

Docker Swarm

Tout d'abord je mets à jour mes VM'S via les commandes :

```
"sudo apt-get update et upgrade"
```

Après ça, je tape mes commandes pour l'installation de mon dépôt docker :

```
"sudo apt-get install -y apt-transport-https ca-certificates curl gnupg lsb-release "
```

```
"curl -fsSL https://download.docker.com/linux/debian/gpg | sudo gpg --dearmor -o /usr/share/keyrings/docker-archive-keyring.gpg "
```

```
"echo "deb [arch=amd64 signed-by=/usr/share/keyrings/docker-archive-keyring.gpg] https://download.docker.com/linux/debian $(lsb_release -cs) stable" | sudo tee /etc/apt/sources.list.d/docker.list > /dev/null"
```

On installe ensuite docker engine :

```
"sudo apt-get install -y docker-ce docker-ce-cli containerd.io"
```

Pour s'assurer que tout est bien installé sur nos 3 VM's, on utilise :

```
"docker --version"
```

On débute les procédures docker...

Je tape la command suivante pour initialiser mon cluster chez mon master :

```
"sudo docker swarm init --advertise-addr 172.16.65.138"
```

Maintenant mon master passe l'est vraiment et passe en mode "manager".

Je note le retour que j'ai après la commande précédente :

```
docker swarm join --token  
SWMTKN-1-1rmtr8x6j3b2co0eb4f8fkp9afv5uqln2m78plztgravbz7t5v-8rsa36r2u6  
jb87uwu0i4ffpl1 172.16.65.138:2377
```

Je passe maintenant sur ma machine slave (worker)...

Je tape la commande suivante pour lier mon slave à mon master en mettant le token obtenue dernièrement :

`"sudo docker swarm join --token <TOKEN> 172.16.65.138:2377"`

Je modifie "<token>" par :

`SWMTKN-1-1rmtr8x6j3b2co0eb4f8fkp9afv5uqln2m78plztgravbz7t5v-8rsa36r2u6jb87uwu0i4ffpl1`

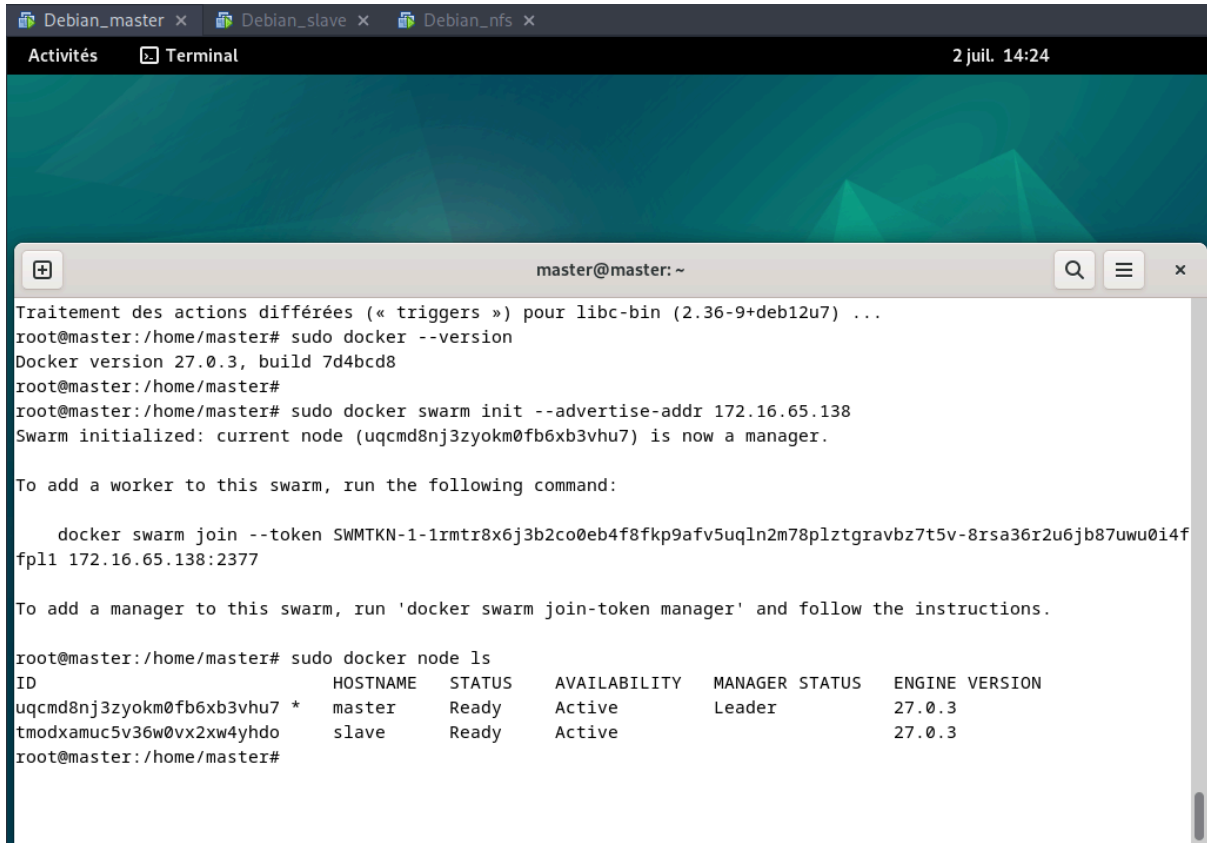
Ce qui donne :

