# netfilter/iptables

1. 概述:

Netfilter/iptables IP 信息包过滤系统是一种工具, 可用于添加、编辑和除去规则, 这些规则是在做信息包过滤决定时, 防火墙所遵循和组成的规则. 这些规则存储在专用的信息包过滤表中, 而这些表集成在Linux内核中. 在信息包过滤表中, 规则被分组放在我们所谓的链(chain)中.

虽netfilter/iptables IP 信息包过滤系统被称为单个实体, 但它实际由两个组件netfilter 和 iptables 组成.

1. netfilter/iptables:

netfilter 组件也称为内核空间(kernelspace), 是内核的一部分, 由一些信息过滤表组成, 这些表包含内核用来控制信息包过滤处理的规则集.

Iptables 组件是一种工具, 也称为用户空间(userspace), 它使插入、修改和除去信息包过滤表中的规则变得容易.

Iptables包含4个表, 5个链. 其中表是按照对数据包的操作区分的, 链是按照不同的Hook点来区分的, 表和链实际上是netfilter的两个维度.

1. 四个表:

四个表: filter, nat, mangle, raw, 默认表是filter(没有指定表的时候就是filter表).

表的处理优先级: raw>mangle>nat>filter.

Filter: 一般的过滤功能

Nat: 用于nat 功能(端口映射, 地址映射等)

Mangle: 用于对特定数据包的修改

Raw: 优先级最高, 设置raw时一般是为了不再让iptables做数据包的链接跟踪处理, 提高性能

RAW表只使用在PREROUTING链和OUTPUT链上, 因为优先级最高, 从而可以对收到的数据包在连接跟踪前进行处理. 一旦用户使用了RAW表, 在某个链上, RAW表处理完后, 将跳过NAT 表和ip\_conntrack处理, 即不再做地址转换和数据包的链接跟踪处理了.

RAW表可以应用在那些不需要做nat的情况下, 以提高性能. 如大量访问的web服务器, 可以让80端口不再让iptables做数据包的链接跟踪处理, 以提高用户的访问速度.

四、5个链: PREROUTING, INPUT, FORWARD, OUTPUT, POSTROUTING.

