



Server

Administration serveur

Thibault Six

9 chapitres

SUPPORTS DE COURS

<https://cours.tsix.be/cours/administration-serveur>

Version 1 · 12/09/2026

PRÉSENTATION

Administration serveur

Contenu

- Configuration d'un VPS
 - Utilisateurs
 - Fail2ban
 - Firewall
- Hébergement statique
- Hébergement node.js avec PM2
- Hébergement PHP
 - Installation
 - PHP
 - MySQL
 - Nginx
 - Configuration
 - Nginx
 - DB
 - Certbot
 - Déploiement
- Configuration d'un hébergement mutualisé
 - Laravel
 - Wordpress
 - Statique

Sommaire

- Administration serveur
- 1 VPS
 - 1.1 Utilisateurs
 - 1.2 Firewall
 - 1.3 Nginx
 - 1.4 Hébergement statique
 - 1.5 MariaDB
 - 1.6 Hébergement PHP
 - 1.7 Certbot
 - 1.8 Deployer

CHAPITRE 1

Introduction

Dans ce chapitre, nous allons nous concentrer sur l'un des aspects les plus importants de l'administration des systèmes : les Serveurs Privés Virtuels (VPS). Un VPS est une machine virtuelle vendue en tant que service par un hébergeur web. Il fonctionne de manière indépendante, tout comme un ordinateur physique, et est utilisé pour exécuter des applications et des services.

Nous allons explorer en détail les différentes facettes de l'administration d'un VPS. Nous commencerons par comprendre les concepts fondamentaux, comme la gestion des utilisateurs et la configuration du pare-feu. Ensuite, nous nous plongerons dans des sujets plus avancés, tels que l'installation et la configuration de services spécifiques comme Nginx, MariaDB ou encore l'hébergement d'applications PHP et Node.js.

Nous aborderons également des outils essentiels pour la sécurité et le monitoring de votre serveur, comme Fail2Ban et les différentes méthodes de surveillance de votre système. Enfin, nous verrons comment déployer vos applications et gérer les certificats SSL avec Certbot.

C'est un voyage passionnant qui vous attend, alors commençons sans plus attendre notre exploration des VPS.

1.1 Utilisateurs

Introduction à SSH

SSH, acronyme de Secure Shell, est un protocole de communication sécurisé largement utilisé pour administrer et communiquer avec des serveurs à distance. Il fournit un canal de communication crypté entre votre machine locale et un serveur distant, ce qui permet d'exécuter des commandes et de transférer des fichiers de manière sécurisée.

Une des caractéristiques clés de SSH est l'utilisation de paires de clés pour l'authentification. Une paire de clés SSH se compose d'une clé publique, qui peut être partagée librement, et d'une clé privée, qui doit être gardée secrète. Lorsque vous vous connectez à un serveur SSH, le serveur utilise la clé publique pour créer un message crypté que seule la clé privée correspondante peut déchiffrer. Si vous pouvez déchiffrer le message et le renvoyer au serveur, le serveur sait que vous êtes en possession de la clé privée et vous autorise à vous connecter.

L'utilisation de clés SSH offre plusieurs avantages par rapport à l'authentification par mot de passe. Tout d'abord, elle permet une authentification sans mot de passe, ce qui est plus pratique et potentiellement plus sûr si votre clé privée est bien protégée. Deuxièmement, elle permet une authentification à deux facteurs lorsque vous utilisez à la fois une clé SSH et un mot de passe. Enfin, elle permet de limiter les connexions SSH à certaines clés publiques, ce qui peut renforcer la sécurité de votre serveur.

Première connexion SSH au serveur

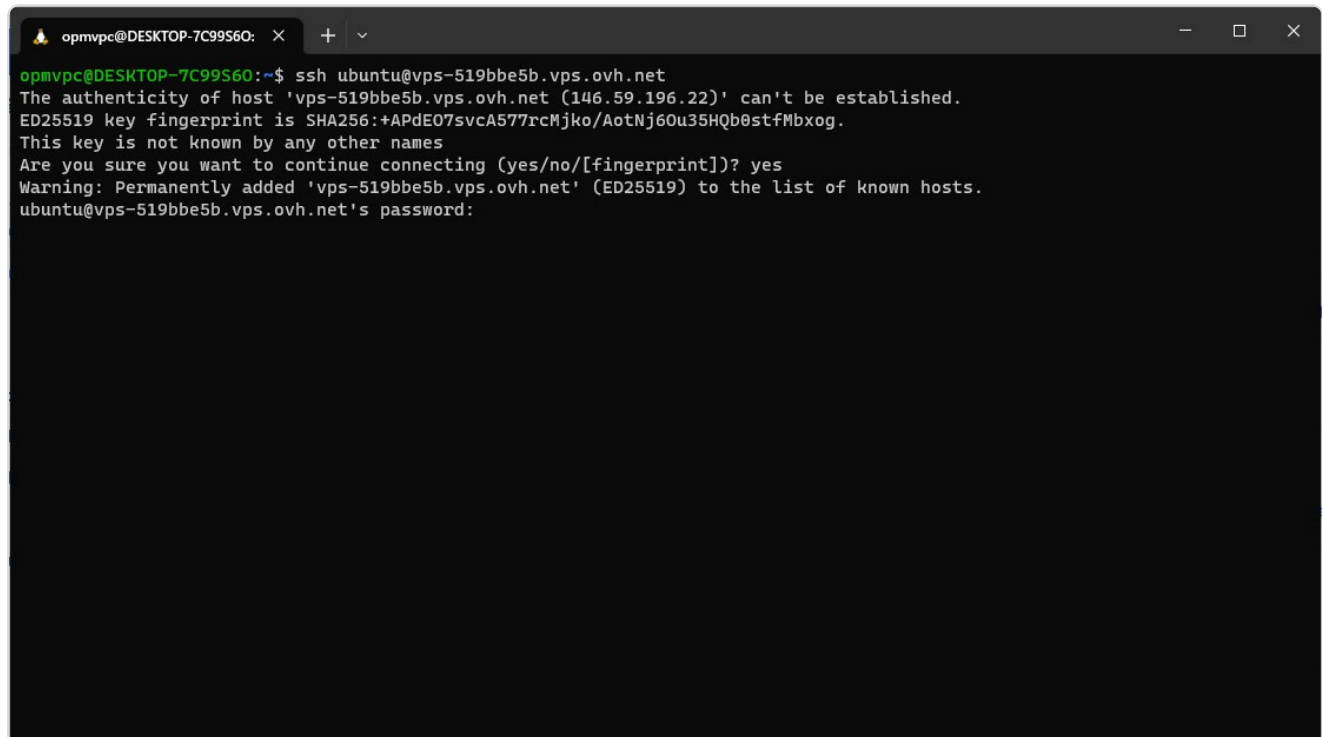
Lorsque vous installez un nouveau VPS, le fournisseur vous envoie généralement les informations de connexion initiales par email. Ces informations comprennent l'adresse IP du serveur, le nom d'utilisateur (habituellement `root` ou `admin`) et un mot de passe.

Pour vous connecter à votre serveur pour la première fois, vous pouvez utiliser la commande `ssh` dans votre terminal. Par exemple :

```
ssh root@your_server_ip
```

BASH

Remplacez `root` par le nom d'utilisateur fourni et `your_server_ip` par l'adresse IP ou le nom d'hôte de votre serveur.



```
opmvpc@DESKTOP-7C99S60: x + v
opmvpc@DESKTOP-7C99S60:~$ ssh ubuntu@vps-519bbe5b.vps.ovh.net
The authenticity of host 'vps-519bbe5b.vps.ovh.net (146.59.196.22)' can't be established.
ED25519 key fingerprint is SHA256:+APdE07svcA577rcMjko/AotNj60u35HQb0stfMbxog.
This key is not known by any other names
Are you sure you want to continue connecting (yes/no/[fingerprint])? yes
Warning: Permanently added 'vps-519bbe5b.vps.ovh.net' (ED25519) to the list of known hosts.
ubuntu@vps-519bbe5b.vps.ovh.net's password:
```

La première fois que vous vous connectez à un serveur, SSH vous demandera de vérifier l'empreinte digitale du serveur. Si vous acceptez, cette empreinte digitale sera stockée et utilisée pour vérifier l'identité du serveur lors des connexions futures.

