防火墙策略

需要root或者sudo权限才能设置,且一按回车后立马生效，若设置了22端口拒绝所有IP，即使自己连了22端口，也会立马被挤掉登录不上，如果有主机，则直接连接屏幕，如果是云服务器，则在官网实例的远程管理里用VNC连接

iptables的主要功能是实现对网络数据包进出设备及转发的控制。当数据包需要进入设备、从设备中流出或者经该设备转发、路由时，都可以使用iptables进行控制。

### **1.iptables中的“四表五链”及“堵通策略”**

A.**“四表”**是指，iptables的功能——filter, nat, mangle, raw.

　　　　filter, 控制数据包是否允许进出及转发（INPUT、OUTPUT、FORWARD）,可以控制的链路有input, forward, output

　　　　nat, 控制数据包中地址转换，可以控制的链路有prerouting, input, output, postrouting

　　　　mangle,修改数据包中的原数据，可以控制的链路有prerouting, input, forward, output, postrouting

　　　　raw,控制nat表中连接追踪机制的启用状况，可以控制的链路有prerouting, output

　　注：在centos7中，还有security表，不过这里不作介绍

B.**“五链”**是指内核中控制网络的NetFilter定义的五个规则链，分别为

　　　　PREROUTING, 路由前

　　　　INPUT, 数据包流入口

　　　　FORWARD, 转发管卡

　　　　OUTPUT, 数据包出口

　　　　POSTROUTING, 路由后

C**.堵通策略**是指对数据包所做的操作，一般有两种操作——“通（ACCEPT）”、“堵（DROP）”，还有一种操作很常见REJECT.DROP是不回应，REJECT是给出明确拒绝；

谈谈REJECT和DROP之间的区别，Ming写了一封信，向Rose示爱。Rose如果不愿意接受，她可以不回应Ming,这个时候Ming不确定Rose是否接到了信；Rose也可以同样写一封信，在信中明确地拒绝Ming。前一种操作就如同执行了DROP操作，而后一种操作就如同REJECT操作。