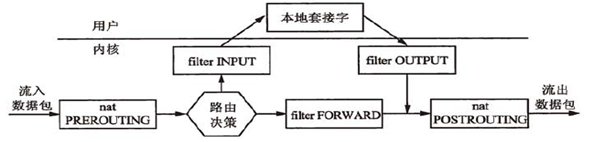
**PREROUTING:** 数据包进入路由表之前

**INPUT:** 通过路由表后目的地为本机

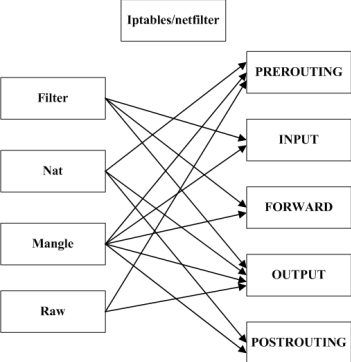
**FORWARDING:** 通过路由表后, 目的地不为本机

**OUTPUT:** 由本机产生, 向外转发

**POSTROUTING:** 发送到网卡接口之前. 如下图:

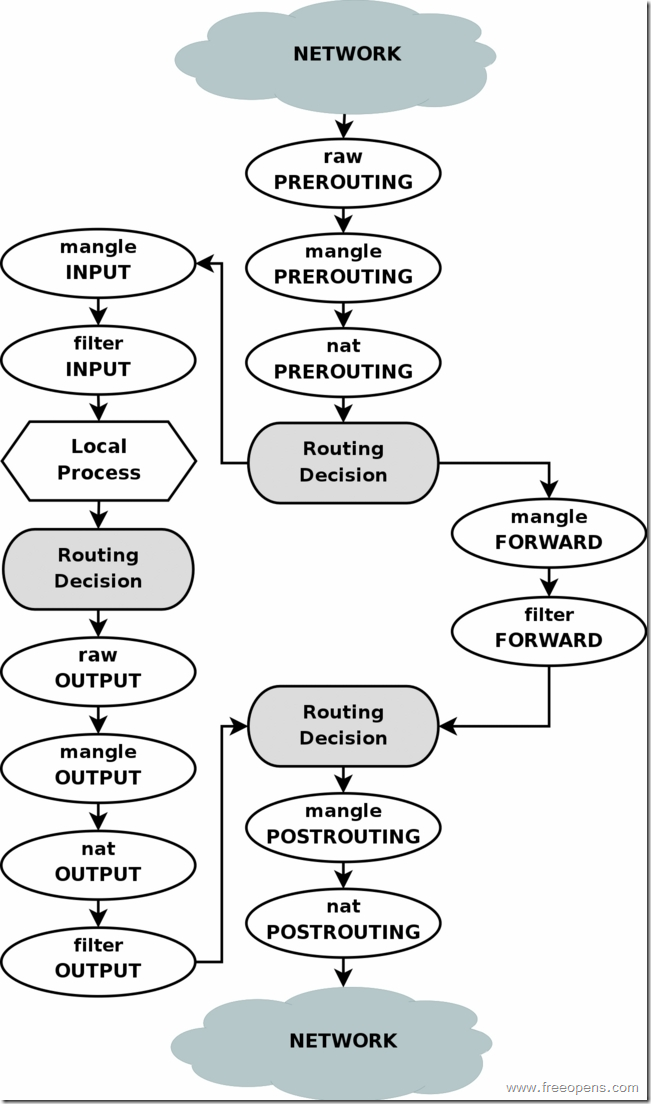


Iptables中表和链的对应关系如下:



1. iptables的数据包的流程.

一个数据包到达时, 依次穿过各个链和表.



基本步骤如下:

1. 数据包到达网络接口, 比如eth0
2. 进入raw表的PREROUTING链, 这个链的作用是赶在连接跟踪之前处理数据包.
3. 如果进行了连接跟踪, 在此处理.
4. 进入mangle表的PREROUTING链, 在此可以修改数据包, 比如TOS等.
5. 进入nat 表的PREROUTING链, 可以在此做DNAT, 但不要做过滤.
6. 决定路由, 看是交给本地主机还是转发给其它主机.

**到了这里我们就得分两种不同情况进行讨论了, 一种情况就是数据包要转发给其它主机, 这时候它会依次经过:**

1. 进入mangle表的FORWARD链, 这里也比较特殊, 这是在第一次路由决定之后, 在进行最后的路由决定之前, 我们仍然可以对数据包进行某些修改.
2. 进入filter表的FORWARD链, 在这里我们可以对所有转发的数据包进行过滤. 需要注意的是: 经过这里的数据包是转发的, 方向是双向的.
3. 进入mangle表的POSTROUTING链, 到这里已经做完了所有的路由决定, 但数据包仍然在本地主机, 我们还可以进行某些修改.
4. 进入nat表的POSTROUTING链, 在这里一般都是用来做SNAT, 不要在这里进行过滤.
5. 进入出去的网络接口. 完毕.

**另一种情况是, 数据包就是发给本地主机的, 那么它会依次穿过:**

7.进入mangle表的INPUT链, 这里是在路由之后, 交由本地主机之前, 我们也可以进行一些相应的修改.

8.进入filter表的INPUT链, 在这里我们可以对流入的所有数据包进行过滤, 无论它来自哪个网络接口.

9.交给本地主机的应用程序进行处理.

10.处理完毕后进行路由决定, 看该往那里发出.

11. 进入raw表的OUTPUT链, 这里是在连接跟踪处理本地的数据包之前.

12.连接跟踪对本地的数据包进行处理.

13.进入mangle表的OUTPUT链, 在这里我们可以修改数据包, 但不要做过滤.

14.进入nat表的OUTPUT链, 可以对防火墙自己发出的数据做NAT.

15.再次进行路由决定.

16.进入filter表的OUTPUT链, 可以对本地出去的数据包进行过滤.

17.进入mangle表的POSTROUTING链, 同上一种情况的第9步. 注意, 这里不光对经过防火墙的数据包进行处理, 还对防火墙自己产生的数据包进行处理.

18.进入nat表的POSTROUTING链, 同上一种情况的第10步.

