PARTE 1

Vagrantfile

Vagrantfile

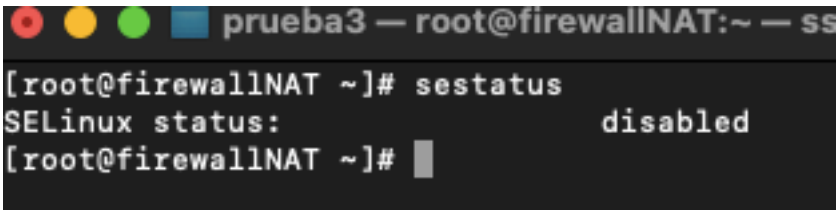
```
# -*- mode: ruby -*-
# vi: set ft=ruby :

Vagrant.configure("2") do |config|

  config.vm.define :clienteNAT do |clienteNAT|
    clienteNAT.vm.box = "generic/centos9s"
    clienteNAT.vm.network :private_network, ip: "192.168.50.2"
    clienteNAT.vm.hostname = "clienteNAT"
  end

  config.vm.define :firewallNAT do |firewallNAT|
    firewallNAT.vm.box = "generic/centos9s"
    firewallNAT.vm.network :private_network, ip: "192.168.50.3"
    firewallNAT.vm.hostname = "firewallNAT"
  end
end
```

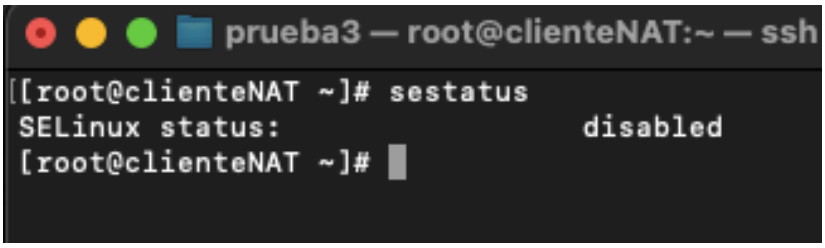
Deshabilitamos el SELinux en ambas maquinas:
`vim /etc/selinux/config`

prueba3 — root@firewallNAT:~ — ssh ‹ vagrant ssh firewallNAT — 7...

```
#
# NOTE: In earlier Fedora kernel builds, SELINUX=disabled would also
# fully disable SELinux during boot. If you need a system with SELinux
# fully disabled instead of SELinux running with no policy loaded, you
# need to pass selinux=0 to the kernel command line. You can use grubby
# to persistently set the bootloader to boot with selinux=0:
#
#    grubby --update-kernel ALL --args selinux=0
#
# To revert back to SELinux enabled:
#
#    grubby --update-kernel ALL --remove-args selinux
#
SELINUX=disabled
# SELINUXTYPE= can take one of these three values:
#     targeted - Targeted processes are protected,
#     minimum - Modification of targeted policy. Only selected processes a
re protected.
#     mls - Multi Level Security protection.
SELINUXTYPE=targeted
```

