DATA2410-1 22V Datanettverk og Skytjenester

GROUP Portfolio Assignment 2 - Docker and Zabbix Real Use-Case

Ulrik Bakken Singsaas | s351917
Adrian Bakstad | s341507
Thea Emilie Haugen | s351879
Andrea Bjørge | s344175

April 25, 2022

1. VM1: Docker containers setup 10%

Originally, we attempted to use Docker containers on the intel1-server to implement our solution. The server however, ran out of storage, forcing us to create virtual machines through VirtualBox as a substitute.

The steps we took to set up the Docker containers can be seen below, but our final solution, utilizing virtual machines, will be shown further down. Even though the environments on intel1 will be referred to as VMs, they are in fact docker containers. All references to VMs before the switch from intel1 to VirtualBox will therefore refer to the docker containers named VM1, VM2 and VM3.

VM Setup

The three preinstalled VMs had a different ubuntu version than what was recommended in the assignment description. Therefore, to ensure that we were in line with the assignment description, we started by deleting them. We then built new images for the VMs, using Dockerfiles, to run the containers with their new custom configurations. The new images automatically start various services on the different containers to circumvent the problem of not having systemd in docker containers. Starting the services in the image ensures that they run on container startup, and minimizes the amount of commands we have to run manually for each container. After the images were ready, we made new containers, now with the focal fossa version, whilst also allowing for docker containers within docker containers.

The following code is what we used for building the images and running the containers, as explained above.

```
# building the image for VM1 from Dockerfile
sudo docker build -t vm-image1 -f data2410-portfolio2/configs/vm1-dockerfile .

# building the image for VM2 from Dockerfile
sudo docker build -t vm-image2 -f data2410-portfolio2/configs/vm2-dockerfile .

# building the image for VM3 from Dockerfile
sudo docker build -t vm-image3 -f data2410-portfolio2/configs/vm3-dockerfile .
```

```
# Running the VM1 container
sudo docker container run --privileged -v /var/run/docker.sock:/var/run/docker.sock -d vm_image
# Running the VM2 container

sudo docker container run --name vm2 -p 8080:8080 -d -t vm-image2
# Running the VM3 container
# Coming soon
```

The image, `vm_image`, was made from this Dockerfile:

```
TODO
- [ ] get from configs/Dockerfile
```

Quad Container Setup

After setting up the three VMs, we used the file `docker-compose.yml`, to set up the four containers with the required config instructions for the assignment. This file can be found in the configs folder.

```
TODO
- [ ] Paste finalized version of this file
```

The assignment does not specify volumes. Therefore, in order to keep the maintenance simple, we used four files from `this directory` as the volumes for each of the four containers. At first, we ran the docker containers without the volume statements to auto generate the configs, then we edited the configs and pasted them back in to a new compose file to automate our statements. The final files we used can be found below *(alt. The changes we made to the files can be found below).*

The following scripts copies configs to containers, so we can use the volumes.

```
g13@net513:~/data2410-portfolio2$ sudo docker cp vm_data/zabbix/zabbix_server.conf c99a6922714d:/etc/zabbi
g13@net513:~/data2410-portfolio2$ sudo docker cp vm_data/zabbix-agent/zabbix_agentd.conf c99a6922714d:/etc
g13@net513:~/data2410-portfolio2$ sudo docker cp vm_data/zabbix-web/ c99a6922714d:/etc/zabbix/
```

mysql

zabbix-server

zabbix-web

zabbix-agent

2. VM2 and VM3: Install zabbix-agent and zabbix-proxy 10%

VM2

We followed this guide to complete task 1 in part III of the assignment description: https://bestmonitoringtools.com/install-zabbix-proxy-on-ubuntu/ There were a few things we did differently from the guide. These will be described below.