`"sudo docker swarm join --token
SWMTKN-1-1rmtr8x6j3b2co0eb4f8fkp9afv5uqln2m78plztgravbz7t5v-8rsa36r2u6jb87uwu0i4ffpl1 172.16.65.138:2377"`

On obtient en retour : "This node joined a swarm as worker"

Pour s'assurer que notre slave à bien rejoint le noeud, sur notre machine maître on tape : "sudo docker node ls"

Reçu :



The screenshot shows a terminal window with the following content:

```
Traitement des actions différées (« triggers ») pour libc-bin (2.36-9+deb12u7) ...
root@master:/home/master# sudo docker --version
Docker version 27.0.3, build 7d4bcd8
root@master:/home/master#
root@master:/home/master# sudo docker swarm init --advertise-addr 172.16.65.138
Swarm initialized: current node (uqcmd8nj3zyokm0fb6xb3vhu7) is now a manager.

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

    docker swarm join --token SWMTKN-1-1rmtr8x6j3b2co0eb4f8fkp9afv5uqln2m78plztgravbz7t5v-8rsa36r2u6jb87uwu0i4ffpl1 172.16.65.138:2377

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

root@master:/home/master# sudo docker node ls
ID                                HOSTNAME    STATUS    AVAILABILITY    MANAGER STATUS    ENGINE VERSION
uqcmd8nj3zyokm0fb6xb3vhu7 *      master     Ready     Active           Leader             27.0.3
tmodxamuc5v36w0vx2xw4yhdo      slave      Ready     Active           -                  27.0.3
root@master:/home/master#
```

À présent que nous sommes “bon” niveau docker, passons à notre serveur NFS.

On installe d’abord notre serveur NFS sur la VM avec la commande :

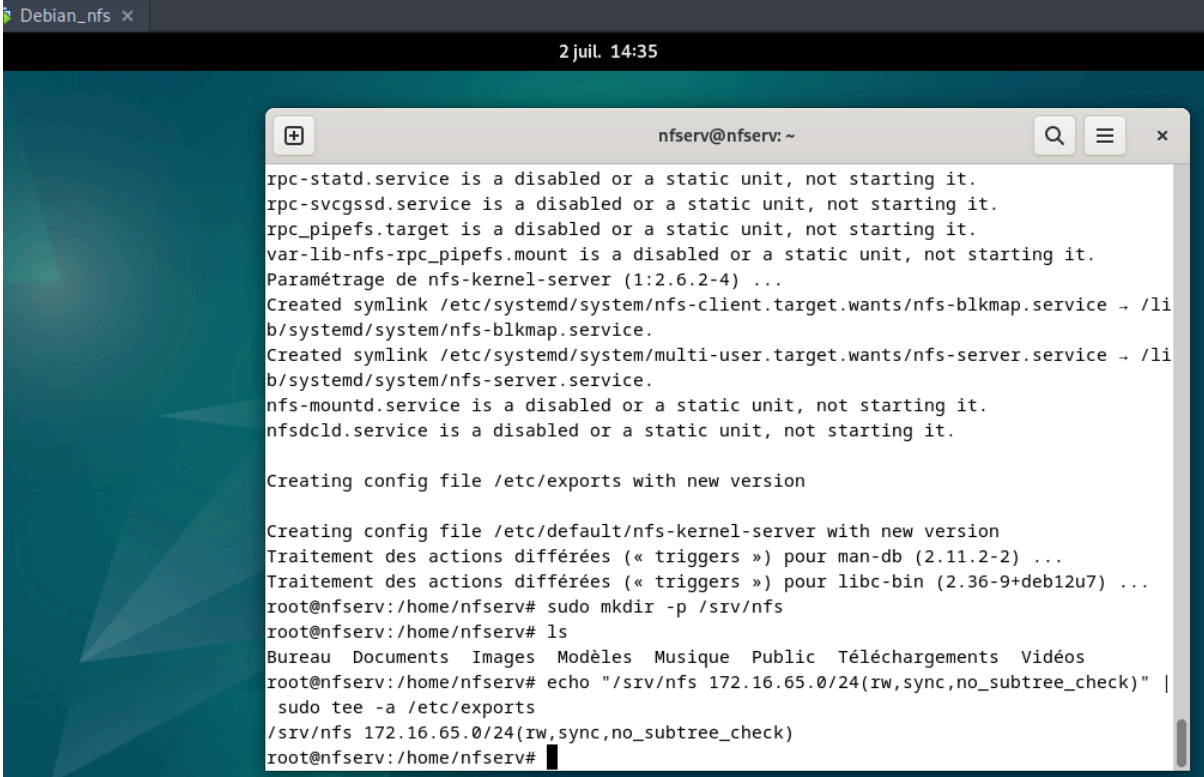
```
“sudo apt-get install -y nfs-kernel-server”
```

