

## Master II BDCC- Big Data et Cloud Computing

## TP 4: Docker (Network, Storage, Sécurité, Compose, Swarm, Kubernetes)

## PARTIE 2 : Swarm

1. Démarrer deux VMs contenant docker avec des adresses IP différentes.

The screenshot shows two Oracle VM VirtualBox windows. The top window, titled 'VM1 [En fonction] - Oracle VM VirtualBox', displays the network configuration for VM1. It shows the 'Adapter 1' settings with 'Promiscuous Mode' set to 'Allow All' and 'Capture' checked. Below this, the network interface 'eth0' is configured with IP 192.168.1.42, netmask 255.255.255.0, and gateway 192.168.1.1. The bottom window shows the terminal output of VM1, where the user runs 'ifconfig' and 'ping 192.168.1.42'. The output shows that the interface is up and running, and the ping test is successful.

2. Créer un cluster SWARM (initialiser une VM comme manager et l'autre comme worker).

The screenshot shows a terminal window where the user runs 'docker swarm init'. The output indicates that the current node (rd2j1xgqyr28g6y61ly570t0u) is now a manager. The user then runs 'docker swarm join --token SWMTKN-1-17b6umj91ebjj7zbp0jgozy6uetjoug9yym7m43d7te52hb8-27crrekjx56dvritghtx8ry5l 192.168.1.33:2377' to add a worker node to the swarm.

3. Afficher les nœuds actifs du SWARM.

The screenshot shows a terminal window where the user runs 'docker node ls'. The output displays a table with columns: ID, HOSTNAME, STATUS, AVAILABILITY, MANAGER STATUS, and ENGINE VERSION. Two nodes are listed: '5exhtrawsqgbhiuvnwiqv7wrd' (vm1) and '9yxwfw4fn219g88d5ygtshpjo' (vm2), both with a status of 'Ready' and availability of 'Active'.

4. Exécuter l'image (TP2.PARTIE3.2) avec docker SWARM sous le nom swarm1.

The screenshot shows a terminal window where the user runs 'docker service create --name swarm1 webapp'. The output indicates that the service 'swarm1' has been created successfully.

5. Lister les services actifs dans le cluster.

The screenshot shows a terminal window where the user runs 'docker service ls'. The output displays a table with columns: ID, NAME, MODE, REPLICAS, IMAGE, and PORTS. One service is listed: 'swarm1' with a mode of 'replicated' and 1/1 replicas.

6. Afficher les détails du service swarm1.

```
root@vm1:/home/vm1# docker service ls
ID            NAME      MODE          REPLICAS  IMAGE      PORTS
qo3pjff7j1o1 swarm1    replicated    1/1        webapp:latest
root@vm1:/home/vm1# docker service ps swarm1
ID            NAME      IMAGE          NODE       DESIRED STATE  CURRENT STATE      ERROR
PORTS
qz5t6y8nj1ph swarm1.1   webapp:latest  vm1        Running         Running 43 seconds ago
```

**Docker service inspect swarm1**

7. Consulter le log du service swarm1.

```
root@vm1:/home/vm1# docker service logs_swarm1
```

8. Accéder à la page index depuis les adresses IP des deux VMs.

**Docker service update --publish-add published=8080,target=80 swarm1**

**http://192.168.1.33:8080**

**http://192.168.1.42:8080**

9. Augmenter la charge 5 pour le service swarm1.

**Docker service update --replicas 5 swarm1**

**ou**

**Docker service scale q03p =5**

10. Afficher les détails du service swarm1.

```
root@vm1:/home/vm1# docker service ls
ID            NAME      MODE          REPLICAS  IMAGE      PORTS
qo3pjff7j1o1 swarm1    replicated    1/1        webapp:latest
root@vm1:/home/vm1# docker service ps swarm1
ID            NAME      IMAGE          NODE       DESIRED STATE  CURRENT STATE      ERROR
PORTS
qz5t6y8nj1ph swarm1.1   webapp:latest  vm1        Running         Running 43 seconds ago
```

**Docker service inspect swarm1**

**Docker service ps swarm1**

11. Faire une mise à jour du service swarm1 sans arrêter le système (une nouvelle image avec un autre tag du TP2.PARTIE3.2 affichant welcome bdcc V2) ; revenir vers la version précédente.

**Docker service update --image webapp:v2 swarm1**

**http://192.168.1.33:8080**

**http://192.168.1.42:8080**

**Docker service rollback swarm1**

**http://192.168.1.33:8080**

**http://192.168.1.42:8080**

12. Activer la HA pour suspendre les conteneurs du nœud worker (en cas de maintenance...) vers l'autre nœud d'une manière équilibrée.

**Docker node update --availability drain 3yxf**

13. Supprimer le service swarm1.

```
root@vm1:/home/vm1# docker service rm swarm1
```

**Docker service rm swarm1**

14. Conditionner le démarrage du service swarm1 pour exécuter l'image (TP2.PARTIE3.2) selon des contraintes (dans le worker avec un nombre de réplication dépendant des ressources disponibles).

**Docker service create --name swarm1 --constraint 'node.role==worker' --limit-memory=1GB webapp**

**Ressources :**

<https://docs.docker.com/engine/swarm/>

**PARTIE 3 : Kubernetes**

1. Installer et démarrer minikube, kubectl.

```
PS C:\Users\Kamal EL GUEMMAT> choco install minikube
```

```
C:\Windows\system32>minikube start
```

**Kubectl get nodes**

2. Déployer l'image (TP2.PARTIE3.2) comme un nouveau service.

**Minikube image pull kelguemmat/tp2**

**Minikube image ls**

**Kubectl create deployment kubernetes1 --image=docker.io. kelguemmat/tp2:latest**

**Kubectl get pods**

3. Exposer ce service pour les accès externes ; accéder à ce service.

**Minikube tunnel**

**Kubectl expose deployment kubernetes1 --type loadbalancer --port=8080**

**Kubectl get services**

**Kubectl get services kubernetes1**

**Minikube ip**

**Minikube service kubernetes1**

**http://192.168.1.33:8080**

4. Décrire l'état du service.

**kubectl get deployments**

**kubectl get pods**

**Kubectl describe services kubernetes1**

5. Augmenter les scales pour assurer un équilibrage de charge.

**Kubectl scale --replicas =3 deployment kubernetes1**

6. Accéder au tableau de bord Kubernetes.

**C:\Minikube dashboard**

7. Supprimer le service.

**Kubectl delete services kubernetes1**

8. Supprimer le déploiement.

**Kubectl delete deployment kubernetes1**

9. Arrêter minikube.

**C:\Minikube stop**

10. Supprimer minikube

**C:\Minikube delete**

**Ressources :**

<https://kubernetes.io/fr/>

<https://kubernetes.io/fr/docs/tutorials/kubernetes-basics/create-cluster/cluster-interactive>

<https://kubernetes.io/fr/docs/tasks/access-application-cluster/web-ui-dashboard/>