19.进入出去的网络接口. 完毕.

六、iptables raw表的使用

增加raw表, 在其他表处理之前, -j NOTRACK跳过其它表处理

状态除了以前的四个还增加了一个UNTRACKED

例如:

可以使用”NOTRACK”target允许规则指定80端口的包不进入链接跟踪/NAT子系统

Iptables -t raw -A PREROUTING -d 1.2.3.4 -p tcp –dport 80 -j NOTRACK

Iptables -t raw -A PREROUTING -s 1.2.3.4 -p tcp –sport 80 -j NOTRACK

Iptables -A FORWARD -m state –state UNTRACKED -j ACCEPT

1. **解决ip\_conntrack: table full, dropping packet的问题**

在启用了iptables web 服务器上, 流量高的时候经常会出现下面的错误:

Ip\_conntrack: table full, dropping packet

这个问题的原因是由于web服务器收到了大量的连接, 在启用了iptables 的情况下, iptables会把所有的链接都做链接跟踪处理, 这样iptables就会有一个链接跟踪表, 当这个表满的时候, 就会出现上面的错误.

Iptables的链接跟踪表最大容量为/proc/sys/net/ipv4/ip\_conntrack\_max, 链接碰到各种状态的超时后就会从表中删除.

**解决方法一般有两个:**

1. 加大ip\_conntrack\_max值

vi /etc/sysctl.conf

net.ipv4.ip\_conntrack\_max=393216

net.ipv4.netfilter.ip\_conntrack\_max=393216

1. 降低ip\_conntrack timeout时间

vi /etc/sysctl.conf

net.ipv4.netfilter.ip\_conntrack\_tcp\_timeout\_established=300

net.ipv4.netfilter.ip\_conntrack\_tcp\_timeout\_time\_wait=120

net.ipv4.netfilter.ip\_conntrack\_tcp\_timeout\_close\_wait=60

net.ipv4.netfilter.ip\_conntrack\_tcp\_timeout\_fin\_wait=120

一般情况下都可以解决问题, 但在极端情况下, 还是不够用, 这样就得反其道而行, 用釜底抽薪的方法. Iptables的raw表是不做数据包的链接跟踪处理的, 我们就把那些连接量非常大的链接加入到iptables raw表.

如一台web服务器可以这样:

Iptables -t raw -A PREROUTING -d 1.2.3.4 -p tcp –dport 80 -j NOTRACK

Iptables -A FORWARD -m state –state UNTRACKED -j ACCEPT

1. 实例说明:
2. 单个规则实例

Iptables -F

# -F 是清除的意思, 作用就是把FILTRE TABLE的所有链的规则都清空

Iptables -A INPUT -s 172.20.20.1/32 -m state –state NEW, ESTABLISHED, RELATED -j ACCEPT

# 在filter表的INPUT链匹配源地址是172.20.20.1的主机, 状态分别是NEW, ESTABLISHED, RELATED的都放行.

Iptables -A INPUT -s 172.20.20.1/32 -m state –state NEW, ESTABLISHED -p tcp -m multiport –dport 123, 110 -j ACCEPT

# -p 指定协议, -m指定模块, multiport模块的作用就是可以连续匹配多各不相邻的端口号. 完整的意思就是源地址是172.20.20.1的主机, 状态分别是NEW, ESTABLISHED, RELATED的, TCP协议, 目的端口分别为123和110的数据包都可以通过.

Iptables -A INPUT -s 172.20.22.0/24 -m state –state NEW, ESTABLISHED -p tcp -m multiport –dport 123, 110 -j ACCEPT

Iptables -A INPUT -s 0/0 -m state –state NEW -p tcp -m multiport –dport 123, 110 -j DROP

#这句意思为源地址是0/0的NEW状态的的TCP数据包都禁止访问我的123和110端口.

Iptables -A INPUT -s ! 172.20.89.0/24 -m state –state NEW -p tcp -m multiport –dport 1230, 110 -j DROP

#”!”号的意思 取反. 就是除了172.20.89.0这个IP段的地址都DROP.

