iptables详解（4）：iptables匹配条件总结之一

朱双印
2年前更新

$ 赞赏

111 696

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

经过前文的总结，我们已经能够熟练的管理规则了，但是我们使用过的"匹配条件"少得可怜，之前的示例中，我们只使用过一种匹配条件，就是将"源地址"作为匹配条件。

那么这篇文章中，我们就来了解一下更多的匹配条件，以及匹配条件的更多用法。

注意：在参照本文进行iptables实验时，请务必在个人的测试机上进行，因为如果iptables规则设置不当，有可能使你无法连接到远程主机中。

匹配条件的更多用法

还是从我们最常用的"源地址"说起吧，我们知道，使用-s选项作为匹配条件，可以匹配报文的源地址，但是之前的示例中，我们每次指定源地址，都只是指定单个IP，示例如下。

```
[www.zsythink.net]#iptables -t filter -F INPUT
[www.zsythink.net]#iptables -t filter -I INPUT -s 192.168.1.146 -j DROP
[www.zsythink.net]#iptables -nvL INPUT
Chain INPUT (policy ACCEPT 9 packets, 680 bytes)
 pkts bytes target     prot opt in     out     source               destination
    0     0 DROP       all  --  *      *       192.168.1.146        0.0.0.0/0
[www.zsythink.net]#
```

其实，我们也可以在指定源地址时，一次指定多个，用"逗号"隔开即可，示例如下。

```
[www.zsythink.net]#iptables -t filter -F INPUT
[www.zsythink.net]#iptables -t filter -I INPUT -s 192.168.1.111,192.168.1.112 -j DROP
[www.zsythink.net]#iptables -nvL INPUT
Chain INPUT (policy ACCEPT 10 packets, 1051 bytes)
 pkts bytes target     prot opt in     out     source               destination
    0     0 DROP       all  --  *      *       192.168.1.112        0.0.0.0/0
    0     0 DROP       all  --  *      *       192.168.1.111        0.0.0.0/0
[www.zsythink.net]#
```

可以看出，上例中，一次添加了两条规则，两条规则只是源地址对应的IP不同，注意，上例中的"逗号"两侧均不能包含空格，多个IP之间必须与逗号相连。

除了能指定具体的IP地址，还能指定某个网段，示例如下

```
[www.zsythink.net]#iptables -t filter -F INPUT
[www.zsythink.net]#iptables -t filter -I INPUT -s 10.6.0.0/16 -j DROP
[www.zsythink.net]#iptables -nvL INPUT
Chain INPUT (policy ACCEPT 8 packets, 576 bytes)
 pkts bytes target     prot opt in     out     source               destination
    0     0 DROP       all  --  *      *       10.6.0.0/16          0.0.0.0/0
[www.zsythink.net]#
```

上例表示，如果报文的源地址IP在10.6.0.0/16网段内，当报文经过INPUT链时就会被DROP掉。

其实，我们还可以对匹配条件取反，先看示例，如下。

```
[www.zsythink.net]#iptables -t filter -F INPUT
[www.zsythink.net]#iptables -t filter -A INPUT ! -s 192.168.1.146 -j ACCEPT
[www.zsythink.net]#iptables -t filter -nvL INPUT
Chain INPUT (policy ACCEPT 0 packets, 0 bytes)
 pkts bytes target     prot opt in     out     source               destination
  102  9111 ACCEPT     all  --  *      *      !192.168.1.146        0.0.0.0/0
[www.zsythink.net]#
```

上图中，使用"! -s 192.168.1.146"表示对 -s 192.168.1.146这个匹配条件取反， -s 192.168.1.146表示报文源IP地址为192.168.1.146即可满足匹配条件，使用 "!" 取反后则表示，报文源地址IP只要不为192.168.1.146即满足条件，那么，上例中规则表达的意思就是，只要发往本机的报文的源地址不是192.168.1.146，就接受报文。

此刻，你猜猜，按照上例中的配置，如果此时从146主机上向防火墙所在的主机发送ping请求，146主机能得到回应吗？（此处不考虑其他链，只考虑filter表的INPUT链）

