

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](#)

概述

阅读这篇文章需要站在前文的基础上，如果你在阅读时遇到障碍，请参考之前的文章。

前文中，我们已经了解了如下动作

ACCEPT、DROP、REJECT、LOG

今天，我们来认识几个新动作，它们是：

SNAT、DNAT、MASQUERADE、REDIRECT

在认识它们之前，我们先来聊聊NAT，如果你对NAT的相关概念已经滚瓜烂熟，可以跳过如下场景描述。

NAT是Network Address Translation的缩写，译为"网络地址转换"，NAT说白了就是修改报文的IP地址，NAT功能通常会被集成到路由器、防火墙、或独立的NA为什么要修改报文的IP地址呢？我们来描述一些场景，即可知道为什么有这方面的需求了。

场景1：

假设，网络内部有10台主机，它们有各自的IP地址，当网络内部的主机与其他网络中的主机通讯时，则会暴露自己的IP地址，如果我们想要隐藏这些主机的IP地址，可以这样办，如下。

当网络内部的主机向网络外部主机发送报文时，报文会经过防火墙或路由器，当报文经过防火墙或路由器时，将报文的源IP修改为防火墙或者路由器的IP地址，当其机收到这些报文时，显示的源IP地址则是路由器或者防火墙的，而不是那10台主机的IP地址，这样，就起到隐藏网络内部主机IP的作用，当网络内部主机的报文经过由器会维护一张NAT表，表中记录了报文来自于哪个内部主机的哪个进程（内部主机IP+端口），当报文经过路由器时，路由器会将报文的内部主机源IP替换为路由把源端口也映射为某个端口，NAT表会把这种对应关系记录下来。

示意图如下：



于是，外部主机收到报文时，源IP与源端口显示的都是路由的IP与端口，当外部网络中的主机进行回应时，外部主机将响应报文发送给路由器，路由器根据刚才NAT录，将响应报文中的目标IP与目标端口再改为内部主机的IP与端口号，然后再将响应报文发送给内部网络中的主机。整个过程中，外部主机都不知道内部主机的IP地还能与外部主机通讯，于是起到了隐藏网络内主机IP的作用。

上述整个过程中，就用到了NAT功能，准确的说是用到了NAPT功能，NAPT是NAT的一种，全称为Network Address Port Translation，说白了就是映射报文IP地址映射其端口号，就像刚才描述的过程一样。

刚才描述的过程中，"IP地址的转换"一共发生了两次。

内部网络的报文发送出去时，报文的源IP会被修改，也就是源地址转换：Source Network Address Translation，缩写为SNAT。

外部网络的报文响应时，响应报文的目标IP会再次被修改，也就是目标地址转换：Destinationnetwork address translation，缩写为DNAT。

但是，上述"整个过程"被称为SNAT，因为"整个过程"的前半段使用了SNAT，如果上述"整个过程"的前半段使用了DNAT，则整个过程被称为DNAT，也就是说，整个SNAT还是DNAT，取决于整个过程的前半段使用了SNAT还是DNAT。

其实刚才描述的场景不仅仅能够隐藏网络内部主机的IP地址，还能够让局域网内的主机共享公网IP，让使用私网IP的主机能够访问互联网。

比如，整个公司只有一个公网IP，但是整个公司有10台电脑，我们怎样能让这10台电脑都访问互联网呢？我们可以为这10台电脑都配置上各自的私网IP，比如"192 P，但是互联网是不会路由私网IP的，如果想要访问互联网，则必须使用公网IP，那么，我们就需要想办法，能让这10台主机共享公司仅有的一个公网IP，没错，这场景其实完全一致，我们只要在路由器上配置公网IP，在私网主机访问公网服务时，报文经过路由器，路由器将报文中的私网IP与端口号进行修改和映射，将其映射为号，这时，内网主机即可共享公网IP访问互联网上的服务了，NAT表示意图如下

Work Station	Internal Four-Tuple				External Four-Tuple				Protocol Used
	Source IP Address	Source Port	Destination IP Address	Destination Port	Source IP Address	Source Port	Destination IP Address	Destination Port	
#1	192.168.2.1	12000	a.b.c.d	20	64.33.104.180	14000	a.b.c.d	20	TCP
#1	192.168.2.1	12001	a.b.c.d	21	64.33.104.180	14001	a.b.c.d	21	TCP
#2	192.168.2.2	12000	a.b.c.d	20	64.33.104.180	14002	a.b.c.d	20	TCP
#2	192.168.2.2	12001	a.b.c.d	21	64.33.104.180	14003	a.b.c.d	21	TCP

