3 Beheer vanop afstand

3.1 Verbinding maken met je Server vanaf je hostcomputer

3.1.1 Let op met IP-nummer!

We veranderen het IP-adres van ServerXX naar 192.168.112.1.

We vragen eerst de huidige connectienaam op.

```
student@serverXX:~$ sudo nmcli con show

[sudo] wachtwoord voor student:

Warning: nmcli (1.52.0) and NetworkManager (1.48.10) versions don't match.
Restarting NetworkManager is advised.

NAME     UUID                                  TYPE      DEVICE

ens160   8f18258e-ee43-3d8a-8ed6-b452b3bc8045  ethernet  ens160

docker0  22924712-c569-41ed-b3d2-3321eeff411a  bridge    docker0

lo       4b771ea1-a470-470f-84ed-a6ef80af6342  loopback  lo
```

Zoals je ziet is de connectienaam ens160 aangezien docker0 te maken heeft met docker en lo de loopback is.

We stellen nu de nieuwe IP-nummer in.

```
student@serverXX:~$ sudo nmcli con mod ens160 ipv4.addresses 192.168.112.1/24
```

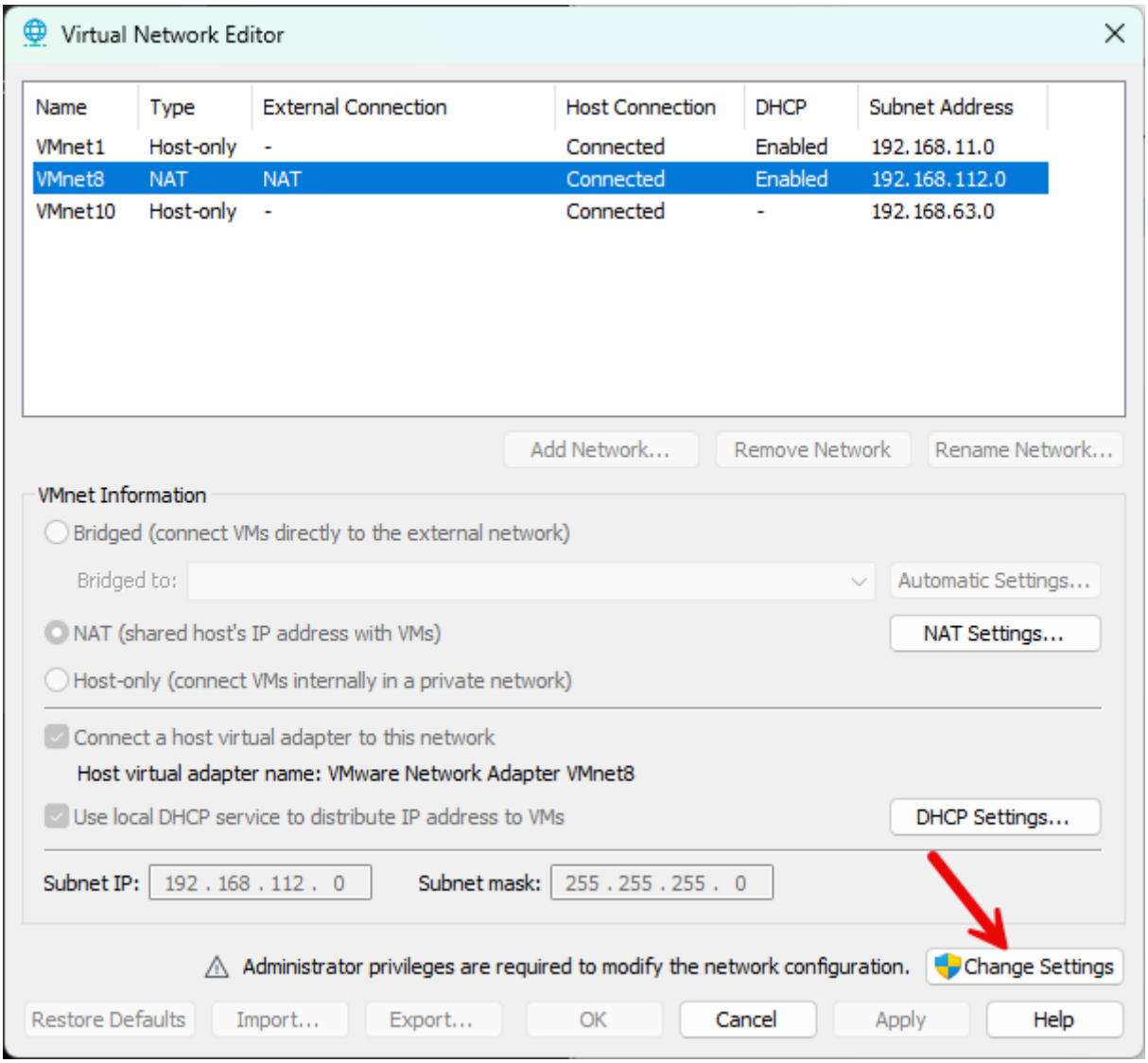
Herstart nu de netwerkverbinding.

```
student@serverXX:~$ sudo nmcli con down ens160 && sudo nmcli con up ens160

…
Fout: Activeren van verbinding is mislukt: Reserveren van de IP-configuratie is
mislukt (geen beschikbare adressen, time-out, enz.)

Hint: use 'journalctl -xe NM_CONNECTION=2318b781-c803-36d2-8642-bb3d5bb513f9 +
NM_DEVICE=ens160' to get more details.
```

