公告

昵称：私有运维
园龄：5年3个月
粉丝：1
关注：1
+加关注

linux基础理解和使用 iptables 防火墙

Posted on 2024-11-08 16:04 私有运维 阅读(118) 评论(0) 编辑 收藏 举报

本文档旨在编写一份详尽的 iptables基础 使用指南，涵盖其核心概念、使用方法以及高级技巧。将结合图表和示例，更好地理解和应用 iptables。

1. 什么是 iptables?

iptables 是 Linux 系统自带的包过滤防火墙，它与内核空间的 netfilter 框架紧密结合。netfilter 负责内核级别的包过滤，而 iptables 则提供用户空间的命令行接口，用于管理和配置 netfilter 规则。两者协同工作，实现对网络数据包的灵活控制。

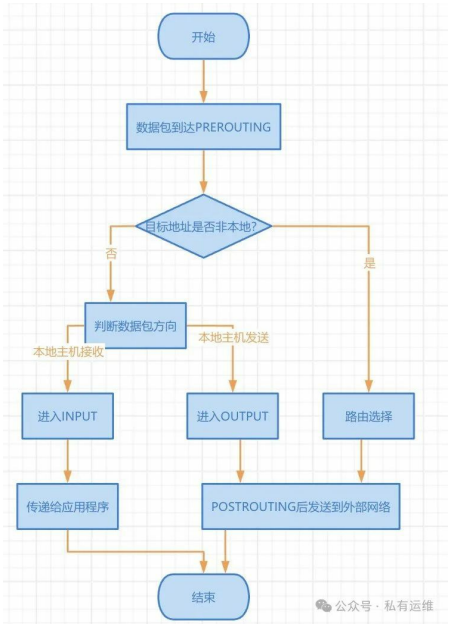
- **netfilter:** 内核空间的包过滤框架，由一系列数据包过滤表组成，这些表包含内核用于控制数据包过滤处理的规则集。它并非以程序或文件形式存在，而是内核的一部分。
- **iptables:** 用户空间的命令行工具，位于 /sbin/iptables 目录下，用于添加、删除、修改和查看 netfilter 规则。

2. 四表五链架构

iptables 的核心架构由四个表和五个链组成，它们按照特定的优先级顺序处理数据包：

表名	功能	规则链	优先级
raw	决定是否对数据包进行状态跟踪	PREROUTING, OUTPUT	最高
mangle	修改数据包的 QoS 等属性，例如 TTL、TOS 等	INPUT, OUTPUT, FORWARD, PREROUTING, POSTROUTING	高
nat	网络地址转换 (NAT)，例如 SNAT、DNAT	PREROUTING, POSTROUTING, OUTPUT	中
filter	过滤数据包，决定是否允许数据包通过	INPUT, OUTPUT, FORWARD	最低

数据包处理流程图:



五链详解:

- **INPUT:** 处理进入防火墙本机的数据包。
- **OUTPUT:** 处理从防火墙本机发出的数据包。
- **FORWARD:** 处理需要由防火墙转发到其他地址的数据包。
- **PREROUTING:** 在路由选择之前处理数据包，常用于 NAT。
- **POSTROUTING:** 在路由选择之后处理数据包，常用于 NAT。

3. 数据包过滤匹配流程和规则链内部匹配原则

iptables 按照预定义的顺序依次检查规则链中的规则。匹配到第一条符合条件的规则后，将停止后续规则的匹配并执行该规则的动作。如果遍历整个链都没有匹配的规则，则执行该链的默认策略 (通常是 ACCEPT 或 DROP)。

4. 规则编写语法

iptables 命令的基本语法如下:

```
iptables [-t table] command [chain] [match-criteria] [-j target]
```