综上所述，SNAT不仅能够隐藏网内的主机IP，还能够共享公网IP，这在IPV4地址较为紧张的今天，是非常有用的。

场景2：

场景1中，我们描述的过程为SNAT的过程，虽然其过程中也牵扯到DNAT，但是由于整个过程的前半段使用了SNAT，所以整个过程称之为SNAT，那么在什么情况下能称之为DNAT呢？

没错，当整个过程的前半段使用了DNAT时，整个过程被称为DNAT，具体场景如下。

公司有自己的局域网，网络中有两台主机作为服务器，主机1提供web服务，主机2提供数据库服务，但是这两台服务器在局域网中使用私有IP地址，只能被局域网内互联网无法访问到这两台服务器，整个公司只有一个可用的公网IP，怎样通过这个公网IP访问到内网中的这些服务呢？我们可以将这个公网IP配置到公司的某台主机，然后对外宣称，这个IP地址对外提供web服务与数据库服务，于是互联网主机将请求报文发送给这公网IP地址，也就是说，此时报文中的目标IP为公网IP，当路由器将报文的目标地址改为对应的私网地址，比如，如果报文的目标IP与端口号为：公网IP+3306，我们就将报文的目标地址与端口改为：主机2的私网IP+3306，同理端口映射为主机1的私网IP+80端口，当私网中的主机回应对应请求报文时，再将回应报文的源地址从私网IP+端口号映射为公网IP+端口号，再由路由器或公网主机发出的主机。

