

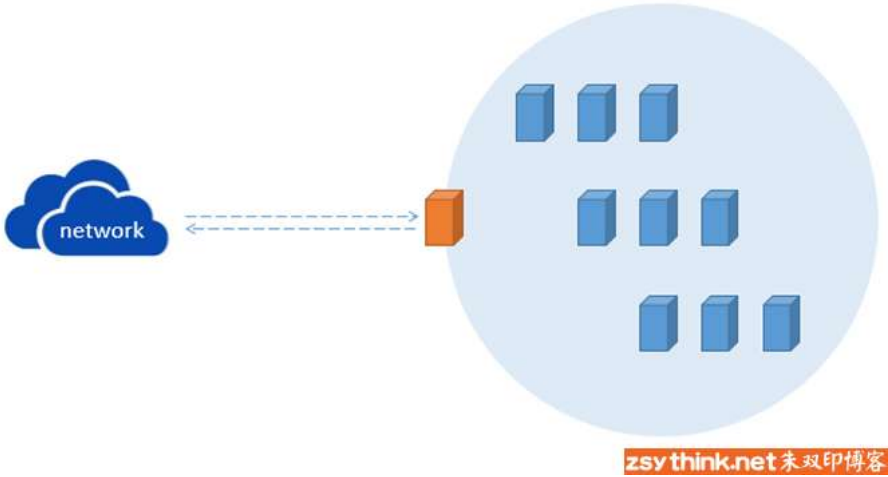
iptables详解（11）：iptables之网络防火墙

在本博客中，从理论到实践，系统的介绍了iptables，如果你想要从头开始了解iptables，可以查看iptables文章列表，直达链接如下
[iptables零基础快速入门系列](#)

阅读这篇文章需要站在前文的基础之上，如果在阅读时遇到障碍，请回顾前文。

我们一起来回顾一下之前的知识，在第一篇介绍iptables的文章中，我们就描述过防火墙的概念，我们说过，防火墙从逻辑上讲，可以分为主机防火墙与网络防火墙
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。

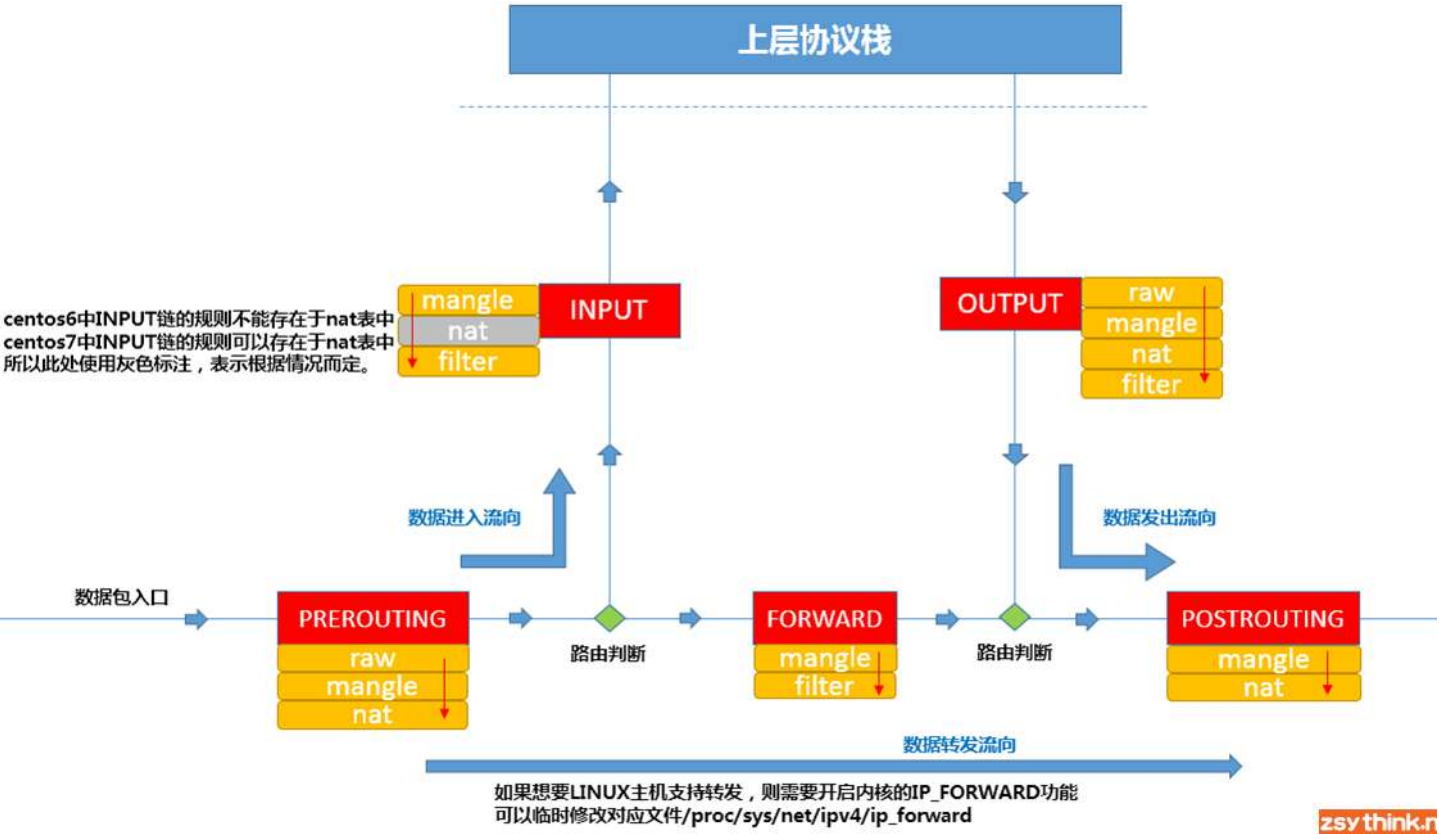
在前文的举例中，iptables都是作为主机防火墙的角色出现的，那么，iptables怎样作为网络防火墙呢？这就是我们今天要聊的话题。
回到刚才的概念，网络防火墙往往处于网络的入口或者边缘，那么，如果想要使用iptables充当网络防火墙，iptables所在的主机则需要处于网络入口处，示意图如