Innstalling Zabbix Proxy

We started by installing `Zabbix Proxy` on VM2 with the following commands:

```
apt-get install wget

wget https://repo.zabbix.com/zabbix/6.0/ubuntu/pool/main/z/zabbix-release/zabbix-release_6.0-1%2Bubuntu20.

#needed this as well as the proxy was the wrong version
wget https://repo.zabbix.com/zabbix/6.0/ubuntu/pool/main/z/zabbix/zabbix-proxy-mysql_6.0.1-1%2Bubuntu20.04

dpkg -i zabbix-release_6.1-1+ubuntu20.04_all.deb

apt-get install -f

apt-get install zabbix-proxy-mysql

apt-get install zabbix-sql-scripts
```

The only notable difference between the guide and what we did in this part of the task, is that we used Zabbix-proxy version 6.1 instead of Zabbix-proxy version 6.0, because the assignment descriptions asks us to use version 6.1.

Configuring database

After we finished installing `Zabbix Proxy`, we installed and configured the database, using `MariaDB`, according to both the guide mentioned above, and the assignment description.

The following block of code describes the installation of `MariaDB` on VM2:

```
sudo apt install software-properties-common -y

curl -LsS -O https://downloads.mariadb.com/MariaDB/mariadb_repo_setup

sudo bash mariadb_repo_setup --mariadb-server-version=10.6

sudo apt update

sudo apt -y install mariadb-common mariadb-server-10.6 mariadb-client-10.6
```

After installing `MariaDB` we began the configuration of the database by running the following commands to start and enable MariaDB, and configure it to start on boot:

```
sudo systemctl start mariadb

sudo systemctl enable mariadb
```

The next step in the configuration of the database was to reset the password. We did that with the commands in the following code block: The following code blocks set the password for new **root password** = 123.

```
sudo mysql_secure_installation

Enter current password for root (enter for none): Press Enter

Switch to unix_socket authentication [Y/n] y

Change the root password? [Y/n] y

New password: <Enter root DB password>

Re-enter new password: <Repeat root DB password>

Remove anonymous users? [Y/n] y

Disallow root login remotely? [Y/n] y

Remove test database and access to it? [Y/n] y

Reload privilege tables now? [Y/n] y
```

The only notable difference between the guide and what we did was that we set the root password to ‘123’, in stead of ’rootDBpass.

After we set the root password, it was time to create the database by running the commands in the following block of code:

```
sudo  mysql -uroot -p'123' -e "create database zabbix_proxy character set utf8mb4 collate utf8mb4_bin;"
sudo mysql -uroot -p'123' -e "grant all privileges on *.* to zabbix@localhost identified by 'zabbixDBpass'
```

The last step in the configuration of the database was to import the initial schema and data with the following command:

```
sudo cat /usr/share/doc/zabbix-sql-scripts/mysql/proxy.sql | mysql -uzabbix -p'zabbixDBpass' zabbix_proxy
```

In the installation and configuration of the database, we followed the guide quite exactly. Therefore, there are very few differences between what we did to install and configure the database, and what is stated in the installation guide.

Configurating Zabbix Proxy

Once the installation and configuration of the database was complete, it was time to configure the `Zabbix Proxy`. The first step when configuring the `Zabbix Proxy` was to open the config file with the following command:

```
sudo gedit /etc/zabbix/zabbix_proxy.conf
```

In the file we changed the following values:

```
DBPassword=zabbixDBpass
ConfigFrequency=100
Server=192.168.50.95
Hostname=Zabbix Proxy
DBName=zabbix_proxy
DBUser=zabbix
```

After editing the necessary values, we saved and exited the file. We configured the ConfigFrequency to be 100 seconds. This parameter determines how often the proxy retrieves data from the configuration file, and is useful to cut down on the waiting time between updates on the status of the `Zabbix Proxy`. The notable differences between our config file, and the config file in the guide is that we have a different IP address for the server, and a different hostname for the proxy itself.

Starting and enabling Zabbix Proxy

Next, we started and enabled the `Zabbix Proxy` to boot on startup with the following commands:

```
sudo systemctl restart zabbix-proxy

sudo systemctl enable zabbix-proxy
```

Find Hostname in the zabbix-proxy config file and note. This is important in order to connect the proxy to the server in the web frontend. In the zabbix-proxy config find DBPassword, and make sure it is set to the correct password.

Registering Zabbix Proxy in the Zabbix frontend

Accessing zabbix web with lynx

The following code block describes how to access the `Zabbix-web`.

```
# This gives you a cli web browser
g13@net513:~$ lynx localhost
```

Accessing zabbix web with lynx

The following code block describes how to access the `Zabbix-web`.