上述过程也牵扯到DNAT与SNAT，但是由于整个过程的前半段使用了DNAT，所以上述过程被称为DNAT

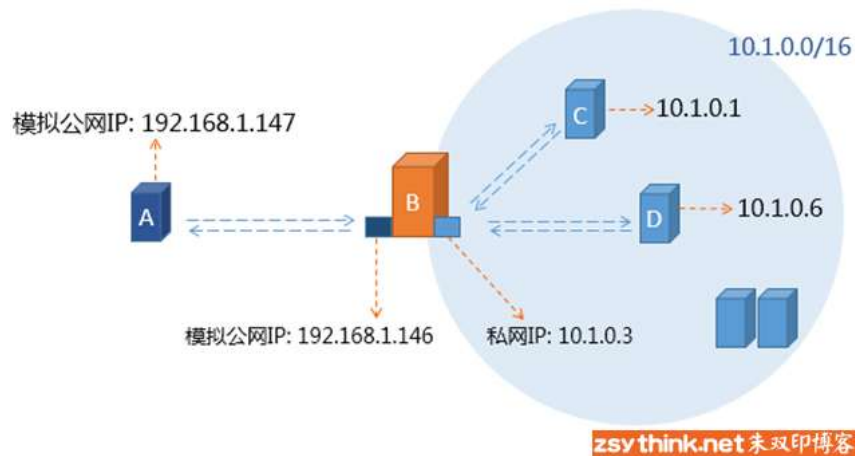
其实，不管是SNAT还是DNAT，都起到了隐藏内部主机IP的作用。

实验环境准备

好了，我们已经了解了SNAT与DNAT的相关概念，那么现在，我们可以动动手了，首先，准备一下实验环境

大致的实验环境是这样的，公司局域网使用的网段为10.1.0.0/16，目前公司只有一个公网IP，局域网内的主机需要共享这个IP与互联网上的主机进行通讯。

由于我们没有真正的公网IP，所以，我们使用私网IP：192.168.1.146模拟所谓的公网IP，示意图如下



如上述示意图所示，实验使用4台虚拟机，A、B、C、D

主机A：扮演公网主机，尝试访问公司提供的服务，IP地址为192.168.1.147

主机B：扮演了拥有NAT功能的防火墙或路由器，充当网关，并且负责NAT，公网、私网通讯的报文通过B主机时，报文会被NAT

主机C：扮演内网web服务器

主机D：扮演内网windows主机

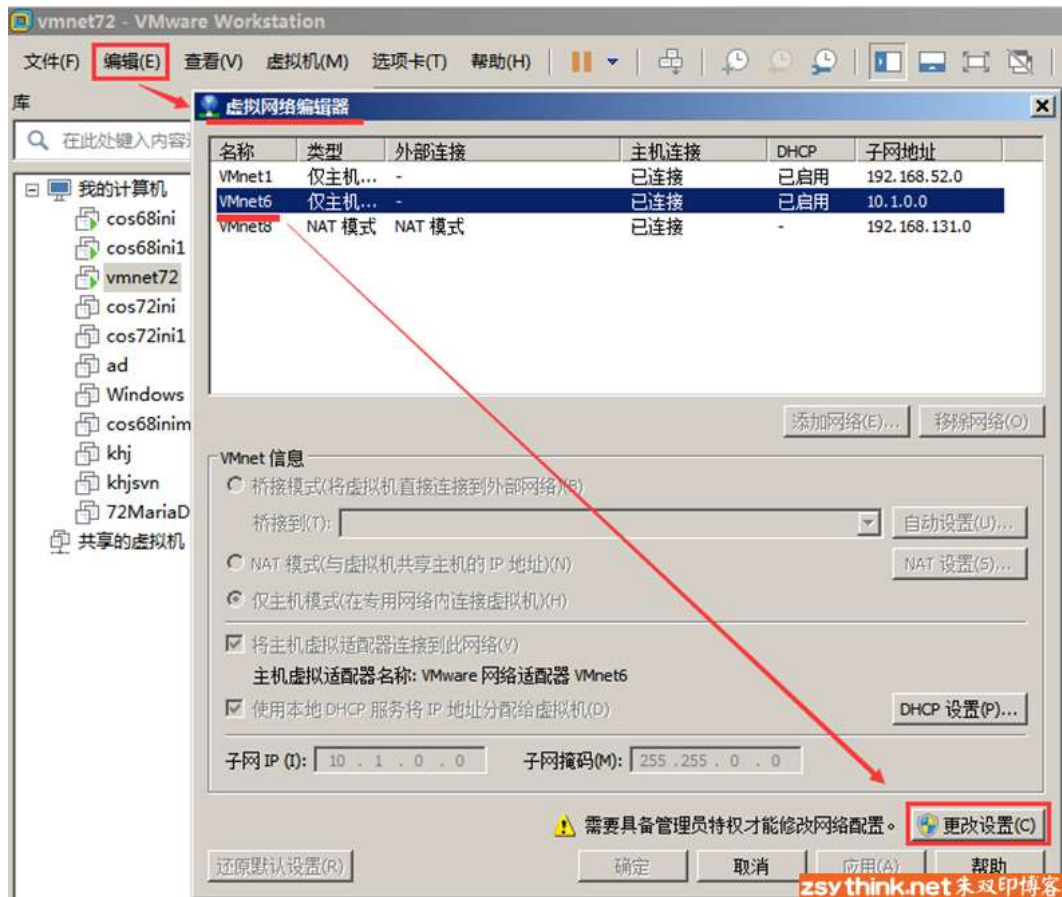
上图圆形所示的逻辑区域表示公司内网，网段为10.1.0.0/16，主机B、C、D都属于内网主机，主机B比较特殊，同时扮演了网关与防火墙，主机B持有公司唯一的（用了一个假的公网IP），局域网内主机如果想与公网主机通讯，需要共享此公网IP，由B主机进行NAT，所以，我们为主机B准备了两块网卡，公网IP与私网IP分别配卡中，同时，在虚拟机中设置了一个“仅主机模式”的虚拟网络，以模拟公司局域网。

聪明如你，应该已经发现了，上述实验环境与之前描述的“网络防火墙”的实验环境相差无几，只不过之前的环境并没有公网，私网的概念，而此刻，圆形逻辑区域之形逻辑区域之外为公网。

环境具体准备过程如下

首先，创建一个虚拟网络，模拟公司内网。

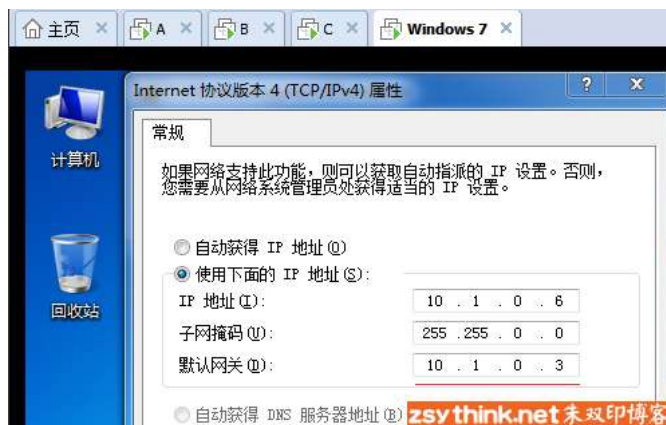
点击vmware虚拟机的编辑菜单，打开“虚拟网络编辑器”，点击更改设置，添加“仅主机模式”的虚拟网络，下图中的VMnet6为已经添加过的虚拟网络，此处不再重



