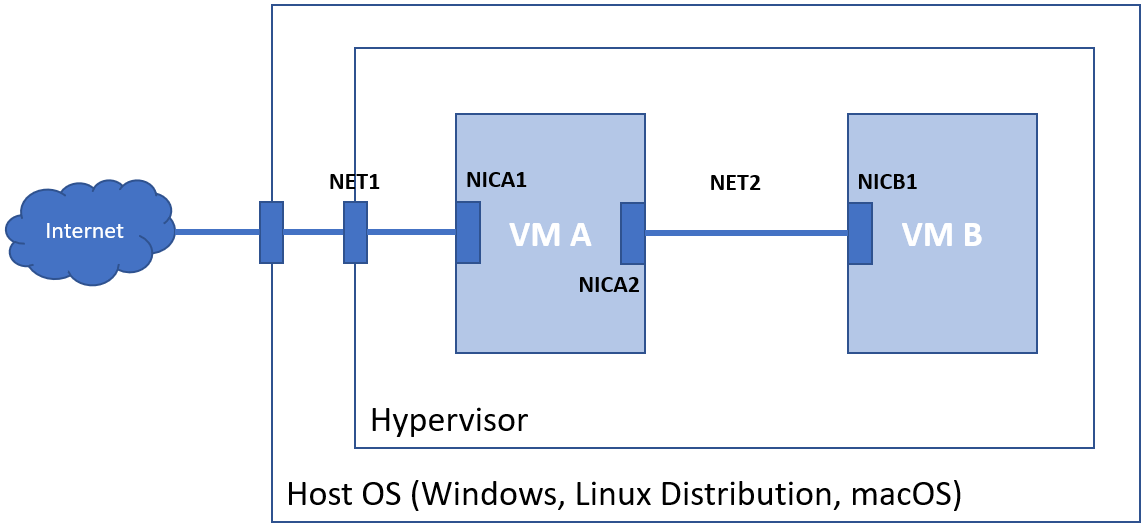
# Solution M2: Network-and-System-Security (Ubuntu)

For both tasks we assume that we are working in the following infrastructure:



Additional assumptions (can be seen on the picture above):

* **VM A** has **two** NICs (**NICA1** and **NICA2**):
  + has connectivity to the Internet
  + first (**NICA1**), connected via external/bridged/NAT mode to the external world (**NET1**)
    - network settings (IP address, subnet mask, gateway, and DNS) match the connected network
  + second (**NICA2**), connected in an internal/private/host-only mode network (**NET2**)
    - with static IP address in the same network as the only NIC (**NICB1**) of the VM B
    - for example, ***IP=192.168.200.1*** *and* ***MASK=255.255.255.0***
* **VM B** has **one** NIC (**NICB1**):
  + connected in an internal/private/host-only mode network (**NET2**)
  + the only NIC (**NICB1**) is set with static IP address in the same network as the second NIC (**NICA2**) of the **VM A** and with gateway set to the IP address of the second NIC (**NICA2**) of **VM A** and DNS set to a public one, for example 8.8.8.8
  + for example, ***IP=192.168.200.2, MASK=255.255.255.0, GW=192.168.200.1,*** *and* ***DNS=8.8.8.8***

# Task 1

**Challenge:** Research and implement two-node network (one machine with two NICs and the second with one) with NAT capabilities based on **firewalld**/**ufw**

**Solution (ufw):**

Before we start to tackle the challenge, we must validate our setup by

* On **VM A**
  + Can successfully ping publicly available resource by IP address, for example **8.8.8.8**
  + Can successfully ping publicly available resource by name, for example **abv.bg**
  + Can successfully ping **NICB1** of **VM B**, for example **192.168.200.2**
* On **VM B**
  + Can successfully ping **NICA2**, for example **192.168.200.1**

We must continue with the rest of the steps, only if the above checks are okay

#### VM A

First, we must make sure that the **ufw** is working

**sudo ufw status**

If not, we must enable it on boot and start it

**sudo ufw enable**

Next, we must change the default forwarding policy

**sudo vi /etc/default/ufw**

To match this

**DEFAULT\_FORWARD\_POLICY="ACCEPT"**

Save and close the file

Next, we must enable the **ip\_forward** flag

**sudo vi /etc/ufw/sysctl.conf**

Set it to (by uncommenting the line)

**net/ipv4/ip\_forward=1**

Save and close the file

Should we want to forward **IPv6** as well, then we must uncomment the appropriate lines as well

There is one more step. We must add the **NAT** rules

**sudo vi /etc/ufw/before.rules**

Add/insert the following lines at the first row

**\*nat**

**:POSTROUTING ACCEPT [0:0]**

**-A POSTROUTING -s 192.168.200.0/24 -o eth0 -j MASQUERADE**

**COMMIT**

Save and close the file

The final step is to reload the firewall

**sudo ufw reload**

That is all, that was required here. This turned our **VM A** to a router (of course, with NAT enabled)

#### VM B

By now, we should be done. Check that you

* Can successfully ping publicly available resource by IP address, for example **8.8.8.8**
* Can successfully ping publicly available resource by name, for example **abv.bg**

# Task 2

**Challenge:** Research and implement two-node network (one machine with two NICs and the second with one) with NAT capabilities based on **nftables**

**Solution:**

Before we start to tackle the challenge, we must stop and disable the firewall

**sudo ufw disable**

And then, validate our setup by

* On **VM A**
  + Can successfully ping publicly available resource by IP address, for example **8.8.8.8**
  + Can successfully ping publicly available resource by name, for example **abv.bg**
  + Can successfully ping **NICB1** of **VM B**, for example **192.168.200.2**
* On **VM B**
  + Can successfully ping **NICA2**, for example **192.168.200.1**

We must continue with the rest of the steps, only if the above checks are okay

#### VM A

Install the required packages

**sudo apt-get update**

**sudo apt-get install nftables**

Open the **nftables** configuration file

**sudo vi /etc/nftables.conf**

And append the following lines to the end

**table ip nat {**

**chain PREROUTING {**

**type nat hook prerouting priority filter; policy accept;**

**}**

**chain POSTROUTING {**

**type nat hook postrouting priority srcnat; policy accept;**

**oifname "eth0" masquerade**

**}**

**}**

Save and close the file

Check the status of the service

**systemctl status nftables**

Enable and start the service if not started (or restart it if already started)

**sudo systemctl enable --now nftables**

Check that the rules are there

**sudo nft list ruleset**

Check the **ip\_forward** flag

**cat /proc/sys/net/ipv4/ip\_forward**

Turn it **on** (**1**) if it is **off** (**0**)

**echo "1" | sudo tee /proc/sys/net/ipv4/ip\_forward**

#### VM B

By now, we should be (almost) done. Check that you

* Can successfully ping publicly available resource by IP address, for example **8.8.8.8**
* Can successfully ping publicly available resource by name, for example **abv.bg**

#### VM A

There is just one more thing. We must make sure that the **ip\_forward** flag will persist during reboot

Open the following configuration file

**sudo vi /etc/sysctl.conf**

Uncomment the line (#28)

**net.ipv4.ip\_forward=1**

Save and close the file

Apply the changes with

**sudo sysctl --system**