马哥教育
IT人的高薪职业学院

Linux防火墙

讲师：王晓春

本章内容

- 防火墙的概念
- iptables的基本认识
- iptables的组成
- iptables的基本语法
- iptables之forward的概念
- iptables之地址转换法则
- SNAT源地址转换的具体实现
- DNAT目标地址转换的具体实现
- firewalld介绍
- firewalld配置命令
- rich规则

安全技术

- 入侵检测与管理系统（Intrusion Detection Systems）：特点是不阻断任何网络访问，量化、定位来自内外网络的威胁情况，主要以提供报告和事后监督为主，提供有针对性的指导措施和安全决策依据。一般采用旁路部署方式
- 入侵防御系统（Intrusion Prevention System）：以透明模式工作，分析数据包的内容如：溢出攻击、拒绝服务攻击、木马、蠕虫、系统漏洞等进行准确的分析判断，在判定为攻击行为后立即予以阻断，主动而有效的保护网络的安全，一般采用在线部署方式
- 防火墙（FireWall）：隔离功能，工作在网络或主机边缘，对进出网络或主机的数据包基于一定的规则检查，并在匹配某规则时由规则定义的行为进行处理的一组功能的组件，基本上的实现都是默认情况下关闭所有的通过型访问，只开放允许访问的策略

防火墙的分类

- 防火墙的分类
  - 主机防火墙：服务范围为当前主机

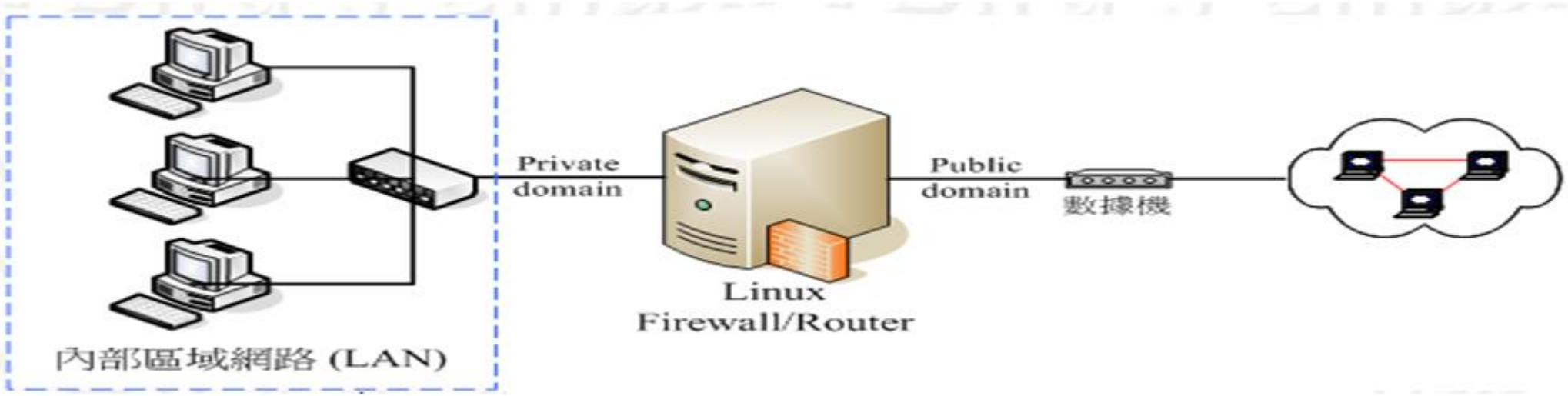
    网络防火墙：服务范围为防火墙一侧的局域网
  - 硬件防火墙：在专用硬件级别实现部分功能的防火墙；另一个部分功能基于软件实现，Checkpoint,NetScreen

    软件防火墙：运行于通用硬件平台之上的防火墙的应用软件
  - 网络层防火墙：OSI模型下四层

    应用层防火墙/代理服务器：代理网关，OSI模型七层

网络型防火墙

- 网络层防火墙
  - 包过滤防火墙
  - 网络层对数据包进行选择，选择的依据是系统内设置的过滤逻辑，被称为访问控制列表（ACL），通过检查数据流中每个数据的源地址，目的地址，所用端口号和协议状态等因素，或他们的组合来确定是否允许该数据包通过
  - 优点：对用户来说透明，处理速度快且易于维护
  - 缺点：无法检查应用层数据，如病毒等



应用层防火墙

- 应用层防火墙/代理服务型防火墙（Proxy Service）
  - 将所有跨越防火墙的网络通信链路分为两段
  - 内外网用户的访问都是通过代理服务器上的“链接”来实现
  - 优点：在应用层对数据进行检查，比较安全
  - 缺点：增加防火墙的负载

- 现实生产环境中所使用的防火墙一般都是二者结合体
  - 即先检查网络数据，通过之后再送到应用层去检查

iptables的基本认识

- Netfilter组件
  - 内核空间，集成在linux内核中
  - 扩展各种网络服务的结构化底层框架
  - 内核中选取五个位置放了五个hook(勾子) function(INPUT、OUTPUT、FORWARD、PREROUTING、POSTROUTING)，而这五个hook function向用户开放，用户可以通过一个命令工具（iptables）向其写入规则
  - 由信息过滤表（table）组成，包含控制IP包处理的规则集（rules），规则被分组放在链（chain）上
- 三种报文流向：
  - 流入本机：PREROUTING --> INPUT-->用户空间进程
  - 流出本机：用户空间进程 -->OUTPUT--> POSTROUTING
  - 转发：PREROUTING --> FORWARD --> POSTROUTING

iptables的基本认识

- 防火墙工具
- iptables
  - 命令行工具，工作在用户空间
  - 用来编写规则，写好的规则被送往netfilter，告诉内核如何去处理信息包
- firewalld