主机C与主机D的网关都指向主机B的私网IP，如下图所示

```
[root@cos72ini ~]# cat /etc/sysconfig/network-scripts/ifcfg-eno16777736
TYPE=Ethernet
#BOOTPROTO=dhcp
IPADDR=10.1.0.1
PREFIX=16
GATEWAY=10.1.0.3
```

zsythink.net 未双印博客



主机B有两块网卡，分别配置了私网IP与公网IP，私网IP为10.1.0.3，私网IP所在的网卡也存在与vmnet6中，模拟公网的IP为192.168.1.146，B主机的公网IP所在的使用桥接模式的虚拟网络，所以，B主机既能与私网主机通讯，也能与公网主机通讯。

```
[www.zsythink.net]# ifconfig | grep "<inet>"
inet 10.1.0.3 netmask 255.255.255.0 broadcast 10.1.0.255
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
```

zsythink.net 未双印博客

由于B主机此时需要负责对报文的修改与转发，所以，需要开启B主机中的核心转发功能，Linux主机默认不会开启核心转发，这在前文中已经详细的描述过，此处你不明白为什么，请回顾前文，使用临时生效的方法开启B主机的核心转发功能，如下图所示。


```
[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]#
```

A主机的IP地址如下，可以与B主机进行通讯，但是不能与C、D进行通讯，因为此刻，A是公网主机，B既是公网主机又是私网主机，C、D是私网的主机，A是不可能的。

```

A x B x C x +
[www.zsythink.net]# ifconfig | grep "inet addr"
    inet addr:192.168.1.147 Bcast:192.168.1.255 Mask:
    inet addr:127.0.0.1 Mask:255.0.0.0
[www.zsythink.net]# ping 192.168.1.146
PING 192.168.1.146 (192.168.1.146) 56(84) bytes of data.
 64 bytes from 192.168.1.146: icmp_seq=1 ttl=64 time=0.301 ms
 64 bytes from 192.168.1.146: icmp_seq=2 ttl=64 time=0.458 ms
 64 bytes from 192.168.1.146: icmp_seq=3
zsythink.net 未双印博客

```

为了能够更好的区分公网服务与私网服务，我们分别在主机A与主机C上启动httpd服务，如下图所示。

```

A x +
[www.zsythink.net]# cat /var/www/html/index.html
outside web server
192.168.1.147
[www.zsythink.net]# service httpd start
Starting httpd:
[www.zsythink.net]#
[ OK ]

C x +
[root@cos72ini ~]# cat /var/www/html/index.html
inside web server
10.1.0.1
[root@cos72ini ~]# systemctl start httpd
zsythink.net 未双印博客

```

好了，实验环境准备完毕，我们来一起动手，实际操作一下。

动作：SNAT

在文章开头的场景中，我们已经描述过，网络内部的主机可以借助SNAT隐藏自己的IP地址，同时还能够共享合法的公网IP，让局域网内的多台主机共享公网IP访问。而此时的主机B就扮演了拥有NAT功能的设备，我们使用iptables的SNAT动作达到刚才所说的目的。

连接到B主机，添加如下规则。

```

A x B x C x +
[www.zsythink.net]# iptables -t nat -A POSTROUTING -s 10.1.0.0/16 -j SNAT --to-source 192.168.1.146
[www.zsythink.net]#
[www.zsythink.net]# iptables -t nat -nvL
Chain PREROUTING (policy ACCEPT 2 packets, 406 bytes)
  pkts bytes target    prot opt in     out     source    destination
Chain INPUT (policy ACCEPT 2 packets, 406 bytes)
  pkts bytes target    prot opt in     out     source    destination
Chain OUTPUT (policy ACCEPT 0 packets, 0 bytes)
  pkts bytes target    prot opt in     out     source    destination
Chain POSTROUTING (policy ACCEPT 0 packets, 0 bytes)
  pkts bytes target    prot opt in     out     source    destination
    0      0 SNAT     all  --  *      *       10.1.0.0/16  0.0.0.0/0
[www.zsythink.net]#
zsythink.net 未双印博客

```

如上图所示，上图中的规则表示将来自于10.1.0.0/16网段的报文的源地址改为公司的公网IP地址。

"-t nat"表示操作nat表，我们之前一直在灌输一个概念，就是不同的表有不同的功能，filter表的功能是过滤，nat表的功能就是地址转换，所以我们需要在nat表中；"-A POSTROUTING"表示将SNAT规则添加到POSTROUTING链的末尾，在centos7中，SNAT规则只能存在于POSTROUTING链与INPUT链中，在centos6中，SNAT规则只能存在于POSTROUTING链中。

你可能会问，为什么SNAT规则必须定义在POSTROUTING链中，我们可以这样认为，POSTROUTING链是iptables中报文发出的最后一个"关卡"，我们应该在报文发出前，修改报文的源地址，否则就再也没有机会修改报文的源地址了，在centos7中，SNAT规则也可以定义在INPUT链中，我们可以这样理解，发往本机的报文经过INPUT链后就到达了本机，如果再不修改报文的源地址，就没有机会修改了。

