Contents

NAT using iptables

iptables provides very flexible NAT options. We can do static source NAT, static destination NAT, dynamic NAT, etc. with the help of iptables NAT features

Using iptables for nat configuration between two physical interfaces

If some machine, say proxy server, has two interfaces one local and one public. Then we can use iptables to NAT outgoing packets with proxy servers public IP. The configuration syntax would be

```
iptables -t nat -I POSTROUTING -s ! <proxy public IP> -o <proxy public interface> -j SNAT --to-sourc
```

Using iptables for nat configuration between two aliased interfaces

If we have two aliased interfaces like eth0 and eth0:1 and we want to configure NAT between them. Then it is more complex then configuring NAT between two physically different interfaces(like eth0, eth1), as iptables will treat both interfaces as eth0. Even kernel routing table will also treat both interfaces as just eth0.

To configure NAT in such scenario we must know which private IP address can contact on interface with private IP, say eth0. Assuming private IP address ranges 10.0.0.0/8, 172.16.0.0/12 and 192.168.0.0/16 we can use below configuration

```
iptables -t nat -I POSTROUTING -s ! 196.12.53.9 -d 10.0.0.0/8 -j CONNMARK --set-mark 0x1
iptables -t nat -I POSTROUTING -s ! 196.12.53.9 -d 172.16.0.0/12 -j CONNMARK --set-mark 0x1
iptables -t nat -I POSTROUTING -s ! 196.12.53.9 -d 192.168.0.0/16 -j CONNMARK --set-mark 0x1
iptables -t nat -I POSTROUTING -s ! 196.12.53.9 -m connmark ! --mark 0x1 -j SNAT --to-source 196.12.
```

The first three lines set mark 1 on all packets which are destined for local network. The last line tells to NAT all packets which are not going to local machines and are not going outside with public IP. In other words NAT with public IP if packets are going outside and are not already having public IP.

Using NAT to forward ports from one machine to another

We need to add both PREROUTING and POSTROUTING rules when we try to forward ports from one machine to another, as shown below:

```
iptables -A PREROUTING -t nat -p tcp -d <ip_of_nat_source> --dport 40 -j DNAT --to-destination <dest
iptables -A POSTROUTING -t nat -p tcp -j SNAT --to-source <ip_of_nat_source>
```

Note that without the second rule destination machine may try to reply to tcp connection source machine directly. Also destination machine will try to reply from port 22 but tcp connection source machine had initiated connection to port 40 of nat source machine. Hence it would not recognize this reply and send TCP reset. Therefore source natting is required whenever we try to forward ports from one machine to another, even if port number is same.

Using NAT on Linux machine with two interfaces used as router

NATting all outgoing connections

We can use NAT on Linux machine with two interfaces and use it as router to create LAN segment. In this scenario we have to source NAT all packets coming from local network and going outside with machines LAN segment IP. That can be achieved by:

```
-A POSTROUTING -s 10.0.0.0/8 -o <WAN_interface>  -j SNAT --to-source <machines_LAN_IP>
-A POSTROUTING -s 172.16.0.0/12 -o <WAN_interface>    -j SNAT --to-source <machines_LAN_IP>
-A POSTROUTING -s 192.168.0.0/16 -o <WAN_interface>  -j SNAT --to-source <machines_LAN_IP>
```

- Note that for this to work the default router of layer 3 switch should be machines local LAN IP or LAN segment IP.

NATting public IPs and Local IPs

We can also static NAT public IPs to local machines in network without actually putting them in same LAN as machines LAN segment. This way we can assign public IPs without changing machines actual VLAN or IP address. We would also have flexibility of changing local IP, public IP etc. which is not so easy if we directly configure public IP on machine. We can also block and log all incoming and outgoing connections with this setup.

```
*nat
-A PREROUTING  -d <public_IP> -i <WAN_interface> -j DNAT --to-destination <Local_IP>
-A POSTROUTING -s <Local_IP> -o <WAN_interface>  -j SNAT --to-source <Public_IP>
*filter
-A FORWARD -s <Public_IP> -j ACCEPT
-A FORWARD -d <Public_IP> -j ACCEPT
-A FORWARD -s <Local_IP> -j ACCEPT
-A FORWARD -d <Local_IP> -j ACCEPT
```

- Note that in above rules <Public_IP> refers to public IP we want to NAT to machine and <Local_IP> refers to existing LAN local IP of machine. None of the above IPs are IPs of machine being used as router.

Allowing forwarding of public IPs

If we do not NAT the public IPs and connect servers directly in Public IP VLAN and configure public IPs directly on servers, then we have to allow those public IPs to be forwarded through router machine's firewall. To do that we can use rules like:

```
-A FORWARD -s <Public_IP> -j ACCEPT
-A FORWARD -d <Public_IP> -j ACCEPT
```

Until we use the above rules, even if configure public IPs directly on machines they will not work.

- Note that instead of allowing everything we can allow only selected protocols and ports to provide firewall protection to public servers from routers machines iptables.

NAT pitfalls

Note that for NAT to work:

- IP forwarding should be enabled. Hence, `sysctl net.ipv4.ip_forward` should return 1. Also we should make the same change in file `/etc/sysctl.conf`
- iptables filter tables, FORWARD chain must allow packets to be forwarded.

- Very often rule '`-A FORWARD -m state --state RELATED,ESTABLISHED -j ACCEPT`' is forgotten.
- If you are giving command on terminal then we have to escape '!' with '\' so that shell does not interprets it differently.