MediaSchool

BTS SIO – Option SLAM
Documentation d'épreuve

Mission 2 : Docker scripts & automatisations

Par Baptiste Grimaldi

Ce document est fourni en complément de la fiche E5 correspondante.

E4-Mission2-Docker-Scripts-Serveurs-Baptiste-Grimaldi

Liens utiles:

Article Serveurs Docker

Check out the portfolio of Baptiste Grimaldi, a full-stack web developer specializing in creating dynamic and responsive websites. Explore his projects and contact him for your next web development project.

https://portfolio.baptistegrimaldi.info/projects/view/serveurs-docker

Article de mon portfolio

SOMMAIRE

Introduction

But de la documentation

Le but de cette documentation est de fournir des instructions pour la mise en place de scripts Docker pour des serveurs.

Les scripts Docker sont des outils essentiels pour la création et la gestion de conteneurs Docker. Ils permettent de déployer des applications dans des environnements de production de manière rapide et efficace.

Dans cette documentation, nous allons décrire les objectifs et les exigences du projet, les environnements dans lesquels les scripts Docker seront utilisés, ainsi que la méthode de mise en place et de tests. Nous aborderons également la sécurité et le monitoring pour garantir un fonctionnement optimal des serveurs. Enfin, nous conclurons en soulignant les points clés et l'importance de ces scripts Docker dans la gestion de serveurs.

Scripts Docker

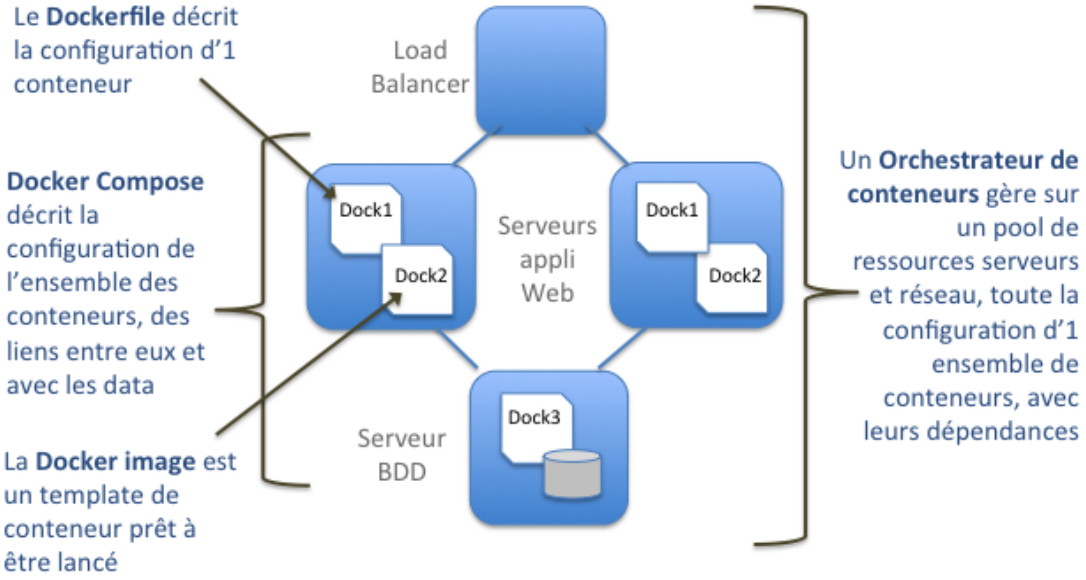
Les scripts Docker sont des outils essentiels pour la création et la gestion de conteneurs Docker. Ils permettent de déployer des applications dans des environnements de production de manière rapide et efficace.

Les scripts Docker sont écrits en utilisant un langage de script comme Bash ou Python. Ils peuvent être utilisés pour automatiser des tâches courantes telles que la création de conteneurs, le déploiement d'applications et la configuration de serveurs. Les scripts Docker peuvent également être utilisés pour la gestion de versions, la sauvegarde et la restauration de données, et la surveillance des performances.

> 💡 Les scripts Docker sont donc un outil essentiel pour tout administrateur de serveur qui souhaite déployer des applications dans des environnements de production de manière rapide et efficace. Ils permettent également de garantir la sécurité et le monitoring des serveurs pour un fonctionnement optimal.