zsy think.net 朱双印博客

上图中，橘黄色主机为iptables所在主机，此时iptables充当的角色即为网络防火墙，上图中的浅蓝色圆形表示网络防火墙所防护的网络区域，圆形内的蓝色矩形表
主机。

当外部网络中的主机与网络内部主机通讯时，不管是由外部主机发往内部主机的报文，还是由内部主机发往外部主机的报文，都需要经过iptables所在的主机，由ip
主机进行"过滤并转发"，所以，防火墙主机的主要工作就是"过滤并转发"，那么，说到这里，我们则不得不再次回顾之前的iptables报文流程图了，如下：

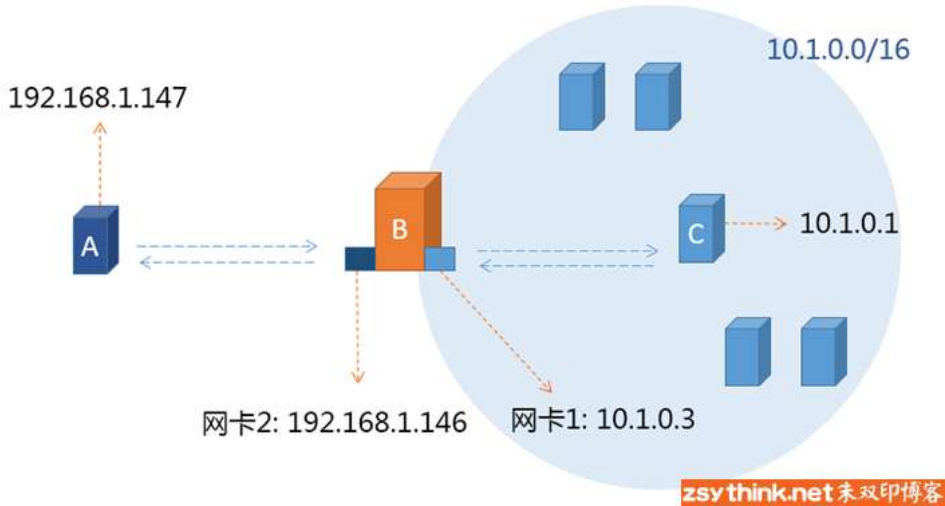


zsy think.net

前文中，iptables都是作为"主机防火墙"的角色出现的，所以我们举例时，只用到了上图中的INPUT链与OUTPUT链，因为拥有"过滤功能"的链只有3条，INPUT、FORWARD，当报文发往本机时，如果想要过滤，只能在INPUT链与OUTPUT链中实现，而此时，iptables的角色发生了转变，我们想要将iptables所在的主机打造成

环境准备

那么为了能够进行实验, 我们来设置一下实验场景, 如下图所示 (后面有对图的解释)



我们假设, 上图中圆形所示的网络为内部网络

注: 此处所描述的内网、外网与我们平常所说的公网、私网不同。

此处描述的内外网你可以理解成两个网段, A网络与B网络, 为了方便描述, 我们把圆形内的主机称为内部主机, 把上图中圆形所表示的网络称为内部网络, 把圆外为外部网络。

