iptables 官方网站：http://netfilter.org/

数据包经过防火墙的路径

图1比较完整地展示了一个数据包是如何经过防火墙的，考虑到节省空间，该图实际上包了三种情况：
来自外部，以防火墙（本机）为目的地的包，在图1中自上至下走左边一条路径。
由防火墙（本机）产生的包，在图1中从“本地进程”开始，自上至下走左边一条路径
来自外部，目的地是其它主机的包，在图1中自上至下走右边一条路径。

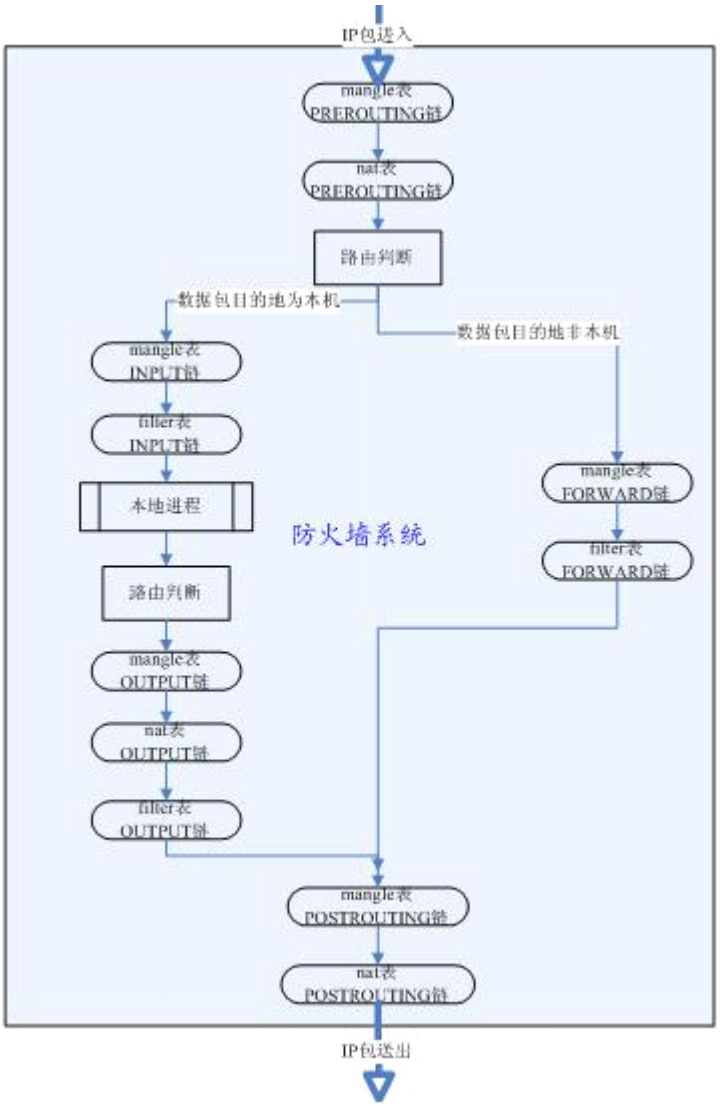


图1

如果我们从上图中略去比较少用的 mangle 表的图示,就有图2所显示的更为清晰的路径图.

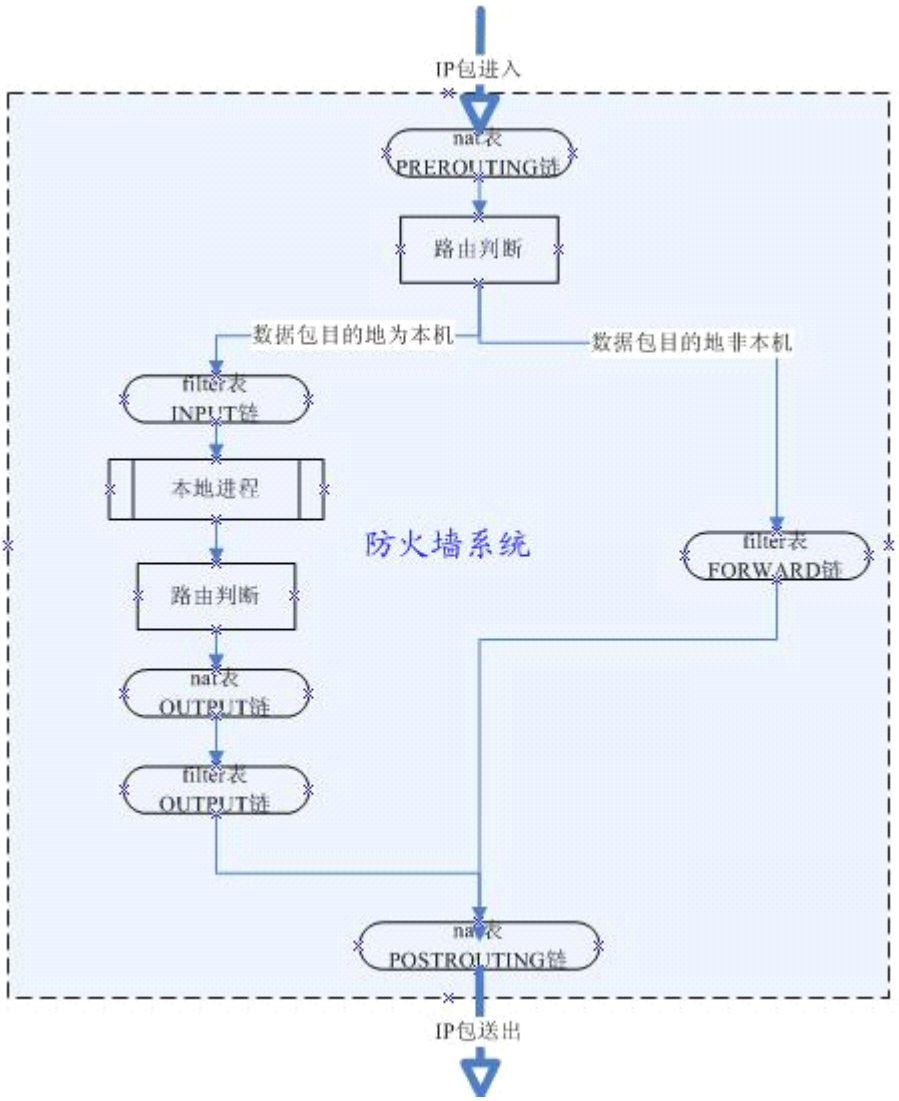


图2

禁止端口的实例

禁止 **ssh** 端口

只允许在192.168.62.1上使用 ssh 远程登录，从其它计算机上禁止使用 ssh
#iptables -A INPUT -s 192.168.62.1 -p tcp --dport 22 -j ACCEPT
#iptables -A INPUT -p tcp --dport 22 -j DROP

禁止代理端口

#iptables -A INPUT -p tcp --dport 3128 -j REJECT

禁止 **icmp** 端口

除192.168.62.1外，禁止其它人 ping 我的主机
#iptables -A INPUT -i eth0 -s 192.168.62.1/32 -p icmp -m icmp --icmp-type echo-request -j ACCEPT
#iptables -A INPUT -i eth0 -p icmp --icmp-type echo-request –j ?DROP
或
#iptables -A INPUT -i eth0 -s 192.168.62.1/32 -p icmp -m icmp --icmp-type 8 -j ACCEPT
#iptables -A INPUT -i eth0 -p icmp -m icmp --icmp-type 8 -j DROP
注：可以用 iptables --protocol icmp --help 查看 ICMP 类型
还有没有其它办法实现?

