Iptables限制端口访问

1、 Iptables介绍

a) 概念

Linux 系统在内核中提供了对报文数据包过滤和修改的官方项目名为 Netfilter，它指的是 Linux 内核中的一个框架，它可以用于在不同阶段将某些钩子函数（hook）作用域网络协议栈。Netfilter 本身并不对数据包进行过滤，它只是允许可以过滤数据包或修改数据包的函数挂接到内核网络协议栈中的适当位置。这些函数是可以自定义的。

iptables 是用户层的工具，它提供命令行接口，能够向 Netfilter 中添加规则策略，从而实现报文过滤，修改等功能。Linux 系统中并不止有 iptables 能够生成防火墙规则，其他的工具如 firewalld 等也能实现类似的功能。（引自 http://liaoph.com/iptables）

iptables 可以检测、修改、转发、重定向和丢弃 IPv4 数据包。过滤 IPv4 数据包的代码已经内置于内核中，并且按照不同的目的被组织成 **表** 的集合。**表** 由一组预先定义的 **链** 组成，**链** 包含遍历顺序规则。每一条规则包含一个谓词的潜在匹配和相应的动作（称为 **目标**），如果谓词为真，该动作会被执行。也就是说条件匹配。iptables 是用户工具，允许用户使用 **链** 和 **规则**。

防火墙策略一般分为两种，一种叫“通”策略，一种叫“堵”策略，通策略，默认门是关着的，必须要定义谁能进。堵策略则是，大门是洞开的，但是你必须有身份认证，否则不能进。所以我们要定义，让进来的进来，让出去的出去，所以通，是要全通，而堵，则是要选择。当我们定义的策略的时候，要分别定义多条功能，其中：定义数据包中允许或者不允许的策略，filter过滤的功能，而定义地址转换的功能的则是nat选项。为了让这些功能交替工作，我们制定出了“表”这个定义，来定义、区分各种不同的工作功能和处理方式

i. Iptables包含五张表（链的集合）
   1. raw用于配置数据包，raw中的数据包不会被系统跟踪
   2. filter存放所有与防火墙操作相关的默认表（常用）
   3. nat用于网络地址转换（常用）
   4. mangle用于对特定数据包的修改
   5. security用于强制访问控制网络规则

ii. 链

表由链组成，链是一些按顺序排列的规则的列表。默认的 filter 表包含 INPUT， OUTPUT 和 FORWARD 3条内建的链，这3条链作用于数据包过滤过程中的不同时间点。nat 表包含PREROUTING， POSTROUTING 和 OUTPUT 链。

iii. 规则

数据包的过滤基于 **规则**。**规则**由一个 *目标*（数据包包匹配所有条件后的动作）和很多 *匹配*（导致该规则可以应用的数据包所满足的条件）指定。一个规则的典型匹配事项是数据包进入的端口（例如：eth0 或者 eth1）、数据包的类型（ICMP, TCP, 或者 UDP）和数据包的目的端口。

iv. 模块儿

模块儿可以扩招iptables的过滤规则，使用更详细的过滤规则

v. 匹配

每个 iptables 规则都包含一组匹配和一个目标动作，后者定义了复合规则的数据包应该采取什么处理行为。iptables 匹配指定是数据包必须匹配的条件

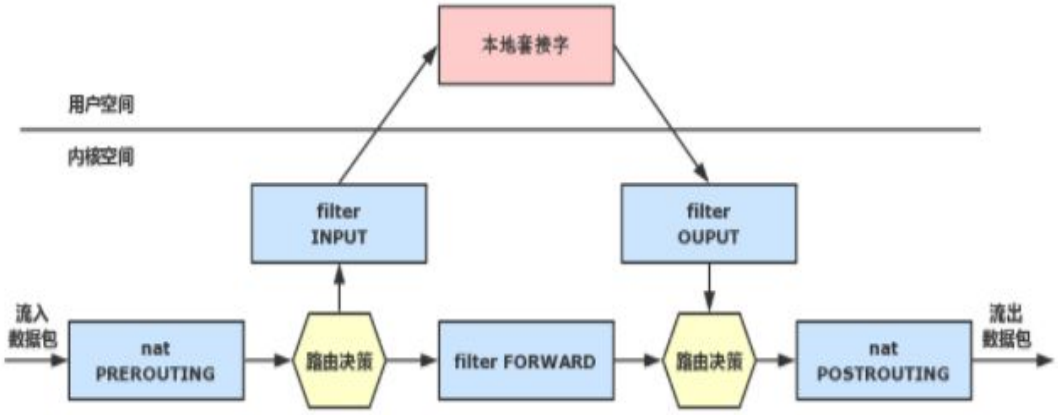
，只有当数据包满足所有的匹配条件时，iptables 才能根据规则的目标所指定的动作来处理该数据包。无论何时匹配了一条规则，相应的 target/jump 动作将会执行

