一、iptables 简介

P1 iptables 基本用法

管理程序位置

✓ /sbin/iptables

指令组成

✓ iptables [-t 表名] 选项 [链名] [条件] [-j 目标操作]

环境准备

主机名	Ip 地址	角色
Node1	192.168.2.100	客户端
Node2	192.168.2.200	服务端

禁止 node2 主机 ping node1 主机

关闭 firewalld 防火墙

```
[root@node1 ~]# systemctl stop firewalld
[root@node1 ~]# systemctl disable firewalld
```

安装 iptables 防火墙，开启防火墙，并设置为开机自启动

```
[root@node1 ~]# yum -y install iptables-services
[root@node1 ~]# systemctl start iptables
[root@node1 ~]# systemctl enable iptables
```

node2 进行 ping 验证

```
[root@node2 ~]# ping   192.168.2.100                #可以成功
```

在 node1 上插入防火墙规则，拒绝其他主机通过 icmp 协议 ping 本机，立刻生效

```
[root@node1 ~]# iptables -t filter -I INPUT -p icmp -j REJECT          #拒绝访问，插入规则时，默认会将规则插入到链的最顶端
[root@node2 ~]# ping 192.168.2.100      #ping不通，但是有信息返回，目标主机不可达
[root@node1 ~]# iptables -t filter -I INPUT -p icmp -j DROP
[root@node2 ~]# ping  192.168.2.100      # node2重新ping主机node1，没有任何回应
```

二、iptables 基础

P1 iptables 用法解析

注意事项/整体规律

- ✓ 可以不指定表，默认为filter表
- ✓ 可以不指定链，默认为对应表的所有链
- ✓ 如果没有匹配的规则，则使用防火墙默认规则
- ✓ 选项/链名/目标操作用大写字母，其余都小写

ACCEPT：允许通过/放行

DROP：直接丢弃，不给出任何回应

REJECT：拒绝通过，必要时会给出提示

LOG：记录日志，然后传给下一条规则

“匹配即停止”规律的唯一例外

node1 清空规则，重新设定规则记录所有 ping 本机的操作

```
[root@node1 ~]# iptables -F                                  #清空所有的防火墙规则
[root@node1 ~]# iptables -t filter -I INPUT -p icmp -j LOG      #在filter表INPUT链中插入一条规则
[root@node2 ~]# ping  192.168.2.100                          #可以ping通
```

日志记录在/var/log/messages，node1上查看日志信息，动态更新日志

```
[root@node1 ~]# tailf /var/log/messages
```

常用的管理选项

类别	选项	用途
添加规则	-A	在链的末尾追加一条规则
	-I	在链的开头（或指定序号）插入一条规则
查看规则	-L	列出所有的规则条目
	-n	以数字形式显示地址、端口等信息
	--line-numbers	查看规则时，显示规则的序号
删除规则	-D	删除链内指定序号（或内容）的一条规则
	-F	清空所有的规则
默认策略	-P	为指定的链设置默认规则

添加新的规则 -A 追加 -I 插入

```
[root@node1 ~]#  iptables -nL          #查看现有的规则，没有指定表，默认查看filter表
```

-A 指向 INPUT 链的最后一行追加一条规则

```
[root@node1 ~]# iptables -t filter -A INPUT -p tcp -j ACCEPT     #-p 指进入本机的数据包，是通过tcp协议进入的， -j 指对数据包的操作，ACCEPT 允许通过
[root@node1 ~]#  iptables  -nL
```

-I 将规则插入到 INPUT 链的最前面

```
[root@node1 ~]#  iptables -I INPUT -p udp -j ACCEPT
[root@node1 ~]#  iptables  -nL
```

在 filter 表的 INPUT 链中的，第二条规则的前面插入一条规则

```
[root@node1 ~]#  iptables -I INPUT 2 -p udp -j ACCEPT
[root@node1 ~]#  iptables  -nL
```

查看规则，显示行号

```
[root@node1 ~]# iptables -nL --line-numbers
```

-D 删除， -F 清空规则

```
[root@node1 ~]# iptables -D INPUT 3   #清除防火墙filter表中，INPUT链中的第三条规则
```

```
[root@node1 ~]# iptables -nL
[root@node1 ~]# iptables -F                #清空filter过滤表中防火墙规则，filter是默认表
[root@node1 ~]# iptables -t nat -F         #清空nat表中的所有规则，【地址转换表】
[root@node1 ~]# iptables -t mangle -F      #清空mangle表中的所有规则,【包标记表】
[root@node1 ~]# iptables -t raw -F         #清空raw表中的所有规则，【状态跟踪表】
```

