|  |
| --- |
| iptables防火墙精讲 |
|  |
| **快捷键：**  Ctrl + 1 标题1  Ctrl + 2 标题2  Ctrl + 3 标题3  Ctrl + 4 实例  Ctrl + 5 程序代码  Ctrl + 6 正文 |
| **格式说明：**  蓝色字体：注释  黄色背景：重要  绿色背景：注意 |

目 录

[1.1 iptable防火墙 1](#_Toc480150036)

[1.2 学好iptables的基础 1](#_Toc480150037)

[1.3 iptables防火墙简介 1](#_Toc480150038)

[1.4 iptables名词和术语 1](#_Toc480150039)

[1.4.1 什么是容器 1](#_Toc480150040)

[1.5 iptables工作流程 2](#_Toc480150041)

[1.6 iptables表(tables)和链(chains) 2](#_Toc480150042)

[1.6.1 filter表 3](#_Toc480150043)

[1.6.2 nat表 3](#_Toc480150044)

[1.6.3 Mangle表 4](#_Toc480150045)

[1.6.4 man iptables 4](#_Toc480150046)

[1.7 iptables表和链工作的流程图 4](#_Toc480150047)

[1.8 iptables命令帮助信息 5](#_Toc480150048)

[1.9 实际测试iptables规则 7](#_Toc480150049)

[1.9.1 启动和查看iptables状态 8](#_Toc480150050)

[1.9.2 iptables默认加载的内核模块 8](#_Toc480150051)

[1.9.3 加载iptables需要的模块 8](#_Toc480150052)

[1.9.4 再次查看加载的模块 9](#_Toc480150053)

[1.9.5 清除默认规则 9](#_Toc480150054)

[1.9.6 禁用规则 10](#_Toc480150055)

[1.9.7 查看规则的编号 12](#_Toc480150056)

[1.9.8 在指定位置插入规则 12](#_Toc480150057)

[1.9.9 在指定位置删除规则 12](#_Toc480150058)

[1.9.10 小结:总结删除规则的方法 12](#_Toc480150059)

[1.9.11 禁止10.0.0.0 网段连入 12](#_Toc480150060)

[1.9.12 取反匹配规则 12](#_Toc480150061)

[1.9.13 封堵ssh端口 13](#_Toc480150062)

[1.9.14 企业工作中解决这个问题 13](#_Toc480150063)

[1.9.15 封掉3306端口 13](#_Toc480150064)

[1.9.16 匹配指定的协议 13](#_Toc480150065)

[1.9.17 匹配指定协议外的所有协议 13](#_Toc480150066)

[1.9.18 匹配主机源IP , 目的IP 14](#_Toc480150067)

[1.9.19 匹配网段 14](#_Toc480150068)

[1.9.20 匹配单一端口 14](#_Toc480150069)

[1.9.21 匹配指定端口之外的端口 14](#_Toc480150070)

[1.9.22 匹配端口范围 14](#_Toc480150071)

[1.9.23 匹配ICMP类型 14](#_Toc480150072)

[1.9.24 匹配指定的网络接口 15](#_Toc480150073)

[1.9.25 匹配网络状态 15](#_Toc480150074)

[1.9.26 限定指定时间包的允许通过数量及并发数 16](#_Toc480150075)

[1.10 手动执行iptables命令配置企业生产环境下的防火墙 16](#_Toc480150076)

[1.10.1 清空自定义的规则和链和计数器 16](#_Toc480150077)

[1.10.2 首先设置允许ssh连接 16](#_Toc480150078)

[1.10.3 设置允许本机lo通信规则 17](#_Toc480150079)

[1.10.4 目前为止的配置已经允许本机可以操作 17](#_Toc480150080)

[1.10.5 修改默认规则 17](#_Toc480150081)

[1.10.6 开启信任的IP网段 18](#_Toc480150082)

[1.10.7 允许业务服务端口对外访问(允许http服务无条件通过) 18](#_Toc480150083)

[1.10.8 允许icmp类型协议通过 18](#_Toc480150084)

[1.10.9 允许关联的状态包通过(web服务不要使用FTP服务) 18](#_Toc480150085)

[1.10.10 最终的配置结果 19](#_Toc480150086)

[1.10.11 2种将iptables临时命令保存为配置文件的方法 20](#_Toc480150087)

[1.10.12 查看已保存的配置文件 20](#_Toc480150088)

[1.10.13 生产维护 21](#_Toc480150089)

[1.10.14 手工封IP 21](#_Toc480150090)

[1.10.15 自动封IP 21](#_Toc480150091)

[1.10.16 企业实战题:请用至少两种方法实现: 23](#_Toc480150092)

[第2章 生产环境iptables脚本讲解 25](#_Toc480150093)

[2.1 问题:企业硬件防火墙和iptables防火墙是否要同时用? 27](#_Toc480150094)

[第3章 nat表的核心功能 27](#_Toc480150095)

[3.1 试验环境准备 27](#_Toc480150096)

[3.1.1 服务器网关B需要如下条件 27](#_Toc480150097)

[3.1.2 环境说明 27](#_Toc480150098)

[3.1.3 加载内核参数 27](#_Toc480150099)

[3.2 局域网的机器 28](#_Toc480150100)

[3.3 局域网共享上网项目案例 28](#_Toc480150101)

[3.3.1 方法1:适合于有固定外网地址的: 28](#_Toc480150102)

[3.3.2 方法2:适合变化外网地址(ADSL) 28](#_Toc480150103)

[3.3.3 为什么用POSTROUTING? 28](#_Toc480150104)

[3.3.4 企业上网到底需不需要linux网关 28](#_Toc480150105)

[3.4 把外部IP地址及端口映射到内部服务器地址及端口(共享上网环境一样) 29](#_Toc480150106)

[3.4.1 在10段主机可以通过访问B10.0.0.5:9000, 即可访问到172.16.1.8:80 29](#_Toc480150107)

[3.4.2 C配置WEB服务 29](#_Toc480150108)

[3.5 删除nat规则 29](#_Toc480150109)

[3.6 iptables常用企业案例 29](#_Toc480150110)

[3.7 实现10网段外网IP和内网172网段IP一对一映射 29](#_Toc480150111)

[3.8 跨不同网段IP服务通信问题 30](#_Toc480150112)

[第4章 要实现的生产局域网共享上网案例架构图 30](#_Toc480150113)

[第5章 iptables生产应用场景 30](#_Toc480150114)

[5.1 映射多个外网IP上网 32](#_Toc480150115)

[第6章 iptables 内核优化 33](#_Toc480150116)

[第7章 企业上网的常见接入方式 34](#_Toc480150117)

## iptable防火墙

关闭两项功能:

1. selinux(生产中也是关闭的),ids入侵检测,md5指纹.

selinux生产中也是关闭的，你想开也能开。开selinux的话要为不同的服务写一个selinux允许的服务。Selinux也有漏洞，所以企业一般不开。企业自己去做安全设置，比如IDS入侵检测，或者使用shell脚本对系统的所有的核心文件全部做指纹识别，将来一出现问题，就可以通过指纹识别看到那些文件被改变了。

1. iptables(生产中看情况,内网关闭,外网打开)

一个基本原则就是有外网IP就要有防火墙。大并发的情况,不能开iptables,影响性能,如果并发非常大的情况下就要使用硬件防火墙。尽量做端口映射不做IP映射。

安全优化:

1.尽可能不给服务器配置外网IP.可以通过代理转发或者通过防火墙进行映射。

2.并发不是特别大的情况在外网IP的环境,要开启iptables防火墙.

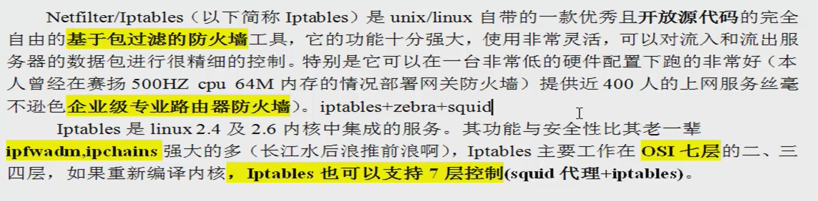
## 学好iptables的基础

1.OSI7层模型以及不同层对应哪些协议?

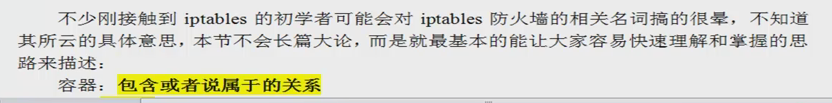
2.TCP/IP三次握手,四次断开的过程,TCP HEADER.

3.常见服务端口要了如指掌.

