

Etudiante : Mariame SANGARE

Classe :M2GL ISI

NB : Etant donné que mon port 8080 est occupé par un agent système, j'ai décidé de prendre un autre port pour mon web 82.

TP3 : Mise en place d'un cluster de 2 nodes

Ce tp consiste à la mise en place d'un cluster avec 2 nodes. Dans cette suite nous allons faire ses différentes étapes :

1. Préparation de l'environnement avec Vagrant

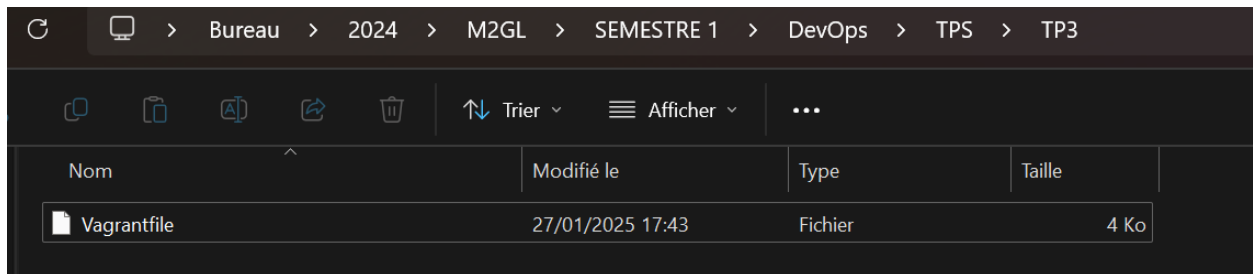
Utiliser un fichier Vagrantfile simplifiera grandement la configuration du cluster. Voici comment procéder :

- Initialiser un Vagrantfile

Tout d'abord on ouvre l'invite de commande puis on se place sur le répertoire où nous allons initialiser notre fichier vagrant ici **TP3** et faire la commande :

```
C:\Users\Mariame\Desktop\2024\M2GL\SEMESTRE 1\DevOps\TPS\TP3>vagrant init
A 'Vagrantfile' has been placed in this directory. You are now
ready to 'vagrant up' your first virtual environment! Please read
the comments in the Vagrantfile as well as documentation on
'vagrantup.com' for more information on using Vagrant.
```

Une fois initialisée nous aurons un fichier Vagrantfile dans notre dossier TP3



A partir de là nous pouvons configurer nos deux machines :

- Une machine web avec Node.js et Spring Boot.

```

14  config.vm.define "webserver" do |webserver|
15      webserver.vm.box = "bento/ubuntu-22.04"
16      webserver.vm.box_check_update = false
17      webserver.vm.network "forwarded_port", guest: 82, host: 8086
18      webserver.vm.network "private_network", ip: "192.168.33.10"
19      webserver.vm.synced_folder "./web", "/var/www/html"
20      webserver.vm.provider "virtualbox" do |vb|
21          vb.gui = false
22          vb.memory = "1024"
23          vb.name = "webserver"
24      end
25  end
26  end

```

Ce fichier de configuration Vagrant définit une machine virtuelle appelée "webserver" basée sur Ubuntu 22.04. La machine a un port forwardé (port 82 sur la machine virtuelle vers le port 8086 sur l'hôte) et utilise un réseau privé avec l'adresse IP 192.168.33.10. Un dossier local est synchronisé avec "/var/www/html" sur la machine virtuelle. La configuration VirtualBox alloue 1 Go de RAM et 1 CPU à la machine, avec l'interface graphique désactivée.

- Une machine db avec MySQL.

```

1  Vagrant.configure("2") do |config|
2      config.vm.define "dbserver" do |dbserver|
3          dbserver.vm.box = "bento/ubuntu-22.04"
4          dbserver.vm.box_check_update = false
5          dbserver.vm.network "forwarded_port", guest: 84, host: 9096
6          dbserver.vm.network "private_network", ip: "192.168.33.11"
7          dbserver.vm.synced_folder "./db", "/var/www/html"
8          dbserver.vm.provider "virtualbox" do |vb|
9              vb.gui = false
10             vb.memory = "1024"
11             vb.name = "dbserver"
12         end
13     end