为了给你思考的空间，我把答案写的远一点。

答案是：能，也就是说，按照上例的配置，146主机仍然能够ping通当前主机，为什么呢？我们来分析一下。

上例中，filter表的INPUT链中只有一条规则，这条规则要表达的意思就是：

只要报文的源IP不是192.168.1.146，那么就接受此报文，但是，某些小伙伴可能会误会，把上例中的规则理解成如下含义，

只要报文的源IP是192.168.1.146，那么就不接受此报文，这种理解与上述理解看似差别不大，其实完全不一样，这样理解是错误的，上述理解才是正确的。

换句话说就是，报文的源IP不是192.168.1.146时，会被接收，并不能代表，报文的源IP是192.168.1.146时，会被拒绝。

上例中，因为并没有任何一条规则指明源IP是192.168.1.146时，该执行怎样的动作，所以，当来自192.168.1.146的报文经过INPUT链时，并不能匹配上例中的规则，于是，此报文就继续匹配后面的规则，可是，上例中只有一条规则，这条规则后面没有其他可以匹配的规则，于是，此报文就会去匹配当前链的默认动作(默认策略)，而上例中，INPUT链的默认动作为ACCEPT，所以，来自146的ping报文就被接收了，如果，把上例中INPUT链的默认策略改为DROP，那么，146的报文将会被丢弃，146上的ping命令将得不到任何回应，但是如果将INPUT链的默认策略设置为DROP，当INPUT链中没有任何规则时，所有外来报文将会被丢弃，包括我们ssh远程连接。

好了，我们通过上例，不仅了解到了怎样对匹配条件取反，还加深了我们对默认策略的了解，一举两得，我们继续聊。

匹配条件：目标IP地址

除了可以通过-s选项指定源地址作为匹配条件，我们还可以使用-d选项指定"目标地址"作为匹配条件。

源地址表示报文从哪里来，目标地址表示报文要到哪里去。

除了127.0.0.1回环地址以外，当前机器有两个IP地址，IP如下。

```
[www.zsythink.net]#ifconfig | awk '/inet addr/{print $1,$2}'
inet addr:192.168.1.101
inet addr:192.168.1.156
inet addr:127.0.0.1
```

假设，我们想要拒绝146主机发来的报文，但是我们只想拒绝146向156这个IP发送报文，并不想要防止146向101这个IP发送报文，我们就可以指定目标地址作为匹配条件，示例如下。

```
[www.zsythink.net]#iptables -t filter -F INPUT
[www.zsythink.net]#iptables -t filter -I INPUT -s 192.168.1.146 -d 192.168.1.156 -j DROP
[www.zsythink.net]#iptables -nvL INPUT
Chain INPUT (policy ACCEPT 48 packets, 3904 bytes)
 pkts bytes target     prot opt in     out     source               destination
    0     0 DROP       all  --  *      *       192.168.1.146        192.168.1.156
[www.zsythink.net]#
```

上例表示只丢弃从146发往156这个IP的报文，但是146发往101这个IP的报文并不会被丢弃，如果我们不指定任何目标地址，则目标地址默认为0.0.0.0/0，同理，如果我们不指定源地址，源地址默认为0.0.0.0/0，0.0.0.0/0表示所有IP，示例如下。

```
[www.zsythink.net]#iptables -t filter -F INPUT
[www.zsythink.net]#iptables -t filter -I INPUT -d 192.168.1.101 -j DROP
[www.zsythink.net]#iptables -nvL INPUT
Chain INPUT (policy ACCEPT 11 packets, 824 bytes)
 pkts bytes target     prot opt in     out     source               destination
    0     0 DROP       all  --  *      *       0.0.0.0/0            192.168.1.101
[www.zsythink.net]#
```