禁止 QQ 端口

#iptables -D FORWARD -p udp --dport 8000 -j REJECT

强制访问指定的站点

图3

要使192.168.52.0/24网络内的计算机(这此计算机的网关应设为192.168.52.10)强制访问指定的站点,在做为防火墙的计算机(192.168.52.10)上应添加以下规则:
1. 打开 ip 包转发功能
echo 1 > /proc/sys/net/ipv4/ip_forward
2. 在 NAT/防火墙计算机上的 NAT 表中添加目的地址转换规则:
iptables -t nat -I PREROUTING -i eth0 -p tcp --dport 80 -j DNAT --to-destination 202.96.134.130:80
iptables -t nat -I PREROUTING -i eth0 -p udp --dport 80 -j DNAT --to-destination 202.96.134.130:80
3. 在 NAT/防火墙计算机上的 NAT 表中添加源地址转换规则:
iptables -t nat -I POSTROUTING -o eth1 -p tcp --dport 80 -s 192.168.52.0/24 -j SNAT --to-source 202.96.134.10:20000-30000
iptables -t nat -I POSTROUTING -o eth1 -p udp --dport 80 -s 192.168.52.0/24 -j SNAT --to-source 202.96.134.10:20000-30000
4. 测试:在内部网的任一台计算机上打开浏览器,输入任一非本网络的 IP,都将指向 IP 为 202.96.134.130的网站.

发布内部网络服务器

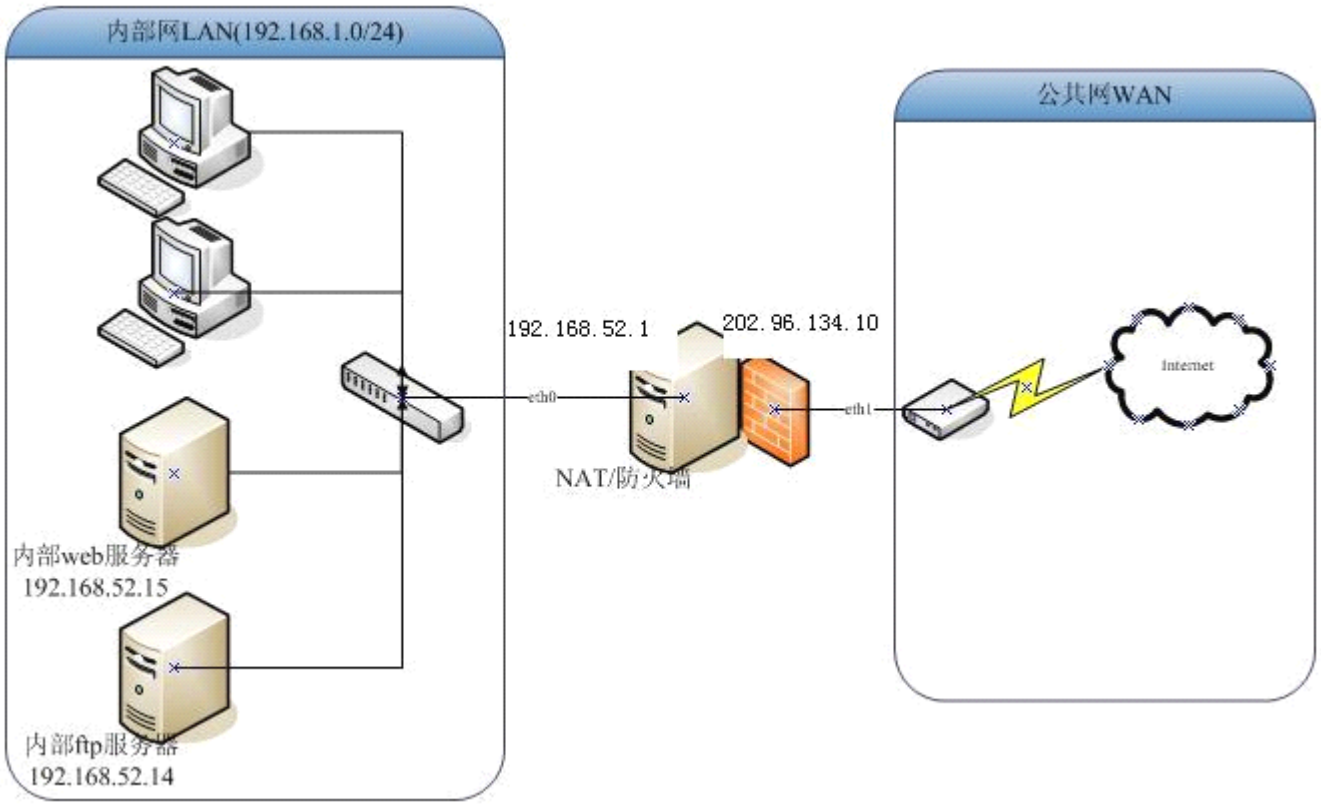


图4

要使因特网上的计算机访问到内部网的 FTP 服务器、WEB 服务器,在做为防火墙的计算机上应添加以下规则:

1. echo 1 > /proc/sys/net/ipv4/ip_forward
2. 发布内部网 web 服务器

iptables -t nat -I PREROUTING -p tcp -i eth1 -s 202.96.134.0/24 --dport 80 -j DNAT --to-destination 192.168.52.15:80

iptables -t nat -I POSTROUTING -p tcp -i eth0 -s 192.168.52.15 --sport 80 -j SNAT --to-source 202.96.134.10:20000-30000

3. 发布内部网 ftp 服务器

iptables -t nat -I PREROUTING -p tcp -i eth1 -s 202.96.134.0/24 --dport 21 -j DNAT --to-destination 192.168.52.14:21

iptables -t nat -I POSTROUTING -p tcp -i eth0 -s 192.168.52.14 --sport 21 -j SNAT --to-source 202.96.134.10:40000-50000

4. 注意:内部网的计算机网关要设置为防火墙的 ip(192.168.52.1)

5. 测试: 用一台 IP 地址为202.96.134.0段的计算机虚拟因特网访问,当在其浏览器中访问 http://202.96.134.10时,实际应看到的是192.168.52.15的的 web 服务;

