三次握手我就不再截图了，说到这里，我出我们今天要说的话题了，就是"--tcp-flags"选项，假设，我现在想要匹配到上文中提到的"第一次握手"的报文，则可以使用如下命令：

```
#iptables -t filter -I INPUT -p tcp -m tcp --dport 22 --tcp-flags SYN,ACK,FIN,RST,URG,PSH SYN -j REJECT
#
```

上图中，"-m tcp --dport 22"的含义在前文中已经总结过，表示使用tcp扩展模块，指定目标端口为22号端口(ssh默认端口)，"--tcp-flags"就是我们今天要讨论的扩件，用于匹配报文tcp头部的标志位，"SYN,ACK,FIN,RST,URG,PSH SYN"是什么意思呢？这串字符就是用于配置我们要匹配的标志位的，我们可以把这串字符拆成解，第一部分为"SYN,ACK,FIN,RST,URG,PSH"，第二部分为"SYN"。

第一部分表示：我们需要匹配报文tcp头中的哪些标志位，那么上例的配置表示，我们需要匹配报文tcp头中的6个标志位，这6个标志位分别为为"SYN、ACK、FIN、G、PSH"，我们可以把这一部分理解成需要匹配的标志位列表。

第二部分表示：第一部分的标志位列表中，哪些标志位必须为1，上例中，第二部分为SYN，则表示，第一部分需要匹配的标志位列表中，SYN标志位的值必须为1，须为0。

所以，上例中的"SYN,ACK,FIN,RST,URG,PSH SYN"表示，需要匹配报文tcp头中的"SYN、ACK、FIN、RST、URG、PSH"这些标志位，其中SYN标志位必须为1，志位必须为0，这与上文中wireshark抓包时的情况相同，正是tcp三次握手时第一次握手时的情况，上文中第一次握手的报文的tcp头中的标志位如下：

```
[TCP Flags: ··········S·]
```

其实，--tcp-flags的表示方法与wireshark的表示方法有异曲同工之妙，只不过，wireshark中，标志位为0的用"点"表示，标志位为1的用对应字母表示，在--tcp-fl先指明需要匹配哪些标志位，然后再指明这些标志位中，哪些必须为1，剩余的都必须为0。

那么，聪明如你一定想到了，如果我想要匹配tcp头中的第二次握手时的标志位的情况，该怎么表示呢？

示例如下（此处省略对源地址与目标地址的匹配，重点在于对tcp-flags的示例）

```
#iptables -t filter -I INPUT -p tcp -m tcp --dport 22 --tcp-flags SYN,ACK,FIN,RST,URG,PSH SYN -j REJECT
#iptables -t filter -I INPUT -p tcp -m tcp --dport 22 --tcp-flags SYN,ACK,FIN,RST,URG,PSH SYN,ACK -j REJECT
#
```

上图中，第一条命令匹配到的报文是第一次握手的报文，第二条命令匹配到的报文是第二次握手的报文。

综上所述，只要我们能够灵活的配置上例中的标志位，即可匹配到更多的应用场景中。

其实，上例中的两条命令还可以简写为如下模样

```
#iptables -t filter -I INPUT -p tcp -m tcp --dport 22 --tcp-flags ALL SYN -j REJECT
#iptables -t filter -I INPUT -p tcp -m tcp --dport 22 --tcp-flags ALL SYN,ACK -j REJECT
#
```

没错，我们可以用ALL表示"SYN,ACK,FIN,RST,URG,PSH"。

其实，tcp扩展模块还为我们专门提供了一个选项，可以匹配上文中提到的"第一次握手"，那就是--syn选项

使用"--syn"选项相当于使用"--tcp-flags SYN,RST,ACK,FIN SYN"，也就是说，可以使用"--syn"选项去匹配tcp新建连接的请求报文。

示例如下：

```
#iptables -t filter -I INPUT -p tcp -m tcp --dport 22 --syn -j REJECT
#
```

小结

结合之前的文章，我们把tcp模块的常用扩展匹配条件再总结一遍，方便以后回顾。

tcp扩展模块常用的扩展匹配条件如下：

--sport

用于匹配tcp协议报文的源端口，可以使用冒号指定一个连续的端口范围

```
#示例
iptables -t filter -I OUTPUT -d 192.168.1.146 -p tcp -m tcp --sport 22 -j REJECT
iptables -t filter -I OUTPUT -d 192.168.1.146 -p tcp -m tcp --sport 22:25 -j REJECT
iptables -t filter -I OUTPUT -d 192.168.1.146 -p tcp -m tcp ! --sport 22 -j ACCEPT
```

--dport

用于匹配tcp协议报文的目标端口，可以使用冒号指定一个连续的端口范围

```
#示例
iptables -t filter -I INPUT -s 192.168.1.146 -p tcp -m tcp --dport 22:25 -j REJECT
iptables -t filter -I INPUT -s 192.168.1.146 -p tcp -m tcp --dport :22 -j REJECT
iptables -t filter -I INPUT -s 192.168.1.146 -p tcp -m tcp --dport 80: -j REJECT
```

--tcp-flags

用于匹配报文的tcp头的标志位

```
#示例
iptables -t filter -I INPUT -p tcp -m tcp --dport 22 --tcp-flags SYN,ACK,FIN,RST,URG,PSH SYN -j REJECT
iptables -t filter -I INPUT -p tcp -m tcp --dport 22 --tcp-flags SYN,ACK,FIN,RST,URG,PSH SYN,ACK -j REJECT
iptables -t filter -I INPUT -p tcp -m tcp --dport 22 --tcp-flags ALL SYN -j REJECT
iptables -t filter -I INPUT -p tcp -m tcp --dport 22 --tcp-flags ALL SYN,ACK -j REJECT
```

--syn

用于匹配tcp新建连接的请求报文，相当于使用"--tcp-flags SYN,RST,ACK,FIN SYN"

```
#示例
iptables -t filter -I INPUT -p tcp -m tcp --dport 22 --syn -j REJECT
```

希望这篇文章能够对你有所帮助~~~常来捧场哦，亲~~~