Solutions to lab tasks 01-19

1

Done

2

Done. Important to connect the floating IP to the internal network.

3

Installed apache with:

```
    apt-get update
    apt-get install apache2

added PHP and MySQL support with:

    apt-get install libapache2-mod-php php-mysql
    apt-get install mysql-client
```

4

Tested it from manager with `wget -q -O - http://<manager-ip>/`

5

Installed haproxy with:

```
    sudo add-apt-repository -y ppa:vbernat/haproxy-1.8
    apt-get update
    apt-get install haproxy socat
```

configured haproxy by adding the following to /etc/haproxy/haproxy.cfg

```
    frontend bookface
        bind *:80
        mode http
```

```
        default_backend nodes

    backend nodes
        mode http
        balance roundrobin
        server www1 10.10.0.123:80 check
        server www2 10.10.0.134:80 check

    listen stats
        bind *:1936
        stats enable
        stats uri /
        stats hide-version
        stats auth someuser:password
```

Note: had to restart haproxy for it to work. Works with both the private and floating IPs (?)

6

Sent the info. Chose the group name “TnT”

7

(Our best attempt. We don’t feel qualified to give insightful input on this question. We’d love to hear a good answer to this question from you.)

Four tasks/incidents that require coordination between three operations teams; application, database and SAN- and infrastructure, even though it only needs to be handled by one of them:

1. **Application servers hacked.** Only the Application people need to debug it, but the rest of the teams should be aware of the scale of the hack so they can respond if appropriate (it’s their call). The database operators might want to do a search for malicious information in their databases, and the SAN people might want to prepare backups (determine how far they would have to go).
2. **Application server hardware switched out.** Application operators might decide to switch out their hardware for new ones (or their VMs for new ones). Database people should know because they’ll see new hardware suddenly using their databases, which could be suspicious if they weren’t forewarned.
3. **Scaling up/down application servers.** Operators of the application servers should feel free to scale the service up and down as they please, but they have to keep the rest of the teams

up-to-date to avoid scaling issues. The database and the SAN-infrastructure could be overloaded or overpowered.

4. **A database server going down.** It's the database server operators responsability to bring it up again, but the application server operators should be aware, so they don't overload the remaining database server(s).

8

This result tells us the amount of src attributes that point to URLs starting with http. This test is not accurate because the src could be set in JavaScript or other ways we are not aware of. Google gives a result of 0 for example.

Solutions to lab tasks 01-26

1

Made instance with name db1.

- medium size
- ubuntu 16.04 amd64
- manager keypair

Installed mysql server:

```
apt-get update
apt-get install mariadb-server
```

(chose password “dynamitt”)

Configured database. Tested with

```
mysqladmin -u root -p version
mysqladmin -u root -p variables
mysqlshow -p
mysqlshow -p mysql
```

Some useful commands:

```
service mysql start|stop
service mysql status
```

Configuring mysql

Config is under `mariadb.conf.d/50-server.cnf`. Opened it and changed the `bind-address` under `[mysqld]` to 0.0.0.0.

Configured root with new privilides with `GRANT ALL PRIVILEGES ON *.* TO 'root'@'localhost'IDENTIFIED BY 'dynamitt'WITH GRANT OPTION; FLUSH PRIVILEGES;`. (To allow root to connect from manager. We undid this after)

See users with `select * from mysql.user\G`

Tried connecting from manager

```
1     mysql -h server -u user -p mysql
```

NOTE: had to install a `mariadb-client-core-10.0` on manager

Works!

Turned on bin logging

..by going to the cnf file and removing the comment from the `log_bin` line under `[mysqld]`

2

Created db for bookface with `CREATE DATABASE bookfacedb` in mysql

Added bookface user with the command `GRANT SELECT,UPDATE,INSERT,CREATE,DROP,DELETE ON bookfacedb.* TO 'harry'@'10.10.0.%'IDENTIFIED BY 'kaboom';`

Tested that could connect from www1 and www2.

3

Followed the instructions on https://git.cs.hioa.no/kyrre/bookface.