当访问 ftp://202.96.134.10时,实际应看到的是192.168.52.14上的的 ftp 服务

智能 DNS

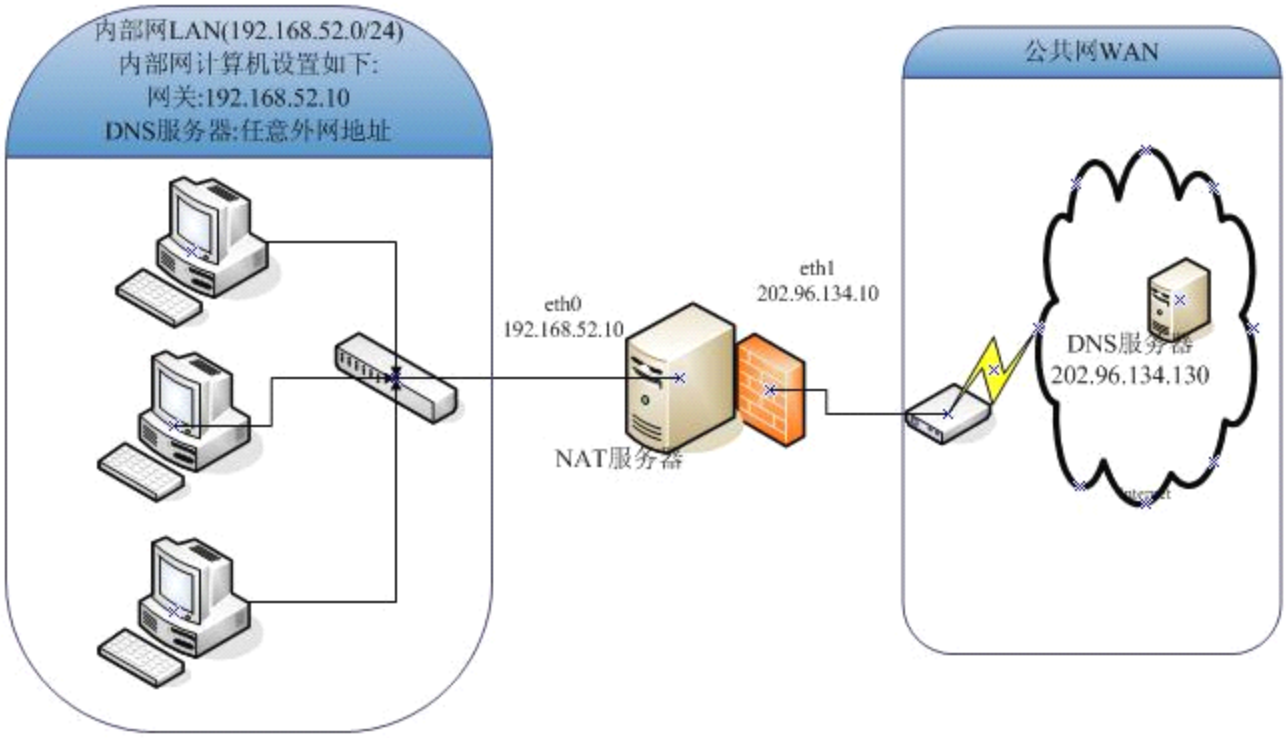


图5

1. echo 1 > /proc/sys/net/ipv4/ip_forward
2. 在 NAT 服务器上添加以下规则:

在 PREROUTING 链中添加目的地址转换规则:

iptables -t nat -I PREROUTING -i eth0 -p tcp --dpor 53 -j DNAT --to-destination 202.96.134.130
iptables -t nat -I PREROUTING -i eth0 -p udp --dpor 53 -j DNAT --to-destination 202.96.134.130

在 POSTROUTING 链中添加源地址转换规则:

iptables -t nat -I POSTROUTING -o eth1 -s 192.168.52.0/24 -p tcp --dpor 53 -j SNAT --to-source 202.96.134.10:40000-50000
iptables -t nat -I POSTROUTING -o eth1 -s 192.168.52.0/24 -p udp --dpor 53 -j SNAT --to-source 202.96.134.10:40000-50000

3. 测试

在内部网任一台计算机上,将 DNS 设置为任意的外网 IP,就可以使用 DNS 测试工具如 nslookup 来解析 DNS 服务器202.96.134.130上的名称.

端口映射

见上节透明代理设置

#iptables -t nat -A PREROUTING -i eth0 -p tcp -s 192.168.62.0/24 --dport 80 -j REDIRECT --to-ports 3128

通过 NAT 上网

典型 NAT 上网

一般做为 NAT 的计算机同时也是局域网的网关，假定该机有两块网卡 eth0、eth1，eth0连接外网，IP 为202.96.134.134；eth1连接局域网，IP 为192.168.62.10
1. 先在内核里打开 ip 转发功能
#echo 1 > /proc/sys/net/ipv4/ip_forward
2.?使局域网用户能访问 internet 所要做的 nat
#iptables -t nat -A POSTROUTING -p tcp -o eth0 -j SNAT --to?202.96.134.134
如果上网的 IP 是动态 IP，则使用以下规则：
#iptables -t nat -A POSTROUTING -o eth0 -s 192.168.62.0/24 -j MASQUERADE
如果是通过 ADSL 上网，且公网 IP 是动态 IP，则使用以下规则：
#iptables -t nat -A POSTROUTING -o ppp0 -s 192.168.62.0/24 -j MASQUERADE
3. 使 internet 用户可以访问局域网内 web 主机所要做的 nat
#iptables -t nat -A PREROUTING -p tcp -d 202.96.134.134 --dport 80 -j DNAT --to-destination 192.168.62.10
注：局域网内的客户端需将默认网关、DNS 设为防火墙的 IP

在我们的网络机房实现 NAT 共享上网

工作环境:上层代理192.168.60.6(4480)，只授予教师机(192.168.62.111)使用该代理的权限
目标：不使用 squid 代理上网，而是使用 NAT 的方式上网
方法：
1) 确保停止教师机(192.168.62.111)的 squid 或其它代理服务
2) 客户端网关、DNS 均指向192.168.62.111，浏览器代理设置为192.168.60.6(4480)。测试在当前情况下能否上网
3) 在教师机(192.168.62.111)上添加如下 iptables 规则：
#iptables -t nat -A POSTROUTING -p tcp -d 192.168.60.6/32 --dport 4480 -j SNAT --to-source 192.168.62.111:10000-30000
解释：对于目的地为192.168.60.6、目的端口为4480的 TCP 包，在经过防火墙路由后，将其源地址转换为192.168.62.111，端口转换为10000-30000间的某个端口。
4) 客户端测试能否上网

IP 规则的保存与恢复

iptables-save 把规则保存到文件中，再由目录 rc.d 下的脚本（/etc/rc.d/init.d/iptables）自动装载
使用命令 iptables-save 来保存规则。一般用
iptables-save > /etc/sysconfig/iptables
生成保存规则的文件 /etc/sysconfig/iptables，
也可以用
service iptables save
它能把规则自动保存在/etc/sysconfig/iptables 中。
当计算机启动时，rc.d 下的脚本将用命令 iptables-restore 调用这个文件，从而就自动恢复了规则。

iptables 指令语法

iptables [-t table] command [match] [-j target/jump]