假设, 内部网络的网段为10.1.0.0/16, 此内部网络中存在主机C, 主机C的IP地址为10.1.0.1。

上图中的主机B充当了网络防火墙的角色, 主机B也属于内部网络, 同时主机B也能与外部网络进行通讯, 如上图所示, 主机B有两块网卡, 网卡1与网卡2, 网卡1的IP地址为10.1.0.3, 网卡2的IP地址为192.168.1.146, 所以, 防火墙主机在内部网络中的IP地址为10.1.0.3, 防火墙主机与外部网络通讯的IP地址为192.168.1.146。

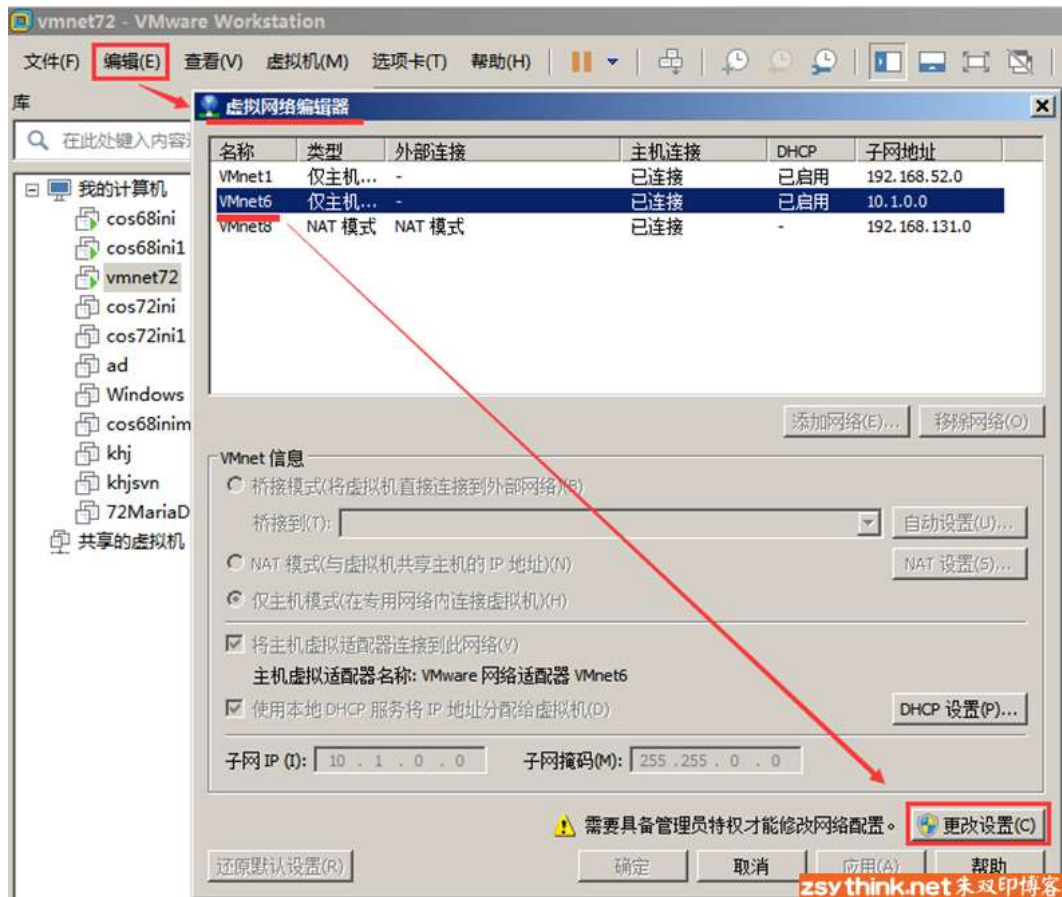
上图中的主机A充当了"外部网络主机"的角色, A主机的IP地址为192.168.1.147, 我们使用主机A访问内部网络中的主机C, 但是需要主机B进行转发, 主机B在转发过程中, 以实现网络防火墙的功能。

我已经准备了3台虚拟机, A、B、C

虚拟机A与虚拟机B的网卡2都使用了桥接模式。

为了能够尽量模拟内部网络的网络入口, 我们将虚拟机B的网卡1与虚拟机C同时放在"仅主机模式"的虚拟网络中, 虚拟机设置如下图所示

点击vmware编辑菜单, 打开虚拟网络编辑器, 点击更改设置按钮, 添加一个仅主机模式的虚拟网络, 下图中的vmnet6为已经添加过的虚拟网络, 此处不再重复添加



由于B主机现在的角色是10.1.0.0中的“网络防火墙”，那么，我们直接将C主机的网关指向B主机的内部网络IP，如下图所示

```
[root@cos72ini ~]# cat /etc/sysconfig/network-scripts/ifcfg-enol6777736
TYPE=Ethernet
#BOOTPROTO=dhcp
IPADDR=10.1.0.1
PREFIX=16
GATEWAY=10.1.0.3
DEFROUTE=yes
PEERDNS=yes
PEERROUTES=yes
NAME=enol6777736
UUID=f57ab594-ca01-4252-b232-134f550f5362
DEVICE=enol6777736
ONBOOT=yes
```

