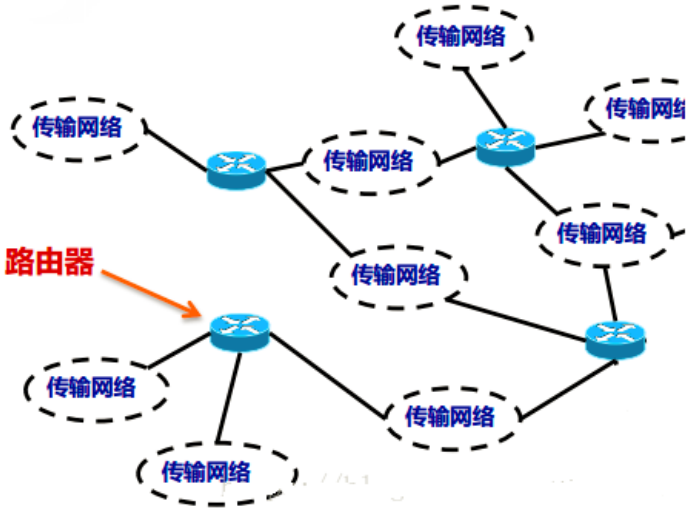
25.网络技术和防火墙基础

网络技术的基础知识

计算机网络：以实现资源共享为目的，一些互相连接的、独立自治的计算机的集合

互联网是由不同类型的传输网络互联而成的网际网



传输速率：指网络中每秒发送或接收的二进制位数，单位为比特每秒，缩写为b/s或bps，有时也称为比特率

传输速率单位：b/s或bps、 kb/s、 Mb/s、 Gb/s和Tb/s；

1kb/s=1000b/s, 1Mb/s=1000kb/s, 1Gb/s=1000Mb/s, 1Tb/s=1000Gb/s

在生活中，我们对于通信协议并不陌生，一种语言本身就是一种协议。在我们寄信或者请假时，信件或假条内容的格式就是一种协议。这样的例子很多。在计算机中，计算机网络由多台主机组成，主机之间需要不断的交换数据。要做到有条不紊的交换数据，就需要一定的或者事先约定好的通信规则。这些规则称为网络协议。

很多网络协议集合就构成了网络体系结构，这样的网络体系结构我们需要知道的有两个：

1、OSI参考模型

OSI是国际标准化组织（ISO）最早定义的网络体系结构，它的全称是**开放系统互连/参考模型**（OSI/RM）

采用分层结构来描述，将网络功能划分成7层，分别是：

物理层、 数据链路层（简称链路层）、 网络层、 传输层、 会话层、表示层和应用层

各层实现的功能：

物理层

实现二进制位流的传输过程

建立用于传播信号的信道

完成二进制位流与信号之间的转换过程

实现信号传输过程

链路层

差错控制功能

将需要传输的数据封装成分组

网络层

核心功能是路由

为分组选择正确的传输路径

传输层

实现进程间通信

数据携带进程标识符，接收端根据标识符送给不同的进程

会话层

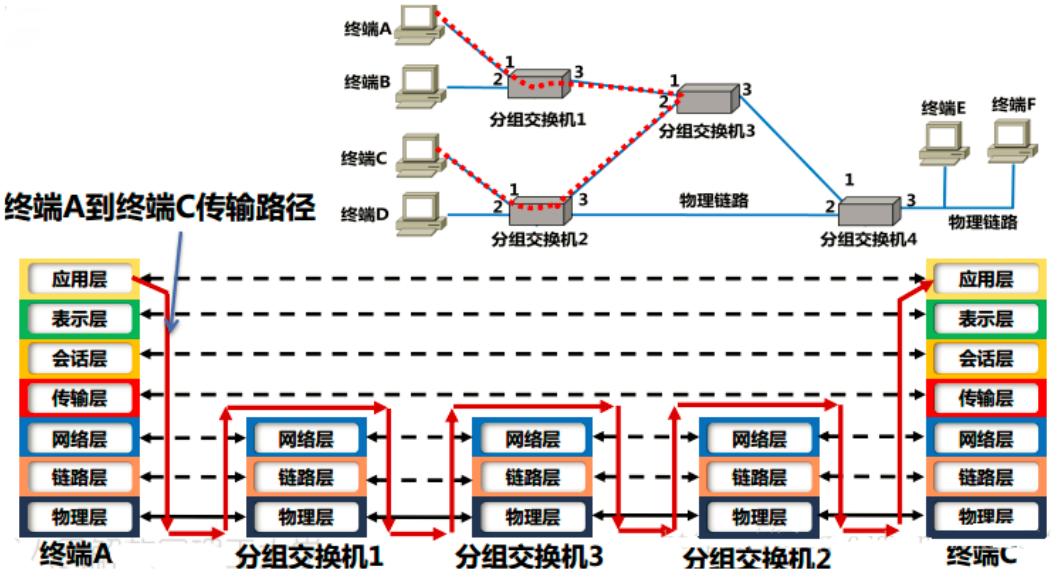
用于管理两个进程间会话的过程，e.g. 服务端和客户机上两个进程间的通信

表示层

统一通信双方描述传递信息所使用的语义和语法，e.g. 语言编码格式

应用层

定义某个应用的消息格式和实现过程



对等层传输的数据单位称为**协议数据单元**（Protocol Data Unit，PDU）

上层协议数据单元提交给下层时，作为下层的**服务数据单元**（Service Data Unit，SDU），本层在服务数据单元的基础上增加本层的协议控制信息后，产生本层的协议数据单元