Mise à jour du système

Une fois que vous avez établi une connexion sécurisée à votre serveur via SSH et que vous avez configuré vos utilisateurs et leurs clés, la première tâche à effectuer est de mettre à jour le système. Cette étape est cruciale pour assurer la sécurité et la stabilité de votre serveur.

Pourquoi mettre à jour le système ?

Les mises à jour du système d'exploitation sont essentielles pour plusieurs raisons :

1. **Sécurité** : Les mises à jour contiennent souvent des correctifs pour des vulnérabilités de sécurité découvertes. En gardant votre système à jour, vous minimisez le risque d'être compromis par des attaquants.
2. **Stabilité** : Les mises à jour apportent également des corrections de bugs qui peuvent affecter la stabilité du système.
3. **Fonctionnalités** : Les nouvelles versions des logiciels peuvent offrir de nouvelles fonctionnalités ou améliorations.

Comment mettre à jour le système ?

Sur un serveur VPS basé sur Ubuntu, vous pouvez utiliser les commandes suivantes pour mettre à jour le système :

```
sudo apt-get update  
sudo apt-get upgrade
```

BASH

La commande `sudo apt-get update` met à jour la liste des paquets disponibles à partir des dépôts configurés dans le système.

La commande `sudo apt-get upgrade` installe les nouvelles versions des paquets installés sur le système.

```

ubuntu@vps-519bbe5b: ~
60 updates can be applied immediately.
46 of these updates are standard security updates.
To see these additional updates run: apt list --upgradable

ubuntu@vps-519bbe5b:~$ sudo apt-get update
Hit:1 http://security.ubuntu.com/ubuntu lunar-security InRelease
Hit:2 http://nova.clouds.archive.ubuntu.com/ubuntu lunar InRelease
Hit:3 http://nova.clouds.archive.ubuntu.com/ubuntu lunar-updates InRelease
Hit:4 http://nova.clouds.archive.ubuntu.com/ubuntu lunar-backports InRelease
Reading package lists... Done
ubuntu@vps-519bbe5b:~$ sudo apt-get upgrade
Reading package lists... Done
Building dependency tree... Done
Reading state information... Done
Calculating upgrade... Done
The following packages have been kept back:
  linux-headers-generic linux-headers-virtual linux-image-virtual linux-virtual sosreport
The following packages will be upgraded:
  apparmor apt apt-utils bind9-dnsutils bind9-host bind9-libs cloud-init curl gcc-13-base inetutils-telnet
  initramfs-tools initramfs-tools-bin initramfs-tools-core libapparmor1 libapt-pkg6.0 libcap2 libcap2-bin
  libcurl3-gnutls libcurl4 libgcc-s1 libpam-cap libperl5.36 libpython3.11 libpython3.11-minimal libpython3.11-stdlib
  libssh-4 libstdc++6 libx11-6 libx11-data libxml2 needrestart open-vm-tools openssh-client openssh-server
  openssh-sftp-server perl perl-base perl-modules-5.36 python3-requests python3.11 python3.11-minimal qemu-guest-agent
  telnet ubuntu-advantage-tools ufw vim vim-common vim-runtime vim-tiny xxd
50 upgraded, 0 newly installed, 0 to remove and 5 not upgraded.
Need to get 4214 kB/39.3 MB of archives.
After this operation, 105 kB of additional disk space will be used.
Do you want to continue? [Y/n]

```

Il est recommandé de redémarrer le serveur après une mise à jour, surtout si le noyau du système ou des services importants ont été mis à jour. Vous pouvez le faire en utilisant la commande suivante :

```
sudo reboot
```

BASH

Modification du mot de passe de root

Le compte root est le compte d'administrateur par défaut sur les systèmes Linux. Il possède des privilèges illimités et peut effectuer n'importe quelle opération sur le système. Par conséquent, il est essentiel de protéger ce compte avec un mot de passe fort.

Comment modifier le mot de passe de root ?

Pour modifier le mot de passe de root, vous pouvez utiliser les commandes suivantes :

```
sudo su
passwd
```

BASH

La première commande, `sudo su`, vous permet de passer au compte root. La seconde commande, `passwd`, vous invite à entrer un nouveau mot de passe pour le compte root.

Sur une distribution Linux, le mot de passe que vous tapez n'apparaîtra pas pour des raisons de sécurité. Vous devrez entrer le mot de passe deux fois pour confirmer que vous l'avez tapé correctement.

Gestionnaire de mots de passe

Il est fortement recommandé d'utiliser un générateur de mots de passe pour créer un mot de passe fort et unique pour le compte root. Les générateurs de mots de passe créent des mots de passe qui sont difficiles à deviner ou à craquer.

De plus, utiliser un gestionnaire de mots de passe peut vous aider à gérer vos mots de passe. Un gestionnaire de mots de passe stocke vos mots de passe de manière sécurisée et vous permet de les récupérer facilement lorsque vous en avez besoin.

<https://bitwarden.com/> - Gestionnaire de mot de passe open-source.

Création des utilisateurs

Une fois que vous avez une compréhension de base de SSH, vous pouvez commencer à configurer votre serveur. La première étape consiste à créer des utilisateurs.

Utilisateur `admin`

Sur votre serveur, vous pouvez utiliser la commande `adduser` pour créer un nouvel utilisateur. Par exemple, pour créer un utilisateur nommé `admin`, vous pouvez utiliser la commande suivante :

```
sudo adduser admin
```

BASH

Cette commande crée un nouvel utilisateur et un groupe avec le même nom, ainsi qu'un répertoire personnel pour l'utilisateur.

Utilisateur `deployer`

De la même manière, pour créer un utilisateur dédié aux déploiements, nommé `deployer`, utilisez la commande suivante :

```
sudo adduser deployer
```

BASH

Ces deux utilisateurs auront des droits standard sur le système. Si vous avez besoin qu'ils aient des droits d'administrateur, vous devrez les ajouter `sudoers`, ce que nous aborderons dans une section ultérieure.

Il est important d'avoir des utilisateurs séparés pour différentes tâches sur votre serveur. Cela permet d'isoler les droits et les responsabilités, ce qui peut améliorer la sécurité et l'organisation de votre serveur.

Gestion des droits d'administrateur

Dans le cadre de l'administration d'un serveur, certains utilisateurs peuvent avoir besoin de droits d'administrateur pour accomplir certaines tâches. Dans Ubuntu et la plupart des autres distributions Linux, ces privilèges peuvent être accordés en ajoutant l'utilisateur au groupe `sudo`.

Ajout des utilisateurs au groupe `sudo`

Au lieu de modifier directement le fichier `sudoers`, qui peut être délicat et risqué, nous allons utiliser la commande `usermod` pour ajouter nos utilisateurs au groupe `sudo`. Les membres de ce groupe ont la capacité d'exécuter des commandes en tant que root via la commande `sudo`.

Voici comment ajouter les utilisateurs `admin` et `deployer` au groupe `sudo` :

```
sudo usermod -aG sudo admin
sudo usermod -aG sudo deployer
```

BASH

La commande `usermod -aG sudo userName` ajoute l'utilisateur spécifié (dans ce cas, `admin` et `deployer`) au groupe `sudo`.

Implications en termes de sécurité et de contrôle

Donner à un utilisateur des privilèges d'administrateur est une décision importante qui a des implications en termes de sécurité et de contrôle. Un utilisateur avec des privilèges d'administrateur peut effectuer des changements majeurs sur le système, y compris des actions qui peuvent rendre le système instable ou vulnérable aux attaques.

Il est donc essentiel de ne donner des privilèges d'administrateur qu'aux utilisateurs qui en ont besoin pour leurs tâches, et de s'assurer qu'ils comprennent les responsabilités qui accompagnent ces privilèges.