同时，为了尽量简化路由设置，我们直接将A主机访问10.1网络时的网关指向B主机的网卡2上的IP，如下图所示。

注：route命令配置的路由条目在网络重启后将会失效

```
[www.zsythink.net]# ifconfig | awk '/inet addr/ {print $1,$2}'
inet addr:192.168.1.147
inet addr:127.0.0.1
[www.zsythink.net]# route add -net 10.1.0.0/16 gw 192.168.1.146
[www.zsythink.net]# route -n
Kernel IP routing table
Destination      Gateway           Genmask          Flags Metric Ref    Use Iface
192.168.1.0      0.0.0.0          255.255.255.0    U        1      0      0 eth1
10.1.0.0         192.168.1.146    255.255.0.0      UG       0      0      0 eth1
[www.zsythink.net]#
```

现在A主机通往10.1网络的网关已经指向了B主机，那么，现在A主机能够达到10.1.0.0/16网络吗？我们来试试

如下图所示，我们直接在A主机上向C主机发起ping请求，并没有得到任何回应。

```

A x B x C x +
[www.zsythink.net]# ping 10.1.0.1
PING 10.1.0.1 (10.1.0.1) 56(84) bytes of data.

```

那么，我们再来试试B主机上的内部网IP，如下图所示，直接在A主机上向B主机的内部网IP发起ping请求，发现是可以ping通的，这是为什么呢？

```

A x B x C x +
[www.zsythink.net]# ping 10.1.0.3
PING 10.1.0.3 (10.1.0.3) 56(84) bytes of data.
64 bytes from 10.1.0.3: icmp_seq=1 ttl=64 time=1.24 ms
64 bytes from 10.1.0.3: icmp_seq=2 ttl=64 time=0.315 ms
64 bytes from 10.1.0.3: icmp_seq=3

```

按照道理来说，10.1.0.1与10.1.0.3都属于10.1.0.0/16网段，为什么B主机上的IP就能通，C主机上的IP却不通呢？

咱们先来聊聊为什么10.1.0.1没有回应。

A主机通过路由表得知，发往10.1.0.0/16网段的报文的网关为B主机，当报文达到B主机时，B主机发现A的目标为10.1.0.1，而自己的IP是10.1.0.2，这时，B主机则文转发给10.1.0.1（也就是C主机），但是，Linux主机在默认情况下，并不会转发报文，如果想要让Linux主机能够转发报文，需要额外的设置，这就是为什么10.1.0.1没有回应的原因，因为B主机压根就没有将A主机的ping请求转发给C主机，C主机压根就没有收到A的ping请求，所以A自然得不到回应。

现在再来聊聊为什么10.1.0.3会回应。

这是因为10.1.0.3这个IP与192.168.1.146这个IP都属于B主机，当A主机通过路由表将ping报文发送到B主机上时，B主机发现自己既是192.168.1.146又是10.1.0.3就直接回应了A主机，并没有将报文转发给谁，所以A主机得到了10.1.0.3的回应。

我想我应该说明白了，那么，我们应该怎样设置，才能让Linux主机转发报文呢？我们一起来设置一遍就好了。

首先，我们可以查看/proc/sys/net/ipv4/ip_forward文件中的内容，如果文件内容为0，则表示当前主机不支持转发。

```

A x B x C x +
[www.zsythink.net]# ifconfig | grep "<inet>"
inet 10.1.0.3 netmask 255.255.0.0 broadcast 10.1.255.255
inet 192.168.1.146 netmask 255.255.255.0 broadcast 192.168.1.255
inet 127.0.0.1 netmask 255.0.0.0
inet 192.168.122.1 netmask 255.255.255.0 broadcast 192.168.122.255
[www.zsythink.net]#
[www.zsythink.net]# cat /proc/sys/net/ipv4/ip_forward
0
[www.zsythink.net]#

```

如果我们想要让当前主机支持核心转发功能，只需要将此文件中的值设置为1即可，示例如下。

```

[www.zsythink.net]# echo 1 > /proc/sys/net/ipv4/ip_forward
[www.zsythink.net]# cat /proc/sys/net/ipv4/ip_forward
1
[www.zsythink.net]#

```

好了，现在我们就开启了B主机的核心转发功能。

除了上述方法，还能使用sysctl命令去设置是否开启核心转发，示例如下。

```

[www.zsythink.net]# sysctl -w net.ipv4.ip_forward=0
net.ipv4.ip_forward = 0
[www.zsythink.net]# cat /proc/sys/net/ipv4/ip_forward
0
[www.zsythink.net]# sysctl -w net.ipv4.ip_forward=1
net.ipv4.ip_forward = 1
[www.zsythink.net]# cat /proc/sys/net/ipv4/ip_forward
1
[www.zsythink.net]#

