TP3 – Déployer DockerCoins sur un cluster Docker swarm

Générer des bitcoins

Le minage c'est le procédé par lequel les transactions Bitcoin sont sécurisées. A cette fin les mineurs effectuent avec leur matériel informatique des calculs mathématiques pour le réseau Bitcoin. Comme récompense pour leurs services, ils collectent les bitcoins nouvellement créés ainsi que les frais des transactions qu'ils confirment.

Les mineurs sont en concurrence et leurs revenus sont proportionnels à la puissance de calcul déployée.

— minage de bitcoins, https://bitcoin.fr/minage/

Dockercoins

Notre application exemple est Dockercoins, une application de minage de Dockercoins!

Toutes les ressources de cette application (fictive et complétement inutile) sont disponibles sur le dépôt *https://github.com/brahimhamdi/dockercoins*.

Elle est composée de 4 microservices dans le dossier dockercoins (+une base de données redis):

- rng = un service web(Python) générant en sortie des nombres aléatoires
- hasher = un service web(Ruby) générant en sortie un hash des données qui lui sont envoyées par HTTP POST
- worker = processus(Python) utilisant rng et hasher
- webui = web interface (JS)

Principe :

- worker demande à rng de lui fournir des données aléatoires
- worker injecte ces données dans hasher , hasher génére un hash, récupéré par worker,
- Pour chaque hash commençant par 0, worker génére un DockerCoin
- Le worker stocke les DockerCoins générés dans une base de données Redis,
- La webui affiche le taux d'hashage par seconde.

Vous remarquerez dans les codes précédents que chaque service est appelé par un nom DNS simple (hasher, rng, etc..). Ce sera notre rôle de démarrer ces différents services avec ce nom pour que le code soit valide.

Dans la suite, nous déploierons la même application Dockercoins sur un cluster composé de : 1 Manager et 2 Workers. Docker Engin doit être installé et fonctionnel sur chacune de ces 3 machines, pour cela nous allons utiliser Vagrant pour préparer cet environnement.

Initialisation du cluster

1. Récupérer l'environnement du TP avec la commande suivante :
   git clone https://github.com/brahimhamdi/docker-swarm-lab
2. Sous le répertoire docker-swarm-lab, démarrez les 3 VMs (Manager, Worker1 et Worker2) en tapant la commande suivante : *vagrant up*

```
ubuntu@formation1:~/docker-swarm-lab$ vagrant up
Bringing machine 'worker1' up with 'virtualbox' provider...
Bringing machine 'worker2' up with 'virtualbox' provider...
==> worker1: Importing base box 'generic/ubuntu2004'...
==> worker1: Matching MAC address for NAT networking...
==> worker1: Checking if box 'generic/ubuntu2004' version '4.2.16' is up to date...
==> worker1: Setting the name of the VM: docker-swarm-lab_worker1_1689514051293_75574
==> worker1: Clearing any previously set network interfaces...
==> worker1: Preparing network interfaces based on configuration...
    worker1: Adapter 1: nat
    worker1: Adapter 2: hostonly
==> worker1: Forwarding ports...
    worker1: 22 (guest) => 2222 (host) (adapter 1)
==> worker1: Running 'pre-boot' VM customizations...
==> worker1: Booting VM...
==> worker1: Waiting for machine to boot. This may take a few minutes...
    worker1: SSH address: 127.0.0.1:2222
    worker1: SSH username: vagrant
    worker1: SSH auth method: private key
```

3. Sous le Manager, initialisez le cluster (le swarm) en annonçant son IP:
   docker swarm init --advertise-addr=192.168.205.1

```
ubuntu@formation1:~$ docker swarm init --advertise-addr=192.168.205.1
Swarm initialized: current node (7j8yc25j5g211y2iqvsylkmhz) is now a manager.

To add a worker to this swarm, run the following command:

    docker swarm join --token SWMTKN-1-4qv3de07qi3wf0e5r6bqa05d0t9ffnagqi9guda8i9r0a4k42l-ckww48b946890q322ut2kpc3w 192.168.205.1:2377

To add a manager to this swarm, run 'docker swarm join-token manager' and follow the instructions.

ubuntu@formation1:~$
```