[-t table] 指定规则表

-t 参数用来，内建的规则表有三个，分别是：nat、mangle 和 filter，当未指定规则表时，则一律视为是 filter。个规则表的功能如下：
nat：此规则表拥有 PREROUTING 和 POSTROUTING 两个规则链，主要功能为进行一对一、一对多、多对多等网址转换工作（SNAT、DNAT），这个规则表除了作网址转换外，请不要做其它用途。
mangle：此规则表拥有 PREROUTING、FORWARD 和 POSTROUTING 三个规则链。除了进行网址转换工作会改写封包外，在某些特殊应用可能也必须去改写封包（TTL、TOS）或者是设定 MARK（将封包作记号，以进行后续的过滤），这时就必须将这些工作定义在 mangle 规则表中，由于使用率不高，我们不打算在这里讨论 mangle 的用法。
filter： 这个规则表是默认规则表，拥有 INPUT、FORWARD 和 OUTPUT 三个规则链，这个规则表顾名思义是用来进行封包过滤的处理动作(例如:DROP、LOG、ACCEPT 或 REJECT)，我们会将基本规则都建立在此规则表中。

command 常用命令列表：

命令 -A, --append
范例 iptables -A INPUT ...
说明 新增规则到某个规则链中，该规则将会成为规则链中的最后一条规则。

命令 -D, --delete
范例 iptables -D INPUT --dport 80 -j DROP
iptables -D INPUT 1
说明 从某个规则链中删除一条规则，可以输入完整规则，或直接指定规则编号加以删除。

命令 -R, --replace
范例 iptables -R INPUT 1 -s 192.168.0.1 -j DROP
说明 取代现行规则，规则被取代后并不会改变顺序。

命令 -I, --insert
范例 iptables -I INPUT 1 --dport 80 -j ACCEPT
说明 插入一条规则，原本该位置上的规则将会往后移动一个顺位。

命令 -L, --list
范例1 iptables -L INPUT
说明 列出某规则链中的所有规则。
范例2 iptables -t nat -L
说明 列出 nat 表所有链中的所有规则。

命令 -F, --flush
范例 iptables -F INPUT
说明 删除 filter 表中 INPUT 链的所有规则。

命令 -Z, --zero
范例 iptables -Z INPUT
说明 将封包计数器归零。封包计数器是用来计算同一封包出现次数，是过滤阻断式攻击不可或缺的工具。

命令 -N, --new-chain
范例 iptables -N allowed
说明 定义新的规则链。

命令 -X, --delete-chain
范例 iptables -X allowed
说明 删除某个规则链。

命令 -P, --policy
范例 iptables -P INPUT DROP
说明 定义过滤政策。 也就是未符合过滤条件之封包， 默认的处理方式。

命令 -E, --rename-chain
范例 iptables -E allowed disallowed
说明 修改某 自定义规则链的名称。

[match] 常用封包匹配参数

参数 -p, --protocol
范例 iptables -A INPUT -p tcp
说明 匹配通讯协议类型是否相符，可以使用 ！运算符进行反向匹配，例如：
-p !tcp
意思是指除 tcp 以外的其它类型，如 udp、icmp ...等。
如果要匹配所有类型，则可以使用 all 关键词，例如：
-p all

参数 -s, --src, --source
范例 iptables -A INPUT -s 192.168.1.1
说明 用来匹配封包的来源 IP，可以匹配单机或网络，匹配网络时请用数字来表示 子网掩码，例如：
-s 192.168.0.0/24
匹配 IP 时可以使用 ！运算符进行反向匹配，例如：
-s !192.168.0.0/24。

参数 -d, --dst, --destination
范例 iptables -A INPUT -d 192.168.1.1
说明 用来匹配封包的目的地 IP，设定方式同上。

参数 -i, --in-interface
范例 iptables -A INPUT -i eth0
说明 用来匹配封包是从哪块网卡进入，可以使用通配字符 + 来做大范围匹配，例如：
-i eth+
表示所有的 ethernet 网卡
也可以使用 ！运算符进行反向匹配，例如：
-i !eth0

参数 -o, --out-interface
范例 iptables -A FORWARD -o eth0

说明 用来匹配封包要从哪 块网卡送出，设定方式同上。

参数 --sport, --source-port
范例 iptables -A INPUT -p tcp --sport 22
说明 用来匹配封包的源端口，可以匹配单一端口，或是一个范围，例如：
--sport 22:80
表示从 22 到 80 端口之间都算是符合条件，如果要匹配不连续的多个端口，则必须使用 --multiport 参数，详见后文。匹配端口号时，可以使用 ！运算符进行反向匹配。

参数 --dport, --destination-port
范例 iptables -A INPUT -p tcp --dport 22
说明 用来匹配封包的目的地端口号，设定方式同上

参数 --tcp-flags
范例 iptables -p tcp --tcp-flags SYN,FIN,ACK SYN
说明匹配 TCP 封包的状态标志，参数分为两个部分，第一个部分列举出想 匹配的标志，第二部分则列举前述标志中哪些有被设置，未被列举的标志必须是空的。TCP 状态标志包括：SYN（同步）、ACK（应答）、FIN（结束）、RST（重设）、URG（紧急） 、PSH（强迫推送） 等均可使用于参数中，除此之外还可以使用关键词 ALL 和 NONE 进行匹配。匹配标志时，可以使用 ！运算符行反向匹配。

参数 --syn
范例 iptables -p tcp --syn
说明 用来表示 TCP 通信协议中，SYN 位被打开，而 ACK 与 FIN 位关闭的分组，即 TCP 的初始连接，与 iptables -p tcp --tcp-flags SYN,FIN,ACK SYN 的作用完全相同，如果使用 !运算符，可用来 匹配非要求连接封包。

参数 -m multiport --source-port
范例 iptables -A INPUT -p tcp -m multiport --source-port 22,53,80,110
说明用来匹配不连续的多个源端口，一次最多可以匹配 15 个端口，可以使用 ！运算符进行反向匹配。

参数 -m multiport --destination-port
范例 iptables -A INPUT -p tcp -m multiport --destination-port 22,53,80,110
说明用来 匹配不连续的多个目的地端口号，设定方式同上

参数 -m multiport --port
范例 iptables -A INPUT -p tcp -m multiport --port 22,53,80,110
说明 这个参数比较特殊，用来匹配源端口和目的端口号相同的封包，设定方式同上。注意：在本范例中，如果来源端口号为 80目的地端口号为 110，这种封包并不算符合条件。

参数 --icmp-type
范例 iptables -A INPUT -p icmp --icmp-type 8
说明用来匹配 ICMP 的类型编号，可以使用代码或数字编号来进行 匹配。请打 iptables -p icmp --help 来查看有哪些代码可用。

参数 -m limit --limit
范例 iptables -A INPUT -m limit --limit 3/hour

说明 用来匹配某段时间内封包的平均流量，上面的例子是用来 匹配：每小时平均流量是否超过一次 3 个封包。 除了每小时平均次外，也可以每秒钟、每分钟或每天平均一次，默认值为每小时平均一次，参数如后： /second、 /minute、/day。 除了进行封 包数量的匹配外，设定这个参数也会在条件达成时，暂停封包的匹配动作，以避免因骇客使用洪水攻击法，导致服务被阻断。

参数 --limit-burst
范例 iptables -A INPUT -m limit --limit-burst 5
说明 用来匹配瞬间大量封包的数量，上面的例子是用来匹配一次同时涌入的封包是否超过 5 个（这是默认值），超过此上限的封包将被直接丢弃。使用效果同上。