Pour le côté partage de données on crée un dossier où seront stockés et partagés nos données avec la commande :

```
“sudo mkdir -p /srv/nfs”
```

On ajoute le partage du fichier au répertoire `/etc/exports` :

```
“echo “/srv/nfs 172.16.65.0/24(rw,sync,no_subtree_check)” | sudo tee -a /etc/exports”
```



The screenshot shows a terminal window titled "Debian_nfs" with a timestamp of "2 juil. 14:35". The terminal output displays the installation of `nfs-kernel-server` (version 1:2.6.2-4). It lists several disabled services: `rpc-statd.service`, `rpc-svcgssd.service`, `rpc_pipefs.target`, `var-lib-nfs-rpc_pipefs.mount`, `nfs-mountd.service`, and `nfsdclld.service`. It then shows the creation of symlinks for `nfs-client.target` and `multi-user.target`. The configuration file `/etc/exports` is updated with the new version. The terminal also shows the execution of `sudo mkdir -p /srv/nfs` and `ls` command, listing the contents of the `/home/nfserv` directory: `Bureau`, `Documents`, `Images`, `Modèles`, `Musique`, `Public`, `Téléchargements`, and `Vidéos`. Finally, the `echo` command is used to append the export rule to `/etc/exports`, and the output is shown as `/srv/nfs 172.16.65.0/24(rw,sync,no_subtree_check)`.

```
rpc-statd.service is a disabled or a static unit, not starting it.
rpc-svcgssd.service is a disabled or a static unit, not starting it.
rpc_pipefs.target is a disabled or a static unit, not starting it.
var-lib-nfs-rpc_pipefs.mount is a disabled or a static unit, not starting it.
Paramétrage de nfs-kernel-server (1:2.6.2-4) ...
Created symlink /etc/systemd/system/nfs-client.target.wants/nfs-blkmap.service → /li
b/systemd/system/nfs-blkmap.service.
Created symlink /etc/systemd/system/multi-user.target.wants/nfs-server.service → /li
b/systemd/system/nfs-server.service.
nfs-mountd.service is a disabled or a static unit, not starting it.
nfsdclld.service is a disabled or a static unit, not starting it.

Creating config file /etc/exports with new version

Creating config file /etc/default/nfs-kernel-server with new version
Traitement des actions différées (« triggers ») pour man-db (2.11.2-2) ...
Traitement des actions différées (« triggers ») pour libc-bin (2.36-9+deb12u7) ...
root@nfserv:/home/nfserv# sudo mkdir -p /srv/nfs
root@nfserv:/home/nfserv# ls
Bureau Documents Images Modèles Musique Public Téléchargements Vidéos
root@nfserv:/home/nfserv# echo "/srv/nfs 172.16.65.0/24(rw,sync,no_subtree_check)" |
sudo tee -a /etc/exports
/srv/nfs 172.16.65.0/24(rw,sync,no_subtree_check)
root@nfserv:/home/nfserv#
```

On redémarre notre serveur par principe :

```
“sudo systemctl restart nfs-kernel-server”
```

Retournons sur nos machines master et slave pour installer le client NFS afin d’avoir accès aux données...

On tape la commande suivante :

```
“sudo apt-get install -y nfs-common”
```

Le client installé, on monte le partage NFS pour qu'on puisse accéder aux données partagées.

Création du répertoire où nos données seront stockées (master + slave) :

```
"sudo mkdir -p /mnt/nfs"
```

Puis on monte le partage NFS : (addr du serveur NFS)