"-s 10.1.0.0/16"表示报文来自于10.1.0.0/16网段，前文中一直在使用这个匹配条件，我想此处应该不用赘述了。

"-j SNAT"表示使用SNAT动作，对匹配到的报文进行处理，对匹配到的报文进行源地址转换。

"--to-source 192.168.1.146"表示将匹配到的报文的源IP修改为192.168.1.146，前文中，我们已经总结过，某些动作会有自己的选项，"--to-source"就是SNAT选项，用于指定SNAT需要将报文的源IP修改为哪个IP地址。

好了，只要站在前文的基础上，理解上述语句应该是分分钟的事情，聪明如你应该已经学会了，那么我们来测试一下。

目前来说，我们只配置了一条SNAT规则，并没有设置任何DNAT，现在，我们从内网主机上ping外网主机，看看能不能ping通，登录内网主机C，在C主机上向A发送请求(假外网IP)，示例如下

```
[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=0.878 ms
64 bytes from 192.168.1.147: icmp_seq=2 ttl=63 time=0.811 ms
64 bytes from 192.168.1.147: icmp_seq=3 ttl=63 time=0.681 ms
64 bytes from 192.168.1.147: icmp_seq=4
```

如上图所示，"内网主机"已经可以依靠SNAT访问"互联网"了。

为了更加清晰的理解整个SNAT过程，在C主机上抓包看看，查看一下请求报文与响应报文的IP地址，如下，在C主机上同时打开两个命令窗口，一个命令窗口中向A请求，另一个窗口中，使用tcpdump命令对指定的网卡进行抓包，抓取icmp协议的包。

```
[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=0.819 ms
64 bytes from 192.168.1.147: icmp_seq=2 ttl=63 time=1.47 ms
64 bytes from 192.168.1.147: icmp_seq=3 ttl=63 time=1.25 ms
64 bytes from 192.168.1.147: icmp_seq=4

[root@cos72ini ~]# tcpdump -i eno16777736 -nn icmp
tcpdump: verbose output suppressed, use -v or -vv for full protocol details
listening on eno16777736, link-type EN10MB (Ethernet), capture size 65535 bytes
17:07:43.879347 IP 10.1.0.1 > 192.168.1.147: ICMP echo request, id=1, seq=1
17:07:43.880137 IP 192.168.1.147 > 10.1.0.1: ICMP echo reply, id=1, seq=1
17:07:44.880332 IP 10.1.0.1 > 192.168.1.147: ICMP echo request, id=2, seq=2
17:07:44.881453 IP 192.168.1.147 > 10.1.0.1: ICMP echo reply, id=2, seq=2
17:07:45.882161 IP 10.1.0.1 > 192.168.1.147: ICMP echo request, id=3, seq=3
17:07:45.883141 IP 192.168.1.147 > 10.1.0.1: ICMP echo reply, id=3, seq=3
17:07:46.884194 IP 10.1.0.1 > 192.168.1.147: ICMP echo request, id=4, seq=4
```

从上图可以看到，10.1.0.1发出ping包，192.168.1.147进行回应，正是A主机的IP地址（用于模拟公网IP的IP地址）

看来，只是用于配置SNAT的话，我们并不用 手动的进行DNAT设置，iptables会自动维护NAT表，并将响应报文的的目标地址转换回来。

那么，我们去A主机上再次重复一遍刚才的操作，在A主机上抓包看看，如下图所示，C主机上继续向A主机的公网IP发送ping请求，在主机A的网卡上抓包看看。

```
[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=0.994 ms
64 bytes from 192.168.1.147: icmp_seq=2 ttl=63 time=0.834 ms
64 bytes from 192.168.1.147: icmp_seq=3

[www.zsythink.net]# tcpdump -i eth1 -nn icmp
tcpdump: verbose output suppressed, use -v or -vv for full protocol details
listening on eth1, link-type EN10MB (Ethernet), capture size 65535 bytes
17:14:04.129660 IP 192.168.1.146 > 192.168.1.147: ICMP echo request, id=1, seq=1
17:14:04.129709 IP 192.168.1.147 > 192.168.1.146: ICMP echo reply, id=1, seq=1
17:14:05.130962 IP 192.168.1.146 > 192.168.1.147: ICMP echo request, id=2, seq=2
17:14:05.130986 IP 192.168.1.147 > 192.168.1.146: ICMP echo reply, id=2, seq=2
17:14:06.131862 IP 192.168.1.146 > 192.168.1.147: ICMP echo request, id=3, seq=3
```