设置默认规则，所有链的初始默认规则均为 ACCEPT，通过-P 选项可重置默认规则 ACCEPT 或者 DROP，默认规则只允许设置 ACCEPT 或 DROP

```
[root@node1 ~]# iptables -nL               #数据包入站，转发，出站默认规则为ACCEPT
[root@node1 ~]# iptables -P FORWARD DROP   #将FORWARD的默认规则设置为DROP，丢弃所有数据包
[root@node1 ~]# iptables -nL
[root@node1 ~]# iptables -P FORWARD ACCEPT
[root@node1 ~]# iptables -nL
```

三、防火墙匹配条件

P1 iptables 用法解析

类别	选项	用法
通用匹配	协议匹配	-p 协议名
	地址匹配	-s 源地址、-d 目标地址
	接口匹配	-i 收数据的网卡、-o 发数据的网卡
隐含匹配	端口匹配	--sport 源端口、--dport 目标端口
	ICMP类型匹配	--icmp-type ICMP类型

需要取反条件时，用叹号！

过滤规则示例

限制特定 IP 或网段的访问

```
[root@node1 ~]# iptables -A INPUT -s 192.168.4.120 -j DROP    #filter表中添加规则，-s 丢弃所有从4.120主机发过来的数据包（-s按照源地址）
[root@node1 ~]# iptables -A INPUT -s 192.168.4.0/24 -j DROP   #filter表中添加规则，-s 丢弃所有从4.0网段的主机发过来的数据包 （按照源地址）
[root@node1 ~]# iptables -nL        #查看防火墙规则，默认查看的是filter过滤表
```

保护特定网络服务

#--dport 端口必须和协议组合起来使用，单独使用报错 ， -p 协议可以单独使用，也可以和端口组合起来使用

```
[root@node1 ~]#  iptables -A INPUT -s 192.168.2.254 -p tcp --dport 22 -j ACCEPT
#-s当源地址192.168.2.254; -p 通过tcp协议；--dport 访问本机目标端口22；允许通过
[root@node1 ~]#  iptables -A INPUT -s 192.168.2.200 -p tcp --dport 22 -j DROP
        #-s当源地址192.168.2.200; -p 通过tcp协议；--dport 访问本机目标端口22；拒绝
测试：2.200主机远程2.100失败
[root@node2 ~]# ssh root@192.168.2.100
```

禁 ping 相关策略处理

ping的通信流程（A ping B）


```
[root@node2 ~]# iptables -F                    #清空filter表的所有规则
```

插入一条规则：想要实现禁止其他所有主机通过 icmp 协议 ping 本机，但本机可以 ping 通其他主机

```
[root@node2 ~]# iptables -I INPUT -p icmp -j REJECT
```

测试验证： node2 主机 ping 主机 node1 时，无法 ping 通；node1 主机 ping 主机 node2 时，也无法 ping 通，目标主机不可达

ping 其他主机时，数据包可以到达其他主机，其他主机在回复数据时也是通过 icmp 协议，被防火墙所阻挡， 导致通信失败

禁止 ICMP 协议里的 echo-request

从进站的角度，设置规则，本机可以 ping 其他主机，其他主机不可以 ping 本机

```
[root@node1 ~]# iptables -F                    #清空filter表的所有防火墙策略
[root@node1 ~]# iptables -A INPUT -p icmp --icmp-type echo-request -j DROP
        #其他主机ping本机时，通过icmp协议，发送过来的请求 echo-request，直接丢弃
[root@node1 ~]# iptables -A INPUT -p icmp ! --icmp-type echo-request -j ACCEPT
        #本机ping其他主机时，其他主机返回的数据包类型不是echo-request的，都接受
（也可以不用敲，默认是允许）
```

测试

```
[root@node1 ~]# ping  192.168.2.200        #node1 ping 主机node2，成功
[root@node2 ~]# ping  192.168.2.100        # node2 ping 主机node1，无法ping通
```