参数 -m mac --mac-source
范例 iptables -A INPUT -m mac --mac-source 00:00:00:00:00:01
说明 用来匹配封包来源网络接口的硬件地址，这个参数不能用在 OUTPUT 和 POSTROUTING 规则链上，这是因为封包要送到网 卡后，才能由网卡驱动程序透过 ARP 通讯协议查出目的地的 MAC 地址，所以 iptables 在进行封包匹配时，并不知道封包会送到哪个网络接口去。

参数 --mark
范例 iptables -t mangle -A INPUT -m mark --mark 1
说明 用来匹配封包是否被表示某个号码，当封包被 匹配成功时，我们可以透过 MARK 处理动作，将该封包标示一个号码，号码最大不可以超过 4294967296。

参数 -m owner --uid-owner
范例
iptables -A OUTPUT -m owner --uid-owner 500
说明 用来匹配来自本机的封包，是否为某特定使用者所产生的，这样可以避免服务器使用 root 或其它身分将敏感数据传送出，可以降低系统被骇的损失。可惜这个功能无法 匹配出来自其它主机的封包。
参数 -m owner --gid-owner
范例 iptables -A OUTPUT -m owner --gid-owner 0
说明 用来匹配来自本机的封包，是否为某特定使用者群组所产生的，使用时机同上。

参数 -m owner --pid-owner
范例
iptables -A OUTPUT -m owner --pid-owner 78
说明 用来匹配来自本机的封包，是否为某特定进程所产生的，使用时机同上。
参数 -m owner --sid-owner
范例
iptables -A OUTPUT -m owner --sid-owner 100
说明 用来匹配来自本机的封包，是否为某特定 连接（Session ID）的响应封包，使用时机同上。
参数 -m state --state
范例
iptables -A INPUT -m state --state RELATED,ESTABLISHED
说明 用来匹配连接状态， 连接状态共有四种：INVALID、ESTABLISHED、NEW 和 RELATED。
INVALID 表示该封包的连接编号（Session ID）无法辨识或编号不正确。
ESTABLISHED 表示该封包属于某个已经建立的连接。

NEW 表示该封包想要起始一个连接（重设连接或将连接重导向）。
RELATED 表示该封包是属于某个已经建立的连接，所建立的新连接。例如：FTP-DATA 连接必定是源自某个 FTP 连接。

[-j target/jump] 常用的处理动作：

-j 参数用来指定要进行的处理动作，常用的处理动作包括：ACCEPT、REJECT、DROP、REDIRECT、MASQUERADE、LOG、DNAT、SNAT、MIRROR、QUEUE、RETURN、MARK，分别说明如下：
ACCEPT： 将封包放行，进行完此处理动作后，将不再匹配其它规则，直接跳往下一个规则链（natostrouting）。
REJECT： 拦阻该封包，并传送封包通知对方，可以传送的封包有几个选择：ICMP port-unreachable、ICMP echo-reply 或是 tcp-reset（这个封包会要求对方关闭 连接），进行完此处理动作后，将不再匹配其它规则，直接中断过滤程序。 范例如下：
iptables -A FORWARD -p TCP --dport 22 -j REJECT --reject-with tcp-reset
DROP： 丢弃封包不予处理，进行完此处理动作后，将不再匹配其它规则，直接中断过滤程序。
REDIRECT： 将封包重新导向到另一个端口（PNAT），进行完此处理动作后，将会继续匹配其它规则。 这个功能可以用来实现透明代理或用来保护 web 服务器。例如：
iptables -t nat -A PREROUTING -p tcp --dport 80 -j REDIRECT --to-ports 8080
MASQUERADE： 改写封包来源 IP 为防火墙 NIC IP，可以指定 port 对应的范围，进行完此处理动作后，直接跳往下一个规则 链（manglepostrouting）。这个功能与 SNAT 略有不同，当进行 IP 伪装时，不需指定要伪装成哪个 IP，IP 会从网卡直接读取，当使用拨 号接连时，IP 通常是由 ISP 公司的 DHCP 服务器指派的，这个时候 MASQUERADE 特别有用。范例如下：

iptables -t nat -A POSTROUTING -p TCP -j MASQUERADE --to-ports 1024-31000

LOG： 将封包相关讯息纪录在 /var/log 中，详细位置请查阅 /etc/syslog.conf 配置文件，进行完此处理动作后，将会继续匹配其规则。例如：
iptables -A INPUT -p tcp -j LOG --log-prefix "INPUT packets"
SNAT： 改写封包来源 IP 为某特定 IP 或 IP 范围，可以指定 port 对应的范围，进行完此处理动作后，将直接跳往下一个规则（mangleostrouting）。范例如下：
iptables -t nat -A POSTROUTING -p tcp -o eth0 -j SNAT
--to-source?194.236.50.155-194.236.50.160:1024-32000
DNAT： 改写封包目的地 IP 为某特定 IP 或 IP 范围，可以指定 port 对应的范围，进行完此处理动作后，将会直接跳往下一个规则链（filter:input 或 filter:forward）。范例如下：
iptables -t nat -A PREROUTING -p tcp -d 15.45.23.67 --dport 80 -j DNAT --to-destination
192.168.1.1-192.168.1.10:80-100
MIRROR： 镜射封包，也就是将来源 IP 与目的地 IP 对调后，将封包送回，进行完此处理动作后，将会中断过滤程序。
QUEUE： 中断过滤程序，将封包放入队列，交给其它程序处理。通过自行开发的处理程序，可以进行其它应用，例如：计算连接费用等。

RETURN： 结束在目前规则链中的过滤程序，返回主规则链继续过滤，如果把自定义规则链看成是一个子程序，那么这个动作，就相当于提前结束子程序并返回到主程序中。
MARK： 将封包标上某个代号，以便提供作为后续过滤的条件判断依据，进行完此处理动作后，将会继续匹配其它规则。范例如下：
iptables -t mangle -A PREROUTING -p tcp --dport 22 -j MARK --set-mark 2

iptables 实例

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。请大家注意。
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（就是 3306 端口）
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
```

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
```

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 键#####################################
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
#########################FORWARD 链############################
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 "色情" -j DROP
iptables -I FORWARD -p tcp --sport 80 -m string --string "广告" -j DROP
#禁止 ay2000.net，宽频影院，色情，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！
```

squid+iptables

[原创] squid+iptables 实现网关防火墙
http://www.chinaunix.net 作者:jackylau 发表于：2007-05-27 10:40:01
【发表评论】【查看原文】【Proxy 服务器讨论区】【关闭】
需求说明：此服务器用作网关、MAIL(开启 web、smtp、pop3)、FTP、DHCP 服务器，内部一台机器(192.168.0.254)对外提供 dns 服务,为了不让无意者轻易看出此服务器开启了 ssh 服务器，故把 ssh 端口改为2018.另把 proxy 的端口改为60080
eth0:218.28.20.253,外网口
eth1:192.168.0.1/24,内网口
[jackylau@proxyserver init.d]$cat /etc/squid/squid.conf(部份如下)