2、TCP/IP体系结构

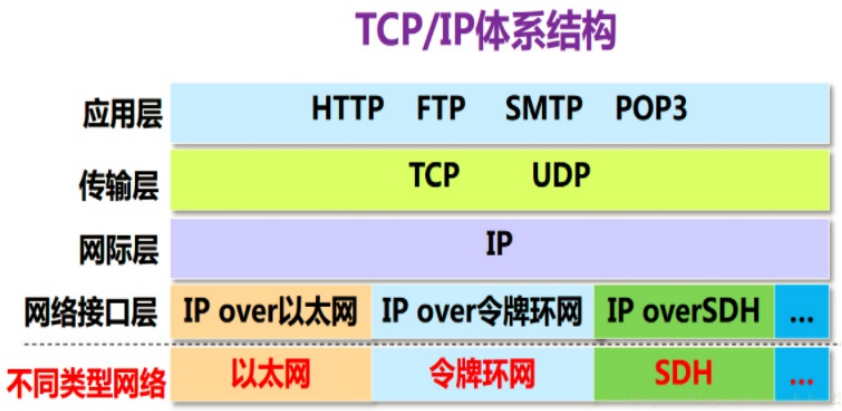
注意：因特网采用的就是TCP/IP模型，不用OSI参考模型，因特网中TCP/IP体系才是事实上的标准，原因：

1、osi模型出现的比TCP/IP出现的时间晚，在osi开始使用前，TCP/IP已经被广泛的应用了。

2、如果要换成osi模型也不太现实。osi是专家们讨论，最后形成的，没有经过实践，导致该协议实现起来很复杂，很多公司不愿意用osi，与此相比，

TCP/IP协议比较简单，实现起来也比较容易，它是从公司中产生的，更符合市场的要求。

3、综合各种因素，最终osi没有被广泛的应用。



IP地址

IP地址（Internet Protocol Address），缩写为IP Adress，是一种在Internet上的给主机统一编址的地址格式，也称为网络协议（IP协议）地址。它为互联网上的每一个网络和每一台主机分配一个逻辑地址，常见的IP地址，分为IPv4与IPv6两大类，当前广泛应用的是IPv4，目前IPv4几乎耗尽，下一阶段必然会进行版本升级到IPv6；如无特别注明，一般我们讲的的IP地址所指的是IPv4。

IP地址分类

每一个IP地址包括两部分：网络地址和主机地址

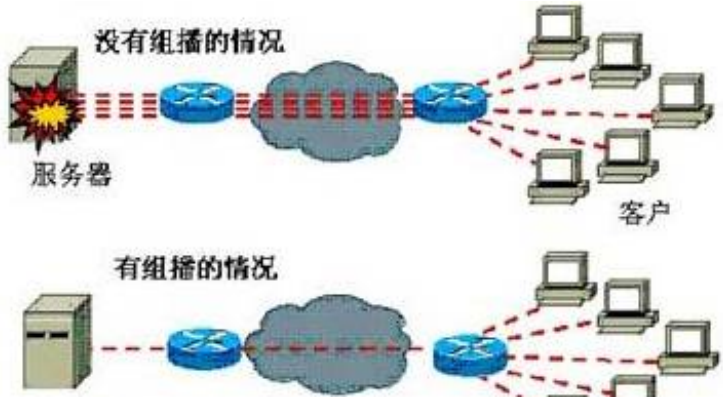
IP地址分5类，常见的地址是A、B、C 三类

A类地址:范围从0-127，0是保留的并且表示所有IP地址，而127也是保留的地址，并且是用于测试环回口用的。因此A类地址的可用的范围其实是从1-126之间。以子网掩码：255.0.0.0

B类地址：范围从128-191，如172.168.1.1，以子网掩码来进行区别：255.255.0.0

C类地址：范围从192-223，以子网掩码来进行区别： 255.255.255.0

D类地址：范围从224-239，被用在多点广播(Multicast)中。多点广播地址用来一次寻址一组计算机，它标识共享同一协议的一组计算机。



E类地址：范围从240-254，为将来使用保留。

注意： 留出了3块IP地址空间（1个A类地址段，16个B类地址段，256个C类地址段）作为私有的内部使用的地址。在这个范围内的IP地址不能被路由到Internet骨干网上；Internet路由器将丢弃该私有地址。