SELinux disabled





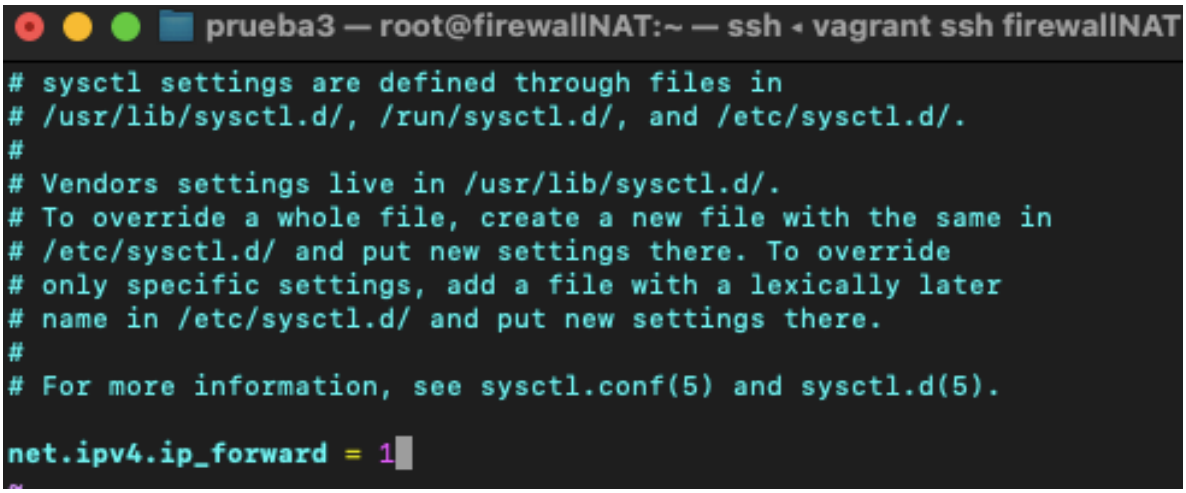
Permitir el reenvió de paquetes

Se debe modificar el archivo /etc/sysctl.conf: EN firewallNAT

```
vim /etc/sysctl.conf
```

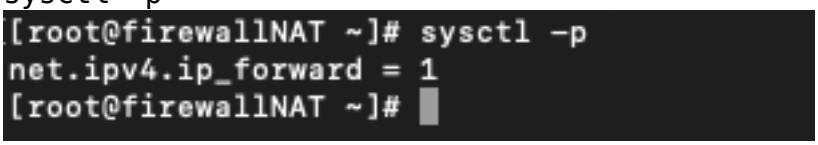
Agregamos:

```
net.ipv4.ip_forward = 1
```



Comprobamos su funcionamiento:

```
sysctl –p
```



Iniciamos el firewalld

```
service firewalld start
```

```
service firewalld status
```

```
[[root@firewallNAT ~]# service firewalld status
Redirecting to /bin/systemctl status firewalld.service
● firewalld.service - firewalld - dynamic firewall daemon
     Loaded: loaded (/usr/lib/systemd/system/firewalld.service; enabled; preset>
     Active: active (running) since Sat 2023-09-16 18:05:05 UTC; 10min ago
       Docs: man:firewalld(1)
   Main PID: 673 (firewalld)
      Tasks: 2 (limit: 11129)
     Memory: 43.9M
        CPU: 1.270s
     CGroup: /system.slice/firewalld.service
             └─673 /usr/bin/python3 -s /usr/sbin/firewalld --nofork --nopid

Sep 16 18:05:03 firewallNAT systemd[1]: Starting firewalld - dynamic firewall d>
Sep 16 18:05:05 firewallNAT systemd[1]: Started firewalld - dynamic firewall da>
[
```

Definimos las zonas

Verificamos zonas:

```
firewall-cmd --get-zones
```

```
prueba3 — root@firewallNAT:~ — ssh ◂ vagrant ssh firewallNA
[[root@firewallNAT ~]# firewall-cmd --get-zones
block dmz drop external home internal nm-shared public trusted work
[root@firewallNAT ~]#
```

Verificamos zonas activas:

```
firewall-cmd --get-active-zones
```

```
[[root@firewallNAT ~]# firewall-cmd --get-active-zones
public
  interfaces: eth0 eth1
```

Usamos los siguientes comandos para acomodar las zonas:

```
firewall-cmd  --zone=public --remove-interface=eth1
firewall-cmd  --zone=internal --add-interface=eth1
```

Deben quedar asi:

```
[[root@firewallNAT ~]# firewall-cmd --get-active-zones
internal
  interfaces: eth1
public
  interfaces: eth0
[root@firewallNAT ~]#
```

Definir reglas de reenvio del NAT

```
firewall-cmd --direct --add-rule ipv4 nat POSTROUTING 0 -o eth0 -j
MASQUERADE

firewall-cmd --direct --add-rule ipv4 filter FORWARD 0 -i eth1 -o
eth0 -j ACCEPT

firewall-cmd --direct --add-rule ipv4 filter FORWARD 0 -i eth0 -o
eth1 -m state --state RELATED,ESTABLISHED -j ACCEPT
```

```
[root@firewallNAT ~]# firewall-cmd --direct --add-rule ipv4 nat POSTROUTING 0 -o eth0
 -j MASQUERADE
success
[root@firewallNAT ~]# firewall-cmd --direct --add-rule ipv4 filter FORWARD 0 -i eth1
-o eth0 -j ACCEPT
success
[root@firewallNAT ~]# firewall-cmd --direct --add-rule ipv4 filter FORWARD 0 -i eth0
-o eth1 -m state --state RELATED,ESTABLISHED -j ACCEPT
success
[root@firewallNAT ~]#
```