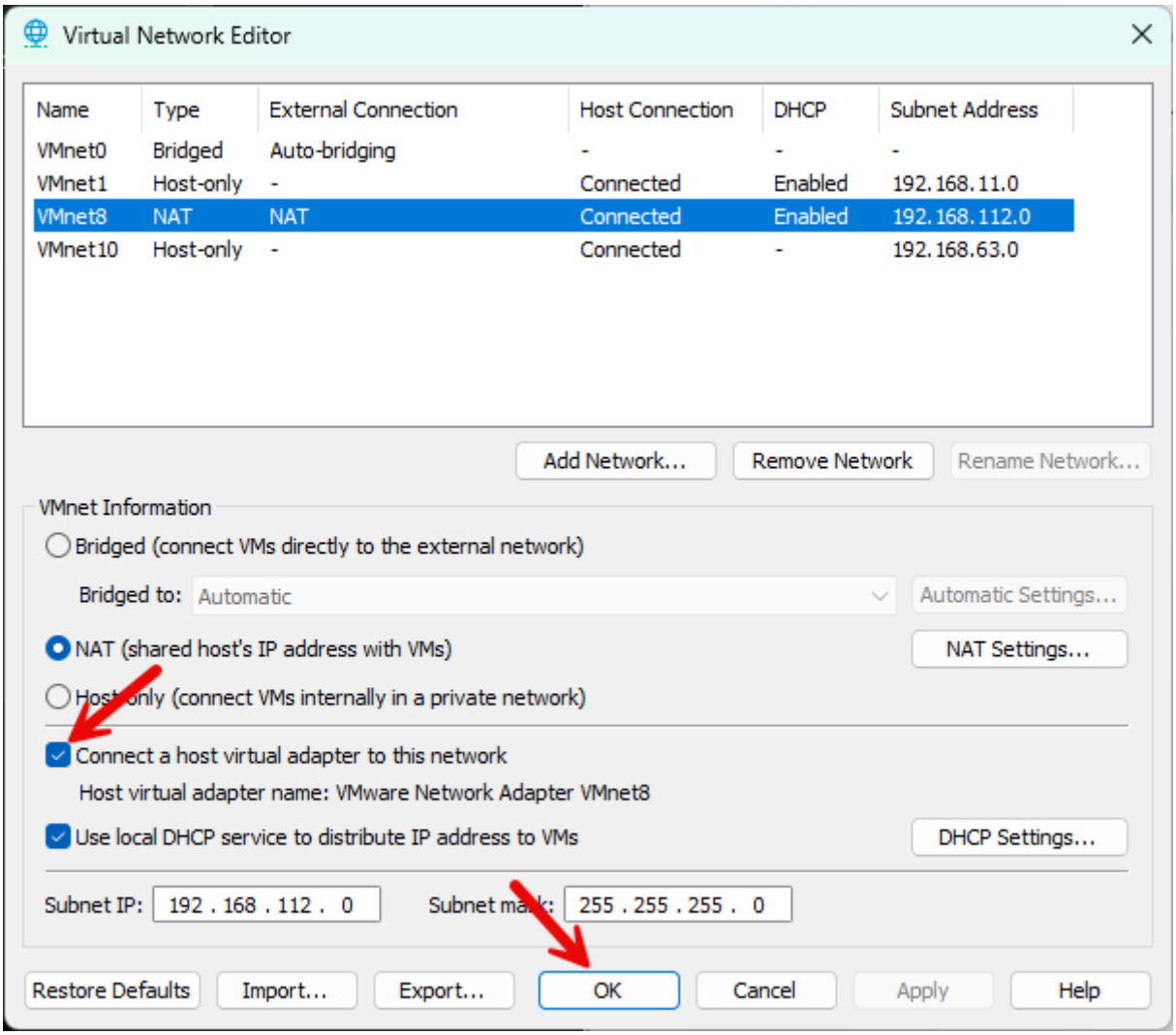
3.1.2 Oorzaak probleem

Er is een IP-conflict.

Klik daarom op Edit, Virtual Network Editor… en selecteer NAT. Klik erna rechtsonder op Change Settings en klik op Ja.