   - Que génère cette commande à la fin ?
   - Vérifiez que le cluster est crée et que la VM Manager est le seul nœud dans ce cluster jusqu'à maintenant.
     docker node ls

```
ubuntu@formation1:~$ docker node ls
ID                            HOSTNAME     STATUS    AVAILABILITY   MANAGER STATUS   ENGINE VERSION
7j8yc25j5g211y2iqvsylkmhz *   formation1   Ready     Active         Leader           20.10.21
ubuntu@formation1:~$
ubuntu@formation1:~$
```

4. Joindre worker1 et worker2 au cluster (en tant que workers).

vagrant ssh worker1

docker swarm join --token …

```
ubuntu@formation1:~$ cd docker-swarm-lab/
ubuntu@formation1:~/docker-swarm-lab$ vagrant ssh worker1
vagrant@worker1:~$ sudo  docker swarm join --token SWMTKN-1-4qv3de07qi3wf0e5r6bqa05d0t9ffnagqi9guda8i9r0a4k42l-ckww48b946890q322ut2kpc3w 192.1
68.205.1:2377
This node joined a swarm as a worker.
vagrant@worker1:~$
vagrant@worker1:~$
```

Pareil pour worker2.

```
vagrant@worker1:~$ exit
logout
ubuntu@formation1:~/docker-swarm-lab$ vagrant ssh worker2
vagrant@worker2:~$ sudo docker swarm join --token SWMTKN-1-4qv3de07qi3wf0e5r6bqa05d0t9ffnagqi9guda8i9r0a4k42l-ckww48b946890q322ut2kpc3w 192.16
8.205.1:2377
This node joined a swarm as a worker.
vagrant@worker2:~$
vagrant@worker2:~$
```

- Sur le Manager, vérifiez que les 2 workers sont dans le cluster et « ready ».

```
vagrant@worker2:~$ exit
logout
ubuntu@formation1:~/docker-swarm-lab$ docker node ls
ID                            HOSTNAME     STATUS    AVAILABILITY   MANAGER STATUS   ENGINE VERSION
7j8yc25j5g211y2iqvsylkmhz *   formation1   Ready     Active         Leader           20.10.21
vvvjjhzemnh2dn0fxpw2zjwgr     worker1      Ready     Active                          24.0.2
qoycpsbpkf1oqbjv5gefmc6rx     worker2      Ready     Active                          24.0.2
ubuntu@formation1:~/docker-swarm-lab$
ubuntu@formation1:~/docker-swarm-lab$
```

5. Quels sont les réseaux crées suite à l'initialisation du cluster.

   docker network ls

```
ubuntu@formation1:~/docker-swarm-lab$ docker network ls
NETWORK ID     NAME              DRIVER    SCOPE
15c26d892c9b   bridge            bridge    local
85983dd658d1   docker_gwbridge   bridge    local
2a55a3d75042   host              host      local
0101dha3vlry   ingress           overlay   swarm
9d7cf89781de   none              null      local
ubuntu@formation1:~/docker-swarm-lab$
ubuntu@formation1:~/docker-swarm-lab$
```

Lancement de l'application Dockercoins sur le Swarm

Maintenant que le cluster est prêt, on va déployer les services de l'application Dockercoins.

6. Créer le réseau dockercoins-net de type overlay. On va connecter tous les conteneurs de l'application à ce réseau.

```
ubuntu@formation1:~/docker-swarm-lab$ docker network create -d overlay dockercoins-net
l6v6ocuzsknuz8nbvulehj2ji
ubuntu@formation1:~/docker-swarm-lab$
ubuntu@formation1:~/docker-swarm-lab$ docker network ls
NETWORK ID     NAME              DRIVER    SCOPE
15c26d892c9b   bridge            bridge    local
85983dd658d1   docker_gwbridge   bridge    local
l6v6ocuzsknu   dockercoins-net   overlay   swarm
2a55a3d75042   host              host      local
0101dha3vlry   ingress           overlay   swarm
9d7cf89781de   none              null      local
ubuntu@formation1:~/docker-swarm-lab$
```

