**Iptables笔记**

1. **关于iptables**

Linux的防火墙体系主要工作在网络层，针对TCP/IP数据包实施过滤和限制，属于典型的包过滤防火墙（或称网络层防火墙）。iptables是一个命令行防火墙实用程序，它使用策略链来允许或阻止通信。当连接试图在你的系统上建立自己时，iptables在它的列表中寻找一条规则来匹配它。如果找不到，则采取默认操作。

netfilter/iptables过滤防火墙系统是一种功能强大的工具，可用于添加、编辑和除去规则，这些规则是在做信息包过滤决定时，防火墙所遵循和组成的规则。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。在信息包过滤表中，规则被分组放在我们所谓的链（chain）中。

虽然netfilter/iptables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。

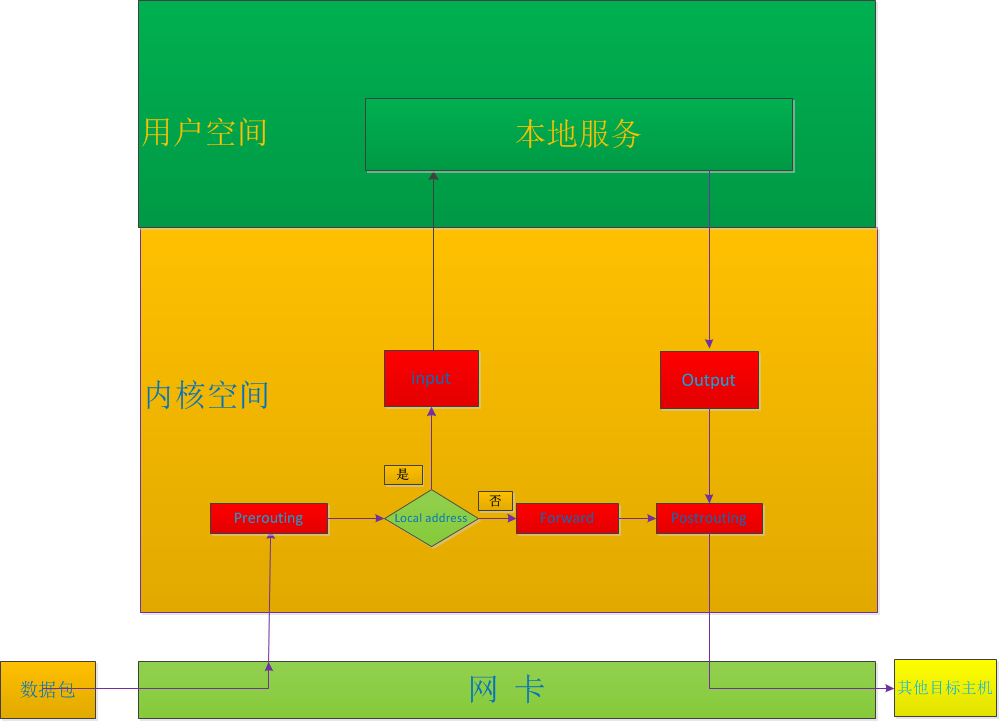
1. **iptables工作原理**

iptables的原理就是通过iptables中的规则对数据包进行控制，iptables中定义了5条链，每条链中可以定义多条规则，每当数据包到达一个链，iptables就会从链中第一条规则开始检查，看该数据包是否满足规则所定义的条件。如果满足，系统就会根据该条规则所定义的方法处理该数据包；否则iptables将继续检查下一条规则，如果该数据包不符合链中任一条规则，iptables就会根据该链预先定义的默认策略来处理数据包

一个数据包进入网卡时，它首先进入PREROUTING链，内核根据数据包目的IP判断是否需要转发出去。

如果数据包就是进入本机的，它就会沿着图向上移动，到达INPUT链。数据包到了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，这些数据包会经过OUTPUT链，然后到达POSTROUTING链输出。