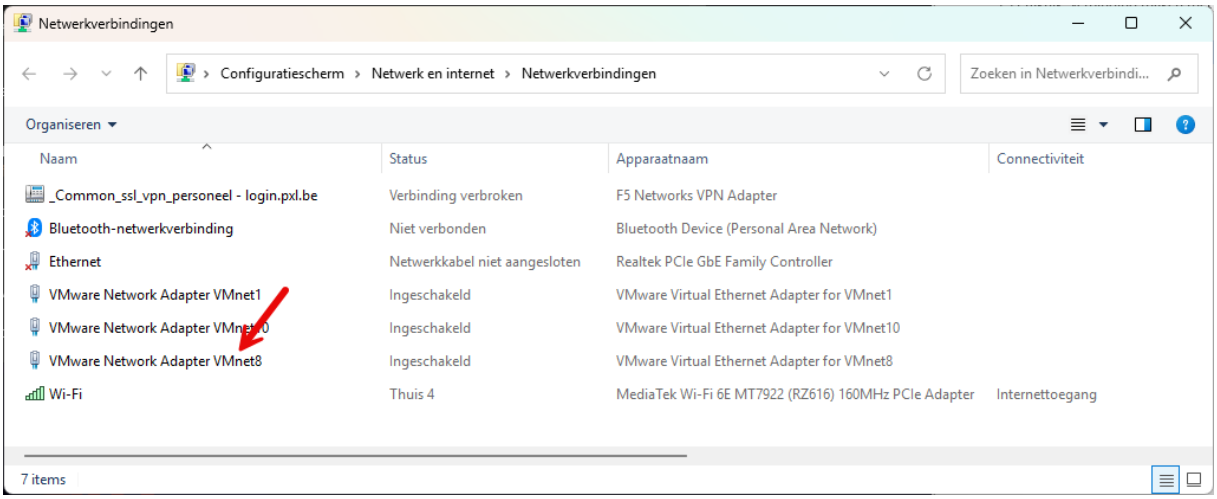


Selecteer NAT en je ziet dat er een vinkje staat voor Connect a host virtual adapter to this network (we hebben dat zo ingesteld tijdens instructies bijlage A).