Dans le cas de cette mission, donc distinguerons deux types de scripts:

- Les YAMLS
  - Ici, les YAMLS sont utilisé pour **docker compose**.
- Les Bash / Python
  - Les scripts pour mettre un **environnement** autour des **conteneurs**.



courte explication de docker-compose

Cahier des charges

Objectifs

L'objectif principal de ce projet est de mettre en place des scripts Docker pour faciliter la création et la gestion de conteneurs Docker sur les serveurs. Les scripts doivent être capables de déployer des applications dans des environnements de production de manière rapide et efficace, tout en rajoutant une couche de sécurité.

Les objectifs spécifiques comprennent la mise en place de scripts YAML pour Docker Compose et de scripts Bash/Python pour la création d'environnements

autour des conteneurs et de leurs utilisateurs. La méthodologie doit inclure des étapes pour la mise en place des scripts, les tests et la mise en œuvre dans des environnements de semi-production.

> 💡 On parle ici de semi-production car les scripts restent en interne.

Exigences

Les scripts Docker doivent être compatibles avec les dernières versions de Docker et doivent pouvoir être déployés sur différents systèmes d'exploitation. Ils doivent également être facilement modifiables pour permettre l'ajout de nouvelles fonctionnalités. Les scripts doivent avoir une documentation claire et complète pour faciliter leur utilisation et leur maintenance.

C’est dans cette optique que les scripts sont écrits en bash principalement car ne nécessitant pas de pré-installation sur tout les os:

MacOS: Basé unix donc exécute les fichiers bash nativement.

Windows:
Les scripts bash peuvent être exécutés sur Windows à l'aide d'un émulateur de terminal, comme Git Bash ou Cygwin. Ces émulateurs fournissent un environnement Unix-like qui permet d'exécuter des scripts bash. *Mais aujourd’hui windows tant vers le linux pour ses commandes hors powershell et sera donc considéré comme compatible pour les tâches les plus simples.*

Linux: Natif

Environnements

Présentation

L'environnement est un **Linux Ubuntu Server LTS 22.**

Ubuntu Server est une distribution Linux populaire pour les serveurs. Elle est utilisée pour les applications web, les bases de données, les fichiers, la virtualisation et plus encore. Ubuntu Server est un système d'exploitation léger qui peut être installé sur des machines virtuelles, des serveurs dédiés ou des serveurs cloud.

> 💡 Ubuntu Server 22.04 LTS est la dernière version de cette distribution. Elle est conçue pour la stabilité et la sécurité à long terme. Elle bénéficie d'un support de 5 ans pour les mises à jour de sécurité et les correctifs.

En ce qui concerne les outils pré-installés, Ubuntu Server 22.04 LTS offre une grande variété d'outils et de logiciels pour les serveurs, y compris Docker.

Sécurité

Notons qu'Ubuntu Server dispose d'un ensemble de protections de réseau et de systèmes bien configurés par défaut. En outre, Ubuntu Server est compatible avec une grande variété d'outils de sécurité tiers, tels que les ***pare-feu*** *et les programmes* ***anti-virus****, qui peuvent être facilement intégrés pour offrir une protection encore plus complète.*

> 💡 Il est important de garder à l'esprit que la sécurité est un processus continu et qu'il est donc important de mettre à jour régulièrement les scripts Docker pour inclure les dernières mises à jour de sécurité.

Les automatisations

Définition des machines

Définition des concepts de l’infrastructure logique

Nom	Description	Utilisation
groupmng	Script Python pour la gestion des utilisateurs et des groupes sous forme d'arbre.	Vérification des groupes actifs.
docker compose all up	Démarre tous les conteneurs vitaux pour mes serveurs.	Est exécuté au démarrage de la machine hôte.
docker compose services	Un “service” docker-compose est une application définie par docker-compose.	Plusieurs services de la même famille sont définies ensembles.