## iptables防火墙简介



## iptables名词和术语



### 什么是容器

容器是包含和被包含的关系

Iptables是表的容器

Iptables包含四张表

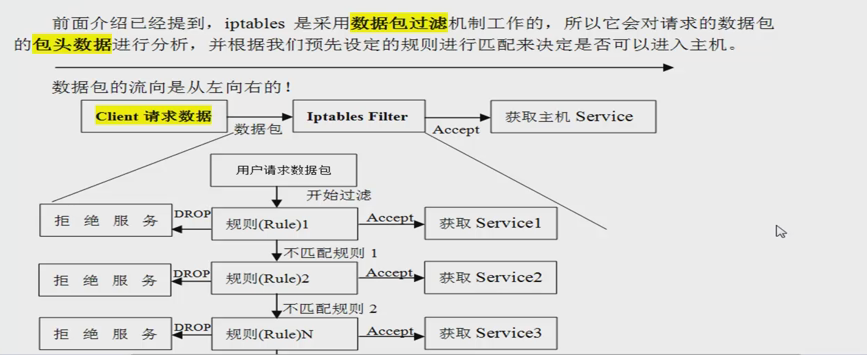
表是链的容器，每个表中都包含若干的链

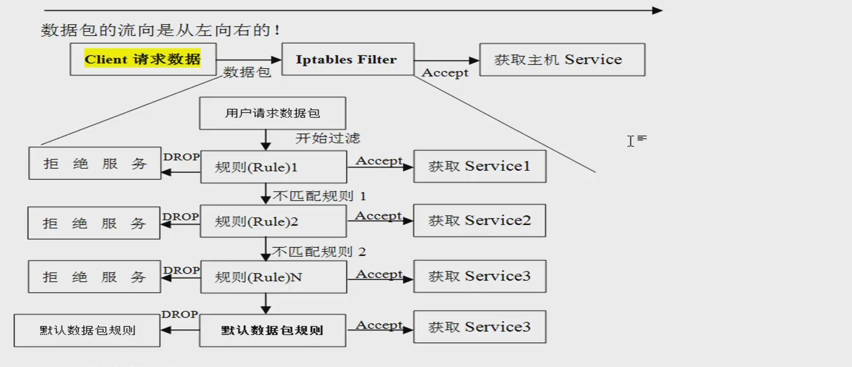
链是规则的容器，真正的过滤规则是属于链里面的

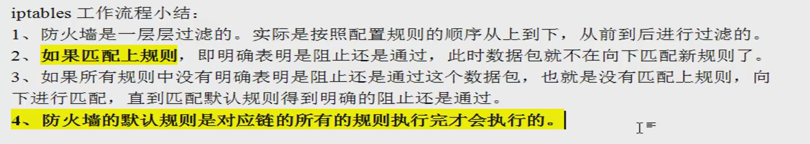
Iptables国家，表是省，链是市，规则相当于县镇

表（table） 链（chain） 规则（policy）

## iptables工作流程







所以要把你最想生效的规则，放在这个规则表的最前面。

## iptables表(tables)和链(chains)



Forward表示流经服务器 PREROUTING POSTROUTING

Filter表，包含三个链INPUT,FORWARD,OUTPUT

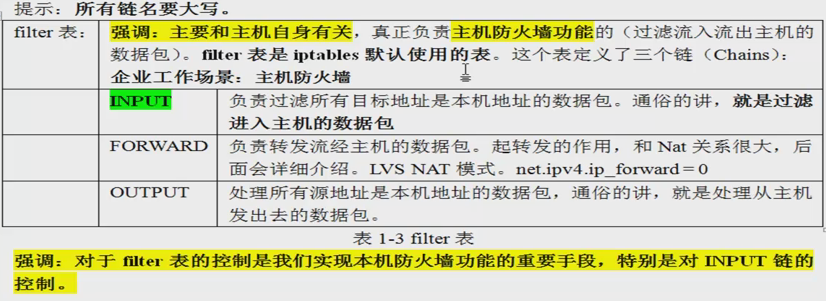
Filter 真正的防火墙功能

Nat 负责数据包改写,网关共享上网,IP和端口映射

Mangle 路由相关,用到的不多

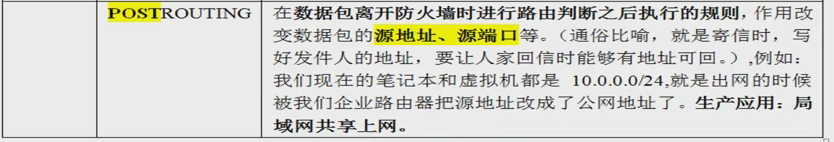
RAW 不常用

### filter表



### nat表

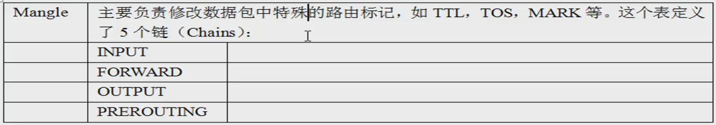




**Mangle表和RAW表使用的相对较少，老师工作10年都不会用到。**

**PREROUTING相当于是先锋部队，要打仗它冲在最前面。**

### Mangle表



### man iptables

filter:

This is the default table (if no -t option is passed). It contains the built-in chains INPUT

(for packets destined to local sockets), FORWARD (for packets being routed through the box),

and OUTPUT (for locally-generated packets).

nat:

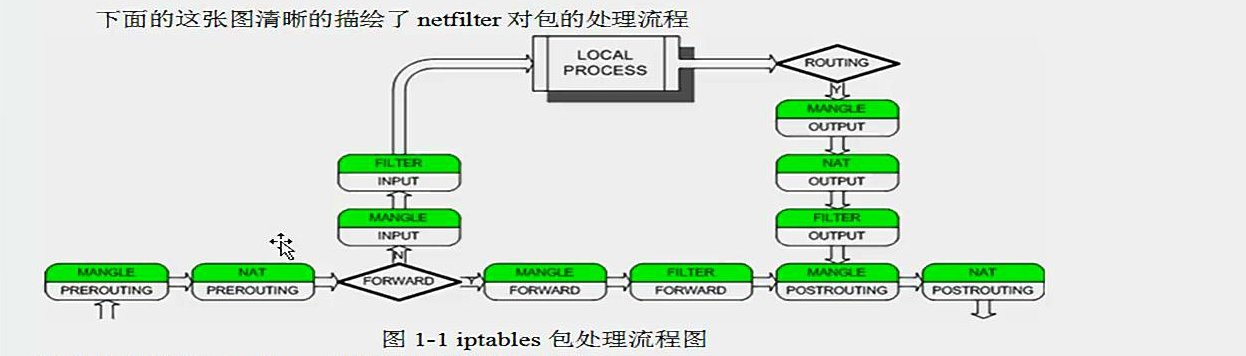
This table is consulted when a packet that creates a new connection is encountered. It con-

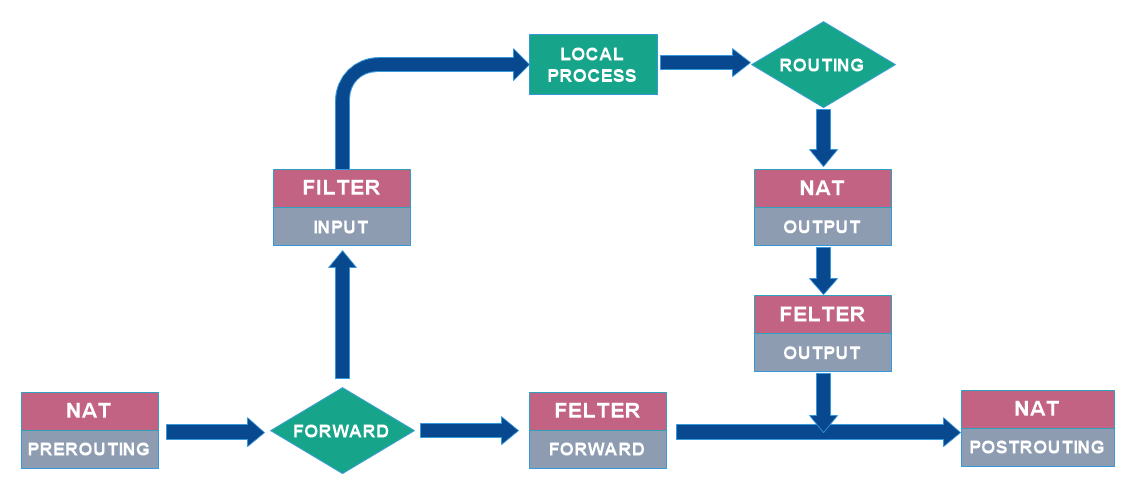
sists of three built-ins: PREROUTING (for altering packets as soon as they come in), OUTPUT

(for altering locally-generated packets before routing), and POSTROUTING (for altering pack-

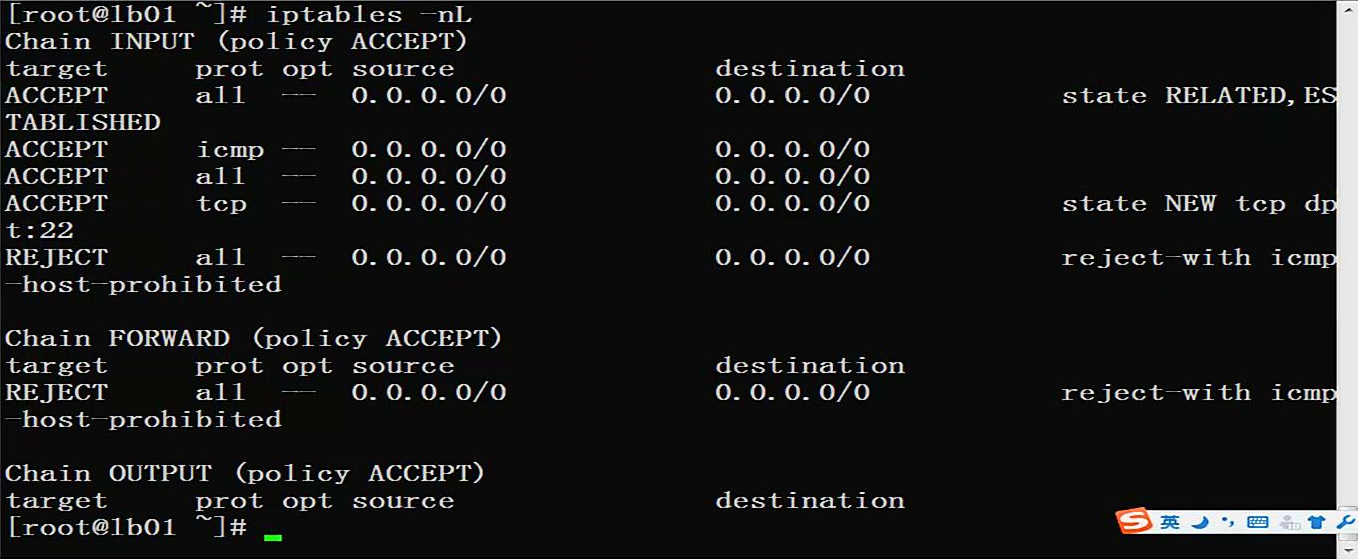
ets as they are about to go out).

