8 Podman Compose in combinatie met Containerfile

8.1 Inleiding

Eén of meerdere container files en compose.yml kunnen samenwerken om het bouwen, configureren en beheren van containers te vereenvoudigen.

- Een Containerfile is een script met instructies voor het bouwen van een Podman-image. Het bevat alle stappen die nodig zijn om een omgeving te creëren zoals al behandeld:
    - De basisimage (FROM).
    - Het kopiëren van bestanden (COPY).
    - Installatie (RUN).
    - Het definiëren van een startcommando (CMD/ENTRYPOINT).
    - Status checken (HEALTHCHECK)
- Met compose.yml kan je meerdere containers definiëren aan de hand van images. Dit bestand maakt het mogelijk om:
    - Verschillende services te definiëren (zoals databases en applicaties).
    - Netwerken en volumes aan te maken.
    - Poorten door te sturen en omgevingsvariabelen in te stellen.

8.2 Voorbeeld 1

We bouwen een eenvoudige container die een bash-script uitvoert.

Maak een map vb1.

```
student@serverXX:~$ mkdir vb1
```

Hierin zullen we Dockerfile, script.sh en compose.yml plaatst.

8.2.1 Containerfile

In dit voorbeeld vertrekken we van een registry.access.redhat.com/ubi10/ubi-image en voegen we een simpel bash-script toe dat in de container wordt uitgevoerd.

```
student@serverXX:~$ nano vb1/Dockerfile

FROM registry.access.redhat.com/ubi10/ubi

COPY script.sh /scripts/script.sh

RUN chmod +x /scripts/script.sh

WORKDIR /scripts

CMD ["/bin/bash", "./script.sh"]
```

Dit Containerfile-bestand doet het volgende:

- Het gebruikt registry.access.redhat.com/ubi10/ubi -image als basis.
- Kopieert een bash-script (script.sh) naar de container.
- Geeft het script uitvoerrechten.
- Stelt de werkdirectory in op /scripts
- Voert het bash-script uit bij het starten van de container.

8.2.2 Bash script (script.sh)

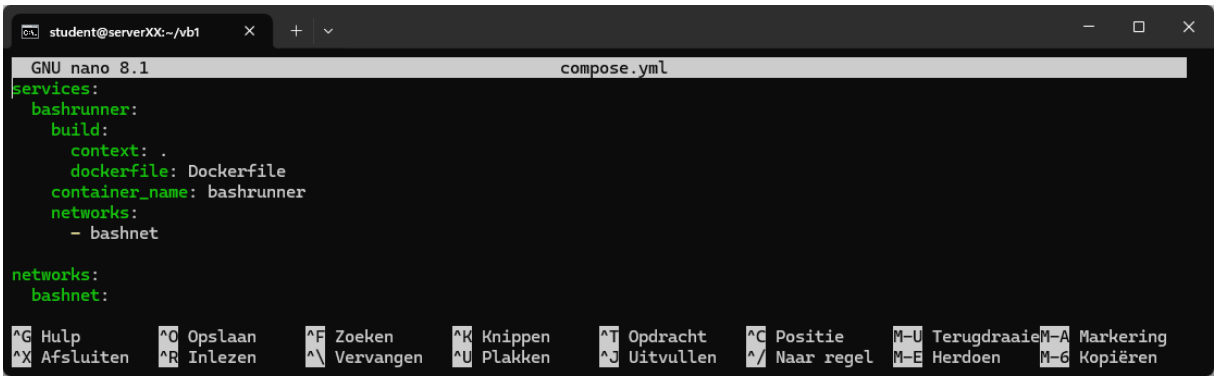
Hier is een voorbeeld van het bash-script dat in de container wordt uitgevoerd. Dit script print gewoon een simpele bericht naar de console.

```
student@serverXX:~$ nano vb1/script.sh

#!/bin/bash

echo "Hallo, dit is een bash-script dat draait in een RHEL-container!"
```

8.2.3 compose.yml

Dit is het Compose-bestand dat een container aanmaakt op basis van de dockerfile.

```
student@serverXX:~$ nano vb1/compose.yml

services:

  bashrunner:

    build:

      context: .

      dockerfile: Dockerfile

    container_name: bashrunner

    networks:

      - bashnet


networks:

  bashnet:
```



Dit Compose-bestand doet het volgende:

- Definieert een service genaamd bashrunner.
- Gebruikt de Dockerfile om de container te bouwen.
- Plaatst de container op een netwerk genaamd bashnet.