#### 永久保存

这样的设置好了，我们只是临时的， 重启服务器还是会恢复原来没有设置的状态

还要使用 service iptables save 进行保存

看到信息 firewall rules 防火墙的规则 其实就是保存在 /etc/sysconfig/iptables

可以打开文件查看 vi /etc/sysconfig/iptables

iptables-save > /etc/sysconfig/iptables

### **2.iptables命令的语法规则**

iptables [-t table] COMMAND [chain] CRETIRIA -j ACTION

　　-t table，是指操作的表，filter、nat、mangle或raw, 默认使用filter

　　COMMAND，子命令，定义对规则的管理

　　chain, 指明链路

　　CRETIRIA, 匹配的条件或标准

　　ACTION,操作动作

　　例如，不允许10.8.0.0/16网络对80/tcp端口进行访问，

iptables -A INPUT -s 10.8.0.0/16 -d 172.16.55.7 -p tcp --dport 80 -j DROP

　　查看iptables列表

iptables -nL

### **3.链管理**

　　-N, --new-chain chain：新建一个自定义的规则链；

　　-X, --delete-chain [chain]：删除用户自定义的引用计数为0的空链；

　　-F, --flush [chain]：清空指定的规则链上的规则；

　　-E, --rename-chain old-chain new-chain：重命名链；

　　-Z, --zero [chain [rulenum]]：置零计数器；

　　-P, --policy chain target， 设置链路的默认策略

### **4.规则管理**

　　-A, --append chain rule-specification：追加新规则于指定链的尾部；

　　-I, --insert chain [rulenum] rule-specification：插入新规则于指定链的指定位置，默认为首部；

　　-R, --replace chain rulenum rule-specification：替换指定的规则为新的规则；

　　-D, --delete chain rulenum：根据规则编号删除规则；

### **5.查看规则**

　　-L, --list [chain]：列出规则；

　　-v, --verbose：详细信息；

　　　　-vv， -vvv  更加详细的信息

　　-n, --numeric：数字格式显示主机地址和端口号；

　　-x, --exact：显示计数器的精确值；

　　--line-numbers：列出规则时，显示其在链上的相应的编号；

　　-S, --list-rules [chain]：显示指定链的所有规则；

　　查看规则的一般内容：



### **6.匹配条件**

匹配条件包括通用匹配条件和扩展匹配条件。

通用匹配条件是指针对源地址、目标地址的匹配，包括单一源IP、单一源端口、单一目标IP、单一目标端口、数据包流经的网卡以及协议。

扩展匹配条件指通用匹配之外的匹配条件。

**6.1通用匹配条件**

[!] -s, --source address[/mask][,...]：检查报文的源IP地址是否符合此处指定的范围，或是否等于此处给定的地址；

[!] -d, --destination address[/mask][,...]：检查报文的目标IP地址是否符合此处指定的范围，或是否等于此处给定的地址；

[!] -p, --protocol protocol：匹配报文中的协议，可用值tcp, udp, udplite, icmp, icmpv6,esp, ah, sctp, mh 或者 "all", 亦可以数字格式指明协议；  
[!] -i, --in-interface name：限定报文仅能够从指定的接口流入；only for packets entering the INPUT, FORWARD and PREROUTING chains.

[!] -o, --out-interface name：限定报文仅能够从指定的接口流出；for packets entering the FORWARD, OUTPUT and POSTROUTING chains.

**6.2扩展匹配条件**

* **隐含扩展匹配条件**

-p tcp：可直接使用tcp扩展模块的专用选项；  
　　[!] --source-port,--sport port[:port] 匹配报文源端口；可以给出多个端口，但只能是**连续的端口范围** ；

　　[!] --destination-port,--dport port[:port] 匹配报文目标端口；可以给出多个端口，但只能是**连续**的端口范围 ；

　　[!] --tcp-flags mask comp 匹配报文中的tcp协议的标志位；Flags are: SYN ACK FIN RST URG PSH ALL NONE；  
　　　　mask：要检查的FLAGS list，以逗号分隔；  
　　　　comp：在mask给定的诸多的FLAGS中，其值必须为1的FLAGS列表，余下的其值必须为0；

　　[!] --syn： --tcp-flags SYN,ACK,FIN,RST SYN  
  
-p udp：可直接使用udp协议扩展模块的专用选项：

　　[!] --source-port,--sport port[:port]

　　[!] --destination-port,--dport port[:port]

-p icmp  
　　[!] --icmp-type {type[/code]|typename}

　　　　0/0：echo reply

　　　　8/0：echo request

* 显式扩展匹配条件

　　必须用**-m option**选项指定扩展匹配的类型，常见的有以下几种，

**1、multiport**

以离散或连续的 方式定义多端口匹配条件，最多15个；  
  
　　[!] --source-ports,--sports port[,port|,port:port]...：指定多个源端口；

　　[!] --destination-ports,--dports port[,port|,port:port]...：指定多个目标端口；

iptables -I INPUT -d 172.16.0.7 -p tcp -m multiport --dports 22,80,139,445,3306 -j ACCEPT

**2、iprange**

以连续地址块的方式来指明多IP地址匹配条件；

　　[!] --src-range from[-to]

　　[!] --dst-range from[-to]

# iptables -I INPUT -d 172.16.0.7 -p tcp -m multiport --dports 22,80,139,445,3306 -m iprange --src-range 172.16.0.61-172.16.0.70 -j REJECT

**3、time**

匹配数据包到达的时间

　　--timestart hh:mm[:ss]

　　--timestop hh:mm[:ss]  
  
　　[!] --weekdays day[,day...]

　　[!] --monthdays day[,day...]