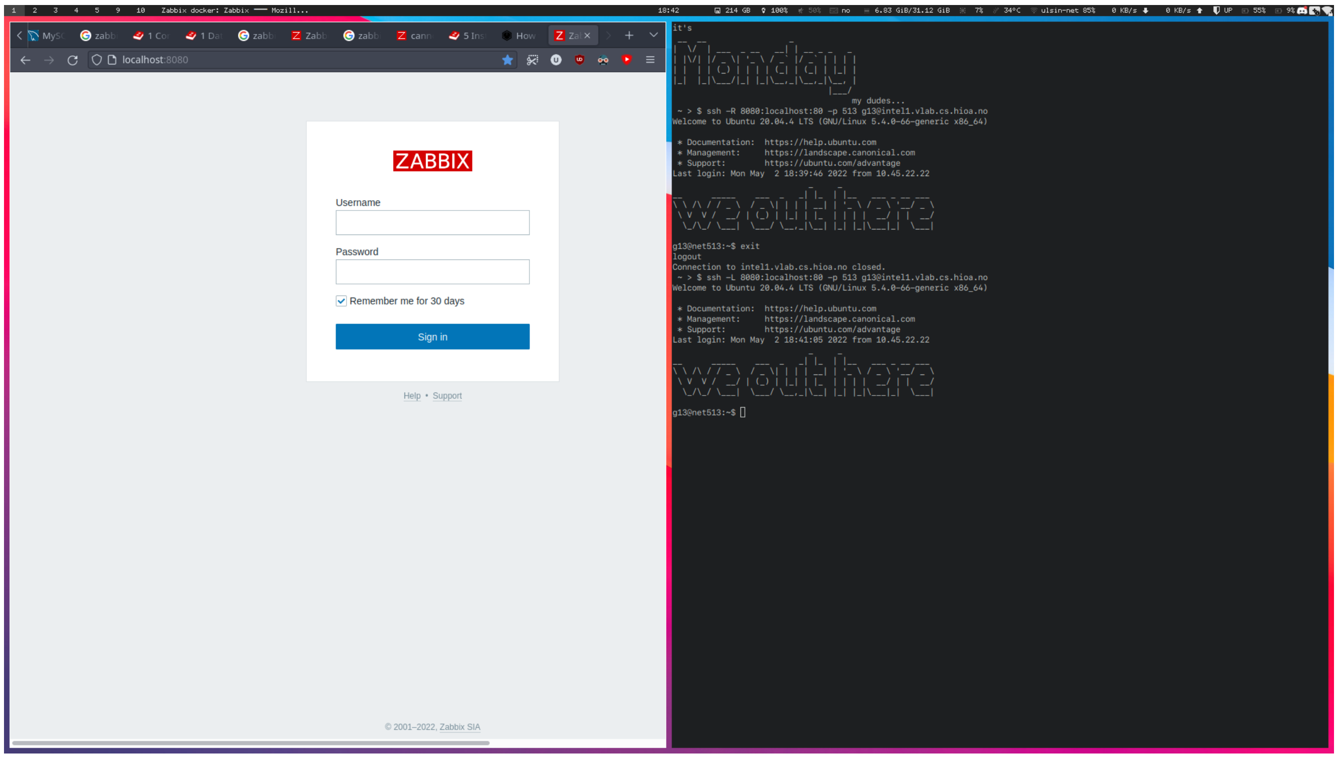
```
# This gives you a cli web browser
g13@net513:~$ lynx localhost
```

Acessing zabbix web with ssh tunneling, setting up the new host

We needed to set up a new host on the frontend to make the active checks work, used a ssh tunnel to let us open the intel1 localhost:80 in our own browsers

