防火墙的过滤与NAT技术

一、防火墙

防火墙是整个数据包进入主机前的第一道关卡。是一种位于内部网络与外部网络之间的网络安全系统，是一项信息安全的防护系统，依照特定的规则，允许或是限制传输的数据通过。防火墙主要通过Netfilter与TCPwrappers两个机制来管理的。

Linux的防火墙功能都是内核级别。内核2.4的版本中，也就是RHEL 6中都是使用的netfilter，用的管理工具是iptables，在rhel7中，内核3.0的版本后，防火墙默认用firewalld，管理工具是firewall-cmd(firewall-config)。rhel7中默认有三个防火墙（firewalld、iptables、ebtables），开启的只有firewalld，这三个都能实现内核级别的netfilter功能，但是它们之间的deamon存在冲突，所以建议mask其他两个。

```
[root@test .gnupg]# systemctl mask iptables.service
[root@test .gnupg]# systemctl mask ebtables.service
```

三、RHEL6 中iptables（虚拟机环境为RHEL7）

iptables是由内核来实现的，规则是从上往下匹配，匹配任何一条，不在往下匹配，不同服务之间的规则不存在影响。

```
[root@test ~]# yum install iptables-services
[root@test ~]# systemctl restart iptables
[root@test ~]# systemctl stop firewalld
[root@test ~]# systemctl mask firewalld
[root@test ~]# systemctl disable firewalld
```

1、iptables有表（table）和链（chain）的概念。

表分别有filter、nat、mangle

table（表名）	说明
filter	专门过滤包的。内建三个链：INPUT FORWARD OUTPUT。FORWARD链过滤所有不是本地产生的且目标不是本地的包；INPUT针对那些目的地是本地的包；OUTPUT是用来过滤所有本地生产的包
nat	主要用于网络地址转换 PREROUTING链的作用是在包刚刚到达防火墙时改变它的目标地址 OUTPUT链改变本地产生的包的目的地址 POSTROUTING链在包就要离开防火墙之前改变其源地址
mangle	主要用来操作数据包的。可以改变不同的包或包头的内容，比如：TTL这个表有五个内建链：PREROUTING POSTROUTING、INPUT、OUTPUT FORWARD。

2、filter表中的三条链

chain input　用来过滤进入的数据包

chain output　用来过滤出去的数据包

chain forward　用来决定是否转发

Filter表示iptables的默认表，因此如果你没有自定义表，那么就默认使用filter表，它具有以下三种内建链：

- INPUT链　- 处理来自外部的数据。
- OUTPUT链　- 处理向外发送的数据。
- FORWARD链　- 将数据转发到本机的其他网卡设备上。

3、nat表的三条链

chain prerouting　DNAT目的地址转换

chain postrouting　SNAT源地址转换

chain output　是否允许防火墙自身地址做转换

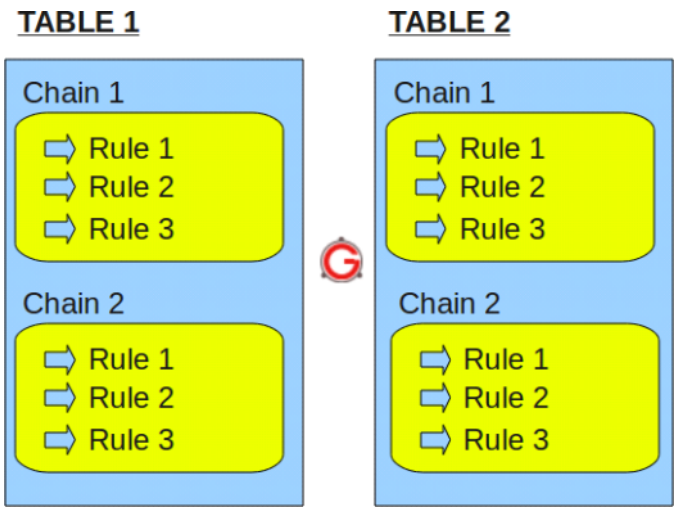
NAT表有三种内建链：

- PREROUTING链　- 处理刚到达本机并在路由转发前的数据包。它会转换数据包中的目标IP地址（destination ip address），通常用于DNAT(destination NAT)。
- POSTROUTING链　- 处理即将离开本机的数据包。它会转换数据包中的源IP地址（source ip address），通常用于SNAT（source NAT）。
- OUTPUT链　- 处理本机产生的数据包。

当数据包进入服务器时，Linux Kernel会查找对应的链，直到找到一条规则与数据包匹配。如果该规则的target是ACCEPT，就会跳过剩下的规则，数据包会被继续发送。如果该规则的target是DROP，该数据包会被拦截掉，kernel不会再参考其他规则。

