-----------iptables介绍------------

防火墙在做信息包过滤决定时，有一套遵循和组成的规则，这些规则存储在专用的信息包过滤表中，而这些表集成在 Linux 内核中。信息包过滤表中，规则被分组放在我们所谓的链（chain）中。

netfilter/iptables IP 信息包过滤系统是一款功能强大的工具，可用于添加、编辑和移除规则，虽然 netfilter/iptables IP 信息包过滤系统被称为单个实体，但它实际上由两个组件netfilter 和 iptables 组成。

netfilter 组件也称为内核空间（kernelspace），是内核的一部分，由一些信息包过滤表组成，这些表包含内核用来控制信息包过滤处理的规则集。

iptables 组件是一种工具，也称为用户空间（userspace），它使插入、修改和除去信息包过滤表中的规则变得容易。iptables就是用来设置、维护和检查Linux内核的IP包过滤规则的。

iptables命令规则：iptables [-t table] command CHAIN [NUM ] match criteria -j TARGET

-----------iptables配置------------

/etc/sysconfig/iptables ----iptables的配置文件

直接修改/etc/sysconfig/iptables 使其生效的办法是修改好后先service iptables restart，然后才调用/etc/rc.d/init.d/iptables save，

这跟用命令添加规则后执行的步骤刚好相反，用命令执行增改规则是先 sava 后restart。！

-----------iptables启动,关闭-------

#设置iptables开机启动或关闭

chkconfig iptables on

chkconfig iptables off

#即时启动iptables

service iptables start

#重启iptables

service iptables restart

#即时关闭iptables

service iptables stop

#ubuntu中启动及关闭iptables ,在ubuntu中由于不存在 /etc/init.d/iptales文件，所以无法使用service等命令来启动iptables，需要用modprobe命令。

#启动iptables

modprobe ip\_tables

#关闭iptables（关闭命令要比启动复杂）

iptables -F

iptables -X

iptables -Z

iptables -P INPUT ACCEPT

iptables -P OUTPUT ACCEPT

iptables -P FORWARD ACCEPT

modprobe -r ip\_tables

依次执行以上命令即可关闭iptables，否则在执行modproble -r ip\_tables时将会提示FATAL: Module ip\_tables is in use.