从上图可以看出，C主机向A主机发起ping请求时得到了回应，但是在A主机上，并不知道是C主机发来的ping请求，A主机以为是B主机发来的ping请求，从抓包的主机以为B主机通过公网IP：192.168.1.146向自己发起了ping请求，而A主机也将响应报文回应给了B主机，所以，整个过程，A主机都不知道C主机的存在，都以为自己发送请求，即使不是在公网私网的场景中，我们也能够使用这种方法，隐藏网络内的主机，只不过此处，我们所描述的环境就是私网主机共享公网IP访问互联网到，私网中的主机已经共享了192.168.1.146这个"伪公网IP"，那么真的共享了吗？我们使用内网主机D试试，主机D是一台windows虚拟机，我们使用它向主机A发看看能不能ping通。如下

```
C:\Windows\system32\cmd.exe
C:\Users\zz\Desktop>ping 192.168.1.147

正在 Ping 192.168.1.147 具有 32 字节的数据:
来自 192.168.1.147 的回复: 字节=32 时间<1ms TTL=63
来自 192.168.1.147 的回复: 字节=32 时间<1ms TTL=63
来自 192.168.1.147 的回复: 字节=32 时间<1ms TTL=63
来自 192.168.1.147 的回复: 字节=32 时间<1ms TTL=63

192.168.1.147 的 Ping 统计信息:
    数据包: 已发送 = 4, 已接收 = 4, 丢失 = 0 (0% 丢失),
    往返行程的估计时间(以毫秒为单位):
        最短 = 0ms, 最长 = 0ms, 平均 = 0ms
```

windows主机也ping通了外网主机，在A主机上抓包，看到的仍然是B主机的IP地址。


```

[www.zsythink.net]# tcpdump -i eth1 -nn icmp
tcpdump: verbose output suppressed, use -v or -vv for full protocol decode
listening on eth1, link-type EN10MB (Ethernet), capture size 65535 bytes
17:23:05.167966 IP 192.168.1.146 > 192.168.1.147: ICMP echo request, id 1, s
17:23:05.167988 IP 192.168.1.147 > 192.168.1.146: ICMP echo reply, id 1, s
17:23:06.174332 IP 192.168.1.146 > 192.168.1.147: ICMP echo request, id 1, s
17:23:06.174352 IP 192.168.1.147 > 192.168.1.146: ICMP echo reply, id 1, s

```

那么，C主机与D主机能够访问外网服务吗？我们来看看。

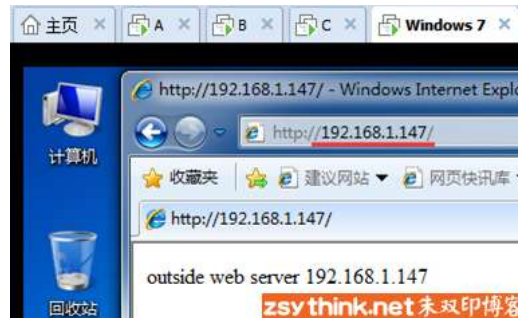
在C主机上访问A主机的web服务，如下图所示，访问正常。

```

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

```

同理，在windows主机中访问A主机的web服务，如下图所示，访问正常。



好了，源地址转换，已经完成了，我们只依靠了一条iptables规则，就能够是内网主机能够共享公网IP访问互联网了。

动作DNAT

公司只有一个公网IP，但是公司的内网中却有很多服务器提供各种服务，我们想要通过公网访问这些服务，该怎么办呢？

没错，使用DNAT即可，我们对外宣称，公司的公网IP上既提供了web服务，也提供了windows远程桌面，不管是访问web服务还是远程桌面，只要访问这个公网IP，利用DNAT，将公网客户端发送过来的报文的目标地址与端口号做了映射，将访问web服务的报文转发到了内网中的C主机中，将访问远程桌面的报文转发到了内网中的D主机中，好了，理论说完了，来动手实践一下。

如下配置由 [运维工程师 王圣杰] 提供，我们一起来讨论一下。

如果我们想要实现刚才描述的场景，则需要在B主机中进行如下配置。

```

[www.zsythink.net]# iptables -t nat -F
[www.zsythink.net]# iptables -t nat -I PREROUTING -d 192.168.1.146 -p tcp --dport 3389 -j DNAT --to-destination 10.1.0.6:3389
[www.zsythink.net]#

```

如上图所示，我们先将nat表中的规则清空了，从头来过，清空nat表规则后，定义了一条DNAT规则。

"-t nat -I PREROUTING"表示在nat表中的PREROUTING链中配置DNAT规则，DNAT规则只配置在PREROUTING链与OUTPUT链中，为什么DNAT规则只能存在与PREROUTING链中呢？我们知道，PREROUTING链处于路由层面之前，如果我们不在PREROUTING链中修改目标地址，当报文到达路由层面时，路由就会根据目标地址发送报文，所以，应该在报文到达路由层面之前就修改报文的目标地址，所以，我们应该在PREROUTING链或OUTPUT链中定义DNAT规则。