7. Créez le service de la base de données redis.

docker service create --name redis --network dockercoins-net redis

```
ubuntu@formation1:~/docker-swarm-lab$ docker service create --name redis --network dockercoins-net redis
bk0rte0ih61v4i8x8ttnts1yw
overall progress: 1 out of 1 tasks
1/1: running   [==================================================>]
verify: Service converged
ubuntu@formation1:~/docker-swarm-lab$
ubuntu@formation1:~/docker-swarm-lab$
```

- Attendez que le serveur soit à l'état "running".

docker service ps redis

```
ubuntu@formation1:~/docker-swarm-lab$ docker service ls
ID             NAME      MODE          REPLICAS   IMAGE          PORTS
bk0rte0ih61v   redis     replicated    1/1        redis:latest
ubuntu@formation1:~/docker-swarm-lab$
ubuntu@formation1:~/docker-swarm-lab$ docker service ps redis
ID             NAME      IMAGE          NODE          DESIRED STATE   CURRENT STATE            ERROR     PORTS
8ljqr5297k97   redis.1   redis:latest   formation1    Running         Running 44 seconds ago
ubuntu@formation1:~/docker-swarm-lab$
ubuntu@formation1:~/docker-swarm-lab$
```

- Une fois que redis est prêt, créez les services hasher, rng et worker.

docker service create --name hasher --network dockercoins-net brahimhamdi/hasher

docker service create --name rng --network dockercoins-net brahimhamdi/rng

docker service create --name worker --network dockercoins-net brahimhamdi/worker

```
ubuntu@formation1:~/docker-swarm-lab$ docker service create --name hasher --network dockercoins-net brahimhamdi/hasher
z9cj853mfh5st9jk4gben7e9k
overall progress: 1 out of 1 tasks
1/1: running   [==================================================>]
verify: Waiting 3 seconds to verify that tasks are stable...
verify: Service converged
ubuntu@formation1:~/docker-swarm-lab$
ubuntu@formation1:~/docker-swarm-lab$ docker service create --name rng --network dockercoins-net brahimhamdi/rng
f7eehf22dkejd8vmlv831mqp2
overall progress: 1 out of 1 tasks
1/1: running   [==================================================>]
verify: Service converged
ubuntu@formation1:~/docker-swarm-lab$
ubuntu@formation1:~/docker-swarm-lab$ docker service create --name worker --network dockercoins-net brahimhamdi/worker
m6zhv5h7j119p670jni9z4ps2
overall progress: 1 out of 1 tasks
1/1: running   [==================================================>]
verify: Service converged
ubuntu@formation1:~/docker-swarm-lab$
ubuntu@formation1:~/docker-swarm-lab$
```

- Vérifiez que tous les services ont été bien crées. Sur quels nœuds les conteneurs sont-ils crées ?

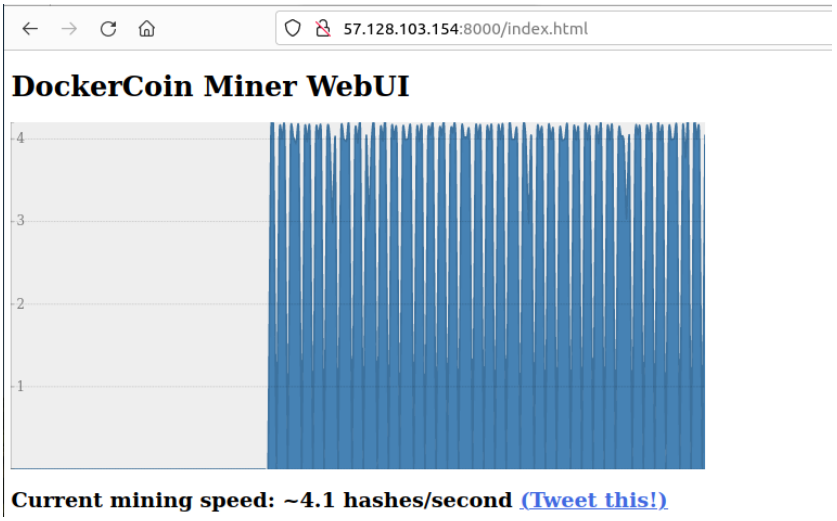
```
ubuntu@formation1:~/docker-swarm-lab$ docker service ls
ID             NAME      MODE         REPLICAS   IMAGE                        PORTS
z9cj853mfh5s   hasher    replicated   1/1        brahimhamdi/hasher:latest
bk0rte0ih61v   redis     replicated   1/1        redis:latest
f7eehf22dkej   rng       replicated   1/1        brahimhamdi/rng:latest
m6zhv5h7j119   worker    replicated   1/1        brahimhamdi/worker:latest
ubuntu@formation1:~/docker-swarm-lab$
ubuntu@formation1:~/docker-swarm-lab$ docker service ps hasher redis rng worker
ID             NAME       IMAGE                        NODE         DESIRED STATE   CURRENT STATE                ERROR     PORTS
nqmgvmo0oosy   hasher.1   brahimhamdi/hasher:latest    worker1      Running         Running about a minute ago
8ljqr5297k97   redis.1    redis:latest                 formation1   Running         Running 3 minutes ago
xrkjp3nfp5vt   rng.1      brahimhamdi/rng:latest       worker2      Running         Running about a minute ago
k7jujw1o7s97   worker.1   brahimhamdi/worker:latest    formation1   Running         Running 55 seconds ago
ubuntu@formation1:~/docker-swarm-lab$
ubuntu@formation1:~/docker-swarm-lab$
```