Iptables -R INPUT 1 -s 192.168.6.99 -p tcp –dport 22 -j ACCEPT

替换INPUT链中的第一条规则

Iptables -t filter -L INPUT -vn

以数字形式详细显示filter表INPUT链的规则

#----------------------------NAT IP----------------------------

#以下操作是在NAT TABLE里面完成的.

Iptables -t nat -F

Iptables -t nat -A PREROUTING -d 192.168.102.55 -p tcp –dport 90 -j DNAT –to 172.20.11.1:800

#-A PREROUTING 指定在路由前做的. 完整的意思是在NAT TABLE的路由前处理, 目的地为192.168.102.55的目的端口为90的我们做DNAT处理, 给他转向到172.20.11.1:800那里去.

Iptables -t nat -A POSTROUTING -d 172.20.11.1 -j SNAT –to 192.168.102.55

#-A POSTROUTING路由后. 意思为在NAT TABLE的路由后处理, 凡是目的地为172.20.11.1的, 我们都给他做SNAT转换, 把源地址改写成192.168.102.55

Iptables -A INPUT -d 192.168.20.0/255.255.255.0 -i eth1 -j DROP

Iptables -A INPUT -s 192.168.20.0/255.255.255.0 -i eth1 -j DROP

Iptables -A OUTPUT -d 192.168.20.0/255.255.255.0 -o eth1 -j DROP.

Iptables -A OUTPUT -s 192.168.20.0/255.255.255.0 -o eth1 -j DROP’

#上例中, eth1是一个与外部internet相连, 而192.168.20.0则是内部网的网络号, 上述规则用来防止IP欺骗, 因为出入eth1的包的IP应该是公共IP

Iptables -A INPUT -s 255.255.255.255 -i eth0 -j DROP

Iptables -A INPUT -s 224.0.0.0/224.0.0.0 -i eth0 -j DROP

Iptables -A INPUT -d 0.0.0.0 -I eth0 -j DROP

#防止广播包从IP代理服务器进入局域网:

Iptables -A INPUT -p tcp -m tcp –sport 5000 -j DROP

Iptables -A INPUT -p udp -m udp –sport 5000 -j DROP

Iptables -A OUTPUT -p tcp -m tcp –dport 5000 -j DROP

Iptables -A OUTPUT -p udp -m udp –dport 5000 -j DROP

#屏蔽端口5000

Iptables -A INPUT -s 211.148.130.129 -i eth1 -p tcp -m tcp –dport 3306 -j DROP

Iptables -A INPUT -s 192.168.20.0/255.255.255.0 -i eth0 -p tcp -m tcp –dport 3306 -j ACCEPT

Iptables -A INPUT -s 211.148.130.128/255.255.255.240 -i eth1 -p tcp -m tcp –dport 3306 -j ACCEPT