　　--datestart YYYY[-MM[-DD[Thh[:mm[:ss]]]]]

　　--datestop YYYY[-MM[-DD[Thh[:mm[:ss]]]]]

　　--kerneltz：使用内核配置的时区而非默认的UTC；  
  
**4、string**  
匹配数据包中的字符  
  
　　--algo {bm|kmp}

　　[!] --string pattern

　　[!] --hex-string pattern  
  
　　--from offset

　　--to offset

~]# iptables -I OUTPUT -m string --algo bm --string "gay" -j REJECT

**5、connlimit**  
用于限制同一IP可建立的连接数目  
  
　　--connlimit-upto n

　　--connlimit-above n

~]# iptables -I INPUT -d 172.16.0.7 -p tcp --syn --dport 22 -m connlimit --connlimit-above 2 -j REJECT

**6、limit**  
限制收发数据包的速率  
  
　　--limit rate[/second|/minute|/hour|/day]

　　--limit-burst number

~]# iptables -I OUTPUT -s 172.16.0.7 -p icmp --icmp-type 0 -j ACCEPT

**7、state**  
限制收发包的状态  
  
　　[!] --state state

　　INVALID, ESTABLISHED, NEW, RELATED or UNTRACKED.  
  
　　NEW: 新连接请求；

　　ESTABLISHED：已建立的连接；

　　INVALID：无法识别的连接；

　　RELATED：相关联的连接，当前连接是一个新请求，但附属于某个已存在的连接；

　　UNTRACKED：未追踪的连接；  
  
state扩展：

内核模块装载：  
　　nf\_conntrack  
　　nf\_conntrack\_ipv4  
  
手动装载：  
　　nf\_conntrack\_ftp  
  
追踪到的连接：  
　　/proc/net/nf\_conntrack  
  
调整可记录的连接数量最大值：  
　　/proc/sys/net/nf\_conntrack\_max  
  
超时时长：  
　　/proc/sys/net/netfilter/\*timeout\*

iptables -F：清空"fliter"表中的所有的规则；

注意：-F：表示清空指定表中的所有链中的规则；

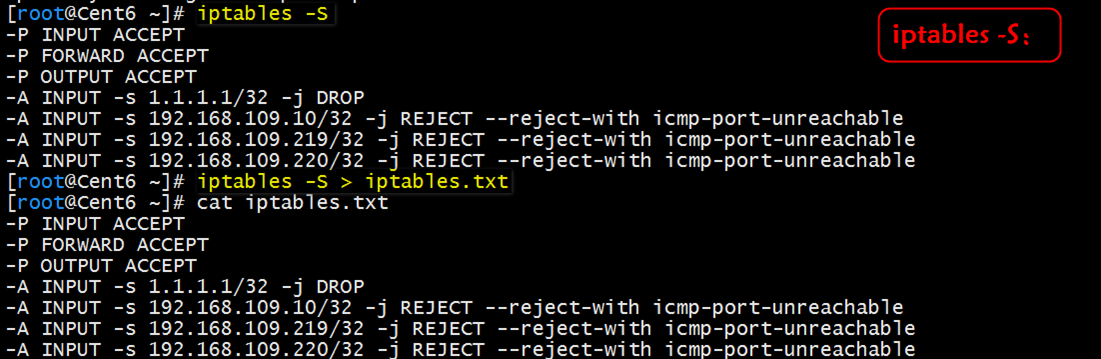
日常例子：

iptables -I INPUT -s 1.1.1.1 -j DROP：在"fliter"表中的INPUT链中，链首插入一条规则；

****iptables -R INPUT 1 -s 2.2.2.2 -j REJECT****：表示替换第一条规则；

****注意：****-R，replace，替换指定的规则；

****iptables -S****：以iptables-save命令格式显示fliter链上的规则；



****iptables -A INPUT -s 192.168.109.10 -d 192.168.109.4 -j REJECT****：拒绝来自192.168.109.10的主机访问192.168.109.4的ip地址；

iptables -A INPUT -s 192.168.109.10 -d 192.168.109.4,192.168.109.5 -j REJECT：拒绝192.168.109.10的主机访问192.168.109.4，192.168.109.5这两个ip地址。