A类 10.0.0.0到10.255.255.255

B类 172.16.0.0到172.31.255.255

C类 192.168.0.0到192.168.255.255

端口

端口可分为虚拟端口和物理端口

其中虚拟端口指计算机内部或交换机路由器内的端口，不可见。例如计算机中的80端口、21端口、23端口等。物理端口又称为接口，是可见端口，计算机背板的RJ45网口，交换机路由器集线器等RJ45端口。

电脑运行的系统程序，其实就像一个闭合的圆圈，但是电脑是为人服务的，他需要接受一些指令，并且要按照指令调整系统功能来工作，于是系统程序设计者，就把这个圆圈截成好多段，这些线段接口就叫端口（通俗讲是断口，就是中断），系统运行到这些端口时，一看端口是否打开或关闭，如果关闭，就是绳子接通了，系统往下运行，如果端口是打开的，系统就得到命令，有外部数据输入，接受外部数据并执行。

协议端口

如果把IP地址比作一间房子，端口就是出入这间房子的门。真正的房子只有几个门，但是一个IP地址的端口可以有65536（即：2^16）个之多！端口是通过端口号来标记的，端口号只有整数，范围是从0 到65535（2^16-1）。

TCP和UDP 常用端口号名称

21	**ftp**	**文件传输服务**
22	**ssh**	**安全远程连接服务**
23	**telnet**	**远程连接服务**
25	**smtp**	**电子邮件服务**
53	**DNS**	**域名解析服务，有tcp53也有用udp53端口传输**
80	**http**	**web服务**
443	**https**	**安全web服务**

防火墙的概念和分类

从逻辑上讲。防火墙可以大体分为主机防火墙和网络防火墙。

主机防火墙：针对于单个主机进行防护。

网络防火墙：往往处于网络入口或边缘，针对于网络入口进行防护，服务于防火墙背后的本地局域网。

网络防火墙和主机防火墙并不冲突，可以理解为，网络防火墙主外（集体）， 主机防火墙主内（个人）。

从物理上讲，防火墙可以分为硬件防火墙和软件防火墙。

硬件防火墙：在硬件级别实现部分防火墙功能，另一部分功能基于软件实现，性能高，成本高。

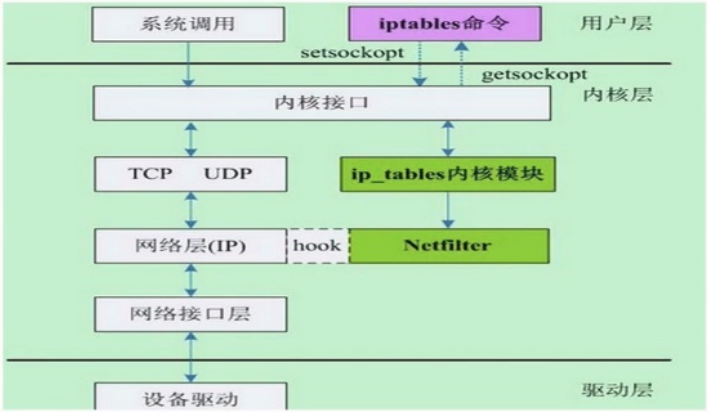
软件防火墙：应用软件处理逻辑运行于通用硬件平台之上的防火墙，性能低，成本低。

iptables防火墙（CentOS6默认）

iptables其实不是真正的防火墙，我们可以把它理解成一个客户端代理，用户通过iptables这个代理，将用户的安全设定执行到对应的"安全框架"中，这个"安全框架"才是真正的防火墙，这个框架的名字叫**netfilter**