Verificamos las reglas con:

```
firewall-cmd --direct --get-all-rules
```

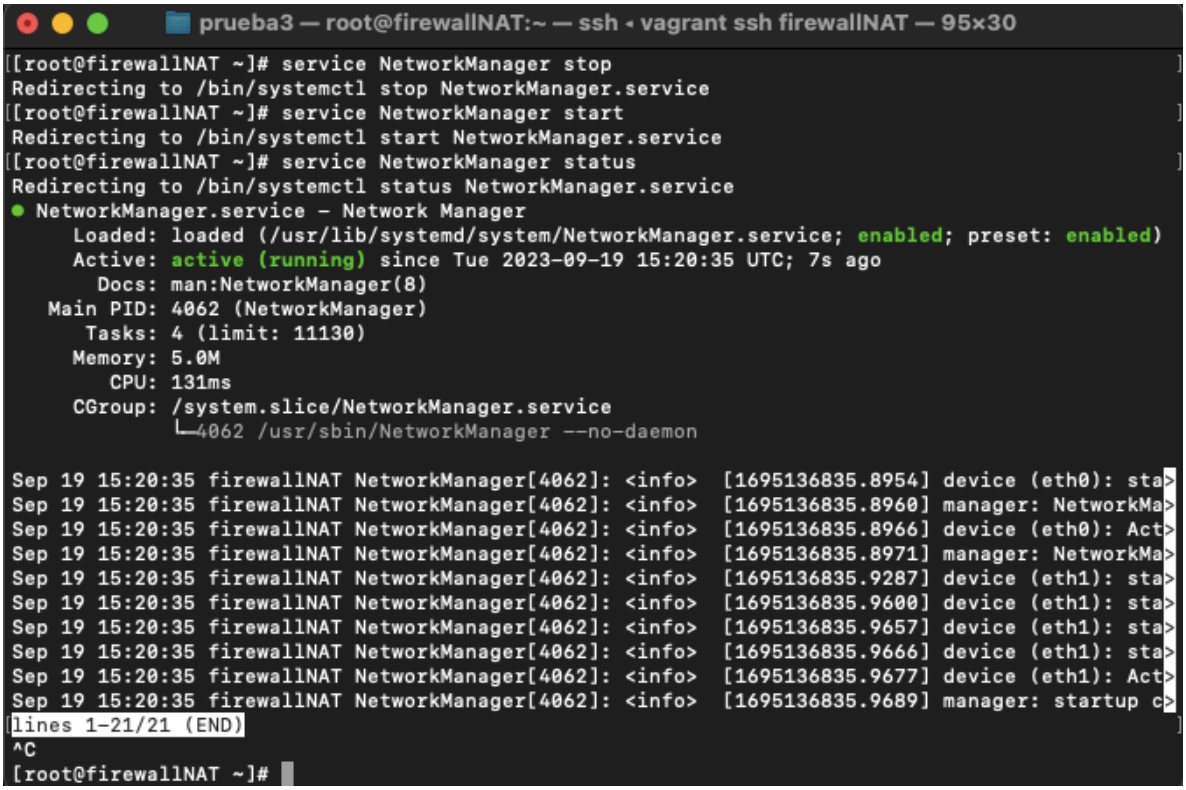
```
[root@firewallNAT ~]# firewall-cmd --direct --get-all-rules
ipv4 nat POSTROUTING 0 -o eth0 -j MASQUERADE
ipv4 filter FORWARD 0 -i eth1 -o eth0 -j ACCEPT
ipv4 filter FORWARD 0 -i eth0 -o eth1 -m state --state RELATED,ESTABLISHED -j ACCEPT
[root@firewallNAT ~]#
```

Añadir servicios a las Zonas

```
firewall-cmd --zone=public --add-service=http
firewall-cmd --zone=public --add-service=https
firewall-cmd --zone=public --add-service=dns

firewall-cmd --zone=internal --add-service=http
firewall-cmd --zone=internal --add-service=https
firewall-cmd --zone=internal --add-service=dns
```