```

上述两种方法都能控制是否开启核心转发，但是通过上述两种方法设置后，只能临时生效，当重启网络服务以后，核心转发功能将会失效。

如果想要永久生效，则需要设置/etc/sysctl.conf文件（centos7中配置/usr/lib/sysctl.d/00-system.conf文件），添加（或修改）配置项 net.ipv4.ip_forward = 1。

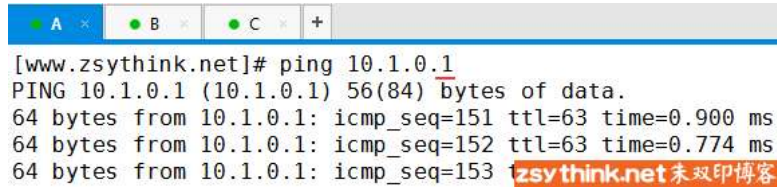
```

A x B x C x +
10
11 net.ipv4.ip_forward = 1

```

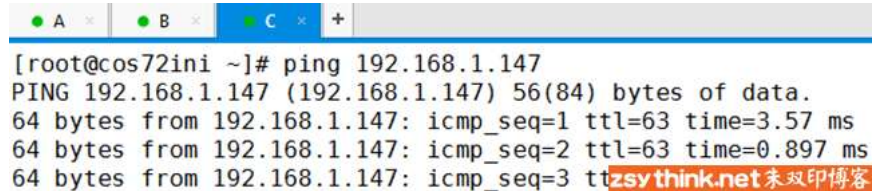

现在，B主机已经具备了核心转发功能，已经可以转发报文了，现在，我们再次回到A主机中，向C主机发起ping请求，如下图所示，已经可以ping通。

注：如果你仍然无法ping通，可能是因为你使用route命令配置了C主机的默认网关，这种情况下，请查看C主机的路由配置是否自动消失了，如果没有对应的路由配置，同时，如果你的主机C如果有多块网卡，可以暂时禁用其他网卡试试



```
[www.zsythink.net]# ping 10.1.0.1
PING 10.1.0.1 (10.1.0.1) 56(84) bytes of data.
64 bytes from 10.1.0.1: icmp_seq=151 ttl=63 time=0.900 ms
64 bytes from 10.1.0.1: icmp_seq=152 ttl=63 time=0.774 ms
64 bytes from 10.1.0.1: icmp_seq=153 ttl=63 time=0.897 ms
```

同时，从主机C向主机A发起ping请求，也可以ping通，如下图所示



```
[root@cos72ini ~]# ping 192.168.1.147
PING 192.168.1.147 (192.168.1.147) 56(84) bytes of data.
64 bytes from 192.168.1.147: icmp_seq=1 ttl=63 time=3.57 ms
64 bytes from 192.168.1.147: icmp_seq=2 ttl=63 time=0.897 ms
64 bytes from 192.168.1.147: icmp_seq=3 ttl=63 time=0.897 ms
```

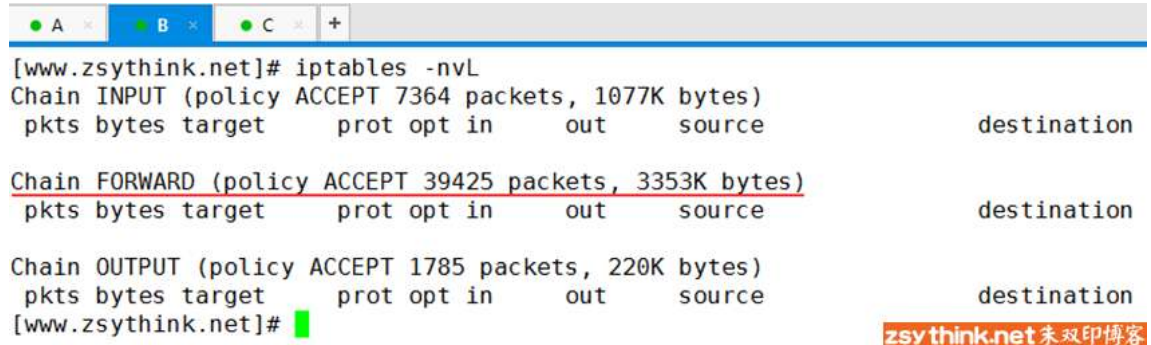
好了，我们的测试环境已经准备完毕，现在可以开始测试了。

但是在开始之前，请确定主机A与主机C上没有对应的iptables规则，因为此处我们主要是用来测试“网络防火墙”的，为了减少主机防火墙带来的影响，我们直接将主机的规则清空。