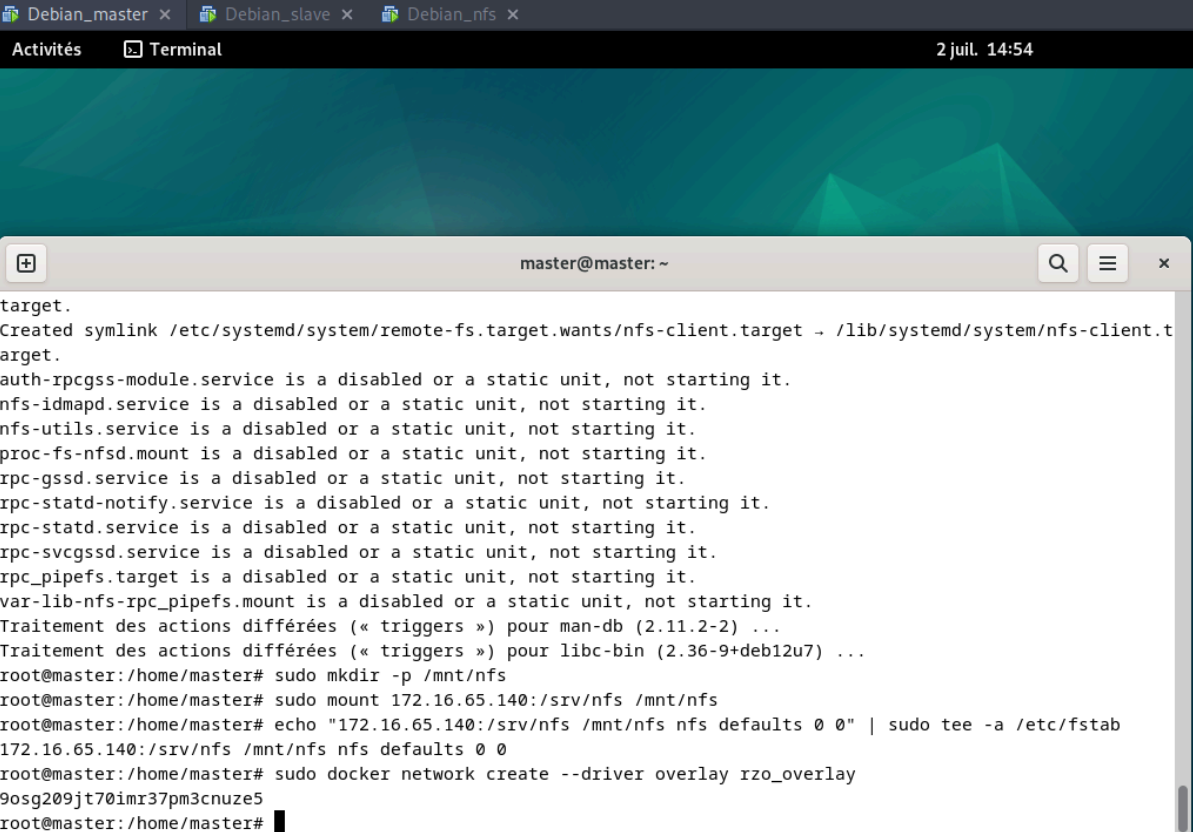
```
"sudo mount 172.16.65.140:/srv/nfs /mnt/nfs"
```

J'ajoute une entrée dans mon fichier fstab pour que le montage se fasse automatiquement à chaque démarrage : (master + slave)

```
"echo "172.16.65.140:/srv/nfs /mnt/nfs nfs defaults 0 0" | sudo tee -a /etc/fstab"
```

On va créer un réseau overlay sur notre vm master via la commande :

```
"sudo docker network create --driver overlay rzo_overlay"
```