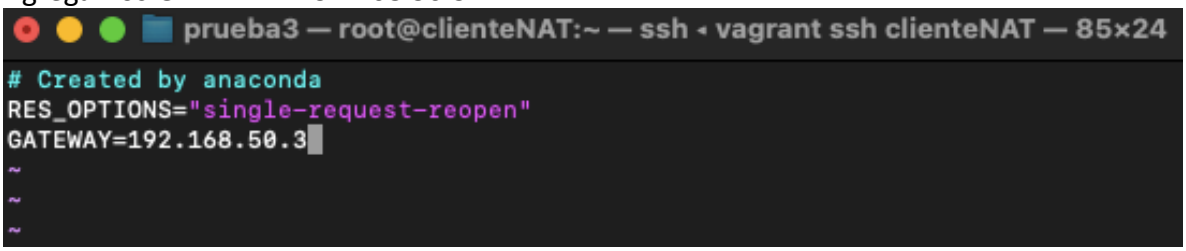
```
[root@firewallNAT ~]# firewall-cmd --zone=public --add-service=http
success
[root@firewallNAT ~]# firewall-cmd --zone=public --add-service=https
success
[root@firewallNAT ~]# firewall-cmd --zone=public --add-service=dns
success
[root@firewallNAT ~]# firewall-cmd --zone=internal --add-service=http
success
[root@firewallNAT ~]# firewall-cmd --zone=internal --add-service=https
success
[root@firewallNAT ~]# firewall-cmd --zone=internal --add-service=dns
success
[root@firewallNAT ~]#
```

Detenemos y volvemos a levanter el NetworkManager



CONFIGURACION CLIENTE (Puerta de enlace)

Vamos a:
vim /etc/sysconfig/network
Agregamos GATEWAY=192.168.50.3



Reiniciamos la maquina:
Reboot

Instalamos el netstat:
`yum install net-tools`

```
[[root@clienteNAT ~]# sudo yum install net-tools
Extra Packages for Enterprise Linux 9 - x86_64   61 kB/s |  66 kB     00:01
Extra Packages for Enterprise Linux 9 - Next -   62 kB/s |  62 kB     00:00
Package net-tools-2.0-0.62.20160912git.el9.x86_64 is already installed.
Dependencies resolved.
Nothing to do.
Complete!
```

Verificamos si usamos el GATEWAY:
`netstat -rn`

```
[[root@clienteNAT ~]# netstat -rn
Kernel IP routing table
Destination     Gateway         Genmask         Flags   MSS Window  irtt Iface
0.0.0.0         192.168.50.3    0.0.0.0         UG        0 0          0 eth1
0.0.0.0         10.0.2.2        0.0.0.0         UG        0 0          0 eth0
10.0.2.0        0.0.0.0         255.255.255.0   U         0 0          0 eth0
192.168.50.0    0.0.0.0         255.255.255.0   U         0 0          0 eth1
[root@clienteNAT ~]#
```

Borramos una de las rutas de salida con:
`sudo route del -net 0.0.0.0 gw 10.0.2.2 netmask 0.0.0.0 dev eth0`

Hacemos una prueba
`ping -I eth1 8.8.8.8`

```
[[root@clienteNAT ~]# ping -I eth1 8.8.8.8
PING 8.8.8.8 (8.8.8.8) from 192.168.50.4 eth1: 56(84) bytes of data.
64 bytes from 8.8.8.8: icmp_seq=1 ttl=61 time=32.9 ms
64 bytes from 8.8.8.8: icmp_seq=2 ttl=61 time=33.9 ms
64 bytes from 8.8.8.8: icmp_seq=3 ttl=61 time=34.3 ms
64 bytes from 8.8.8.8: icmp_seq=4 ttl=61 time=35.5 ms
64 bytes from 8.8.8.8: icmp_seq=5 ttl=61 time=32.5 ms
^C
--- 8.8.8.8 ping statistics ---
5 packets transmitted, 5 received, 0% packet loss, time 3996ms
rtt min/avg/max/mdev = 32.459/33.807/35.490/1.077 ms
[root@clienteNAT ~]#
```