-----------iptables保存与恢复---------------

#防火墙规则只在计算机处于开启状态时才有效。如果系统被重新引导，这些规则就会自动被清除并重设。

#配置文件/etc/sysconfig/iptables 文件中的规则会在服务启动或重新启动时（包括机器被重新引导时）被应用。

iptables save ---保存规则以便今后载入

iptables restore ---iptables还原

-----------iptables书写规则-----------------

#规则：iptables [-t table] command CHAIN [NUM ] match criteria -j TARGET

CHAIN: 对链进行的操作

-N：new 新建一条链

-X 删除一条用户自定义链（空链）

-F：flush 清空一条链，默认清空表中所有链

-Z：zero 清空计数器，iptables中每条规则默认有两个计数器，用于记录本条规则所匹配到的数据包的个数和本条规则所匹配到的数据包的总大小

-P：policy 定义链的默认处理策略

-E 重命名链

RULE：对规则进行的操作

-A:append 追加，在链的最后加一条规则

-I:insert 插入一条规则 一般使用-I CHAIN NUM 给规则加一个编号。

-R:replace 替换某条规则，规则被替换并不会改变顺序，必须要指定替换的规则编号：-R CHAIN NUM。

-D:delete 删除一条规则，可以输入完整规则，或者直接指定标号加以删除：-D CHAIN NUM。

LIST：查看

-L：list 列出链中的所有规则

辅助性子命令：

-n numeric 以数字的形式来显示地址，默认显示主机名称

-v verbose 显示详细信息 ，支持-vv -vvv格式，v越多，信息越详细。

-x 显示原有信息，不要做单位换算

--line-numbers 显示规则的行号

Match Creteria（匹配规则）：

基本匹配

-s，--src，--source 匹配数据包的源地址

-d，--dst，--destination 匹配数据包的目标地址

-i， 指定数据包的流入接口（逻辑接口）

-o， 指定数据包的流出接口

-p， 做协议匹配 protocol，（tcp|udp|icmp）

扩展匹配：对某一种功能的扩展

隐含扩展 ：对某一种协议的扩展

-p tcp

--sport 指定源端口

--dport 指定目的端口

--tcp-flags（SYN，ACK，FIN，PSH，URG，RST，ALL，NONE）指定TCP的标志位

需要跟两个标志位列表，如：SYN，ACK，FIN，RST SYN 第一个列表表示要检查的位，第二个

列表表示第一个列表中出现的位必须为1，未出现的必须为0

--syn 只允许新连接

-p udp 无连接协议

--sport 指定源端口

--dport 指定目的端口

-p icmp

--icmp-type echo-request，8（ping出去，请求回应，） echo-reply，0（给予回应）

显式扩展 ：额外附加的更多的匹配规则,功能性地扩展

-m state 状态检测扩展

NEW 用户发起一个全新的请求

ESTABLISHED 对一个全新的请求进行回应

RELATED 两个完整连接之间的相互关系，一个完整的连接，需要依赖于另一个完整的连接

INVALID 无法识别的状态

-m multiport

--sports 22,80,443 指定多个源端口

--dports 22，80,443 指定多个目标端口

--ports 非连续端口

-m connlimit 限定并发连接速率

！--connlimit-above 5 高于五个将拒绝

-m string 字符串匹配

--algo bm|kmp 指定算法

--string pattern

-m time 基于时间的匹配

--timestart

--timestop

--days

-j TARGET 处理动作

ACCEPT 接受

DROP 悄悄丢弃，请求端没有任何回应

REJECT 明确拒绝

SNAT 源地址转换

DNAT 目标地址转换

REDIRECT 端口重定向

LOG 将访问记录下来

-----------iptables基础操作-----------------

iptables -L -n ---查看本机关于IPTABLES的设置情况

iptables -F ---清除预设表filter中的所有规则链的规则

iptables -X ---清除预设表filter中使用者自定链中的规则

iptables save ---将现有规则保存到/etc/sysconfig/iptables文件里

service iptables restart ---iptables配置的修改,需要重启防火墙

iptables -P INPUT DROP ---定义过滤政策，也就是未符合过滤条件之封包，预设的处理方式。对于INPUT链,也就是流入的数据包,采取DROP

iptables -P OUTPUT ACCEPT ---定义过滤政策，也就是未符合过滤条件之封包，预设的处理方式。对于OUTPUT链,也就是流出的数据包,不用做太多限制,而是采取ACCEPT

iptables -P FORWARD DROP ---定义过滤政策，也就是未符合过滤条件之封包，预设的处理方式。对于FORWARD链,也就是流入通过的数据包的转发,采取DROP

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

---开启22端口可访问，数据包可传入

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

---开启22端口数据包可传出，如果预设规则iptables -P OUTPUT DROP存在,就需要写上这部分，22端口的包才能传出去

iptables -A INPUT -p tcp --dport 53 -j ACCEPT

---做了DNS服务器,开启53端口

iptables -A OUTPUT -p icmp -j ACCEPT (OUTPUT设置成DROP的话)

iptables -A INPUT -p icmp -j ACCEPT (INPUT设置成DROP的话)

---允许icmp包通过,也就是允许ping

iptables -A INPUT -i lo -p all -j ACCEPT (如果是INPUT DROP)

iptables -A OUTPUT -o lo -p all -j ACCEPT(如果是OUTPUT DROP)