## iptables表和链工作的流程图





为了效率我们在工作中只是对进来的数据包进行控制不会对流出去的数据包进行控制，进来的数据包控制住了，那么流出去的数据包就会好很多。



这是一个iptables的默认配置。可以看到在这个配置里面没有指定是哪个表，所以就是Filter表。

然后指定了几个链，Chain INPUT和 Chain FORWARD和Chain OUTPUT这几个链。

其中我们可以看到在Chain INPUT这里写了（policy ACCEPT）就可以知道这个默认的规则是通过。

也就是说下面的详细的配置如果都没有匹配上，那么就走这条默认的。

还有一个以前经常用到的lsof -i:8080查看一个端口是否被占用的命令

## iptables命令帮助信息

[root@nginx-8 ~]# iptables -V

iptables v1.4.7

[root@nginx-8 ~]# rpm -qa iptables

iptables-1.4.7-16.el6.x86\_64

[root@nginx-8 ~]# iptables -h

iptables v1.4.7

Usage: iptables -[ACD] chain rule-specification [options]

iptables -I chain [rulenum] rule-specification [options]

iptables -R chain rulenum rule-specification [options]

iptables -D chain rulenum [options]

iptables -[LS] [chain [rulenum]] [options]

iptables -[FZ] [chain] [options]

iptables -[NX] chain

iptables -E old-chain-name new-chain-name

iptables -P chain target [options]

iptables -h (print this help information)

Commands:

Either long or short options are allowed.

--append -A chain Append to chain

--check -C chain Check for the existence of a rule

--delete -D chain Delete matching rule from chain

--delete -D chain rulenum

Delete rule rulenum (1 = first) from chain

--insert -I chain [rulenum]

Insert in chain as rulenum (default 1=first)

--replace -R chain rulenum

Replace rule rulenum (1 = first) in chain

--list -L [chain [rulenum]]

List the rules in a chain or all chains

--list-rules -S [chain [rulenum]]

Print the rules in a chain or all chains

--flush -F [chain] Delete all rules in chain or all chains

--zero -Z [chain [rulenum]]

Zero counters in chain or all chains

--new -N chain Create a new user-defined chain

--delete-chain

-X [chain] Delete a user-defined chain

--policy -P chain target

Change policy on chain to target

--rename-chain

-E old-chain new-chain

Change chain name, (moving any references)

Options:

[!] --proto -p proto protocol: by number or name, eg. `tcp'

[!] --source -s address[/mask][...]

source specification

[!] --destination -d address[/mask][...]

destination specification

[!] --in-interface -i input name[+]

network interface name ([+] for wildcard)

--jump -j target

target for rule (may load target extension)

--goto -g chain

jump to chain with no return

--match -m match

extended match (may load extension)

--numeric -n numeric output of addresses and ports

[!] --out-interface -o output name[+]

network interface name ([+] for wildcard)

--table -t table table to manipulate (default: `filter')

--verbose -v verbose mode

--line-numbers print line numbers when listing

--exact -x expand numbers (display exact values)

[!] --fragment -f match second or further fragments only

--modprobe=<command> try to insert modules using this command

--set-counters PKTS BYTES set the counter during insert/append

[!] --version -V print package version.

## iptables参数翻译

选项名 功能及特点

-A 在指定链的末尾添加（--append）一条新的规则

-D 删除（--delete）指定链中的某一条规则，按规则序号或内容确定要删除的规则

-I 在指定链中插入（--insert）一条新的规则，若未指定插入位置，则默认在链的开头插入

-R 修改、替换（--replace）指定链中的某一条规则，按规则序号或内容确定要替换的规则

-t 指定表，也可以不指定默认是filter

-L 列出（--list）指定链中的所有的规则进行查看，若未指定链名，则列出表中所有链的内容

-F 清空（--flush）指定链中的所有规则，若未指定链名，则清空表中所有链的内容

-N 新建（--new-chain）一条用户自己定义的规则链

-X 删除指定表中用户自定义的规则链（--delete-chain）

-P 设置指定链的默认策略（--policy）

-p 指定过滤的协议(all.tcp,udp.icmp)默认all

--sport 指定源端口

-s 指定源地址 ！ 取反

-d 目的地址

--dport 指定目标端口（用户请求的端口）

-j 处理的行为 ACCPET接收、DROP丢弃、REJECT拒绝（会给对方拒绝信息，相当于透漏了自己的信息）

-n 使用数字形式（--numeric）显示输出结果，若显示主机的IP地址而不是主机名

-Z 链的计数器清零

-i 匹配进入的网卡接口 （eth0 eth1）

-o 匹配出去的网卡接口 （eth0 eth1）

-m state --state 匹配网络状态

-v 查看规则列表时显示详细（--verbose）的信息

-V 查看iptables命令工具的版本（--Version）信息

-h 查看命令帮助信息（--help）

--line-number 查看规则列表时，同时显示规则在链中的顺序号

-mlimit --limit n/{second/minute/hour}限制限定时间包的允许通过数量及并发数

## 实际测试iptables规则

[root@nginx-8 ~]# /etc/init.d/iptables status 查看iptables状态

[root@nginx-8 ~]# /etc/init.d/iptables start

iptables: No config file. [WARNING]

如果/etc/init.d/iptables启动不了,需要创建iptables文件

[root@nginx-8 ~]# touch /etc/sysconfig/iptables

[root@nginx-8 ~]# /etc/init.d/iptables start

iptables: Applying firewall rules: [ OK ]

或者

[root@nginx-8 ~]# iptables -F

[root@nginx-8 ~]# service iptables save

iptables: Saving firewall rules to /etc/sysconfig/iptables:[ OK ]

[root@nginx-8 ~]# service iptables restart

iptables: Setting chains to policy ACCEPT: filter [ OK ]

iptables: Flushing firewall rules: [ OK ]

iptables: Unloading modules: [ OK ]

iptables: Applying firewall rules:

### 启动和查看iptables状态

iptables -L -n 或 iptables -L -n -v -x

[root@nginx-8 ~]# iptables -L -n

Chain INPUT (policy ACCEPT)

target prot opt source destination

Chain FORWARD (policy ACCEPT)

target prot opt source destination

Chain OUTPUT (policy ACCEPT)

target prot opt source destination

[root@nginx-8 ~]# /etc/init.d/iptables status

Table: filter

Chain INPUT (policy ACCEPT)

num target prot opt source destination

Chain FORWARD (policy ACCEPT)

num target prot opt source destination

Chain OUTPUT (policy ACCEPT)

num target prot opt source destination

### iptables默认加载的内核模块

[root@nginx-8 ~]# lsmod |egrep 'nat|filter'

iptable\_filter 2793 0

ip\_tables 17831 1 iptable\_filte

lsmod (list modules)

lsmod 命令：是一个小程序，用来显示文件、proc/modules的信息，也就是显示当前内核模块装载的模块。

比如iptables不好使了，我们就可以使用这个命令lsmod | egrep “nat|filter”这样就可以查看到我们的内核是否加载了防火墙功能。

### 加载iptables需要的模块

[root@nginx-8 ~]# modprobe ip\_tables

[root@nginx-8 ~]# modprobe iptable\_filter

[root@nginx-8 ~]# modprobe iptable\_nat

[root@nginx-8 ~]# modprobe ip\_conntrack

[root@nginx-8 ~]# modprobe ip\_conntrack\_ftp

[root@nginx-8 ~]# modprobe ip\_nat\_ftp

[root@nginx-8 ~]# modprobe ipt\_state

### 再次查看加载的模块

[root@nginx-8 ~]# lsmod |egrep 'nat|filter' 再次查看加载的模块，发现模块多了，所以前面的命令生效了

nf\_nat\_ftp 3443 0

nf\_conntrack\_ftp 12049 1 nf\_nat\_ftp

iptable\_nat 5923 0

nf\_nat 22676 2 nf\_nat\_ftp,iptable\_nat

nf\_conntrack\_ipv4 9186 3 iptable\_nat,nf\_nat

nf\_conntrack 79537 6 xt\_state,nf\_nat\_ftp,nf\_conntrack\_ftp,iptable\_nat,nf\_nat,nf\_conntrack\_ipv4

iptable\_filter 2793 0

ip\_tables 17831 2 iptable\_nat,iptable\_filter

### 清除默认规则

iptables -F ###清除所有规则,不会处理默认的规则

iptables -X ###删除用户自定义的链

iptables -Z ###链的计数器清零

实例演示2:

[root@nginx-8 ~]# iptables -nL

Chain INPUT (policy ACCEPT) 这个是默认规则，iptables -F不会清理默认规则

target prot opt source destination

Chain FORWARD (policy ACCEPT)

target prot opt source destination

Chain OUTPUT (policy ACCEPT)

target prot opt source destination

[root@nginx-8 ~]# iptables -F

[root@nginx-8 ~]# iptables -X

[root@nginx-8 ~]# iptables -Z

### 禁用规则

禁止ssh端口

(1)找出当前的ssh端口

[root@nginx-8 ~]# netstat -lntup|egrep ssh

tcp 0 0 0.0.0.0:22 0.0.0.0:\* LISTEN 1258/sshd

tcp 0 0 :::22 :::\* LISTEN 1258/sshd

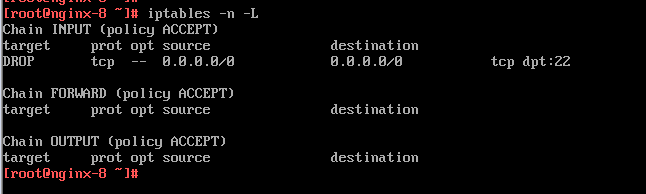
(2)禁止掉当前ssh端口,这里是默认的22

[root@nginx-8 ~]# iptables -t filter -A INPUT -p tcp --dport 22 -j DROP

[root@nginx-8 ~]# ###此时Xshell已经断掉了

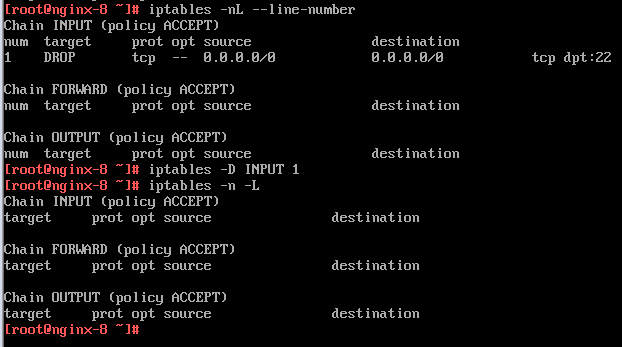
-j 指定要进行的处理行为，基本的处理行为: ACCEPT(接受),DROP(丢弃),REJECT(拒绝) 注意:命令行里的规则,仅仅在内存里临时生效

(3)查看刚才自定义的规则,注意,这里是在服务器里直接操作的,因为ssh端口已经被禁用了!



