AlphaVPS

- Docker, Docker Compose, Portainer
- Sécuriser et Configurer un VPS avec UFW & Nginx Proxy Manager

Docker, Docker Compose, Portainer

1. Installation de Docker

Mettre à jour le système :

```
apt update && apt upgrade -y
```

Installer les dépendances nécessaires :

```
apt install -y ca-certificates curl gnupg
```

Ajouter la clé GPG officielle de Docker :

```
install -m 0755 -d /etc/apt/keyrings
curl -fsSL https://download.docker.com/linux/debian/gpg | gpg --dearmor -o /etc/apt/keyrings/docker.gpg
chmod a+r /etc/apt/keyrings/docker.gpg
```

Ajouter le dépôt officiel de Docker :

```
echo "deb [arch=$(dpkg --print-architecture) signed-by=/etc/apt/keyrings/docker.gpg]
https://download.docker.com/linux/debian $(lsb_release -cs) stable" > /etc/apt/sources.list.d/docker.list
```

Mettre à jour les paquets et installer Docker :

```
apt update
apt install -y docker-ce docker-ce-cli containerd.io docker-buildx-plugin docker-compose-plugin
```

Vérifier que Docker fonctionne correctement :

```
docker run hello-world
```

Si tout fonctionne, Docker est bien installé.

Si non, il se peu qu'il y ait un problème avec le daemon. Alors fais ça:

```
update-alternatives --set iptables /usr/sbin/iptables-legacy
apt reinstall docker-ce
```

---

2. Installation de Docker Compose

> **i Depuis Docker 2.x, `docker-compose` est inclus dans Docker sous forme de plugin.**
> Tu peux vérifier la version installée avec :

```
docker compose version
```

Si tu veux utiliser `docker-compose` comme une commande indépendante, installe-le avec :

```
curl -SL https://github.com/docker/compose/releases/latest/download/docker-compose-linux-$(uname -m) -o
/usr/local/bin/docker-compose
chmod +x /usr/local/bin/docker-compose
```

Vérifie l'installation :

```
docker-compose version
```

---

3. Installation de Portainer

Créer un volume pour stocker les données de Portainer :

```
docker volume create portainer_data
```

Lancer le conteneur Portainer :

```
docker run -d -p 8000:8000 -p 9443:9443 --name portainer --restart=always \
    -v /var/run/docker.sock:/var/run/docker.sock \
    -v portainer_data:/data portainer/portainer-ce:latest
```

Accéder à l'interface web de Portainer :

- Ouvre un navigateur et va sur :
  `https://<TON_IP>:9443`
- Crée un compte administrateur lors de la première connexion.

---

4. Vérification et gestion

Voir les conteneurs actifs :

```
docker ps
```

Démarrer/arrêter un conteneur :

```
docker start portainer
docker stop portainer
```

Mettre à jour Portainer :

```
docker stop portainer
docker rm portainer
docker pull portainer/portainer-ce:latest
docker run -d -p 8000:8000 -p 9443:9443 --name portainer --restart=always \
    -v /var/run/docker.sock:/var/run/docker.sock \
    -v portainer_data:/data portainer/portainer-ce:latest
```

[EN]

1. Install Docker

Update system packages:

```
apt update && apt upgrade -y
```

Install required dependencies:

```
apt install -y ca-certificates curl gnupg
```

Add Docker’s official GPG key:

```
install -m 0755 -d /etc/apt/keyrings
curl -fsSL https://download.docker.com/linux/debian/gpg | gpg --dearmor -o /etc/apt/keyrings/docker.gpg
chmod a+r /etc/apt/keyrings/docker.gpg
```

Add the official Docker repository:

```
echo "deb [arch=$(dpkg --print-architecture) signed-by=/etc/apt/keyrings/docker.gpg]
https://download.docker.com/linux/debian $(lsb_release -cs) stable" > /etc/apt/sources.list.d/docker.list
```

Update package lists and install Docker:

```
apt update
apt install -y docker-ce docker-ce-cli containerd.io docker-buildx-plugin docker-compose-plugin
```

Verify Docker is installed correctly:

```
docker run hello-world
```

If you see a success message, Docker is correctly installed. □□

If not, there might be a problem with the daemon. So, do this:

```
update-alternatives --set iptables /usr/sbin/iptables-legacy
apt reinstall docker-ce
```

---

2. Install Docker Compose

> **i Since Docker 2.x, `docker-compose` is included as a Docker plugin.**
> Check if it’s already installed:

```
docker compose version
```

If you need to install `docker-compose` as a standalone command:

```
curl -SL https://github.com/docker/compose/releases/latest/download/docker-compose-linux-$(uname -m) -o /usr/local/bin/docker-compose
chmod +x /usr/local/bin/docker-compose
```

Verify the installation:

```
docker-compose version
```

---

3. Install Portainer

Create a volume for Portainer data:

```
docker volume create portainer_data
```

Run the Portainer container:

```
docker run -d -p 8000:8000 -p 9443:9443 --name portainer --restart=always \
    -v /var/run/docker.sock:/var/run/docker.sock \
    -v portainer_data:/data portainer/portainer-ce:latest
```

Access the Portainer web interface:

- Open a browser and go to:
  `https://<YOUR_SERVER_IP>:9443`
- Create an **admin account** on the first login.