Gestion des clés SSH

Maintenant que vous avez vos utilisateurs, vous voulez probablement être en mesure de vous connecter à votre serveur via SSH. Pour ce faire, vous devez générer une paire de clés SSH sur votre machine locale et copier la clé publique sur votre serveur.

Génération d'une paire de clés SSH

Sur votre machine locale (Ubuntu 22 avec WSL), vous pouvez utiliser la commande `ssh-keygen` pour générer une nouvelle paire de clés SSH. Par exemple :

```
ssh-keygen -t rsa -b 4096 -C "your_email@example.com"
```

BASH

Cette commande génère une nouvelle paire de clés RSA de 4096 bits. Vous pouvez remplacer `"your_email@example.com"` par votre propre adresse e-mail ou tout autre identifiant que vous souhaitez utiliser.

Lors de la génération des clés, vous pourrez les nommer comme vous le désirez. Je vous conseille de donner un nom composé des informations importantes. Par exemple, `[nom_utilisateur]-[nom_serveur]-[nom_ordinateur_client]`.

<https://manpages.ubuntu.com/manpages/lunar/en/man1/ssh-keygen.1.html>

Copie des clés SSH sur le serveur

Une fois que vous avez une paire de clés, vous devez copier la clé publique sur votre serveur. Vous pouvez le faire avec la commande `ssh-copy-id`. Par exemple, pour copier votre clé publique sur le compte `admin` sur votre serveur, vous pouvez utiliser la commande suivante :

```
ssh-copy-id -i ~/.ssh/nomDeLaClé admin@your_server_ip
```

BASH

De même, pour copier la clé publique sur le compte `deployer`, utilisez la commande suivante :

```
ssh-copy-id -i ~/.ssh/nomDeLaClé deployer@your_server_ip
```

BASH

Remplacez `your_server_ip` par l'adresse IP ou le nom d'hôte de votre serveur.

Une fois que vous avez copié vos clés publiques, vous devriez être en mesure de vous connecter à votre serveur avec la commande `ssh` sans avoir besoin de mot de passe. Par exemple :

```
ssh admin@your_server_ip
```

BASH

ou

```
ssh deployer@your_server_ip
```

BASH

Utilisation de SSH Agent et du fichier .ssh/config

Lorsque vous travaillez avec plusieurs serveurs distants et différents utilisateurs, la gestion des clés SSH peut devenir complexe. Pour simplifier ce processus, vous pouvez utiliser l'agent SSH et le fichier de configuration SSH.

Qu'est-ce que l'agent SSH ?

L'agent SSH est un programme en arrière-plan qui gère vos clés privées SSH pour vous. Il garde vos clés privées en mémoire et utilise les clés pour se connecter aux serveurs distants sans avoir à taper votre mot de passe à chaque fois.

Comment utiliser l'agent SSH ?

Voici une série de commandes qui illustrent comment utiliser l'agent SSH :

1. **Démarrer l'agent SSH** : L'agent SSH doit être en cours d'exécution pour pouvoir utiliser vos clés. Vous pouvez démarrer l'agent SSH avec la commande suivante :

```
eval $(ssh-agent)
```

BASH

2. **Ajouter votre clé privée à l'agent** : Pour que l'agent SSH puisse utiliser votre clé privée, vous devez l'ajouter à l'agent. Vous pouvez le faire avec la commande suivante :

```
ssh-add /chemin/vers/la/clé/privée
```

BASH

3. **Ajouter plusieurs clés** : Vous pouvez ajouter plusieurs clés à l'agent SSH. Par exemple :

```
ssh-add ~/.ssh/id_rsa  
ssh-add ~/.ssh/id_rsa_entreprise
```

BASH

4. **Vérifier les clés ajoutées** : Vous pouvez vérifier quelles clés sont actuellement gérées par l'agent SSH avec la commande suivante :

```
ssh-add -l
```

BASH

Qu'est-ce que le fichier .ssh/config ?

Le fichier .ssh/config est un fichier de configuration qui permet de définir des paramètres pour chaque hôte, ce qui facilite la connexion à différents serveurs.

Comment utiliser le fichier .ssh/config ?

Pour utiliser le fichier .ssh/config, vous devez créer ou modifier le fichier dans votre répertoire ~/.ssh . Vous pouvez le faire avec la commande suivante :

```
touch ~/.ssh/config
```

BASH

Ensuite, vous pouvez éditer le fichier pour définir les paramètres pour chaque hôte. Voici un exemple de configuration :

```
Host admin.serveur.com
    HostName serveur.com
    User admin
    IdentitiesOnly yes
    IdentityFile ~/.ssh/admin-vps-pc

Host deployer.serveur.com
    HostName serveur.com
    User deployer
    IdentitiesOnly yes
    IdentityFile ~/.ssh/deployer-vps-pc
```

Dans cet exemple :

- **Host** : C'est le nom que vous utiliserez pour vous connecter au serveur.
- **HostName** : C'est le nom d'hôte réel ou l'adresse IP du serveur.
- **User** : C'est le nom d'utilisateur que vous utiliserez pour vous connecter au serveur.
- **IdentitiesOnly yes** : Cette option indique à SSH de n'utiliser que les clés indiquées dans le fichier de configuration.
- **IdentityFile** : C'est le chemin vers le fichier de clé privée que vous utiliserez pour vous connecter au serveur.

Grâce à l'agent SSH et le fichier .ssh/config, vous pouvez gérer vos connexions SSH de manière plus efficace et sécurisée.

Maintenant, pour vous connecter au serveur en tant qu'administrateur avec cette commande :

```
ssh admin.serveur.com
```

BASH

La connexion se fait maintenant avec votre clé SSH, plus besoin de mot de passe !

Désactivation de la connexion par mot de passe et sécurité des clés SSH

Lorsqu'il s'agit de sécuriser les connexions SSH à votre serveur, l'utilisation de clés SSH est souvent considérée comme plus sûre que l'utilisation de mots de passe. Une des raisons principales est la résistance des clés SSH aux attaques de force brute, qui sont une menace courante pour les serveurs accessibles sur Internet.

Pourquoi les clés SSH sont-elles plus sûres ?

Une clé SSH est une paire de clés cryptographiques, l'une publique et l'autre privée. La clé privée reste sur votre machine locale et doit rester secrète. La clé publique peut être partagée et est ajoutée au serveur. Lorsque vous vous connectez au serveur, il utilise la clé publique pour crypter un message qui ne peut être déchiffré qu'avec la clé privée correspondante.

Par conséquent, même si un attaquant réussit à intercepter la communication, il ne pourra pas déchiffrer le message et donc ne pourra pas se connecter au serveur. De plus, une clé SSH est généralement beaucoup plus longue qu'un mot de passe, ce qui la rend beaucoup plus difficile à craquer par force brute.

Comment désactiver la connexion par mot de passe ?

Pour améliorer la sécurité de votre serveur, une fois que vous avez configuré la connexion par clé SSH, vous pouvez désactiver la connexion par mot de passe. Pour ce faire, vous devez modifier le fichier de configuration SSH sur le serveur.

1. Ouvrez le fichier de configuration SSH avec la commande suivante :

```
sudo nano /etc/ssh/sshd_config
```

BASH

2. Recherchez la ligne qui contient `#PasswordAuthentication yes`, décommentez-la en supprimant le `#` au début et changez `yes` en `no`. Cela devrait ressembler à ceci :

```
PasswordAuthentication no
```

3. Enregistrez et fermez le fichier.
4. Redémarrez le service SSH pour que les modifications prennent effet :

```
sudo service ssh restart
```

BASH

En désactivant la connexion par mot de passe, vous réduisez le risque d'attaques de force brute réussies et augmentez la sécurité de votre serveur.

Attention, si vous pouvez vous connecter à un utilisateur qui ne possède pas de clé ssh, il existe sans doute une autre configuration qui vient écraser le paramètre `PasswordAuthentication`. Vous pouvez chercher dans le dossier `/etc/ssh/sshd_config.d/` et modifier les fichiers si besoin.