netfilter才是防火墙真正的安全框架（framework），netfilter位于内核空间。

iptables其实是一个命令行工具，位于用户空间，我们用这个工具操作真正的框架。



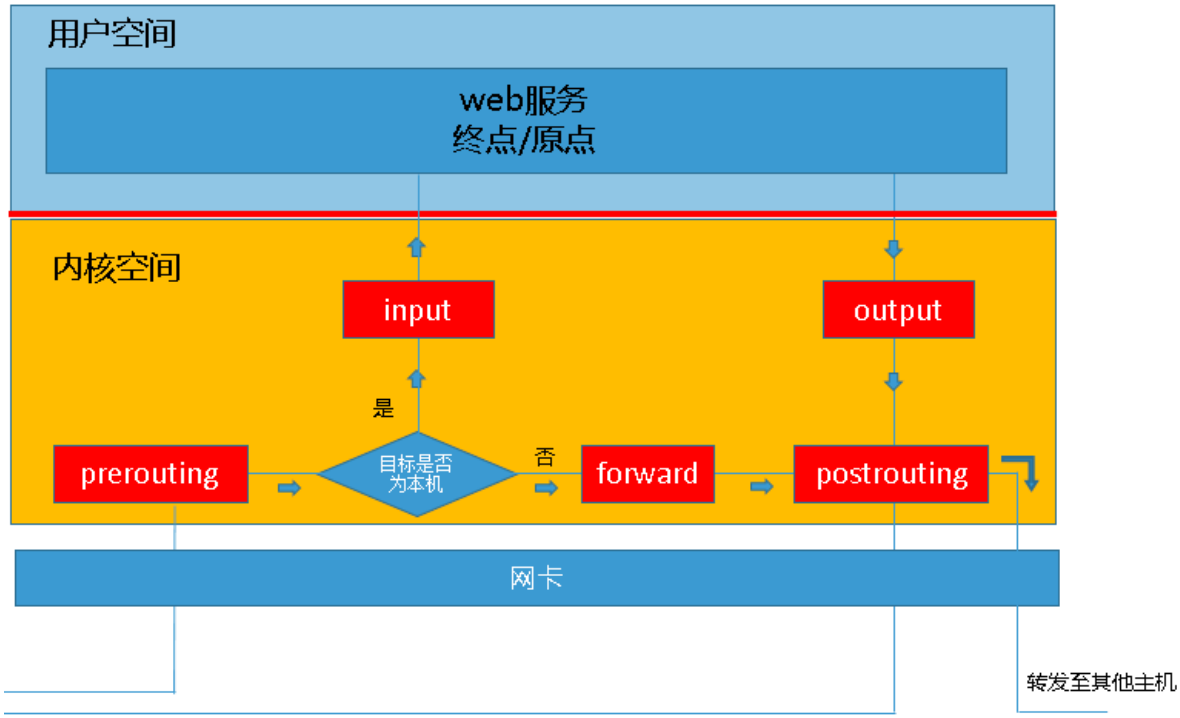
Netfilter具有如下功能：

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

所以说，虽然我们使用service iptables start启动iptables"服务"，但是其实准确的来说，iptables并没有一个守护进程，所以并不能算是真正意义上的服务，而应该算是内核提供的功能。

我们知道iptables是按照规则来办事的，规则其实就是网络管理员预定义的条件，规则一般的定义为"如果数据包头符合这样的条件，就这样处理这个数据包"。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。

当客户端访问服务器的web服务时，客户端发送报文到网卡，而tcp/ip协议栈是属于内核的一部分，所以，客户端的信息会通过内核的TCP协议传输到用户空间中的web服务中，而此时，客户端报文的目标终点为web服务所监听的套接字（IP：Port）上，当web服务需要响应客户端请求时，web服务发出的响应报文的目标终点则为客户端，这个时候，web服务所监听的IP与端口反而变成了原点，我们说过，netfilter才是真正的防火墙，它是内核的一部分，所以，如果我们想要防火墙能够达到"防火"的目的，则需要在内核中设置关卡，所有进出的报文都要通过这些关卡，经过检查后，符合放行条件的才能放行，符合阻拦条件的则需要被阻止，于是，就出现了INPUT、OUTPUT、PREROUTING、FORWARD、POSTROUTING等关卡，而这些关卡在iptables中不被称为"关卡",而被称为"链"。



