Lab 3:

Linux   
network settings

Linux Server Security  
 2024-2025

© Daan Pareit

## Introduction

# Lab concept

We will create the topology below, using the VMs you created previously. The Debian and RHEL will be part of the local (NAT) network ‘behind’ the pfSense, rather than being attached directly to VMware’s default NAT network.



# Learning goals

* Configuring network interface in Debian
* Configuring network interface in RHEL

# Practicalities and prerequisites

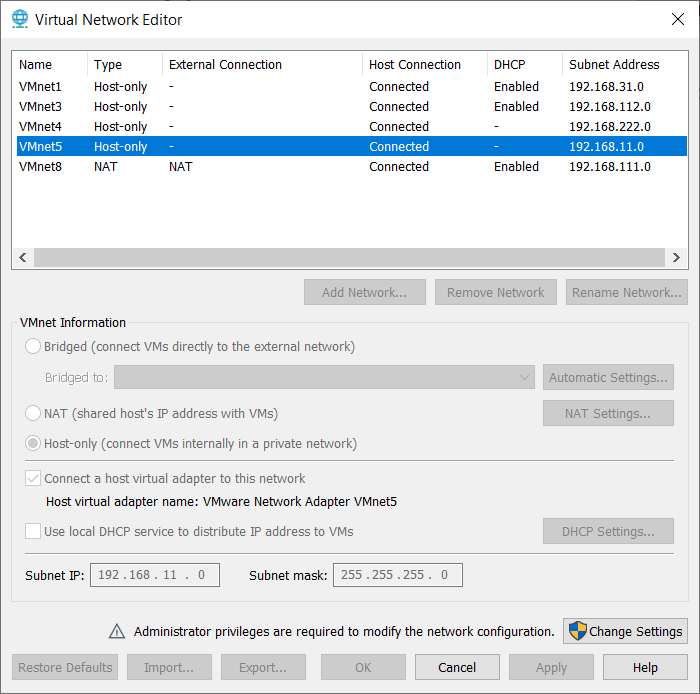
You’ll need the following:

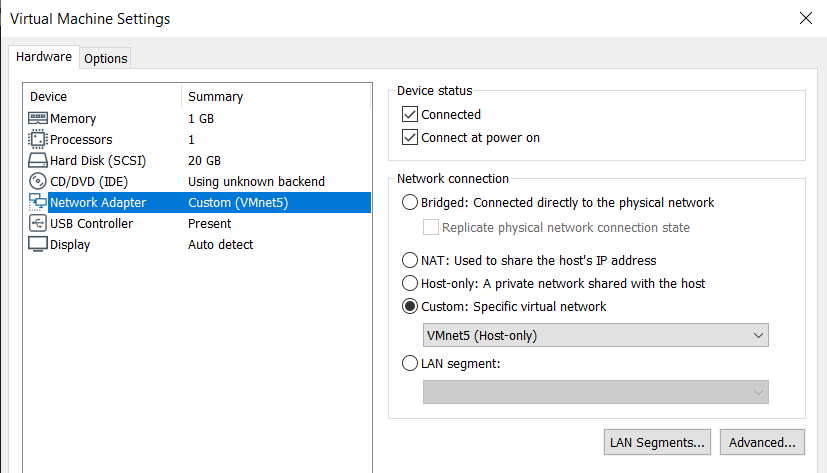
* Debian, RHEL and pfSense VMs as installed in previous labs

## VMware settings: connect RHEL and Debian to your pfSense

🡺 Verify which VMnet (e.g. VMnet5) you allocated 192.168.11.0/24 in the ‘Virtual Network editor’

🡺 Connect your RHEL and Debian to this same VMnet (which would in real life be the equivalent of plugging the network cable from your server to a different switch). Your pfSense should already have its second interface (its ‘LAN’ interface) connected to this VMnet.





## Debian network settings

A server doesn’t typically obtain an IP address via DHCP but has a fixed IP address. We’ll set Debian’s IP address statically to **192.168.11.10/24** and set its default gateway to pfSense’s IP address (of its LAN interface).

# Change the IP configuration at runtime, without rebooting nor changing config files

Let’s use various *ip* commands to give the interface of your Debian its fixed IP address and default gateway. Note: avoid using *ifconfig*, *route* etc. These are still commonly used but are deprecated and replaced by the *ip* command.

🡺 Consult the manpage of *ip* to get an idea of its functionality. At one of the latest sections, “SEE ALSO”, it refers to other specific man pages as well, most notably ‘ip-address’ and ‘ip-route’ with specific info for adding/deleting IP addresses and IP routes.