Note：如果从始至终都没有一条规则与数据包匹配，而且表末尾又没有drop all的规则，那末该数据包会被根据默认策略accept/drop。

iptables的结构：iptables -> Tables -> Chains -> Rules




```
[root@test ~]# iptables -t filter -L -n    # 参数-t表示查看的是什么类型的表，-L表示用列表方式显示，-n表示禁止反向解析
Chain INPUT (policy ACCEPT 默认策略)
target     prot(协议:tcp/ip/all/icmp) opt source               destination
ACCEPT     all  --  0.0.0.0/0            0.0.0.0/0                                 state RELATED,ESTABLISHED(类型状态)
ACCEPT     icmp --  0.0.0.0/0            0.0.0.0/0
ACCEPT     all  --  0.0.0.0/0            0.0.0.0/0
ACCEPT     tcp  --  0.0.0.0/0            0.0.0.0/0                                 state NEW tcp dpt:22
REJECT     all  --  0.0.0.0/0            0.0.0.0/0                                reject-with icmp-host-prohibited

Chain FORWARD (policy ACCEPT)
target     prot opt source               destination
REJECT     all  --  0.0.0.0/0            0.0.0.0/0           reject-with icmp-host-prohibited

Chain OUTPUT (policy ACCEPT)
target     prot opt source               destination

[root@hf ~]# iptables -t  nat  -L -n  查看iptables   查看nat表
默认情况下不输入-t代表就是查看的filter表
[root@hf ~]# iptables  -L -n   查看filter表
4、iptables的语法和使用
[root@test ~]# iptables -t filter|nat  \         # 参数-t表示指明使用的表，如果不写，默认使用的是filter表
> -A|D|I   \                                    # 参数-A表示添加，-D表示删除，-I表示插入
> INPUT|OUTPUT|FORWARD   \                      # 指定链
> -p  |tcp|udp|icmp|ip   \                      # 指定协议
> -s 192.168.24.0/24|  ! -s 192.168.24.0/24\   # -s 指明源地址（如果地址前加！表示排除这个地址，可以理解为除了这个地址）
> --sport  m:n --dport m:n  \  这里指明的是源端口和目的端口的范围，可以只写源端口，也可以只写目的端口，当然也可以只写一个端口
> -j  ACCEPT|DROP|REJECT|LOG    参数-j表示执行的动作，分别表示允许、丢弃、拒绝和记录到日志。
```

实例：

```
[root@test ~]# iptables -A INPUT -p tcp -s 192.168.88.0/24   --dport 22 -j REJECT    定义了一个防后墙规则，在filter表中的input链上，如果源地址是192.168.88.0网段，目的端口是22的tcp连接都要拒绝。
[root@test html]# iptables -t filter -I INPUT 1 -p tcp -s 192.168.88.141 --dport 22 -j REJECT
```

注：在默认的防火墙规则中，-A添加的规则会被置为最后，如果此时前面的规则中已经允许了ssh的放行规则，那么这条添加的规则是不会生效的。所以，我们应该将这个规则删除后修改为插入。

```
[root@test ~]# iptables -L -n --line-number   # 查看防火墙规则时显示规则的序号，有利于添加删除
[root@test ~]# iptables -D INPUT 6   删除filter表的input链上的第6个规则

[root@test ~]# iptables -I INPUT 1 -p tcp -s 192.168.88.11 --dport 22 -j REJECT
针对一个特定的IP做的过滤。
[root@test ~]# iptables -I INPUT 1 -p tcp -m mac --mac=00:0c:29:45:de:d0 --dport 22 -j REJECT   在filter表中的INPUT链上插入成第一个规则，满足协议是tcp，mac地址和目的端口22的数据包都拒绝

[root@test ~]# systemctl restart iptables # 重启防火墙，如果写入的规则没有保存，重启防火墙后所有规则会复位成默认的

[root@test ~]# iptables -I INPUT -s 192.168.88.0/24 -p tcp --dport 22 -j ACCEPT
[root@test ~]# iptables -A INPUT -p tcp --dport 22 -j REJECT

当所有链中什么都没有的情况下，表中的链会依据默认的策略来执行。而默认策略通常都是policy accept。
[root@test ~]# iptables -F   清空防火墙规则
[root@test ~]# iptables -P INPUT DROP  改变INPUT链默认的策略为DROP
```

备注：防火墙中的数据包还可以根据类型状态来定义
NEW：某个连接的第一个包
ESTABLISHED：是一个已经建立了连接的包。一个链接从new状态变为established状态，只需要收到应答包既可。ICMP的错误和重定向等信息包都可以看做是ESTABLISHED状

态。
RELATED：前提是首先存在了ESTABLISHED状态，在这个状态的基础上产生了一个额外的连接
INVALID：这个包没有已知的流与它存在关联。这样的包可能是包头出现了问题，或者数据本身传输有丢失等，通知我们都要把它DROP。

