Doc Installation Français

Traduction Google !!!

Installer l'application Fabmanager en production avec Docker

Ce fichier README tente de décrire toutes les étapes de mise en production d'une application fabmanager sur un serveur, à partir d'une solution utilisant Docker et Docker-composer. Nous recommandons DigitalOcean, mais ces étapes fonctionneront sur n'importe quel fournisseur de cloud compatible Docker ou serveur local.

Pour que cela fonctionne, veuillez utiliser la même structure de répertoires que celle décrite dans ce guide dans votre dossier de l'application fabmanager. Vous devrez être root à travers le reste de l'installation.

Table des matières

1. [Étapes préliminaires] (# étapes préliminaires)
    1. configurer le serveur
    2. acheter un nom de domaine et le lier avec la gouttelette
    3. se connecter à la gouttelette via SSH
    4. se préparer le serveur
    5. dossiers d'installation et fichier env
    6. Configurer le fichier nginx
    7. Certificat SSL avec LetsEncrypt
    8. exigences
2. [Installer Fabmanager] (# install-fabmanager)
    1. Ajouter le fichier docker-compose.yml
    2. Tirer des images
    3. base de données d'installation
    4. construire des actifs
    5. préparer Elasticsearch (moteur de recherche)
    6. démarrer tous les services
3. [Générer un certificat SSL par Letsencrypt] (# generate-ssl-certificate-by-letsencrypt)
    1. [Utilitaires Docker] (# docker-utils)
    2. [Mettre à jour Fabmanager] (# update-fabmanager)
    3. Étapes
    4. Bon à savoir

Étapes préliminaires

configurer le serveur

Allez dans [DigitalOcean] (https://www.digitalocean.com/) et créez une droplet avec des applications en un clic ** “Docker sur Ubuntu 16.04 LTS” ** (Docker et Docker-composent sont préinstallés). Vous avez besoin d'au moins 2 Go de mémoire adressable (RAM + échange) pour installer et utiliser

FabManager. Nous recommandons 4 Go de RAM pour les communautés plus importantes. Choisissez un centre de données. Définissez le nom d'hôte en tant que votre nom de domaine.

acheter un nom de domaine et le lier avec le serveur

1. Achetez un nom de domaine sur [OVH] (https://www.ovh.com/fr/)
2. Remplacez l'adresse IP du domaine par l'adresse IP de la droplet (vous pouvez activer l'ip flexible et l'utiliser)
3. ** N'essayez pas ** d'accéder à votre nom de domaine tout de suite, les DNS ne sont pas encore au courant du changement, donc ** ATTENDEZ ** et soyez patient.

se connecter au serveur via SSH

Vous pouvez déjà vous connecter au serveur avec cette commande: `ssh root @ server-ip`. Lorsque la propagation DNS sera effectuée, vous serez en mesure de connectez-vous au serveur avec `ssh root @ your-domain-name`.

préparer le serveur

Nous vous recommandons de:

1. ugprade votre système
2. ajouter au moins 2GB de swap
3. Vérifiez que vous utilisez une connexion via une clé SSH. Si c'est le cas, vous pouvez définir le mot de passe racine (pour la console de débogage) et désactiver la connexion par mot de passe.

Pour ce faire, vous pouvez utiliser le script suivant:

```
`` `bash
cd / root
git clone https://github.com/sleede/lazyscripts.git
cd lazyscripts /
chmod a + x prepare-vps.sh
./prepare-vps
`` `
```

dossiers d'installation et fichier env

Créez le dossier de configuration: `` `bash mkdir -p / apps / fabmanager / config `` `

Faites une copie du fichier ** docker / env.example ** et utilisez-le comme point de départ. Définissez toutes les variables d'environnement nécessaires à votre application. Veuillez vous reporter au [README de FabManager] (https://github.com/LaCasemate/fab-manager/blob/master/README.md#environment-configuration) pour des explications sur ces variables.