根据上图，报文的流向：

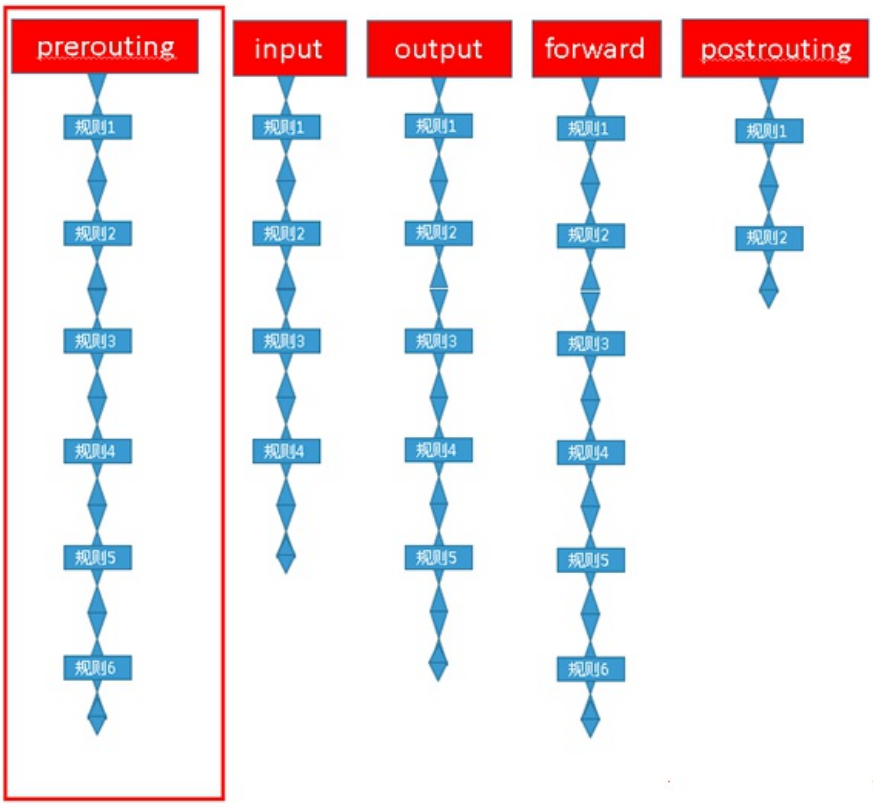
到本机某进程的报文：PREROUTING --> INPUT

由本机转发的报文：PREROUTING --> FORWARD --> POSTROUTING

由本机的某进程发出报文（通常为响应报文）：OUTPUT --> POSTROUTING

链的概念

前面讲的“关卡”为什么叫“链”，防火墙的作用就在于对经过的报文匹配"规则"，然后执行对应的"动作"，所以，当报文经过这些关卡的时候，关卡上可能不止有一条规则，而是有很多条规则，当我们把这些规则串到一个链条上的时候，就形成了"链"



表的概念

每个"链"上都放置了一串规则，但是这些规则有些很相似，比如，A类规则都是对IP或者端口的过滤，B类规则是

修改报文，那么这个时候，我们是不是能把实现相同功能的规则放在一起，我们把具有相同功能的规则的集合叫做"表"，不同功能的规则，我们可以放置在不同的表中进行管理，而iptables已经为我们定义了4种表，每种表对应了不同的功能，而我们定义的规则也都逃脱不了这4种功能的范围，iptables为我们提供了如下规则的分类，或者说，iptables为我们提供了如下"表"：

filter表：负责过滤功能，真正的防火墙；内核模块：iptables_filter

nat表：network address translation，网络地址转换功能；内核模块：iptable_nat

mangle表：拆解报文，做出修改，并重新封装 的功能；内核模块：iptable_mangle

raw表：跳过nat表上的连接追踪机制，以提高性能。如大量访问的web服务器，可以让80端口不再让iptables做数据包的链接跟踪处理，以提高用户的访问速度。内核模块：iptable_raw

表链关系

某些"链"中注定不会包含"某类规则"，就像某些"关卡"天生就不具备某些功能一样，比如，A"关卡"只负责打击陆地敌人，没有防空能力，B"关卡"只负责打击空中敌人，没有防御步兵的能力，C"关卡"可能比较NB，既能防空，也能防御陆地敌人，D"关卡"最屌，海陆空都能防。

