防火墙

1. iptables

1.1 iptables与netfilter的关系

- `iptables`是应用层的，其实质是一个定义规则的配置工具，而核心的数据包拦截和转发是`Netfiler`

1.2 Netfilter

Netfilter是Linux操作系统核心层内部的一个数据包处理模块

1.2.1 功能

- 网络地址转换(Network Address Translate)
- 数据包内容修改
- 以及数据包过滤的防火墙功能

1.2.2 挂载点（hook point）

- Netfilter 平台中制定了数据包的五个挂载点（Hook Point，我们可以理解为回调函数点，分别是`PRE_ROUTING`、`INPUT`、`OUTPUT`、`FORWARD`、`POST_ROUTING`
- Netfilter所设置的规则（`XXtables` - Netfilter的配置表，由表`tables`、链`chains`、规则`rules`组成）是存放在内核内存中的，iptables 是一个应用层的应用程序，它通过 Netfilter 放出的接口来对存放在内核内存中的 XXtables进行修改（类似的程序还有`firewalld`）

1.2.3 规则表

1.2.3.1 filter表

主要用于对数据包进行过滤，根据具体的规则决定是否放行该数据包（如DROP、ACCEPT、REJECT、LOG）

- filter 表对应的内核模块为`iptable_filter`，包含三个规则链：
  - `INPUT`链：INPUT针对那些目的地是本地的包
  - `FORWARD`链：FORWARD过滤所有不是本地产生的并且目的地不是本地(即本机只是负责转发)的包
  - `OUTPUT`链：OUTPUT是用来过滤所有本地生成的包

1.2.3.2 net表

主要用于修改数据包的IP地址、端口号等信息（网络地址转换，如SNAT、DNAT、MASQUERADE、REDIRECT）。属于一个流的包(因为包的大小限制导致数据可能会被分成多个数据包)只会经过这个表一次。如果第一个包被允许做NAT或Masqueraded，那么余下的包都会自动地被做相同的操作，也就是说，余下的包不会再通过这个表。表对应的内核模块为`iptable_nat`，包含三个链：

- `PREROUTING`链：作用是在包刚刚到达防火墙时改变它的目的地址
- `OUTPUT`链：改变本地产生的包的目的地址
- `POSTROUTING`链：在包就要离开防火墙之前改变其源地址

1.2.3.3 mangle表

主要用于修改数据包的`TOS`（Type Of Service，服务类型）、`TTL`（Time To Live，生存周期）以及为数据包设置Mark标记，以实现`Qos`(Quality Of Service，服务质量)调整以及策略路由等应用，由于需要相应的路由设备支持，因此应用并不广泛。包含五个规则链——PREROUTING，POSTROUTING，INPUT，OUTPUT，FORWARD。

1.2.3.4 row表

是自1.2.9以后版本的iptables新增的表，主要用于决定数据包是否被状态跟踪机制处理。在匹配数据包时，raw表的规则要**优先于**其他表。包含两条规则链——OUTPUT、PREROUTING

iptables中数据包和4种被跟踪连接的4种不同状态：

- `NEW`：该包想要开始一个连接（重新连接或将连接重定向）
- `RELATED`：该包是属于某个已经建立的连接所建立的新连接。例如：FTP的数据传输连接就是控制连接所 RELATED出来的连接。`--icmp-type 0`（ping 应答）就是`--icmp-type 8`(ping 请求)所RELATED出来的。
- `ESTABLISHED`：只要发送并接到应答，一个数据连接从NEW变为ESTABLISHED,而且该状态会继续匹配这个连接的后续数据包。
- `INVALID`：数据包不能被识别属于哪个连接或没有任何状态比如内存溢出，收到不知属于哪个连接的ICMP错误信息，一般应该DROP这个状态的任何数据。

