CONTAINERISATION avec Docker

Jacky VISITTHIDETH

Octobre 2022

# Objectifs du TP et organisation

## Les objectifs de cette séance sont les suivants :

— Acquérir les bases de la containerisation

— Comprendre les concepts de Docker

— Manipuler des containers Docker avec les réseaux et volumes

— Débugger des containers

# Organisation, recommandation et notation du TP

Vous travaillerez en binôme.

Faites impérativement un compte rendu au fur et à mesure avec des copies d’écran pour les différents points.

Un travail doit être enregistré avec les noms des personnes dans le nom du fichier et l’intitulé du fichier doit être clair (par ex : TP\_intitulé\_du\_tp\_Etudiant1\_Etudiant).

Les délais sont parfois et exceptionnellement mais une fois fixés doivent être respectés sous peine d’une note nulle.

Les comptes-rendu sont à envoyer par mail vlaujacky@gmail.com

# Explorer et comprendre Docker

Aller sur le site de Docker Hub (<https://hub.docker.com/>) et se créer un compte.

Ce compte vous servira sur la suite du TP.

Explorer le site Docker Hub :

1. Combien d’images Docker sont considéré comme Docker Official Image ?
2. Quel est la différence entre les images Docker Official Image et Verified Publisher ?
3. Quels sont les 5 images Docker les plus utilisés sur Docker Hub (les 5 premières qui apparaissent en premier) ?
4. Dans les filtres, il semblerait que l’on puisse sélectionner Linux/Windows, que signifie cela ?

Microsoft ont-ils des images Docker sur Docker Hub ?

1. Aller sur la page Docker Hub de l’image Debian :
   1. Quels sont les architectures supportées par l’image Debian ?
   2. Sur quelle version de Debian pointe la version « stable » ?

# Manipuler Docker

* 1. Démarrer une VM Debian qui a accès à internet (ESXi/KVM comme vous voulez)
  2. Installer docker dans votre VM <https://docs.docker.com/desktop/install/debian/>
  3. Utiliser la commande « docker login » avec votre compte Docker Hub pour pouvoir récupérer des images Docker
  4. Utiliser la commande « docker pull » pour récupérer les images :
* debian :latest
* debian :8.11
  1. Prouvez-moi qu’elles ont bien été récupéré en cache sur votre machine avec la commande « docker image »
  2. Démarrer deux container debian en faisant tourner un processus bash, avec la commande « docker run »
  3. Stoppez-en un des deux avec la commande « docker stop »
  4. Afficher les containers sur la machine avec la commande « docker ps »
  5. Quel est la différence entre la commande « docker ps » et « docker ps –a » ?
  6. Lancer les deux images Debian de versions différentes (latest/8.11) et « rentrer dans le container » avec la commande « docker attach » : prouvez-moi que ce sont deux versions différentes ? (Indice : /etc/os-release)
  7. On peut aussi le faire avec la commande « docker exec », montrez-le-moi ?
  8. Au passage, créer un fichier toto.txt dans le container
  9. Quel est la différence entre « docker attach » et « docker exec » ?
  10. Afficher les réseaux docker avec la commande « docker network » : quels sont ceux par défaut ?
  11. Utiliser la commande « docker inspect » sur un container : quel type de réseau Docker est utilisé par défaut pour un container ?
  12. Faites un oneliner (une ligne de commande bash) pour stopper vos containers
  13. Pareil pour les supprimer, oneliner aussi
  14. Relancer un container Debian, votre fichier toto.txt existe toujours ?
  15. Relancer un container Debian avec l’argument - -volume pour rendre le fichier persistent
  16. Lancer un container nginx :latest ; quel est son IP dans le réseau Docker (« docker inspect ») ?
  17. Faites un curl pour vérifier <http://IP_DOCKER_DU_NGINX:PORT> pour vérifier que le site marche
  18. Le site est-il accessible depuis l’IP de votre VM ? (curl <http://IP_VM:PORT>)
  19. Faites-en sorte que ce soit possible en relançant un container avec l’argument “- -port »
  20. Régarder les logs de votre container nginx avec la commande « docker logs »

# Dockerfile et Docker-Compose

1. Faites une petite page Web HTML simple
2. A l’aide d’un Dockerfile, créer/builder une nouvelle image Docker intégrant votre page HTML en vous basant sur une image nginx :lastest (Commande « docker build »)
3. Démarrer cette image en utilisant Docker-compose et un fichier compose.yml, dans laquelle vous allez renseigner votre image récemment build