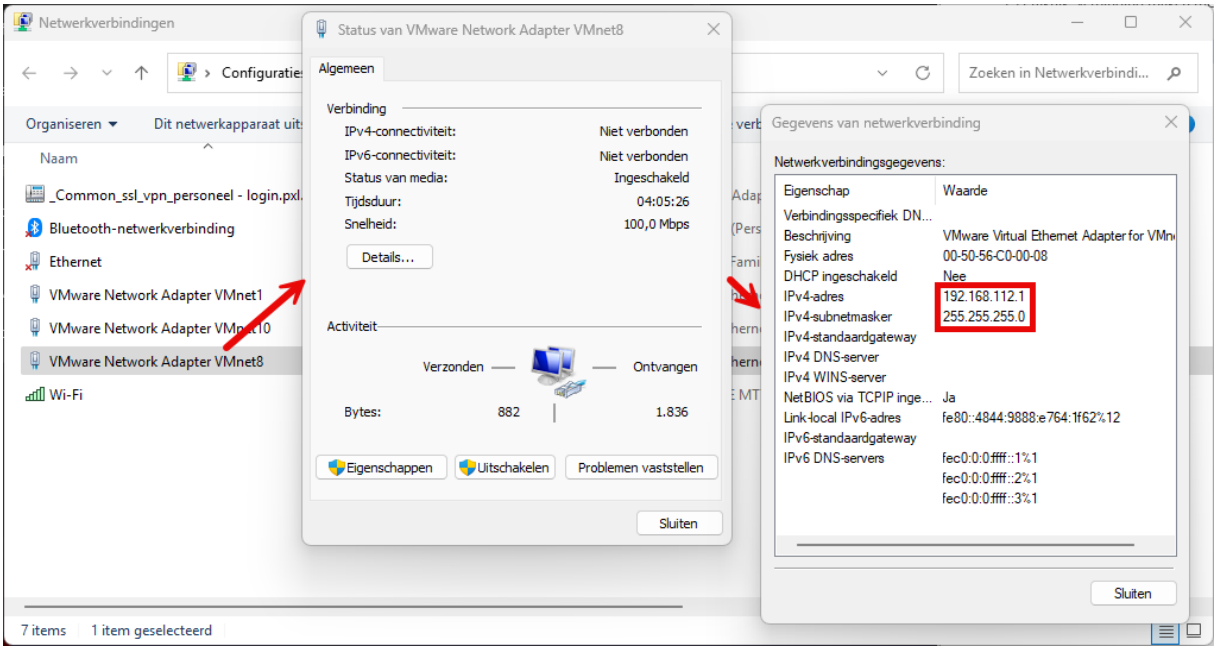


Klik erna op OK.

Ga nu naar de netwerkverbindingen op je host (Windows 11 computer normaliter).



We bekijken het IP-nummer van deze virtuele netwerkkaart zoals hieronder grafisch staat aangegeven.



Zoals je weet is dat ook het IP-nummer die ingesteld is voor de server! Hier zit dus het probleem.

3.1.3 Probleem oplossen

We vragen het IP-nummer van ens160 op.

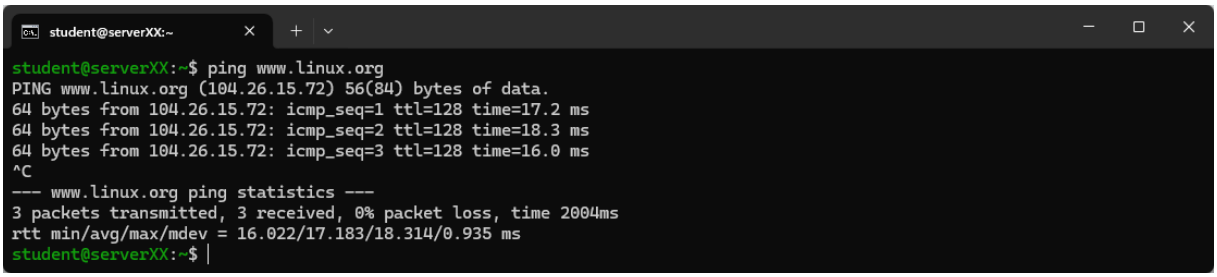
```
student@serverXX:~$ nmcli -g IP4.ADDRESS dev show ens160
```