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



**3.Iptables过滤流程**

1.防火墙是一层层过滤的。实际是按照配置规则的顺序从上到下，从前到后进行过滤的。

2.如果匹配上规则，即明确表是阻止还是通过，此时数据包就不在向下匹配新规则了。

3.如果所有规则中没有明确是阻止还是通过这个数据包，也就是没有匹配上新规则，向下进行匹配，直到匹配默认规则得到明确的组织还是通过

4.防火墙的默认规则是对应链的所有规则执行完才会执行的。

**4.iptables表、链、规则**

**Iptables规则**

规则就是网络管理员在iptables上预定义的条件，规则一般定义为“如果数据包头符合这样的条件，就处理这个数据包”，规则存储在内核空间的信息包过滤表中，这些规则分别指定了源地址、目的地址、传输协议（如TCP、UDP、ICMP）和服务类型（如HTTP、FTP和SMTP）等。当数据包与规则匹配时，iptables就根据规则所定义的方法来处理这些数据包，如放行（accept）、拒绝（reject）和丢弃（drop）等。配置防火墙的主要工作就是添加、修改和删除这些规则。

**Iptables链**

上述规则一般都是有很多条, 将很多条规则串起来, 即形成了(规则)链. 内核检查的时候, 按照链中的顺序, 一条条的进行检查，可以将链比做成一个检查站，每个要经过检查站的数据包，都会对其进行检查，查看数据包是否符合检查站上的规则，如果有符合条件的规则，则执行规则相应的动作。Iptables上定义了五条默认的规则链，分别为INPUT、OUTPUT、FORWARD、PREROUTING、POSTROUTING。其应用时间点分别对应如下：

INPUT链：通过路由表后目的地为本机，应用此链中的规则。

OUTPUT链：由本机产生, 向外转发时，应用此链中的规则。

FORWARD链：通过路由表后,当接收到需要通过防火墙发送给其他地址的数据包（转发）时，应用此链中的规则。

PREROUTING链：在对数据包作路由选择之前，应用此链中的规则。

POSTROUTING链：在对数据包作路由选择之后，应用此链中的规则。

**Iptables表**

我们把具有相同功能的规则的集合叫做表，所以说，不同功能的规则，我们可以放置在不同功能的表中进行管理，iptables中定义了四张表，每张表对应了不同的功能，每张表的功能如下：

Filter:一般的数据包过滤，Filter表是默认的表，如果没有指定哪个表，iptables 就默认使用filter表来执行所有命令

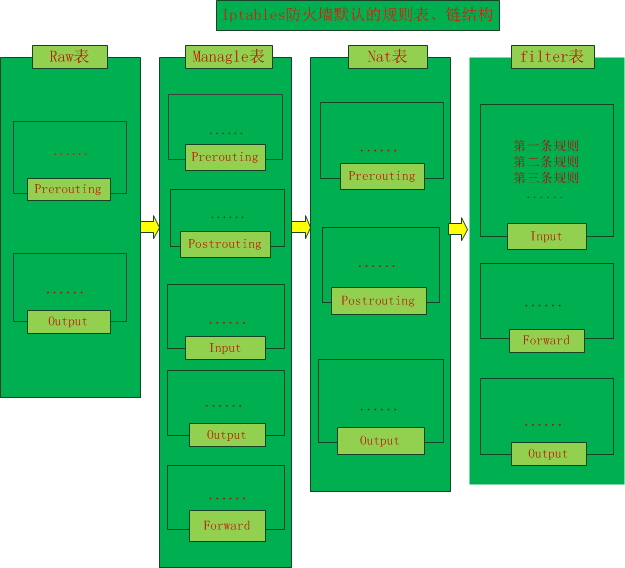
Nat:网络地址转换（端口映射，地址映射等，可以使用nat表进行局域网共享上网）

Managle:用于对特定数据包的修改（修改数据包的服务类型、TTL、并且可以配置路由实现QOS）

Raw: 决定数据包是否被状态跟踪机制处理，raw表优先级最高

表的处理优先级: raw > mangle > nat > filter

**表链关系**

上面说了，表是具有相同功能的规则的集合，而链是有多条规则串起来的，那么可以想一下，有些链是没有具备某种功能的规则的，也就是说，有些链上的规则只存在于某些表中，比如，postrouting链的规则只存在于managle表和nat表，下图是iptables表与链的对应关系图：