上例表示，所有IP发送往101的报文都将被丢弃。

与-s选项一样，-d选项也可以使用"叹号"进行取反，也能够同时指定多个IP地址，使用"逗号"隔开即可。

但是请注意，不管是-s选项还是-d选项，取反操作与同时指定多个IP的操作不能同时使用。

需要明确的一点就是：当一条规则中有多个匹配条件时，这多个匹配条件之间，默认存在"与"的关系。

说白了就是，当一条规则中存在多个匹配条件时，报文必须同时满足这些条件，才算做被规则匹配。

就如下例所示，下图中的规则包含有两个匹配条件，源地址与目标地址，报文必须同时能被这两个条件匹配，才算作被当前规则匹配，也就是说，下例中，报文必须来自146，同时报文的目标地址必须为101，才会被如下规则匹配，两个条件必须同时满足。

```
[www.zsythink.net]#iptables -t filter -F INPUT
[www.zsythink.net]#iptables -t filter -I INPUT -s 192.168.1.146 -d 192.168.1.101 -j ACCEPT
[www.zsythink.net]#iptables -nvL INPUT
Chain INPUT (policy ACCEPT 28 packets, 2551 bytes)
 pkts bytes target     prot opt in     out     source               destination
    0     0 ACCEPT     all  --  *      *       192.168.1.146        192.168.1.101
[www.zsythink.net]#
```

我们除了能够使用-s选项和-d选项匹配源IP与目标IP以外，还能够匹配"源端口"与"目标端口"，但是我们一会儿再聊怎样匹配端口，我们先聊聊其他选项。

匹配条件：协议类型

我们可以使用-p选项，指定需要匹配的报文的协议类型。

假设，我们只想要拒绝来自146的tcp类型的请求，那么可以进行如下设置

```
[www.zsythink.net]#iptables -F INPUT
[www.zsythink.net]#iptables -I INPUT -s 192.168.1.146 -d 192.168.1.156 -p tcp -j REJECT
[www.zsythink.net]#iptables -nvL INPUT
Chain INPUT (policy ACCEPT 6 packets, 496 bytes)
 pkts bytes target     prot opt in     out     source               destination
    0     0 REJECT     tcp  --  *      *       192.168.1.146        192.168.1.156
[www.zsythink.net]#
```

上图中，防火墙拒绝了来自146的tcp报文发往156这个IP，那么我们来测试一下，我们在146上使用ssh连接101这个IP试试（ssh协议的传输层协议属于tcp协议类型）

```
[www.zsythink.net]# ssh 192.168.1.156
ssh: connect to host 192.168.1.156 port 22: Connection refused
```

如上图所示，ssh连接被拒绝了，那么我们使用ping命令试试 (ping命令使用icmp协议)，看看能不能ping通156。

```
[www.zsythink.net]# ssh 192.168.1.156
ssh: connect to host 192.168.1.156 port 22: Connection refused
[www.zsythink.net]# ping 192.168.1.156
PING 192.168.1.156 (192.168.1.156) 56(84) bytes of data.
64 bytes from 192.168.1.156: icmp_seq=1 ttl=64 time=0.627 ms
64 bytes from 192.168.1.156: icmp_seq=2 ttl=64 time=0.374 ms
64 bytes from 192.168.1.156: icmp_seq=3 ttl=64 time=0.290 ms
^C
```

可以看到，PING命令可以ping通156，证明icmp协议并没有被规则匹配到，只有tcp类型的报文被匹配到了。

那么，-p选项都支持匹配哪些协议呢？我们总结一下

centos6中，-p选项支持如下协议类型

tcp, udp, udplite, icmp, esp, ah, sctp

centos7中，-p选项支持如下协议类型

tcp, udp, udplite, icmp, icmpv6,esp, ah, sctp, mh

当不使用-p指定协议类型时，默认表示所有类型的协议都会被匹配到，与使用-p all的效果相同。

匹配条件：网卡接口

我们再来认识一个新的匹配条件，当本机有多个网卡时，我们可以使用 -i 选项去匹配报文是通过哪块网卡流入本机的。

我们先动手做个小例子，对-i选项有一个初步的了解以后，再结合理论去看。

当前主机的网卡名称为eth4，如下图

```
[www.zsythink.net]#ifconfig
eth4      Link encap:Ethernet  HWaddr 00:0C:29:B7:F4:D1
          inet addr:192.168.1.156  Bcast:192.168.1.255  Mask:255.255.255.0
```