8. Une fois les 4 services sont prêts, lancez le service webui en mappant le port 8000 de l'hôte sur le port 80 du service.

docker service create --name webui --network dockercoins-net -p 8000:80 brahimhamdi/webui

```
ubuntu@formation1:~/docker-swarm-lab$ docker service create --name webui --network dockercoins-net -p 8000:80 brahimhamdi/webui
7gsm38zlg5zkdt1xi6qeq3niw
overall progress: 1 out of 1 tasks
1/1: running   [==================================================>]
verify: Service converged
ubuntu@formation1:~/docker-swarm-lab$
ubuntu@formation1:~/docker-swarm-lab$ docker service ps webui
ID             NAME      IMAGE                     NODE      DESIRED STATE   CURRENT STATE            ERROR     PORTS
jhgbmtzs52p5   webui.1   brahimhamdi/webui:latest   worker1   Running         Running 11 seconds ago
ubuntu@formation1:~/docker-swarm-lab$
ubuntu@formation1:~/docker-swarm-lab$
```

- Vous pouvez maintenant affichez l'interface de votre application qui tourne sur un cluster Docker Swarm.



Passage à l'échelle sous Swarm

9. Maintenant que l'application tourne sur plusieurs hôtes Docker, on va répliquer les services de l'application pour augmenter la productivité. Ça s'appelle le mode répliqué, c'est le mode par défaut de Docker Sworm.

Multipliez le nombre de worker par 10.

docker service scale worker=10

```
ubuntu@formation1:~/docker-swarm-lab$ docker service scale worker=10
worker scaled to 10
overall progress: 10 out of 10 tasks
1/10: running   [==================================================>]
2/10: running   [==================================================>]
3/10: running   [==================================================>]
4/10: running   [==================================================>]
5/10: running   [==================================================>]
6/10: running   [==================================================>]
7/10: running   [==================================================>]
8/10: running   [==================================================>]
9/10: running   [==================================================>]
10/10: running   [==================================================>]
verify: Waiting 1 seconds to verify that tasks are stable...
verify: Service converged
ubuntu@formation1:~/docker-swarm-lab$
ubuntu@formation1:~/docker-swarm-lab$ docker service ps worker
ID             NAME        IMAGE                     NODE         DESIRED STATE   CURRENT STATE            ERROR     PORTS
k7jujw1o7s97   worker.1    brahimhamdi/worker:latest   formation1   Running         Running 7 minutes ago
qohezgmho1d3   worker.2    brahimhamdi/worker:latest   formation1   Running         Running 12 seconds ago
hmizbxqyp2qq   worker.3    brahimhamdi/worker:latest   worker1      Running         Running 12 seconds ago
w76jfl0uxrzc   worker.4    brahimhamdi/worker:latest   worker1      Running         Running 12 seconds ago
gcpbxtnhuypw   worker.5    brahimhamdi/worker:latest   formation1   Running         Running 12 seconds ago
eg719zahmiqj   worker.6    brahimhamdi/worker:latest   worker2      Running         Running 11 seconds ago
5w15dnwvl8ii   worker.7    brahimhamdi/worker:latest   worker1      Running         Running 12 seconds ago
uh7kk5b2dpzz   worker.8    brahimhamdi/worker:latest   worker2      Running         Running 11 seconds ago
1ij4a7gsobav   worker.9    brahimhamdi/worker:latest   worker2      Running         Running 11 seconds ago
lcga52oqg8t7   worker.10   brahimhamdi/worker:latest   formation1   Running         Running 12 seconds ago
ubuntu@formation1:~/docker-swarm-lab$
```