1.3 Iptables

1.3.1 规则链与规则

1.3.1.1 链

- iptables供涉及5种默认规则链，从应用时间点的角度理解这些链：

- `INPUT` 链：当接收到防火墙本机地址的数据包（入站）时，应用此链中的规则。
- `OUTPUT` 链：当防火墙本机向外发送数据包（出站）时，应用此链中的规则。
- `FORWARD` 链：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，如DNAT。
- `POSTROUTING` 链：在对数据包作路由选择之后，应用此链中的规则，如SNAT。

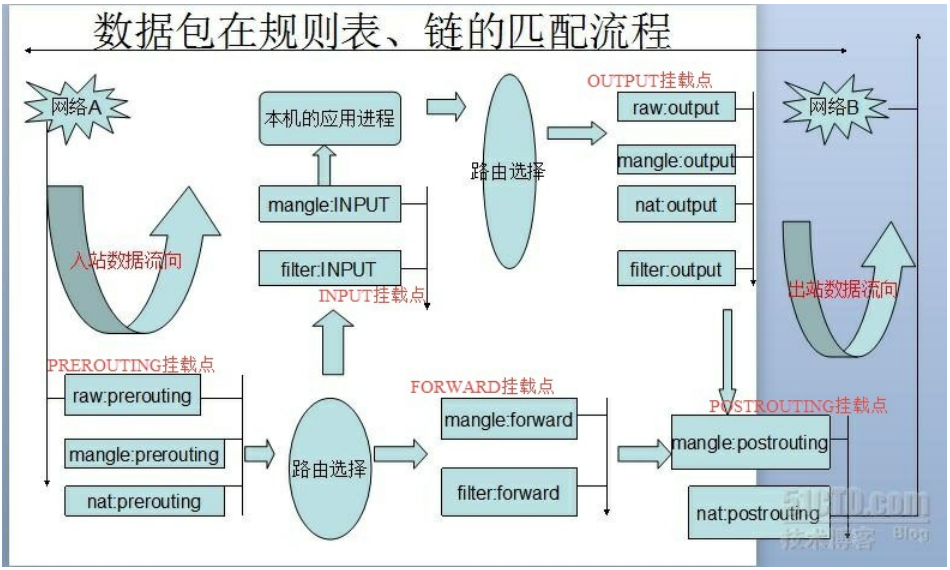
```
-->PREROUTING-->[ROUTE]-->FORWARD-->POSTROUTING-->
     mangle        |       mangle        ^ mangle
      nat          |       filter        |  nat
                   |                     |
                   |                     |
                   v                     |
                 INPUT                 OUTPUT
                   | mangle              ^ mangle
                   | filter              |  nat
                   v ------>local------->| filter
```

1.3.1.2 规则

- 除去最后一个 `LOG`，前3条规则匹配数据包后，该数据包不会再往下继续匹配了，所以编写的规则顺序极其关键
  - `ACCEPT`：允许数据包通过
  - `DROP`：直接丢弃数据包，不给任何回应信息
  - `REJECT`：拒绝数据包通过，必要时会给数据发送端一个响应的信息。
  - `SNAT`：源地址转换。在进入路由层面的route之后，出本地的网络栈之前，改写源地址，目标地址不变，并在本机建立NAT表项，当数据返回时，根据NAT表将目的地址数据改写为数据发送出去时候的源地址，并发送给主机。解决内网用户用同一个公网地址上网的问题。 `MASQUERADE`，是SNAT的一种特殊形式，适用于像adsl这种临时会变的ip上
  - `DNAT`:目标地址转换。和SNAT相反，IP包经过route之前，重新修改目标地址，源地址不变，在本机建立NAT表项，当数据返回时，根据NAT表将源地址修改为数据发送过来时的目标地址，并发给远程主机。可以隐藏后端服务器的真实地址。（感谢网友提出之前这个地方与SNAT写反了） `REDIRECT`：是DNAT的一种特殊形式，将网络包转发到本地host上（不管IP头部指定的目标地址是啥），方便在本机做端口转发。
  - `LOG`：在/var/log/messages文件中记录日志信息，然后将数据包传递给下一条规则