  CentOS 7 引入了新的前端管理工具

  管理工具：

  firewall-cmd 命令行

  firewall-config 图形

iptables的组成

- iptables由五个表和五个链以及一些规则组成
  - 五个表table：filter、nat、mangle、raw、security

    filter表：过滤规则表，根据预定义的规则过滤符合条件的数据包

    nat表：network address translation 地址转换规则表

    mangle：修改数据标记位规则表

    raw：关闭NAT表上启用的连接跟踪机制，加快封包穿越防火墙速度

    security：用于强制访问控制（MAC）网络规则，由Linux安全模块（如SELinux）实现

    优先级由高到低的顺序为:security -->raw-->mangle-->nat-->filter
  - 五个内置链chain

    INPUT

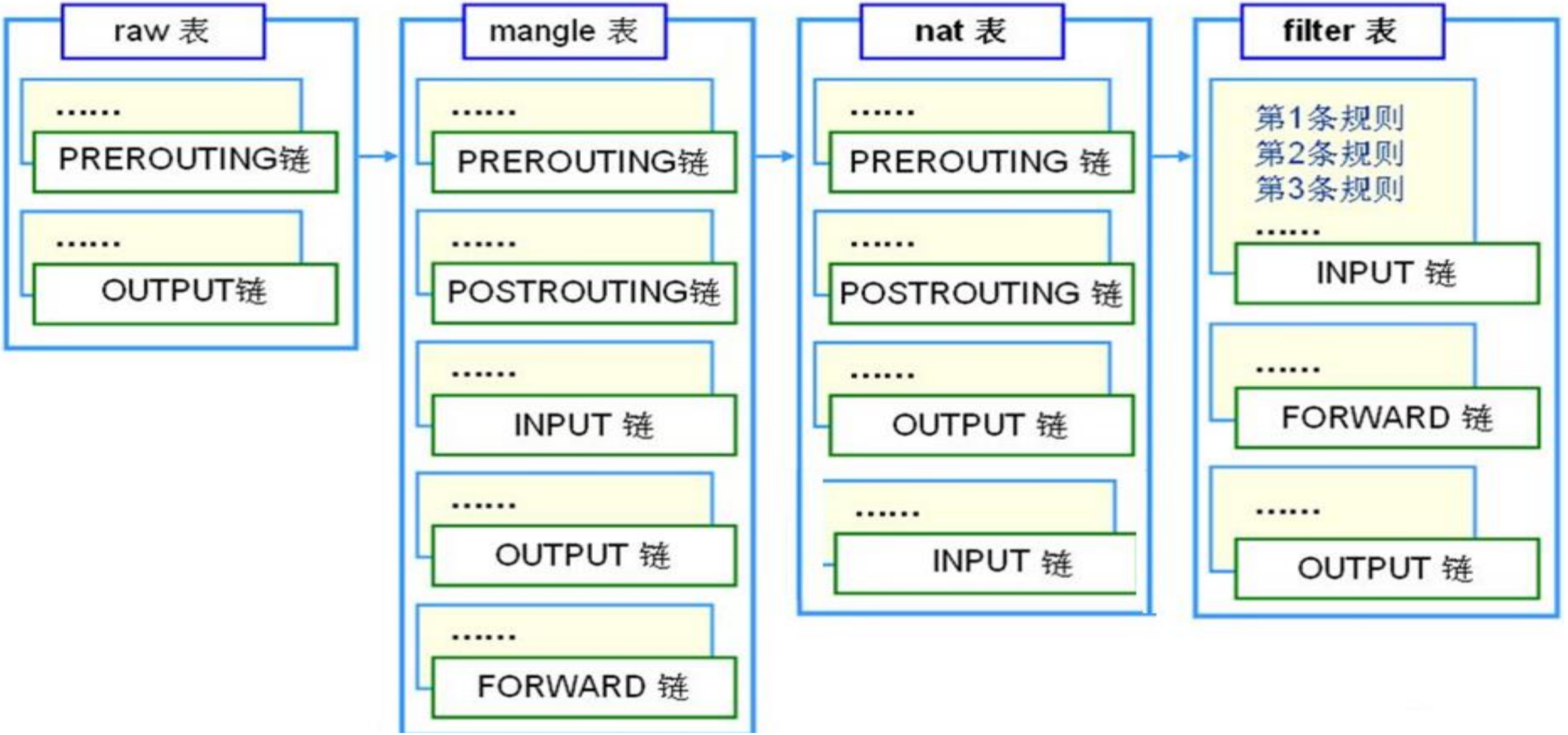
    OUTPUT

    FORWARD

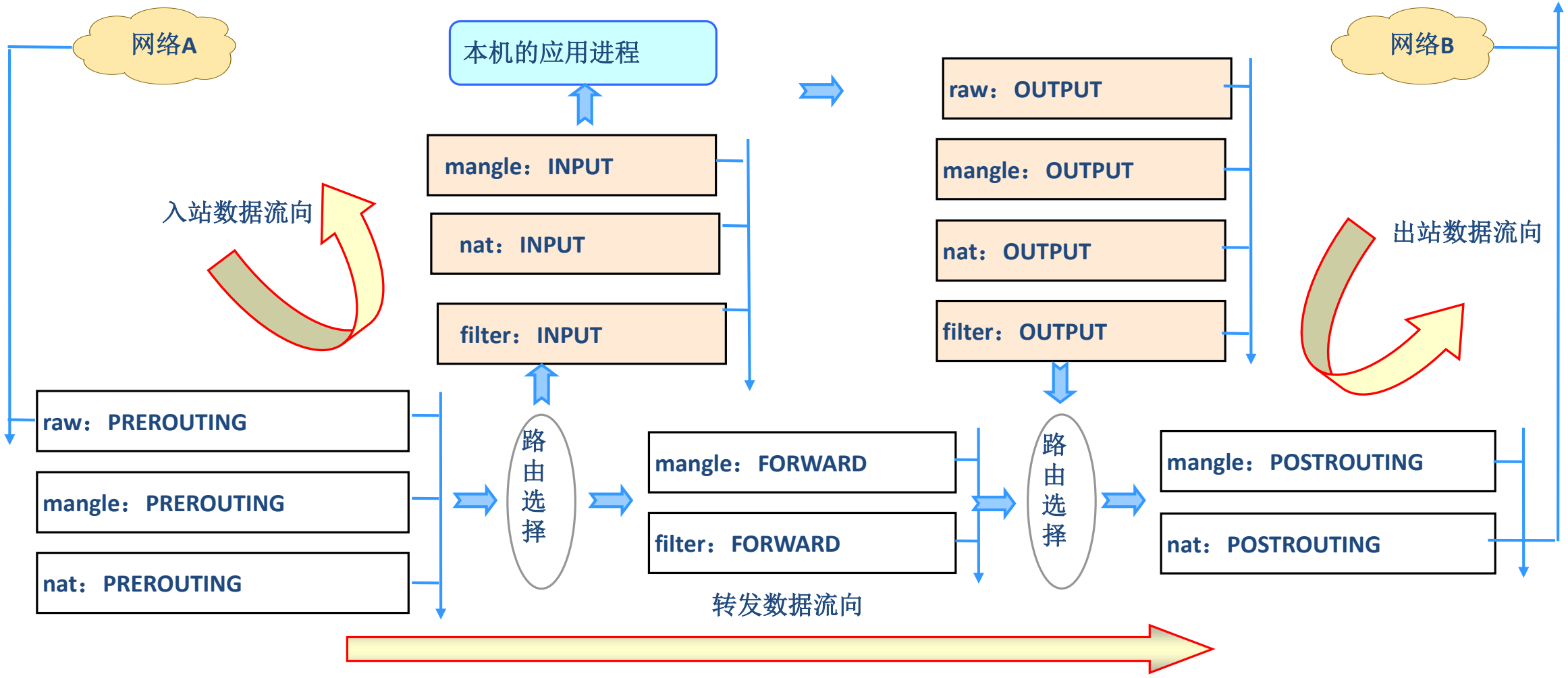
    PREROUTING

    POSTROUTING

Netfilter表和链对应关系



数据包过滤匹配流程

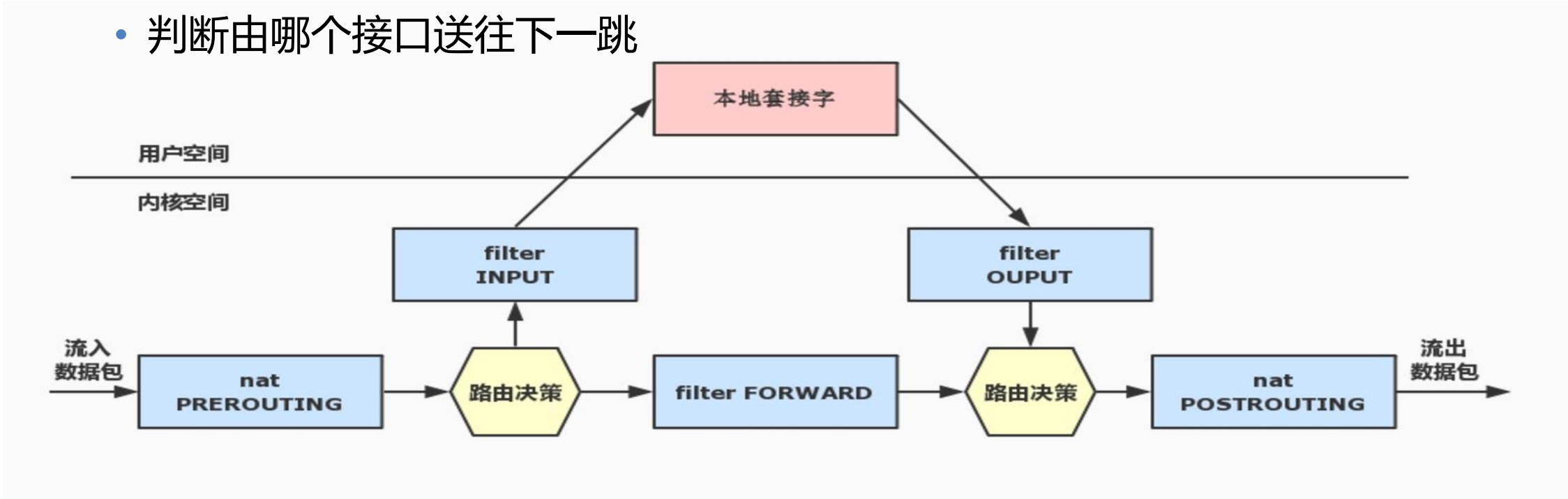

网络A
本机的应用进程
网络B
raw：OUTPUT
mangle：INPUT
mangle：OUTPUT
入站数据流向
nat：INPUT
nat：OUTPUT
出站数据流向
filter：INPUT
filter：OUTPUT
raw：PREROUTING
路由选择
mangle：FORWARD
路由选择
mangle：POSTROUTING
mangle：PREROUTING
filter：FORWARD
nat：POSTROUTING
nat：PREROUTING
转发数据流向

IPTABLES和路由

- 路由功能发生的时间点
  - 报文进入本机后