Restriction de la connexion SSH à certains utilisateurs

Pour renforcer la sécurité de votre serveur, vous pouvez limiter les utilisateurs qui peuvent se connecter via SSH. Cela peut être réalisé en modifiant le fichier de configuration du démon SSH (`sshd_config`).

Limiter les utilisateurs avec la directive `AllowUsers`

La directive `AllowUsers` permet de spécifier explicitement quels utilisateurs sont autorisés à se connecter via SSH. Pour l'utiliser, ouvrez le fichier `sshd_config` avec des privilèges root ou sudo :

```
sudo nano /etc/ssh/sshd_config
```

BASH

Recherchez la directive `AllowUsers` dans le fichier. Si elle n'existe pas, vous pouvez la créer n'importe où dans le fichier. Après la directive, listez les utilisateurs qui devraient être autorisés à se connecter via SSH :

```
AllowUsers user1 user2
```

Enregistrez et fermez le fichier. Redémarrez le démon pour appliquer vos modifications :

```
sudo service ssh restart
```

BASH

Limiter les utilisateurs avec la directive `AllowGroups`

Si vous préférez gérer les utilisateurs en groupes, vous pouvez utiliser la directive `AllowGroups`. Cette directive vous permet de spécifier un groupe d'utilisateurs autorisés à se connecter via SSH.

Enregistrez et fermez le fichier. Créez d'abord le groupe système correspondant (sans répertoire personnel) :

```
sudo groupadd -r sshmembers
```

BASH

Assurez-vous d'ajouter tous les utilisateurs nécessaires à ce groupe. Cela peut être fait en tapant :

BASH

```
sudo usermod -aG sshmembers user1  
sudo usermod -aG sshmembers user2
```

Ensuite, dans le fichier `sshd_config`, ajoutez la directive `AllowGroups` suivie du nom du groupe :

```
AllowGroups sshmembers
```

Redémarrez le démon SSH pour appliquer vos modifications :

BASH

```
sudo service ssh restart
```

En limitant les utilisateurs qui peuvent se connecter via SSH, vous renforcez la sécurité de votre serveur en réduisant le nombre de points d'entrée possibles pour un attaquant.

Désactivation de la connexion SSH pour l'utilisateur root

Dans le cadre de la sécurisation de votre serveur, il est souvent recommandé de désactiver complètement la connexion SSH pour l'utilisateur root une fois que vous avez configuré un compte utilisateur SSH avec des privilèges sudo.

Pourquoi désactiver la connexion SSH pour l'utilisateur root ?

L'utilisateur root a des privilèges illimités et peut effectuer n'importe quelle opération sur le système. Si un attaquant parvient à se connecter en tant que root, il pourrait causer des dommages irréparables. Par conséquent, pour minimiser le risque, il est préférable de désactiver la connexion SSH pour l'utilisateur root.

Comment désactiver la connexion SSH pour l'utilisateur root ?

Pour désactiver la connexion SSH pour l'utilisateur root, vous devez modifier le fichier de configuration du démon SSH (`sshd_config`). Voici comment faire :

1. Ouvrez le fichier `sshd_config` avec des privilèges root ou sudo :

BASH

```
sudo nano /etc/ssh/sshd_config
```

2. Recherchez la directive `PermitRootLogin` . Si elle est commentée, décommentez-la. Changez la valeur à `no` :

```
PermitRootLogin no
```

3. Enregistrez et fermez le fichier. Redémarrez le démon SSH pour appliquer vos modifications :

```
sudo service ssh restart
```

BASH

En désactivant la connexion SSH pour l'utilisateur root, vous ajoutez une couche supplémentaire de sécurité à votre serveur.

Suppression de l'utilisateur fourni par le fournisseur de service cloud

Lorsque vous configurez un serveur sur un fournisseur de services cloud, il est courant que le fournisseur crée un utilisateur par défaut avec des privilèges d'administration. Une fois que vous avez terminé vos configurations et que vous avez créé vos propres utilisateurs, il est recommandé de supprimer cet utilisateur fourni par le fournisseur pour renforcer la sécurité de votre serveur.

Suppression de l'utilisateur

Une fois que vous avez vérifié l'existence de l'utilisateur fourni par le fournisseur, vous pouvez procéder à sa suppression. Utilisez la commande suivante pour supprimer l'utilisateur :

```
sudo userdel -r ubuntu
```

BASH

Assurez-vous de remplacer "ubuntu" par le nom d'utilisateur fourni par votre fournisseur de services cloud. L'option `-r` supprime également le répertoire personnel de l'utilisateur.

Conclusion

Supprimer l'utilisateur fourni par le fournisseur de services cloud une fois que vous avez terminé vos configurations est une étape importante pour renforcer la sécurité de votre serveur. Assurez-vous de configurer vos propres utilisateurs avec des privilèges appropriés avant de supprimer l'utilisateur fourni par le fournisseur.

Bonnes pratiques

La gestion des utilisateurs et des clés SSH est un aspect essentiel de l'administration d'un serveur. Voici quelques bonnes pratiques à garder à l'esprit :

Protection des clés SSH par mot de passe

Lorsque vous générez une nouvelle paire de clés SSH, `ssh-keygen` vous demande si vous souhaitez protéger la clé privée avec un mot de passe. Il est fortement recommandé d'utiliser un mot de passe. Cela ajoute une couche de sécurité supplémentaire : même si quelqu'un parvient à obtenir votre clé privée, il ne pourra pas l'utiliser sans le mot de passe.

Authentification par clés plutôt que par mot de passe

L'authentification par clés est généralement plus sûre et plus pratique que l'authentification par mot de passe. Avec l'authentification par clés, vous n'avez pas à vous souvenir d'un mot de passe et vous n'avez pas à taper le mot de passe à chaque fois que vous vous connectez. De plus, l'authentification par clés est résistante aux attaques de force brute qui tentent de deviner votre mot de passe.

Pour désactiver l'authentification par mot de passe et forcer l'authentification par clés, vous pouvez modifier le fichier de configuration SSH sur votre serveur (`/etc/ssh/sshd_config`) et définir la directive `PasswordAuthentication` sur `no`.

Permissions appropriées pour les fichiers de clés SSH

SSH est très pointilleux sur les permissions des fichiers de clés. Si les permissions ne sont pas correctes, SSH peut refuser de vous laisser vous connecter. Les permissions correctes sont généralement les suivantes :

- Le répertoire `~/.ssh` sur votre machine locale doit avoir les permissions `700` (c'est-à-dire, accessible en lecture, écriture et exécution uniquement par le propriétaire).
- Le fichier `~/.ssh/id_rsa` (ou tout autre fichier de clé privée) sur votre machine locale doit avoir les permissions `600` (c'est-à-dire, accessible en lecture et écriture uniquement par le propriétaire).
- Le fichier `~/.ssh/authorized_keys` sur le serveur doit également avoir les permissions `600`.

Vous pouvez définir ces permissions avec la commande `chmod`. Par exemple :

```
chmod 700 ~/.ssh
chmod 600 ~/.ssh/id_rsa
```

BASH

En suivant ces bonnes pratiques, vous pouvez améliorer la sécurité et l'efficacité de votre administration serveur.