1. 清空自定义规则，通过指定需要的方式进行删除，这里能看到序号是1的规则被删除

注意，使用序号删除的时候，链要对上，链名称一定要大写。



(5)恢复刚才断掉的SSH连接

1) 去机房重启系统或者登录服务器删除刚才的禁止规则

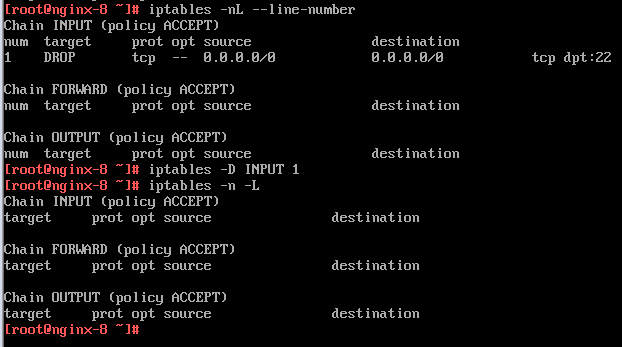
2) 让机房人员重启服务器或者让机房人员拿用户密码登陆进去

3) 通过服务器的远程管理卡管理(推荐)

4) 先写一个定时任务,每五分钟就停止防火墙

5) 测试环境测试好,写成脚本,批量执行

我们恢复的方法,登录服务器删掉刚才的规则



当然也可以执行iptables -F /etc/init.d/iptables stop等.

练习: 禁止用户访问80端口或3306:

[root@nginx-8 ~]# iptables -t filter -A INPUT -p tcp --dport 80 -j DROP

[root@nginx-8 ~]# iptables -nL

Chain INPUT (policy ACCEPT)

target prot opt source destination

DROP tcp -- 0.0.0.0/0 0.0.0.0/0 tcp dpt:80

Chain FORWARD (policy ACCEPT)

target prot opt source destination

Chain OUTPUT (policy ACCEPT)

target prot opt source destination

使用-I和-A的顺序,防火墙的过滤规则顺序的.

-A 是添加规则到指定链的结尾,最后一条

-I 是添加规则到链的开头,第一条

### 查看规则的编号

[root@nginx-8 ~]# iptables -nL --line-number

### 在指定位置插入规则

iptables -I INPUT 2 -p tcp --dport 10000 -j DROP ####在第二条前插入一条规则

### 在指定位置删除规则

iptables -D INPUT 2 ###删除第2条规则

### 小结:总结删除规则的方法

1) iptables -D INPUT -p tcp --dropt 8080 -j DROP

2) iptables -F 删除所有规则

3) /etc/init.d/iptables restart (用iptables命令行配置的命令都是临时生效)

4) iptables -D INPUT规则序号

5)删除nat表的POSTROUTING链

[root@nginx-8 ~]# iptables -t nat -D POSTROUTING 2

### 禁止10.0.0.0 网段连入

iptables -t filter -A INPUT -i eth0 -s 10.0.0.0/24 -j DROP

iptables -A INPUT -i eth0 -s 10.0.0.0/24 -j DROP

注意: iptables默认用的就是filter表,因此以上两条命令.等价.

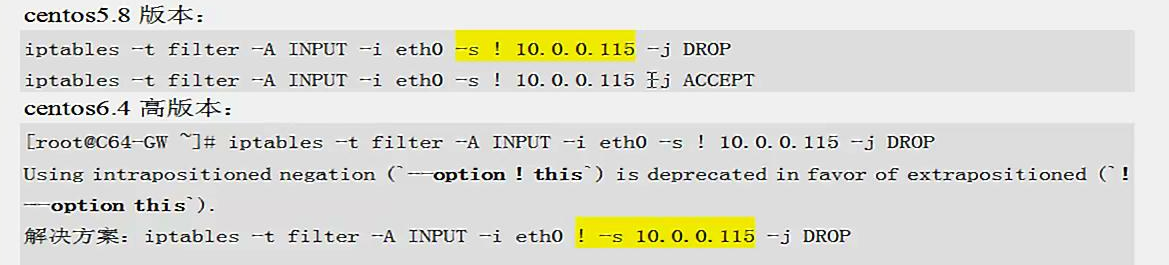
执行以上命令可以发现,这里已经无法远程连接了.

登陆虚拟机,删除刚才禁止的来源地址为10网段的命令.

iptables -D INPUT -i eth0 -s 10.0.0.0/24 -j DROP(完整策略规则删除)

iptables -D INPUT 1 (根据策略在链中的序号删,每条链都是各自从1编号)

### 取反匹配规则



iptables -A INPUT ! -s 10.0.0.0/24 -j DROP

1.源地址不是10.0.0.101 单个IP的禁止连接

iptables -t filter -I INPUT -i eth0 -s ! 10.0.0.101 -j DROP

iptables -A INPUT -p all -i eth0 ! -s 10.0.0.106 -j DROP (-p udp tcp icmp all)

iptables -t filter -I INPUT -p icmp --icmp-type 8 -i eth0 -s ! 10.0.0.115 -j DROP

2.源地址不是10.0.0..0/24的网段禁止连接

iptables -t filter -I INPUT -i eth0 -s ! 10.0.0.0/24 -j DROP

等价于

iptables -t filter -I INPUT -i eht0 -s 10.0.0.0/24 -j ACCEPT

工作场景:

### 封堵ssh端口

iptables -A INPUT -p tcp --dport 22 ! -s 10.0.0.0/24 -j DROP

iptables -A INPUT -p tcp --dport 22 ! -s 172.16.1.0/24 -j DROP

在默认规则为允许的情况下,上述可以封堵SSH访问

### 企业工作中解决这个问题

(1)vpn服务(拨号拨到vpn上,然后以VPN的内网地址去访问内部的机器)

(2)前端对外提供服务的机器SSH端口都做禁止外部IP访问限制,可以开启后端或者不对外提供服务的机器,保留SSH服务(更改root和ssh端口).然后,我们平时就先连接此机器,再去连接其它机器

这种被老师称作为牤牛阵

(3)流量特别大的外网机器不要开启防火墙,会影响性能,购买硬件防火墙

### 封掉3306端口

iptables -A INPUT -p tcp --dport 3306 -j DROP

### 匹配指定的协议

iptables -A INPUT -p tcp

iptables -A INPUT -p udp

如果不指定-p 默认就是

### 匹配指定协议外的所有协议

iptables -A INPUT -p ! tcp

[root@nginx-8 ~]# iptables -A INPUT ! -p tcp -s 10.0.0.110 -j DROP

[root@nginx-8 ~]# dmesg|tail -5

e1000: eth0 NIC Link is Down

e1000: eth0 NIC Link is Up 1000 Mbps Full Duplex, Flow Control: None

e1000: eth0 NIC Link is Down

e1000: eth0 NIC Link is Up 1000 Mbps Full Duplex, Flow Control: None

nf\_conntrack version 0.5.0 (3796 buckets, 15184 max)

### 匹配主机源IP , 目的IP

iptables -A INPUT -s 10.0.0.14

iptables -A INPUT -s ! 10.0.0.14

iptables -t nat -A PREROUTING -d 10.0.0.14

iptables -t nat -A PREROUTING -d ! 10.0.0.14

### 匹配网段

iptables -A INPUT -s 10.0.0.0/24

iptables -A INPUT -s ! 10.0.0.0/24

### 匹配单一端口

iptables -A INPUT -p tcp --sport 53

iptables -A INPUT -p udp --sport 53

### 匹配指定端口之外的端口

iptables -A INPUT -p tcp --sport ! 22

iptables -I INPUT -p tcp ! --dport 22 -s 10.0.0.110 -j DROP

### 匹配端口范围

iptables -A INPUT -p tcp --sport 22:80

iptables -I INPUT -p tcp --dport 21,22,23,24 -j ACCEPT ===>错误语法

iptables -I INPUT -p tcp -m multiport --dport 21,22,23,24 -j ACCEPT

iptables -I INPUT -p tcp --dport 3306:8809 -j ACCEPT

iptables -I INPUT -p tcp --dport 18:80 -j DROP <===最佳方法

### 匹配ICMP类型

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

例如:

iptables -I INPUT -p icmp --icmp-type 8 -j DROP

###icmp有很多类型,其中的8是ping,所以这里相当于禁ping，这样ping这个ip就会超时。

iptables -I INPUT -p icmp --icmp-type 8 -j ACCEPT

iptables -A INPUT -p icmp -m icmp --icmp-type any -j ACCEPT

iptables -A FORWARD -s 10.0.0.0/24 -p icmp -m icmp --icmp-type any -j ACCEPT



还可以指定接收一个IP范围内的ping

### 匹配指定的网络接口

iptables -A INPUT -i eth0

iptables -A FORWARD -o eth0

记忆方法:

--in-interface -i [!] input name[+]

network interface name ([+] for wildcard)

--out-interface -o [!] output name[+]

network interface name ([+] for wildcard)

### 匹配网络状态

-m state --state

NEW: 已经或将启动新的连接

ESTABLISHED: 已建立的连接

RELATED: 正在启动的新连接

INVALID: 非法或无法识别的

FTP 服务是特殊的, 需要配状态

INVALLD

meaning that the packet is accociated with np know connetcion

NEW

meaning that the packet has started a new connection, or otherwise associated with a connection which has not seen packets in both directions,and

ESTABLISHED

meaning that the packet is associated with a connection which has seen packets in both directions

RELATED

meaning that the packet is starting a new connection, but is associated with an existing connection,such as an FTP data transfer , or an ICMP error.