8.2.4 Uitvoer

Voer het volgende uit in de command line vanuit de directory waar je docker-compose.yml staat:

```
student@serverXX:~$ cd vb1

student@serverXX:~/vb1$ podman-compose up –build
…
COMMIT vb1_bashrunner

…

[bashrunner] | Hallo, dit is een bash-script dat draait in een RHEL-container!
```

Aangezien we geen imagenaam gespecifieerd hebben krijgt deze een naam als volgt: <projectnaam>_<servicenaam>. De projectnaam bij het gebruik van podman-compose wordt standaard afgeleid van de naam van de directory waarin het compose-bestand zich bevindt. Vandaar de naam vb1_bashrunner voor de image.

Je kan erna de container ook starten zonder eerst de image te builden omdat deze nu reeds bestaat.

```
student@serverXX:~/vb1$ podman-compose up

[bashrunner] | Hallo, dit is een bash-script dat draait in een RHEL-container!
```

8.3 Voorbeeld 2

Hier bouwen zelf een eenvoudige webserver op basis van UBI10 met Apache.

Maak een map vb2 aan met daarin:

- Bestand Containerfile
- Map mywebsite/ met daarin bestand index.html

- Bestand compose.yml

We maken eerst de map vb2 aan.

```
student@serverXX:~$ mkdir –p vb2/mywebsite
```

8.3.1 Containerfile

```
student@serverXX:~$ nano vb2/Containerfile

FROM registry.access.redhat.com/ubi10/httpd-24

COPY mywebsite/ /var/www/html/

EXPOSE 8080
```

In dit Containerfile-bestand:

- We gebruiken de httpd-server gebaseerd op ubi10.
- We plaatsen de inhoud van de subdirectory mywebsite in /var/www/html/
- We documenteren poort 8080 voor verkeer (⇔ publiceren!). Dit is niet verplicht.

De container moet niet draaiende gehouden worden aangezien registry.access.redhat.com/ubi10/httpd-24 daar zelf voor zorgt.

8.3.2 Index.html

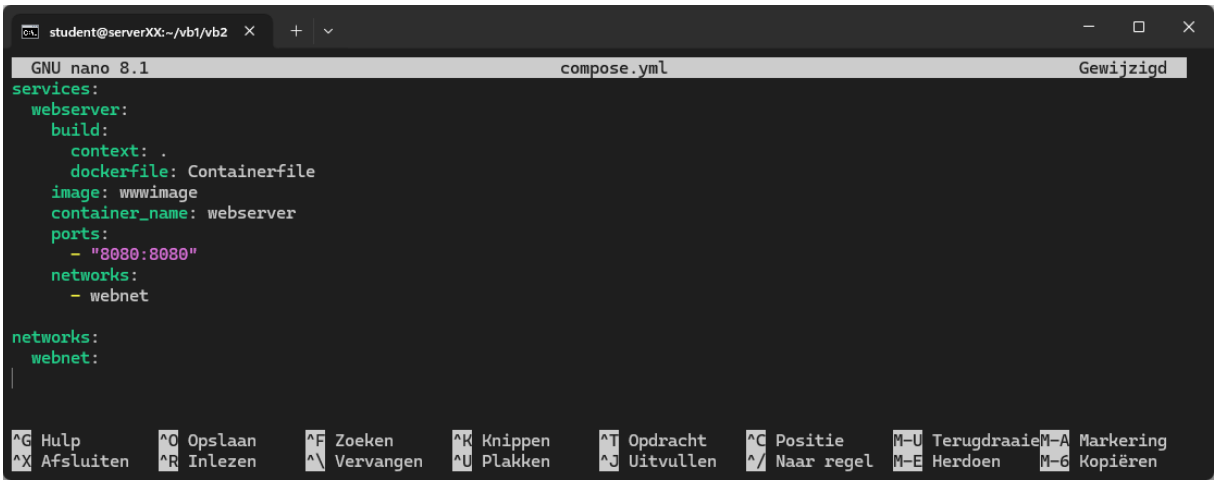
We plaatsen dit bestand in mywebsite/index.html.

```
student@serverXX:~$ nano vb2/mywebsite/index.html

<head>

    <title>Voorbeeld 2 website</title>

</head>

    <body>

        <h1>Welkom bij Voorbeeld 2 website!</h1>

    </body>

</html>
```