网络防火墙测试

之前说过，iptables作为网络防火墙时，主要负责“过滤与转发”，既然要过滤，则需配置filter表，既然要转发，则需在FORWARD链中定义规则，所以，我们应该在FORWARD链中配置规则。

那么，我们先来看看主机B上的filter表中是否已经存在规则，如下



```
[www.zsythink.net]# iptables -nvL
Chain INPUT (policy ACCEPT 7364 packets, 1077K bytes)
pkts bytes target      prot opt in      out     source      destination

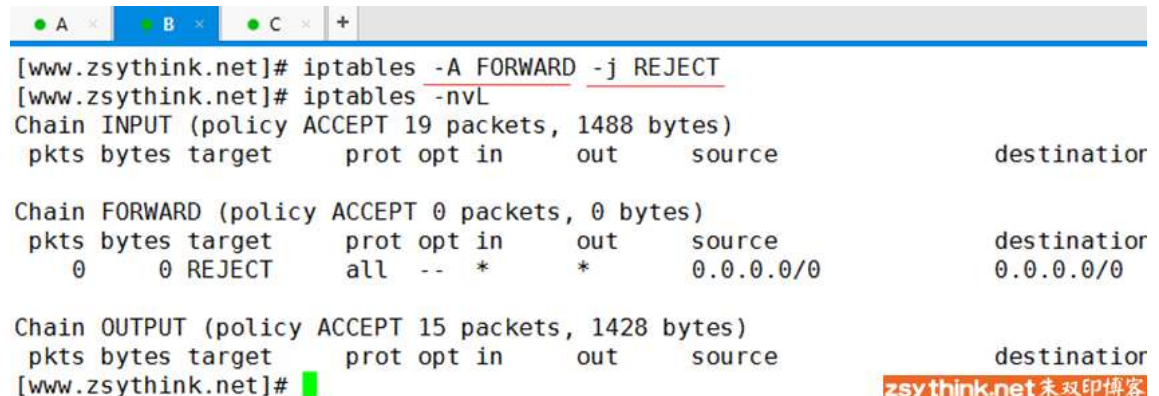
Chain FORWARD (policy ACCEPT 39425 packets, 3353K bytes)
pkts bytes target      prot opt in      out     source      destination

Chain OUTPUT (policy ACCEPT 1785 packets, 220K bytes)
pkts bytes target      prot opt in      out     source      destination
```

从上图可以看出，FORWARD链中没有任何规则，默认策略为ACCEPT，我们可以使用“白名单机制”（如果忘了请回顾前文：黑白名单机制）

在主机B中FORWARD链的末端添加一条默认拒绝的规则，然后将“放行规则”设置在这条“默认拒绝规则”之前即可。

示例如下



```
[www.zsythink.net]# iptables -A FORWARD -j REJECT
[www.zsythink.net]# iptables -nvL
Chain INPUT (policy ACCEPT 19 packets, 1488 bytes)
pkts bytes target      prot opt in      out     source      destination

Chain FORWARD (policy ACCEPT 0 packets, 0 bytes)
pkts bytes target      prot opt in      out     source      destination
 0      0 REJECT      all  --  *      *       0.0.0.0/0   0.0.0.0/0

Chain OUTPUT (policy ACCEPT 15 packets, 1428 bytes)
pkts bytes target      prot opt in      out     source      destination
```

好了，配置完上述规则后，主机A与主机C已经无法通讯了，因为它们之间如果想要通讯，则需要靠主机B进行转发，而上述规则设置完成后，所有报文都无法通过FORWARD链了，所以任何经过转发的报文在经过FORWARD链时都会被拒绝，外部主机的报文无法转发到内部主机中，内部网主机的报文也无法转发到外部主机中，因为主机B拒绝了所有报文。

现在，我们同时将A主机与C主机中的web服务启动，以便进行测试。

首先，我们启动A主机的httpd服务

```

[www.zsythink.net]# cat /var/www/html/index.html
outside web server
[www.zsythink.net]# service httpd start
Starting httpd:
[www.zsythink.net]#

```

[OK]

zsythink.net 宋双印博客

同时，启动C主机的httpd服务

```

[root@cos72ini testdir]# cat /var/www/html/index.html
inside web server
[root@cos72ini testdir]# systemctl start httpd
[root@cos72ini testdir]#