    - 判断目标主机是否为本机

      是：INPUT

      否：FORWARD
  - 报文离开本机之前
    - 判断由哪个接口送往下一跳



内核中数据包的传输过程

- 内核中数据包的传输过程
  - 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
  - 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，数据包就会向右移动，经过FORWARD链，然后到达POSTROUTING链输出

iptables规则

- 规则rule：根据规则的匹配条件尝试匹配报文，对匹配成功的报文根据规则定义的处理动作作出处理
  - 匹配条件：默认为与条件，同时满足
    - 基本匹配：IP，端口，TCP的Flags（SYN,ACK等）
    - 扩展匹配：通过复杂高级功能匹配
  - 处理动作：称为target，跳转目标
    - 内建处理动作：ACCEPT,DROP,REJECT,SNAT,DNATMASQUERADE,MARK,LOG...
    - 自定义处理动作：自定义chain，利用分类管理复杂情形
- 规则要添加在链上，才生效；添加在自定义上不会自动生效
- 链chain：
  - 内置链：每个内置链对应于一个钩子函数
  - 自定义链：用于对内置链进行扩展或补充，可实现更灵活的规则组织管理机制；只有Hook钩子调用自定义链时，才生效

iptables添加要点

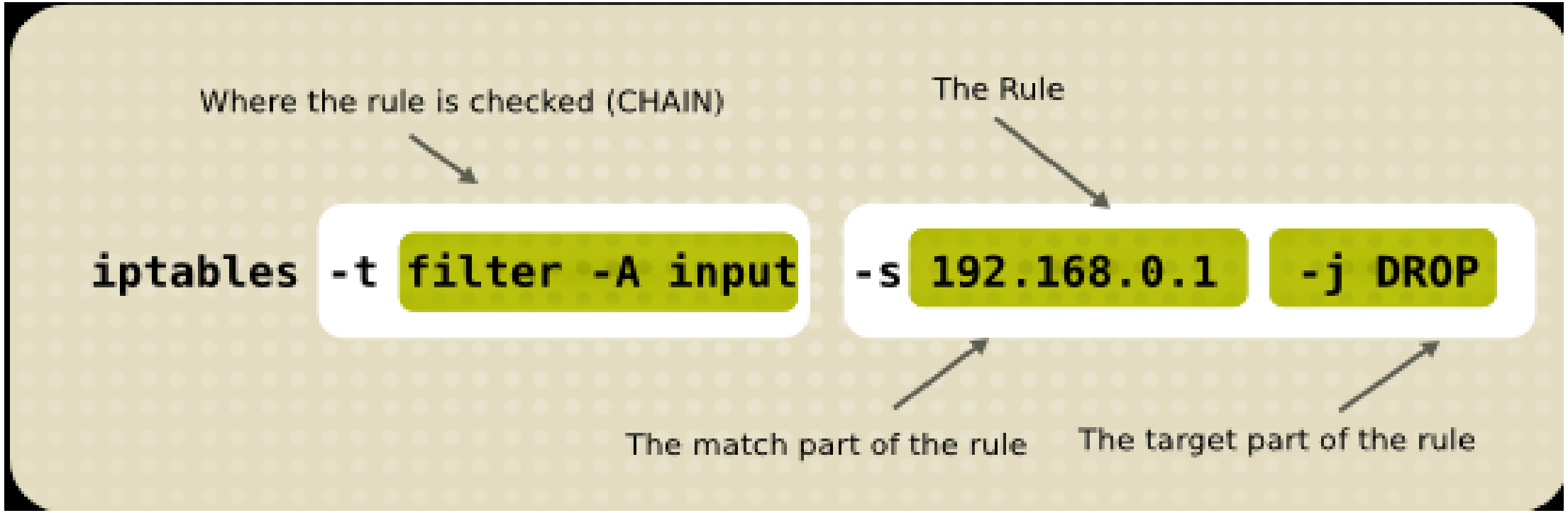
- iptables规则添加时考量点
  - 要实现哪种功能：判断添加在哪张表上
  - 报文流经的路径：判断添加在哪个链上
  - 报文的流向：判断源和目的
  - 匹配规则：业务需要
- 实验环境准备：
  - Centos7：systemctl stop firewalld.service
    systemctl disable firewalld. service
  - Centos6：service iptables stop
    chkconfig iptables off