7.允许关联的状态包通过(web服务不需要FTP服务)

#other RELATED ftp协议

#允许关联的状态包

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

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

#来回都得允许(好比看电影上厕所回来不来进来)

### 限定指定时间包的允许通过数量及并发数

-m limit --limit n/{second/minute/hour}:

指定时间内的请求速率"n"为速率,后面时间分别为:秒,分,时

--limit-burst [n]:

在同一时间内允许通过的请求"n"为数字,不指定默认为5

[root@nginx-8 ~]# iptables -I INPUT -s 10.0.0.0/24 -p icmp --icmp-type 8 -m limit --limit 5/min --limit-burst 2 -j ACCEPT

[root@nginx-8 ~]# iptables -nL

Chain INPUT (policy ACCEPT)

target prot opt source destination

ACCEPT icmp -- 10.0.0.0/24 0.0.0.0/0 icmp type 8 limit: avg 5/min burst 2

DROP !tcp -- 10.0.0.110 0.0.0.0/0

Chain FORWARD (policy ACCEPT)

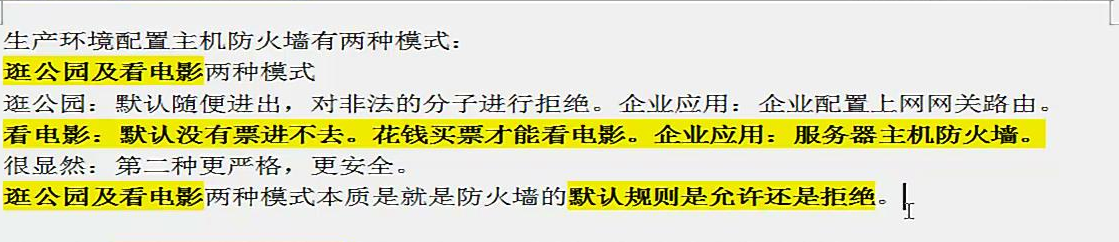
target prot opt source destination

Chain OUTPUT (policy ACCEPT)

target prot opt source destination

## 手动执行iptables命令配置企业生产环境下的防火墙（一个练手的项目）

生产环境配置主机防火墙（看电影模式，默认是拒绝的）的两种模式:



### 清空自定义的规则和链和计数器

[root@nginx-8 ~]# iptables -F

[root@nginx-8 ~]# iptables -X

[root@nginx-8 ~]# iptables -Z

### 首先设置允许ssh连接

[root@nginx-8 ~]# iptables -A INPUT -p tcp -s 10.0.0.0/24 -j ACCEPT

[root@nginx-8 ~]# iptables -A INPUT -p tcp --dport 22 -j ACCEPT

上面两条只要使用一个就可以了。就是允许ssh连接，要么允许指定ip范围，要么指定ssh端口

[root@nginx-8 ~]# iptables -nL

###提示: 此步是为了防止执行下面的步骤,把自己关在外面,除非你是本地处理,这步可以不做.

### 设置允许本机lo通信规则

[root@nginx-8 ~]# iptables -A INPUT -i lo -j ACCEPT

lo会换接口要允许，不能主机和主机通信也不让

[root@nginx-8 ~]# iptables -A OUTPUT -o lo -j ACCEPT

### 目前为止的配置已经允许本机可以操作

[root@nginx-8 ~]# iptables -nL

Chain INPUT (policy ACCEPT)

target prot opt source destination

ACCEPT tcp -- 10.0.0.0/24 0.0.0.0/0

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

ACCEPT all -- 0.0.0.0/0 0.0.0.0/0

Chain FORWARD (policy ACCEPT)

target prot opt source destination

Chain OUTPUT (policy ACCEPT)

target prot opt source destination

ACCEPT all -- 0.0.0.0/0 0.0.0.0/0

### 修改默认规则

[root@nginx-8 ~]# iptables -P INPUT DROP

[root@nginx-8 ~]# iptables -P FORWARD DROP

[root@nginx-8 ~]# iptables -P OUTPUT ACCEPT

[root@nginx-8 ~]# iptables -nL

Chain INPUT (policy DROP) 这一行就是默认规则，上面就是为了用来修改这个

target prot opt source destination

ACCEPT tcp -- 10.0.0.0/24 0.0.0.0/0

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

ACCEPT all -- 0.0.0.0/0 0.0.0.0/0

Chain FORWARD (policy DROP)

target prot opt source destination

Chain OUTPUT (policy ACCEPT)

target prot opt source destination

ACCEPT all -- 0.0.0.0/0 0.0.0.0/0

### 开启信任的IP网段

#setting Lan access rules

#允许IDC LAN/WAN和办公网IP的访问,及对外合作机构访问

iptables -A INPUT -s 124.43.62.96/27 -p all -j ACCEPT <---办公室固定IP段

iptables -A INPUT -s 192.168.1.0/24 -p all -j ACCEPT <---IDC机房的内网网段

iptables -A INPUT -s 10.0.0.0/24 -p all -j ACCEPT <---其他机房的内网网段

iptables -A INPUT -s 203.83.24.0/24 -p all -j ACCEPT <---IDC机房的外网网段

iptables -A INPUT -s 201.82.34.0/24 -p all -j ACCEPT <---其他IDC机房的外网网段

###提示: 本步骤表示允许IDC LAN和办公网IP的无条件连接访问,因为是我们自己人,所以要信任通过,但是,对于外部用户还无法访问服务器的任何服务.

### 允许业务服务端口对外访问(允许http服务无条件通过)

#output service|允许普通的外部服务80

iptables -A INPUT -p tcp --dport 80 -j ACCEPT

### 允许icmp类型协议通过

iptables -A INPUT -p icmp -m icmp --icmp-type any -j ACCEPT

提示:如果不想开,就不执行此命令

如果对内开,对外不开就用下面的方式

iptables -A INPUT -p icmp -s 10.0.0.0/24 -m icmp --icmp-type any -j ACCEPT

#iptables -A INPUT -p icmp --icmp-typr 8 -j ACCEPT

### 允许关联的状态包通过(web服务不要使用FTP服务)

#others RELATED ftp 协议

#允许关联的状态包

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

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

### 最终的配置结果

[root@nginx-8 ~]# iptables -nL

Chain INPUT (policy DROP)

target prot opt source destination

ACCEPT tcp -- 10.0.0.0/24 0.0.0.0/0

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

ACCEPT all -- 0.0.0.0/0 0.0.0.0/0

ACCEPT all -- 124.43.62.96/27 0.0.0.0/0

ACCEPT all -- 192.168.1.0/24 0.0.0.0/0

ACCEPT tcp -- 0.0.0.0/0 0.0.0.0/0 tcp dpt:80

ACCEPT icmp -- 10.0.0.0/24 0.0.0.0/0 icmp type 255

Chain FORWARD (policy DROP)

target prot opt source destination

Chain OUTPUT (policy ACCEPT)

target prot opt source destination

ACCEPT all -- 0.0.0.0/0 0.0.0.0/0

ACCEPT all -- 0.0.0.0/0 0.0.0.0/0 state RELATED,ESTABLISHED

### 2种将iptables临时命令保存为配置文件的方法

[root@nginx-8 ~]# /etc/init.d/iptables save

iptables: Saving firewall rules to /etc/sysconfig/iptables:[ OK ]

或

[root@nginx-8 ~]# iptables-save >/etc/sysconfig/iptables

### 查看已保存的配置文件

[root@nginx-8 ~]# cat /etc/sysconfig/iptables

# Generated by iptables-save v1.4.7 on Sun Apr 16 11:01:16 2017

\*nat

:PREROUTING ACCEPT [2:104]

:POSTROUTING ACCEPT [585:38610]

:OUTPUT ACCEPT [585:38610]

COMMIT

# Completed on Sun Apr 16 11:01:16 2017

# Generated by iptables-save v1.4.7 on Sun Apr 16 11:01:16 2017

\*filter

:INPUT DROP [16:2100]

:FORWARD DROP [0:0]

:OUTPUT ACCEPT [8:512]

-A INPUT -s 10.0.0.0/24 -p tcp -j ACCEPT

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

-A INPUT -i lo -j ACCEPT

-A INPUT -s 124.43.62.96/27 -j ACCEPT

-A INPUT -s 192.168.1.0/24 -j ACCEPT

-A INPUT -p tcp -m tcp --dport 80 -j ACCEPT

-A INPUT -s 10.0.0.0/24 -p icmp -m icmp --icmp-type any -j ACCEPT

-A OUTPUT -o lo -j ACCEPT

-A OUTPUT -m state --state RELATED,ESTABLISHED -j ACCEPT

COMMIT

# Completed on Sun Apr 16 11:01:16 2017

### 生产维护

(1)确定的规则:

编辑/etc/sysconfig/iptables

加入想要的规则:例如:

-A INPUT -p tcp -m tcp --dport 873 -j ACCEPT

/etc/init.d/iptables reload

或者改配置的同时命令行在执行,也是永久生效

[root@nginx-8 ~]# /etc/init.d/iptables reload

iptables: Trying to reload firewall rules: [ OK ]

[root@nginx-8 ~]# /etc/init.d/iptables restart

iptables: Setting chains to policy ACCEPT: nat filter [ OK ]

iptables: Flushing firewall rules: [ OK ]

iptables: Unloading modules: [ OK ]

iptables: Applying firewall rules: [ OK ]

(2)命令试错,没问题了,然后保存到配置文件.这时不需要重启了

### 手工封IP

iptables -I INPUT -s 10.0.0.116 -j DROP #粗,范围大,外部攻击者.

iptables -I INPUT -s 10.0.0.106 -j FROP

iptables -I INPUT -p tcp -s 10.0.0.106 --drpot 80 -j DROP #细,范围小,内部

### 自动封IP

分析web或应用日志或者网络连接状态封掉辣鸡IP

[root@nginx-8 ~]# cat drop\_ip.sh

#!/bin/bash

/bin/netstat -na|grep ESTABLISHED|awk '{print $5}'|awk -F: '{print $1}'|sort|uniq -c|sort -rn|head -10|grep -v -E '192.168|127.0'|awk '{if ($2!=null && $1>4) {print $2}}'>/home/shell/dropip

for i in $(cat /home/shell/dropip)

do

/sbin/iptables -I INPUT -s $i -j DROP

#-m limit --limit 20/min --limit-burst 6

echo "$i kill at $(date)">>/var/log/ddos

done

[root@nginx-8 ~]# cat drop\_ip2.sh

#!/bin/bash

##search connection established ips

#netstat -an|grep EST|awk -F" " '{print $5}'|cut -d ":" -f4|grep "[0-9]\*.[0-9]\*.[0-9]\*.[0-9]\*"|sort |uniq -c|sort -rn|head -50>/home/shell/ip\_connection.log

netstat -an|grep EST|awk -F "[: ]+" '{print $6}'|sort |uniq -c|sort -rn|head -50>/home/shell/ip...