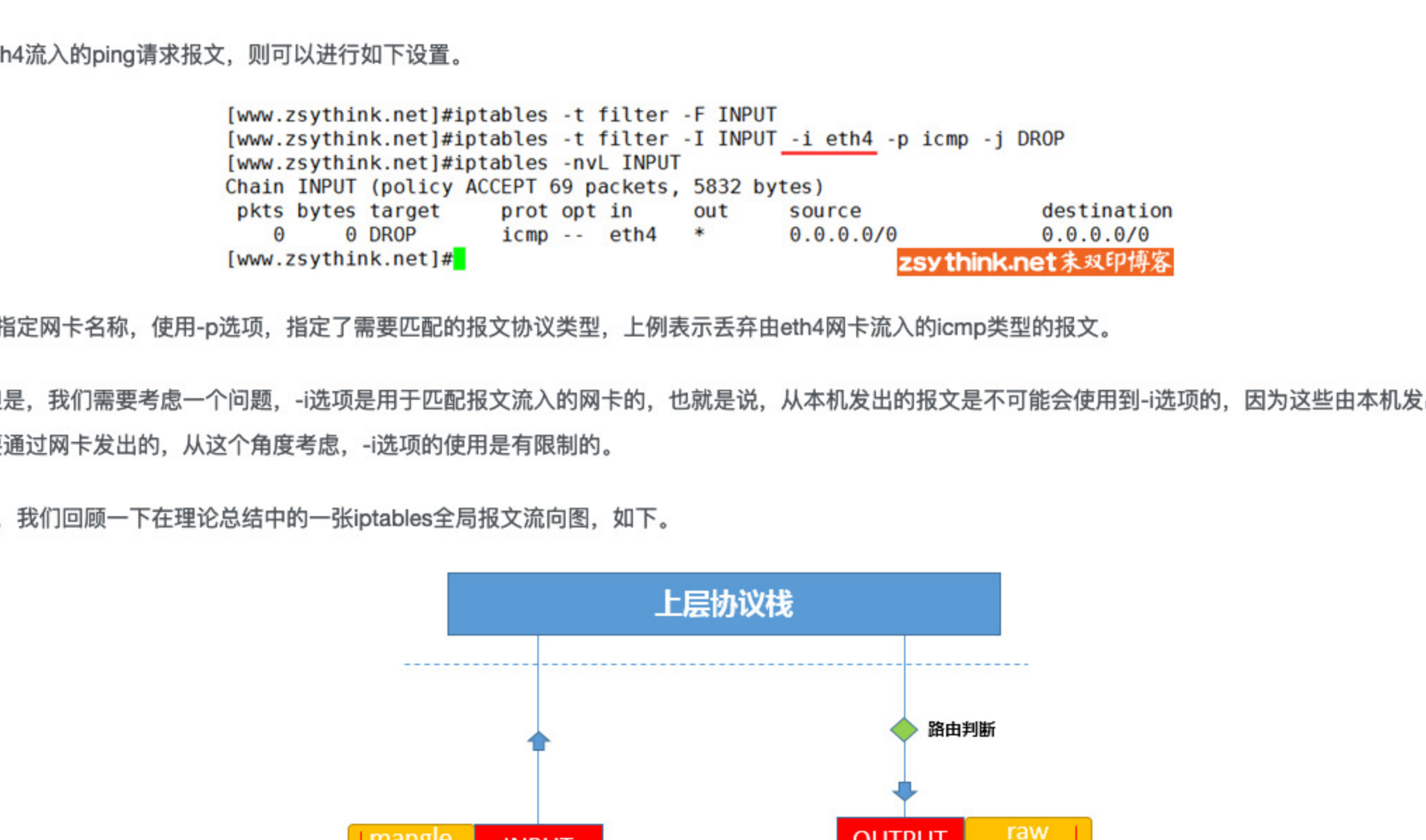
假设想要拒绝由网卡eth4流入的ping请求报文，则可以进行如下设置。

```
[www.zsythink.net]#iptables -t filter -F INPUT
[www.zsythink.net]#iptables -t filter -I INPUT -i eth4 -p icmp -j DROP
[www.zsythink.net]#iptables -nvL INPUT
Chain INPUT (policy ACCEPT 69 packets, 5832 bytes)
 pkts bytes target     prot opt in     out     source               destination
    0     0 DROP       icmp --  eth4   *       0.0.0.0/0            0.0.0.0/0
[www.zsythink.net]#
```