Nom	Description	Utilisation
docker networks	Un réseau docker permet aux conteneurs Docker de communiquer entre eux et avec d'autres serveurs, de manière isolée et sécurisée. ✱	Seuls les docker nécessitant de communiquer ensemble sont sur le même réseau.
docker volumes	Façon de rendre persistant de la donnée sur une machine docker.	Maintenir et backup les configurations des services conteneurs.
docker cpus	Surveillance et contrôle du status et de l’utilisation des CPU	Surveillance avec netdata et contrôle des performances avec docker stats
docker stats	Affiche les statistiques des conteneurs Docker en cours d'exécution	Surveillance des performances
netdata	Netdata est un outil de surveillance en temps réel pour Linux et Docker. Il affiche les statistiques de performance et peut alerter en cas de problèmes. Utile pour surveiller les serveurs et les conteneurs.	Surveillance des connexions réseau par “groupe” docker
uptime bot	Envoie des alertes en cas de panne de service	Surveillance des services
backup	Script pour sauvegarder les données	Sauvegarde des données
update n upgrade	Script pour mettre à jour le système d'exploitation et les paquets	Mise à jour du système
create sftp user	Script pour créer un utilisateur SFTP et créer et paramétrer son environnement.	Gestion des utilisateurs
create dns challenge	Script pour créer un challenge DNS ACME	Gestion des groupements de certificats SSL chez OVH.

> Un réseau docker est un réseau virtuel qui permet aux conteneurs Docker de communiquer entre eux et avec d'autres serveurs, de manière isolée et sécurisée. Il peut être configuré pour permettre l'accès à Internet ou à des réseaux internes.

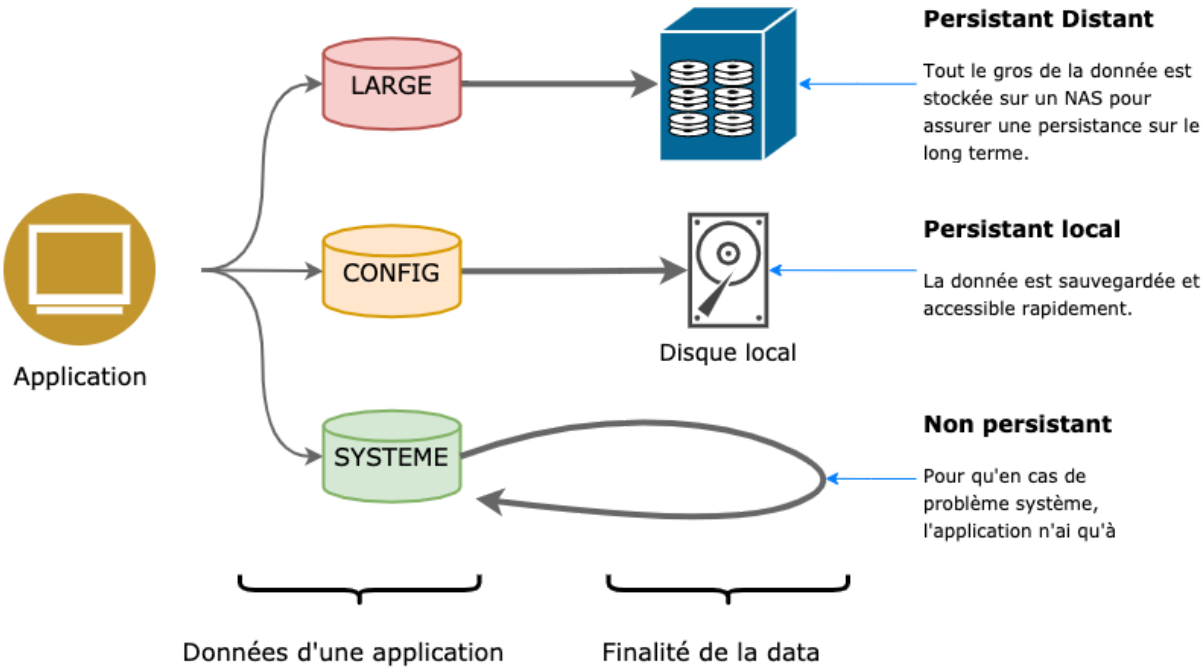
Mise en commun des machines

Persistance des données

> Les volumes Docker permettent de stocker des données de manière persistante en dehors du conteneur, ce qui est essentiel pour éviter la perte de données en cas de suppression ou de mise à jour du conteneur.

Les volumes peuvent être utilisés pour stocker des fichiers de configuration, des bases de données, des fichiers journaux et d'autres données importantes. Il est important de mettre en place des sauvegardes régulières des volumes pour garantir la récupération des données en cas de problème.

Dans ma machine hôte, la charge des données est répartie en fonction de l'utilisation et de la taille du volume:



Organisation de la persistance des donnés sur mon serveur.