8.3.3 Compose.yml

We geven dit bestand onderstaande inhoud.

```
student@serverXX:~$ nano vb2/compose.yml
services:
  webserver:
    build:
      context: .
      dockerfile: Containerfile
    image: wwwimage
    container_name: webserver
    ports:
      - "8080:8080"
    networks:
      - webnet


networks:
  webnet:
```



In dit Compose-bestand:

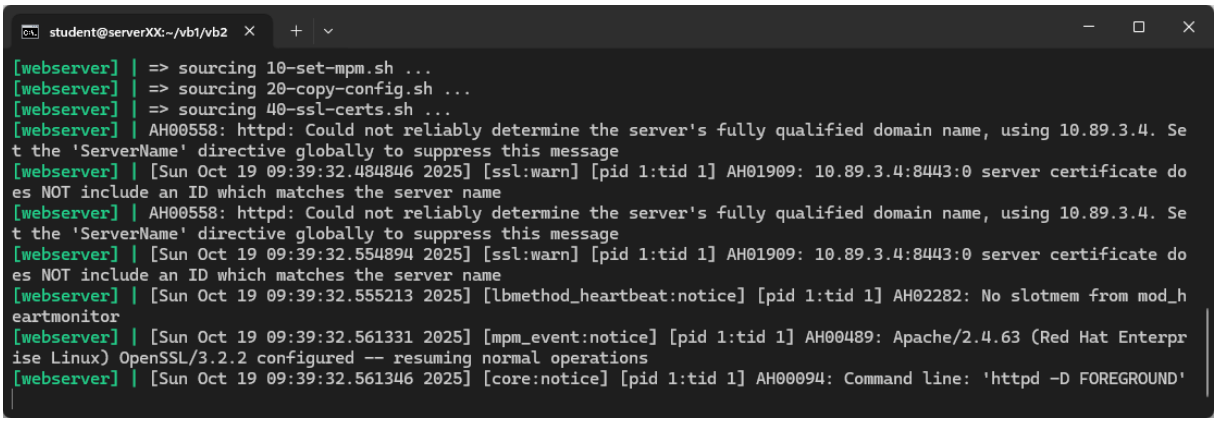
- definiëren we een service webserver die de Containerfile gebruikt om de container te bouwen.
- Hier is gekozen om zelf een naam toe te wijzen voor de image die wordt aangemaakt: wwwimage.
- De poort 80 van de container wordt gekoppeld aan poort 8080 op de hostmachine, zodat je de website kunt bezoeken via o.a. http://localhost:8080.
- De container draait op een webnet-netwerk.

8.3.4 Uitvoer

Voer in de map vb2 het volgende commando uit om de container te bouwen en te draaien:

```
student@serverXX:~$ cd vb2

student@serverXX:~/vb2$ podman-compose up --build
```

Zoals je ziet blijft de webserver draaien.



Ga naar een ander terminalvenster en voer curl localhost:8080 of curl 192.168.112.100:8080 uit.

```
student@serverXX:~$ curl localhost:8080
<head>
    <title>Voorbeeld 2 website</title>
</head>
    <body>
        <h1>Welkom bij Voorbeeld 2 website!</h1>
    </body>
</html>
```

Je ziet dat de webpagina beschikbaar is.