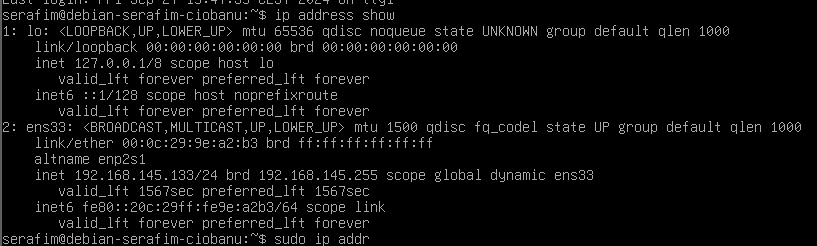
🡺 Now, use the *ip* command to ‘delete’ the dynamically received IP address. (Hint: keyword ‘flush’)

🡺 Then, use the *ip* command to add the static IP address 192.168.11.10/24.

è Now, use the *ip* command to add pfSense’s IP address on its LAN interface as Debian’s default gateway.

🡺 Verify your IP address and IP routing table using ***ip address show*** and ip ***route show.*** These are your basic commands for future checking/troubleshooting IP connectivity.

ip address show



sudo ip addr flush dev ens33

A screenshot of a computer

Description automatically generatedsudo ip addr add 192.168.11.10/24 dev ens33

A screenshot of a computer

Description automatically generatedsudo ip route add default via 192.168.11.254

A screenshot of a computer

Description automatically generated

ip addr show + ip route

A screenshot of a computer program

Description automatically generated

# Make the IP configuration persistent, the Debian legacy way (aka ‘ifupdown’)

All settings made with the *ip* tool are taking effect immediately but are lost at reboot: verify this by rebooting your VM. To specify the IP configuration at boot time, we need to define this in a config file somewhere. There are two ways to do so. Let’s start with the most-known method.

🡺 To make the fixed IP addresses also persistent after reboot in Debian, you’ll have to change the **/etc/network/interfaces** file. Have a look at the manpage of *interfaces* for more information about this file and for example configs.

🡺 Change the /etc/network/interfaces file for your ethernet interface to have a static IP address 192.168.11.10/24 and to use 192.168.11.254 as default gateway.

🡺 Apply the changes by disabling/enabling the interface by performing a sudo **ifdown** ens33 and sudo **ifup** ens33 (or reboot).

🡺 Verify your IP address and IP routing table using ***ip address show*** and ***ip route show.*** These are your basic commands for checking/troubleshooting IP connectivity.

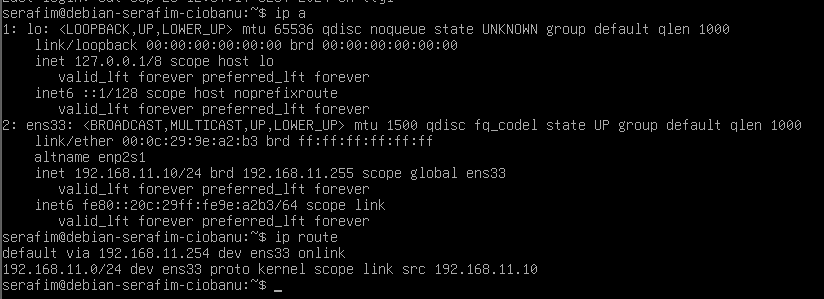
🡺 Check if you can ping pfSense (192.168.11.254) and the VMware gateway in VMnet8 (192.168.x.2). If you can’t, check your routing table via *ip route show*.

A computer screen shot of a computer

Description automatically generated

sudo ifdown ens33 && sudo ifup ens33

reboot and check



Pings also are supposed to work (.11.254 + .145.2)

A screenshot of a computer program

Description automatically generated

# Make the IP configuration persistent, the new way (aka ‘networkd’)

Now, let’s try another way to configure IP addresses in Debian-based distro, using ini-style text config ‘.network’ files instead.

🡺 The previous ifupdown way could be used simultaneously in parallel for other interfaces or network settings. However, to avoid confusion, let’s disable it completely by renaming /etc/network/interfaces to e.g. /etc/network/interfaces.unused . You can verify that after a reboot, you no longer receive an IP address (not statically nor dynamically)

sudo mv interfaces interfaces.unused

🡺 Create a “<2 digits preferably>-<name\_of\_your\_choice>**.network**” config file in **/etc/systemd/network** , e.g. /etc/systemd/network/20-static.network . The man page ‘systemd.network’ gives information on how to write your config. Here, specify the interface and its IP settings. I’m using ‘ens33’ as the name of the network interface here, but it could be different in your situation.