"-d 192.168.1.146 -p tcp --dport 3389"表示报文的目标地址为公司的公网IP地址，目标端口为tcp的3389号端口，而我们知道，windows远程桌面使用的默认端口是3389，当外部主机访问公司公网IP的3389号端口时，报文则符合匹配条件。

"-j DNAT --to-destination 10.1.0.6:3389"表示将符合条件的报文进行DNAT，也就是目标地址转换，将符合条件的报文的目标地址与目标端口修改为10.1.0.6:3389，"--to-destination"就是动作DNAT的常用选项。

那么综上所述，上图中定义的规则的含义为，当外网主机访问公司公网IP的3389时，其报文的目标地址与端口将会被映射到10.1.0.6:3389上。

好了，DNAT规则定义完了，现在能够直接使用外网主机访问私网中的服务了吗？

理论上只要完成上述DNAT配置规则即可，但是在测试时，只配置DNAT规则后，并不能正常访问，经过测试发现，将相应的SNAT规则同时配置后，即可正常访问，又配置了SNAT

示例如下。

```
[www.zsythink.net]# iptables -t nat -A POSTROUTING -s 10.1.0.0/16 -i SNAT --to-source 192.168.1.146
[www.zsythink.net]# iptables -t nat -nvL
Chain PREROUTING (policy ACCEPT 0 packets, 0 bytes)
  pkts bytes target     prot opt in     out     source            destination
    0      0 DNAT      tcp  --  *      *      0.0.0.0/0         192.168.1.146      tcp dpt:3389 to:10.1.0.6:3389
Chain INPUT (policy ACCEPT 0 packets, 0 bytes)
  pkts bytes target     prot opt in     out     source            destination
Chain OUTPUT (policy ACCEPT 0 packets, 0 bytes)
  pkts bytes target     prot opt in     out     source            destination
Chain POSTROUTING (policy ACCEPT 0 packets, 0 bytes)
  pkts bytes target     prot opt in     out     source            destination
    0      0 SNAT      all  --  *      *      10.1.0.0/16       0.0.0.0/0          to:192.168.1.146
[www.zsythink.net]#
```

注：理论上只配置DNAT规则即可，但是如果在测试时无法正常DNAT，可以尝试配置对应的SNAT，此处按照配置SNAT的流程进行。没错，与刚才定义SNAT时使用的规则完全一样。

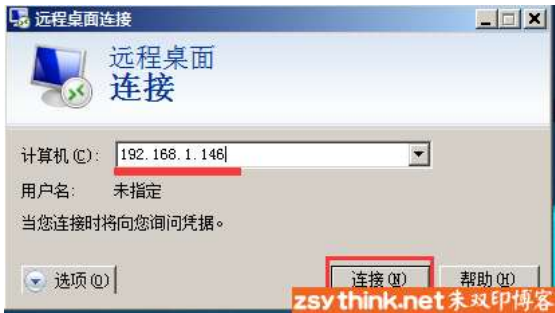
好了，完成上述配置后，我们则可以通过B主机的公网IP，连接D主机（windows主机）的远程桌面了，示例如下。

找到公网中的一台windows主机，打开远程程序



输入公司的公网IP，点击连接按钮

注意：没有指定端口的情况下，默认使用3389端口进行连接，同时，为了确保能够连接到windows虚拟主机，请将windows虚拟主机设置为允许远程连接。



输入远程连接用户的密码以后，即可连接到windows主机



连接以后，远程连接程序显示我们连接到了公司的公网IP，但是当我们查看IP地址时，发现被远程机器的IP地址其实是公司私网中的D主机的IP地址。上图证明，我们已经成功的通过公网IP访问到了内网中的服务。

同理，使用类似的方法，我们也能够在外网中访问到C主机提供的web服务。

示例如下。

```
[www.zsythink.net]# iptables -t nat -I PREROUTING -d 192.168.1.146 -p tcp --dport 801 -j DNAT --to-destination 10.1.0.1:80
[www.zsythink.net]# iptables -nvL -t nat
Chain PREROUTING (policy ACCEPT 0 packets, 0 bytes)
  pkts bytes target    prot opt in     out     source               destination
    0      0 DNAT      tcp  --  *      *       0.0.0.0/0           192.168.1.146      tcp dpt:801 to:10.1.0.1:80
    2    104 DNAT      tcp  --  *      *       0.0.0.0/0           192.168.1.146      tcp dpt:3389 to:10.1.0.6:3389

Chain INPUT (policy ACCEPT 0 packets, 0 bytes)
  pkts bytes target    prot opt in     out     source               destination

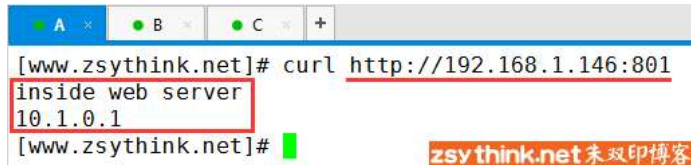
Chain OUTPUT (policy ACCEPT 0 packets, 0 bytes)
  pkts bytes target    prot opt in     out     source               destination

Chain POSTROUTING (policy ACCEPT 0 packets, 0 bytes)
  pkts bytes target    prot opt in     out     source               destination
    7    420 SNAT      all  --  *      *       10.1.0.0/16          0.0.0.0/0           to:192.168.1.146
[www.zsythink.net]#
```