Sources :

1. [SSH Essentials: Working with SSH Servers, Clients, and Keys](#) : Ce guide explique les bases de SSH, y compris la génération de clés SSH, la connexion à un serveur SSH et la configuration des autorisations de clés.
2. [Initial Server Setup with Ubuntu 22.04](#) : Ce guide explique les étapes initiales de configuration d'un serveur Ubuntu, y compris la création d'un nouvel utilisateur, l'ajout de l'utilisateur au groupe sudo et la configuration du pare-feu.
3. https://help.ovhcloud.com/csm/fr-dedicated-servers-getting-started-dedicated-server?id=kb_article_view&sysparm_article=KB0043481
4. https://help.ovhcloud.com/csm/fr-vps-security-tips?id=kb_article_view&sysparm_article=KB0047708
5. <https://manpages.ubuntu.com/manpages/lunar/en/man1/ssh-keygen.1.html>
6. <https://www.ssh.com/academy/ssh/copy-id>
7. <https://linuxize.com/post/how-to-delete-users-in-linux-using-the-userdel-command/>
8. <https://gist.github.com/bonnopc/c78920431284ce3fc2a5270016205116>
9. <https://saturncloud.io/blog/could-not-open-a-connection-to-your-authentication-agent/>
10. <https://docs.github.com/en/authentication/connecting-to-github-with-ssh/using-ssh-agent-forwarding>

1.2 Pare-feu pour les applications web

Introduction au pare-feu

Dans le domaine de la sécurité informatique, un pare-feu joue un rôle essentiel pour protéger les serveurs et les applications web contre les menaces extérieures. Un pare-feu permet de contrôler le trafic réseau entrant et sortant, en autorisant uniquement les connexions aux services spécifiés. Dans ce chapitre, nous allons explorer l'utilisation du pare-feu UFW (Uncomplicated Firewall) pour sécuriser un serveur Ubuntu 22.04 utilisé pour des applications web.

Présentation de UFW (Uncomplicated Firewall)

UFW est une interface utilisateur simplifiée pour iptables, le pare-feu par défaut dans de nombreuses distributions Linux. Il offre une approche conviviale pour configurer et gérer les règles du pare-feu, permettant aux administrateurs système de contrôler facilement le trafic réseau sur leur serveur.

Les avantages de l'utilisation de UFW sont nombreux. Tout d'abord, il simplifie la syntaxe complexe d'iptables en utilisant des commandes plus intuitives. De plus, UFW permet de créer des profils d'application préconfigurés, ce qui facilite la gestion des règles pour des services spécifiques tels que les applications web.

Installation et configuration de UFW

Avant de commencer à utiliser UFW, il est nécessaire de l'installer sur votre serveur s'il ne l'est pas encore. Pour cela, vous pouvez exécuter la commande suivante :

```
sudo apt install ufw
```

BASH

Une fois UFW installé, vous pouvez commencer à configurer les règles du pare-feu en utilisant les commandes fournies par UFW. Par exemple, pour autoriser les connexions SSH afin de pouvoir vous connecter à votre serveur, vous pouvez utiliser la commande suivante :

```
sudo ufw allow OpenSSH
```

BASH

Gestion des profils d'application avec UFW

UFW permet de gérer les profils d'application, ce qui simplifie la configuration des règles pour des services spécifiques. Les profils d'application sont des fichiers de configuration qui définissent les règles de pare-feu pour une application donnée. Par exemple, le profil OpenSSH est automatiquement enregistré avec UFW lors de l'installation d'OpenSSH sur le serveur.

Pour afficher la liste des profils d'application disponibles, vous pouvez exécuter la commande suivante :

```
sudo ufw app list
```

BASH

Cette commande affichera les applications pour lesquelles des profils ont été enregistrés avec UFW. Dans notre cas, nous verrons le profil OpenSSH qui permet de gérer les règles de pare-feu pour les connexions SSH.

Activation et désactivation du pare-feu

Une fois que vous avez configuré les règles du pare-feu, vous devez les activer pour qu'elles prennent effet. Utilisez la commande suivante pour activer le pare-feu UFW :

BASH

```
sudo ufw enable
```

Lorsque vous activez le pare-feu, toutes les règles que vous avez configurées seront appliquées. Vous pouvez vérifier l'état du pare-feu en utilisant la commande suivante :

BASH

```
sudo ufw status
```

Cette commande affichera l'état du pare-feu ainsi que les règles actuellement appliquées. Assurez-vous que les connexions nécessaires aux applications web sont autorisées.

Si vous avez besoin de désactiver temporairement le pare-feu pour effectuer des modifications ou pour d'autres raisons, vous pouvez utiliser la commande suivante :

BASH

```
sudo ufw disable
```

N'oubliez pas de réactiver le pare-feu une fois que vous avez terminé en utilisant la commande `sudo ufw enable`.

1.3 Nginx

Introduction à Nginx

Nginx est un serveur web léger, performant et open-source. Il est largement utilisé pour servir des sites web, des applications web et des services web. Contrairement aux serveurs web traditionnels, tels que Apache, Nginx a été conçu pour gérer efficacement un grand nombre de connexions simultanées avec une utilisation optimale des ressources système. Il est souvent utilisé comme proxy inverse, équilibrage de charge et serveur de fichiers statiques.

Installation de Nginx sur Ubuntu

Pour utiliser Nginx comme votre serveur web principal et ne plus utiliser Apache, vous pouvez suivre les étapes ci-dessous :

1. Mettez à jour les packages du système :

BASH

```
sudo apt-get update
```

2. Installez Nginx en utilisant la commande suivante :

```
sudo apt-get install nginx
```

BASH

3. Une fois l'installation terminée, vous pouvez vérifier l'état du service Nginx en utilisant la commande suivante :

```
sudo service nginx status
```

BASH

Si le service est en cours d'exécution, vous devriez voir un message indiquant que Nginx est actif.

4. Arrêtez et désactivez le service Apache :

- Arrêtez le service Apache :

```
sudo systemctl stop apache2
```

BASH

- Désactivez le service Apache :

```
sudo systemctl disable apache2
```

BASH

5. Désinstallez Apache (facultatif) :

```
sudo apt-get remove apache2
```

Cela supprimera le package Apache de votre système.

Maintenant, Nginx est installé et configuré comme votre serveur web principal.

Configuration du pare-feu pour Nginx

Pour assurer la sécurité de votre serveur Nginx, vous pouvez configurer le pare-feu pour autoriser tout le trafic lié à Nginx en utilisant l'option "Full" dans `ufw`. Cela permettra à la fois le trafic HTTP (port 80) et le trafic HTTPS (port 443) vers Nginx.

Voici comment configurer le pare-feu avec l'option "Full" pour Nginx :

1. Vérifiez si `ufw` est installé :

```
sudo ufw status
```

BASH

Si `ufw` n'est pas installé, vous pouvez l'installer en utilisant la commande suivante :

```
sudo apt install ufw
```

BASH

2. Autorisez le trafic HTTP et HTTPS vers Nginx en utilisant les commandes suivantes :

```
sudo ufw allow 'Nginx Full'
```

BASH

Cette commande ajoute une règle au pare-feu pour permettre à la fois le trafic HTTP (port 80) et le trafic HTTPS (port 443) vers Nginx.

3. Vérifiez l'état du pare-feu pour vous assurer que la règle a été ajoutée avec succès :

```
sudo ufw status
```

BASH

Vous devriez voir une sortie indiquant que le trafic HTTP et HTTPS sont autorisés.

Maintenant, votre serveur Nginx est accessible à la fois via HTTP et HTTPS à partir de l'extérieur, tout en bénéficiant de la sécurité fournie par le pare-feu `ufw`.

N'oubliez pas de configurer d'autres règles de pare-feu si nécessaire pour répondre à vos besoins spécifiques en matière de sécurité.

Sources

<https://www.digitalocean.com/community/tutorials/how-to-install-linux-nginx-mysql-php-lemp-stack-on-ubuntu-22-04>

<https://www.nginx.com/>

1.4 Introduction

L'hébergement statique est une méthode d'hébergement de sites web où le serveur délivre le même contenu HTML à tous les utilisateurs. Contrairement à l'hébergement dynamique, où le contenu est généré à la volée, l'hébergement statique offre des performances supérieures et une meilleure sécurité. Dans ce chapitre, nous allons utiliser Nginx, un serveur web populaire, pour héberger un site statique simple sur un serveur Ubuntu.

Configuration de l'environnement

Avant de commencer, assurez-vous que Nginx est bien installé sur votre serveur. Vous pouvez vérifier cela en tapant la commande suivante dans votre terminal :

```
nginx -v
```

BASH

Si Nginx est installé, cette commande affichera la version de Nginx.