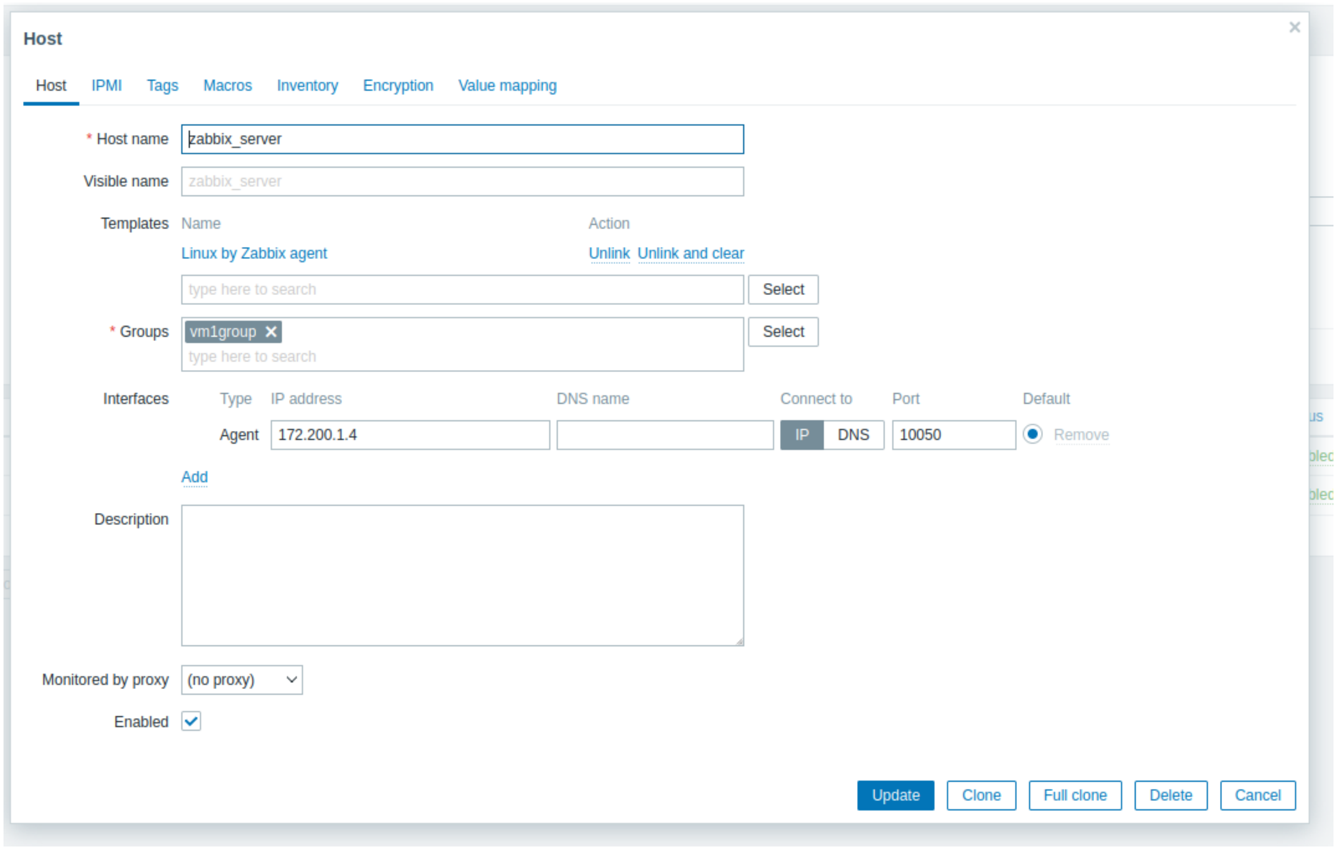
```
ssh -L 8080:localhost:80 -p 513 g13@intel1.vlab.cs.hioa.no
```

then we could use the localhost:8080 in our browser to login with `user: Admin password: admin`