vi. 目标动作

iptables对匹配的数据包执行一个目标动作，目标动作由 -j 或--jump来指定

ACCEPT	放行
DROP	丢弃
REJECT	发送ICMP拒绝
REDIRECT	端口重定向
LOG	记录日志
SNAT	源网络地址转换
DNAT	目的网络地址转换
MASQUERADE	源地址伪装
RETURN	返回

b) **数据包过滤流程**



i. PREROUTING (路由前)
ii. INPUT (数据包流入口)
iii. FORWARD (转发管卡)
iv. OUTPUT(数据包出口)
v. POSTROUTING（路由后）

这是NetFilter规定的五个规则链，任何一个数据包，只要经过本机，必将经过这五个链中的其中一个链。对于filter来讲一般只能做在3个链上INPUT ，FORWARD ，OUTPUT对于nat来讲一般也只能做在3个链上PREROUTING ，OUTPUT ，POSTROUTING

iptables是采用规则堆栈的方式来进行过滤，当一个封包进入网卡，会先检查 Prerouting，然后检查目的IP判断是否需要转送出去，接着就会跳到 INPUT 或 Forward 进行过滤，如果封包需转送处理则检查 Postrouting，如果是来自本机封包，则检查 OUTPUT 以及Postrouting。过程中如果符合某条规则将会进行处理，处理动作除了 ACCEPT、REJECT、DROP、REDIRECT 和MASQUERADE 以外，还多出 LOG、ULOG、DNAT、SNAT、MIRROR、QUEUE、RETURN、TOS、TTL、MARK等，其中某些处理动作不会中断过滤程序，某些处理动作则会中断同一规则链的过滤，并依照前述流程继续进行下一个规则链的过滤（注意：这一点与ipchains不同），一直到堆栈中的规则检查完毕为止。透过这种机制所带来的好处是，我们可以进行复杂、多重的封包过滤，简单的说，iptables可以进行纵横交错式的过滤（tables）而非链状过滤（chains）。**ACCEPT 将封包放行，进行完此处理动作后，将不再比对其它规则，直接跳往下一个规则链（nat:postrouting）**

c) 命令参考（使用—help查看命令参考
   i. 链管理命令
      1. -P（设置链的默认策略）
         默认策略一般只有两种
         iptables -P INPUT (DROP|ACCEPT)
      2. -F（清空链中的规则或清空表中的所有链）
         iptables -t nat -F INPUT　　//清空INPUT链中的所有规则
         iptables -t nat -F　　//清空表nat中的链
      3. -N（新建空的链）
      4. -X（删除自定义空链）
      5. -E（重命名： -E oldname newname)
      6. -X（清空链以及链中默认规则计数器）
   ii. 规则管理命令
      1. -A（在当前链的最后新加一个规则）
      2. -I（插入一条规则：-I num）
      3. -D（删除一条规则：-D num）
      4. -R（替换一条规则：-R num）
   iii. 查看管理命令
      1. -n（以数字形式显示主机名）
      2. -v（显示详细信息）
      3. -L（列出链中的规则）
      4. --line-numbers（显示规则行号）
d) 详细匹配
   i. -s
      指定源地址匹配（-s IP | IP/MASK ），并且地址前可以加！取反
   ii. -d
      匹配目标地址
   iii. -p(TCP|UDP|ICMP)
      匹配协议
   iv. -i|-o
      指定网络接口
   v. 隐含匹配（和协议搭配使用）
      --dport XX-XX | XX
      --sport XX-XX | XX
      --tcp-flags (SYN|ACK|FIN|PSH|RST)
      --tcpflags syn,ack,fin,rst syn//后面两个参数：检查的标志位，必须为1的标志位
      --icmp-type
   vi. -m modules(multiport|iprange|mac|state)
e) 配置范例

