Exam - Cloud Computing

16 octobre 2019 - Documents autorisés

1 Questions

Q.1) - Quel est l'intéret d'utiliser un orchestrateur comme swarm ou kubernetes, par rapport à de simples dockers ?

Q.2) - Lorsqu'une machine virtuelle est crée dans votre hyperviseur, son disque doit d'abord être copié depuis un template. Cette copie est très rapide (quelques secondes) alors que le disque est assez gros. Comment cela est-il possible ? Est-ce grâce à la puissance des disque ? À la méthde de copie ?

Q.3) - Votre docker ne démarre pas, quel est la première commande à taper pour avoir des information sur ce qui se passe ?

2 Exercice

Vous devez mettre en place un site web wordpress via une image docker dont la documentation est dans la suite du sujet. Ce dernier doit utiliser une base de données mysql existante :

— La base est sur le serveur `base.fai.fr`
— Le port du serveur de base de données est 23386
— La base de donnée est `BASEWP`
— L'utilisateur de la base est `TOTO` et son mot de passe `secret`
— Le site doit être accessible sur le port 9080 de la machine hôte
— Le répertoire du site wordpress `/var/www/html/wp/uploads/` doit être partagé avec l'hôte pour des besoins de sauvegarde.
— Il faut que le site utilise la version de php7
— Le docker doit avoir le nom `SiteWP`

Q.4) - Donnez la commande docker run permettant de créer un le docker.

On souhaite packager le site avec un docker mysql via un fichier `docker-compose.xml`. Contrairement à ce qui est demandé dans la question précédante, la base de données sera donc sur un docker hébergé par la même machine. L'utilisateur, le mot de passe et le nom de la basse de données ne doivent pas changer. De plus, il faut que le répertoire `/var/lib/mysql` de la base soit partagé avec l'hôte afin d'être sauvegardé.

Q.5) - Donnez le contenu du fichier docker compose necessaire au lancement des 2 dockers.

A Documentation sur le docker WordPress

A.1 Supported tags and respective Dockerfile links

— 4.9.6-php5.6-apache, 4.9-php5.6-apache, 4-php5.6-apache, php5.6-apache, 4.9.6-php5.6, 4.9-php5.6, 4-php5.6, php5.6 (php5.6/apache/Dockerfile)
— 4.9.6-php5.6-fpm, 4.9-php5.6-fpm, 4-php5.6-fpm, php5.6-fpm (php5.6/fpm/Dockerfile)
— 4.9.6-php7.0-fpm, 4.9-php7.0-fpm, 4-php7.0-fpm, php7.0-fpm (php7.0/fpm/Dockerfile)
— ...

A.2 Quick reference

Where to get help :
the Docker Community Forums, the Docker Community Slack, or Stack Overflow
Where to file issues :
`https://github.com/docker-library/wordpress/issues`
Maintained by :
the Docker Community
Supported architectures : (more info)
amd64, arm32v5, arm32v6, arm32v7, arm64v8, i386, ppc64le, s390x
Published image artifact details :
repo-info repo's repos/wordpress/ directory (history) (image metadata, transfer size, etc)
Image updates :
official-images PRs with label library/wordpress official-images repo's library/wordpress file (history)
Source of this description :
docs repo's wordpress/ directory (history)
Supported Docker versions :
the latest release (down to 1.6 on a best-effort basis)

A.3 What is WordPress ?

WordPress is a free and open source blogging tool and a content management system (CMS) based on PHP and MySQL, which runs on a web hosting service. Features include a plugin architecture and a template system. WordPress is used by more than 22.0

A.4 How to use this image

```
$ docker run --name some-wordpress --link some-mysql:mysql -d wordpress
```

The following environment variables are also honored for configuring your WordPress instance :

—
— `WORDPRESS_DB_HOST=...` (defaults to the IP and port of the linked mysql container use the value IP :PORT)
— `WORDPRESS_DB_USER=...` (defaults to "root")
— `WORDPRESS_DB_PASSWORD=...` (defaults to the value of the `MYSQL_ROOT_PASSWORD` environment variable from the linked mysql container)
— `WORDPRESS_DB_NAME=...` (defaults to "wordpress")
— `WORDPRESS_TABLE_PREFIX=...` (defaults to "", only set this when you need to override the default table prefix in `wp-config.php`)
— ...
— `WORDPRESS_DEBUG=1` (defaults to disabled, non-empty value will enable `WP_DEBUG` in wp-config.php)

If the `WORDPRESS_DB_NAME` specified does not already exist on the given MySQL server, it will be created automatically upon startup of the wordpress container, provided that the `WORDPRESS_DB_USER` specified has the necessary permissions to create it.

If you'd like to be able to access the instance from the host without the container's IP, standard port mappings can be used :

```
$ docker run --name some-wordpress --link some-mysql:mysql -p 8080:80 -d wordpress
```

Then, access it via `http://localhost:8080` or `http://host-ip:8080` in a browser.

B Rappel des variables d'environnments utilisées par le docker mysql

— `MYSQL_RANDOM_ROOT_PASSWORD` : When this variable is true (which is its default state, unless `MYSQL_ROOT_PASSWORD` is set or `MYSQL_ALLOW_EMPTY_PASSWORD` is set to true), a random password for the server's root user is generated when the Docker container is started. The password is printed to stdout of the container and can be found by looking at the container's log.

- MYSQL_ONETIME_PASSWORD : When the variable is true (which is its default state, unless MYSQL_ROOT_PASSWORD is set or MYSQL_ALLOW_EMPTY_PASSWORD is set to true), the root user's password is set as expired and must be changed before MySQL can be used normally. This variable is only supported for MySQL 5.6 and later.
- MYSQL_DATABASE : This variable allows you to specify the name of a database to be created on image startup. If a user name and a password are supplied with MYSQL_USER and MYSQL_PASSWORD, the user is created and granted superuser access to this database (corresponding to GRANT ALL). The specified database is created by a CREATE DATABASE IF NOT EXIST statement, so that the variable has no effect if the database already exists.
- MYSQL_USER, MYSQL_PASSWORD : These variables are used in conjunction to create a user and set that user's password, and the user is granted superuser permissions for the database specified by the MYSQL_DATABASE variable. Both MYSQL_USER and MYSQL_PASSWORD are required for a user to be created ; if any of the two variables is not set, the other is ignored. If both variables are set but MYSQL_DATABASE is not, the user is created without any privileges.
- MYSQL_ROOT_HOST : By default, MySQL creates the 'root'@'localhost' account. This account can only be connected to from inside the container. To allow root connections from other hosts, set this environment variable. For example, the value 172.17.0.1, which is the default Docker gateway IP, allows connections from the host machine that runs the container. The option accepts only one entry, but wildcards are allowed (for example, MYSQL_ROOT_HOST=172.*.*.* or MYSQL_ROOT_HOST=%).
- MYSQL_LOG_CONSOLE : When the variable is true (which is its default state for MySQL 8.0 server containers), the MySQL Server's error log is redirected to stderr, so that the error log goes into the Docker container's log and is viewable using the docker logs command.
- MYSQL_ROOT_PASSWORD : This variable specifies a password that is set for the MySQL root account.
- MYSQL_ALLOW_EMPTY_PASSWORD. Set it to true to allow the container to be started with a blank password for the root user.