Iptables -A INPUT -p tcp -m tcp –dport 3306 -j DROP

#防止internet网的用户访问Mysql服务器

Iptables -A FORWARD -p TCP –dport 22 -j REJECT –reject-with tcp-reset

#REJECT, 类似于DROP, 但向发送该包的主机回复由—reject-with指定的信息, 从而可以很好地隐藏防火墙的存在

1. www的iptables实例

#!/bin/bash

export PATH=/sbin:/usr/sbin:/bin:/usr/bin

#加载相关模块

modprobe iptable\_nat

modprobe ip\_nat\_ftp

modprobe ip\_nat\_irc

modprobe ip\_conntrack

modprobe ip\_conntrack\_ftp

modprobe ip\_conntrack\_irc

modprobe ipt\_limit

echo 1 >;/proc/sys/net/ipv4/icmp\_echo\_ignore\_broadcasts

echo 0 >;/proc/sys/net/ipv4/conf/all/accept\_source\_route

echo 0 >;/proc/sys/net/ipv4/conf/all/accept\_redirects

echo 1 >;/proc/sys/net/ipv4/icmp\_ignore\_bogus\_error\_responses

echo 1 >;/proc/sys/net/ipv4/conf/all/log\_martians

echo 1 >;/proc/sys/net/ipv4/tcp\_syncookies

iptables -F

iptables -X

iptables -Z

##允许本地回路 Loopback -Allow unlimited traffic

Iptables -A INPUT -i lo -j ACCEPT

Iptables -A OUTPUT -o lo -j ACCEPT

##防止SYN洪水 SYN-Flooding Protection

Iptables -N syn-flood

Iptables -A INPUT -i ppp0 -p tcp –syn -j syn-flood

Iptables -A syn-flood -m limit –limit 1/s –limit-burst 4 -j RETURN

Iptables -A syn-flood -j DROP

##确保新连接是设置了SYN标记的包 Make sure that new TCP connections are SYN packets

Iptables -A INPUT -i eth0 -p tcp ! –syn -m state –state NEW -j DROP

##允许HTTP的规则

Iptables -A INPUT -i ppp0 -p tcp -s 0/0 –sport 80 -m state –state ESTABLISHED, RELATED -j ACCEPT

Iptables -A INPUT -i ppp0 -p tcp -s 0/0 –sport 443 -m state –state ESTABLISHED, RELATED -j ACCEPT

Iptables -A INPUT -i ppp0 -p tcp -d 0/0 –dport 80 -j ACCEPT

Iptables -A INPUT –i ppp0 -p tcp -d 0/0 –dport 443 -j ACCEPT

## 允许DNS的规则

Iptables -A INPUT –i ppp0 -p udp -s 0/0 –sport 53 -m state –state ESTABLISHED -j ACCEPT

Iptables -A INPUT -i ppp0 -p udp -d 0/0 –dport 53 -j ACCEPT

## IP包流量限制 IP packets limit

Iptables -A INPUT -f -m limit –limit 100/s –limit-burst 100 -j ACCEPT

Iptables -A INPUT -i eth0 -p icmp -j DROP

## 允许SSH

Iptables -A INPUT -p tcp -s ip1/32 –dport 22 -j ACCEPT

Iptables -A INPUT -p tcp -s ip2/32 –dport 22 -j ACCEPT

## 其它情况不允许 Anything else not allowed

Iptables -A INPUT -i eth0 -j DROP

1. 一个包过滤防火墙实例

环境：redhat9 加载了string time等模块

eth0 接外网──ppp0

eth1 接内网──192.168.0.0/24

#!/bin/sh

#

modprobe ipt\_MASQUERADE

modprobe ip\_conntrack\_ftp

modprobe ip\_nat\_ftp

iptables -F

iptables -t nat -F

iptables -X

iptables -t nat -X

###########################INPUT#####################

iptables -P INPUT DROP

iptables -A INPUT -m state --state ESTABLISHED,RELATED -j ACCEPT

iptables -A INPUT -p tcp -m multiport --dports 110,80,25 -j ACCEPT

iptables -A INPUT -p tcp -s 192.168.0.0/24 --dport 139 -j ACCEPT

#允许内网samba,smtp,pop3,连接

iptables -A INPUT -i eth1 -p udp -m multiport --dports 53 -j ACCEPT

#允许dns连接

iptables -A INPUT -p tcp --dport 1723 -j ACCEPT

iptables -A INPUT -p gre -j ACCEPT

#允许外网vpn连接

iptables -A INPUT -s 192.186.0.0/24 -p tcp -m state --state ESTABLISHED,RELATED -j ACCEPT

iptables -A INPUT -i ppp0 -p tcp --syn -m connlimit --connlimit-above 15 -j DROP

#为了防止DoS太多连接进来,那么可以允许最多15个初始连接,超过的丢弃

iptables -A INPUT -s 192.186.0.0/24 -p tcp --syn -m connlimit --connlimit-above 15 -j DROP