设置 filter 表 INPUT 链的默认策略为 DROP

```
iptables -P INPUT DROP
```

允许源地址为 172.16.0.0/16 网段的主机连接本机 SSH

```
iptables -A INPUT -s 172.16.0.0/16 -p tcp --dport 22 -j ACCEPT
```

允许 ICMP 请求报文

```
iptables -A INPUT -p icmp --icmp-type 8
```

允许 80 的 443 端口的访问（使用离散端口扩展模块）

```
iptables -A INPUT -p tcp -m multiport --dports 80,443 -j ACCEPT
```

允许 Telnet 的 23 端口访问，并限制同时只能有 5 个连接

```
iptables -A INPUT -p tcp --dport 23 -m connlimit --connlimit-above 5 -j REJECT
```

拒绝本地端口 80 发出的，含有 “communist” 关键字的响应报文的发出

```
iptables -A OUTPUT -p tcp --sport 80 -m string --algo kmp --string "communist" -j DROP
```

仅允许工作日的工作时间访问本机 UDP 53 端口

```
iptables -A INPUT -p udp --dport 53 -m time --timestart 08:00 --timestop 18:00 --weekdays Mon,Tue
```

允许 172.16.0.1 ~ 172.16.0.100 的主机访问本机 TCP 3306 端口

```
iptables -A INPUT -p tcp --dport 3306 -m iprange --src-range 172.16.0.1-172.16.0.100 -j ACCEPT
```

屏蔽指定主机（LOG后的--log-level可以指定日志级别）
iptables -A INPUT -p tcp -s 111.111.111.111 -j LOG
iptables -A INPUT -p tcp -s 111.111.111.111 -j DROP

仅允许指定网段通过指定网口通过ssh连接本机
iptables -A INPUT -i eth0 -p tcp -s 192.168.100.0/24 --dport 22 -m state --state NEW,ESTABLISHED -j ACCEPT
iptables -A OUTPUT -i eth0 -p tcp --sport 22 -m state --state ESTABLISHED -j ACCEPT

允许本机通过ssh连接其他主机
iptables -A INPUT -i eth0 -p tcp -s 192.168.100.0/24 --dport 22 -m state --state ESTABLIHED -j ACCEPT
iptables -A OUTPUT -i eth0 -p tcp --sport 22 -m state --state NEW,ESTABLISHED

-j ACCEPT

网口转发
iptables -A FORWARD -i eth0 -o eth1 -j ACCEPT

端口转发
iptables -t nat -A PREROUTING -p tcp -d 192.168.100.102 --dport 8888 -j DNAT --to 192.168.100.102:80

2、 安装配置限制端口访问

a) 在fedora，centos最新版本中使用firewalld作为默认防火墙，内部调用iptables
   i. Systemctl stop firewalld
   ii. Sudo yum install iptables*
   iii. Systemctl start iptables
   iv. 安装之后配置文件在/etc/sysconfig/iptables-config，过滤规则 /etc/sysconfig/iptables
b) 配置规则
   i. 配置filter表默认规则拒绝
      iptables -P INPUT DROP
   ii. 放通目的端口22（目的端口）
      iptables -A INPUT -p tcp -dport 22 -j ACCEPT
   iii. 允许源端口22发出的ssl数据
      iptables -A OUTPUT -p tcp –sport 22 -j ACCEPT
   iv. 拒绝本地不明端口发出的主动网络请求
      iptables -A OUTPUT -p all -j LOG
      iptables -A OUTPUT -p all -j DROP
   v. 也可基于状态检测
      iptables -A INPUT -i eth0 -p tcp --dport 22 -m state --state NEW,ESTABLISHED -j ACCEPT
      iptables -A OUTPUT -i eth0 -p tcp --sport 22 -m state --state ESTABLISHED -j ACCEPT

文档仅包含iptables部分，ipset，rsyslog另起文档说明