上图中，使用-i选项，指定网卡名称，使用-p选项，指定了需要匹配的报文协议类型，上例表示丢弃由eth4网卡流入的icmp类型的报文。

是不是很容易理解，但是，我们需要考虑一个问题，-i选项是用于匹配报文流入的网卡的，也就是说，从本机发出的报文是不可能会使用到-i选项的，因为这些由本机发出的报文压根不是从网卡流入的，而是要通过网卡发出的，从这个角度考虑，-i选项的使用是有限制的。

为了更好的解释-i选项，我们回顾一下在理论总结中的一张iptables全局报文流向图，如下。



既然-i选项是用于判断报文是从哪个网卡流入的，那么，-i选项只能用于上图中的PREROUTING链、INPUT链、FORWARD链，这是-i选项的特殊性，因为它只是用于判断报文是从哪个网卡流入的，所以只能在上图中"数据流入流向"的链中与FORWARD链中存在，而上图中的"数据发出流向"经过的链中，是不可能使用-i选项的，比如上图中的OUTPUT链与POSTROUTING链，他们都不能使用-i选项。

理解完-i选项，再来理解-o选项就好办了。

当主机有多块网卡时，可以使用-o选项，匹配报文将由哪块网卡流出，没错，-o选项与-i选项是相对的，-i选项用于匹配报文从哪个网卡流入，-o选项用于匹配报文将从哪个网卡流出。

聪明如你，一定想到了，-i选项只能用于PREROUTING链、INPUT链、FORWARD链，那么-o选项只能用于FORWARD链、OUTPUT链、POSTROUTING链。

因为-o选项是用于匹配报文将由哪个网卡"流出"的，所以与上图中的"数据进入流向"中的链没有任何缘分，所以，-o选项只能用于FORWARD链、OUTPUT链、POSTROUTING链中。

看来，FORWARD链属于"中立国"，它能同时使用-i选项与-o选项。

扩展匹配条件

好了，现在，我们就要聊聊，怎样匹配报文的"源端口"与"目标端口"。

在上文中，我们总结了"源地址"与"目标地址"以后，就顺便提到了"源端口"与"目标端口"，但是，为什么刚才不介绍"源端口"与"目标端口"，非要现在介绍呢？这是因为"源端口"与"目标端口"属于扩展匹配条件，"源地址"与"目标地址"属于基本匹配条件，上文中介绍到的匹配条件，都属于基本匹配条件，所以，我们单独把"源端口"与"目标端口"，放在后面总结，是为了引出扩展匹配条件的概念。

那么，先来了解一下，什么是扩展匹配条件。

不是基本匹配条件的就是扩展匹配条件，这样说好像是句废话，我们可以这样理解，基本匹配条件我们可以直接使用，而如果想要使用扩展匹配条件，则需要依赖一些扩展模块，或者说，在使用扩展匹配条件之前，需要指定相应的扩展模块才行，这样说不容易明白，我们做个例子，就能够明白。

我们知道，sshd服务的默认端口为22，当我们使用ssh工具远程连接主机时，默认会连接服务端的22号端口，假设，我们现在想要使用iptables设置一条规则，拒绝来自192.168.1.146的ssh请求，我们就可以拒绝146上的报文能够发往本机的22号端口，这个时候，就需要用到"目标端口"选项。

使用选项–dport可以匹配报文的目标端口，–dport意为destination-port，即表示目标端口。

注意，与之前的选项不同，–dport前有两条"横杠"，而且，使用–dport选项时，必须事先指定了使用哪种协议，即必须先使用-p选项，示例如下

```
[www.zsythink.net]#iptables -t filter -F INPUT
[www.zsythink.net]#iptables -t filter -I INPUT -s 192.168.1.146 -p tcp -m tcp --dport 22 -j REJECT
[www.zsythink.net]#iptables -vnL INPUT
Chain INPUT (policy ACCEPT 8 packets, 683 bytes)
 pkts bytes target     prot opt in     out     source               destination
    0     0 REJECT     tcp  --  *      *       192.168.1.146        0.0.0.0/0            tcp dpt:22 reject-with icmp-port-unreachable
[www.zsythink.net]#
```