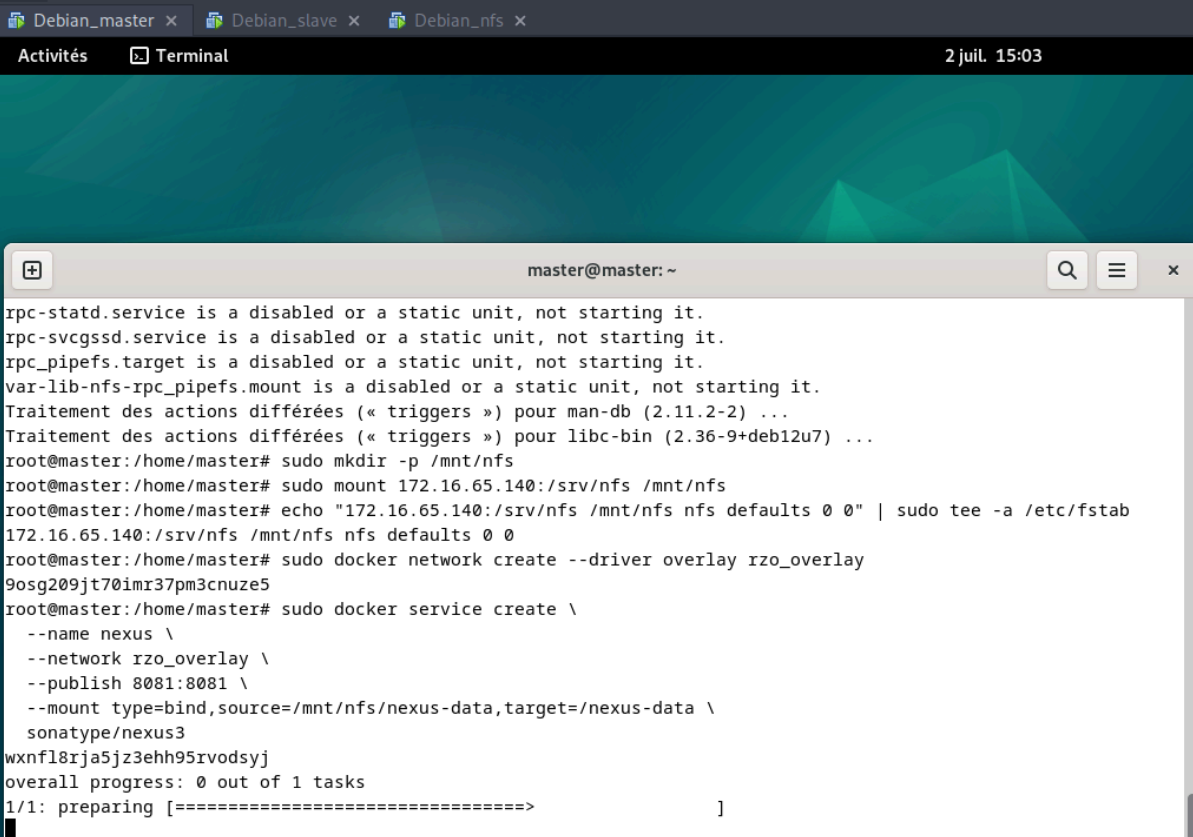


The screenshot shows a terminal window with three tabs: 'Debian_master', 'Debian_slave', and 'Debian_nfs'. The active tab is 'Debian_master'. The terminal output shows the following commands and their results:

```
target.  
Created symlink /etc/systemd/system/remote-fs.target.wants/nfs-client.target → /lib/systemd/system/nfs-client.t  
target.  
auth-rpcgss-module.service is a disabled or a static unit, not starting it.  
nfs-idmapd.service is a disabled or a static unit, not starting it.  
nfs-utils.service is a disabled or a static unit, not starting it.  
proc-fs-nfsd.mount is a disabled or a static unit, not starting it.  
rpc-gssd.service is a disabled or a static unit, not starting it.  
rpc-statd-notify.service is a disabled or a static unit, not starting it.  
rpc-statd.service is a disabled or a static unit, not starting it.  
rpc-svcgssd.service is a disabled or a static unit, not starting it.  
rpc_pipefs.target is a disabled or a static unit, not starting it.  
var-lib-nfs-rpc_pipefs.mount is a disabled or a static unit, not starting it.  
Traitement des actions différées (« triggers ») pour man-db (2.11.2-2) ...  
Traitement des actions différées (« triggers ») pour libc-bin (2.36-9+deb12u7) ...  
root@master:/home/master# sudo mkdir -p /mnt/nfs  
root@master:/home/master# sudo mount 172.16.65.140:/srv/nfs /mnt/nfs  
root@master:/home/master# echo "172.16.65.140:/srv/nfs /mnt/nfs nfs defaults 0 0" | sudo tee -a /etc/fstab  
172.16.65.140:/srv/nfs /mnt/nfs nfs defaults 0 0  
root@master:/home/master# sudo docker network create --driver overlay rzo_overlay  
9osg209jt70imr37pm3cnuze5  
root@master:/home/master#
```

Après ça, on s'occupe du repository local (nexus), sur notre noeud master on tape la commande suivante :

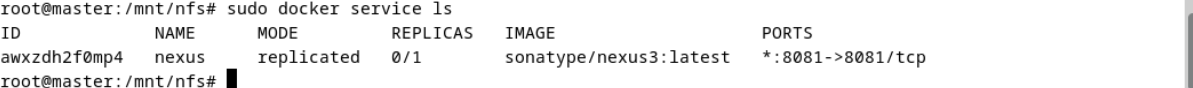
```
"sudo docker service create \
  --name nexus \
  --network rzo_overlay \
  --publish 8081:8081 \
  --mount type=bind,source=/mnt/nfs/nexus-data,target=/nexus-data \
  sonatype/nexus3"
```



The screenshot shows a terminal window with three tabs: 'Debian_master', 'Debian_slave', and 'Debian_nfs'. The active tab is 'Debian_master'. The terminal output shows the following commands and their results:

```
rpc-statd.service is a disabled or a static unit, not starting it.
rpc-svcgssd.service is a disabled or a static unit, not starting it.
rpc_pipefs.target is a disabled or a static unit, not starting it.
var-lib-nfs-rpc_pipefs.mount is a disabled or a static unit, not starting it.
Traitement des actions différées (« triggers ») pour man-db (2.11.2-2) ...
Traitement des actions différées (« triggers ») pour libc-bin (2.36-9+deb12u7) ...
root@master:/home/master# sudo mkdir -p /mnt/nfs
root@master:/home/master# sudo mount 172.16.65.140:/srv/nfs /mnt/nfs
root@master:/home/master# echo "172.16.65.140:/srv/nfs /mnt/nfs nfs defaults 0 0" | sudo tee -a /etc/fstab
172.16.65.140:/srv/nfs /mnt/nfs nfs defaults 0 0
root@master:/home/master# sudo docker network create --driver overlay rzo_overlay
9osg209jt70imr37pm3cnuze5
root@master:/home/master# sudo docker service create \
  --name nexus \
  --network rzo_overlay \
  --publish 8081:8081 \
  --mount type=bind,source=/mnt/nfs/nexus-data,target=/nexus-data \
  sonatype/nexus3
wxnfl8rja5jz3ehh95rvodsyj
overall progress: 0 out of 1 tasks
1/1: preparing [=====>] ]
```

Nexus créer :



The screenshot shows a terminal window with the following output:

```
root@master:/mnt/nfs# sudo docker service ls
ID                NAME      MODE      REPLICAS  IMAGE                                  PORTS
awxzdh2f0mp4     nexus    replicated 0/1        sonatype/nexus3:latest              *:8081->8081/tcp
root@master:/mnt/nfs#
```

Maintenant créons tous nos services dockers :

-MariaDB

```
sudo docker service create \
  --name mariadb \
  --network rzo_overlay \
  --mount type=volume,source=mariadb_data,target=/var/lib/mysql \
  --env MYSQL_ROOT_PASSWORD=my-secret-pw \
  mariadb:latest
```