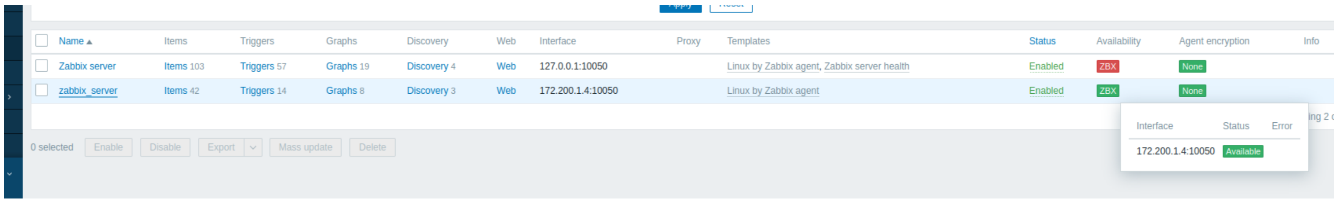
```
it's
 __  __                 _
|  \/  | ___  _ __   __| | __ _ _   _
| |\/| |/ _ \| '_ \ / _` |/ _` | | | |
| |  | | (_) | | | | (_| | (_| | |_| |
|_|  |_|\___/|_| |_|\__,_|\__,_|\__, |
                                |___/
                                     my dudes...
 ~ > $ ssh -R 8080:localhost:80 -p 513 g13@intel1.vlab.cs.hioa.no
Welcome to Ubuntu 20.04.4 LTS (GNU/Linux 5.4.0-66-generic x86_64)

 * Documentation:  https://help.ubuntu.com
 * Management:     https://landscape.canonical.com
 * Support:        https://ubuntu.com/advantage
Last login: Mon May  2 18:39:46 2022 from 10.45.22.22
                           _     _
__      _____    ___  _   _| |_  | |__   ___ _ __ ___
\ \ /\ / / _ \  / _ \| | | | __| | '_ \ / _ \ '__/ _ \
 \ V  V /  __/ | (_) | |_| | |_  | | | |  __/ | |  __/
  \_/\_/ \___|  \___/ \__,_|\__| |_| |_|\___|_|  \___|

g13@net513:~$ exit
logout
Connection to intel1.vlab.cs.hioa.no closed.
 ~ > $ ssh -L 8080:localhost:80 -p 513 g13@intel1.vlab.cs.hioa.no
Welcome to Ubuntu 20.04.4 LTS (GNU/Linux 5.4.0-66-generic x86_64)

 * Documentation:  https://help.ubuntu.com
 * Management:     https://landscape.canonical.com
 * Support:        https://ubuntu.com/advantage
Last login: Mon May  2 18:41:05 2022 from 10.45.22.22
                           _     _
__      _____    ___  _   _| |_  | |__   ___ _ __ ___
\ \ /\ / / _ \  / _ \| | | | __| | '_ \ / _ \ '__/ _ \
 \ V  V /  __/ | (_) | |_| | |_  | | | |  __/ | |  __/
  \_/\_/ \___|  \___/ \__,_|\__| |_| |_|\___|_|  \___|

g13@net513:~$
```

Then we set up a new host to make the active changes work, configured like this:



* Host name	zabbix_server
Visible name	zabbix_server
Templates	Name: Linux by Zabbix agent — Action: Unlink, Unlink and clear
* Groups	vm1group
Interfaces	Type: Agent — IP address: 172.200.1.4 — DNS name: — Connect to: IP — Port: 10050 — Default: ● — Remove
Description	
Monitored by proxy	(no proxy)
Enabled	✓

Proof that it is working:



Name ▲	Items	Triggers	Graphs	Discovery	Web	Interface	Proxy	Templates	Status	Availability	Agent encryption	Info
Zabbix server	Items 103	Triggers 57	Graphs 19	Discovery 4	Web	127.0.0.1:10050		Linux by Zabbix agent, Zabbix server health	Enabled	ZBX	None	
zabbix_server	Items 42	Triggers 14	Graphs 8	Discovery 3	Web	172.200.1.4:10050		Linux by Zabbix agent	Enabled	ZBX	None	

Interface	Status	Error
172.200.1.4:10050	Available	

VM 3

The following code block must be run as root on VM3 to install the `Zabbix-agent`.

```
# TODO delete these, they have been put into vm3 dockerfile
 wget https://repo.zabbix.com/zabbix/6.1/ubuntu/pool/main/z/zabbix-release/zabbix-release_6.1-1%2Bubuntu20

 dpkg -i zabbix-release_6.1-1+ubuntu20.04_all.deb

 apt-get install -f

 apt-get install zabbix-agent
```

The following code block creates the psk encryption key.

```
ubuntu1@ubuntu1-VirtualBox:~$ openssl rand -hex 32 > zabbix_agent.psk
ubuntu1@ubuntu1-VirtualBox:~$ cat zabbix_agent.psk
e8126679667a8594bc8d3d76121b6ba2a5fb4b6d41bea2cd62190c163fbc6c6b
```