Ensuite, nous allons créer le dossier qui contiendra notre site web. Dans le terminal, tapez :

```
sudo mkdir -p /var/www/html/hello
```

BASH

Maintenant, nous allons créer un fichier HTML simple dans ce dossier. Utilisez votre éditeur de texte préféré pour créer le fichier `/var/www/html/hello/index.html` et ajoutez le code suivant :

```
<!DOCTYPE html>
<html>
<head>
  <title>Hello World</title>
  <style>
    body {
      background: linear-gradient(to right, red, orange, yellow, green, blue,
indigo, violet);
      color: white;
      font-size: 3em;
      font-weight: bold;
      display: flex;
      justify-content: center;
      align-items: center;
      height: 100vh;
      margin: 0;
      padding: 0;
      box-sizing: border-box;
    }
  </style>
</head>
<body>
  Hello World
</body>
</html>
```

XML

Configuration de Nginx

Les fichiers de configuration de Nginx se trouvent dans le dossier `/etc/nginx`. Pour configurer notre site, nous allons modifier le fichier de configuration par défaut. Ouvrez le fichier `/etc/nginx/sites-available/default` dans votre éditeur de texte et ajoutez le bloc de serveur suivant :

```
server {  
    listen 80 default_server;  
  
    location / {  
        root /var/www/html/hello;  
        index index.html;  
        try_files $uri $uri/ =404;  
    }  
}
```

La page web est désormais accessible en passant par l'adresse IP ou les domaines qui renvoient vers l'adresse IP du serveur.

Test et validation

Une fois la configuration terminée, redémarrez Nginx pour que les modifications prennent effet :

```
sudo service nginx restart
```

BASH

Vous devriez maintenant pouvoir accéder à votre site en visitant votre domaine ou l'adresse IP de votre serveur dans un navigateur web.



Hello World

Ajout de nouveaux sites

Si vous souhaitez héberger plusieurs sites statiques sur votre serveur, vous devrez créer une configuration distincte pour chaque site. Voici les étapes à suivre pour ajouter un nouveau site :

1. Création d'une nouvelle configuration : Dans le dossier `/etc/nginx/sites-available`, créez un nouveau fichier de configuration pour votre site. Par exemple, vous pouvez utiliser le nom de domaine de votre site comme nom de fichier. Par exemple, `/etc/nginx/sites-available/monsite.com`.
2. Configuration du fichier : Ouvrez le fichier de configuration que vous venez de créer dans votre éditeur de texte et ajoutez le bloc de serveur suivant :

```
server {  
    listen 80;  
    server_name monsite.com www.monsite.com;  
  
    location / {  
        root /var/www/html/monsite;  
        index index.html;  
        try_files uriuri uriuri/ =404;  
    }  
}
```

Assurez-vous de remplacer `monsite.com` et `www.monsite.com` par les noms de domaine de votre site. De plus, veillez à spécifier le chemin correct vers le dossier où se trouvent les fichiers de votre site.

1. Activation de la configuration : Créez un lien symbolique vers le fichier de configuration que vous venez de créer dans le dossier `/etc/nginx/sites-enabled` . Utilisez la commande suivante :

```
sudo ln -s /etc/nginx/sites-available/monsite.conf /etc/nginx/sites-enabled/
```

BASH

1. Vérification de la configuration : Utilisez la commande suivante pour vérifier que la configuration est correctement écrite et qu'il n'y a pas d'erreurs :

```
sudo nginx -t
```

BASH

Assurez-vous que la commande affiche `syntax is ok` et `test is successful` .

1. Redémarrage de Nginx : Redémarrez Nginx pour prendre en compte la nouvelle configuration :

```
sudo service nginx restart
```

BASH

Maintenant, vous devriez pouvoir accéder à votre nouveau site en utilisant le nom de domaine que vous avez spécifié dans la configuration.

N'oubliez pas de répéter ces étapes pour chaque nouveau site que vous souhaitez héberger sur votre serveur.

Conclusion

En suivant ces étapes, vous pouvez facilement ajouter de nouveaux sites statiques à votre serveur en utilisant Nginx. Assurez-vous de créer une nouvelle configuration pour chaque site, de spécifier le nom de domaine approprié, le dossier racine des fichiers du site, d'activer la configuration et de redémarrer Nginx pour appliquer les modifications.

N'hésitez pas à consulter la documentation officielle de Nginx ou à rechercher des tutoriels supplémentaires pour obtenir des informations plus détaillées sur la configuration de Nginx pour l'hébergement de sites statiques.

Conclusion

L'hébergement statique avec Nginx est une méthode simple et efficace pour déployer des sites web. Dans ce chapitre, nous avons vu comment configurer Nginx pour servir un site statique sur un serveur Ubuntu. En suivant ces étapes, vous pouvez héberger n'importe quel site statique sur votre propre serveur.

Références

- [Documentation officielle de Nginx](#)
- [Guide de l'hébergement statique](#)

1.5 MariaDB

Introduction à MariaDB

MariaDB est un système de gestion de base de données relationnelle open source qui est une alternative à MySQL. Il a été développé par les créateurs originaux de MySQL et est conçu pour être une solution fiable, performante et évolutive pour la gestion des données.

Différences entre MariaDB et MySQL

MariaDB partage de nombreuses similitudes avec MySQL, mais il existe également des différences significatives entre les deux systèmes. Voici quelques-unes des différences clés :

1. **Performances améliorées** : MariaDB est réputé pour ses performances supérieures par rapport à MySQL. Il utilise un moteur de stockage optimisé et des algorithmes améliorés pour offrir des requêtes plus rapides et une meilleure utilisation des ressources.
2. **Connexions simultanées accrues** : MariaDB permet un plus grand nombre de connexions simultanées par rapport à MySQL, ce qui peut être particulièrement avantageux pour les applications à forte charge.
3. **Fonctionnalités supplémentaires** : MariaDB propose des fonctionnalités supplémentaires qui ne sont pas disponibles dans MySQL, telles que des moteurs de stockage supplémentaires, des extensions de sécurité et des améliorations de performances.
4. **Indépendance vis-à-vis d'Oracle** : MariaDB est développé et maintenu indépendamment d'Oracle, la société qui possède MySQL. Cela signifie que MariaDB est moins soumis aux décisions d'Oracle et offre une plus grande liberté de développement.

Avantages de MariaDB par rapport à MySQL

Voici quelques-uns des avantages de l'utilisation de MariaDB par rapport à MySQL :

- Fonctionnalités supplémentaires : MariaDB propose des fonctionnalités supplémentaires, telles que des moteurs de stockage supplémentaires et des extensions de sécurité, qui offrent une plus grande flexibilité et des options de personnalisation.
- Communauté active : MariaDB bénéficie d'une communauté active de développeurs et d'utilisateurs qui contribuent à son développement et offrent un support technique.
- Migration facile : Étant donné que MariaDB est une alternative compatible avec MySQL, il est relativement facile de migrer une application existante de MySQL vers MariaDB sans nécessiter de modifications majeures.

Installation de MariaDB

Pour installer MariaDB sur votre système, suivez les étapes ci-dessous :

1. Assurez-vous d'avoir les privilèges d'administration sur votre système.
2. Ouvrez un terminal et exécutez la commande suivante pour mettre à jour les packages de votre système :

```
sudo apt update
```

BASH

3. Une fois la mise à jour terminée, exécutez la commande suivante pour installer MariaDB :

```
sudo apt install mariadb-server
```

BASH

4. Une fois l'installation terminée, vous pouvez vérifier que MariaDB est en cours d'exécution en utilisant la commande suivante :

```
sudo systemctl status mariadb
```

BASH

Vous devriez voir un message indiquant que le service MariaDB est actif et en cours d'exécution.