---

4. Useful Docker & Portainer Commands

Check running containers:

```
docker ps
```

Start/stop a container:

```
docker start portainer
docker stop portainer
```

Update Portainer to the latest version:

```
docker stop portainer
docker rm portainer
docker pull portainer/portainer-ce:latest
docker run -d -p 8000:8000 -p 9443:9443 --name portainer --restart=always \
    -v /var/run/docker.sock:/var/run/docker.sock \
    -v portainer_data:/data portainer/portainer-ce:latest
```

Sécuriser et Configurer un VPS avec UFW & Nginx Proxy Manager

□□ Objectifs

□ Bloquer **tous les accès extérieurs** sauf :

- **HTTP (80) & HTTPS (443)** pour tout le monde (pour les noms de domaine).
- **SSH et autres ports sensibles** accessibles uniquement depuis **tes IPs autorisées**.
  □ Installer **Docker** et **Nginx Proxy Manager**.
  □ Pointer **ton domaine** vers ton VPS pour gérer les accès.

1□ Configurer le pare-feu UFW

1.1 - Installer UFW (si ce n'est pas fait)

```
apt update && apt install ufw -y
```

1.2 - Bloquer tout trafic entrant par défaut

```
ufw default deny incoming
ufw default allow outgoing
```

1.3 - Autoriser uniquement tes IPs à accéder aux services sensibles

Remplace **100.100.10.11**, **100.100.10.12**, **100.100.10.13** par tes **adresses IP autorisées**.

```
ufw allow from 100.100.10.11
ufw allow from 100.100.10.12
```

```
ufw allow from 100.100.10.13
```

1.4 - Autoriser uniquement les ports nécessaires

□□ Autoriser HTTP (80), HTTPS (443) et l'interface de Nginx Proxy Manager (81) pour tout le monde :

```
ufw allow 80   # HTTP
ufw allow 443  # HTTPS
ufw allow 81   # Interface Nginx Proxy Manager
```

□□ Autoriser SSH uniquement pour tes IPs privées :

```
ufw allow from 100.100.10.11 to any port 22
ufw allow from 100.100.10.12 to any port 22
ufw allow from 100.100.10.13 to any port 22
```

1.5 - Empêcher UFW de bloquer le traffic interne

```
ufw allow from 172.16.0.0/12
```

1.6 - Activer UFW et vérifier les règles

```
ufw enable
ufw status verbose
```

□□ Ton serveur est maintenant sécurisé :

- **Tout est bloqué** sauf **les ports web (80, 443, 81)** et **l'accès SSH restreint** à tes IPs.

2□ Installer et Configurer Nginx Proxy Manager

2.1 - Créer un dossier pour Nginx Proxy Manager

```
mkdir -p /opt/npm && cd /opt/npm
```

2.2 - Créer le fichier `docker-compose.yml`

```
nano docker-compose.yml
```

Ajoute ce contenu :

```
services:
  npm:
    image: 'jc21/nginx-proxy-manager:latest'
    container_name: nginx_proxy_manager
    restart: unless-stopped
    ports:
      - "80:80"     # HTTP
      - "443:443"   # HTTPS
      - "81:81"     # Interface d'administration
    volumes:
      - ./data:/data
      - ./letsencrypt:/etc/letsencrypt
```

2.3 - Démarrer Nginx Proxy Manager

```
docker-compose up -d
```

2.4 - Vérifier que le conteneur tourne

```
docker ps
```

---

3 Configurer le Domaine

3.1 - Ajouter les enregistrements DNS

Sur ton **registrar (OVH, Cloudflare, Namecheap, etc.)**, configure **les entrées DNS** :

- **A Record** pour pointer vers l'IP du VPS :

  ```
  example.com → 100.100.10.10
  ```

- **CNAME Record** (si tu veux un sous-domaine) :

  ```
  www.example.com → example.com
  ```

Attends quelques minutes/heures que la propagation DNS se fasse.

4 Configurer les Redirections dans Nginx Proxy Manager

4.1 - Accéder à l'interface web

Ouvre **http://100.100.10.10:81** dans ton navigateur.

4.2 - Se connecter

Identifiants par défaut :

- **Email** : admin@example.com
- **Mot de passe** : changeme

⚠ **Change immédiatement ton mot de passe !**

4.3 - Ajouter une redirection avec SSL

1 **Dans Nginx Proxy Manager**, va dans Proxy Hosts et clique sur Add Proxy Host.
2 **Ajoute un domaine** :

- **Domain Name** : example.com
- **Forward Hostname/IP** : l'IP locale de ton service
- **Forward Port** : le port du service (ex : 3000 pour une app, 8080 pour un serveur web, etc.)
- **Access List** : Laisse vide (optionnel)
  3 **Active SSL** avec Let's Encrypt :
- **Coche "Enable SSL"**
- **Coche "Force SSL"**
- **Clique sur "Request a new SSL Certificate"**

Après validation, accède à https://example.com et teste !

---

5 Tester et Vérifier

Vérifie que **https://example.com** **fonctionne** bien.
Vérifie que **l'accès SSH est bien restreint** à tes IPs autorisées :

```
ufw status verbose
```

 Vérifie que **les services Docker tournent** correctement :

```
docker ps
```

 Teste si **l’IP du serveur est bloquée** en essayant d’accéder directement à `http://100.100.10.10`.

□□ Résumé Final

□□ Sécurité	□□ Accès Web	□□ Outils
SSH limité à tes IPs	**Sites accessibles via noms de domaine**	**Docker installé**
Ports 80, 443, 81 ouverts pour tout le monde	**SSL automatique avec Let's Encrypt**	**Nginx Proxy Manager opérationnel**
Tous les autres ports bloqués	**Redirections faciles via Proxy Manager**	**Pare-feu UFW actif**

Commandes utiles pour UFW

Liste des règles

```
ufw status numbered
```

Effacer une règle par son numéro (par ex 7)

```
ufw delete 7
```