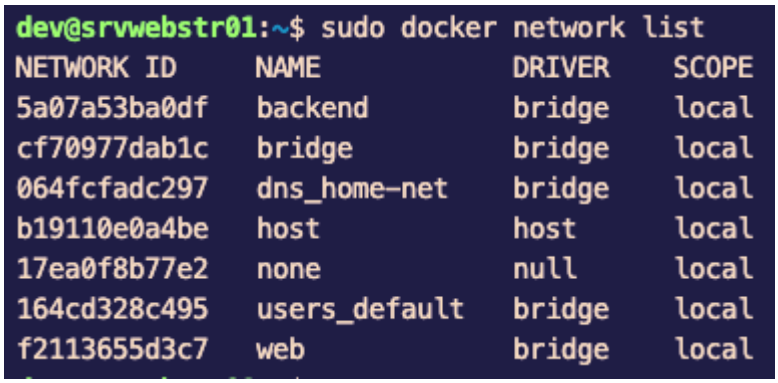
Les réseaux docker

> Les réseaux docker permettent aux conteneurs Docker de communiquer entre eux et avec d'autres serveurs de manière isolée et sécurisée. L'utilisation de réseaux distincts pour chaque service ou groupe de services peut renforcer la sécurité en limitant les possibilités de propagation des attaques.

Il est important de surveiller régulièrement les serveurs pour détecter toute activité suspecte et pour garantir un fonctionnement optimal des scripts Docker. Des outils de surveillance tels que les journaux système, les alertes de sécurité et les outils de détection d'intrusion peuvent être utilisés pour assurer la sécurité des serveurs.

Il est également important de documenter clairement et complètement les scripts Docker pour faciliter leur utilisation et leur maintenance. Les scripts doivent être facilement modifiables pour permettre l'ajout de nouvelles fonctionnalités et doivent être compatibles avec les dernières versions de Docker et les différents systèmes d'exploitation.

Dans cette optique, au début de chaque redémarrage, les réseaux vitaux au bon fonctionnement de tout les services sont vérifiés. Egalement certaines machines sont statiques sur le réseau afin d’assurer un bon fonctionnement avec le reverse proxy.



Taefik | reverse proxy

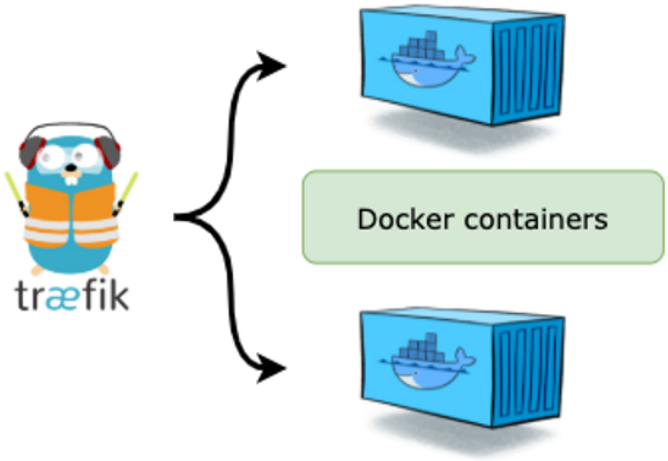
Taefik | reverse proxy

Traefik est un reverse proxy et un load-balancer open source qui permet de gérer le trafic entrant et sortant des conteneurs Docker. Il permet de faire du routage dynamique, de la répartition de charge, de l'ajout et du retrait de conteneurs en temps réel, de la gestion des certificats SSL et de la surveillance des performances.

Traefik peut être configuré pour s'intégrer avec Docker et Kubernetes, ce qui en fait une solution idéale pour les environnements de conteneurs. Il dispose également d'une interface utilisateur Web pour une gestion facile et une visualisation des statistiques de trafic.

Traefik est utilisé comme reverse proxy pour permettre l'accès à des sous-domaines pour des machines différentes. Cela permet de gérer le trafic entrant et sortant des conteneurs Docker, de faire du routage dynamique, de la répartition de charge, de

l'ajout et du retrait de conteneurs en temps réel, de la gestion des certificats SSL et de la surveillance des performances.



Taefik redirige en fonction du sous-domaine dans mon cas. (portfolio.baptistegrimaldi.info ou (taberu.baptistegrimaldi.info)

Les CPU sur docker

Les outils de surveillance des performances des conteneurs Docker tels que `netdata` et `docker stats` doivent être configurés pour surveiller les performances des CPU. Cela permet de déterminer si les conteneurs consomment trop ou trop peu de CPU. Les résultats de ces outils peuvent être utilisés pour ajuster les ressources allouées aux conteneurs et pour améliorer les performances globales du système.