每个"链"中的规则都存在于哪些"表"中：

PREROUTING　　的规则可以存在于：raw表，mangle表，nat表。

INPUT　　的规则可以存在于：mangle表，filter表，（centos7中还有nat表，centos6中没有）。

FORWARD　　的规则可以存在于：mangle表，filter表。

OUTPUT　　的规则可以存在于：raw表mangle表，nat表，filter表。

POSTROUTING　　的规则可以存在于：mangle表，nat表。

反过来：

表（功能）<-->　链（钩子）：

raw　表中的规则可以被哪些链使用：PREROUTING，OUTPUT

mangle　表中的规则可以被哪些链使用：PREROUTING，INPUT，FORWARD，OUTPUT，POSTROUTING

nat　表中的规则可以被哪些链使用：PREROUTING，OUTPUT，POSTROUTING（centos7中还有INPUT，centos6中没有）

filter　表中的规则可以被哪些链使用：INPUT，FORWARD，OUTPUT

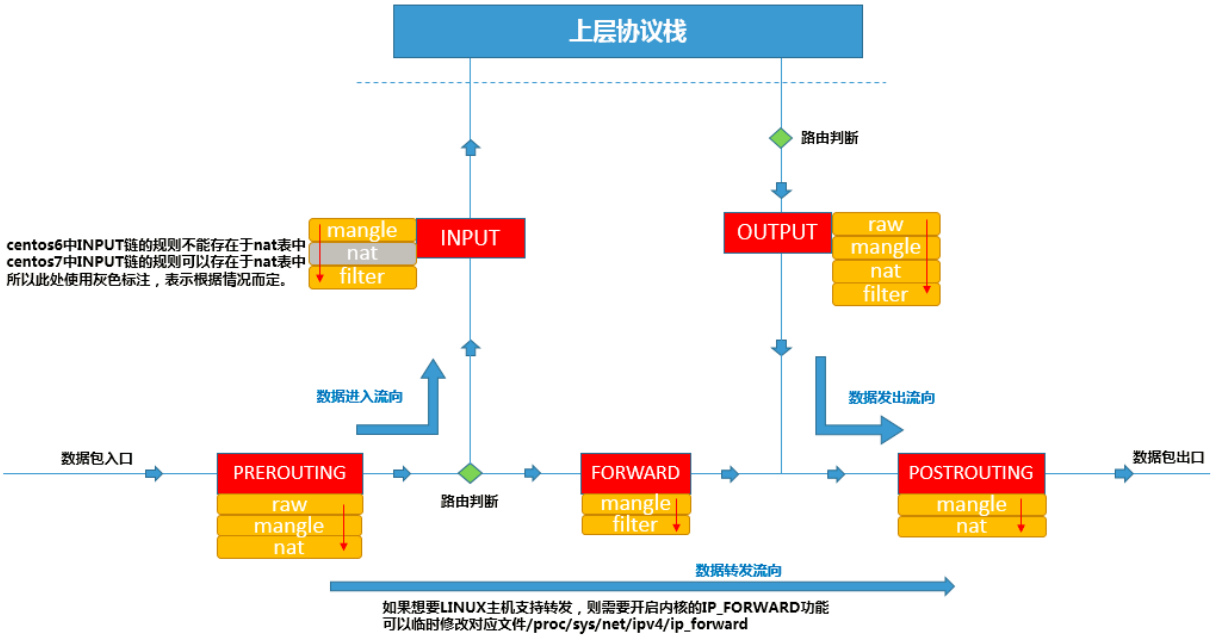
在同一条链中，的规则那些在前那些在后，是有优先规则的：

iptables为我们定义了4张"表",当他们处于同一条"链"时，执行的优先级如下。

优先级次序（由高而低）：

raw --> mangle --> nat --> filter

将数据包通过防火墙的流程总结为下图



处理动作

iptables对数据包的常见动作：

ACCEPT：允许数据包通过。

DROP：直接丢弃数据包，不给任何回应信息，这时候客户端会感觉自己的请求泥牛入海了，过了超时时间才会有反应。

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，客户端刚请求就会收到拒绝的信息。

SNAT：源地址转换，解决内网用户用同一个公网地址上网的问题，修改报文的源地址。

MASQUERADE：是SNAT的一种特殊形式，适用于动态的、会变的ip上。

DNAT：目标地址转换，修改报文的目的地址。

REDIRECT：在本机做端口映射。

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，也就是说除了记录以外不对数据包做任何其他操作。

iptables的命令组成

	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

常用命令：

-A 追加规则-->iptables -A INPUT 在所有规则最后加入规则

-D 删除规则-->iptables -D INPUT 1(编号)

-R 修改规则-->iptables -R INPUT 1 -s 192.168.12.0 -j DROP 取代现行规则，顺序不变(1是位置)

-I 插入规则-->iptables -I INPUT 1 --dport 80 -j ACCEPT 插入一条规则，原本位置上的规则将会往后移动一个顺位，不指定编号，就会在最前加入规则

-L 查看规则-->iptables -L INPUT 列出规则链中的所有规则，**加--line-numbers显示规则编号**

-N 新的规则-->iptables -N allowed 定义新的规则

-n：以数字的方式显示ip，它会将ip直接显示出来，如果不加-n，则会将ip反向解析成主机名。**和-L一起用，n要放前面**

-P :设置默认策略的（设定默认门是关着的还是开着的）

默认策略一般只有两种

iptables -P INPUT (DROP|ACCEPT)

默认是关的/默认是开的

比如：

iptables -P INPUT DROP

通用参数：

-p 协议 例：iptables -A INPUT -p tcp

-s源地址 例：iptables -A INPUT -s 192.168.1.1 ip范围： 192.168.0.0/24

-d目的地址 例：iptables -A INPUT -d 192.168.12.1

--sport源端口 例:iptables -A INPUT -p tcp --sport 22

--dport目的端口 例:iptables -A INPUT -p tcp --dport 22

-i指定入口网卡 例:iptables -A INPUT -i eth0

-o指定出口网卡 例:iptables -A FORWARD -o eth0

-m扩展各种模块

-m multiport：表示启用多端口扩展

-j 指定要进行的处理动作

常用的ACTION：

DROP：丢弃

REJECT：明示拒绝

ACCEPT：接受

SNAT基于原地址的转换

source--指定地址

比如我们现在要将所有192.168.10.0网段的IP在经过的时候全都转换成172.16.100.1这个假设出来的外网地址：

iptables -t nat -A POSTROUTING -s 192.168.10.0/24 -j SNAT --to-source 172.16.100.1(外网有效ip)