实例：我们可以依据这四种包状态来进阶设置防火墙，进一步阻止类似灰鸽子这样的软件或木马主动从内向外发起请求。
[root@test ~]# iptables -P OUTPUT DROP　首先将OUTPUT链上所有出去的包都设置为drop
[root@test ~]# iptables -I OUTPUT -p tcp -m state --state=RELATED,ESTABLISHED -j ACCEPT
-m state: 启用状态匹配模块（state matching module）
--state:　状态匹配模块的参数。当SSH客户端第一个数据包到达服务器时，状态字段为NEW；建立连接后数据包的状态字段都是ESTABLISHED
设置防火墙output链，满足条件是协议是tcp，包的状态是RELATED和ESTABLISHED两种状态时可以放。换句话说，防火墙主动发起的请求都不允许出去。

iptables防火墙的规则备份与恢复：
[root@test ~]# iptables-save > iptables.backup　　# 备份防火墙规则到指定的文件（重定向）
[root@test ~]# iptables -F　　# 清空防火墙规则，默认策略不会被清空，仍然保留
[root@test ~]# iptables-restore < iptables.backup #

iptables中的防火墙和NAT
N1（192.168.24.88）---(192.168.24.1)FW(192.168.88.170)--- （192.168.88.156）N2
我们在防火墙上开启IPV4转发功能
[root@FW ~]# vim /etc/sysctl.conf
net.ipv4.ip_forward=1
net.ipv4.ip_forward = 1

[root@FW ~]# iptables -t nat -A POSTROUTING -s 10.0.0.0/24 -j SNAT --to-source 20.0.0.1
nat的实现数据包的结构会有变化：
当N1向N2发起ping的请求时，数据包的源地址是10.0.0.2，目的地址是20.0.0.2。数据包到达FW后，FW根据NAT表的规则，将源地址转换成了20.0.0.1。所以对于N2来说，它的回复数据包的源地址是20.0.0.2，目的地址是20.0.0.1。当然在防火墙上会存在一张NAT表，记录了将地址转换的映射关系，自然可以将N2的回复的目的地址再替换成原始的地址10.0.0.2，而这个过程对于n1和n2来说都是透明的。

[root@FW ~]# iptables -t nat -D POSTROUTING 1　删除SNAT
[root@FW ~]# iptables -t nat -A POSTROUTING -s 10.0.0.0/24 -j MASQUERADE
上面这个命令可以实现动态的地址转换，假设我们的公网地址是一个动态的，不可能每次都手工去指定地址的转换，所以通过关键词masquerade来实现动态的转换

DNAT（目的地址转换）
企业中如果内部需要向外部提供一些服务，但又不希望将公网地址直接在服务上使用，或者内部的服务较多，不可能都一一绑定公网地址。这时可以使用目的地址转换的方式。防火墙会根据我们事先定义的规则，将数据包的目的地址转换成特定的地址，对于目的地址转换，有时也叫做端口映射。
为了实现目的地址转换，我们首先在目的服务器上开启一个apache服务。
[root@N2 ~]# yum install httpd*　-y　　安装apache服务器
[root@N2 ~]# vim /etc/httpd/conf/httpd.conf　编辑apache的配置文件，更新ServerName内容，去掉注释并将网站地址改为自身ip，这里就是20.0.0.2:80
[root@N2 ~]# echo “hello，this is page”　> /var/www/html/index.html
生成一个网页
[root@N2 ~]# service　httpd restart　重启网站服务
[root@N2 ~]# iptables -I INPUT -p tcp --dport 80 -j ACCEPT
[root@N2 ~]# service iptables save
[root@FW ~]# iptables -t nat -A PREROUTING -p tcp -d 10.0.0.1 --dport 80 -j DNAT --to-dest 20.0.0.2
[root@test ~]# iptables -t nat -A PREROUTING -p tcp -d 192.168.88.170 --dport 80 -j DNAT --to-dest 192.168.88.156
设置DNAT，指明如果数据包访问的目的地址是10.0.0.1，并且目的端口是80，那么防火墙就将目的地址10.0.0.1替换成20.0.0.2并转发

Nat表中OUTPUT链的使用：对于防火墙自身来说，如果他也希望通过10.0.0.1去访问这个内部的网站，实际情况时不可能的，以为防火墙在默认情况下不会对自己做地址转换，如果希望实现这个效果，需要对output链做配置。
[root@FW ~]# iptables -t nat　-A OUTPUT -p tcp -d 10.0.0.1 --dport 80 -j DNAT --to-dest 20.0.0.2