##drop ips

cat /home/shell/ip\_connection.log|egrep -v "[a-z][A-Z]\*"|egrep -v"202.165|127.0"|while read line

do

cat /home/shell/ip\_connection.log|egrep -v "[a-zA-Z]\*"|egrep -v"202.165|127.0"|while read line

do

count=$(echo $line|awk '{print $1}'

ip=$(echo $line|awk '{print $2}'

if [ $count -ge 100 ]

then

echo "all established status connection > 100 ip is: $ip"

echo "apply rule,drop $ip access 100 !"

/sbin/iptables -I INPUT 6 -s $ip -p tcp --dport 80 -j DROP

/sbin/service iptables save

fi

done

### 企业实战题:请用至少两种方法实现:(一个练习题)

写一个脚本解决DOS攻击生产案例

提示:根据web日志或网络连接数,监控当某个IP并发连接数或者短时间内PV达到100,即调用防火墙命令封掉对应的IP,监控的频率为每隔3分钟

防火墙命令为:iptables -I INPUT -s 10.0.0.10 -j DROP

将前一天封掉的IP给解禁

第一种是ip多就禁掉,属于TCP的

第二种是如果访问连接数多就禁掉,属于http的

方法一:

[root@nginx-8 ~]# cat drop\_ip3.sh

#!/bin/bash

. /etc/init.d/functions

conut=10

Path=/server/scripts/access.log.2017-03-23

function ipt(){

awk '{print $1}'$Path|sort|uniq -c|sort -rn >/tmp/tmp.log

exec < /tmp/tmp.log

while read line

do

ip=echo $line|awk '{print $2}'

if [ echo $line|awk '{print $1}' -ge $conut -a iptables -L -n|grep "$ip"|wc -l -lt 1 ]

then

iptables -I INPUT -s $ip -j DROP

RETVAL=$?

if [ $RETVAL -eq 0 ]

then

action "iptables -I INPUT -s $ip -j DROP" /bin/true

echo "$ip" >>/tmp/ip\_$(date +%F).log

else

action "iptables -I INPUT -s $ip -j DROP" /bin/false

fi

fi

done

}

function del(){

[ -f /tmp/ip\_$(date +%F -d '-1 day').log ]||{

echo "log is not exist"

exit 1}

exec </tmp/ip\_$(date +%F -d '-1 day').log

while read line

do

if [ iptables -L -n|grep "$line"|wc -l -ge 1 ]

then

iptables -D INPUT -s $line -j DROP

fi

done

}

function main(){

flag=0

while true

do

sleep 180

((falg++))

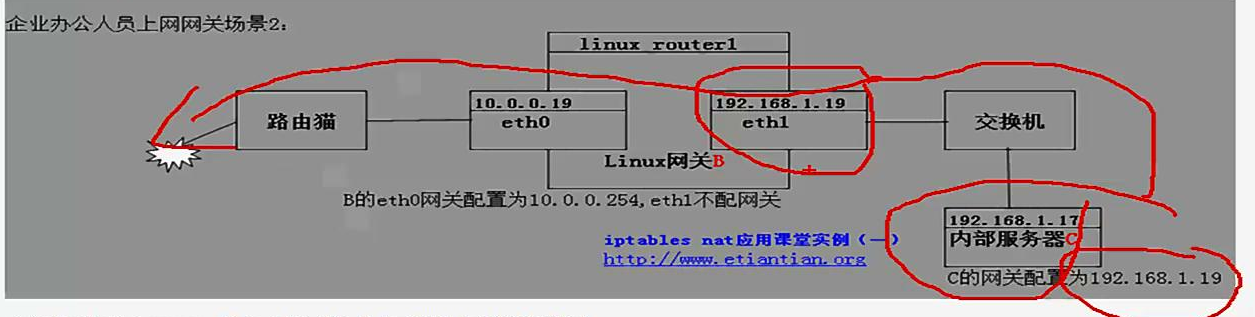
ipt

[ $flag -ge 480 ] && del && flag=0

done

}

main



# 生产环境iptables脚本讲解

见脚本,

技巧:具备外网IP的服务器上不对外的服务最好要做源地址限制

对外提供的服务,不能做源地址限制,例如:80端口

[root@nginx-8 ~]# cat ipt.sh

#!/bin/bash

IPT=/sbin/iptables

#Remove any existing rules

$IPT -F

$IPT -X

$IPT -z

#setting dafault filewall policy

$IPT --policy OUTPUT ACCEPT

$IPT --policy FORWARD DROP

$IPT -p INPUT DROP

#setting for loopback interface

$IPT -A INPUT -i lo -j ACCEPT

$IPT -A OUTPUT -o lo -j ACCEPT

#source Address Spoofing and Other Bad Address

$IPT -A INPUT -i eth0 -s 172.16.0.0/12 -j DROP

$IPT -A INPUT -i eth0 -s 0.0.0.0/8 -j DROP

$IPT -A INPUT -i eth0 -s 169.254.0.0/16 -j DROP

$IPT -A INPUT -i eth0 -s 192.0.2.0/24 -j DROP

#prevent all stealth scans and TCP state falgs

$IPT -A INPUT -p tcp --tcp-flags ALL ALL -j DROP

#All of the bits are cleared

$IPT -A INPUT -p tcp --tcp-flags ALL NONE -j DROP

$IPT -A INPUT -p tcp --tcp-flags ALL FIN,URG,PSH -j DROP

#SYN and RST are both set

#下面这个要学习了tcp的flag之后才能明白是什么意思。比如SYN和RST等等这几个包是有顺序的，不能一个包还没有到其他的包就来了。所以这里添加了一个顺序的验证

$IPT -A INPUT -p tcp --tcp-flags SYN,RST,SYN,RST -j DROP

#SYN and FIN are both set

$IPT -A INPUT -p tcp --tcp-flags SYN,FIN SYN,FIN -j DROP

#FIN and RST are both set

$IPT SYN -A INPUT -p tcp --tcp-flags FIN,RST FIN,RST -j DROP

#FIN is the only bit set,without the expected accompanying ACK

$IPT -A INPUT -p tcp --tcp-flags ACK,FIN FIN -j DROP

#PSH is the only bit set,without the expected accompanying ACK

$IPT -A INPUT -p tcp --tcp-flags ACK,PSH PSH -j DROP

#URG is the only bit set,without the expected accompanying ACK

$IPT -A INPUT -p tcp --tcp-flags ACK,URG PSH -j DROP

#second,port access rules

#nagios

$IPT -A INPUT -s 10.0.10.0/24 -p tcp --dport 5666 -j ACCEPT

$IPT -A INPUT -s 10.0.0.0/24 -p tcp --dport 5666 -j ACCEPT

#db

$IPT -A INPUT -s 10.0.0.0/24 -p tcp --dport 3306 -j ACCEPT

$IPT -A INPUT -s 10.0.0.0/24 -p tcp --dport 3307 -j ACCEPT

#SSH difference from other servers here.>>

$IPT -A INPUT -s 10.0.0.0/24 -p tcp --dport 52113 -j ACCEPT

$IPT -A INPUT -s 10.0.10.0/24 -p tcp --dport 52113 -j ACCEPT

$IPT -A INPUT -s 10.0.10.0/24 -p tcp --dport 22 -j ACCEPT

#http

$IPT -A INPUT -p tcp --dport 80 -j ACCEPT

#snmp

$IPT -A INPUT -s 10.0.0.0/24 -p udp --dport 161 -j ACCEPT

$IPT -A INPUT -s 10.0.0.10/24 -p udp --dport 161 -j ACCEPT

#rsync读音是 [r,sync]也就[啊兴科]