zsythink.net 朱双印博客

如上图所示，我们将公司公网IP的801号端口映射到了公司内网中C主机的80端口，所以，当外网主机访问公司公网IP的801端口时，报文将会发送到C主机的80端口。这次，我们不用再次定义SNAT规则了，因为之前已经定义过SNAT规则，上次定义的SNAT规则只要定义一次就行，而DNAT规则则需要根据实际的情况去定义。

好了，完成上述DNAT映射后，我们在A主机上访问B主机的801端口试试，如下



```
[www.zsythink.net]# curl http://192.168.1.146:801
inside web server
10.1.0.1
[www.zsythink.net]#
```

zsythink.net 朱双印博客

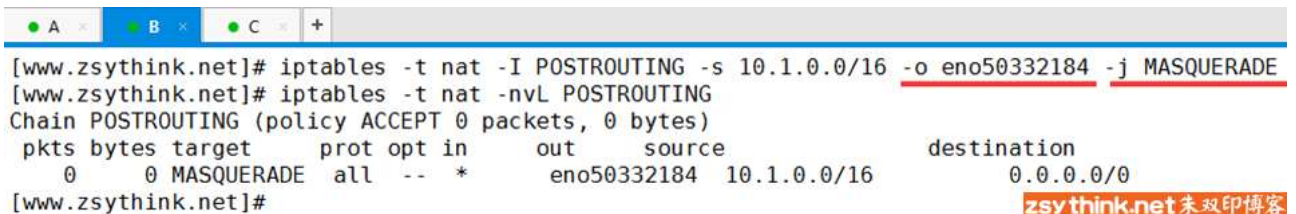
可以看到，我们访问的是B主机的公网IP，但是返回结果显示的却是C主机提供的服务内容，证明DNAT已经成功。

而上述过程中，外网主机A访问的始终都是公司的公网IP，但是提供服务的却是内网主机，但是我们可以对外宣称，公网IP上提供了某些服务，快来访问吧！我觉得我说明白了，你听明白了吗？

动作MASQUERADE

上文中，我们已经描述了SNAT，也就是源地址转换，那么我们现在来认识一个与SNAT类似的动作：MASQUERADE

当我们拨号上网时，每次分配的IP地址往往不同，不会长期分给我们一个固定的IP地址，如果这时，我们想要让内网主机共享公网IP上网，就会很麻烦，因为每次IP以后，我们都要重新配置SNAT规则，这样显示不是很人性化，我们通过MASQUERADE即可解决这个问题，MASQUERADE会动态的将源地址转换为可用的IP地址。实现的功能完全一致，都是修改源地址，只不过SNAT需要指明将报文的源地址改为哪个IP，而MASQUERADE则不用指定明确的IP，会动态的将报文的源地址修改为可用的IP地址，示例如下：



```
[www.zsythink.net]# iptables -t nat -I POSTROUTING -s 10.1.0.0/16 -o eno50332184 -j MASQUERADE
[www.zsythink.net]# iptables -t nat -nvL POSTROUTING
Chain POSTROUTING (policy ACCEPT 0 packets, 0 bytes)
  pkts bytes target    prot opt in     out     source               destination
    0      0 MASQUERADE all  --  *      *       10.1.0.0/16          0.0.0.0/0
[www.zsythink.net]#
```

zsythink.net 朱双印博客

如上图所示，我们指定，通过外网网卡出去的报文在经过POSTROUTING链时，会自动将报文的源地址修改为外网网卡上可用的IP地址，这时，即使外网网卡中的IP地址发生了改变，也能够正常的、动态的将内部主机的报文的源IP映射为对应的公网IP。

可以把MASQUERADE理解为动态的、自动化的SNAT，如果没有动态SNAT的需求，没有必要使用MASQUERADE，因为SNAT更加高效。

动作REDIRECT

使用REDIRECT动作可以在本机上进行端口映射

比如，将本机的80端口映射到本机的8080端口上

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

经过上述规则映射后，当别的机器访问本机的80端口时，报文会被重定向到本机的8080端口上。

REDIRECT规则只能定义在PREROUTING链或者OUTPUT链中。

小结