RHEL7中的防火墙firewalld
从rhel7开始，iptables被默认的firewalld替换。rhel7中默认支持3个防火墙，分部是iptables、ebtables和firewalld。默认开启的是firewalld，但是为了防止防火墙间出现冲突，我们建议将其他两个防火墙关闭或者mask。
[root@test ~]# systemctl status firewalld　查看防火墙的运行状态，默认启动
1、图形化使用firewalld
[root@test ~]# firewall-config　这个命令可以直接运行图形化工具
　　firewall有两种状态，默认运行的是runtime。这个状态所有的设置都是当前有效，重启后都会失效。实际生产环境中我们不要使用这个配置。另一个状态叫permanent，这个状态定义为下次启动生效，并且是永久生效的。
　　firewalld是按照区域来划分的。缺省的区域有9个区域。不同区域提供的安全级别也不同。默认使用的是public。drop区域执行最严格的策略，丢弃一切对外数据，并且不产生任何提示。block也是拒绝一切，但是他有提示信息。最宽松策略区域由trusted区域提供，它默认都允许。默认的public区域中的策略只允许ssh和dhcp6-client。

区域	默认策略规则
trusted	允许所有的数据包
home	拒绝流入的流量，除非与流出的流量相关；而如果流量与ssh、mdns、ipp-client、amba-client与dhcpv6-client服务相关，则允许流量
internal	等同于home区域
work	拒绝流入的流量，除非与流出的流量数相关；而如果流量与ssh、ipp-client与dhcpv6-client服务相关，则允许流量
public	拒绝流入的流量，除非与流出的流量相关；而如果流量与ssh、dhcpv6-client服务相关，则允许流量
external	拒绝流入的流量，除非与流出的流量相关；而如果流量与ssh服务相关，则允许流量
dmz	拒绝流入的流量，除非与流出的流量相关；而如果流量与ssh服务相关，则允许流量
block	拒绝流入的流量，除非与流出的流量相关
drop	拒绝流入的流量，除非与流出的流量相关

2、通过命令行配置firewalld

```
[root@test ~]# firewall-cmd --add-port=80/tcp
[root@test ~]# firewall-cmd --add-service=http   配置防火墙，允许http服务
[root@test ~]# firewall-cmd --add-service=http  --permanent
```

配置防火墙，允许http服务，permanent表示这个配置永久生效，但必须重启。

```
[root@test ~]# firewall-cmd --get-default-zone   查看防火墙默认的区域
[root@test ~]# firewall-cmd --get-services  查看防火墙默认支持过滤的服务
[root@test ~]# firewall-cmd --get-zones  查看默认支持的区域策略
[root@test ~]# firewall-cmd --zone=home --add-interface=ens33  把网卡ens33加入到home区域
[root@test ~]# firewall-cmd --get-zone-of-interface=ens33
```

查看网卡当前所在区域

```
[root@test html]# firewall-cmd --get-active-zones  # 查看当前区域
```

```
[root@test ~]# firewall-cmd --zone=home --query-service=http
```

查看home区域下的http服务是否开启

```
[root@test ~]# firewall-cmd --zone=home --add-service=http
```

向区域home中添加http服务

```
[root@test ~]# firewall-cmd --zone=home --remove-service=http
```

向区域home中删除http服务

```
[root@test ~]# firewall-cmd  --zone=home --query-port=80/tcp
```

查看区域home下的tcp端口80是否启用

```
[root@test html]# firewall-cmd --zone=home --add-port=80/tcp
[root@test html]# firewall-cmd --zone=home --add-port=80/tcp --per
```

```
[root@test ~]# firewall-cmd  --zone=home --list-all
```

查看区域home下的所有信息，list后面跟上查看的其他参数可以细化查看分类信息

```
[root@test ~]# firewall-cmd --zone=public --add-port=8080-8081/tcp --permanent
[root@test ~]# firewall-cmd --zone=public --list-ports --permanent
```

```
[root@test ~]# firewall-cmd --add-forward-port port=80:proto=tcp:toaddr=192.168.88.11:toport=80 --zone=public
```

开启端口转发，将向tcp80端口的请求的数据包全部交给源地址是192.168.88.11的80端口

```
root@yujmo:~# firewall-cmd --add-forward-port port=12345:proto=tcp:toaddr=111.230.148.160:toport=22
root@yujmo:~# firewall-cmd --add-forward-port port=12345:proto=tcp:toaddr=111.230.148.160:toport=22 --per

root@yujmo:~# firewall-cmd --remove-forward-port port=8301:proto=udp:toport=8301:toaddr=192.168.1.126
root@yujmo:~# firewall-cmd --remove-forward-port port=8301:proto=udp:toport=8301:toaddr=192.168.1.126 --per
```

RHEL7中的rich rules（富规则）：

```
[root@test ~]# firewall-cmd --zone=public --list-rich-rules
```

查看当前public区域所有的富规则

```
[root@test ~]# firewall-cmd  --zone=public --add-rich-rule='rule family=ipv4 source address="192.168.24.240" port port=22 protocol=tcp  reject'
```

拒绝192.168.24.240通过tcp协议的22端口访问