[Match]

Name=ens33

[Network]

Address=192.168.11.10/24

Gateway=192.168.11.254

cd /etc/systemd/network

sudo vim 20-static.network

A screen shot of a computer

Description automatically generated

🡺 Use the command: “sudo systemctl enable --now **systemd-networkd**” to immediately apply your network config and ensure the config is loaded at boot time. (More on systemctl and systemd in a next lesson.)

🡺 Verify your IP address and IP routing table using ***ip address show*** and ***ip route show*** . These are your basic commands for checking/troubleshooting IP connectivity.

A screenshot of a computer

Description automatically generated

🡺 If you would change your config file, you need to reload it via “**networkctl reload**”.

è The “**networkctl**” command gives you extra possibilities to check your network config when using systemd-networkd, try e.g. using networkctl list and networkctl status .

A screenshot of a computer

Description automatically generated

On a side note, the systemd-networkd also allows you to give your network interface an easier to remember name than ‘ens33’ or its alternative name ‘enp2s1’. The reason Debian gave it the ‘ens33’ name is because the (virtual) ethernet controller is installed on PCI Express hotplug slot index 33. That is located on PCI 02:01.0, hence its alternative name ‘enp2s1’ (“PCI bus 2, slot 1”). To give it a more ‘human’ name, such as ‘lan0’ or ‘eth0’, you can use .link files.

🡺 Create a “<2 digits preferably>-<name\_of\_your\_choice>**.link**” config file in **/etc/systemd/network** , e.g. /etc/systemd/network/10-lan0.link . The man page ‘systemd.link’ gives information on how to write your config. You can use e.g. the following to give ens33 two other alternative names ‘lan0’ and ‘eth0’:

[Match]

OriginalName=ens33

[Link]

AlternativeName=lan0

AlternativeName=eth0

sudo vim 10-lan0.link

A black screen with white text

Description automatically generated

🡺 This will now be applied at boot time. Or apply it immediately by using “sudo **systemctl restart systemd-udev-trigger.service**” to make systemd-networkd apply this new .link file.

A screenshot of a computer

Description automatically generated

Not the most useful function I guess

# DNS name server

The name server to be used is stated in the **/etc/resolv.conf** file, which can dynamically be overwritten when using DHCP, or manually specified if you use a static IP address.

🡺 Check if your DNS name server is set to your pfSense VM in the /etc/resolv.conf file.

🡺 Check if you can successfully access google.be via wget (with pfSense running).

Had to manually change the IP address of the nameserver in the /etc/resolv.conf

A screenshot of a computer

Description automatically generated

A screenshot of a computer

Description automatically generated

wget also works

## RHEL network settings

A server doesn’t typically obtain an IP address via DHCP but has a fixed IP address. We’ll set RHEL’s IP address to **192.168.11.20/24** .

Fun fact that it already got some default values from pfsense I believe

A screenshot of a computer

Description automatically generated

# Change the IP configuration at runtime, without rebooting nor changing config files

Let’s use various *ip* commands to give the interface of your RHEL its fixed IP address and default gateway, similarly as you did for your Debian machine. Same note as before: avoid using *ifconfig*, *route* etc. These are still commonly used but are deprecated and replaced by the *ip* command.

THIS METHOD DOES NOT WANT TO WORK IN MY CASE

🡺 Now, use the *ip* command to ‘throw away’ the dynamically received IP address. (Hint: keyword ‘flush’)

sudo ip addr flush dev ens160

The command would not work, so I just shutdown pfsense machine for now. Do not know what am I doing wrong, but yet again it is the IP address that is getting assigned by the DHCP service.

🡺 Then, use the *ip* command to add the static IP address 192.168.11.20/24.

sudo ip addr add 192.168.11.20/24 dev ens160

è Now, use the *ip* command to add pfSense’s IP address on its LAN interface as RHEL’s default gateway.

sudo ip route add default via 192.168.11.254

🡺 Verify your IP address and IP routing table using ***ip address show*** and ***ip route show*** . These are your basic commands for checking/troubleshooting IP connectivity, also on RHEL.