---允许loopback!(不然会导致DNS无法正常关闭等问题)

iptables -A INPUT -s 192.168.0.3 -p tcp --dport 22 -j ACCEPT

---只允许192.168.0.3的机器进行SSH连接

iptables -A INPUT -s 192.168.0.0/24 -p tcp --dport 22 -j ACCEPT

---只允许192.168.0.1-255的机器进行SSH连接

iptables -A FORWARD -i eth1 -o eh0 -j ACCEPT

---允许eth1到eho网卡数据包的转发

iptables -A FORWARD -f -m limit --limit 100/s --limit-burst 100 -j ACCEPT

---处理IP碎片数量,防止攻击,允许每秒100个

iptables -A FORWARD -p icmp -m limit --limit 1/s --limit-burst 10 -j ACCEPT

---设置ICMP包过滤,允许每秒1个包,限制触发条件是10个包

iptables -A FORWARD -p TCP ! --syn -m state --state NEW -j DROP

---丢弃坏的TCP包

iptables -t nat -L ---查看本机关于NAT的设置情况

iptables -t nat -A PREROUTING -i eth0 -s 10.0.0.0/8 -j DROP

iptables -t nat -A PREROUTING -i eth0 -s 192.168.0.0/16 -j DROP

iptables -t nat -A PREROUTING -i eth0 -s 172.16.0.0/12 -j DROP

---防止外网用内网IP欺骗,实际上就是来源ip是172.16.\*\*,192.168.\*\*等都丢弃

iptables -t nat -A PREROUTING -d 211.101.46.253 -j DROP

---禁止与211.101.46.253的所有连接

iptables -t nat -A PREROUTING -p tcp --dport 21 -j DROP

---禁用FTP(21)端口

iptables -A INPUT -m state --state INVALID -j DROP

iptables -A OUTPUT -m state --state INVALID -j DROP

iptables -A FORWARD -m state --state INVALID -j DROP

---drop非法连接

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

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

---允许所有已经建立的和相关的连接

-----------iptables实例------------

#内容过滤

iptables -I FORWARD -m string --string "360" -j DROP

iptables -I FORWARD -s 192.168.3.159 -m string --string "什么啊" -j DROP

iptables -I FORWARD -d 192.168.3.0/24 -m string --string "宽频影院" -j DROP

iptables -I FORWARD -s 192.168.3.0/24 -m string --string "色情" -j DROP

iptables -I FORWARD -p tcp --sport 80 -m string --string "广告" -j DROP

#备注应用

iptables -I FORWARD -s 192.168.3.159 -p tcp --dport 80 -j DROP -m comment --comment "the bad guy can not online"

iptables -I FORWARD -s 192.168.3.159 -m string --string "广告" -j DROP -m comment --comment "denny go to what"

#并发连接应用

## 模块 connlimit作用:连接限制,

## --connlimit-above n 限制为多少个

## --connlimit-mask n 这组主机的掩码，默认是connlimit-mask 32，即每ip

## 这个主要可以限制内网用户的网络使用，对服务器而言则可以限制每个ip发起的连接数...比较实用

##只允许每个ip同时5个80端口转发，超过的丢弃

iptables -I FORWARD -p tcp --syn --dport 80 -m connlimit --connlimit-above 5 -j DROP

##只允许每组ip同时10个80端口转发

iptables -I FORWARD -p tcp --syn --dport 80 -m connlimit --connlimit-above 10 --connlimit-mask 24 -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 192.186.1.0/24 -p tcp --syn -m connlimit --connlimit-above 15 -j DROP

iptables -A INPUT -s 192.186.1.0/24 -p tcp -m state --state ESTABLISHED,RELATED -j ACCEPT

#ip范围应用

iptables -A FORWARD -m iprange --src-range 192.168.1.5-192.168.1.124 -j ACCEPT

#每隔N个匹配

iptables -t mangle -A PREROUTING -m nth --every 10 -j DROP

#封杀BT类P2P软件

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

#配额匹配

iptables -I FORWARD -s 192.168.3.159 -p tcp --dport 80 -m quota --quota 500 -j DROP

iptables -I FORWARD -s 192.168.3.159 -p tcp --dport 80 -m quota --quota 500 -j ACCEPT