$IPT -A INPUT -s 10.0.0.0/24 -m tcp --dport 873 -j ACCEPT

$IPT -A INPUT -s 10.0.0.10/24 -m rcp --dport 873 -j ACCEPT

#icmp

$IPT -A INPUT -p icmp -m icmp --icmp-type any -j ACCEPT

#others related

$IPT -A INPUT -m state --state ESTABLISHED,RELATED -j ACCEPT

$IPT -A OUTPUT -m state --state ESTABLISHED,RELATED -j ACCEPT

## 问题:企业硬件防火墙和iptables防火墙是否要同时用?

# nat表的核心功能

前面讲的是主机防火墙，也就是filter表的核心应用，现在我们讲nat表的核心应用。

nat表的核心功能: IP及端口的映射转换

1.共享上网

2.端口映射

## 试验环境准备

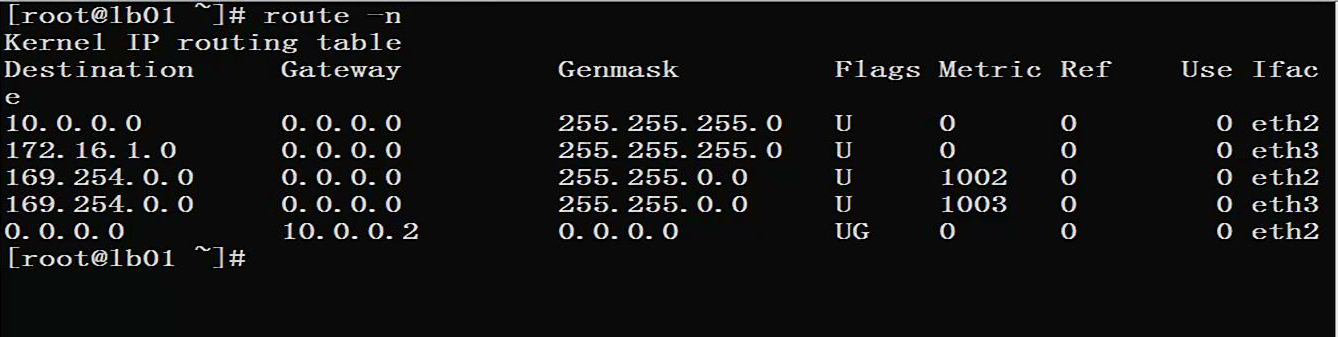
### 服务器网关B需要如下条件

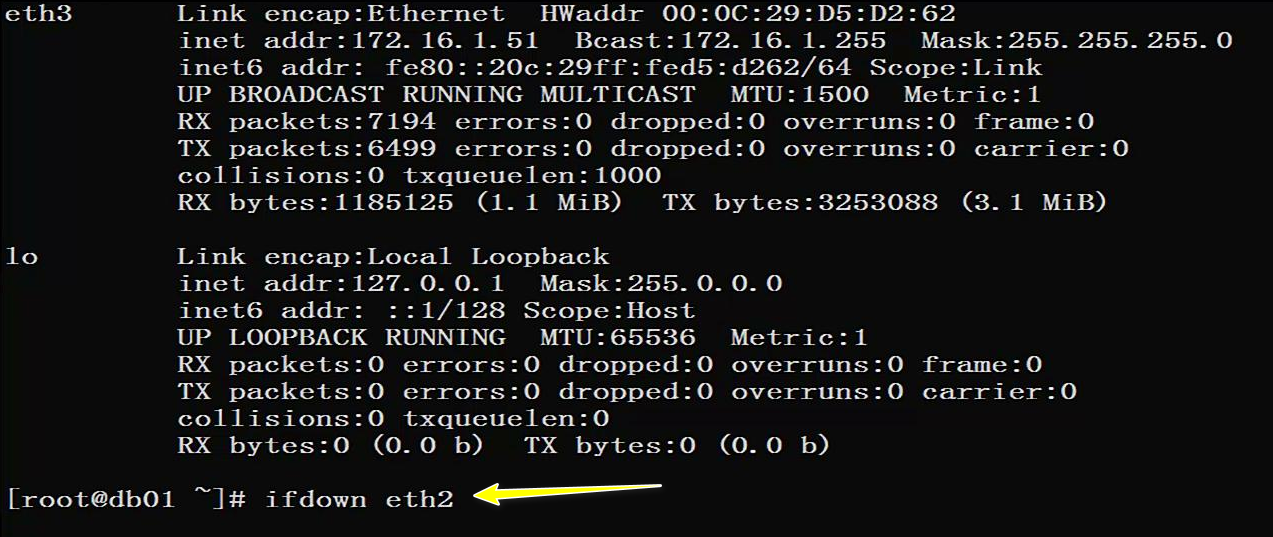
1. 物理条件是具备双网卡,建议：

eth0外网地址()这里是10.0.0.7,gw 10.0.0.254)

eth1内网地址()这里是192.168.1.19,内网卡不配GW)

如何查看使用的网关，使用如下的命令，可以看到网关是10.0.0.2





关于ifdown可以查看<https://blog.csdn.net/lu_embedded/article/details/53215324>

2)确保服务器网关B要可以上网,(B上网才能代理别的及其上网).可以通过ping

3)内核文件/etc/syscrl.conf里开启转发功能

在服务器网关B 192.168.1.19机器上开启路由转发功能,编辑/etc/sysctl.conf修改内容为net.ipv4.ip\_forward=1,然后执行sysctl -p使修改生效..

4)iptables 的filter表的FORWARD链允许转发

5)不要filter防火墙功能,共享上网,因此,最好暂时停掉防火墙测试/etc/init.d/iptables stop

### 环境说明

网关B:

eth2 10.0.0.5 配上级网关 10.0.0.254

eth3 172.16.1.5 不配网关

服务器C:

eth1 172.16.1.8 网关172.16.1.5

### 加载内核参数

## 局域网的机器

1)局域网的机器有一块网卡即可,确保局域网的机器C,默认网关设置了网关服务器Bde eth1内网IP()

## 局域网共享上网项目案例

1.实现C可经过B,通过A上英特网

linux网关B:

局域网共享的两条命令方法:

### 方法1:适合于有固定外网地址的:

iptables -t nat -A POSTROUTING -s 172.16.1.8/24 -o eth0 -j SNAT --to-source 10.0.0.254

-t nat使用nat表格

-s 表示源地址是172.16.1.8/24的数据包

-o eth0 从eth0网卡出去

-j SNAT --to-source把数据包的源地址改变成为10.0.0.254

(1)-s 172.16.1.8/24 办公室或IDC内网网段

(2)-o eth0 为网关的外网卡接口

(3)-j SNAT --to-source 10.0.0.254 是网关外网卡IP地址

### 方法2:适合变化外网地址(ADSL)

iptables -t nat -A POSTROUTING -s 172.16.1.8/24 -j MASQURADE <--伪装

比如你在拨号的网络中没有固定的外网ip地址，这个时候一定就要使用MASQURADE。

### 为什么用POSTROUTING?

企业共享上网:

1.办公网络共享上网（网关要有外网IP，否则用路由zebra）

2.IDC内网机器上网

### 企业上网到底需不需要linux网关

解答:

1)如果由企业路由器的情况下,可以不需要上网网关,使用网关只是解决路由器无法解决的需要.

(例如:上网行为,IP及端口的映射,杀毒)

2)IDC机房,大厦有固定外网IP的宽带,直接用网关解决上网及控制问题.

## 把外部IP地址及端口映射到内部服务器地址及端口(共享上网环境一样)

### 在10段主机可以通过访问B10.0.0.5:9000, 即可访问到172.16.1.8:80

NET PREROUTING

IP映射和端口映射就使用NET PREROUTING链，这样一进来我就能进行修改。

这个需求的意思是，原来在172网段直接使用172.16.1.8:80这个IP就可以访问我们的页面。但是现在我想要做的是在大网10.0.0.5:9000这个IP和端口就可以直接的访问到这个页面，这个时候就需要进行端口映射。

### C配置WEB服务

解答:

(1)在172.16.1.8开启http服务监听9000端口,然后在网关服务器B可以访问

(2)具体转换命令（就这一条命令就解决了）

iptables -t nat -A PREROUTING -d 10.0.0.5 -p tcp --dport 9000 -j DNAT --to-destination 172.16.1.8:80

强调:有个别同学说网关服务器需要开启80服务,但不需要对外服务?

测试结果:网管开启httpd 80后

此时:来自80端口的请求转发依然会转发到后端的服务器,但是当iptables nat规则后,此时就到达了http服务监听的80端口,所以显示的是默认页面.

## 删除nat规则

iptables iptables -t nat -D PREROUTING -d 10.0.0.5 -p tcp --dport 9000 -j DNAT --to-destination 172.16.1.8:80

## iptables常用企业案例

1.linux主机防火墙(表:FILTER 链:INPUT)

2.局域网机器共享上网(表:NAT 链:POSTROUTING)

iptables -t nat -A POSTROUTING -s 172.16.1.0/24 -o eth0 -j SNAT --to-source 10.0.0.5

3.外部地址和端口,映射为内部地址和端口(表: NAT 链:PREROUTING)

iptables -t nat -A POSTROUTING -d 10.0.0.5 -p tcp --dport 9000 -j DNAT --to-destination 172.16.1.8:80

## 实现10网段外网IP和内网172网段IP一对一映射

需求是比如现在有一个笔记本有一个内网IP，然后现在我们要给这个笔记本配置一个外网IP.

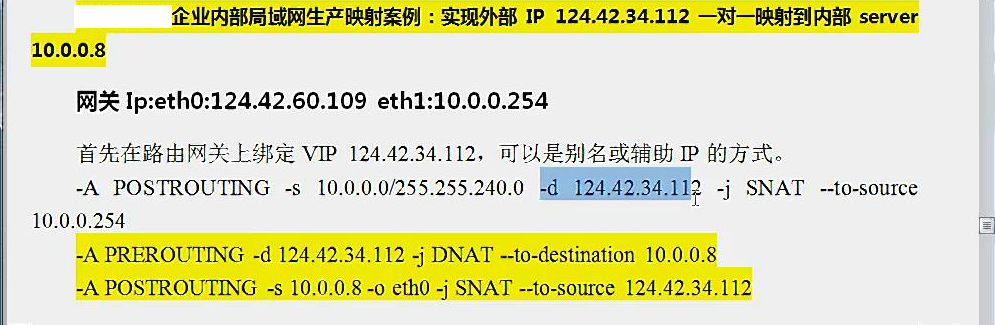
类似于FTP仅靠映射端口没法映射，很难做映射。就需要使用这种方法。

首先在路由网关上绑定124.42.34.112,可以是别名的方式

-A POSTROUTING -s 10.0.0.0/255.255.240.0 -d 124.42.34.112 -j SNAT --to-source 10.0.0.254

-A PREROUTING -d 124.42.34.112 -j DNAT --to-destination 10.0.0.8

-A POSTROUTING -s 10.0.0.8 -o eth0 -j SNAT --to-source 124.42.34.112



为什么要有-A POSTROUTING -s 10.0.0.0/255.255.240.0 -d 124.42.34.112 -j SNAT --to-source 10.0.0.254