从出站的角度，设置规则，本机可以 ping 其他主机，其他主机不可以 ping 本机

```
[root@node1 ~]# iptables -F                    #清空filter表的所有防火墙策略
[root@node1 ~]# iptables -A OUTPUT -p icmp --icmp-type echo-request -j ACCEPT
        #本机ping其他主机时，发送数据包的类型为 echo-request，防火墙通过，可以发送出去（也可以不用敲，默认是允许）
[root@node1 ~]# iptables -A OUTPUT -p icmp ! --icmp-type echo-request -j DROP
        #本机发送的数据包不是 echo-request, 则代表的是数据的回应，直接丢弃该数据包
```

测试

```
[root@node1 ~]# ping  192.168.2.200      # node1 ping 主机node2，成功
[root@node2 ~]# ping 192.168.2.100       #node2 ping 主机node1，无法ping通
```

帮助信息查询

```
[root@node2 ~]# iptables -p icmp --help
```

四、主机型/网络型防火墙

P1 防火墙防护类型

环境准备

主机名要求	网卡、IP地址以及网关设置要求
client	eth0:192.168.4.100 网关：192.168.4.5
proxy	eth0:192.168.4.5 eth1:192.168.2.5
web1	eth1:192.168.2.100 网关：192.168.2.5

备注：准备环境时，可以先把所有防火墙规则情况iptables -F

环境准备由于之前已经做过很多次，本次不在写详细的步骤

proxy 主机开启路由转发功能

```
[root@proxy ~]# vim /etc/sysctl.conf
net.ipv4.ip_forward = 1
[root@proxy ~]# sysctl -p
```

client 客户端需要配置网关

```
[root@client ~]# nmcli connection modify ens33 ipv4.method manual ipv4.gateway
   192.168.4.5  connection.autoconnect yes
[root@client ~]# nmcli connection up ens33
```

web1 主机需要配置网关

```
[root@web1 ~]# nmcli connection modify ens33 ipv4.method manual ipv4.gateway
192.168.2.5  connection.autoconnect yes
[root@web1 ~]# nmcli connection up ens33
```

测试：4.100 的虚拟机 client 可以 ping 通 2.100 的虚拟机 web1

```
[root@client ~]# ping  192.168.2.100
```

2.100 的虚拟机 web1 可以 ping 通 4.100 的虚拟机 client

```
[root@web1 ~]# ping  192.168.4.100
```

web1 配置 web 服务

```
[root@web1 ~]# yum -y install httpd
[root@web1 ~]# echo "test page" > /var/www/html/index.html
[root@web1 ~]# systemctl restart httpd
[root@web1 ~]# setenforce 0                    #关闭selinux
```

为了方便后续的操作，先清空三台主机的防火墙规则

```
[root@client ~]# iptables -F
[root@proxy ~]# iptables -F
[root@web1 ~]# iptables -F
```

没有防火墙的情况下测试连接 web 服务

```
[root@client ~]# curl http://192.168.2.100
test page
```

配置防火墙规则，禁止 2.100 访问 web1 的 80 端口

web1 主机设置防火墙，INPUT 链，进站访问

```
[root@web1 ~]#  iptables -I INPUT -s 192.168.4.100 -p tcp --dport 80 -j REJECT
```

client 测试：

```
[root@client ~]# curl http://192.168.2.100                    #访问失败
```

proxy 主机设置防火墙规则，FORARD 链，可以针对路由转发的数据进行限制

```
[root@web1 ~]# iptables -F
[root@client ~]# curl http://192.168.2.100
[root@proxy ~]# iptables -I FORWARD -s 192.168.4.100 -p tcp --dport 80 -j REJECT
```

client 测试：

```
[root@client ~]# curl http://192.168.2.100          #访问失败
```

其他服务不受影响，可以正常 ping 通

```
[root@client ~]# ping  192.168.2.100
```

client 也可以 ssh 远程 web1

```
[root@client ~]# ssh root@192.168.2.100
```

此时，若不想其他的协议端口访问，可以直接拒接从源地址发来的所有数据

```
[root@proxy ~]# iptables -I FORWARD -s 192.168.4.100  -j  REJECT
```

五、扩展匹配规则

P1 概述

扩展条件的方法

前提条件

✓ 有对应的防火墙模块支持

基本用法

✓ -m 扩展模块 --扩展条件 条件值

✓ 示例： -m mac --mac-source 00:0C:29:74:BE:21

类别	选项	用法
扩展匹配	MAC地址匹配	-m mac --mac-source MAC地址
	多端口匹配	-m multiport --sports 源端口列表 -m multiport --dports 目标端口列表
	IP范围匹配	-m iprange --src-range IP1-IP2 -m iprange --dst-range IP1-IP2