#web应用iptables规则示例

iptables --delete-chain

iptables --flush

iptables -P INPUT DROP

iptables -P FORWARD DROP

iptables -P OUTPUT DROP

##设置INPUT,FORWARD,OUTPUT链默认target为DROP，也就是外部与服务器不能通信

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

##设置当连接状态为RELATED和ESTABLISHED时，允许数据进入服务器

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

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

iptables -A INPUT -p tcp -m tcp --dport 21 -j ACCEPT

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

##设置外部客户端连接服务器端口80,22,21,873

iptables -A INPUT -i lo -j ACCEPT

##允许内部数据循回

iptables -A INPUT -p icmp -m icmp --icmp-type 8 -j ACCEPT

iptables -A INPUT -p icmp -m icmp --icmp-type 11 -j ACCEPT

##允许外部ping服务器

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

##设置状态为RELATED和ESTABLISHED的数据可以从服务器发送到外部

iptables -A OUTPUT -p udp -m udp --dport 53 -j ACCEPT

##允许服务器使用外部dns解析域名

iptables -A OUTPUT -o lo -j ACCEPT

##允许内部数据循回

iptables -A OUTPUT -p tcp -m tcp --dport 80 -j ACCEPT

##设置服务器连接外部服务器端口80

iptables -A OUTPUT -p tcp -m tcp --dport 25 -j ACCEPT

##允许服务器发送邮件

iptables -A OUTPUT -p icmp -m icmp --icmp-type 8 -j ACCEPT

iptables -A OUTPUT -p icmp -m icmp --icmp-type 11 -j ACCEPT

##允许从服务器ping外部

service iptables save

service iptables restart

上面的web应用iptables规则存为脚本iptables. sh，执行sh iptables. sh自动配置防火墙。

-----------iptables命令以及选项--------

[command]

命令 -A, --append

范例 iptables -A INPUT ...

说明 新增规则到某个规则炼中，该规则将会成为规则炼中的最后一条规则。

命令 -D, --delete

范例 iptables -D INPUT --dport 80 -j DROP

iptables -D INPUT 1

说明 从某个规则炼中删除一条规则，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 iptables -L INPUT

说明 列出某规则炼中的所有规则。

命令 -F, --flush

范例 iptables -F INPUT

说明 删除某规则炼中的所有规则。

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 定义新的规则炼。

命令 -X, --delete-chain

范例 iptables -X allowed

说明 删除某个规则炼。

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，预设的处理方式。

命令 -E, --rename-chain

范例 iptables -E allowed disallowed

说明 修改某自订规则炼的名称。

[option]

参数 -p, --protocol

范例 iptables -A INPUT -p tcp

说明 比对通讯协议类型是否相符，可以使用 ! 运算子进行反向比对，例如：-p ! tcp ，意思是指除 tcp 以外的其它类型，包含 udp、icmp ...等。如果要比对所有类型，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 用来比对封包的来源 IP，可以比对单机或网络，比对网络时请用数字来表示屏蔽，例如：-s 192.168.0.0/24，比对 IP 时也可以使用 ! 运算子进行反向比对，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 用来比对封包的目的地 IP，设定方式同上。

参数 -i, --in-interface

范例 iptables -A INPUT -i eth0

说明 用来比对封包是从哪片网卡进入，可以使用通配字符 + 来做大范围比对，例如：-i eth+ 表示所有的 ethernet 网卡，也可以使用 ! 运算子进行反向比对，例如：-i ! eth0。

参数 -o, --out-interface

范例 iptables -A FORWARD -o eth0

说明 用来比对封包要从哪片网卡送出，设定方式同上。

参数 --sport, --source-port

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

说明 用来比对封包的来源埠号，可以比对单一埠，或是一个范围，例如：--sport 22:80，表示从 22 到 80埠之间都算是符合条件，

如果要比对不连续的多个埠，则必须使用 --multiport 参数，详见后文。比对埠号时，可以使用 !运算子进行反向比对。

参数 --dport, --destination-port

范例 iptables -A INPUT -p tcp --dport 22

说明 用来比对封包的目的地埠号，设定方式同上。

参数 --tcp-flags

范例 iptables -p tcp --tcp-flags SYN,FIN,ACK SYN

说明 比对 TCP封包的状态旗号，参数分为两个部分，第一个部分列举出想比对的旗号，第二部分则列举前述旗号中哪些有被设定，未被列举的旗号必须是空的。TCP

状态旗号包括：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 进行比对。比对旗号时，可以使用 ! 运算子进行反向比对。