```

- Créer les répertoires web et db

Bureau > 2024 > M2GL > SEMESTRE 1 > DevOps > TPS > TP3 >					Rec
Trier ▾ Afficher ▾ ...					
Nom	Modifié le	Type	Taille		
.vagrant	25/02/2025 13:39	Dossier de fichiers			
admin-app	27/01/2025 18:14	Dossier de fichiers			
db	25/02/2025 13:40	Dossier de fichiers			
web	25/02/2025 13:40	Dossier de fichiers			
Vagrantfile	25/02/2025 13:39	Fichier	1 Ko		

Maintenant on peut valider puis démarrer nos machines avec la commande :

```
vagrant@vagrant:~$ Connection to 127.0.0.1 closed by remote host.
PS C:\Users\Mariame\Desktop\2024\M2GL\SEMESTRE 1\DevOps\TPS\TP3> vagrant up webserver
Bringing machine 'webserver' up with 'virtualbox' provider...
==> webserver: Importing base box 'bento/ubuntu-22.04'...
==> webserver: Matching MAC address for NAT networking...
==> webserver: Setting the name of the VM: webserver
==> webserver: Clearing any previously set network interfaces...
==> webserver: Preparing network interfaces based on configuration...
webserver: Adapter 1: nat
webserver: Adapter 2: hostonly
==> webserver: Forwarding ports...
webserver: 82 (guest) => 8086 (host) (adapter 1)
webserver: 22 (guest) => 2222 (host) (adapter 1)
==> webserver: Running 'pre-boot' VM customizations...
==> webserver: Booting VM...
==> webserver: Waiting for machine to boot. This may take a few minutes...
webserver: SSH address: 127.0.0.1:2222
webserver: SSH username: vagrant
webserver: SSH auth method: private key
webserver:
webserver: Vagrant insecure key detected. Vagrant will automatically replace
webserver: this with a newly generated keypair for better security.
webserver:
webserver: Inserting generated public key within guest...
webserver: Removing insecure key from the guest if it's present...
webserver: Key inserted! Disconnecting and reconnecting using new SSH key...
==> webserver: Machine booted and ready!
==> webserver: Checking for guest additions in VM...
webserver: The guest additions on this VM do not match the installed version of
webserver: VirtualBox! In most cases this is fine, but in rare cases it can
```

On fait pareil pour dbserver

Nous pourrions accéder à chaque machine avec :

```
PS C:\Users\Mariame\Desktop\2024\M2GL\SEMESTRE 1\DevOps\TPS\TP3> vagrant ssh webserver
Welcome to Ubuntu 22.04.5 LTS (GNU/Linux 5.15.0-133-generic x86_64)

 * Documentation:  https://help.ubuntu.com
 * Management:    https://landscape.canonical.com
 * Support:       https://ubuntu.com/pro

System information as of Tue Feb 25 01:55:15 PM UTC 2025

System load:  0.83           Processes:            156
Usage of /:   15.1% of 30.34GB Users logged in:          0
Memory usage: 21%           IPv4 address for eth0: 10.0.2.15
Swap usage:   0%

This system is built by the Bento project by Chef Software
More information can be found at https://github.com/chef/bento

Use of this system is acceptance of the OS vendor EULA and License Agreements.
vagrant@vagrant:~$
```

```
PS C:\Users\Mariame\Desktop\2024\M2GL\SEMESTRE 1\DevOps\TPS\TP3> vagrant ssh webserver
Welcome to Ubuntu 22.04.5 LTS (GNU/Linux 5.15.0-133-generic x86_64)

 * Documentation:  https://help.ubuntu.com
 * Management:    https://landscape.canonical.com
 * Support:       https://ubuntu.com/pro

System information as of Tue Feb 25 02:19:26 PM UTC 2025

System load:  0.3           Processes:            156
Usage of /:   15.8% of 30.34GB Users logged in:          0
Memory usage: 23%           IPv4 address for eth0: 10.0.2.15
Swap usage:   0%

This system is built by the Bento project by Chef Software
More information can be found at https://github.com/chef/bento

Use of this system is acceptance of the OS vendor EULA and License Agreements.
Last login: Tue Feb 25 13:55:16 2025 from 10.0.2.2
vagrant@vagrant:~$
```