**5.iptables命令操作**

**Iptables命令格式**

Iptables [-t 表] [-命令参数 链] [-匹配条件 {扩展匹配}] [-j 动作]

**Iptables命令参数**

-P  --policy        <链名>  设置指定链的默认策略

-L  --list          <链名>  列出所选链中的所有规则。 如果未选择链, 则列出所有链。

-A  --append        <链名>  将一个或多个规则追加到选定链的末尾。

-I  --insert        <链名>  在指定的位置插入1条规则，如果没有指定规则号，则默认插入链的顶部

-D  --delete        <链名>  从选定的链中删除一个或多个规则。

-R  --replace       <链名>  替换选定链中的规则。

-F  --flush         <链名>  删除表中所有规则

-Z  --zero          <链名>  将所有链中的数据包和字节计数器，或者只给定的链，或者只有链中给定的规则归零。

-X  --delete-chain  <链名>  删除用户自定义的链，如果没有给出参数，删除所有非内建链

-v  --verbose       <链名>  与-L他命令一起使用显示更多更详细的信息

-C --check <链名> 检查链规则-规范检查所选链中是否存在与规范匹配的规则。此命令使用与-D相同的逻辑来查找匹配项, 但不会更改现有的 iptables 配置, 并使用其退出代码指示成功或失败。

-N --new-chain <链名> 新建一条用户自定义的规则链

-E –-rename-chain <旧链名> <新链名> 将用户指定的链重命名

--line-numbers : 显示规则的行号

-n 以数字的方式显示地址或端口信息

**Iptables匹配规则**

! 在匹配规则前面加上!表示取反

-p #指定协议类型，类型可以是tcp,udp,udplite,icmp,icmpv6,esp,sctp或者特殊关键字‘all’

--sport #指定源端口或者端口范围

--dport #指定目标端口或者端口范围

-s --src #指定数据包的源地址或源网段

-d --dst #指定数据包的目标地址或目标网段

-i #数据包进入的网卡

-o #数据包出去的网卡

-m --match #指定数据包规则所使用的模块

**Iptables扩展模块**

-m state --state #匹配状态(状态包括NEW,ESTABLISHED,RELATED,INVALID)

-m multiport --dports|--sports #表示匹配多个不连续的端口，使用dports指定目标端口号，sports指定源端口号

-m iprange --src-range 192.168.0.2-192.168.0.20 #指定一段连续的IP地址段

-m limit --limit 3/minute #指定时间内的请求速率

-m limit --limit-burst 3 #在同一时间内允许通过的请求数量

-m string --algo bm|kmp --string “xxxx” #使用string匹配字符串，algo表示设置字符匹配的查询算法，一般默认使用bm算法

-m time --timestart 8:00 --timestop 12:00 #表示某个时间段

-m time --datestart 2018-5-28T8:00 --datestop 2018-5-29T8:00 #定义一个绝对的时间范围

-m time --weekdays|monthdays 6,7|20,21 #指定每周的周几或者每月的几号

-m icmp --icmp-type type[/code] #匹配icmp协议的不同类型，可以使用iptables -m icmp -h 查看icmp协议所有的类型

可以使用‘iptables -m 扩展模块 -h’查看模块的使用帮助

**Iptables动作**

ACCEPT #允许数据包通过

DROP #丢弃数据包

REJECT #拦阻该数据包，并返回数据包通知对方，可以使用—reject-with选项设置提示信息，可以设置的提示信息有icmp-net-unreachable、icmp-host-unreachable、icmp-proto-unreachable、icmp-port-unreachable、icmp-net-prohibited、icmp-host-prohibited、icmp-admin-prohibited、tcp-reset