参数 --syn

范例 iptables -p tcp --syn

说明 用来比对是否为要求联机之 TCP 封包，与 iptables -p tcp --tcp-flags SYN,FIN,ACK SYN 的作用完全相同，如果使用 ! 运算子，可用来比对非要求联机封包。

参数 -m multiport --source-port

范例 iptables -A INPUT -p tcp -m multiport --source-port 22,53,80,110

说明 用来比对不连续的多个来源埠号，一次最多可以比对 15 个埠，可以使用 ! 运算子进行反向比对。

参数 -m multiport --destination-port

范例 iptables -A INPUT -p tcp -m multiport --destination-port 22,53,80,110

说明 用来比对不连续的多个目的地埠号，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，用来比对来源埠号和目的埠号相同的封包，设定方式同上。注意：在本范例中，如果来源端口号为 80 但目的地埠号为 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 用来比对 ICMP 的类型编号，可以使用代码或数字编号来进行比对。请打 iptables -p icmp --help 来查看有哪些代码可以用。

参数 -m limit --limit

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

说明 用来比对某段时间内封包的平均流量，上面的例子是用来比对：每小时平均流量是否超过一次 3 个封包。

除了每小时平均一次外，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比对外，设定这个参数也会在条件达成时，暂停封包的比对动作，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 用来比对瞬间大量封包的数量，上面的例子是用来比对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 用来比对封包来源网络接口的硬件地址，这个参数不能用在 OUTPUT 和 Postrouting

规则炼上，这是因为封包要送出到网卡后，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

在进行封包比对时，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 用来比对封包是否被表示某个号码，当封包被比对成功时，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 用来比对来自本机的封包，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去，

可以降低系统被骇的损失。可惜这个功能无法比对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 用来比对来自本机的封包，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 用来比对来自本机的封包，是否为某特定行程所产生的，使用时机同上。

参数 -m owner --sid-owner

范例 iptables -A OUTPUT -m owner --sid-owner 100

说明 用来比对来自本机的封包，是否为某特定联机（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 用来比对联机状态，联机状态共有四种：INVALID、ESTABLISHED、NEW 和 RELATED。

INVALID 表示该封包的联机编号（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联机。

NEW 表示该封包想要起始一个联机（重设联机或将联机重导向）。

RELATED 表示该封包是属于某个已经建立的联机，所建立的新联机。例如：FTP-DATA 联机必定是源自某个 FTP 联机。

[target]：

-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，将不再比对其它规则，直接跳往下一个规则炼（nat:postrouting）。

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联机），进行完此处理动作后，将不再比对其它规则，直接 中断过滤程序。

如：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，将不再比对其它规则，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，将会继续比对其它规则。这个功能可以用来实作通透式porxy

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炼（mangle:postrouting）。

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，当使用拨接连线时，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。例如：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 组态档，进行完此处理动作后，将会继续比对其它规则。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炼（mangle:postrouting）。

例如：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炼（filter:input 或 filter:forward）。

例如：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。透过自行开发的处理程序，可以进行其它应用，例如：计算联机费用.......等。

RETURN 结束在目前规则炼中的过滤程序，返回主规则炼继续过滤，如果把自订规则炼看成是一个子程序，那么这个动作，就相当于提早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，将会继续比对其它规则。