2. Déploiement du projet Spring Boot

Nous allons tout d'abord cloner le projet

```

vagrant@ubuntu-bionic:~$ git clone https://github.com/ngorseck/admin-app
Cloning into 'admin-app'...
remote: Enumerating objects: 237, done.
remote: Counting objects: 100% (237/237), done.
remote: Compressing objects: 100% (164/164), done.
remote: Total 237 (delta 100), reused 188 (delta 53), pack-reused 0 (from 0)
Receiving objects: 100% (237/237), 893.95 KiB | 525.00 KiB/s, done.
Resolving deltas: 100% (100/100), done.
vagrant@ubuntu-bionic:~$ ls
admin-app  npm-debug.log
vagrant@ubuntu-bionic:~$ cd admin-app
vagrant@ubuntu-bionic:~/admin-app$

```

3. Configuration de la machine db

On va installer mysql dans la machine dbserver puis vérifier le statut de notre serveur mysql

```

root@vagrant:/home/vagrant# apt-cache policy mysql-server
mysql-server:
  Installed: 8.0.41-0ubuntu0.22.04.1
  Candidate: 8.0.41-0ubuntu0.22.04.1
  Version table:
 *** 8.0.41-0ubuntu0.22.04.1 500
      500 http://us.archive.ubuntu.com/ubuntu jammy-updates/main amd64 Packages
      500 http://security.ubuntu.com/ubuntu jammy-security/main amd64 Packages
 100 /var/lib/dpkg/status
 8.0.28-0ubuntu4 500
      500 http://us.archive.ubuntu.com/ubuntu jammy/main amd64 Packages
root@vagrant:/home/vagrant#

```

```

vagrant@vagrant:~$ sudo systemctl status mysql
● mysql.service - MySQL Community Server
   Loaded: loaded (/lib/systemd/system/mysql.service; enabled; vendor preset: enabled)
   Active: active (running) since Wed 2025-02-26 13:06:30 UTC; 29min ago
     Process: 3021 ExecStartPre=/usr/share/mysql/mysql-systemd-start pre (code=exited, status=0/SUCCESS)
    Main PID: 3029 (mysqld)
      Status: "Server is operational"
        Tasks: 38 (limit: 1015)
       Memory: 368.2M
          CPU: 53.048s
      CGroup: /system.slice/mysql.service
              └─3029 /usr/sbin/mysqld

Feb 26 13:06:25 vagrant systemd[1]: Starting MySQL Community Server...
Feb 26 13:06:30 vagrant systemd[1]: Started MySQL Community Server.
vagrant@vagrant:~$

```

Une fois l'installation et le démarrage vérifiée, nous nous connectons à notre serveur mysql

```

vagrant@vagrant:~$ sudo mysql -u root -p
Enter password:
Welcome to the MySQL monitor.  Commands end with ; or \g.
Your MySQL connection id is 11
Server version: 8.0.41-0ubuntu0.22.04.1 (Ubuntu)

Copyright (c) 2000, 2025, Oracle and/or its affiliates.

Oracle is a registered trademark of Oracle Corporation and/or its
affiliates. Other names may be trademarks of their respective
owners.

Type 'help;' or '\h' for help. Type '\c' to clear the current input statement.

mysql>

```

Puis nous créons un utilisateur qui aura accès à l'adresse ip de notre machine web avec l'adresse ip 192.168.33.10 avec le mot de passe 'M@rivme5758' et lui donner tous les privilèges.

```

mysql> create user 'myuser'@'192.168.33.10' IDENTIFIED BY 'M@rivme5758';
Query OK, 0 rows affected (0.02 sec)

mysql>

```

```

mysql> grant all privileges ON adminappdb.* to myuser@'192.168.33.10';
Query OK, 0 rows affected (0.02 sec)

mysql> flush privileges;
Query OK, 0 rows affected (0.01 sec)

mysql>