```

zsythink.net 宋双印博客

由于刚才已经在主机B中设置了默认拒绝的规则，所以此刻，A主机无法访问C主机的web服务，C主机同样无法访问A主机的web服务。

那么，如果我们想要使内部的主机能够访问外部主机的web服务，我们应该怎样做呢？没错，我们需要在FORWARD链中放行内部主机对外部主机的web请求，只需求。

```

[www.zsythink.net]# iptables -I FORWARD -s 10.1.0.0/16 -p tcp --dport 80 -j ACCEPT
[www.zsythink.net]# iptables -nvL FORWARD
Chain FORWARD (policy ACCEPT 0 packets, 0 bytes)
  pkts bytes target    prot opt in     out     source            destination
    0    0 ACCEPT    tcp  --  *      *       10.1.0.0/16       0.0.0.0/0         tcp
    0    0 REJECT    all  --  *      *       0.0.0.0/0         0.0.0.0/0         reject
[www.zsythink.net]#

```

zsythink.net

如上图所示，防火墙放行了内部主机的web请求，因为我们将来自内部网络中目标端口为80的报文都放行了，那么此时，我们在C主机上访问A主机的web服务试试

此时，在主机C上访问主机A的web服务，如下

```

[root@cos72ini ~]# curl 192.168.1.147

```

zsythink.net 宋双印博客

可以看到，主机C并无法访问到主机A上的web服务，这是为什么呢？

聪明如你肯定已经想到了，我们只在主机B上放行了内部主机访问80端口的请求，但是并没有放行外部主机的回应报文，虽然内部主机的请求能够通过防火墙主机B响应的报文则无法进入防火墙，所以，我们仍然需要在主机B上进行如下设置。

```

[www.zsythink.net]# iptables -I FORWARD -d 10.1.0.0/16 -p tcp --sport 80 -j ACCEPT
[www.zsythink.net]# iptables -nvL FORWARD
Chain FORWARD (policy ACCEPT 0 packets, 0 bytes)
  pkts bytes target    prot opt in     out     source            destination
    5   555 ACCEPT    tcp  --  *      *       0.0.0.0/0         10.1.0.0/16       tcp
    9   585 ACCEPT    tcp  --  *      *       10.1.0.0/16       0.0.0.0/0         tcp
    4   240 REJECT    all  --  *      *       0.0.0.0/0         0.0.0.0/0         reject

```

zsythink.net

如上图所示，当外部主机中的web服务响应内部主机时，目标地址肯定为内部主机，所以，我们需要放行目标IP属于内部主机网段的报文，源端口为80，因为外部主机的回应报文则无法进入防火墙，所以，我们仍然需要在主机B上进行如下设置。

完成上述配置后，再次回到C主机上，访问A主机的web服务，可以看到，已经能够正常访问了。

```

[root@cos72ini ~]# curl 192.168.1.147
outside web server

```

从上述示例可以看出，当iptables作为“网络防火墙”时，在配置规则时，往往需要考虑“双向性”，也就是说，我们为了达成一个目的，往往需要两条规则才能完成。

那么此时，A主机能够访问C主机中的web服务吗？我想你已经知道答案了，没错，A主机此时无法访问C主机中的web服务，因为B主机中并没有放行相关报文。

结合之前的知识，我们可以将上述规则配置进行优化，比如，不管是由内而外，还是由外而内，只要是“响应报文”，我们统统放行，配置如下

注：如果你没有明白如下配置的含义，请回顾之前的文章


```

[www.zsythink.net]# iptables -nvL FORWARD
Chain FORWARD (policy ACCEPT 0 packets, 0 bytes)
  pkts bytes target    prot opt in     out     source               destination
    15 1665 ACCEPT    tcp  --  *      *       0.0.0.0/0           10.1.0.0/16         tcp spt:80
    21 1395 ACCEPT    tcp  --  *      *       10.1.0.0/16         0.0.0.0/0           tcp dpt:80
     8  480 REJECT    all  --  *      *       0.0.0.0/0           0.0.0.0/0           reject-with icmp
[www.zsythink.net]# iptables -D FORWARD 1
[www.zsythink.net]#
[www.zsythink.net]# iptables -I FORWARD -m state --state ESTABLISHED,RELATED -j ACCEPT
[www.zsythink.net]# iptables -nvL FORWARD
Chain FORWARD (policy ACCEPT 0 packets, 0 bytes)
  pkts bytes target    prot opt in     out     source               destination
     0     0 ACCEPT    all  --  *      *       0.0.0.0/0           0.0.0.0/0           state RELATED,ES
    21 1395 ACCEPT    tcp  --  *      *       10.1.0.0/16         0.0.0.0/0           tcp dpt:80
     8  480 REJECT    all  --  *      *       0.0.0.0/0           0.0.0.0/0           reject-with icmp
[www.zsythink.net]#

```

如上图所示，先将“web响应报文放行规则”删除，同时增加了上图中的规则，只需要在网络防火墙主机的FORWARD链中添加如上一条规则，就可以将绝大多数响应不管是外部响应内部，还是内部响应外部，一条规则就能搞定，当iptables作为网络防火墙时，每次配置规则时都要考虑“双向”的问题，但是配置完上述规则后，我求报文的方向就行了，而回应报文，上述一条规则就能搞定，这样配置，即使以后有更多服务的响应报文需要放行，我们也不用再去针对响应报文设置规则了（具体详细的总结过），应该会让我们省去不少规则吧。