实例，配置防火墙规则：

1、开启所有地址对本机的tcp（80、22、10-21）端口的访问权限 httpd，静态的web站点服务

2、开放所有地址对本机ICMP是Internet控制报文协议的数据包的访问权限

3、其他没有被允许的端口禁止访问

```
# netstat -tuln   查端口
# iptables   查iptables版本信息
# iptables -L    查询设置的规则信息  service iptables status
# iptables -F   删除已有的规则信息
# iptables -I INPUT -p tcp --dport 80 -j ACCEPT
# iptables -I INPUT -p tcp --dport 22 -j ACCEPT
# iptables -I INPUT -p tcp --dport 10:21 -j ACCEPT
# iptables -L
# iptables -I INPUT -p icmp -j ACCEPT
# iptables -A INPUT  -j  REJECT   前面先是允许的规则，剩下的全部拒绝
# yum install -y nmap    安装一个端口扫描工具，在192.168.238.4的机器上，上面的命令是在机器192.168.238.6（默认）上
# nmap -sS -p 0-1000 192.168.238.6   在192.168.238.4的机器上，看结果
# iptables -D INPUT 4
# nmap -sS -p 0-1000 192.168.238.6   在192.168.238.4的机器上，看结果
```

4、本机没办法访问本地，需要我们开启回环网址的访问权限。

```
# yum install -y telnet
# telnet 127.0.0.1 22  #访问不了
# iptables -I INPUT -i lo -j ACCEPT  #-i 是允许设备  lo表示回环网卡的设备名称
```

telnet 127.0.0.1 22 在测试本机能访问本机了

5、本机不能访问其它主机，要放相应权限

curl http://www.baidu.com 访问后面的url地址

iptables -I INPUT -m state --state ESTABLISHED,RELATED -j ACCEPT 配置扩展模块，允许本机访问出去的请求，相应回来的数据通过

curl http://www.baidu.com 可以访问了

6、限制某个端口的服务，只能是某个ip地址访问，比如80端口只需192.168.238.4访问

iptables -I INPUT -p tcp -s 192.168.238.0/24 --dport 80 -j ACCEPT 如果是ip范围的话，就设置网段号，192.168.238.0/24

最后提一下：

实质上，上面的设置命令就是将规则写到配置文件：/etc/sysconfig/iptables，所以也可以直接修改这个配置文件来添加或修改规则，但是改完后要再执行命令 /etc/init.d/iptables reload 使变更的规则生效，而命令的方法则不用

查询防火墙状态 : [root@localhost ~]# service iptables status

停止防火墙 : [root@localhost ~]# service iptables stop

启动防火墙 : [root@localhost ~]# service iptables start

重启防火墙 : [root@localhost ~]# service iptables restart

永久关闭防火墙 : [root@localhost ~]# chkconfig iptables off

永久关闭后启用 : [root@localhost ~]# chkconfig iptables on

firewalld防火墙（CentOS7默认）

```
systemctl status firewalld.service    #查看firewalld状态
systemctl stop firewalld       #关闭
systemctl start firewalld       #开启
systemctl disable firewalld     #开机自动关闭
systemctl enable firewalld     #开机自动启动
```

firewalld和iptables的关系：

1、CentOS7里firewalld和iptables是共存的，真正的防火墙是netfilter，firewalld，iptables都是netfilter对用户提供的一个操作接口

2、firewalld底层是调用iptables的，是建立在iptables之上的防火墙

3、firewalld比iptables更人性化，可以理解为firewalld是iptables的升级版本，本质上是同一个东西

firewalld和iptables的区别：

1、iptables 用命令一条一条规则的写进去，如果改配置文件还需要执行命令 /etc/init.d/iptables reload 新规则才生效

2、firewalld新引入了域的概念，不同的域规则不同，同一个域规则一样，用域对规则进行了分组

查看firewalld的域操作firewalld的命令firewall-cmd

firewall-cmd --list-all-zones 查看所有的域以及他们的规则配置，默认有 9个

firewall-cmd --list-all 查看firewalld的默认域以及域中的规则配置

```
[root@CentOS7 zones]# firewall-cmd --list-all
public (active)
  target: default
  icmp-block-inversion: no
  interfaces: ens33
  sources:
  services: ssh dhcpv6-client
  ports:
  protocols:
  masquerade: no
  forward-ports:
  source-ports:
  icmp-blocks:
  rich rules:
```

默认使用的域

源ip地址

允许对外提供的服务

firewall-cmd --set-default-zone=work 切换默认的域

firewall-cmd --get-active-zone 查看激活的域

firewalld的域和网卡接口

1、一个网卡接口只能属于一个zone

2、一个zone可以绑定多个网卡接口

3、只要配置上网卡接口或者配置上源ip地址，zone就变成了活动域

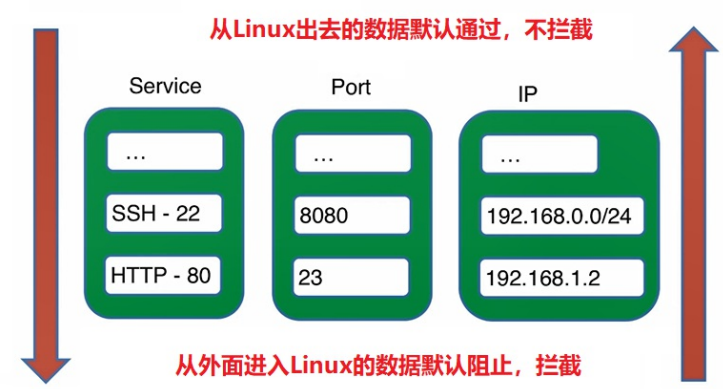
firewall-cmd --zone=home --change-interface=ens33 这样改后，public域还是默认的域，但缺不是激活的域了