LOG #对符合条件的报文的相关信息记录在日志中

SNAT #源地址转换，用于局域网共享上网，隐藏内网IP

DNAT #目的地址转换，可配置外网访问内网服务

MASQUERADE #地址伪装，与SNAT不同的是，他不需要指定映射的IP，它会自动将源IP映射给外网网卡上可用的IP地址

REDIRECT #端口重定向，可以将本机的端口映射到另一个端口上

MARK #给数据包打上标记，可为后续过滤的条件判断依据

**Iptables实例**

查看iptables详细规则

iptables -nvL

禁止所有人访问22端口

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

只允许173.168.16.0网段访问22端口

iptables -I INPUT -s 173.168.16.0/24 -p tcp --dport 22 -j ACCEPT

允许关联的状态包通过

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

允许访问Samba服务的137与138这两个udp端口

iptables -A INPUT -p udp --dport 137 -j ACCEPT

iptables -A INPUT -p udp --dport 138 -j ACCEPT

禁止别人ping通自己主机

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

iptables -A OUTPUT -p icmp -m icmp --icmp-type 0 -j REJECT

数字8代表的是icmp报文的类型为8，也就是‘echo-request’，表示回显请求，

而数字0代表icmp报文的类型为0，也就是‘echo-reply’,表示回显应答，更多icmp报文的类型和编可以使用‘iptables -m icmp -h’查看，如果为了省事可以直接使用iptables -I INPUT -p icmp -j REJECT,不过这样就把所有的icmp包给拒绝了，我们也就无法ping通别人了

将INPUT链上的默认策略设置为拒绝

iptables -P INPUT DROP

将本机的220端口转发到22端口

iptables -t nat -A PREROUTING -p tcp --dport 220 -j REDIRECT --to-ports 22

将INPUT链里规则序号为2的规则删除

iptables -D INPUT 2

接受源地址为173.168.16.0/24网段的IP访问22和80端口

iptables -A INPUT -s 173.168.16.0/24 -p tcp -m multiport --dports 22,80 -j ACCEPT

对源IP为173.168.16.8的包超过10个时，限制速率为每五秒钟匹配一个数据包，并对其他未放行的数据包进行丢弃

iptables -A INPUT -s 173.168.16.8 -p tcp -m limit --limit 12/minute --limit-burst 10 -j ACCEPT

iptables -A INPUT -s 173.168.16.8 -p tcp -j DROP

这里因为我的INPUT链默认规则是ACCEPT，第一条规则是每五秒钟匹配一次数据包，那么第五秒钟之前没有匹配到的数据包如果没有对应的规则匹配，那么只能匹配INPUT链的默认规则ACCEPT，那这样限速是完全没用的，于是我们需要添加一条规则来将那些没有匹配到的数据包进行丢弃

**iptables使用nat表做局域网共享上网实例**

在一个局域网里只有一台电脑能上外网，其他的都不能，现在需要使用iptables的nat表来将局域网里的ip映射到能上外网的电脑上

环境

主机A 外网ip:60.205.177.173

内网ip:173.168.16.1

主机B 内网ip：173.168.16.2 网关：173.168.16.1

将主机B的网关修改成能上外网的主机A的ip

开启主机A的内核转发功能

临时开启

echo 1 > /proc/sys/net/ipv4/ip\_forward

永久开启

[root@localhost ~]# vim /etc/sysctl.conf

net.ipv4.ip\_forward = 1

[root@localhost ~]# sysctl -p #重新加载内核参数

在主机A添加规则如下

iptables -t nat -A POSTROUTING -o eth0 -s 173.168.16.0/24 -j SNAT --to-source 60.205.177.173

如果外网ip是动态的，可以使用MASQUERADE动作，他跟SNAT动作相似，只不过他会自动映射一个可用的外网ip,规则如下：

iptables -t nat -A POSTROUTING -o eth0 -s 173.168.16.0/24 -j MASQUERADE

**iptables使用nat表配置外网访问内网服务实例**

**内网中有一台web服务器，但是外网不能直接访问到内网，需要进行ip映射到内网才可以进行访问**

**环境配置不变，添加规则如下：**

**iptables -t nat -A PREROUTING -d 60.205.177.173 -p tcp -i eth0 --dport 800 -j DNAT --to-destination 173.168.16.2:80**

**这样就可以通过访问60.205.177.173的800端口来访问173.168.16.2上的web服务了，如果不能进行访问，可以试着将对应的SNAT规则配置一下**