- table: 指定操作的表 (raw, mangle, nat, filter)，默认为 filter。
- command: 操作类型，例如 -A (append), -I (insert), -D (delete), -L (list), -F (flush), -P (policy), -E (rename), -X (delete chain), -Z (zero counters), -R (replace)。
- chain: 指定操作的链 (INPUT, OUTPUT, FORWARD, PREROUTING, POSTROUTING)。
- match-criteria: 匹配条件，用于指定要处理的数据包特征，例如源 IP 地址、目标端口等。
- target: 控制类型，指定匹配数据包后的动作，例如 ACCEPT, DROP, REJECT, LOG, DNAT, SNAT, MASQUERADE, REDIRECT。

5. 匹配条件和控制类型示例

匹配条件:

- -p protocol: 指定协议 (tcp, udp, icmp 等)。
- -s source: 指定源 IP 地址或网络。
- -d destination: 指定目标 IP 地址或网络。
- --sport port: 指定源端口。
- --dport port: 指定目标端口。
- -m multiport --sports/dports port1,port2,...: 多端口匹配。
- -m iprange --src-range/dst-range start-ip-address-end-ip-address: IP 范围匹配。
- -m mac --mac-source mac-address: MAC 地址匹配。
- -m conntrack --ctstate state: 连接状态匹配 (ESTABLISHED, RELATED 等)。
- -m state --state NEW,ESTABLISHED: 状态匹配 (NEW, ESTABLISHED, RELATED 等)

控制类型:

- ACCEPT: 允许数据包通过。
- DROP: 丢弃数据包，不发送任何回应。
- REJECT: 拒绝数据包，并发送 ICMP 错误消息。
- LOG: 记录日志信息到 /var/log/messages，然后继续处理数据包。
- DNAT: 目标地址转换。
- SNAT: 源地址转换。
- MASQUERADE: 一种特殊的 SNAT，用于动态获取公网 IP 地址。
- REDIRECT: 将数据包重定向到本地端口。

6. 实际案例

(a) 允许内网 (192.168.1.0/24) 访问 SSH (端口 22):

```
iptables -A INPUT -p tcp -s 192.168.1.0/24 --dport 22 -j ACCEPT
```

(b) 丢弃所有来自 10.0.0.1 的数据包:

```
iptables -A INPUT -s 10.0.0.1 -j DROP
```

(c) 将所有指向 80 端口的流量重定向到 8080 端口:

```
iptables -t nat -A PREROUTING -p tcp --dport 80 -j REDIRECT --to-ports 8080
```

7. iptables 的管理

- **查看规则:** iptables -L -n -v (-n 使用数字显示 IP 地址和端口， -v 显示详细信息)
- **保存规则:** iptables-save > /etc/iptables/rules.v4
- **加载规则:** iptables-restore < /etc/iptables/rules.v4
- **启动/停止 iptables 服务:** 这取决于你的 Linux 发行版，例如 systemctl start/stop iptables 或 service iptables start/stop。 (注意：有些发行版使用 nftables 替代 iptables)

8. 安全注意事项

- 谨慎使用 iptables -P INPUT DROP 或 iptables -P FORWARD DROP，这会阻止所有入站或转发流量，除非你配置了允许的规则。
- 定期备份你的 iptables 规则。
- 在生产环境中测试你的规则，避免意外中断网络连接。

ps: 更高级的应用，例如 ipset 的使用、复杂的 NAT 配置以及与其他网络工具的集成，建议查阅相关文档和教程。

私有运维
粉丝 - 1 关注 - 1
+加关注

编辑推荐:
- 使用 C# 入门深度学习：线性代数
- .NET 9正式发布，亮点是.NET Aspire和AI
- 开发人员，千万不要去碰那该死的业务参数，无论什么时候！
- SQL Server 数据太多如何优化
- 带团队后的日常思考（十六）

阅读排行:
- .NET现在可以做什么，有哪些公司在用的?
- 使用 C# 入门深度学习：线性代数
- .NET 8 强大功能 IHostedService 与 BackgroundService 实战
- 在网页上调起本机C#程序
- .NET 创建动态方法方案及 Natasha V9