firewall-cmd --zone=home --remove-interface=ens33 移除指定的网卡接口

firewall-cmd --zone=home --add-interface=ens33 添加指定的网卡接口

firewalld的规则配置

firewalld的默认规则：



我们要做的工作就是在白名单上加入允许尽量的端口、服务、ip地址

举例1：开启80端口

 # firewall-cmd --zone=public --add-port=80/tcp　这样的配置只是临时生效，防火墙重启后，配置丢失

 # firewall-cmd --zone=public --add-port=80/tcp --permanent　想要永久生效，后面加要给参数--permnent 表示永久

 # firewall-cmd --zone=public --remove-port=80/tcp --permanent

 # firewall-cmd --reload　防火墙热重载，如果方向新规则没生效可以热重载试试

举例2：开启服务，其实服务和端口是一致的，因为没个服务必然绑定对应的端口

/usr/lib/firewalld/services这个目录下有很多的xml配置文件，对应着可以配置的服务，可以看一下

如果我们要加，是可以自己新建xml文件放这里的，这里有xml配置文件的服务就可以用服务的方法放开防火墙的限制

 # firewall-cmd --zone=public --remove-port=80/tcp　先移除刚刚设置的端口

 # firewall-cmd --zone=public --add-service=http 然后设置服务，同样这样是临时的

 # firewall-cmd --zone=public --add-service=http --permanent　永久生效

 # firewall-cmd --reload　重载一下

firewalld中区域默认规则设定

阻塞区域（block）：任何传入的网络数据包都将被阻止。

工作区域（work）：相信网络上的其他计算机，不会损害你的计算机。

家庭区域（home）：相信网络上的其他计算机，不会损害你的计算机。

公共区域（public）：不相信网络上的任何计算机，只有选择接受传入的网络连接。

隔离区域（DMZ）：隔离区域也称为非军事区域，内外网络之间增加的一层网络，起到缓冲作用。对隔离区域，只有选择接受传入的网络连接。

信任区域（trusted）：所有的网络连接都可以接受。

丢弃区域（drop）：任何传入的网络连接都被拒绝。

内部区域（internal）：信任网络上的其他计算机，不会损害你的计算机。只有选择接受传入的网络连接。

外部区域（external）：不相信网络上的其他计算机，不会损害你的计算机。只有选择接受传入的网络连接。

举例3：限制所有的ip地址访问，除开白名单上的ip地址

firewall-cmd --set-default-zone=drop

firewall-cmd --zone=trusted --add-source=192.168.238.0/24　#同样要删除，add 改成 remove

常见的网络攻击

1. DDOS
2. CC
3. SYN

DDOS

一群恶霸试图让对面那家有着竞争关系的商铺无法正常营业，恶霸们扮作普通客户一直拥挤在对手的商铺，赖着不走，真正的购物者却无法进入；恶霸们完成这些坏事有时凭单干难以完成，需要叫上很多人一起，网络安全领域中DoS和DDoS攻击就遵循着这些思路。

DoS（Denial of Service），即拒绝服务攻击，该攻击方式利用目标系统网络服务功能缺陷或者直接消耗其系统资源，使得该目标系统无法提供正常的服务。

DdoS的攻击方式有很多种，最基本的DoS攻击就是利用合理的服务请求来占用过多的服务资源，从而使合法用户无法得到服务的响应。单一的DoS攻击一般是采用一对一方式的，当攻击目标CPU速度低、内存小或者网络带宽小等等各项指标不高的性能，它的效果是明显的。随着计算机与网络技术的发展，计算机的处理能力迅速增长，内存大大增加，同时也出现了千兆级别的网络，这使得DoS攻击的困难程度加大了-目标对恶意攻击包的"消化能力"加强了不少。这时候分布式的拒绝服务攻击手段（DDoS）就应运而生了。DDoS就是利用更多的傀儡机（肉鸡）来发起进攻，以比从前更大的规模来进攻受害者。

CC

DDoS攻击打的是网站的服务器，而CC攻击是针对网站的页面攻击的，用术语来说就是，一个是WEB网络层拒绝服务攻击（DDoS），一个是WEB应用层拒绝服务攻击（CC）

SYN

SYN攻击属于DOS攻击的一种，它利用TCP协议缺陷，通过发送大量的半连接请求，耗费目标的CPU和内存资源。服务器进入SYN_RECV状态后，如果未收到客户端的确认包时，会重发请求包，一直到超时。