1.4 路由原理

网口数据包由底层的网卡NIC接收，通过数据链路层的解包之后(去除数据链路帧头)，就进入了TCP/IP协议栈(本质就是一个处理网络数据包的内核驱动)和Netfilter混合的数据包处理流程中了



1.5. 命令

1.5.1 命令格式

	table	command	chain	Parameter & Xmatch	target
iptables	-t filter nat	-A -D -L -F -P -I -R -n	INPUT FORWARD OUTPUT PREROUTING POSTROUTING	-p tcp -s -d --sport --dport --dports -m tcp state multiport	-j ACCEPT DROP REJECT DNAT SNAT

欲知如何使用，自己使用`man`命令查看；

1.6 博文参考

- Linux防火墙配置(iptables, firewalld) - 此文中有大量的示例，可参考理解

2. firewalld

> 相比iptables，firewalld的使用效能更高，处理速度更快，对用户透明，而且更加稳定

- `firewalld` - dynamic firewall

2.1 优缺点

2.1.1 优点

- 动态修改规则
  - iptables在修改规则后需要重新刷新才能生效；
- 更加人性化

2.2 原理

2.2.1 概念

- **过滤规则集合：zone** - 区域

> **一个zone就是一套过滤规则，数据包必须要经过某个zone才能入站或出站。**不同zone中规则粒度粗细、安全强度都不尽相同。可以把zone看作是一个个出站或入站必须经过的安检门，有的严格、有的宽松、有的检查的细致、有的检查的粗略。

  - 每个zone单独对应一个xml配置文件，文件名为<zone名称>.xml。自定义zone只需要添加一个<zone名称>.xml文件，然后在其中添加过滤规则即可。
  - 每个zone都有一个默认的处理行为，包括：**default**(省缺), ACCEPT, %%REJECT%%（*注：不知为和它会有百分号*）, DROP
  - firewalld提供了9个zone：
    1. drop - 任何流入的包都被丢弃，不做任何响应。只允许流出的数据包。
    2. block - 任何流入的包都被拒绝，返回icmp-host-prohibited报文(ipv4)或icmp6-

adm-prohibited报文(ipv6)。只允许由该系统初始化的网络连接
3. **public** - 默认的zone。部分公开，不信任网络中其他计算机，只放行特定服务。
4. external - 只允许选中的服务通过，用在路由器等启用伪装的外部网络。认为网路中其他计算器不可信。
5. dmz - 允许隔离区(dmz)中的电脑有限的被外界网络访问，只允许选中的服务通过。
6. work - 用在工作网络。你信任网络中的大多数计算机不会影响你的计算机，只允许选中的服务通过。
7. home - 用在家庭网络。信任网络中的大多数计算机，只允许选中的服务通过。
8. internal - 用在内部网络。信任网络中的大多数计算机，只允许选中的服务通过。
9. trusted - 允许所有网络连接，即使没有开放任何服务，那么使用此zone的流量照样通过（一路绿灯）。

2.2.2 过滤规则优先级

1. source 源地址
2. interface 接收请求的网卡
3. firewalld.conf中配置的默认zone

2.3 应用

2.3.1 配置文件位置

firewalld的配置文件以xml为主（主配置文件firewalld.conf除外），有两个存储位置：

- /etc/firewalld/ 存放修改过的配置（优先查找，找不到再找默认的配置）
- /usr/lib/firewalld/ 默认的配置

2.3.2 操作

2.3.2.1 服务

- systemctl start firewalld - 启动服务
- systemctl enable firewalld - 开机启动
- systemctl status firewalld.service - 查看服务运行
- systemctl disable firewalld - 开机启动

2.3.2.2 firewall-cmd

- `firewall-cmd` 是命令行操作firewalld的工具，详细参数和用法自己查看man

2.4 博文参考

- 用活firewalld防火墙中的zone - 通俗易懂，讲解到位