例如：iptables -t mangle -A PREROUTING -p tcp --dport 22 -j MARK --set-mark 2

-----------------------------------

-t table

这个选项指定命令要操作的匹配包的表。如果内核被配置为自动加载模块，这时若模块没有加载，（系统）将尝试（为该表）加载适合的模块。

这些表如下：filter，这是默认的表，包含了内建的链INPUT（处理进入的包）、FORWARD（处理通过的包）和OUTPUT（处理本地生成的包）。

nat，这个表被查询时表示遇到了产生新的连接的包，由三个内建的链构成：PREROUTING （修改到来的包）、OUTPUT（修改路由之前本地的包）、POSTROUTING（修改准备出去的包）。

mangle 这个表用来对指定的包进行修改。它有两个内建规则：PREROUTING（修改路由之前进入的包）和OUTPUT（修改路由之前本地的包）。

-A --append

在所选择的链末添加一条或更多规则。当源（地址）或者/与 目的（地址）转换为多个地址时，这条规则会加到所有可能的地址（组合）后面。

-D --delete

从所选链中删除一条或更多规则。这条命令可以有两种方法：可以把被删除规则指定为链中的序号（第一条序号为1），或者指定为要匹配的规则。

-R --replace

从选中的链中取代一条规则。如果源（地址）或者/与 目的（地址）被转换为多地址，该命令会失败。规则序号从1开始。

-I --insert

根据给出的规则序号向所选链中插入一条或更多规则。所以，如果规则序号为1，规则会被插入链的头部。这也是不指定规则序号时的默认方式。

-L --list

显示所选链的所有规则。如果没有选择链，所有链将被显示。也可以和z选项一起使用，这时链会被自动列出和归零。精确输出受其它所给参数影响。

-F --flush

清空所选链。这等于把所有规则一个个的删除。

-Z --zero

把所有链的包及字节的计数器清空。它可以和 -L配合使用，在清空前察看计数器，

-N --new-chain

根据给出的名称建立一个新的用户定义链。这必须保证没有同名的链存在。

-X --delete-chain

删除指定的用户自定义链。这个链必须没有被引用，如果被引用，在删除之前你必须删除或者替换与之有关的规则。如果没有给出参数，这条命令将试着删除每个非内建的链。

-P --policy

设置链的目标规则。

-E --rename-chain

根据用户给出的名字对指定链进行重命名，这仅仅是修饰，对整个表的结构没有影响。TARGETS参数给出一个合法的目标。

只有非用户自定义链可以使用规则，而且内建链和用户自定义链都不能是规则的目标。

-h

帮助。给出当前命令语法非常简短的说明。

-p --protocal [!]protocol

规则或者包检查（待检查包）的协议。指定协议可以是tcp、udp、icmp中的一个或者全部，也可以是数值，代表这些协议中的某一个。

当然也可以使用在/etc/protocols中定义的协议名。在协议名前加上"!"表示相反的规则。

数字0相当于所有all。Protocol all会匹配所有协议，而且这是缺省时的选项。在和check命令结合时，all可以不被使用。

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

指定源地址，可以是主机名、网络名和清楚的IP地址。mask说明可以是网络掩码或清楚的数字，在网络掩码的左边指定网络掩码左边"1"的个数，

因此，mask值为24等于255.255.255.0。在指定地址前加上"!"说明指定了相反的地址段。标志 --src 是这个选项的简写。

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

指定目标地址，要获取详细说明请参见 -s标志的说明。标志 --dst 是这个选项的简写。

-j --jump target

目标跳转,指定规则的目标；也就是说，如果包匹配应当做什么。目标可以是用户自定义链（不是这条规则所在的），某个会立即决定包的命运的专用内建目标，

或者一个扩展（参见下面的EXTENSIONS）。如果规则的这个选项被忽略，那么匹配的过程不会对包产生影响，不过规则的计数器会增加。

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

进入的（网络）接口 [!][名称],这是包经由该接口接收的可选的入口名称，包通过该接口接收（在链INPUT、FORWORD和PREROUTING中进入的包）。

当在接口名前使用"!"说明后，指的是相反的名称。如果接口名后面加上"+"，则所有以此接口名开头的接口都会被匹配。如果这个选项被忽略，会假设为"+"，那么将匹配任意接口。

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

输出接口[名称],这是包经由该接口送出的可选的出口名称，包通过该口输出（在链FORWARD、OUTPUT和POSTROUTING中送出的包）那么将匹配所有任意接口。