Lors de la création de scripts Docker, il est important de veiller à documenter de manière claire et complète leur fonctionnement pour faciliter leur utilisation et leur maintenance. Il est également essentiel de prévoir la possibilité de modifier facilement les scripts pour ajouter de nouvelles fonctionnalités et de s'assurer qu'ils sont compatibles avec les dernières versions de Docker et les différents systèmes d'exploitation.

Enfin, il est crucial de mettre en place des mesures de sécurité pour protéger les scripts Docker et les données des utilisateurs. Cela peut inclure la configuration de pare-feu, l'utilisation de mots de passe sécurisés et la limitation des droits d'accès.

NAME	CPU %	MEM USAGE / LIMIT	MEM %
web-portfolio	0.00%	100.3MiB / 400MiB	25.07%
nextcloud-home-nextcloud-1	0.01%	487.6MiB / 7.531GiB	6.32%
web-whatsmyiris	0.00%	123.6MiB / 512MiB	24.15%
sandbox-fred	0.00%	92.16MiB / 256MiB	36.00%
php-taberu	0.01%	36.62MiB / 64MiB	57.23%
mariadb	0.03%	174.6MiB / 512MiB	34.11%
home-dns	0.00%	7.152MiB / 128MiB	5.59%
phpmyadmin	0.01%	92.46MiB / 128MiB	72.23%
traefik_reverse-proxy_1	0.00%	49.06MiB / 512MiB	9.58%

répartition de CPU sur mon serveur

💡 Ces données sont configurées dans le fichier docker-compose du service.

Sécurité des machines

Les réseaux docker pour la sécurité

Les réseaux docker pour la sécurité

Les réseaux docker permettent de segmentariser les conteneurs dans des réseaux virtuels. Cela peut renforcer la sécurité en limitant les possibilités de propagation des attaques. Dans les configurations Docker Compose, les réseaux sont utilisés pour isoler les conteneurs qui ont besoin de communiquer entre eux.

💡 Par exemple, la base de données est sur le réseau backend, et seuls les conteneurs sur ce réseau peuvent la voir.

Il est important de surveiller régulièrement les serveurs pour détecter toute activité suspecte et pour garantir un fonctionnement optimal des scripts Docker. Des outils de surveillance, tels que les journaux système, les alertes de sécurité et les outils de détection d'intrusion, peuvent être utilisés pour assurer la sécurité des serveurs.

Il est également important de documenter clairement et complètement les scripts Docker pour faciliter leur utilisation et leur maintenance. Les scripts doivent être facilement modifiables pour permettre l'ajout de nouvelles fonctionnalités et doivent être compatibles avec les dernières versions de Docker et les différents systèmes d'exploitation.

Les certificats SSL

Le but de la configuration des certificats SSL avec Traefik et OVH pour les dockers est de garantir la sécurité des connexions HTTPS entre les utilisateurs et les conteneurs Docker.

> 💡 Traefik est un reverse proxy utilisé pour gérer le trafic entrant et sortant des conteneurs Docker. Il dispose d'une fonctionnalité de gestion des certificats SSL qui permet de générer et de renouveler automatiquement les certificats SSL pour les sous-domaines utilisés par les conteneurs Docker.

La configuration des certificats SSL sur Traefik se fait en utilisant les fichiers de configuration YAML de Docker Compose. Une fois la configuration effectuée, Traefik détecte automatiquement les sous-domaines utilisés par les conteneurs Docker et génère les certificats SSL requis.

OVH est un fournisseur de services d'hébergement qui propose des certificats SSL gratuits pour les domaines hébergés sur ses serveurs. La configuration des certificats SSL sur OVH implique la création d'un compte OVH, la génération des certificats SSL pour les domaines concernés et la configuration des fichiers de configuration YAML de Docker Compose pour utiliser les certificats SSL générés par OVH.

La configuration des certificats SSL sur Docker Compose est importante pour garantir la sécurité des connexions HTTPS entre les utilisateurs et les conteneurs Docker. Les certificats SSL permettent de chiffrer les données échangées entre les utilisateurs et les conteneurs Docker, ce qui garantit la confidentialité et l'intégrité des données.