-PHP

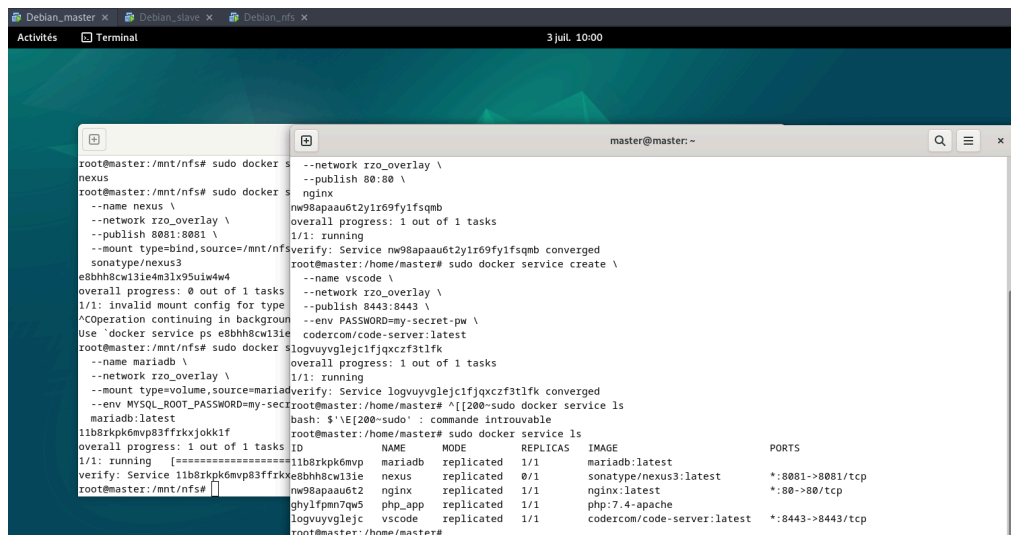
```
sudo docker service create \
  --name php_app \
  --network rzo_overlay \
  php:7.4-apache
```

-Nginx

```
sudo docker service create \
  --name nginx \
  --network rzo_overlay \
  --publish 80:80 \
  nginx
```

-VScode

```
sudo docker service create \
  --name vscode \
  --network rzo_overlay \
  --publish 8443:8443 \
  --env PASSWORD=my-secret-pw \
  codercom/code-server:latest
```



The screenshot shows a terminal window with two panes. The left pane shows the commands used to create the services, and the right pane shows the output of the commands, including the status of the services and a table of the created services.

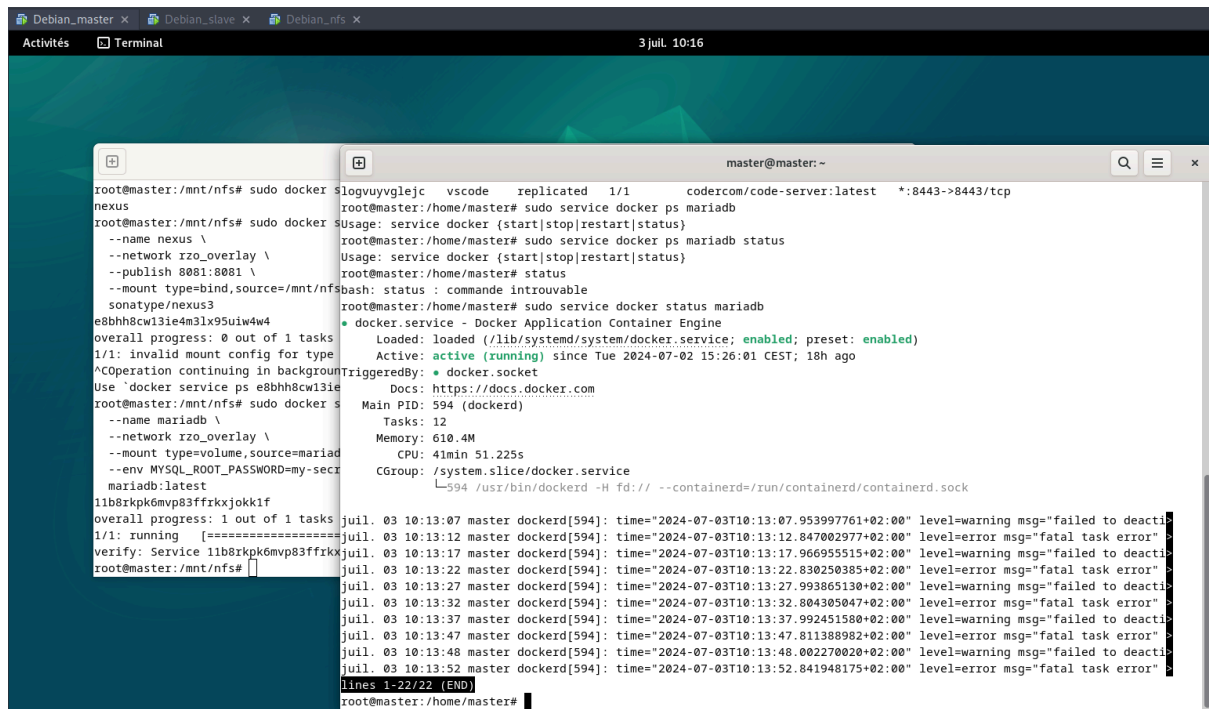
```
root@master:/mnt/nfs# sudo docker service create --name nexus \
root@master:/mnt/nfs# sudo docker service create --name nexus \
--network rzo_overlay \
--publish 80:80 \
nginx
root@master:/mnt/nfs# sudo docker service create --name nexus \
--network rzo_overlay \
--publish 80:80 \
nginx
root@master:/mnt/nfs# sudo docker service create --name mariadb \
--network rzo_overlay \
--mount type=volume,source=mariadb_data,target=/var/lib/mysql \
--env MYSQL_ROOT_PASSWORD=my-secret-pw \
mariadb:latest
root@master:/mnt/nfs# sudo docker service create --name php_app \
--network rzo_overlay \
php:7.4-apache
root@master:/mnt/nfs# sudo docker service create --name vscode \
--network rzo_overlay \
--publish 8443:8443 \
--env PASSWORD=my-secret-pw \
codercom/code-server:latest
```