iptables命令

- man 8 iptables
- iptables [-t table] {-A|-C|-D} chain rule-specification
- iptables [-t table] -I chain [rulenum] rule-specification
- iptables [-t table] -R chain rulenum rule-specification
- iptables [-t table] -D chain rulenum
- iptables [-t table] -S [chain [rulenum]]
- iptables [-t table] {-F|-L|-Z} [chain [rulenum]] [options...]
- iptables [-t table] -N chain
- iptables [-t table] -X [chain]
- iptables [-t table] -P chain target
- iptables [-t table] -E old-chain-name new-chain-name
- rule-specification = [matches...] [target]
- match = -m matchname [per-match-options]
- target = -j targetname [per-target-options]

简单示例

- Filter表中INPUT规则



iptables命令

- 规则格式：iptables [-t table] SUBCOMMAND chain [-m matchname [per-match-options]] -j targetname [per-target-options]
- -t table：

  raw, mangle, nat, [filter]默认
- SUBCOMMAND：
  - 1、链管理：

    -N：new, 自定义一条新的规则链

    -X：delete，删除自定义的空的规则链

    -P：Policy，设置默认策略；对filter表中的链而言，其默认策略有：

    ACCEPT：接受

    DROP：丢弃

    -E：重命名自定义链；引用计数不为0的自定义链不能够被重命名，也不能被删除

iptables命令

- 2、查看：

-L：list, 列出指定鏈上的所有规则，本选项须置后

-n：numberic，以数字格式显示地址和端口号

-v：verbose，详细信息

-vv 更详细

-x：exactly，显示计数器结果的精确值,而非单位转换后的易读值

--line-numbers：显示规则的序号

常用组合：

-vnL

-vvnxL --line-numbers

-S selected,以iptables-save 命令格式显示链上规则

iptables命令

- 3、规则管理：

  -A：append，追加

  -I：insert, 插入，要指明插入至的规则编号，默认为第一条

  -D：delete，删除

  (1) 指明规则序号

  (2) 指明规则本身

  -R：replace，替换指定链上的指定规则编号

  -F：flush，清空指定的规则链

  -Z：zero，置零

  iptables的每条规则都有两个计数器

  (1) 匹配到的报文的个数

  (2) 匹配到的所有报文的大小之和

- chain：PREROUTING，INPUT，FORWARD，OUTPUT，POSTROUTING

iptables命令

- 匹配条件

  基本：通用的，PARAMETERS

  扩展：需加载模块，MATCH EXTENTIONS

- 1、基本匹配条件：无需加载模块，由iptables/netfilter自行提供

  [!] -s, --source address[/mask][,...]：源IP地址或范围

  [!] -d, --destination address[/mask][,...]：目标IP地址或范围

  [!] -p, --protocol protocol：指定协议，可使用数字如0（all）

  protocol: tcp, udp, icmp, icmpv6, udplite,esp, ah, sctp, mh or “all “

  参看：/etc/protocols

  [!] -i, --in-interface name：报文流入的接口；只能应用于数据报文流入环节，只应用于INPUT、FORWARD、PREROUTING链

  [!] -o, --out-interface name：报文流出的接口；只能应用于数据报文流出的环节，只应用于FORWARD、OUTPUT、POSTROUTING链

iptables命令

- 2 扩展匹配条件：需要加载扩展模块（/usr/lib64/xtables/*.so），方可生效
- 查看帮助 man iptables-extensions
- (1)隐式扩展：在使用-p选项指明了特定的协议时，无需再用-m选项指明扩展模块的扩展机制，不需要手动加载扩展模块
  - tcp协议的扩展选项

    [!] --source-port, --sport port[:port]：匹配报文源端口,可为端口范围

    [!] --destination-port,--dport port[:port]：匹配报文目标端口,可为范围

    [!] --tcp-flags mask comp

    mask 需检查的标志位列表，用,分隔

    例如 SYN,ACK,FIN,RST

    comp 在mask列表中必须为1的标志位列表，无指定则必须为0，用,分隔

tcp协议的扩展选项

- 示例：

--tcp-flags SYN,ACK,FIN,RST SYN 表示要检查的标志位为SYN,ACK,FIN,RST四个，其中SYN必须为1，余下的必须为0

--tcp-flags SYN,ACK,FIN,RST SYN,ACK

--tcp-flags ALL ALL

--tcp_flags ALL NONE

- [!] --syn：用于匹配第一次握手

相当于：--tcp-flags SYN,ACK,FIN,RST SYN

iptables命令

- udp

  [!] --source-port, --sport port[:port]：匹配报文的源端口或端口范围

  [!] --destination-port,--dport port[:port]：匹配报文的目标端口或端口范围

- icmp

  [!] --icmp-type {type[/code]|typename}

  type/code

  0/0 echo-reply icmp应答

  8/0 echo-request icmp请求

- (2)显式扩展：必须使用-m选项指明要调用的扩展模块的扩展机制，要手动加载扩展模块

  [-m matchname [per-match-options]]

iptables命令

- 处理动作：
- -j targetname [per-target-options]

  简单：ACCEPT，DROP

  扩展：REJECT：--reject-with:icmp-port-unreachable默认

  RETURN：返回调用链

  REDIRECT：端口重定向

  LOG：记录日志，dmesg

  MARK：做防火墙标记

  DNAT：目标地址转换

  SNAT：源地址转换

  MASQUERADE：地址伪装

  ...