-f --fragment

分片,这意味着在分片的包中，规则只询问第二及以后的片。自那以后由于无法判断这种把包的源端口或目标端口（或者是ICMP类型的），这类包将不能匹配任何指定对他们进行匹配的规则。

如果"!"说明用在了"-f"标志之前，表示相反的意思。

-v --verbose

详细输出。这个选项让list命令显示接口地址、规则选项（如果有）和TOS（Type of Service）掩码。

包和字节计数器也将被显示，分别用K、M、G（前缀）表示1000、1,000,000和1,000,000,000倍（不过请参看-x标志改变它），

对于添加，插入，删除和替换命令，这会使一个或多个规则的相关详细信息被打印。

-n --numeric

数字输出。IP地址和端口会以数字的形式打印。默认情况下，程序试显示主机名、网络名或者服务（只要可用）。

-x --exact

扩展数字。显示包和字节计数器的精确值，代替用K,M,G表示的约数。这个选项仅能用于 -L 命令。

--line-numbers

当列表显示规则时，在每个规则的前面加上行号，与该规则在链中的位置相对应。

---------------iptables扩展模块----------------------------

#iptables可以使用扩展模块来进行数据包的匹配，语法就是 -m module\_name。

#如:-m tcp 的意思是使用 tcp 扩展模块的功能 (tcp扩展模块提供了 --dport, --tcp-flags, --syn等功能）

#其实只用 -p tcp 了话， iptables也会默认的使用 -m tcp 来调用 tcp模块提供的功能。

#但是 -p tcp 和 -m tcp是两个不同层面的东西，一个是说当前规则作用于 tcp 协议包，而后一是说明要使用iptables的tcp模块的功能 (--dport 等)

#以下就是含于基本包内的扩展包，而且他们大多数都可以通过在前面加上！来表示相反的意思。

#当 --protocol tcp 被指定，且其他匹配的扩展未被指定时，这些扩展被装载。它提供以下选项：

--source-port [!] [port[:port]]

源端口或端口范围指定。这可以是服务名或端口号。使用格式端口：端口也可以指定包含的（端口）范围。

如果首端口号被忽略，默认是"0"，如果末端口号被忽略，默认是"65535"，如果第二个端口号大于第一个，那么它们会被交换。这个选项可以使用 --sport的别名。

--destionation-port [!] [port:[port]]

目标端口或端口范围指定。这个选项可以使用 --dport别名来代替。

--tcp-flags [!] mask comp

匹配指定的TCP标记。第一个参数是我们要检查的标记，一个用逗号分开的列表，第二个参数是用逗号分开的标记表，是必须被设置的。

标记如下：SYN ACK FIN RST URG PSH ALL NONE。因此这条命令：iptables -A FORWARD -p tcp --tcp-flags SYN,ACK,FIN,RST SYN只匹配那些SYN标记被设置而ACK、FIN和RST标记没有设置的包。

--syn

只匹配那些设置了SYN位而清除了ACK和FIN位的TCP包。这些包用于TCP连接初始化时发出请求；

例如，大量的这种包进入一个接口发生堵塞时会阻止进入的TCP连接，而出去的TCP连接不会受到影响。

这等于 --tcp-flags SYN,RST,ACK SYN。如果"--syn"前面有"!"标记，表示相反的意思。

--tcp-option [!] number

匹配设置了TCP选项的。

#当--protocol udp 被指定，且其他匹配的扩展未被指定时，这些扩展被装载，它提供以下选项：

--source-port [!] [port:[port]]

源端口或端口范围指定。详见 TCP扩展的--source-port选项说明。

--destination-port [!] [port:[port]]

目标端口或端口范围指定。详见 TCP扩展的--destination-port选项说明。

#当protocol icmp被指定，且其他匹配的扩展未被指定时，该扩展被装载。它提供以下选项：

--icmp-type [!] typename

这个选项允许指定ICMP类型，可以是一个数值型的ICMP类型，或者是某个由命令iptables -p icmp -h所显示的ICMP类型名。

#limit模块匹配标志用一个标记桶过滤器一一定速度进行匹配，它和LOG目标结合使用来给出有限的登陆数.