#为了防止DoS太多连接进来,那么可以允许最多15个初始连接,超过的丢弃

iptables -A INPUT -p icmp -m limit --limit 3/s -j LOG --log-level INFO --log-prefix "ICMP packet IN: "

iptables -A INPUT -p icmp -j DROP

#禁止icmp通信-ping 不通

iptables -t nat -A POSTROUTING -o ppp0 -s 192.168.0.0/24 -j MASQUERADE

#内网转发

iptables -N syn-flood

iptables -A INPUT -p tcp --syn -j syn-flood

iptables -I syn-flood -p tcp -m limit --limit 3/s --limit-burst 6 -j RETURN

iptables -A syn-flood -j REJECT

#防止SYN攻击 轻量

iptables -P FORWARD DROP

iptables -A FORWARD -p tcp -s 192.168.0.0/24 -m multiport --dports 80,110,21,25,1723 -j ACCEPT

iptables -A FORWARD -p udp -s 192.168.0.0/24 --dport 53 -j ACCEPT

iptables -A FORWARD -p gre -s 192.168.0.0/24 -j ACCEPT

iptables -A FORWARD -p icmp -s 192.168.0.0/24 -j ACCEPT

#允许 vpn客户走vpn网络连接外网

iptables -A FORWARD -m state --state ESTABLISHED,RELATED -j ACCEPT

iptables -I FORWARD -p udp --dport 53 -m string --string "tencent" -m time --timestart 8:15 --timestop 12:30 --days Mon,Tue,Wed,Thu,Fri,Sat -j DROP

#星期一到星期六的8:00-12:30禁止qq通信

iptables -I FORWARD -p udp --dport 53 -m string --string "TENCENT" -m time --timestart 8:15 --timestop 12:30 --days Mon,Tue,Wed,Thu,Fri,Sat -j DROP

#星期一到星期六的8:00-12:30禁止qq通信

iptables -I FORWARD -p udp --dport 53 -m string --string "tencent" -m time --timestart 13:30 --timestop 20:30 --days Mon,Tue,Wed,Thu,Fri,Sat -j DROP

iptables -I FORWARD -p udp --dport 53 -m string --string "TENCENT" -m time --timestart 13:30 --timestop 20:30 --days Mon,Tue,Wed,Thu,Fri,Sat -j DROP

#星期一到星期六的13:30-20:30禁止QQ通信

iptables -I FORWARD -s 192.168.0.0/24 -m string --string "qq.com" -m time --timestart 8:15 --timestop 12:30 --days Mon,Tue,Wed,Thu,Fri,Sat -j DROP

#星期一到星期六的8:00-12:30禁止qq网页

iptables -I FORWARD -s 192.168.0.0/24 -m string --string "qq.com" -m time --timestart 13:00 --timestop 20:30 --days Mon,Tue,Wed,Thu,Fri,Sat -j DROP

#星期一到星期六的13:30-20:30禁止QQ网页

iptables -I FORWARD -s 192.168.0.0/24 -m string --string "ay2000.net" -j DROP

iptables -I FORWARD -d 192.168.0.0/24 -m string --string "宽频影院" -j DROP

iptables -I FORWARD -s 192.168.0.0/24 -m string --string "×××" -j DROP

iptables -I FORWARD -p tcp --sport 80 -m string --string "广告" -j DROP

#禁止ay2000.net，宽频影院，×××，广告网页连接 !但中文 不是很理想

iptables -A FORWARD -m ipp2p --edk --kazaa --bit -j DROP

iptables -A FORWARD -p tcp -m ipp2p --ares -j DROP

iptables -A FORWARD -p udp -m ipp2p --kazaa -j DROP

#禁止BT连接

iptables -A FORWARD -p tcp --syn --dport 80 -m connlimit --connlimit-above 15 --connlimit-mask 24 -j DROP

#只允许每组ip同时15个80端口转发

sysctl -w net.ipv4.ip\_forward=1 &>/dev/null

#打开转发

sysctl -w net.ipv4.tcp\_syncookies=1 &>/dev/null

#打开 syncookie (轻量级预防 DOS 攻击)

sysctl -w net.ipv4.netfilter.ip\_conntrack\_tcp\_timeout\_established=3800 &>/dev/null

#设置默认 TCP 连接痴呆时长为 3800 秒(此选项可以大大降低连接数)

sysctl -w net.ipv4.ip\_conntrack\_max=300000 &>/dev/null

#设置支持最大连接树为 30W(这个根据你的内存和 iptables 版本来，每个 connection 需要 300 多个字节)

iptables -I INPUT -s 192.168.0.50 -j ACCEPT

iptables -I FORWARD -s 192.168.0.50 -j ACCEPT

#192.168.0.50是我的机子，全部放行!