Gave Kyrre our IP. He set up traffic for us. It's working fine :D

Solutions to lab0202

1

Made new instance `dockertest`

- ubuntu 16.04 amd64
- medium size
- manager key

Installing docker ce

Used this installation guide to install Docker CE: https://docs.docker.com/install/linux/docker-ce/ubuntu/

```
    sudo apt-get update

    sudo apt-get install \
        apt-transport-https \
        ca-certificates \
        curl \
        software-properties-common

    curl -fsSL https://download.docker.com/linux/ubuntu/gpg | sudo apt-
       key add -

    sudo apt-key fingerprint 0EBFCD88

    sudo add-apt-repository \
        "deb [arch=amd64] https://download.docker.com/linux/ubuntu \
        $(lsb_release -cs) \
        stable"

    sudo apt-get update

    sudo apt-get install docker-ce

    sudo docker run hello-world
```

Made it into a script that can be found in this directory under the name `dockerInstall.sh`

2

Basic www container (interactive setup)

Made container with `docker run --name wwwtest2 -i -t ubuntu:16.04 /bin/bash`

Followed the instructions on installing bookface from previous lab (**NOTE**: Replaced all occurances of php5 with php).

Committed changes with `docker commit -m "somemessage"wwwtest2 wwwtest3`

Ran new container with same command as above, but exposed port 80 with `-p 80:80`

Have to manually start apache2 upon startup…

Dockerfiled image (made after lab)

- Cloned down bookface from https://git.cs.hioa.no/kyrre/bookface and did the necessary preparations listed in the readme (making `config.php`, filling it with the appropriate values, etc.)
- Made a Dockerfile in the root of the repository that looks like this:

```
1  FROM ubuntu:16.04
2  MAINTAINER Thomas Løkeborg "thomahl@stud.ntnu.no"
3
4  # update and install all necessary services
5  RUN apt-get -y update && apt-get -y install apache2 libapache2-mod
       -php php-mysql
6
7  # ask nicely to expose port 80
8  EXPOSE 80
9
10 # start apache2 by default
11 CMD /usr/sbin/apache2ctl -D FOREGROUND
```

built it by doing `docker build -t dockerfiletest .` and run it by doing `docker run -v /home/ubuntu/bookface/code:/var/www/html -itd -P dockerfiletest`. Can spin up new containers with the same run command, and check their ports with `docker ps`

3

Started up two of the basic www containers. (Have to leave a shell open for each of them… This is bad, but did it cause we spent way too much time messing around with trying to get apache2 running on startup)

4

Added a backend definition to haproxy on manager by adding:

```
    backend dockerNodes
        mode http
        balance roundrobin
        server container1 10.10.1.5:36768 check
        server container1 10.10.1.5:36769 check
```

Made the server use the new backend by altering the default-

Changed the `default_backend` to dockerNodes

5

Diagram is in this directory with the name `infrastructure_diagram.jpg`

6

`docker ps -a` lists all containers (both running and stopped)

7

`docker rm` is used to remove a stopped docker instance

8

`--name` when doing `docker run`, `docker rename <OLD> <NEW>` when already created instance