#当达到这个极限值时，使用这个扩展包的规则将进行匹配.（除非使用了"!"标记）

--limit rate

最大平均匹配速率：可赋的值有'/second','/minute','/hour',or '/day'这样的单位，默认是3/hour。

--limit-burst number

待匹配包初始个数的最大值：若前面指定的极限还没达到这个数值，则概数字加1.默认值为5

#multiport模块匹配一组源端口或目标端口，最多可以指定15个端口。只能和-p tcp 或者 -p udp 连着使用。

--source-port [port[,port]]

如果源端口是其中一个给定端口则匹配

--destination-port [port[,port]]

如果目标端口是其中一个给定端口则匹配

--port [port[,port]]

若源端口和目的端口相等并与某个给定端口相等，则匹配。

#mark模块和与netfilter过滤器标记字段匹配（就可以在下面设置为使用MARK标记）。

--mark value [/mask]

匹配那些无符号标记值的包（如果指定mask，在比较之前会给掩码加上逻辑的标记）。

#owner模块试为本地生成包匹配包创建者的不同特征。只能用于OUTPUT链，而且即使这样一些包（如ICMP ping应答）还可能没有所有者，因此永远不会匹配。

--uid-owner userid

如果给出有效的user id，那么匹配它的进程产生的包。

--gid-owner groupid

如果给出有效的group id，那么匹配它的进程产生的包。

--sid-owner seessionid

根据给出的会话组匹配该进程产生的包。

#state模块，当与连接跟踪结合使用时，允许访问包的连接跟踪状态。

--state state

这里state是一个逗号分割的匹配连接状态列表。可能的状态是：INVALID表示包是未知连接，ESTABLISHED表示是双向传送的连接，NEW表示包为新的连接，

否则是非双向传送的，而RELATED表示包由新连接开始，但是和一个已存在的连接在一起，如FTP数据传送，或者一个ICMP错误。

#tos模块匹配IP包首部的8位tos（服务类型）字段（也就是说，包含在优先位中）。

--tos tos

这个参数可以是一个标准名称，（用iptables -m tos -h 察看该列表），或者数值。

#LOG模块为匹配的包开启内核记录。当在规则中设置了这一选项后，linux内核会通过printk（）打印一些关于全部匹配包的信息（诸如IP包头字段等）。

--log-level level

记录级别（数字或参看 syslog.conf⑸）。

--log-prefix prefix

在纪录信息前加上特定的前缀：最多14个字母长，用来和记录中其他信息区别。

--log-tcp-sequence

记录TCP序列号。如果记录能被用户读取那么这将存在安全隐患。

--log-tcp-options

记录来自TCP包头部的选项。

--log-ip-options

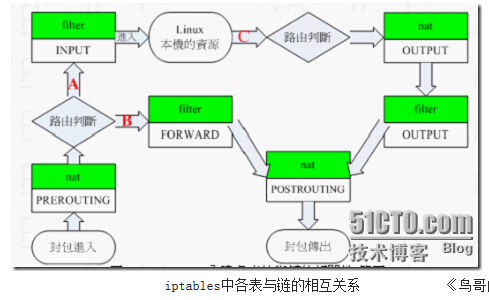
记录来自IP包头部的选项。

-------------iptables数据包传输过程--------------------------

1 当一个数据包进入网卡时，数据包首先进入PREROUTING链，内核根据数据包目的IP判断是否需要转送出去。

2 如果数据包就是进入本机的，数据包就会沿着图向下移动，到达INPUT链。数据包到达INPUT链后，任何进程都会收到它。本机上运行的程序可以发送数据包，这些数据包经过OUTPUT链，然后到达POSTROTING链输出。

3 如果数据包是要转发出去的，且内核允许转发，数据包就会向右移动，经过FORWARD链，然后到达POSTROUTING链输出。



-----------------------------------------------------------------