上图中，我们就使用了扩展匹配条件–dport，指定了匹配报文的目标端口，如果外来报文的目标端口为本机的22号端口（ssh默认端口），则拒绝之，而在使用–dport之前，我们使用-m选项，指定了对应的扩展模块为tcp，也就是说，如果想要使用–dport这个扩展匹配条件，则必须依靠某个扩展模块完成，上例中，这个扩展模块就是tcp扩展模块，最终，我们使用的是tcp扩展模块中的dport扩展匹配条件。

现在，我们再回过头来看看扩展匹配条件的概念，就更加明白了。

扩展匹配条件被使用时，则需要依赖一些扩展模块，或者说，在使用扩展匹配条件之前，需要指定相应的扩展模块才行。

现在你明白了吗？ -m tcp表示使用tcp扩展模块，–dport表示tcp扩展模块中的一个扩展匹配条件，可用于匹配报文的目标端口。

注意，-p tcp与 -m tcp并不冲突，-p用于匹配报文的协议，-m 用于指定扩展模块的名称，正好，这个扩展模块也叫tcp。

其实，上例中，我们可以省略-m选项，示例如下。

```
[www.zsythink.net]#iptables -t filter -I INPUT -s 192.168.1.146 -p tcp --dport 22 -j REJECT
[www.zsythink.net]#iptables -vnL INPUT
Chain INPUT (policy ACCEPT 7 packets, 627 bytes)
 pkts bytes target     prot opt in     out     source               destination
    0     0 REJECT     tcp  --  *      *       192.168.1.146        0.0.0.0/0            tcp dpt:22 reject-with icmp-port-unreachable
    0     0 REJECT     tcp  --  *      *       192.168.1.146        0.0.0.0/0            tcp dpt:22 reject-with icmp-port-unreachable
[www.zsythink.net]#
```

当使用-p选项指定了报文的协议时，如果在没有使用-m指定对应的扩展模块名称的情况下，使用了扩展匹配条件， iptables默认会调用与-p选项对应的协议名称相同的模块。

上例中，我们使用-p选项指定了协议名称，使用扩展匹配条件–dport指定了目标端口，在使用扩展匹配条件的时候，如果没有使用-m指定使用哪个扩展模块，iptables会默认使用"-m 协议名"，而协议名就是-p选项对应的协议名，上例中，-p 对应的值为tcp，所以默认调用的扩展模块就为-m tcp，如果-p对应的值为udp，那么默认调用的扩展模块就为-m udp。

所以，上例中，其实"隐式"的指定了扩展模块，只是没有表现出来罢了。

所以，在使用扩展匹配条件时，一定要注意，如果这个扩展匹配条件所依赖的扩展模块名正好与-p对应的协议名称相同，那么则可省略-m选项，否则则不能省略-m选项，必须使用-m选项指定对应的扩展模块名称，这样说可能还是不是特别明了，在后续的举例中，我们会更加明了的理解这些概念。

有"目标端口"，就有"源端口"，代表"源端口"的扩展匹配条件为–sport

使用–sport可以判断报文是否从指定的端口发出，即匹配报文的源端口是否与指定的端口一致，–sport表示source-port，即表示源端口之意。

因为我们已经搞明白了dport，那么sport我就不再赘述了，示例如下

```
[www.zsythink.net]#iptables -t filter -I INPUT -s 192.168.1.146 -p tcp --sport 22 -j ACCEPT
[www.zsythink.net]#iptables -nvL INPUT
Chain INPUT (policy ACCEPT 9 packets, 680 bytes)
 pkts bytes target     prot opt in     out     source               destination
    0     0 ACCEPT     tcp  --  *      *       192.168.1.146        0.0.0.0/0            tcp spt:22
    9   476 REJECT     tcp  --  *      *       192.168.1.146        0.0.0.0/0            tcp dpt:22 reject-with icmp-port-unreachable
[www.zsythink.net]#
```

上例中，隐含了"-m tcp"之意，表示使用了tcp扩展模块的–sport扩展匹配条件。

扩展匹配条件是可以取反的，同样是使用"!"进行取反，比如 "! –dport 22"，表示目标端口不是22的报文将会被匹配到。

不管是–sport还是–dsport，都能够指定一个端口范围，比如，–dport 22:25表示目标端口为22到25之间的所有端口，即22端口、23端口、24端口、25端口，示例如下

```
[www.zsythink.net]#iptables -t filter -F INPUT
[www.zsythink.net]#iptables -t filter -I INPUT -s192.168.1.146 -p tcp --dport 22:25 -j REJECT
[www.zsythink.net]#iptables -nL INPUT
Chain INPUT (policy ACCEPT)
target     prot opt source               destination
REJECT     tcp  --  192.168.1.146        0.0.0.0/0            tcp dpts:22:25 reject-with icmp-port-unreachable
[www.zsythink.net]#
```