Configuration de MariaDB

Après l'installation de MariaDB, vous devez exécuter le script de sécurité inclus pour configurer certains paramètres de sécurité par défaut moins sécurisés. Suivez les étapes ci-dessous :

1. Exécutez la commande suivante pour lancer le script de sécurité :

```
sudo mysql_secure_installation
```

BASH

2. Le script vous guidera à travers une série de prompts où vous pourrez apporter des modifications aux options de sécurité de votre installation MariaDB.
3. Lorsque vous êtes invité à entrer le mot de passe root actuel pour la base de données, appuyez sur ENTER pour indiquer "none", car vous n'avez pas encore défini de mot de passe.
4. À la prochaine invite, vous serez invité à définir un mot de passe root pour la base de données. Sur Ubuntu, le compte root de MariaDB est étroitement lié à la maintenance automatisée du système, il est donc recommandé de ne pas modifier les méthodes d'authentification configurées pour ce compte. Tapez N et appuyez sur ENTER.
5. Vous pouvez maintenant appuyer sur Y puis sur ENTER pour accepter les valeurs par défaut pour les questions suivantes. Cela supprimera certains utilisateurs anonymes et la base de données de test, désactivera les connexions root à distance et chargera les nouvelles règles pour que MariaDB mette immédiatement en œuvre les modifications que vous avez apportées.

Avec cela, vous avez terminé la configuration initiale de MariaDB. Dans la section suivante, nous vous guiderons dans la création d'utilisateurs dans MariaDB.

Création d'utilisateurs dans MariaDB

Dans cette section, nous allons vous guider à travers le processus de création d'un utilisateur administrateur dans MariaDB.

1. Ouvrez le terminal de votre système et exécutez la commande suivante pour accéder à l'invite de commandes MariaDB :

```
sudo mariadb
```

BASH

2. Une fois dans l'invite de commandes MariaDB, exécutez la commande suivante pour créer un nouvel utilisateur avec les privilèges root et l'authentification basée sur un mot de passe. Assurez-vous de remplacer "admin" par le nom d'utilisateur de votre choix et "password" par un mot de passe sécurisé :

```
GRANT ALL ON *.* TO 'admin'@'localhost' IDENTIFIED BY 'password' WITH GRANT OPTION;
```

SQL

Cette commande accorde tous les privilèges sur toutes les bases de données et toutes les tables à l'utilisateur "admin" se connectant depuis l'ordinateur local ("localhost").

3. Après avoir exécuté la commande, exécutez la commande suivante pour actualiser les privilèges et les rendre disponibles dans la session en cours :

```
FLUSH PRIVILEGES;
```

4. Pour quitter l'invite de commandes MariaDB, exécutez la commande suivante :

```
exit
```

BASH

Avec cela, vous avez créé un nouvel utilisateur administrateur dans MariaDB avec des privilèges équivalents à ceux du compte root, mais configuré pour une authentification basée sur un mot de passe.

Vous pouvez maintenant vous connecter à votre base de données avec un logiciel tel que MySQL Workbench en utilisant la connexion par tunnel SSH. Vous pourrez ainsi créer de nouvelles bases de données et des utilisateurs pour celles-ci.

Sources

<https://www.digitalocean.com/community/tutorials/how-to-install-mariadb-on-ubuntu-20-04>

1.6 Hébergement PHP

Introduction à PHP 8.1

PHP 8.1 est une version majeure du langage de programmation PHP. Il apporte de nombreuses améliorations et nouvelles fonctionnalités qui facilitent le développement web. En utilisant Ubuntu 23, nous sommes malheureusement limités à cette version de PHP lors de l'écriture de ces lignes.

Vérification des prérequis

Avant de commencer l'installation de PHP 8.1, il est important de s'assurer que votre système est à jour. Vous pouvez le faire en exécutant la commande suivante :

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

BASH

Cette commande met à jour la liste des paquets disponibles et met à niveau les paquets installés sur votre système.

Installation

Une fois votre système à jour, vous pouvez procéder à l'installation de PHP 8.1 en utilisant la commande suivante :

```
sudo apt install php8.1
```

BASH

Cette commande installe PHP 8.1 sur votre système.

Installation des extensions PHP

PHP est souvent utilisé avec un certain nombre d'extensions qui fournissent des fonctionnalités supplémentaires. Vous pouvez installer ces extensions en utilisant la commande suivante :

```
sudo apt install php8.1-  
{bcmath,fpm,xml,mysql,zip,intl,gd,cli,bz2,curl,mbstring,opcache}
```

BASH

Cette commande installe les extensions suivantes :

- `bcmath` : pour les calculs mathématiques de précision arbitraire
- `fpm` : pour l'interface FastCGI Process Manager
- `xml` : pour le traitement des documents XML
- `mysql` : pour la connexion aux bases de données MySQL
- `zip` : pour la manipulation des fichiers zip
- `intl` : pour les fonctionnalités d'internationalisation
- `gd` : pour la manipulation des images
- `cli` : pour l'interface en ligne de commande de PHP
- `bz2` : pour la compression et la décompression des fichiers bzip2
- `curl` : pour les requêtes HTTP
- `mbstring` : pour le traitement des chaînes de caractères multibyte
- `opcache` : pour l'optimisation et la mise en cache des scripts PHP

D'accord, je vais simplifier les titres et enlever les numéros de sous-section.

Installation de Composer

Composer est un outil de gestion de dépendances pour PHP. Il permet de déclarer les bibliothèques dont votre projet a besoin et il les gère pour vous.

Pour installer Composer, suivez les instructions ci-dessous :

1. **Téléchargement de l'installateur** : Exécutez la commande suivante pour télécharger l'installateur de Composer dans le répertoire courant.

```
php -r "copy('https://getcomposer.org/installer', 'composer-setup.php');" BASH
```

1. **Vérification de l'installateur** : Pour vous assurer que l'installateur n'a pas été corrompu pendant le téléchargement, vérifiez le SHA-384 en exécutant cette commande :

```
php -r "if (hash_file('sha384', 'composer-setup.php') ===  
'e21205b207c3ff031906575712edab6f13eb0b361f2085f1f1237b7126d785e826a450292b6cfd1d64d92  
e6563bbde02') { echo 'Installer verified'; } else { echo 'Installer corrupt';  
unlink('composer-setup.php'); } echo PHP_EOL;" BASH
```

1. **Exécution de l'installateur** : Cette commande vérifie certains paramètres de php.ini, vous avertit s'ils sont incorrectement définis, et télécharge le dernier composer.phar dans le répertoire courant.

```
php composer-setup.php BASH
```

1. **Suppression de l'installateur** : Une fois l'installation terminée, vous n'avez plus besoin de l'installateur. Supprimez-le avec cette commande :

```
php -r "unlink('composer-setup.php');" BASH
```

1. **Déplacement de composer.phar** : Pour pouvoir appeler simplement `composer` à partir de n'importe quel répertoire, déplacez composer.phar dans un répertoire de votre PATH avec cette commande :

```
sudo mv composer.phar /usr/local/bin/composer BASH
```

Maintenant, vous avez installé Composer sur votre système. Vous pouvez vérifier que l'installation a réussi en exécutant `composer` dans votre terminal. Vous devriez voir une sortie indiquant la version de Composer que vous avez installé.

Nous allons aussi installer 7zip pour accélérer l'installation de packages avec composer:

```
sudo apt install p7zip-full BASH
```

Création et configuration d'un projet Laravel

Laravel est un framework web PHP qui facilite le développement d'applications modernes. Il offre une syntaxe expressive et élégante qui vise à rendre le développement web agréable et créatif.

Création d'un répertoire pour le projet

D'abord, nous allons ajouter nos utilisateurs au groupe `www-data` avec la commande suivante:

```
sudo usermod -aG www-data admin
sudo usermod -aG www-data deployer
```

BASH

Nous devons nous déconnecter et reconnecter pour que ces changements prennent effet.

