Diaby Mohamed

1. Créez un réseau Docker nommé "monitoring_network" pour simuler l'environnement de production.

docker network create monitoring_network

2. Créez un volume Docker nommé "shared_data" pour le partage sécurisé de fichiers entre les services.

Docker volume create shared_data

```
C:\Users\md>docker network create monitoring_network
6f8bfc51997a8dadfabbbc4a4ed21de3a7756d89766bfb06d8594508fbabd66d

C:\Users\md>docker network ls
NETWORK ID     NAME                 DRIVER    SCOPE
6f6564b91301   bridge               bridge    local
bd1d53b1031e   host                 host      local
6f8bfc51997a   monitoring_network   bridge    local
08c0d243caba   none                 null      local

C:\Users\md>docker volume create shared_data
shared_data
```

3. Créez deux conteneurs Docker en mode interactif, nommés "file-generator" et "file-monitor", connectés au réseau "monitoring_network" et montant le volume "shared_data".

4. Dans chaque conteneur, installez les outils nécessaires (sudo, cron, nano) via apt-get install pour simuler un environnement de production minimal.

Docker run –-name file-generator -it –-network monitoring_network –v shared_data/:shared_data ubuntu

Apt-get update

Apt-get install sudo

Apt-get install cron

Apt-get install nano

Docker run –-name file-monitor -it –-network monitoring_network –v shared_data/:shared_data ubuntu

Apt-get update

Apt-get install sudo

Apt-get install cron

Apt-get install nano

```
C:\Users\md>docker run --name file-generator -it --network monitoring_network -v shared_data:/shared_data ubuntu
root@56c960bf4e36:/#
```

5. Vérifiez que les deux conteneurs peuvent communiquer entre eux en utilisant la commande ping.

Pour re-acceder a un des terminal bash : docker exec -it file-monitor bash

apt-get install iputils-ping

Ping file-generator

```
root@56c960bf4e36:/# apt-get install iputils-ping
```

```
root@56c960bf4e36:/# ping file-monitor
PING file-monitor (172.18.0.3) 56(84) bytes of data.
64 bytes from file-monitor.monitoring_network (172.18.0.3): icmp_seq=1 ttl=64 time=0.060 ms
64 bytes from file-monitor.monitoring_network (172.18.0.3): icmp_seq=2 ttl=64 time=0.034 ms
64 bytes from file-monitor.monitoring_network (172.18.0.3): icmp_seq=3 ttl=64 time=0.037 ms
64 bytes from file-monitor.monitoring_network (172.18.0.3): icmp_seq=4 ttl=64 time=0.039 ms
```

6. Configurez les tâches suivantes :

a. Dans le conteneur "file-generator" (cont1) :

- Créez un script shell qui liste les fichiers du conteneur et écrit cette liste dans un fichier "file_inventory.txt" sur le volume partagé.

Ls / > /shared_data/file_inventory.txt

```
ls /  > /shared_data/file_inventory.txt
```

- Configurez une tâche cron pour exécuter ce script toutes les 5 minutes.

Crontab –e

*/5 * * * * /fichier_list.sh

```
# Edit this file to introduce tasks to be run by cron.
#
# Each task to run has to be defined through a single line
# indicating with different fields when the task will be run
# and what command to run for the task
#
# To define the time you can provide concrete values for
# minute (m), hour (h), day of month (dom), month (mon),
# and day of week (dow) or use '*' in these fields (for 'any').
#
# Notice that tasks will be started based on the cron's system
# daemon's notion of time and timezones.
#
# Output of the crontab jobs (including errors) is sent through
# email to the user the crontab file belongs to (unless redirected).
#
# For example, you can run a backup of all your user accounts
# at 5 a.m every week with:
# 0 5 * * 1 tar -zcf /var/backups/home.tgz /home/
#
# For more information see the manual pages of crontab(5) and cron(8)
#
# m h  dom mon dow   command
*/5 * * * * /fichier_list.sh
```

```
:\Users\ibrah>docker exec -it file-generator bash
oot@bf46ba9c6da7:/# nano /fichier_list.sh
oot@bf46ba9c6da7:/# chmod +x /fichier_list.sh
oot@bf46ba9c6da7:/# crontab -e
o crontab for root - using an empty one
rontab: installing new crontab
oot@bf46ba9c6da7:/# crontab -e
o modification made
oot@bf46ba9c6da7:/# crontab -e
o modification made
oot@bf46ba9c6da7:/#
```

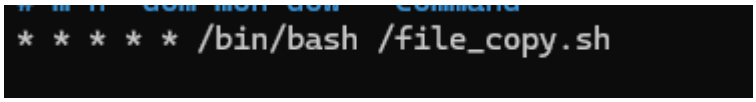
b. Dans le conteneur "file-monitor" (cont2) :

- Créez un script shell qui :

1) Vérifie l'existence du fichier "file_inventory.txt" toutes les 2 minutes.

Crontab –e

*/2 * * * * /file_copy.sh

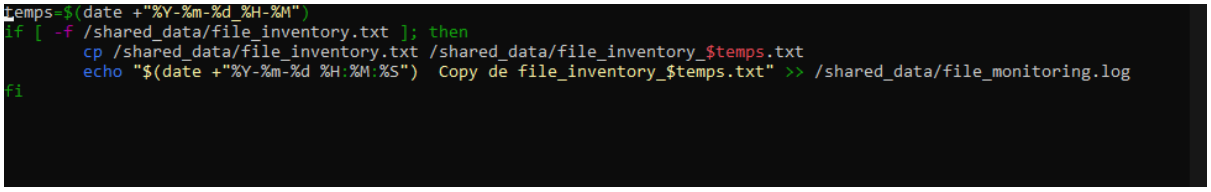


2) S'il existe, crée une copie datée (par exemple, "file_inventory_2024-10-16_14-30.txt").

```
temps=$(date +"%Y-%m-%d_%H-%M")
if [ -f /shared_data/file_inventory.txt ]; then
   cp /shared_data/file_inventory.txt /shared_data/file_inventory_$temps.txt
fi
```

3) Ajoute une entrée dans un fichier de log "file_monitoring.log" avec la date, l'heure et le nom du fichier copié.

```
 echo "$(date +"%Y-%m-%d %H:%M:%S")  Copy de file_inventory_$Timed.txt" >> /shared_data/file_monitoring.log
```



```
temps=$(date +"%Y-%m-%d_%H-%M")
if [ -f /shared_data/file_inventory.txt ]; then
        cp /shared_data/file_inventory.txt /shared_data/file_inventory_$temps.txt
        echo "$(date +"%Y-%m-%d %H:%M:%S")  Copy de file_inventory_$temps.txt" >> /shared_data/file_monitoring.log
fi
```

7. Testez le système en laissant tourner les conteneurs pendant au moins 15 minutes pour vérifier que les fichiers sont correctement générés, copiés et loggés.