  自定义链：

iptables命令

- 显式扩展：必须显式地指明使用的扩展模块进行的扩展
- 使用帮助：

  CentOS 6: man iptables

  CentOS 7: man iptables-extensions
- 1、multiport扩展

  以离散方式定义多端口匹配,最多指定15个端口

  [!] --source-ports,--sports port[,port|,port:port]...

  指定多个源端口

  [!] --destination-ports,--dports port[,port|,port:port]...

  指定多个目标端口

  [!] --ports port[,port|,port:port]...多个源或目标端口

示例：

iptables -A INPUT -s 172.16.0.0/16 -d 172.16.100.10 -p tcp -m multiport --dports 20:22,80 -j ACCEPT

iptables命令

- 2、iprange扩展

  指明连续的（但一般不是整个网络）ip地址范围

  [!] --src-range from[-to]　　源IP地址范围

  [!] --dst-range from[-to]　　目标IP地址范围

  示例：

  iptables -A INPUT -d 172.16.1.100 -p tcp --dport 80 -m iprange --src-range 172.16.1.5-172.16.1.10 -j DROP

iptables命令

- 3、mac扩展

  指明源MAC地址

  适用于：PREROUTING, FORWARD，INPUT chains

  [!] --mac-source XX:XX:XX:XX:XX:XX

  示例：

  iptables -A INPUT -s 172.16.0.100 -m mac --mac-source 00:50:56:12:34:56 -j ACCEPT

  iptables -A INPUT -s 172.16.0.100 -j REJECT

iptables命令

- 4、string扩展

  对报文中的应用层数据做字符串模式匹配检测

  --algo {bm|kmp}　　字符串匹配检测算法

  　　bm：Boyer-Moore

  　　kmp：Knuth-Pratt-Morris

  --from offset　　开始偏移

  --to offset　　结束偏移

  [!] --string pattern　　要检测的字符串模式

  [!] --hex-string pattern要检测字符串模式，16进制格式

  示例：

  iptables -A OUTPUT -s 172.16.100.10 -d 0/0 -p tcp --sport 80 -m string --algo bm --string　“google" -j REJECT

iptables命令

- 5、time扩展

  根据将报文到达的时间与指定的时间范围进行匹配

  --datestart YYYY[-MM[-DD[Thh[:mm[:ss]]]]] 日期

  --datestop YYYY[-MM[-DD[Thh[:mm[:ss]]]]]

  --timestart hh:mm[:ss] 时间

  --timestop hh:mm[:ss]

  [!] --monthdays day[,day...] 每个月的几号

  [!] --weekdays day[,day...] 星期几，1 – 7 分别表示星期一到星期日

  --kerneltz：内核时区，不建议使用，CentOS7系统默认为UTC

  注意： centos6 不支持kerneltz ， --localtz指定本地时区(默认)

  示例：

  iptables -A INPUT -s 172.16.0.0/16 -d 172.16.100.10 -p tcp --dport 80 -m time --timestart 14:30 --timestop 18:30 --weekdays Sat,Sun --kerneltz -j DROP

iptables命令

- 6、connlimit扩展

根据每客户端IP做并发连接数数量匹配

可防止CC(Challenge Collapsar挑战黑洞)攻击

--connlimit-upto #：连接的数量小于等于#时匹配

--connlimit-above #：连接的数量大于#时匹配

通常分别与默认的拒绝或允许策略配合使用

示例：

iptables -A INPUT -d 172.16.100.10 -p tcp --dport 22 -m connlimit --connlimit-above 2 -j REJECT

iptables命令

- 7、limit扩展

  基于收发报文的速率做匹配

  令牌桶过滤器

  --limit #[/second|/minute|/hour|/day]

  --limit-burst number

  示例：

  iptables -I INPUT -d 172.16.100.10 -p icmp --icmp-type 8 -m limit --limit 10/minute --limit-burst 5 -j ACCEPT

  iptables -I INPUT 2 -p icmp -j REJECT

iptables命令

- 8、state扩展

  根据”连接追踪机制“去检查连接的状态，较耗资源

- conntrack机制：追踪本机上的请求和响应之间的关系
- 状态有如下几种：

  NEW：新发出请求；连接追踪信息库中不存在此连接的相关信息条目，因此，将其识别为第一次发出的请求

  ESTABLISHED：NEW状态之后，连接追踪信息库中为其建立的条目失效之前期间内所进行的通信状态

  RELATED：新发起的但与已有连接相关联的连接，如：ftp协议中的数据连接与命令连接之间的关系

  INVALID：无效的连接，如flag标记不正确

  UNTRACKED：未进行追踪的连接，如raw表中关闭追踪

iptables命令

- [!] --state state
- 示例：

```
iptables -A INPUT -d 172.16.1.10 -p tcp -m multiport --dports 22,80 -m state --state NEW,ESTABLISHED -j ACCEPT
iptables -A OUTPUT -s 172.16.1.10 -p tcp -m multiport --sports 22,80 -m state --state ESTABLISHED -j ACCEPT
```

- 已经追踪到的并记录下来的连接信息库

  /proc/net/nf_conntrack

- 调整连接追踪功能所能够容纳的最大连接数量

  /proc/sys/net/nf_conntrack_max

- 不同的协议的连接追踪时长

  /proc/sys/net/netfilter/

- 注意：CentOS7 需要加载模块： modprobe nf_conntrack

iptables命令

- iptables的链接跟踪表最大容量为/proc/sys/net/nf_conntrack_max，各种状态的超时链接会从表中删除；当模板满载时，后续连接可能会超时
- 解决方法两个：