The right pane shows the output of the commands, including the status of the services and a table of the created services.

```
root@master:/home/master# sudo docker service ls
ID                NAME      MODE      REPLICAS  IMAGE                                  PORTS
11b8rpk6mvp83frrkxjokk1f  mariadb  replicated  1/1        mariadb:latest
e8bhh8cw13ie4m3lx95uiw4w4  nexus    replicated  0/1        sonatype/nexus3:latest              *:8081->8081/tcp
logvuyvglejcljfgkczf3tlfk  nginx    replicated  1/1        nginx:latest                         *:80->80/tcp
logvuyvglejcljfgkczf3tlfk  php_app  replicated  1/1        php:7.4-apache
logvuyvglejcljfgkczf3tlfk  vscode   replicated  1/1        codercom/code-server:latest         *:8443->8443/tcp
```

Via la commande “*sudo docker service ls*”, je liste tous mes services installés pour m’assurer que tout est là.

Pour voir leur statut je fais “*sudo service docker status nom_service*”.



```
Debian_master x Debian_slave x Debian_nfs x
3 juil. 10:16

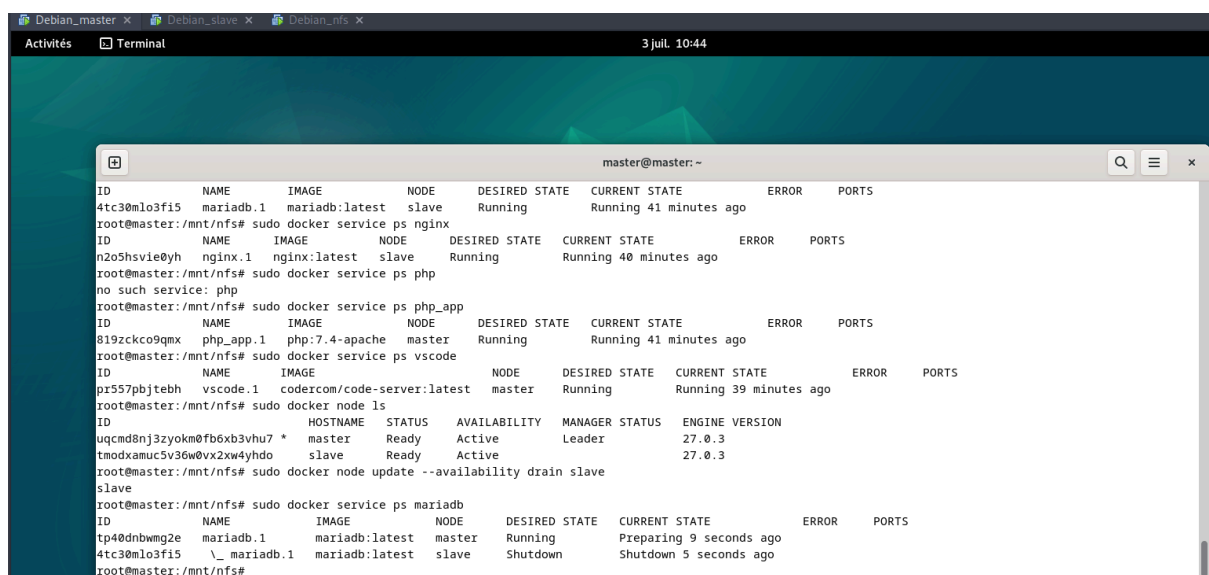
root@master:/mnt/nfs# sudo docker service ls
nexus
root@master:/mnt/nfs# sudo docker service docker ps mariadb
--name nexus \
--network zzo_overlay \
--publish 8081:8081 \
--mount type=bind,source=/mnt/nfs,target=/nexus
overall progress: 0 out of 1 tasks
1/1: invalid mount config for type
^Coperation continuing in background
Use 'docker service ps e8bhh8cw13ie' to view details
root@master:/mnt/nfs# sudo docker service status mariadb
--name mariadb \
--network zzo_overlay \
--mount type=volume,source=mariadb,target=/var/lib/mysql \
--env MYSQL_ROOT_PASSWORD=secret
mariadb:latest
11b8rpk6mvp83ffrkxjokk1f
overall progress: 1 out of 1 tasks
1/1: running [=====]
verify: Service 11b8rpk6mvp83ffrkxjokk1f is running
root@master:/mnt/nfs#

logvuyvglej vscod replicated 1/1 codercom/code-server:latest *:8443->8443/tcp
root@master:/home/master# sudo service docker ps mariadb
Usage: service docker {start|stop|restart|status}
root@master:/home/master# sudo service docker ps mariadb status
Usage: service docker {start|stop|restart|status}
root@master:/home/master# status
bash: status: commande introuvable
root@master:/home/master# sudo service docker status mariadb
• docker.service - Docker Application Container Engine
Loaded: loaded (/lib/systemd/system/docker.service; enabled; preset: enabled)
Active: active (running) since Tue 2024-07-02 15:26:01 CEST; 18h ago
TriggeredBy: • docker.socket
Docs: https://docs.docker.com
Main PID: 594 (dockerd)
Tasks: 12
Memory: 610.4M
CPU: 41min 51.225s
CGroup: /system.slice/docker.service
└─594 /usr/bin/dockerd -H fd:// --containerd=/run/containerd/containerd.sock

juil. 03 10:13:07 master dockerd[594]: time="2024-07-03T10:13:07.953997761+02:00" level=warning msg="failed to deactivate"
juil. 03 10:13:12 master dockerd[594]: time="2024-07-03T10:13:12.847002977+02:00" level=error msg="fatal task error"
juil. 03 10:13:17 master dockerd[594]: time="2024-07-03T10:13:17.966955515+02:00" level=warning msg="failed to deactivate"
juil. 03 10:13:22 master dockerd[594]: time="2024-07-03T10:13:22.830250385+02:00" level=error msg="fatal task error"
juil. 03 10:13:27 master dockerd[594]: time="2024-07-03T10:13:27.993865138+02:00" level=warning msg="failed to deactivate"
juil. 03 10:13:32 master dockerd[594]: time="2024-07-03T10:13:32.804305047+02:00" level=error msg="fatal task error"
juil. 03 10:13:37 master dockerd[594]: time="2024-07-03T10:13:37.992451580+02:00" level=warning msg="failed to deactivate"
juil. 03 10:13:47 master dockerd[594]: time="2024-07-03T10:13:47.811388982+02:00" level=error msg="fatal task error"
juil. 03 10:13:48 master dockerd[594]: time="2024-07-03T10:13:48.002270020+02:00" level=warning msg="failed to deactivate"
juil. 03 10:13:52 master dockerd[594]: time="2024-07-03T10:13:52.841948175+02:00" level=error msg="fatal task error"
lines 1-22/22 (END)
root@master:/home/master#
```