Les outils à ma dispositions

Groupmng

Vous trouverez ici, le script énoncé:
https://github.com/GrimalDev/RandomDevProjects/tree/main/LinuxScripts

Le script Python "groupmng" est mon outil utilisé pour la gestion des utilisateurs et des groupes sous forme d'arbre. Il permet de vérifier les groupes actifs et de faciliter la gestion des utilisateurs et des groupes dans un environnement de production. Le script est disponible sur mon GitHub.

Docker-compose-all-up.sh

Le script `docker-compose-all-up.sh` est utilisé pour démarrer tous les conteneurs vitaux pour le bon fonctionnement des serveurs. Il est exécuté au démarrage de la machine hôte.

A chaque exécutions, le script réinstalle réellement les machines et relance les réseaux.

backup.sh

```
#!/bin/bash

DATE=$(date '+%F')

# Define the backup directories
DOCKER_BACKUP_DIR=/mnt/remote/backup/docker-backups
SFTP_BACKUP_DIR=/mnt/remote/backup/sftp-backups

# Create a new backup
echo -e "Creating a new backup...\n"
sudo rsync -aAXv --delete /docker/ $DOCKER_BACKUP_DIR/$DATE
sudo rsync -aAXv --delete /sftp/ $SFTP_BACKUP_DIR/$DATE
```

sftp users

Les prisons en SSH sont une méthode de sécurité qui permet de restreindre l'accès des utilisateurs à certaines parties du système de fichiers en leur donnant un accès limité. Cela peut être utile pour limiter les risques de sécurité en cas d'attaques potentielles ou de comportement malveillant de la part d'utilisateurs non autorisés.

Mon script utilise **ces prisons**, créer un **environnement sécurisé** et créer l'**utilisateur autour** de cet environnement.

Méthodologie

Mise en place

La mise en place des scripts Docker doit suivre une méthodologie rigoureuse pour garantir leur bon fonctionnement. Les étapes comprennent la configuration de l'environnement, l'installation de Docker et la mise en place des scripts Docker. Il est important de tester les scripts à chaque étape pour garantir leur bon fonctionnement.

Je travaille en mode projet avec un **kanban** pour m'organiser et être sûr d'effectuer mes améliorations de manière ordonnée.

Voici les avantages du kanban pour l'organisation de mes projets :

- **Hiérarchisation des tâches par ordre de priorité** : avec le kanban, je peux facilement identifier les tâches les plus importantes et les prioriser en conséquence. Cela me permet de consacrer suffisamment de temps et de ressources aux tâches les plus importantes et de m'assurer qu'elles sont effectuées dans les délais impartis.
- **Suivi visuel de l'avancement des tâches** : le kanban est une méthode visuelle qui utilise un tableau pour représenter toutes les tâches et leurs statuts respectifs. Chaque tâche est représentée par une carte qui peut être déplacée de colonne en colonne, en fonction de son état d'avancement. Les colonnes représentent généralement les états suivants : à faire, en cours, terminé.
- **Identification rapide des tâches qui prennent plus de temps que prévu** : le kanban me permet de suivre l'avancement de chaque tâche et de savoir où j'en suis à tout moment. Je peux facilement identifier les tâches qui prennent plus de temps que prévu et prendre les mesures nécessaires pour les terminer dans les délais impartis.

En utilisant cette méthode, je peux garantir que toutes les tâches sont effectuées dans les délais impartis et que les améliorations sont effectuées de manière ordonnée et efficace.

> 💡 Le kanban est une méthode efficace pour organiser ses projets et ses améliorations. Il permet de hiérarchiser les tâches par ordre de priorité, de suivre l'avancement de chaque tâche de manière visuelle et d'identifier rapidement les tâches qui prennent plus de temps que prévu.

Tests

Les tests doivent être effectués régulièrement pour garantir que les scripts Docker fonctionnent correctement. Les tests peuvent inclure des tests de charge, des tests d'intégration et des tests de sécurité. Les résultats des tests doivent être documentés pour permettre une analyse ultérieure.

💬 EXEMPLE DE TEST

Pour tester un environnement docker-compose, vous pouvez utiliser la commande `docker-compose up`, qui permet de démarrer les conteneurs définis dans le fichier `docker-compose.yml`. Cette commande crée également les réseaux et les volumes nécessaires pour les conteneurs.