比如，我们除了想要让内部主机能够访问外部的web服务，还想让内部主机能够访问外部的sshd服务，那么，我们则可以进行如下设置。

```

[www.zsythink.net]# iptables -I FORWARD -s 10.1.0.0/16 -p tcp --dport 22 -j ACCEPT
[www.zsythink.net]# iptables -nvL FORWARD
Chain FORWARD (policy ACCEPT 0 packets, 0 bytes)
  pkts bytes target    prot opt in     out     source               destination
     0     0 ACCEPT    tcp  --  *      *       10.1.0.0/16         0.0.0.0/0           tcp dpt:22
    49 7765 ACCEPT    all  --  *      *       0.0.0.0/0           0.0.0.0/0           state RELATED,ES
    22 1455 ACCEPT    tcp  --  *      *       10.1.0.0/16         0.0.0.0/0           tcp dpt:80
     9  540 REJECT    all  --  *      *       0.0.0.0/0           0.0.0.0/0           reject

```

如上图所示，我们只要考虑内部主机的请求方向的报文规则即可，因为响应报文的规则已经被之前配置的规则“承包了”。

此刻，使用C主机即可访问A主机的22端口。

```

[root@cos72ini ~]# ssh 192.168.1.147
root@192.168.1.147's password:
Last login: Sun May  7 10:57:29 2017 from 192.168.1.60
[www.zsythink.net]#

```

目前，我们只允许内部主机访问外部主机的web服务与sshd服务，但是外部主机还无法访问内部主机的服务，那么具体怎么配置我们就不赘述了，就由客官你去负责

备注：在之前的一次实验中，使用centos6.8作为网络防火墙，出现了即使开启核心转发，也无法转发报文的情况，具体原因仍未查明，遇到过类似场景的朋友如果欢迎赐教。

小结

为了方便以后回顾，我们将上述过程提炼总结一下。

- 1 #如果想要iptables作为网络防火墙，iptables所在主机开启核心转发功能，以便能够转发报文。
- 2 #使用如下命令查看当前主机是否已经开启了核心转发，0表示为开启，1表示已开启
- 3 `cat /proc/sys/net/ipv4/ip_forward`
- 4 #使用如下两种方法均可临时开启核心转发，立即生效，但是重启网络配置后会失效。
- 5 方法一：`echo 1 > /proc/sys/net/ipv4/ip_forward`
- 6 方法二：`sysctl -w net.ipv4.ip_forward=1`
- 7 #使用如下方法开启核心转发功能，重启网络服务后永久生效。
- 8 配置/etc/sysctl.conf文件（centos7中配置/usr/lib/sysctl.d/00-system.conf文件），在配置文件中将 net.ipv4.ip_forward 设置为1
- 9
- 10 #由于iptables此时的角色为“网络防火墙”，所以需要在filter表中的FORWARD链中设置规则。

```
11 #可以使用"白名单机制", 先添加一条默认拒绝的规则, 然后再为需要放行的报文设置规则。
12 #配置规则时需要考虑"方向问题", 针对请求报文与回应报文, 考虑报文的源地址与目标地址, 源端口与目标端口等。
13 #示例为允许网络内主机访问网络外主机的web服务与sshd服务。
14 iptables -A FORWARD -j REJECT
15 iptables -I FORWARD -s 10.1.0.0/16 -p tcp --dport 80 -j ACCEPT
16 iptables -I FORWARD -d 10.1.0.0/16 -p tcp --sport 80 -j ACCEPT
17 iptables -I FORWARD -s 10.1.0.0/16 -p tcp --dport 22 -j ACCEPT
18 iptables -I FORWARD -d 10.1.0.0/16 -p tcp --sport 22 -j ACCEPT
19
20 #可以使用state扩展模块, 对上述规则进行优化, 使用如下配置可以省略许多"回应报文放行规则"。
21 iptables -A FORWARD -j REJECT
22 iptables -I FORWARD -s 10.1.0.0/16 -p tcp --dport 80 -j ACCEPT
23 iptables -I FORWARD -s 10.1.0.0/16 -p tcp --dport 22 -j ACCEPT
24 iptables -I FORWARD -m state --state ESTABLISHED,RELATED -j ACCEPT
```

一些注意点：

- 1、当测试网络防火墙时，默认前提为网络已经正确配置。
- 2、当测试网络防火墙时，如果出现问题，请先确定主机防火墙规则的配置没有问题。



我的微信公众号

关注"实用运维笔记"微信公众号, 当博客中有新文章时, 可第一时间得知哦~