Ensuite, copiez le fichier `env.example` précédemment personnalisé en tant que` / apps / fabmanager / config / env`

programme d'installation nginx

Créez le dossier nginx: `` `bash mkdir -p / apps / fabmanager / config / nginx `` `

Personnalisez le fichier docker / nginx_with_ssl.conf.example * Remplacez ** MAIN_DOMAIN ** (exemple: fab-manager.com). * Remplacez ** URL_WITH_PROTOCOL_HTTPS ** (exemple: https://www.fab-manager.com). * Remplacer ** ANOTHER_URL_1 **, ** ANOTHER_URL_2 ** (exemple: .fab-manager.fr)

** Utilisez nginx.conf.example si vous ne souhaitez pas utiliser le protocole SSL pour votre application. **

Alors, Copiez le `nginx_with_ssl.conf.example` précédemment personnalisé en tant que` / apps / fabmanager / config / nginx / fabmanager.conf`

OU

Copiez le `nginx.conf.example` précédemment personnalisé en tant que` / apps / fabmanager / config / nginx / fabmanager.conf` si vous ne voulez pas utiliser ssl (non recommandé!).

Certificat SSL avec LetsEncrypt

** SUIVEZ CES INSTRUCTIONS UNIQUEMENT SI VOUS VOULEZ UTILISER SSL **.

Let's Encrypt est une nouvelle autorité de certification gratuite, automatisée et ouverte. Les certificats Let's Encrypt expirent après 90 jours. L'automatisation du renouvellement de vos certificats est donc importante. Voici la configuration d'un minuteur et d'un service pour renouveler les certificats et redémarrer le conteneur Docker de l'application:

`` `bash mkdir -p / apps / fabmanager / config / nginx / ssl `` ` Exécutez `openssl dhparam -out dhparam.pem 4096` dans le dossier / apps / fabmanager / config / nginx / ssl (générez le fichier dhparam.pem) `` `bash mkdir -p / apps / fabmanager / letsencrypt / config / `` ` Copiez le `webroot.ini.example` personnalisé précédemment comme` / appsfabmanager / letsencrypt / config / webroot.ini` `` `bash mkdir -p / apps / fabmanager / letsencrypt / etc / webrootauth `` `

Exécutez `docker pull quay.io/letsencrypt/letsencrypt: latest`

Créez le fichier (avec sudo) /etc/systemd/system/letsencrypt.service et collez-y la configuration suivante:

`` `bash [Unité] Description = letsencrypt Nécessite = docker.service

[Un service] Type = oneshot ExecStart = / usr / bin / docker run –rm —name letsencrypt -v “/ apps / fabmanager / log: / var / log / letencrypt” -v “/ apps / fabmanager / letencrypt / etc: / etc / letsencrypt” -v “/ apps / fabmanager / letsencrypt / config: / letsencrypt-config “quay.io/letsencrypt/letsencrypt:latest -c” /letsencrypt-config/webroot.ini “certonly ExecStartPost = - / usr / bin / docker redémarrez fabmanager_nginx_1 `` `

Créez un fichier (avec sudo) /etc/systemd/system/letsencrypt.timer et collez la configuration suivante: `` `bash [Unité] Description = temporisateur OneShotShot Nécessite = docker.service

[Minuteur] OnCalendar = * - * - 1 06:00:00 Persistant = vrai Unit = letsencrypt.service

[Installer] WantedBy = timers.target `` `

C'est tout pour le moment. Continuez l'installation, nous terminerons cette partie après le déploiement dans [Générer un certificat SSL par Letsencrypt] (# generate-ssl-cert-letsencrypt).

Exigences

Vérifiez que Docker et Docker-composer sont installés: (C'est normalement le cas si vous avez utilisé une image préconfigurée.)

`` `bash info docker docker-compose -v `` `

Sinon, vous pouvez installer docker sur ubuntu avec les instructions suivantes: https://docs.docker.com/engine/installation/linux/ubuntu/#install-using-the-repository

Pour installer docker-compose:

`` `bash curl -L https://github.com/docker/compose/releases/download/1.13.0/docker-compose-`uname -s`-`uname -m`> ./docker-compose sudo mkdir -p / opt / bin sudo mv docker-compose / opt / bin / sudo chmod + x / opt / bin / docker-compose `` `

Installer Fabmanager

Ajouter le fichier docker-compose.yml

Copiez docker-compose.yml dans le dossier de votre application `/ apps / fabmanager`. Les commandes docker-composer doivent être lancées à partir du dossier `/ apps / fabmanager`.

tirer des images

`` `bash docker-composer tirer `` `

base de données d'installation

`` `bash docker-compose exécuter –rm fabmanager bundle exec rake db: créer # créer la base de données docker-compose exécuter –rm fabmanager bundle exec rake db: migrer # exécuter toutes les migrations docker-compose run –rm -e ADMIN_EMAIL = xxx ADMIN_PASSWORD = xxx paquet fabmanager exec rake db: graines # seed la base de données `` `

construire des actifs

`docker-compose exécuter –rm fabmanager bundle exec rake assets: précompiler`

préparer Elasticsearch (moteur de recherche)