Une fois les conteneurs démarrés, vous pouvez vérifier leur état en utilisant la commande `docker ps`. Cette commande affiche une liste des conteneurs en cours d'exécution, avec leur ID, leur nom, leur image, leur état et leurs ports exposés.

Vous pouvez également vérifier l'état des réseaux et des volumes en utilisant les commandes `docker network ls` et `docker volume ls`. Ces commandes affichent respectivement une liste des réseaux et des volumes Docker en cours d'utilisation.

Mise en œuvre

Scripts Docker

Une fois les scripts Docker mis en place et testés, ils peuvent être déployés dans des environnements de semi-production. Les scripts Docker doivent être documentés de manière claire et complète pour faciliter leur utilisation et leur maintenance.

Sécurité

Il est important de mettre en place des mesures de sécurité pour protéger les scripts Docker et les données de l'utilisateur. Les mesures de sécurité peuvent inclure la configuration de pare-feux, l'utilisation de mots de passe sécurisés et la limitation des droits d'accès.

Monitoring

Solutions

Il est important de surveiller régulièrement les serveurs pour détecter toute activité suspecte et pour garantir un fonctionnement optimal des scripts Docker. Les outils de surveillance peuvent inclure des journaux système, des alertes de sécurité et des outils de détection d'intrusion.

Alertes de performance

Les outils de surveillance des performances des conteneurs Docker tels que `netdata` et `docker stats` doivent être configurés pour surveiller les performances des CPU. Cela permet de déterminer si les conteneurs consomment trop ou trop peu de CPU. Les résultats de ces outils peuvent être utilisés pour ajuster les ressources allouées aux conteneurs et pour améliorer les performances globales du système.

Les tests de sécurité

Les tests de sécurité sont cruciaux pour garantir que les scripts Docker sont sûrs et fiables. Les tests peuvent inclure des tests de pénétration, des tests d'intrusion et des tests de vulnérabilité. Les résultats des tests doivent être documentés pour permettre une analyse ultérieure et une amélioration continue de la sécurité.

Gestion des mises à jour

La gestion des mises à jour des scripts Docker est importante pour garantir leur bon fonctionnement et leur sécurité. Les mises à jour peuvent inclure des correctifs de sécurité, des améliorations de fonctionnalités et des mises à jour de compatibilité avec les dernières versions de Docker et les différents systèmes d'exploitation.

Il est important de suivre les meilleures pratiques pour la gestion des mises à jour, telles que la mise en place de tests rigoureux avant le déploiement des mises à jour et la documentation claire et complète des modifications apportées aux scripts Docker.

Conclusion

Points clés

- Il est important de surveiller régulièrement les serveurs pour détecter toute activité suspecte et pour garantir un fonctionnement optimal des scripts Docker. Des outils de surveillance peuvent inclure des journaux système, des alertes de sécurité et des outils de détection d'intrusion.
- Les tests de sécurité sont cruciaux pour garantir que les scripts Docker sont sûrs et fiables. Les tests peuvent inclure des tests de pénétration, des tests d'intrusion et des tests de vulnérabilité. Les résultats des tests doivent être documentés pour permettre une analyse ultérieure et une amélioration continue de la sécurité.

- La gestion des mises à jour des scripts Docker est importante pour garantir leur bon fonctionnement et leur sécurité. Les mises à jour peuvent inclure des correctifs de sécurité, des améliorations de fonctionnalités et des mises à jour de compatibilité avec les dernières versions de Docker et les différents systèmes d'exploitation.

Importance

La mise en place de mesures de sécurité est cruciale pour protéger les scripts Docker et les données de l'utilisateur. La configuration de pare-feux, l'utilisation de mots de passe sécurisés et la limitation des droits d'accès sont des mesures importantes à prendre.

Il est également important de documenter clairement et complètement les scripts Docker pour faciliter leur utilisation et leur maintenance. Les scripts doivent être facilement modifiables pour permettre l'ajout de nouvelles fonctionnalités et doivent être compatibles avec les dernières versions de Docker et les différents systèmes d'exploitation.

Enfin, il est crucial de surveiller régulièrement les serveurs pour détecter toute activité suspecte et pour garantir un fonctionnement optimal des scripts Docker. Des outils de surveillance tels que les journaux système, les alertes de sécurité et les outils de détection d'intrusion peuvent être utilisés pour assurer la sécurité des serveurs.

Avis de droits d'auteur

Informations légales sur les droits d'auteur de cette documentation

Par Baptiste Grimaldi