(1) 加大nf_conntrack_max 值

vi /etc/sysctl.conf

net.nf_conntrack_max = 393216

net.netfilter.nf_conntrack_max = 393216

(2) 降低 nf_conntrack timeout时间

vi /etc/sysctl.conf

net.netfilter.nf_conntrack_tcp_timeout_established = 300

net.netfilter.nf_conntrack_tcp_timeout_time_wait = 120

net.netfilter.nf_conntrack_tcp_timeout_close_wait = 60

net.netfilter.nf_conntrack_tcp_timeout_fin_wait = 120

iptables -t nat -L -n

iptables命令

- 开放被动模式的ftp服务
- (1) 装载ftp连接追踪的专用模块：

  跟踪模块路径：/lib/modules/kernelversion/kernel/net/netfilter

  vim /etc/sysconfig/iptables-config 配置文件

  IPTABLES_MODULES=“nf_conntrack_ftp"

  modproble nf_conntrack_ftp
- (2) 放行请求报文：

  命令连接：NEW, ESTABLISHED

  数据连接：RELATED, ESTABLISHED

  iptables –I INPUT -d LocalIP -p tcp -m state --state ESTABLISHED,RELATED -j ACCEPT

  iptables -A INPUT -d LocalIP -p tcp --dport 21 -m state --state NEW -j ACCEPT
- (3) 放行响应报文：

  iptables -I OUTPUT -s LocalIP -p tcp -m state --state ESTABLISHED -j ACCEPT

开放被动模式的ftp服务示例

- yum install vsftpd
- systemctl start vsftpd
- modprobe nf_conntrack_ftp
- iptables -F
- iptables -A INPUT -m state --state ESTABLISHED,RELATED -j ACCEPT
- iptables -A INPUT -p tcp --dport 21 -m state --state NEW -j ACCEPT
- iptables -A OUTPUT -m state --state ESTABLISHED -j ACCEPT
- iptables -P INPUT DROP
- iptables -P OUTPUT DROP
- iptables -vnL

iptables命令

- Target：
  - ACCEPT， DROP， REJECT， RETURN
  - LOG， SNAT， DNAT， REDIRECT， MASQUERADE，..
  - LOG：非中断target,本身不拒绝和允许,放在拒绝和允许规则前
    并将日志记录在/var/log/messages系统日志中
    --log-level level 级别： debug，info，notice, warning, error, crit, alert,emerg
    --log-prefix prefix 日志前缀，用于区别不同的日志，最多29个字符
  - 示例：

```
iptables -I INPUT  -s 10.0.1.0/24 -p tcp -m multiport --dports 80,21,22,23 -m state --state NEW -j LOG --log-prefix "new connections: "
```

iptables命令

- 任何不允许的访问，应该在请求到达时给予拒绝
- 规则在链接上的次序即为其检查时的生效次序
- 基于上述，规则优化
  - 1 安全放行所有入站和出站的状态为ESTABLISHED状态连接
  - 2 谨慎放行入站的新请求
  - 3 有特殊目的限制访问功能，要在放行规则之前加以拒绝
  - 4 同类规则（访问同一应用），匹配范围小的放在前面，用于特殊处理
  - 5 不同类的规则（访问不同应用），匹配范围大的放在前面
  - 6 应该将那些可由一条规则能够描述的多个规则合并为一条
  - 7 设置默认策略，建议白名单（只放行特定连接）
    1） iptables -P，不建议
    2） 建议在规则的最后定义规则做为默认策略

iptables命令

- 规则有效期限：

  使用iptables命令定义的规则，手动删除之前，其生效期限为kernel存活期限

- 保存规则：

  保存规则至指定的文件

  CentOS 6

  ```
  service iptables save
  ```

  将规则覆盖保存至/etc/sysconfig/iptables文件中

  CentOS 7

  ```
  iptables-save > /PATH/TO/SOME_RULES_FILE
  ```

iptables命令

- CentOS 6：

  service iptables restart

  会自动从/etc/sysconfig/iptables 重新载入规则

- CentOS 7 重新载入预存规则文件中规则：

  iptables-restore < /PATH/FROM/SOME_RULES_FILE

  -n, --noflush：不清除原有规则

  -t, --test：仅分析生成规则集，但不提交

开机自动重载规则

- 开机自动重载规则文件中的规则：
- (1) 用脚本保存各iptables命令；让此脚本开机后自动运行
  /etc/rc.d/rc.local文件中添加脚本路径
  /PATH/TO/SOME_SCRIPT_FILE
- (2) 用规则文件保存各规则，开机时自动载入此规则文件中的规则
  /etc/rc.d/rc.local文件添加
  iptables-restore < /PATH/FROM/IPTABLES_RULES_FILE
- (3)自定义Unit File，进行iptables-restore

网络防火墙

- iptables/netfilter网络防火墙：
  (1) 充当网关
  (2) 使用filter表的FORWARD链

- 注意的问题：
  (1) 请求-响应报文均会经由FORWARD链，要注意规则的方向性
  (2) 如果要启用conntrack机制，建议将双方向的状态为ESTABLISHED的报文直接放行

NAT

- NAT: network address translation

  PREROUTING，INPUT，OUTPUT，POSTROUTING

  请求报文：修改源/目标IP，由定义如何修改

  响应报文：修改源/目标IP，根据跟踪机制自动实现

- SNAT：source NAT POSTROUTING, INPUT

  让本地网络中的主机通过某一特定地址访问外部网络，实现地址伪装

  请求报文：修改源IP

- DNAT：destination NAT PREROUTING , OUTPUT

  把本地网络中的主机上的某服务开放给外部网络访问(发布服务和端口映射)，但隐藏真实IP

  请求报文：修改目标IP

- PNAT：port nat，端口和IP都进行修改

SNAT

- nat表的target：
- SNAT：固定IP

  --to-source [ipaddr[-ipaddr]][:port[-port]]

  --random
- iptables -t nat -A POSTROUTING -s LocalNET ! -d LocalNet -j SNAT --to-source ExtIP
- 示例：

iptables -t nat -A POSTROUTING -s 10.0.1.0/24 ! –d 10.0.1.0/24 -j SNAT --to-source 172.18.1.6-172.18.1.9

SNAT

- MASQUERADE：动态IP，如拨号网络

  --to-ports port[-port]

  --random

- iptables -t nat -A POSTROUTING -s LocalNET ! -d LocalNet -j MASQUERADE
- 示例：

iptables -t nat -A POSTROUTING -s 10.0.1.0/24 ! –d 10.0.1.0/24 -j MASQUERADE

DNAT

- DNAT

  --to-destination [ipaddr[-ipaddr]][:port[-port]]

- iptables -t nat -A PREROUTING -d ExtIP -p tcp|udp --dport PORT -j DNAT --to-destination InterSeverIP[:PORT]

- 示例:

  iptables -t nat -A PREROUTING -s 0/0 -d 172.18.100.6 -p tcp --dport 22 -j DNAT --to-destination 10.0.1.22

  iptables -t nat -A PREROUTING -s 0/0 -d 172.18.100.6 -p tcp --dport 80 -j DNAT --to-destination 10.0.1.22:8080

转发

- REDIRECT：

  NAT表

  可用于：PREROUTING OUTPUT 自定义链

  通过改变目标IP和端口，将接受的包转发至不同端口

  --to-ports port[-port]