`docker-compose run –rm fabmanager paquet rake exec fablab: es_build_stats`

démarrer tous les services

`docker-compose jusqu'à -d

Générer un certificat SSL par Letsencrypt

** Important: l'application doit être exécutée sur http avant de commencer letsencrypt **

Démarrer le service letsencrypt: `` `bash sudo systemctl démarre letsencrypt.service `` `

Si le certificat a été généré avec succès, mettez à jour le fichier de configuration nginx et activez le port ssl et le certificat éditer le fichier `/ apps / fabmanager / config / nginx / fabmanager.conf`.

Supprimez le conteneur de votre application et exécutez votre application pour appliquer les modifications exécutant les commandes suivantes: `` `bash docker-compose vers le bas docker-compose up -d `` `

Enfin, si tout va bien, démarrez letsencrypt timer pour mettre à jour le certificat tous les 1 du mois:

`` `bash sudo systemctl active letsencrypt.timer sudo systemctl démarre letsencrypt.timer (cochez) sudo systemctl list-timers `` `

Docker utils avec docker-compose

Redémarrer l'application

`docker-compose redémarre fabmanager`

Supprimer l'application

`docker-compose vers le bas fabmanager`

Redémarrez tous les conteneurs

`redémarrage docker-compose`

Enlever tous les conteneurs

`docker-compose vers le bas`

Démarrer tous les conteneurs

`docker-compose jusqu'à -d

Ouvrir un bash dans le contexte de l'application

`docker-composez run –rm fabmanager bash`

Afficher le statut des services

`docker-compose ps`

Redémarrer le conteneur nginx

`docker-compose redémarre nginx`

Exemple de commande passant des variables d'environnement

docker-compose run –rm -e ADMIN_EMAIL = xxx ADMIN_PASSWORD = xxx paquet fabmanager rake db: seed

mise à jour Fabmanager

* Cette procédure met à jour fabmanager vers la version la plus récente par défaut. *

Pas

Lorsqu'une nouvelle version est disponible, voici comment mettre à jour l'application fabmanager dans un environnement de production, en utilisant docker-compose:

1. allez dans le dossier de votre application

```
`cd / apps / fabmanager`
```

2. tirer les dernières images du docker

```
`traction de docker-composer
```

3. arrêtez l'application

```
`docker-compose stop fabmanager`
```

4. supprimer les anciens actifs

```
`rm -Rf public / assets /`
```

5. compiler de nouveaux atouts

```
`docker-compose exécuter --rm fabmanager bundle exec rake assets:
précompiler`
```

6. exécuter des commandes spécifiques

- * N'oubliez pas ** de vérifier s'il y a des commandes à exécuter pour votre mise à niveau. Ces commandes

sont toujours spécifiés dans [CHANGELOG] (https://github.com/LaCasemate/fab-manager/blob/master/CHANGELOG.md) et préfixés par ** [TODO DEPLOY] **.

```
 Ils sont également présents dans la [page des versions]
(https://github.com/LaCasemate/fab-manager/releases).
```

Ces commandes exécutent des tâches spécifiques et doivent être exécutées à la main.

7. redémarrez tous les conteneurs

```
`` `bash
  docker-compose vers le bas
  docker-compose up -d
`` `
```

Vous pouvez vérifier que tous les conteneurs fonctionnent avec `docker ps`.

Bon à savoir

Est-il possible de mettre à jour plusieurs versions en même temps?

Oui en effet. C'est le comportement par défaut lorsque la commande `docker-composer pull` va récupérer les dernières versions des images du docker. Veillez à exécuter toutes les commandes spécifiques répertoriées dans [CHANGELOG](https://github.com/LaCasemate/fab-manager/blob/master/CHANGELOG.md) entre vos et la nouvelle version dans l'ordre séquentiel. (Exemple: pour mettre à jour de 2.4.0 à 2.4.3, vous allez exécuter les commandes spécifiques pour le 2.4.1, puis pour le 2.4.2 et ensuite pour le 2.4.3).

From:
https://chanterie37.fr/fablab37110/ - **Castel'Lab le Fablab MJC de Château-Renault**

Permanent link:
https://chanterie37.fr/fablab37110/doku.php?id=start:fabmanager:doc1

Last update: **2023/01/27 16:08**