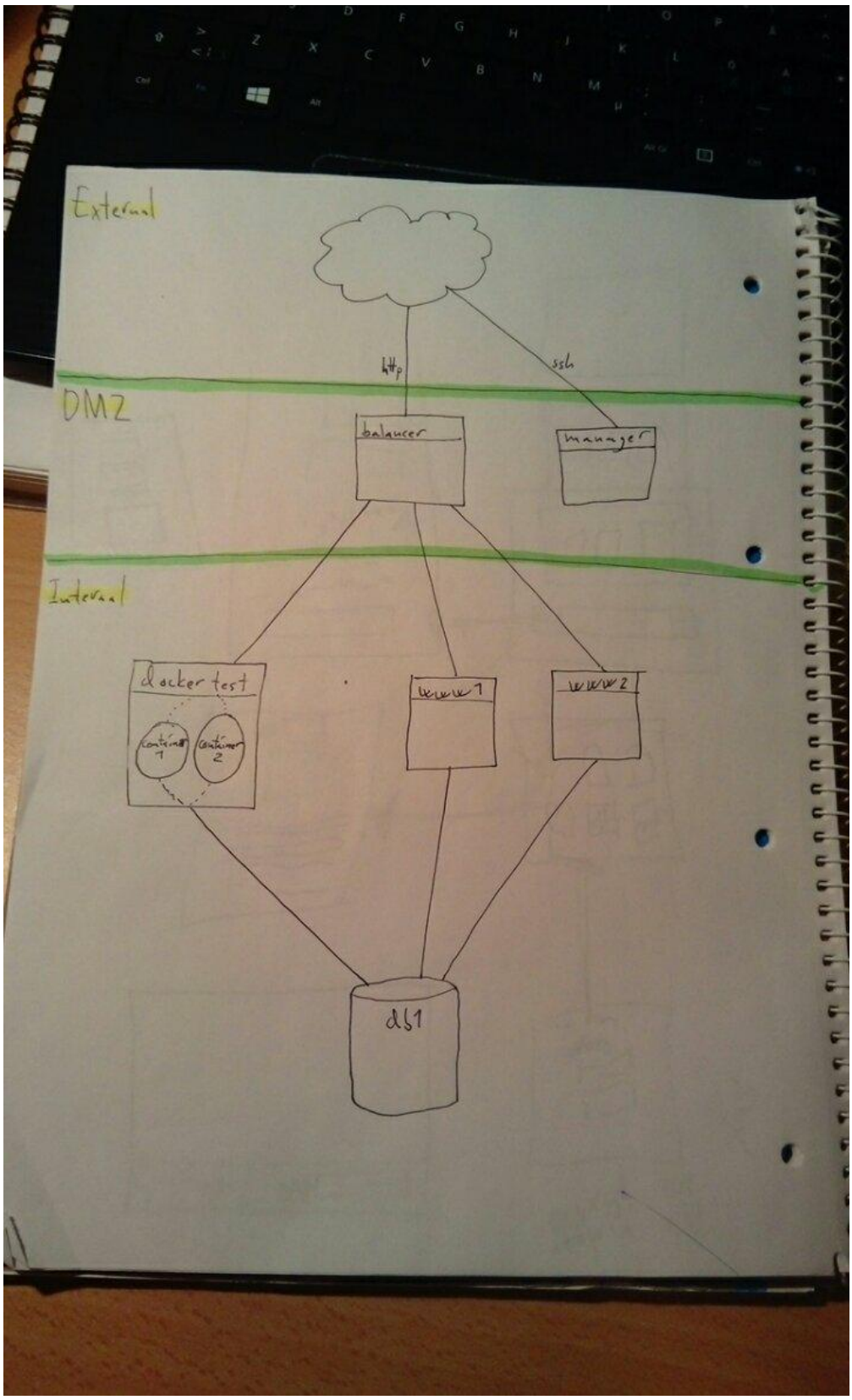

Figure 1: *Diagram*

9

According to our tests. no. Not by default

A quick google search shows that this is possible

10

You can ping the other container if you know it's IP address. By default the containers are started in the network "bridge". To check out their IPs you can run `docker inspect network bridge`

11

Docker swarm is where multiple machines running docker collaborate on running containers. Our architecture could benefit by us running both the db and the www servers in containers, allowing for easy horizontal scaling.

12

Yes we could do this. This means we could easily spin up more docker-enabled VMs to run our containers on. (This also means adding docker-enabled VMs to our swarm would be less painless)

Lab solutions for 0209

1

Formed this sql in mysql cli:

```
SELECT COUNT(*)
FROM user
```

Made it into this bash command: `echo 'SELECT COUNT(*)FROM user'| mysql -u root -pdynamitt bookfacedb`

Made it into a bash function that takes a table as argument

2

Script is found in this directory with the name “task2.sh”

3

Script is found in this directory with the name “task3.sh”

Couldn’t get rid of the ls error message. (Tried sending the output to `/dev/null`)

4

A script version is found in this directory with the name “task4.sh”. Could easily be turned into an alias by either copying it into .bashrc or making the alias refer to the script.

5

Followed the instructions. `openstack server list` works.

6

Did that.

7

Did that. Apparently adding a floating IP that is bound to another server moves it from one server t the other.

Yes, it can be useful when updating the entrypoint to the system. (Creating a new balanacer and moving old floating ip to new one)

8

Script is found in this directory “task8_in.sh” and “task8_out.sh”. They move the floating IP users use to access our website between the balancer and a placeholder server.

9

wc prints newline, bytecount and wordcount for each file

10

- The first command shows partitions and disk usage
- The second command shows directories sorted after disk usage (ascending)

11

- `mkdir /home/ubuntu/supperask` creates a directory named superrask.
- `mount -t tmpfs -o size=20m tmpfs /home/ubuntu/superrask` makes the directory a sticky directory. Meaning that the directory can not be changed by users.
- `chown ubuntu:ubuntu /home/ubuntu/superrask` lets the user ubuntu have permissions to change the sticky directory.
- `umount /home/ubuntu/superrask` reverts the superrask directory back to a normal directory.