示例：

iptables -t nat -A PREROUTING -d 172.16.100.10 -p tcp --dport 80 -j REDIRECT --to-ports 8080

firewalld服务

- firewalld是CentOS 7.0新推出的管理netfilter的工具
- firewalld是配置和监控防火墙规则的系统守护进程。可以实现iptables,ip6tables,ebtables的功能
- firewalld服务由firewalld包提供
- firewalld支持划分区域zone,每个zone可以设置独立的防火墙规则
- 归入zone顺序：
  - 先根据数据包中源地址，将其纳为某个zone
  - 纳为网络接口所属zone
  - 纳入默认zone，默认为public zone,管理员可以改为其它zone
- 网卡默认属于public zone,lo网络接口属于trusted zone

firewalld zone分类

zone名称	默认配置
trusted	允许所有流量
home	拒绝除和传出流量相关的，以及ssh,mdsn,ipp-client,samba-client,dhcpv6-client预定义服务之外其它所有传入流量
internal	和home相同
work	拒绝除和传出流量相关的，以及ssh,ipp-client,dhcpv6-client预定义服务之外的其它所有传入流量
public	拒绝除和传出流量相关的，以及ssh,dhcpv6-client预定义服务之外的其它所有传入流量，新加的网卡默认属于public zone
external	拒绝除和传出流量相关的，以及ssh预定义服务之外的其它所有传入流量，属于external zone的传出ipv4流量的源地址将被伪装为传出网卡的地址。
dmz	拒绝除和传出流量相关的，以及ssh预定义服务之外的其它所有传入流量
block	拒绝除和传出流量相关的所有传入流量
drop	拒绝除和传出流量相关的所有传入流量（甚至不以ICMP错误进行回应）

预定义服务

服务名称	配置
ssh	Local SSH server. Traffic to 22/tcp
dhcpv6-client	Local DHCPv6 client. Traffic to 546/udp on the fe80::/64 IPv6 network
ipp-client	Local IPP printing. Traffic to 631/udp.
samba-client	Local Windows file and print sharing client. Traffic to 137/udp and 138/udp.
mdns	Multicast DNS (mDNS) local-link name resolution. Traffic to 5353/udp to the 224.0.0.251 (IPv4) or ff02::fb (IPv6) multicast addresses.

firewalld配置

- firewall-cmd --get-services 查看预定义服务列表
- /usr/lib/firewalld/services/*.xml预定义服务的配置
- 三种配置方法
  - firewall-config （firewall-config包）图形工具
  - firewall-cmd （firewalld包）命令行工具
  - /etc/firewalld 配置文件，一般不建议

firewall-cmd 命令选项

- --get-zones 列出所有可用区域
- --get-default-zone 查询默认区域
- --set-default-zone=<ZONE> 设置默认区域
- --get-active-zones 列出当前正使用的区域
- --add-source=<CIDR>[--zone=<ZONE>] 添加源地址的流量到指定区域，如果无--zone= 选项，使用默认区域
- --remove-source=<CIDR> [--zone=<ZONE>] 从指定区域中删除源地址的流量，如无--zone= 选项，使用默认区域
- --add-interface=<INTERFACE>[--zone=<ZONE>] 添加来自于指定接口的流量到特定区域，如果无--zone= 选项，使用默认区域

firewall-cmd 命令选项

- --change-interface=<INTERFACE>[--zone=<ZONE>] 改变指定接口至新的区域，如果无--zone= 选项，使用默认区域
- --add-service=<SERVICE> [--zone=<ZONE>] 允许服务的流量通过，如果无--zone= 选项，使用默认区域
- --add-port=<PORT/PROTOCOL>[--zone=<ZONE>] 允许指定端口和协议的流量，如果无--zone= 选项，使用默认区域

firewall-cmd 命令选项

- --remove-service=<SERVICE> [--zone=<ZONE>]　　从区域中删除指定服务，禁止该服务流量，如果无--zone= 选项，使用默认区域
- --remove-port=<PORT/PROTOCOL>[--zone=<ZONE>]　　从区域中删除指定端口和协议，禁止该端口的流量，如果无--zone= 选项，使用默认区域
- --reload　删除当前运行时配置，应用加载永久配置
- --list-services　查看开放的服务
- --list-ports　查看开放的端口
- --list-all [--zone=<ZONE>]　列出指定区域的所有配置信息，包括接口，源地址，端口，服务等，如果无--zone= 选项，使用默认区域

firewall-cmd 命令示例

- 查看默认zone

  firewall-cmd --get-default-zone

- 默认zone设为dmz

  firewall-cmd --set-default-zone=dmz

- 在internal zone中增加源地址192.168.0.0/24的永久规则

  firewall-cmd --permanent --zone=internal --add-source=192.168.0.0/24

- 在internal zone中增加协议mysql的永久规则

  firewall-cmd --permanent –zone=internal --add-service=mysql

- 加载新规则以生效

  firewall-cmd --reload

实验：配置firewalld

- systemctl mask iptables
- systemctl mask ip6tables
- systemctl status firewalld
- systemctl enable firewalld
- systemctl start firewalld
- firewall-cmd --get-default-zone
- firewall-cmd --set-default-zone=public
- firewall-cmd --permanent --zone=public --list-all
- firewall-cmd --permanent --zone=public --add-port 8080/tcp
- firewall-cmd ---reload

其它规则

- 当基本firewalld语法规则不能满足要求时，可以使用以下更复杂的规则
- rich-rules 富规则，功能强,表达性语言
- Direct configuration rules 直接规则，灵活性差

  帮助：man 5 firewalld.direct

管理rich规则

- rich规则比基本的firewalld语法实现更强的功能，不仅实现允许/拒绝，还可以实现日志syslog和auditd，也可以实现端口转发，伪装和限制速率
- rich语法：

  rule
    [source]
    [destination]
    service|port|protocol|icmp-block|masquerade|forward-port
    [log]
    [audit]
    [accept|reject|drop]
- man 5 firewalld.richlanguage

规则

- 规则实施顺序：
  - 该区域的端口转发，伪装规则
  - 该区域的日志规则
  - 该区域的允许规则
  - 该区域的拒绝规则
- 每个匹配的规则生效，所有规则都不匹配，该区域默认规则生效

rich规则选项

选项	描述
--add-rich-rule='<RULE>'	Add <RULE> to the specified zone, or the default zone if no zone is specified.
--remove-rich-rule='<RULE>'	Remove <RULE> to the specified zone, or the default zone if no zone is specified.
--query-rich-rule='<RULE>'	Query if <RULE> has been added to the specified zone, or the default zone if no zone is specified. Returns 0 if the rule is present, otherwise 1.
--list-rich-rules	Outputs all rich rules for the specified zone, or the default zone if no zone is specified.