注意：一条命令添加两条规则；

注意：源地址也可以使用这种方式，快速添加规则；

注意：该命令在规则中的语义的含义，-s、-d属于并且的关系，即：拒绝源是192.168.109.10的ip，且源是192.168.109.4的访问；而192.168.109.4和192.168.109.5在逻辑上属于或者的关系。

iptables -A INPUT -s 192.168.109.10 -p tcp --dport 22 -j REJECT：拒绝192.168.109.10主机使用tcp协议访问我的所有ip的22端口。

注意：-p：指定协议；可以通过"/etc/protocols"文件查看协议类型；

注意：--dport：指定目标主机的端口号；（拒绝别人链接我。）

注意：--sport：指定源主机的端口号；（拒绝我链接别人。）

注意：此处，注意区分dport和sport；

iptables -A INPUT -s 192.168.109.10 -p icmp -j REJECT：拒绝来自192.168.109.10主机发来的ping包。

注意：icmp协议没有端口号，即不属于tcp也不属于udp。由于icmp协议的特性，这条命令的结果会造成，192.168.109.10无法ping通自己，自己也无法ping通192.168.109.10主机。

注意：如果要想实现别人无法ping通自己，自己可以ping通别人，需要根据state来判断。0表示回应包，8表示请求包。

iptables -A INPUT -s 192.168.109.10 -i eth0 -j REJECT：拒绝192.168.109.10主机访问我的eth0网卡。

注意：-i：加上网卡名，表示报文流入的接口；

注意：-o：加上网卡名，表示报文流出的接口；

iptables -A INPUT ! -s 192.168.109.4 -p tcp --dport 80 -j REJECT：除了192.168.109.4的主机可以访问我的tcp的80端口，其他的主机全部拒绝；

****注意：****！：表示"非"；一般情况下，很少使用该方式。

iptables -A INPUT -s 192.168.109.4 -p icmp --icmp-type 8 -j REJECT：拒绝来自192.168.109.10主机发送来的icmp的请求包。即192.168.109.10主机无法ping通我，我可以ping通它。

注意：icmp协议中，ping通需要满足两个条件，首先源主机向目标主机发送一个请求包(8)，目标主机收到之后，便会返回一个回应包(0)。

注意：-p：指定协议；

注意：--icmp-type：指定ping包的类型。（code可以省略；）

iptables -A INPUT -s 192.168.109.10 -p tcp -m multiport --dports 20:22,80 -j REJECT：拒绝来自192.168.109.10主机发送来的访问我的20-22,80端口的包。

注意：-m：后面跟模块名，multiport：以离散形式表示的多端口（最多可以跟15个端口）；

注意：--dports：目标端口是多个端口；

注意：使用samba服务测试多端口的话，可能存在问题。想要抓包测试的话，可以使用"tcpdump -i eth0 -nn host 192.168.109.4"；

iptables -A INPUT -m iprange --src-range 192.168.109.1-192.168.109.10 -j REJECT：拒绝192.168.109.1~192.168.109.10范围内的所有主机的访问。