Je ziet dat je geen antwoord krijgt… Dat is omdat het IP-adres niet toegewezen kan worden omwille van het conflict.

We zullen ens160 op de server terug het IP-adres 192.168.112.100/24 geven om het op te lossen.

```
student@serverXX:~$ sudo nmcli con mod ens160 ipv4.addresses 192.168.112.100/24


student@serverXX:~$ sudo nmcli con up ens160

Verbinding is met succes geactiveerd (actief D-Bus-pad:
/org/freedesktop/NetworkManager/ActiveConnection/6)
```

Vanaf je server kan je nu terug op het internet.



Ping nu vanaf de host naar je server…. Ook dit gaat nu.

3.2 SSH

3.2.1 Inleiding

SSH maakt het mogelijk om op afstand een SSH-sessie uit te voeren op een andere computer. Je kan via SSH opdrachten uitvoeren, bestanden kopiëren, en interactieve shells openen. Dit is reeds behandeld bij Linux Advanced maar we herhalen het hier beknopt.

3.2.2 SSH-server installeren en configureren

Dit is reeds uitvoerig besproken in Linux Advanced maar wordt hier beknopt herhaald.

De SSH-server is al geïnstalleerd als je bijlage A hebt gevolgd maar hier staat hoe je het kan doen als je niet de nodige pakketten geïnstalleerd hebt.

```
student@serverXX:~$ sudo dnf install -y openssh-server
```

Start de SSH-service op.

```
student@serverXX:~$ sudo systemctl start sshd
```

Start de SSH-service op tijdens opstarten van het systeem.

```
student@serverXX:~$ sudo systemctl enable sshd
```

Stel de firewall in zodat er een gat gemaakt wordt voor SSH (ook na opstarten van het systeem).

```
student@serverXX:~$ sudo firewall-cmd --permanent --add-service=ssh

student@serverXX:~$ sudo firewall-cmd --reload
```

3.2.3 Verbinding maken met SSH-server

We maken vanaf de Linux-client verbinding met server<jeinitialen>.

```
[student@clientXX ~]$ ssh student@192.168.112.100

The authenticity of host '192.168.112.100 (192.168.112.100)' can't be
established.

ED25519 key fingerprint is SHA256:F1750QXtB+sxrh3GrCCpXqTM6MZ+uzzmXhXFMbmRDD8.

This key is not known by any other names

Are you sure you want to continue connecting (yes/no/[fingerprint])? yes  #
eerste keer

Warning: Permanently added '192.168.112.100' (ED25519) to the list of known
hosts.

student@192.168.112.100's password: # geef wachtwoord in van student op
192.168.112.100

…

student@serverXX:~$
```

Door exit in te geven ga je terug naar de client.

```
student@serverXX:~$ exit

uitgelogd

Connection to 192.168.112.100 closed.

[student@clientXX ~]$
```

Je kan, zoals ook reeds gezien bij Linux Advanced, ook gebruik maken van een sleutelpaar.