- Une fois que tous les replicas sont à l'état "running", affichez l'interface webui.

  - Quel est le taux de génération de dockercoins?
  - Sur quels nœuds les réplicas sont-ils crées ?

10. Par défaut, tous les services sont crées en mode répliqués. Le mode global permet d'exécuter exactement un conteneur par nœud. On va appliquer ce mode sur le service rng.

- Malheureusement, ce mode ne peut pas être activé/désactivé pour un service existant. Donc, Il faut commencer par supprimer le service rng existant.

  docker service rm rng

```
ubuntu@formation1:~/docker-swarm-lab$ docker service ls
ID             NAME      MODE          REPLICAS   IMAGE                          PORTS
z9cj853mfh5s   hasher    replicated    1/1        brahimhamdi/hasher:latest
bk0rte0ih61v   redis     replicated    1/1        redis:latest
f7eehf22dkej   rng       replicated    1/1        brahimhamdi/rng:latest
7gsm38zlg5zk   webui     replicated    1/1        brahimhamdi/webui:latest       *:8000->80/tcp
m6zhv5h7j119   worker    replicated    10/10      brahimhamdi/worker:latest
ubuntu@formation1:~/docker-swarm-lab$
ubuntu@formation1:~/docker-swarm-lab$ docker service rm rng
rng
ubuntu@formation1:~/docker-swarm-lab$
ubuntu@formation1:~/docker-swarm-lab$
```

- Puis le recréer avec le scheduling global:

  docker service create --name rng --mode global --network dockercoins-net brahimhamdi/rng

```
ubuntu@formation1:~/docker-swarm-lab$ docker service create --name rng --mode global --network dockercoins-net brahimhamdi/rng
1q7k2881hfoejf2a1tue4nnzn
overall progress: 3 out of 3 tasks
qoycpsbpkf1o: running   [==================================================>]
7j8yc25j5g21: running   [==================================================>]
vvvjjhzemnh2: running   [==================================================>]
verify: Service converged
ubuntu@formation1:~/docker-swarm-lab$
ubuntu@formation1:~/docker-swarm-lab$ docker service ls
ID             NAME      MODE          REPLICAS   IMAGE                          PORTS
z9cj853mfh5s   hasher    replicated    1/1        brahimhamdi/hasher:latest
bk0rte0ih61v   redis     replicated    1/1        redis:latest
1q7k2881hfoe   rng       global        3/3        brahimhamdi/rng:latest
7gsm38zlg5zk   webui     replicated    1/1        brahimhamdi/webui:latest       *:8000->80/tcp
m6zhv5h7j119   worker    replicated    10/10      brahimhamdi/worker:latest
ubuntu@formation1:~/docker-swarm-lab$
```

- Vérifiez qu'il y a exactement une seule instance de rng exécutée sur chaque nœud

  docker service ps rng

```
ubuntu@formation1:~/docker-swarm-lab$ docker service ps rng
ID             NAME                          IMAGE                    NODE         DESIRED STATE   CURRENT STATE                ERROR     PO
RTS
o0irkqyzm4y5   rng.7j8yc25j5g211y2iqvsylkmhz   brahimhamdi/rng:latest   formation1   Running         Running about a minute ago
9kr4a00axjvg   rng.qoycpsbpkf1oqbjv5gefmc6rx   brahimhamdi/rng:latest   worker2      Running         Running about a minute ago
w0gufle7cvwk   rng.vvvjjhzemnh2dn0fxpw2zjwgr   brahimhamdi/rng:latest   worker1      Running         Running about a minute ago
ubuntu@formation1:~/docker-swarm-lab$
ubuntu@formation1:~/docker-swarm-lab$
```