```

Pour vérifier que notre utilisateur a été créé avec notre adresse ip on fait la commande ci-dessous :

```

mysql> SELECT user, host FROM mysql.user;
+-----+-----+
| user          | host          |
+-----+-----+
| myuser        | 192.168.33.10 |
| debian-sys-maint | localhost    |
| mysql.infoschema | localhost    |
| mysql.session  | localhost    |
| mysql.sys      | localhost    |
| root          | localhost    |
+-----+-----+
6 rows in set (0.01 sec)

mysql>

```

Dans cette image ci-dessous on voit que la communication à la machine webserver marche.

```
bye
vagrant@vagrant:~$ ping 192.168.33.10
PING 192.168.33.10 (192.168.33.10) 56(84) bytes of data.
64 bytes from 192.168.33.10: icmp_seq=1 ttl=64 time=11.9 ms
64 bytes from 192.168.33.10: icmp_seq=2 ttl=64 time=1.13 ms
64 bytes from 192.168.33.10: icmp_seq=3 ttl=64 time=1.18 ms
64 bytes from 192.168.33.10: icmp_seq=4 ttl=64 time=1.89 ms
64 bytes from 192.168.33.10: icmp_seq=5 ttl=64 time=1.21 ms
64 bytes from 192.168.33.10: icmp_seq=6 ttl=64 time=1.39 ms
64 bytes from 192.168.33.10: icmp_seq=7 ttl=64 time=1.12 ms
```

Creation de la base de données adminapp-db

```
mysql> CREATE DATABASE `adminapp-db`;
Query OK, 1 row affected (0.04 sec)

mysql>
```

Après création de l'utilisateur myuser et de notre base de données on modifie le fichier application.yml en mettant les informations liées à notre serveur web.

```
server:
  port: 8080

spring:
  datasource:
    url: jdbc:mysql://${DB_HOST:localhost}:3306/${DB_NAME:adminapp-db}?createDatabaseIfNotExist=true&useUnicode=true&
    username: ${DB_USERNAME:myuser}
    password: ${DB_PASSWORD:M@rivme5758}
    driverClassName: com.mysql.cj.jdbc.Driver
    #Pour aws, nous n'avons pas besoin de driver
#    url: jdbc:mysql://database-mysql1.cv3dshuqxbdr.us-east-1.rds.amazonaws.com:3306/db_test?createDatabaseIfNotExist
#    username: admin
#    password: h1KQUq[#R{XW+wh)Ji!xs8Vs:X0[
#    url: jdbc:mysql://adminapp-db.cv3dshuqxbdr.us-east-1.rds.amazonaws.com:3306/adminappdb
#    username: admin
#    password: SamanecorporationPasser123!
  application:
    name: admin-app
#SamanecorporationPasser123!
  jpa:
    hibernate:
      ddl-auto: update
    show-sql: true
    properties:
      hibernate:
        format_sql: true
```

On sauvegarde le fichier et tapons cette commande pour nettoyer et installer les paquets tout en ignorant les tests pour éviter que ça nous génère des erreurs

```
vagrant@vagrant:~/admin-app$ mvn clean install -DskipTests
[INFO] Scanning for projects...
Downloading from central: https://repo.maven.apache.org/maven2/org/springframework/boot/spring-boot-starter-parent/3.1.2/spring-boot-starter-parent-3.1.2.pom
Downloaded from central: https://repo.maven.apache.org/maven2/org/springframework/boot/spring-boot-starter-parent/3.1.2/spring-boot-starter-parent-3.1.2.pom (13 kB at 7.4 kB/s)
Downloading from central: https://repo.maven.apache.org/maven2/org/springframework/boot/spring-boot-dependencies/3.1.2/spring-boot-dependencies-3.1.2.pom
Downloaded from central: https://repo.maven.apache.org/maven2/org/springframework/boot/spring-boot-dependencies/3.1.2/spring-boot-dependencies-3.1.2.pom (94 kB at 236 kB/s)
```

Et Bim !!!! Build success

```

vagrant@vagrant:~/admin-app/target$ ls
admin-app.jar          classes                generated-test-sources  maven-status
admin-app.jar.original generated-sources       maven-archiver         test-classes
vagrant@vagrant:~/admin-app/target$

```

On compile le fichier jar puis le démarrons

```
vagrant@vagrant:~/admin-app$ java -jar target/admin-app.jar
```

[illegible]

Connexion avec postman et le navigateur