```
http_port 192.168.0.1:60080
httpd_accel_port 80
httpd_accel_host virtual
httpd_accel_with_proxy on
httpd_accel_uses_host_header on
acl allow_lan src 192.168.0.0/24
http_access allow allow_lan
visible_hostname proxyserver
```

[jackylau@proxyserver init.d]$ cat firewall

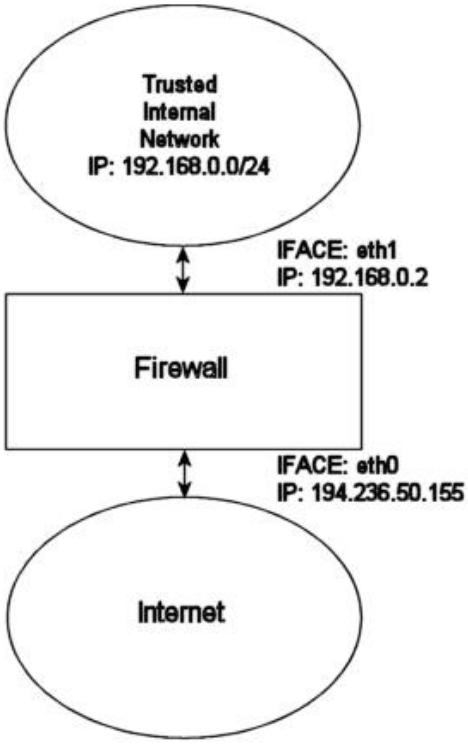
```
#!/bin/sh
# Author: jackylau <squidipt@yahoo.com.cn>;
# chkconfig: 2345 08 92
# description: firewall
# Time on 2005.08.02

# killproc
# Set ENV
INET_IP="218.28.20.253"
INET_IFACE="eth0"
LAN_IP="192.168.0.1"
LAN_IP_RANGE="192.168.0.0/24"
LAN_BROADCAST_ADDRESS="192.168.0.255"
LAN_IFACE="eth1"
LO_IFACE="lo"
LO_IP="127.0.0.1"
IPTABLES="/sbin/iptables"
```

```
start(){
echo -n $"Starting firewall:"
/sbin/depmod -a
/sbin/modprobe ip_tables
/sbin/modprobe ip_conntrack
/sbin/modprobe iptable_filter
/sbin/modprobe iptable_mangle
/sbin/modprobe iptable_nat
/sbin/modprobe ipt_LOG
/sbin/modprobe ipt_limit
/sbin/modprobe ipt_state
echo "1" >; /proc/sys/net/ipv4/ip_forward
# Set policies
$IPTABLES -P INPUT DROP
$IPTABLES -P OUTPUT DROP
$IPTABLES -P FORWARD DROP
# Add bad_tcp_packets, allowed and icmp_packets
$IPTABLES -N bad_tcp_packets
$IPTABLES -N tcp_packets
$IPTABLES -N udp_packets
$IPTABLES -N allowed
$IPTABLES -N icmp_packets
# bad_tcp_packets
$IPTABLES -A bad_tcp_packets -p tcp ! --syn -m state --state NEW -j LOG --log-level INFO
--log-prefix "New not syn:"
$IPTABLES -A bad_tcp_packets -p TCP ! --syn -m state --state NEW -j DROP
# allowed
$IPTABLES -A allowed -p TCP --syn -j ACCEPT
$IPTABLES -A allowed -p TCP -m state --state ESTABLISHED,RELATED -j ACCEPT
$IPTABLES -A allowed -p TCP -j DROP
$IPTABLES -A INPUT -p ALL -i $LAN_IFACE -d $LAN_BROADCAST_ADDRESS -j ACCEPT
# TCP rules
$IPTABLES -A tcp_packets -p TCP -s 0/0 --dport 20 -j allowed
$IPTABLES -A tcp_packets -p TCP -s 0/0 --dport 21 -j allowed
$IPTABLES -A tcp_packets -p TCP -s 0/0 --dport 25 -j allowed
$IPTABLES -A tcp_packets -p TCP -s 0/0 --dport 80 -j allowed
$IPTABLES -A tcp_packets -p TCP -s 0/0 --dport 110 -j allowed
$IPTABLES -A tcp_packets -p TCP -s 0/0 --dport 2018 -j allowed
# UDP rules
$IPTABLES -A udp_packets -p UDP -s 0/0 --destination-port 67 -j ACCEPT
# ICMP rules
$IPTABLES -A icmp_packets -p ICMP -s 0/0 --icmp-type 8 -j ACCEPT
$IPTABLES -A icmp_packets -p ICMP -s 0/0 --icmp-type 11 -j ACCEPT
# INPUT chain
$IPTABLES -A INPUT -p tcp -j bad_tcp_packets
$IPTABLES -A INPUT -p ALL -i $LAN_IFACE -s $LAN_IP_RANGE -j ACCEPT
$IPTABLES -A INPUT -p ALL -i $LO_IFACE -s $LO_IP -j ACCEPT
$IPTABLES -A INPUT -p ALL -i $LO_IFACE -s $LAN_IP -j ACCEPT
$IPTABLES -A INPUT -p ALL -i $LO_IFACE -s $INET_IP -j ACCEPT
$IPTABLES -A INPUT -p ALL -d $INET_IP -m state --state ESTABLISHED,RELATED -j
ACCEPT
$IPTABLES -A INPUT -p TCP -i $INET_IFACE -j tcp_packets
$IPTABLES -A INPUT -p UDP -i $INET_IFACE -j udp_packets
```

```
$IPTABLES -A INPUT -p ICMP -i $INET_IFACE -j icmp_packets
$IPTABLES -A INPUT -m limit --limit 3/minute --limit-burst 3 -j LOG --log-level DEBUG
--log-prefix "IPT INPUT packet died: "
# FORWARD chain
$IPTABLES -A FORWARD -p tcp -j bad_tcp_packets
$IPTABLES -A FORWARD -i $LAN_IFACE -j ACCEPT
$IPTABLES -A FORWARD -m state --state ESTABLISHED,RELATED -j ACCEPT
$IPTABLES -A FORWARD -m limit --limit 3/minute --limit-burst 3 -j LOG --log-level DEBUG
--log-prefix "IPT FORWARD packet died: "
# OUTPUT chain
$IPTABLES -A OUTPUT -p tcp -j bad_tcp_packets
$IPTABLES -A OUTPUT -p ALL -s $LO_IP -j ACCEPT
$IPTABLES -A OUTPUT -p ALL -s $LAN_IP -j ACCEPT
$IPTABLES -A OUTPUT -p ALL -s $INET_IP -j ACCEPT
$IPTABLES -A OUTPUT -m limit --limit 3/minute --limit-burst 3 -j LOG --log-level DEBUG
--log-prefix "IPT OUTPUT packet died: "
# SNAT table
$IPTABLES -t nat -A POSTROUTING -o $INET_IFACE -j SNAT --to-source $INET_IP
# DNAT table
$IPTABLES -t nat -A PREROUTING -p ! icmp -d $INET_IP -dport 53 -j DNAT --to-destination
192.168.0.254:53
# REDIRECT
$IPTABLES -t nat -A PREROUTING -i $LAN_IFACE -p tcp -s $LAN_IP_RANGE --dport 80 -j
REDIRECT --to-ports 60080
touch /var/lock/subsys/firewall
}
stop(){
echo -n $"Stoping firewall:"
echo "0">;/proc/sys/net/ipv4/ip_forward
$IPTABLES -P INPUT ACCEPT
$IPTABLES -P FORWARD ACCEPT
$IPTABLES -P OUTPUT ACCEPT
$IPTABLES -t nat -P PREROUTING ACCEPT
$IPTABLES -t nat -P POSTROUTING ACCEPT
$IPTABLES -t nat -P OUTPUT ACCEPT
$IPTABLES -t mangle -P PREROUTING ACCEPT
$IPTABLES -t mangle -P OUTPUT ACCEPT
$IPTABLES -F
$IPTABLES -t nat -F
$IPTABLES -t mangle -F
$IPTABLES -X
$IPTABLES -t nat -X
$IPTABLES -t mangle -X
rm -f /var/lock/subsys/firewall
}
status(){
clear
echo "------------------------------------------------------------------"
$IPTABLES -L
echo "------------------------------------------------------------------"
$IPTABLES -t nat -L POSTROUTING
echo "------------------------------------------------------------------"
```