因为，比如老板要求我给一个外网IP地址映射到内网IP地址，只要有第二和第三条两个规则，就完成了这个功能，为什么还要第一条，因为映射的功能完成了，那么内网的用户可能也在使用外网的IP地址去访问这个电脑，本来在内网直接使用内网地址就可以解决的问题，现在变成了，需要先访问外网地址，然后在从外网转到内网然后在访问。这样可能会形成数据包的环路。所以对于这样的请求的数据包我们需要设置一个第一条规则，有了这个规则之后，我们就可以对于内网中请求124.42.34.112的数据包直接转到内网的网关，让他们在内网直接通信。这样就不会出现问题了。

## 跨不同网段IP服务通信问题

(1)生产环境大于254台机器网段划分及路由解决方案详解01   
http://v.youku.com/v\_show/id\_XNTAyMjAwMzI0.html   
(2) linux route命令深入浅出与实战案例精讲  
http://oldboy.blog.51cto.com/2561410/1119453  
http://oldboy.blog.51cto.com/2561410/974194  
必看3遍以上。

# 要实现的生产局域网共享上网案例架构图

http://oldboy.blog.51cto.com/2561410/974194

考试题一：linux下如何添加路由(百度面试题)

以上是原题，老男孩老师翻译成如下3道题。

a.如何用命令行方式给linux机器添加一个默认网关，假设网关地址为10.0.0.254？

b. 192.168.1.0网段， 192.168.1.1网关的某一服务器想连入172.16.1.0/24段,该如何添加路由（奇虎360）

iptables -A INPUT --dport 80 -j ACCEPT

iptables -P INPUT DROP

c.如果添加一个主机路由？

请分别解答。

解答：route -net 172.16.1.0/24 gw 192.168.1.1

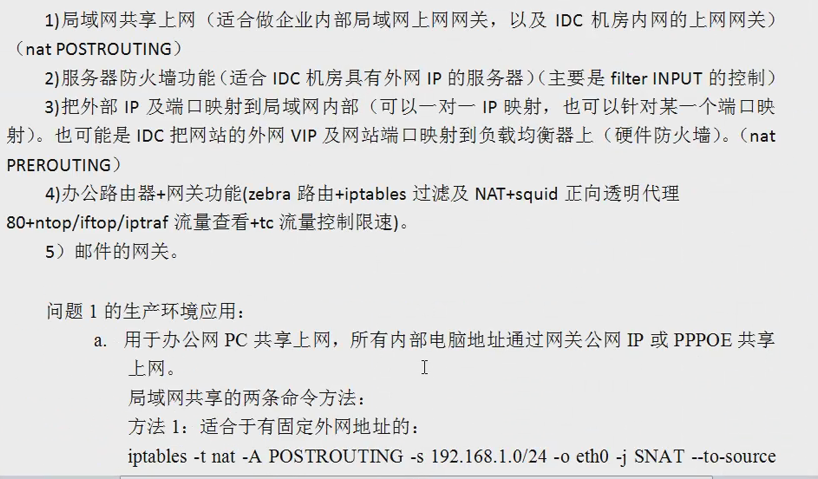
Route -n

Route -host

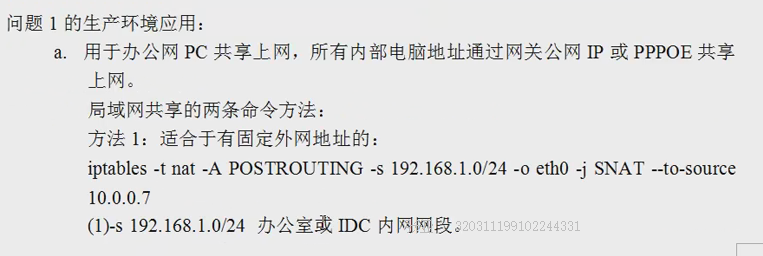
Route -net

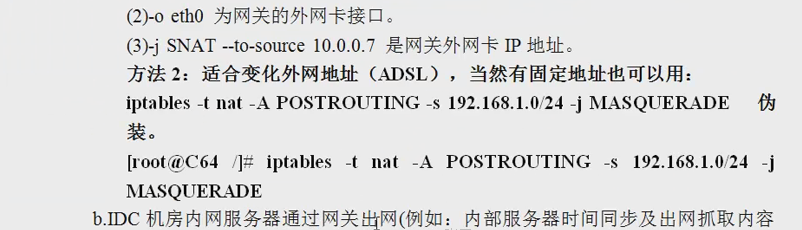
主机路由和网络路由

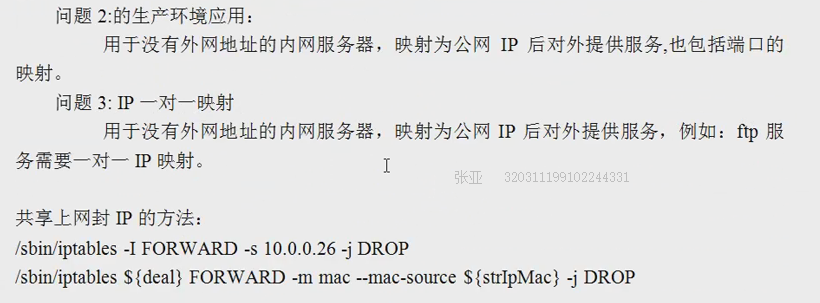
# iptables生产应用场景



ntop等软件可以用来实时监控网络流量，用来限制带宽。防止大流量进行下载，把带宽全部占用了。







## 映射多个外网IP上网

映射多个外网IP上网

iptables -t nat -A POSTROUTING -s 10.0.1.0/255.255.240.0 -o eth0 -j SNAT --to-source 124.42.60.11-124.42.60.16（这是一个范围，而不是一个单独的IP）

iptables -t nat -A POSTROUTING -s 10.0.1.0/255.255.255.0 -o eth0 -j SNAT --to-source 124.42.60.11

iptables -t nat -A POSTROUTING -s 10.0.2.0/255.255.255.0 -o eth0 -j SNAT --to-source 124.42.60.12

问题:国内IP缺乏，IPV4都快耗尽了,国内上网基本都是NAT模式。所有局域网IP都映射成为一个IP所以导致一些问题，可能就会导致我们被封掉。比如很多人一起使用2000人，都映射成为一个IP，然后这个IP访问百度。如果百度的过滤规则设计的不合理，百度可能就认为这是一个DDOS攻击，所以就会把我们封锁掉。另外就是端口是有限的，2000多人高峰期端口可能就不够用了。

1.2000人.被封

2.65535端口资源有限

# iptables 内核优化





这就是iptables的连接跟踪表满了。Iptables有一个连接跟踪，要不你（数据包）去了，然后回来了，我iptables怎么知道？调整的方法就是加大

C64：centos 6.4版本

net.nf\_conntrack\_max = 25000000

net.netfilter.nf\_conntrack\_max = 25000000

前面两个参数是总的队列长度

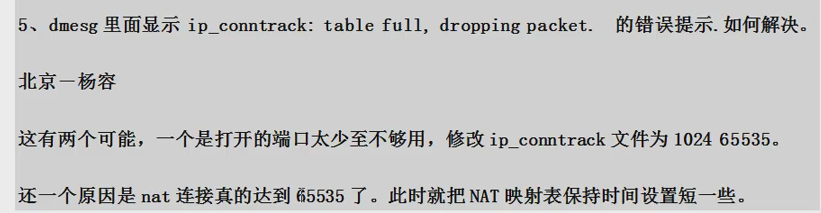
下面的四个参数就是超时时间

net.netfilter.nf\_conntrack\_tcp\_timeout\_established = 180

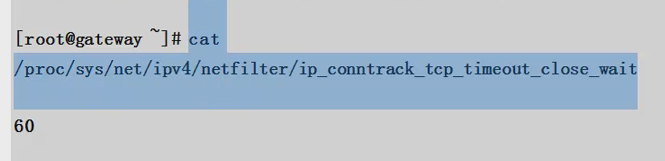
net.netfilter.nf\_conntrack\_tcp\_timeout\_time\_wait = 120

net.netfilter.nf\_conntrack\_tcp\_timeout\_close\_wait = 60

net.netfilter.nf\_conntrack\_tcp\_timeout\_fin\_wait = 120



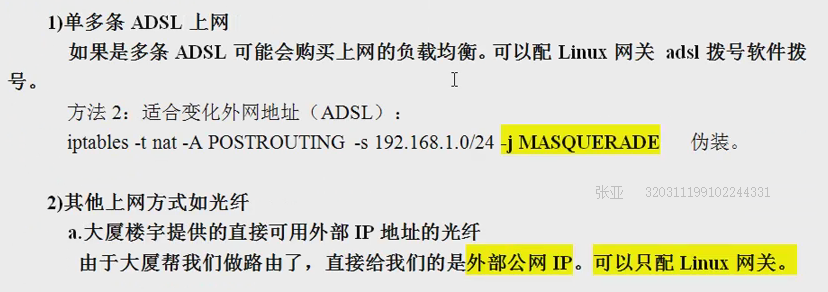
上面的那些参数是有对应的内核文件的，文件如下：

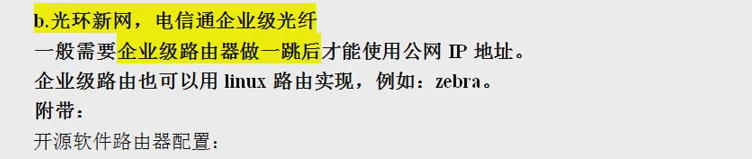


然后老师说如下的五篇文章需要看看：



# 企业上网的·常见接入方式





## 配置zebra路由

Linux上配置zebra(quagga)路由:

client(config)#int eth0

client(config-if)#ip add 10.1.34.81 255.255.255.0

client(config-if)#int eth1

client(config-if)#ip add 110.233.24.96 255.255.255.224

client(config)#ip route 0.0.0.0 0.0.0.0 10.1.32.1

client(config)#ip route 110.233.24.96/27 eth1