根据 MAC 地址封锁主机，在之前创建好的 proxy 主机和 web1 主机上完成

此实验目的： 服务器通过 IP 地址定义了防火墙规则，拒绝客户端发送过来的所有数据包,当客户端修改 IP 地址以后， 可以继续和服务器通信。

清空所有防火墙规则

```
[root@proxy ~]# iptables -F
[root@proxy ~]# iptables -A INPUT -s 192.168.2.100 -j REJECT          #编写防火墙规则，拒绝2.100发送的所有数据包
```

web1 访问测试

```
[root@web1 ~]# ping 192.168.2.5          #ping测试失败
[root@web1 ~]# ssh root@192.168.2.5          #ssh连接被拒绝
```

web1 修改 IP 地址为 192.168.2.110

```
[root@web1 ~]# nmcli connection modify ens33 ipv4.method manual ipv4.addresses 192.168.2.110/24 connection.autoconnect yes
```

```
[root@web1 ~]# nmcli connection up ens33
```

重新访问测试

```
[root@web1 ~]# ping 192.168.2.5                    #可以ping通proxy虚拟机
[root@web1 ~]# ssh 192.168.2.5                     #可以ssh远程连接
```

根据 MAC 地址封锁主机

```
[root@web1 ~]# ip address show ens33               #查看主机web1的mac地址
.....
link/ether 00:0c:29:2b:66:25
```

清空防火墙规则，添加 mac 地址，拒绝 web1 发送过来的所有数据包

```
[root@proxy ~]# iptables -I INPUT -m mac --mac-source 00:0c:29:98:8f:b3 -j  REJECT
```

web1 测试，所有数据包被拒绝通过

```
[root@web1 ~]# ping 192.168.2.5        #失败
[root@web1 ~]# ssh 192.168.2.5             #失败
```

测试再把 IP 改回原来的 2.100，也是 ping 不通的

多端口案例

实现一条规则开放多个端口

```
[root@proxy ~]# iptables -F          #清空所有防火墙规则
[root@proxy ~]# iptables -I INPUT -p tcp -m multiport –dport        \
10:20,25,110,22,200:300 -j ACCEPT    #允许数据通过多个端口和本机通信，对于连续的端口使用10:20表示【代表10-20】，对于不连续的端口用逗号【,】作为分隔符
[root@proxy ~]# iptables -nL INPUT                  #查看INPUT链的防火墙规则
```

根据 IP 范围封锁主机

```
[root@proxy ~]# iptables -F
```

```
[root@proxy ~]# iptables -A INPUT -p tcp --dport 22 -m iprange --src-range
192.168.4.200-192.168.4.254  -j  ACCEPT          # 允许192.168.4.200-192.168.4.254
通过tcp协议访问本机的22端口
[root@proxy ~]# iptables -A INPUT -p tcp --dport 22 -s 192.168.4.0/24 -j REJECT
        #其他主机通过tcp协议访问本机的22端口时，如果请求访问的主机IP地址在4.0网段，
则拒绝访问
[root@proxy ~]# iptables -nL INPUT        #查看INPUT进站流量的防火墙规则
```

iptables 永久防火墙规则设定，将防火墙规则写入配置文件，安装 iptables 永久防火墙规则软件包，启动服务，设置为开机自启动

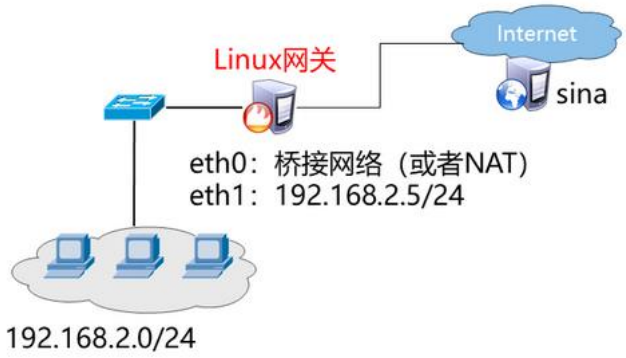
```
[root@proxy ~]# yum -y install iptables-services
[root@proxy ~]# systemctl start iptables
[root@proxy ~]# systemctl enable iptables
[root@proxy ~]# iptables-save > /etc/sysconfig/iptables #此命令会将临时的防火墙规
则，写入配置文件中
```