注意：iprange：ip范围的模块；

注意：--src-range：源地址范围；

注意：--dsc-range：目标地址的范围；

iptables -A INPUT -p tcp --dport 22 -j ACCEPT

iptables -A OUTPUT -p tcp --sport 22 -j ACCEPT

针对这2条命令进行一些讲解吧

-A 参数就看成是添加一条 INPUT 的规则

-p 指定是什么协议 我们常用的tcp 协议，当然也有udp 例如53端口的DNS

到时我们要配置DNS用到53端口 大家就会发现使用udp协议的

而 --dport 就是目标端口 当数据从外部进入服务器为目标端口

反之 数据从服务器出去 则为数据源端口 使用 --sport

4、如何删除规则

首先我们要知道 这条规则的编号，每条规则都有一个编号

通过 iptables -L -n --line-number 可以显示规则和相对应的编号

num target prot opt source destination

1 DROP tcp – 0.0.0.0/0 0.0.0.0/0 tcp dpt:3306

2 DROP tcp – 0.0.0.0/0 0.0.0.0/0 tcp dpt:21

3 DROP tcp – 0.0.0.0/0 0.0.0.0/0 tcp dpt:80

多了 num 这一列， 这样我们就可以 看到刚才的规则对应的是 编号2

那么我们就可以进行删除了

iptables -D INPUT 2

删除INPUT链编号为2的规则。

7、iptables对ftp的设置

现在我开始对ftp端口的设置，按照我们以前的视频，添加需要开放的端口

ftp连接端口有2个 21 和 20 端口，我现在添加对应的规则。

[root@localhost rootiptables -A INPUT -p tcp --dport 21 -j ACCEPT

[root@localhost rootiptables -A INPUT -p tcp --dport 20 -j ACCEPT

[root@localhost rootiptables -A OUTPUT -p tcp --sport 21 -j ACCEPT

[root@localhost rootiptables -A OUTPUT -p tcp --sport 20 -j ACCEPT

好，这样就添加完了，我们用浏览器访问一下ftp,出现超时。

所以我刚才说 ftp 是比较特殊的端口，它还有一些端口是 数据传输端口，

例如目录列表， 上传 ，下载 文件都要用到这些端口。

而这些端口是 任意 端口。。。 这个 任意 真的比较特殊。

如果不指定什么一个端口范围， iptables 很难对任意端口开放的，

如果iptables允许任意端口访问， 那和不设置防火墙没什么区别，所以不现实的。

那么我们的解决办法就是 指定这个数据传输端口的一个范围。

下面我们修改一下ftp配置文件。

我这里使用vsftpd来修改演示，其他ftp我不知道哪里修改，大家可以找找资料。

[root@localhost rootvi /etc/vsftpd.conf

在配置文件的最下面 加入

pasv\_min\_port=30001

pasv\_max\_port=31000

然后保存退出。

这两句话的意思告诉vsftpd, 要传输数据的端口范围就在30001到31000 这个范围内传送。

这样我们使用 iptables 就好办多了，我们就打开 30001到31000 这些端口。

[root@localhost root#iptables -A INPUT -p tcp --dport 30001:31000 -j ACCEPT

[root@localhost root#iptables -A OUTPUT -p tcp --sport 30001:31000 -j ACCEPT

[root@localhost rootservice iptables save

最后进行保存， 然后我们再用浏览器范围下 ftp。可以正常访问

用个账号登陆上去，也没有问题，上传一些文件上去看看。

看到了吧，上传和下载都正常。。 再查看下 iptables 的设置

[root@localhost rootiptables -L -n

Chain INPUT (policy DROP)

target prot opt source destination

ACCEPT tcp – 0.0.0.0/0 0.0.0.0/0 tcp dpt:22

ACCEPT tcp – 0.0.0.0/0 0.0.0.0/0 tcp dpt:21

ACCEPT tcp – 0.0.0.0/0 0.0.0.0/0 tcp dpt:20

ACCEPT tcp – 0.0.0.0/0 0.0.0.0/0 tcp dpts:30001:31000

Chain FORWARD (policy DROP)

target prot opt source destination

Chain OUTPUT (policy DROP)

target prot opt source destination

ACCEPT tcp – 0.0.0.0/0 0.0.0.0/0 tcp spt:22

ACCEPT tcp – 0.0.0.0/0 0.0.0.0/0 tcp spt:21

ACCEPT tcp – 0.0.0.0/0 0.0.0.0/0 tcp spt:20

ACCEPT tcp – 0.0.0.0/0 0.0.0.0/0 tcp spts:30001:31000