也可以写成如下图中的模样，下图中第一条规则表示匹配0号到22号之间的所有端口，下图中的第二条规则表示匹配80号端口以及其以后的所有端口（直到65535）。

```
[www.zsythink.net]#iptables -t filter -F INPUT
[www.zsythink.net]#iptables -t filter -I INPUT -s 182.168.1.146 -p tcp -m tcp --dport :22 -j REJECT
[www.zsythink.net]#iptables -t filter -I INPUT -s 182.168.1.146 -p tcp -m tcp --dport 80: -j REJECT
[www.zsythink.net]#iptables -nL INPUT
Chain INPUT (policy ACCEPT)
target     prot opt source               destination
REJECT     tcp  --  182.168.1.146        0.0.0.0/0            tcp dpts:80:65535 reject-with icmp-port-unreachable
REJECT     tcp  --  182.168.1.146        0.0.0.0/0            tcp dpts:0:22 reject-with icmp-port-unreachable
[www.zsythink.net]#
```

刚才聊到的两个扩展匹配条件都是tcp扩展模块的，其实，tcp扩展模块还有一个比较有用的扩展匹配条件叫做"–tcp-flags"，但是由于篇幅原因，以后再对这个扩展匹配条件进行总结。

借助tcp扩展模块的–sport或者–dport都可以指定一个连续的端口范围，但是无法同时指定多个离散的、不连续的端口，如果想要同时指定多个离散的端口，需要借助另一个扩展模块，"multiport"模块。

我们可以使用multiport模块的–sports扩展条件同时指定多个离散的源端口。

我们可以使用multiport模块的–dports扩展条件同时指定多个离散的目标端口。

示例如下

```
[www.zsythink.net]#iptables -t filter -F INPUT
[www.zsythink.net]#iptables -t filter -I INPUT -s 192.168.1.146 -p tcp -m multiport --dports 22,36,80 -j DROP
[www.zsythink.net]#iptables -nL INPUT
Chain INPUT (policy ACCEPT)
target     prot opt source               destination
DROP       tcp  --  192.168.1.146        0.0.0.0/0            multiport dports 22,36,80
[www.zsythink.net]#
```

上图示例表示，禁止来自146的主机上的tcp报文访问本机的22号端口、36号端口以及80号端口。

上图中，"-m multiport –dports 22,36,80"表示使用了multiport扩展模块的–dports扩展条件，以同时指定了多个离散的端口，每个端口之间用逗号隔开。

上图中的-m multiport是不能省略的，如果你省略了-m multiport，就相当于在没有指定扩展模块的情况下，使用了扩展条件（"–dports"），那么上例中，iptables会默认调用"-m tcp"，但是，"–dports扩展条件"并不属于"tcp扩展模块",而是属于"multiport扩展模块"，所以，这时就会报错。

综上所述，当使用–dports或者–sports这种扩展匹配条件时，必须使用-m指定模块的名称。

其实，使用multiport模块的–sports与–dpors时，也可以指定连续的端口范围，并且能够在指定连续的端口范围的同时，指定离散的端口号，示例如下。

```
[www.zsythink.net]#iptables -t filter -F INPUT
[www.zsythink.net]#iptables -t filter -I INPUT -s 182.168.1.146 -p tcp -m multiport --dports 22,80:88 -j REJECT
[www.zsythink.net]#iptables -nL INPUT
Chain INPUT (policy ACCEPT)
target     prot opt source               destination
REJECT     tcp  --  182.168.1.146        0.0.0.0/0            multiport dports 22,80:88 reject-with icmp-port-unreachable
[www.zsythink.net]#
```