六、NAT 应用案例

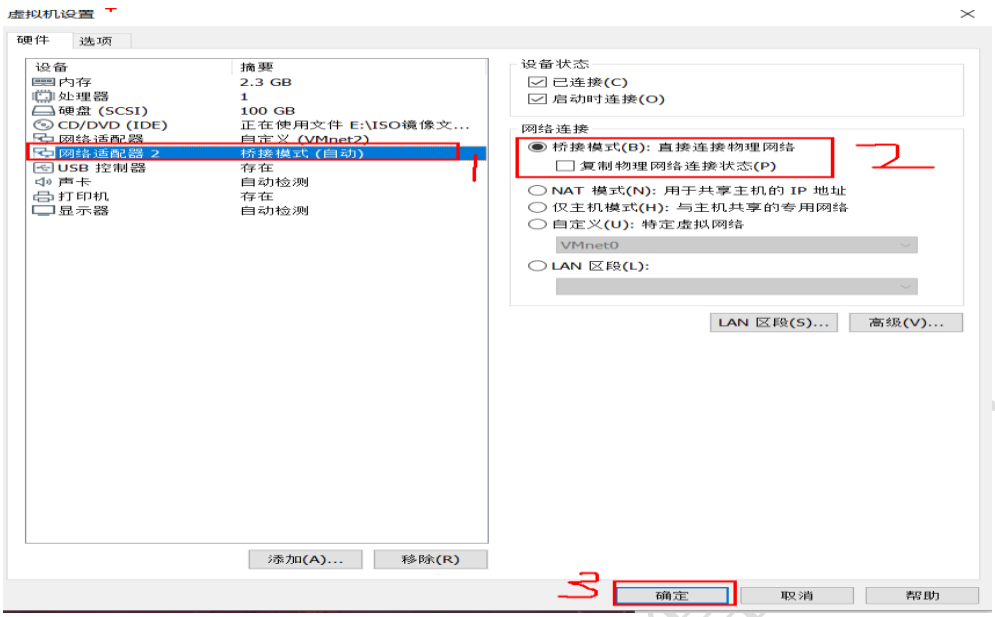
P1 案例环境

局域网共享公网 IP 上网

环境准备，使用 proxy，web1 操作



修改虚拟机 proxy 中 4.0 网段的虚拟网卡为桥接模式



重启虚拟机 proxy，将 4.0 网段的网卡改为 DHCP 模式

```
[root@proxy ~]# nmcli connection modify ens33 ipv4.method auto
connection.autoconnect yes          #修改网卡ens33为auto，自动分配IP模式
[root@proxy ~]# nmcli connection up ens33          #激活网卡
[root@proxy ~]# ifconfig ens33 | head -2          #查看配置后网卡的IP地址
[root@proxy ~]# ping www.baidu.com          #ping百度，检查是否可以连通外网
```

内网服务器配置： 虚拟机 web1

```
[root@web1 ~]# echo "nameserver 8.8.8.8" >> /etc/resolv.conf  #配置DNS服务器的地址
[root@web1 ~]# ip route show
[root@web1 ~]# ping www.baidu.com                    #此时web1还不能上公网
```

配置 SNAT 共享上网

```
[root@proxy ~]# iptables -F                    #清空防火墙规则,默认表为filter
[root@proxy ~]# iptables -t nat -A POSTROUTING -s 192.168.2.0/24 -j SNAT --to-source 10.1.10.74
```

```
# POSTROUTING(路由后转换) ——> 这里必须写路由后转换，对于2.0网段的主机做地址转换，将其发送的所有数据包中的源IP地址转换成10.1.10.74
[root@web1 ~]# ping www.baidu.com       #虚拟机web1测试,可以连通外网，Ctrl + C 结束
```

地址伪装策略，当公网地址不确定时，使用 -j MASQUERADE 可以自动识别公网 IP

```
[root@proxy ~]# iptables -t nat -F              #清空nat表中的所有规则
[root@proxy ~]# iptables -t nat -nL                     #查看nat表中的规则
[root@proxy ~]# iptables -t nat -A POSTROUTING -s 192.168.2.0/24 -j MASQUERADE
[root@web1 ~]# ping www.baidu.com       #虚拟机web1测试,可以连通外网，Ctrl + C 结束
```