- Maak dat aan op de client.

```
[student@clientXX ~]$ ssh-keygen

Generating public/private rsa key pair.

Enter file in which to save the key (/home/student/.ssh/id_rsa): <Enter>

Enter passphrase (empty for no passphrase): <Enter>

Enter same passphrase again: <Enter>

Your identification has been saved in /home/student/.ssh/id_rsa

Your public key has been saved in /home/student/.ssh/id_rsa.pub

The key fingerprint is:

SHA256:NkwuIpRJav9p2errYVPAdfkc72Kd3FRzh3eC8qXOFnk student@clientXX

The key's randomart image is:

+---[RSA 3072]----+

|  .    . ..  . . |

| o o. . ..... +.*|

|..+  o  . ooo+ o*|

|...   .+   o=.E. |

|  ... ..S  o+o+  |

|   ...=o . o+= . |

|     O .  ...    |

|    o +          |

|    o=.          |

+----[SHA256]-----+
```

- Kopieer erna de publieke sleutel naar de account op je server waar je zonder wachtwoord verbinding mee wil maken.

```
[student@clientXX ~]$ ssh-copy-id student@192.168.112.100

/usr/bin/ssh-copy-id: INFO: Source of key(s) to be installed: ssh-add -L

The authenticity of host '192.168.112.100 (192.168.112.100)' can't be
established.

ED25519 key fingerprint is SHA256:ZP8PS4Cs2UGaLLEyFuny/Y3C8X/qRzF8354RSj/upRs.
```

```
This key is not known by any other names.

Are you sure you want to continue connecting (yes/no/[fingerprint])? <yes>

/usr/bin/ssh-copy-id: INFO: attempting to log in with the new key(s), to filter
out any that are already installed

/usr/bin/ssh-copy-id: INFO: 1 key(s) remain to be installed -- if you are
prompted now it is to install the new keys

student@192.168.112.100's password: <Geef wachtwoord in>


Number of key(s) added: 1


Now try logging into the machine, with: "ssh 'student@192.168.112.100'"

and check to make sure that only the key(s) you wanted were added.
```

Je kan nu verbinding maken zonder wachtwoord.

```
[student@clientXX ~]$ ssh student@192.168.112.100

…

student@serverXX:~$
```

3.3 Remote Docker

3.3.1 Remote docker via opstellen Docker-daemon

In dit onderdeel bespreken we twee methoden om de Docker-daemon extern toegankelijk te maken. De eerste methode is relatief eenvoudig, maar brengt belangrijke beveiligingsrisico's met zich mee: het openstellen van de Docker-daemon via een TCP-poort. Hierdoor kunnen externe clients verbinding maken en opdrachten naar de Docker-daemon sturen.

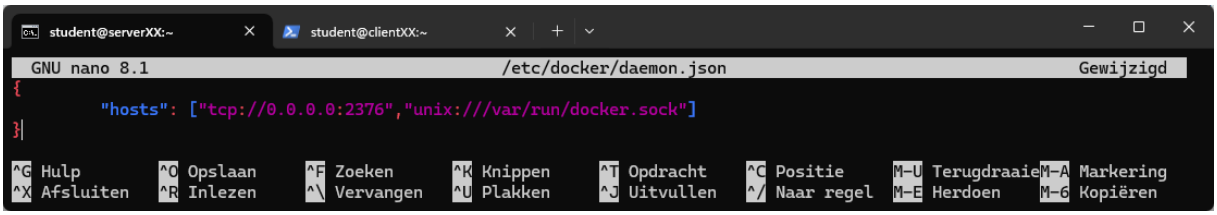
Het grote nadeel van deze methode is dat iedereen met toegang tot deze poort potentieel commando's kan uitvoeren, zelfs zonder beheerdersrechten. Dit kan leiden tot ernstige beveiligingsproblemen.

Een nuttige bron op dit gebied is de website "Protect the Docker daemon socket".

> Hoewel het mogelijk is om de verbinding te beveiligen met TLS-certificaten, behandelen we dat hier niet.
> Deze configuratie maakt gebruik van poort 2375 zonder TLS, en is niet veilig voor productieomgevingen en gebruiken we hier uitsluitend voor demonstratie- of testdoeleinden. In productie wordt uiteraard altijd aangeraden om TLS (poort 2376) te gebruiken.