```
$IPTABLES -t nat -L PREROUTING
}
case "$1" in
start)
start
;;
stop)
stop
;;
restart)
stop
start
;;
*)
echo "$0 [start|stop|restart|status]"
;;
esac
cp firewall /etc/init.d/
chmod 700 /etc/init.d/firewall
chkconfig --add firewall
```

rc.firewall 脚本代码




```
#!/bin/sh
#
# rc.firewall - Initial SIMPLE IP Firewall script for Linux 2.4.x and iptables
#
# Copyright (C) 2001?Oskar Andreasson <bluefluxATkoffeinDOTnet>
#
```

```
# This program is free software; you can redistribute it and/or modify
# it under the terms of the GNU General Public License as published by
# the Free Software Foundation; version 2 of the License.
#
# This program is distributed in the hope that it will be useful,
# but WITHOUT ANY WARRANTY; without even the implied warranty of
# MERCHANTABILITY or FITNESS FOR A PARTICULAR PURPOSE.?See the
# GNU General Public License for more details.
#
# You should have received a copy of the GNU General Public License
# along with this program or from the site that you downloaded it
# from; if not, write to the Free Software Foundation, Inc., 59 Temple
# Place, Suite 330, Boston, MA?02111-1307?USA
#
###############################################################################
#
# 1.  配置选项.
#
#
# 1.1 Internet  相关变量设置.
#
INET_IP="194.236.50.155"
INET_IFACE="eth0"
INET_BROADCAST="194.236.50.255"
#
# 1.1.1 DHCP 相关设置
#
#
# 1.1.2 PPPoE 相关设置
#
#
# 1.2  局域网相关变量设置.
#
LAN_IP="192.168.0.2"   #防火墙连接局域网的 IP 地址
LAN_IP_RANGE="192.168.0.0/16"   #局域网地址
LAN_IFACE="eth1"   #防火墙连接局域网的网络接口
#
# 1.3 DMZ  非军事区相关变量设置.
#
#
# 1.4  本机相关变量设置.
#
LO_IFACE="lo"   #本地接口名称
LO_IP="127.0.0.1" #本地接口 IP
#
# 1.5 IPTables  路径设置.
#
IPTABLES="/usr/sbin/iptables"
#
# 1.6  其它配置.
#
```

```
#############################################################################
#
# 2.  要加载的模块.
#
#
#  初始加载的模块
#
/sbin/depmod -a
#
# 2.1  需加载的模块
#
/sbin/modprobe ip_tables
/sbin/modprobe ip_conntrack
/sbin/modprobe iptable_filter
/sbin/modprobe iptable_mangle
/sbin/modprobe iptable_nat
/sbin/modprobe ipt_LOG
/sbin/modprobe ipt_limit
/sbin/modprobe ipt_state
#
# 2.2  不需加载的模块
#
#/sbin/modprobe ipt_owner
#/sbin/modprobe ipt_REJECT
#/sbin/modprobe ipt_MASQUERADE
#/sbin/modprobe ip_conntrack_ftp
#/sbin/modprobe ip_conntrack_irc
#/sbin/modprobe ip_nat_ftp
#/sbin/modprobe ip_nat_irc
#############################################################################
#
# 3. /proc  设置.
#
#
# 3.1  需要的 proc 配置
#
echo "1" > /proc/sys/net/ipv4/ip_forward
#
# 3.2  不需要的 proc 配置
#
#echo "1" > /proc/sys/net/ipv4/conf/all/rp_filter
#echo "1" > /proc/sys/net/ipv4/conf/all/proxy_arp
#echo "1" > /proc/sys/net/ipv4/ip_dynaddr
#############################################################################
#
# 4.  建立规则.
#
######
# 4.1 Filter 表
#
```

```
#
# 4.1.1 建立策略
#
$IPTABLES -P INPUT DROP
$IPTABLES -P OUTPUT DROP
$IPTABLES -P FORWARD DROP
#
# 4.1.2 创建自定义链 Create userspecified chains
#
#
# 为不可靠的 tcp 包建立自定义链 Create chain for bad tcp packets
#
$IPTABLES -N bad_tcp_packets
#
# 分别为 ICMP, TCP 和 UDP 协议建立自定义链 Create separate chains for ICMP, TCP and UDP
to traverse
#
$IPTABLES -N allowed
$IPTABLES -N tcp_packets
$IPTABLES -N udp_packets
$IPTABLES -N icmp_packets
#
# 4.1.3 在自定义链建立规则 Create content in userspecified chains
#
#
# bad_tcp_packets 链 bad_tcp_packets chain
#
#这条链包含的规则检查进入包（incoming packet）的包头是否不正常或有没有其他问题，并进
行相应地处理。但事实上，我们使用它只是为了过滤掉一些特殊的包：没有设置 SYN 位但又是
NEW 状态的 TCP 包，还有那些设置了 SYN/ACK 但也被认为是 NEW 状态的 TCP 包。这条链
可以用来检查所有可能的不一致的东西

$IPTABLES -A bad_tcp_packets -p tcp --tcp-flags SYN,ACK SYN,ACK -m state --state NEW -j
REJECT --reject-with tcp-reset
$IPTABLES -A bad_tcp_packets -p tcp ! --syn -m state --state NEW -j LOG --log-prefix "New not
syn:"
$IPTABLES -A bad_tcp_packets -p tcp ! --syn -m state --state NEW -j DROP
#
# allowed 链?allowed chain
#
$IPTABLES -A allowed -p TCP --syn -j ACCEPT
$IPTABLES -A allowed -p TCP -m state --state ESTABLISHED,RELATED -j ACCEPT
$IPTABLES -A allowed -p TCP -j DROP
#
# TCP 规则?TCP rules
#
$IPTABLES -A tcp_packets -p TCP -s 0/0 --dport 21 -j allowed
$IPTABLES -A tcp_packets -p TCP -s 0/0 --dport 22 -j allowed
$IPTABLES -A tcp_packets -p TCP -s 0/0 --dport 80 -j allowed
$IPTABLES -A tcp_packets -p TCP -s 0/0 --dport 113 -j allowed
```