rich规则示例

- 拒绝从192.168.0.11的所有流量，当address 选项使用source 或 destination 时，必须用family= ipv4 |ipv6

```
firewall-cmd --permanent --zone=classroom --add-rich-rule='rule family=ipv4 source address=192.168.0.11/32 reject ‘
```

- 限制每分钟只有两个连接到ftp服务

```
firewall-cmd --add-rich-rule= ‘rule service name=ftp limit value=2/m accept’
```

- 抛弃esp（ IPsec 体系中的一种主要协议）协议的所有数据包

```
firewall-cmd --permanent --add-rich-rule='rule protocol value=esp drop'
```

- 接受所有192.168.1.0/24子网端口5900-5905范围的TCP流量

```
firewall-cmd --permanent --zone=vnc --add-rich-rule='rule family=ipv4 source address=192.168.1.0/24 port port=5900-5905 protocol=tcp accept'
```

rich日志规则

- log [prefix="*<PREFIX TEXT>*" [level=*<LOGLEVEL>*] [limit value="*<RATE/DURATION>*"]
- *<LOGLEVEL>* 可以是emerg,alert, crit, error, warning, notice, info, debug.
- *<DURATION>* s：秒, m：分钟, h：小时, d：天
- audit [limit value="*<RATE/DURATION>*"]

rich日志规则实例

- 接受ssh新连接，记录日志到syslog的notice级别，每分钟最多三条信息

```
firewall-cmd --permanent --zone=work --add-rich-rule='rule service
name="ssh" log prefix="ssh " level="notice" limit value="3/m" accept
```

- 从2001:db8::/64子网的DNS连接在5分钟内被拒绝，并记录到日志到audit,每小时最大记录一条信息

```
firewall-cmd --add-rich-rule='rule family=ipv6 source
address="2001:db8::/64" service name="dns" audit limit value="1/h"
reject' --timeout=300
```

规则示例

- firewall-cmd --permanent --add-rich-rule='rule family=ipv4 source address=172.25.*X*.10/32 service name="http" log level=notice prefix="NEW HTTP " limit value="3/s" accept'
- firewall-cmd --reload
- tail -f /var/log/messages
- curl http://server*X*.example.com

伪装和端口转发

- NAT网络地址转换，firewalld支持伪装和端口转发两种NAT方式
- 伪装NAT
- firewall-cmd --permanent --zone= *<ZONE>* --add-masquerade
- firewall-cmd --query-masquerade 检查是否允许伪装
- firewall-cmd --add-masquerade 允许防火墙伪装IP
- firewall-cmd --remove-masquerade 禁止防火墙伪装IP

- 示例：

```
firewall-cmd --add-rich-rule='rule family=ipv4 source
address=192.168.0.0/24 masquerade'
```

端口转发

- 端口转发：将发往本机的特定端口的流量转发到本机或不同机器的另一个端口。通常要配合地址伪装才能实现
- firewall-cmd --permanent --zone=*<ZONE>* --add-forward-port=port=*<PORTNUMBER>*:proto=*<PROTOCOL>*[:toport=*<PORTNUMBER>*][:toaddr=*<IPADDR>*]

  说明：toport= 和toaddr= 至少要指定一个
- 示例：

转发传入的连接9527/TCP，到防火墙的80/TCP到public zone 的192.168.0.254

firewall-cmd --add-masquerade 启用伪装

firewall-cmd --zone=public --add-forward-port=port=9527:proto=tcp:toport=80:toaddr=192.168.0.254

rich规则

- rich规则语法：
- forward-port port= *<PORTNUM>* protocol= *tcp|udp* [to-port= *<PORTNUM>*] [to-addr= *<ADDRESS>*]
- 示例：

转发从192.168.0.0/24来的，发往80/TCP的流量到防火墙的端口8080/TCP

firewall-cmd --zone=work --add-rich-rule='rule family=ipv4 source address=192.168.0.0/24 forward-port port=80 protocol=tcp to-port=8080'

rich规则示例

- firewall-cmd --permanent --add-rich-rule 'rule family=ipv4 source address=172.25.*X*.10/32 forward-port port=443 protocol=tcp to-port=22'
- firewall-cmd --reload
- ssh -p 443 serverX.example.com

练习

- 说明：以下练习INPUT和OUTPUT默认策略均为DROP
- 1、限制本地主机的web服务器在周一不允许访问；新请求的速率不能超过100个每秒；web服务器包含了admin字符串的页面不允许访问；web服务器仅允许响应报文离开本机
- 2、在工作时间，即周一到周五的8:30-18:00，开放本机的ftp服务给172.16.0.0网络中的主机访问；数据下载请求的次数每分钟不得超过5个
- 3、开放本机的ssh服务给172.16.x.1-172.16.x.100中的主机，x为你的学号，新请求建立的速率一分钟不得超过2个；仅允许响应报文通过其服务端口离开本机
- 4、拒绝TCP标志位全部为1及全部为0的报文访问本机
- 5、允许本机ping别的主机；但不开放别的主机ping本机

练习

- 练习：判断下述规则的意义

```
iptables -N clean_in
iptables -A clean_in -d 255.255.255.255 -p icmp -j DROP
iptables -A clean_in -d 172.16.255.255 -p icmp -j DROP
iptables -A clean_in -p tcp ! --syn -m state --state NEW -j DROP
iptables -A clean_in -p tcp --tcp-flags ALL ALL -j DROP
iptables -A clean_in -p tcp --tcp-flags ALL NONE -j DROP
iptables -A clean_in -d 172.16.100.7 -j RETURN
iptables -A INPUT -d 172.16.100.7 -j clean_in
iptables -A INPUT  -i lo -j ACCEPT
iptables -A OUTPUT -o lo -j ACCEPT
iptables -A INPUT  -i eth0 -m multiport -p tcp --dports 53,113,135,137,139,445 -j DROP
iptables -A INPUT  -i eth0 -m multiport -p udp --dports 53,113,135,137,139,445 -j DROP
iptables -A INPUT  -i eth0 -m multiport -p tcp --dports 1433,4899 -j DROP
iptables -A INPUT  -p icmp -m limit --limit 10/second -j ACCEPT
```

练习

- 1 实现主机防火墙
  - 放行telnet, ftp, web服务
  - 放行samba服务
  - 放行dns服务(查询和区域传送)
- 2 实现网络防火墙
  - 放行telnet, ftp, web服务
  - 放行samba服务
  - 放行dns服务(查询和区域传送)

关于马哥教育

马哥教育
IT人的高薪职业学院

- 博客：http://mageedu.blog.51cto.com
- 主页：http://www.magedu.com
- QQ：1661815153, 113228115
- QQ群：203585050, 279599283

祝大家学业有成

谢　谢