We stellen eerst in dat Docker ook luistert op TCP-poort 2375 naar aanvragen zonder TLS.

```
student@serverXX:~$ sudo nano /etc/docker/daemon.json

{
        "hosts": ["tcp://0.0.0.0:2376", "unix:///var/run/docker.sock"]
}
```



Zoals je ziet wordt er geluisterd naar:

- tcp://0.0.0.0:2376
  Hier stel je in dat er geluisterd wordt op poort 2376.
- unix:///var/run/docker.sock
  Hiermee stel je in dat Docker CLI (client) verbinding maakt met de Docker-daemon (server)

Aangezien je nu in een apart bestand hebt ingesteld waarnaar geluisterd wordt dien je dit te verwijderen uit de systemd service unit file.

```
student@serverXX:~$ sudo nano /usr/lib/systemd/system/docker.service

…

ExecStart=/usr/bin/dockerd -H fd:// --
containerd=/run/containerd/containerd.sock

…
```

Pas deze regel aan naar onderstaande.

```
ExecStart=/usr/bin/dockerd --containerd=/run/containerd/containerd.sock
```

-H fd:// betekent eigenlijk: “gebruik de socket die systemd al voor mij heeft klaargezet”.

Docker hoeft dan zelf geen nieuwe socket aan te maken, maar neemt de file descriptor (de verwijzing naar dat socket-bestand) over van systemd.

Herlaad systemd en herstart Docker.

```
student@serverXX:~$ sudo systemctl daemon-reload

student@serverXX:~$ sudo systemctl restart docker
```

Erna kan je uiteraard de status controleren.

```
student@serverXX:~$ sudo systemctl status docker
```

Stel nu de firewall in.

```
student@serverXX:~$ sudo firewall-cmd --permanent --add-port=2376/tcp

student@serverXX:~$ sudo firewall-cmd --reload
```

Nu kan Docker TCP-verkeer ontvangen op poort 2376.

We kunnen nu op afstand (clientXX) verbinding maken met de Docker-host en commando's uitvoeren. Op je client moet uiteraard wel de docker-client geïnstalleerd zijn.

We voegen hiervoor eerst de repository van docker zelf toe.

```
[student@clientXX ~]$ sudo dnf config-manager --add-repo
https://download.docker.com/linux/centos/docker-ce.repo
```

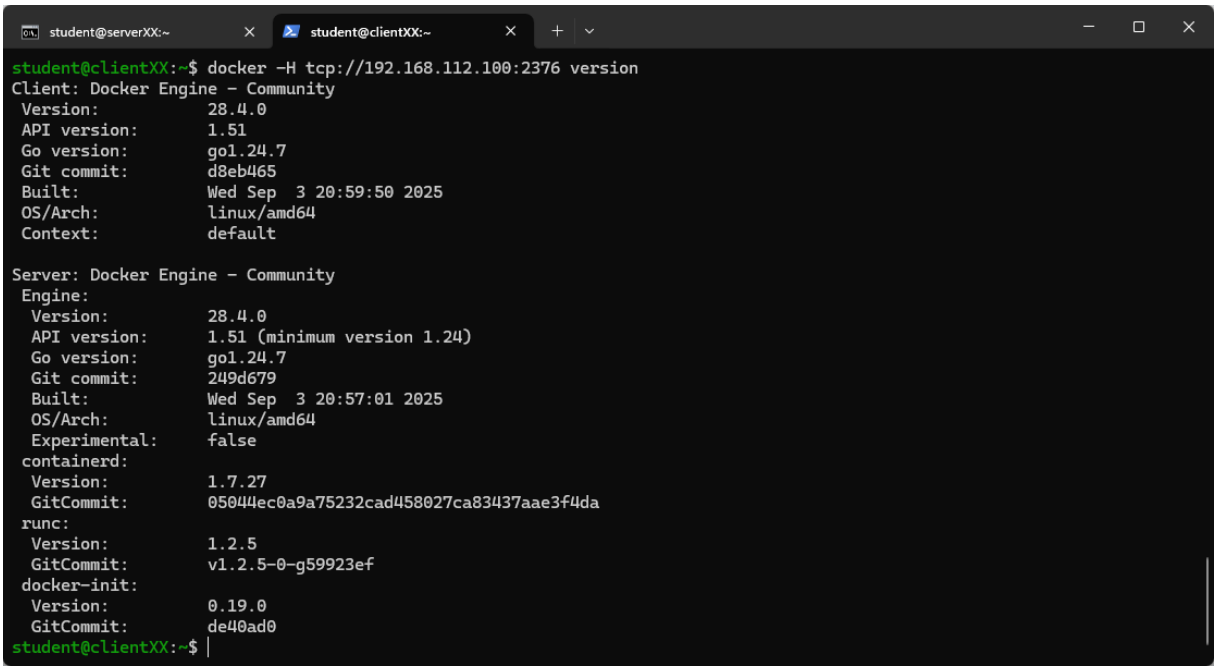
We installeren nu enkel de Docker CLI zodat we verbinding kunnen maken met de server.

```
[student@clientXX ~]$ sudo dnf install docker-ce-cli
```