```
#
# UDP 端口?UDP ports
#
#$IPTABLES -A udp_packets -p UDP -s 0/0 --destination-port 53 -j ACCEPT
#$IPTABLES -A udp_packets -p UDP -s 0/0 --destination-port 123 -j ACCEPT
$IPTABLES -A udp_packets -p UDP -s 0/0 --destination-port 2074 -j ACCEPT
$IPTABLES -A udp_packets -p UDP -s 0/0 --destination-port 4000 -j ACCEPT
#
# 如果网络中存在 Microsoft 网络的话，你会遭遇洪水一样的广播信息，下面的指令将阻止这些广播并在日志中#记录．?In Microsoft Networks you will be swamped by broadcasts. These lines
# will prevent them from showing up in the logs.
#
#$IPTABLES -A udp_packets -p UDP -i $INET_IFACE -d $INET_BROADCAST
--destination-port?135:139 -j DROP
#
# 如果有来自我们网络之外的 DHCP 请求的话，就会很快把我们的日志塞满，下面的指令 If we get DHCP requests from the Outside of our network, our logs will
# be swamped as well. This rule will block them from getting logged.
#
#$IPTABLES -A udp_packets -p UDP -i $INET_IFACE -d 255.255.255.255 --destination-port 67:68
-j DROP
#
#ICMP 规则?ICMP rules
#
$IPTABLES -A icmp_packets -p ICMP -s 0/0 --icmp-type 8 -j ACCEPT
$IPTABLES -A icmp_packets -p ICMP -s 0/0 --icmp-type 11 -j ACCEPT
#
# 4.1.4 INPUT 链?INPUT chain
#
#
# 排除不良 TCP 包?Bad TCP packets we don't want.
#
$IPTABLES -A INPUT -p tcp -j bad_tcp_packets
#
# 非 internet 网络部分的规则?Rules for special networks not part of the Internet
#
$IPTABLES -A INPUT -p ALL -i $LAN_IFACE -s $LAN_IP_RANGE -j ACCEPT
$IPTABLES -A INPUT -p ALL -i $LO_IFACE -s $LO_IP -j ACCEPT
$IPTABLES -A INPUT -p ALL -i $LO_IFACE -s $LAN_IP -j ACCEPT
$IPTABLES -A INPUT -p ALL -i $LO_IFACE -s $INET_IP -j ACCEPT
#
# 有关本地 DHCP 的特殊规则?Special rule for DHCP requests from LAN, which are not caught properly
# otherwise.
#
$IPTABLES -A INPUT -p UDP -i $LAN_IFACE --dport 67 --sport 68 -j ACCEPT
#
# 来自因特网的进入包的规则　Rules for incoming packets from the internet.
#
$IPTABLES -A INPUT -p ALL -d $INET_IP -m state --state ESTABLISHED,RELATED -j
ACCEPT
$IPTABLES -A INPUT -p TCP -i $INET_IFACE -j tcp_packets
```

```
$IPTABLES -A INPUT -p UDP -i $INET_IFACE -j udp_packets
$IPTABLES -A INPUT -p ICMP -i $INET_IFACE -j icmp_packets
#
# 如果防火墙外存在 Microsoft 网络的话，你会遭遇洪水一样的多播信息，下面的指令将丢弃这些包，所以日志就不会被这些东西淹没# 记录 If you have a Microsoft Network on the outside of your firewall, you may
# also get flooded by Multicasts. We drop them so we do not get flooded by
# logs
#
#$IPTABLES -A INPUT -i $INET_IFACE -d 224.0.0.0/8 -j DROP
#
# 将不满足上述规则的形为怪异的包记录在案　Log weird packets that don't match the above.
#
$IPTABLES -A INPUT -m limit --limit 3/minute --limit-burst 3 -j LOG --log-level DEBUG
--log-prefix "IPT INPUT packet died: "
#
# 4.1.5 FORWARD 链?FORWARD chain
#
#
# 排除不良 TCP 包?Bad TCP packets we don't want
#
$IPTABLES -A FORWARD -p tcp -j bad_tcp_packets
#
# 接收想要转发的 TCP 包?Accept the packets we actually want to forward
#
$IPTABLES -A FORWARD -i $LAN_IFACE -j ACCEPT
$IPTABLES -A FORWARD -m state --state ESTABLISHED,RELATED -j ACCEPT
#
# 将不满足上述规则的形为怪异的包记录在案?Log weird packets that don't match the above.
#
$IPTABLES -A FORWARD -m limit --limit 3/minute --limit-burst 3 -j LOG --log-level DEBUG
--log-prefix "IPT FORWARD packet died: "
#
# 4.1.6?OUTPUT 链?OUTPUT chain
#
#
# 排除不良 TCP 包?Bad TCP packets we don't want.
#
$IPTABLES -A OUTPUT -p tcp -j bad_tcp_packets
#
# 决定允许哪个 IP 包 OUTPUT 的规则?Special OUTPUT rules to decide which IP's to allow.
#
$IPTABLES -A OUTPUT -p ALL -s $LO_IP -j ACCEPT
$IPTABLES -A OUTPUT -p ALL -s $LAN_IP -j ACCEPT
$IPTABLES -A OUTPUT -p ALL -s $INET_IP -j ACCEPT
#
# 将不满足上述规则的形为怪异的包记录在案?Log weird packets that don't match the above.
#
$IPTABLES -A OUTPUT -m limit --limit 3/minute --limit-burst 3 -j LOG --log-level DEBUG
--log-prefix "IPT OUTPUT packet died: "
```

```
######
# 4.2 nat 表?nat table
#
#
# 4.2.1  设置策略?Set policies
#
#
# 4.2.2  创建用户自定义链?Create user specified chains
#
#
# 4.2.3  在用户自定义链中建立规则?Create content in user specified chains
#
#
# 4.2.4 PREROUTING 链?PREROUTING chain
#
#
# 4.2.5 POSTROUTING 链?POSTROUTING chain
#
#
#  允许简单的 IP 转发及网络地址转换?Enable simple IP Forwarding and Network Address
Translation
#
$IPTABLES -t nat -A POSTROUTING -o $INET_IFACE -j SNAT --to-source $INET_IP
#
# 4.2.6 OUTPUT 链?OUTPUT chain
#
######
# 4.3 mangle 表?mangle table
#
#
# 4.3.1  设置策略?Set policies
#
#
# 4.3.2  创建用户自定义链?Create user specified chains
#
#
# 4.3.3  在用户自定义链中建立规则?Create content in user specified chains
#
#
# 4.3.4 PREROUTING 链?PREROUTING chain
#
#
# 4.3.5 INPUT 链?INPUT chain
#
#
# 4.3.6 FORWARD 链?FORWARD chain
#
#
# 4.3.7 OUTPUT 链?OUTPUT chain
#
```

#
4.3.8 POSTROUTING 链?POSTROUTING chain
#

初始化

```
IPTABLES -X
IPTABLES -t nat -X
IPTABLES -t mangle -X
iptables -Z
```

定义策略(默认规则)

```
iptables -P INPUT ACCEPT
iptables -P OUTPUT ACCEPT
iptables -P FORWARD ACCEPT
iptables -A INPUT -i lo -j ACCEPT
iptables -A INPUT -m state --state RELATED -j ACCEPT
```

练习

1. 设置 iptables 规则,使某几台计算机能访问本机的 web 服务,其它计算机不能访问本机的 web 服务.