上例中的命令表示拒绝来自192.168.1.146的tcp报文访问当前主机的22号端口以及80到88之间的所有端口号，是不是很方便？有没有很灵活？

不过需要注意，multiport扩展只能用于tcp协议与udp协议，即配合-p tcp或者-p udp使用。

再回过头看之前的概念，我想，你应该就更加明白了。

今天，我们只是初步的认识了扩展模块，以及扩展匹配条件，还有一些模块我们并没有总结，好饭不怕晚，后续会有对它们的总结。

小结

这篇文章中，我们主要总结了一些常用的"基础匹配条件"，并且初步的认识了两个"扩展模块"以及这两个扩展模块中一些常用的扩展条件，为了方便以后回顾，我们将它们总结如下。

首先我们要明确一点，当规则中同时存在多个匹配条件时，多个条件之间默认存在"与"的关系，即报文必须同时满足所有条件，才能被规则匹配。

基本匹配条件总结

-s用于匹配报文的源地址,可以同时指定多个源地址，每个IP之间用逗号隔开，也可以指定为一个网段。

```
#示例如下
iptables -t filter -I INPUT -s 192.168.1.111,192.168.1.118 -j DROP
iptables -t filter -I INPUT -s 192.168.1.0/24 -j ACCEPT
iptables -t filter -I INPUT ! -s 192.168.1.0/24 -j ACCEPT
```

-d用于匹配报文的目标地址,可以同时指定多个目标地址，每个IP之间用逗号隔开，也可以指定为一个网段。

```
#示例如下
iptables -t filter -I OUTPUT -d 192.168.1.111,192.168.1.118 -j DROP
iptables -t filter -I INPUT -d 192.168.1.0/24 -j ACCEPT
iptables -t filter -I INPUT ! -d 192.168.1.0/24 -j ACCEPT
```

-p用于匹配报文的协议类型,可以匹配的协议类型tcp、udp、udplite、icmp、esp、ah、sctp等（centos7中还支持icmpv6、mh）。

```
#示例如下
iptables -t filter -I INPUT -p tcp -s 192.168.1.146 -j ACCEPT
iptables -t filter -I INPUT ! -p udp -s 192.168.1.146 -j ACCEPT
```

-i用于匹配报文是从哪个网卡接口流入本机的，由于匹配条件只是用于匹配报文流入的网卡，所以在OUTPUT链与POSTROUTING链中不能使用此选项。

```
#示例如下
iptables -t filter -I INPUT -p icmp -i eth4 -j DROP
iptables -t filter -I INPUT -p icmp ! -i eth4 -j DROP
```

-o用于匹配报文将要从哪个网卡接口流出本机，于匹配条件只是用于匹配报文流出的网卡，所以在INPUT链与PREROUTING链中不能使用此选项。

```
#示例如下
iptables -t filter -I OUTPUT -p icmp -o eth4 -j DROP
iptables -t filter -I OUTPUT -p icmp ! -o eth4 -j DROP
```

扩展匹配条件总结

我们来总结一下今天认识的两个扩展模块，以及其中的扩展条件（并非全部，只是这篇文章中介绍过的）

tcp扩展模块

常用的扩展匹配条件如下：

-p tcp -m tcp –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

-p tcp -m tcp –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

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

multiport扩展模块

常用的扩展匹配条件如下：

-p tcp -m multiport –sports 用于匹配报文的源端口，可以指定离散的多个端口号,端口之间用"逗号"隔开

-p udp -m multiport –dports 用于匹配报文的目标端口，可以指定离散的多个端口号，端口之间用"逗号"隔开

```
#示例如下
iptables -t filter -I OUTPUT -d 192.168.1.146 -p udp -m multiport --sports 137,138 -j REJECT
iptables -t filter -I INPUT -s 192.168.1.146 -p tcp -m multiport --dports 22,80 -j REJECT
iptables -t filter -I INPUT -s 192.168.1.146 -p tcp -m multiport ! --dports 22,80 -j REJECT
iptables -t filter -I INPUT -s 192.168.1.146 -p tcp -m multiport --dports 80:88 -j REJECT
iptables -t filter -I INPUT -s 192.168.1.146 -p tcp -m multiport --dports 22,80:88 -j REJECT
```

好吧，感谢大家稀稀拉拉的赞赏和评论，希望这篇文章中的内容能对你有所帮助。

© 版权声明
文章版权归作者所有，转载时请在文章头部处注明出处，非常感谢。

THE END