De client is nu beschikbaar, maar niet de server van docker.

We maken nu verbinding met Docker op je server vanaf de client.

```
[student@clientXX ~]$ docker -H tcp://192.168.112.100:2376 version
```



Hoewel de methode functioneert is deze methode, zoals hier ingesteld, erg onveilig zoals reeds besproken.

- Geen SSH

- Iedereen kan verbinding maken met de poort vanaf een andere computer.

3.3.2 Remote Podman/Docker via SSH

Na het inloggen via SSH vanaf de client kun je Docker en Podman-commando's op de server uitvoeren.

Je kan ook onmiddellijk een commando meegeven met podman.

```
student@clientXX ~]$ ssh student@192.168.112.100 "podman version"

student@192.168.112.100's password:

Client:        Podman Engine

Version:       5.4.0

API Version:   5.4.0

Go Version:    go1.23.10 (Red Hat 1.23.10-1.el10_0)

Built:         Wed Jun 25 02:00:00 2025

Build Origin:  Red Hat, Inc. <http://bugzilla.redhat.com/bugzilla>

OS/Arch:       linux/amd64
```

Met docker krijg je een probleem wanneer je "docker version" meegeeft.

```
student@clientXX ~]$ ssh student@192.168.112.100 "docker version"

Client: Docker Engine - Community

 Version:           28.4.0

 API version:       1.51

 Go version:        go1.24.7

 Git commit:        d8eb465

 Built:             Wed Sep  3 20:59:50 2025

 OS/Arch:           linux/amd64

 Context:           default

permission denied while trying to connect to the Docker daemon socket at
unix:///var/run/docker.sock: Get
"http://%2Fvar%2Frun%2Fdocker.sock/v1.51/version": dial unix
/var/run/docker.sock: connect: permission denied
```

Zoals je ziet krijg je een melding dat er geen toegang is (permission denied). Dit kan je oplossen door sudo-rechten te gebruiken. Zoals je weet heeft docker immers sudo-rechten nodig.

We trachten het als volgt op te lossen.

```
[student@clientXX ~]$ ssh student@192.168.112.100 "sudo docker version"

student@192.168.112.100's password:

sudo: a terminal is required to read the password; either use the -S option to
read from standard input or configure an askpass helper

sudo: een wachtwoord is verplicht
```

Sudo heeft op de remote host geen interactieve terminal om het wachtwoord te vragen. Daarom krijg je:

```
sudo: een wachtwoord is verplicht
```

Je kan dat o.a. oplossen door een interactieve terminal bij SSH te forceren.

```
[student@clientXX ~]$ ssh -t student@192.168.112.100 "sudo docker version"
```