The following code block moves the psk encryption key to /opt/zabbix folder.

```
sudo mkdir /opt/zabbix
sudo chmod 777 /opt/zabbix
sudo mv zabbix_agent.psk /opt/zabbix/
```

used these commands to edit the `zabbix-agent.conf` file

```
ubuntu1@ubuntu1-VirtualBox:~$ sudo su
root@ubuntu1-VirtualBox:/home/ubuntu1# vim /etc/zabbix/zabbix_
zabbix_agentd.conf  zabbix_proxy.conf
root@ubuntu1-VirtualBox:/home/ubuntu1# vim /etc/zabbix/zabbix_
zabbix_agentd.conf  zabbix_proxy.conf
root@ubuntu1-VirtualBox:/home/ubuntu1# vim /etc/zabbix/zabbix_
zabbix_agentd.conf  zabbix_proxy.conf
root@ubuntu1-VirtualBox:/home/ubuntu1# vim /etc/zabbix/zabbix_agentd.conf
```

The following code block contains the lines we changed in the `zabbix-agent.conf` file.

```
TLSConnect=psk
TLSAccept=psk
TLSPSKIdentity=cbt_psk_01,
TLSPSKFile=/opt/zabbix_agent.psk
# the local ip of our bridged networked VM2
Server=192.168.50.247
```

Since we are running this in a docker container and not on an actual VM, we don´t have systemd available. Therefore, we cannot *enable* the service, only start it, and have it as a run command in a dockerfile. This means we need to make sure that the service is started every time we run the container the dockerfile is made for.

3. VM2: Nginx proxy 10%

We start by creating the configuration file for the nginx-proxy. This file makes sure that the nginx-proxy listens on port 8080, and redirects all incoming traffic from that port to the `Zabbix-server` using `proxy_pass`followed by the Zabbix-server's IP address and port number.

```
# To be moved to /etc/nginx/sites-enabled/zabbix.conf on VM2

server {
    listen  8080;
    server_name localhost;

    location / {
        proxy_pass http://172.24.0.1:8080;
        proxy_set_header Host $http_host;
        proxy_set_header X-Real-IP $remote_addr;
        proxy_set_header X-Forwarded-For $proxy_add_x_forwarded_for;
        proxy_set_header X-Forwarded-Proto $scheme;
    }
}
```

After setting up the configuration file for the `Nginx-proxy`, we start up a terminal on VM2 and install nginx with the following commands.

```
sudo docker exec -it vm2 /bin/bash
root@47b33e945b34:/# apt-get update
root@47b33e945b34:/# apt-get install nginx
```

Once nginx has been installed, we disable the default virtual host by unlinking it with the following command.

```
root@47b33e945b34:/# unlink /etc/nginx/sites-enabled/default
```

Then we add `reverse-proxy.conf` to the directory `/etc/nginx/sites-available/` with the following commands.

```
root@47b33e945b34:/# cd /etc/nginx/sites-available/
root@47b33e945b34:/# cd nano reverse-proxy.conf
```

To complete the proxy, we activate the directives by linking to `/sites-enabled/` using the following command.

```
root@47b33e945b34:/etc/nginx/sites-available# ln -s /etc/nginx/sites-available/reverse-proxy.conf /etc/ngi
```

Lastly, to see if it works, we run an nginx configuration test and restart the service.

```
root@47b33e945b34:/etc/nginx/sites-available# cd
root@47b33e945b34:~# service nginx configtest
 * Testing nginx configuration                                           [ OK ]
root@47b33e945b34:~# service nginx restart
 * Restarting nginx nginx                                                [ OK ]
root@47b33e945b34:~#
```

This verifies that nginx works as intended.

To reach other docker containers, we have to use the IP address of the host machine (intel1 in this case) 172.19.0.1

```
root@0501deebfda5:/# curl 172.19.0.1
<!DOCTYPE html>
<html>
--------------------------------- # commented put contents of the zabbix page
root@0501deebfda5:/#
```

4. VM1: Zabbix frontend