Je simule une panne sur mon slave en le drainant :
“*sudo docker node update --availability drain slave*”

Pour vérifier que les services tournent encore :
“*sudo docker service ps nom_service*”



```
Debian_master x Debian_slave x Debian_nfs x
3 juil. 10:44

ID NAME IMAGE NODE DESIRED STATE CURRENT STATE ERROR PORTS
4tc30mlo3fi5 mariadb:latest slave Running Running 41 minutes ago
root@master:/mnt/nfs# sudo docker service ps nginx
ID NAME IMAGE NODE DESIRED STATE CURRENT STATE ERROR PORTS
n2o5hsvie0yh nginx:latest slave Running Running 40 minutes ago
root@master:/mnt/nfs# sudo docker service ps php
no such service: php
root@master:/mnt/nfs# sudo docker service ps php_app
ID NAME IMAGE NODE DESIRED STATE CURRENT STATE ERROR PORTS
819zckco9qmx php:7.4-apache master Running Running 41 minutes ago
root@master:/mnt/nfs# sudo docker service ps vscod
ID NAME IMAGE NODE DESIRED STATE CURRENT STATE ERROR PORTS
pr557pbjtebh vscod.1 codercom/code-server:latest master Running Running 39 minutes ago
root@master:/mnt/nfs# sudo docker node ls
ID HOSTNAME STATUS AVAILABILITY MANAGER STATUS ENGINE VERSION
uqcm08n3zyokm@fb6xb3vhu7 * master Ready Active Leader 27.0.3
tmodxamuc5v36w0vx2xw4yhdo slave Ready Active 27.0.3
root@master:/mnt/nfs# sudo docker node update --availability drain slave
slave
root@master:/mnt/nfs# sudo docker service ps mariadb
ID NAME IMAGE NODE DESIRED STATE CURRENT STATE ERROR PORTS
tp40dnbnwmg2e mariadb:latest master Running Preparing 9 seconds ago
4tc30mlo3fi5 \_ mariadb:latest slave Shutdown Shutdown 5 seconds ago
root@master:/mnt/nfs#
```

Je n’oublie pas de réactiver mon slave “*sudo docker node update --availability active slave*”

Pour tester la résilience des données, on crée une base de données sur mariadb, on redémarre notre service et on vérifie que nos données sont toujours disponibles et accessibles.