ip route show

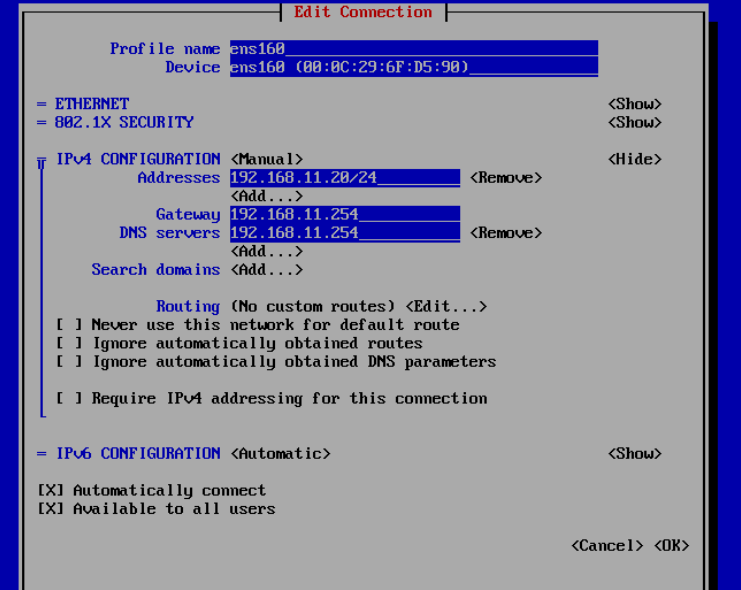
ip address show

# Make the IP configuration persistent, using NetworkManager

As you know from playing around on your Debian, all settings made with the *ip* tool are taking effect immediately but are lost at reboot. You could again verify this by rebooting your VM. To specify the IP configuration at boot time, we again need to define this in a config file somewhere. The default way to do so in RHEL-based distros is different from the Debian way(s). RHEL uses ‘NetworkManager’.

è To make the fixed IP addresses also persistent after reboot in RHEL, you’ll have to configure the interface via the command **nmtui** (= NetworkManager Text User Interface).

è In this tool, fill in the correct IP, netmask, gateway, DNS server.



🡺 Note that the file **“/etc/NetworkManager/system-connections/<your-interface-name>.nmconnection”** will be changed according to your settings.

🡺 As in Debian, the config file is not continuously monitored. You need to reload this changed config file using the **nmcli** (=NetworkManager Command Line Interface) tool:   
“**nmcli connection up** ens160”

This one would be useful before I rebooted, hehe

🡺 Check if you can ping pfSense (192.168.11.254) and the VMware gateway in VMnet8 (192.168.x.2). If you can’t, check your routing table via *ip route show*.

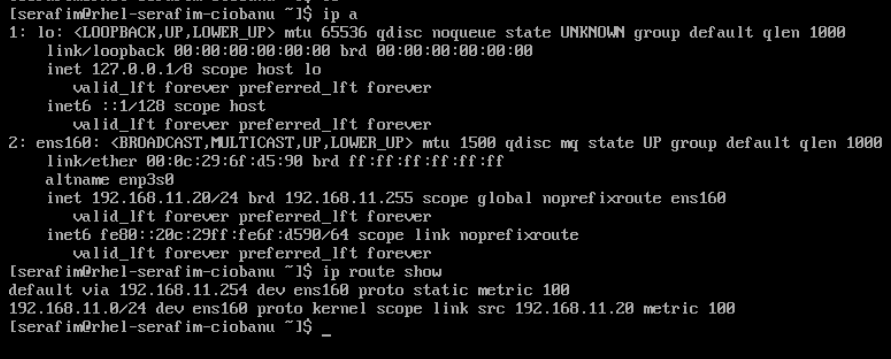
A screenshot of a computer

Description automatically generated

The ping works

A screenshot of a computer

Description automatically generated

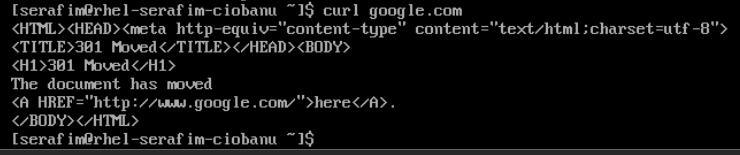


# DNS name server

In RHEL, the /etc/resolv.conf file is managed by NetworkManager.

🡺 Check if your DNS name server is set to your pfSense VM in the /etc/resolv.conf file. If not, run nmtui again to set the DNS name server here if you forgot to do so.

🡺 Check if you can successfully access google.be via curl.



The DNS server was given as from DHCP server by default, as the IP address, however even with the machine down, it would all work strange.

When changed with the `nmtui` is worked perfectly