Pour commencer, créez un répertoire pour votre projet. Nous allons utiliser `/var/www/php` comme exemple :

```
sudo mkdir -p /var/www/php
sudo chmod
```

BASH

Création d'un nouveau projet Laravel

Naviguez dans le répertoire que vous venez de créer et lancez la commande suivante pour créer un nouveau projet Laravel :

```
cd /var/www/php

sudo chown $USER:www-data /var/www/php

composer create-project laravel/laravel laravel.domain.com
```

BASH

Configuration du projet

Après avoir créé le projet, vous devez configurer les permissions de fichiers et Nginx.

Configuration des permissions de fichiers

Une fois que vous avez créé le projet, il est important de configurer correctement les permissions de fichiers. Laravel nécessite certaines permissions spécifiques pour fonctionner correctement.

Voici comment vous pouvez configurer les permissions de fichiers pour votre projet Laravel :

1. **Modification du propriétaire des fichiers** : La première commande change le propriétaire des fichiers du projet. `$USER` est une variable d'environnement qui représente l'utilisateur actuel, et `www-data` est le groupe généralement utilisé par le serveur web.

```
sudo chown -R $USER:www-data /var/www/php/laravel.domain.com
```

BASH

1. **Modification des permissions des fichiers** : La deuxième commande change les permissions des fichiers du projet. `755` signifie que le propriétaire peut lire, écrire et exécuter les fichiers, tandis que les autres utilisateurs peuvent seulement lire et exécuter les fichiers.

```
sudo chmod -R 755 /var/www/php/.domain.com
```

BASH

1. **Modification des permissions des répertoires** `storage` et `bootstrap/cache` : Laravel a besoin d'écrire dans ces répertoires, donc vous devez leur donner les permissions appropriées. `ug+rwX` signifie que le propriétaire et le groupe peuvent lire, écrire et exécuter les fichiers.

```
cd /var/www/php/.domain.com
sudo chmod -R ug+rwX storage bootstrap/cache
```

BASH

1. **Sécurisation du fichier** `.env` : Le fichier `.env` contient des informations sensibles comme les identifiants de la base de données. Vous devez donc le sécuriser. `740` signifie que le propriétaire peut lire et écrire le fichier, mais ne peut pas l'exécuter, tandis que les autres utilisateurs ne peuvent pas lire, écrire ni exécuter le fichier.

```
sudo chmod 740 .env
```

BASH

En configurant correctement les permissions de fichiers, vous vous assurez que Laravel fonctionne correctement et que vos informations sensibles sont sécurisées. Vous pourrez mettre à jour le projet ou utiliser les commandes artisan puisque vous êtes le propriétaire et le serveur aura accès aux fichiers.

Pour la configuration de Nginx, vous devez créer un nouveau fichier de configuration pour votre site. Vous pouvez le faire en éditant un nouveau fichier dans `/etc/nginx/sites-available/` et en ajoutant la configuration appropriée pour votre site. Par exemple `laravel.domain.com` dont voici le contenu:

```

server {
    listen 80;
    listen [::]:80;
    server_name laravel.domain.com;
    root /var/www/php/laravel.domain.com/public;

    add_header X-Frame-Options "SAMEORIGIN";
    add_header X-Content-Type-Options "nosniff";

    index index.php;

    charset utf-8;

    location / {
        try_files uriuri uriuri/ /index.php?query_string;
    }

    location = /favicon.ico { access_log off; log_not_found off; }
    location = /robots.txt  { access_log off; log_not_found off; }

    error_page 404 /index.php;

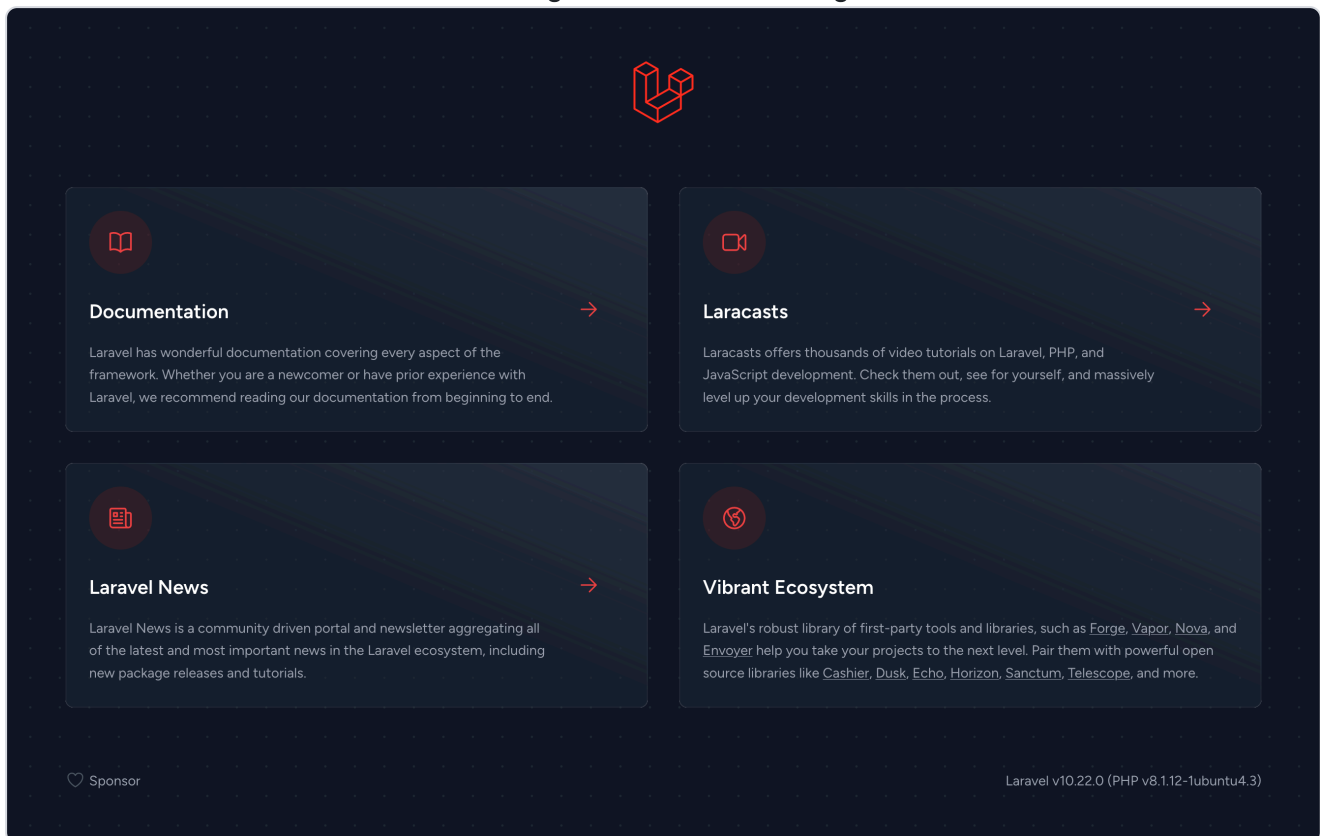
    location ~ /\.php {
        fastcgi_pass unix:/var/run/php/php8.1-fpm.sock;
        fastcgi_param SCRIPT_FILENAME realpathrootrealpath_rootrealpathr
ootfastcgi_script_name;
        include fastcgi_params;
    }

    location ~ /\.(!well-known).* {
        deny all;
    }
}

```

N'oubliez pas de créer le lien symbolique vers le dossier sites-enabled et de relancer nginx si la configuration est correcte comme vu précédemment.

Si tout a fonctionné, vous devriez voir apparaître la page d'accueil du projet Laravel si vous entrez l'URL configurée dans votre navigateur.



Installation de Node.js et npm

Node.js est une plateforme construite sur le moteur JavaScript V8 de Chrome qui permet d'exécuter du code JavaScript côté serveur. npm est le gestionnaire de paquets par défaut pour Node.js. Nous en aurons besoin pour construire les assets du frontend de notre application.

Pour installer Node.js et npm, suivez les instructions ci-dessous :

1. **Mise à jour des paquets** : Commencez par mettre à jour la liste des paquets disponibles pour votre système.

```
sudo apt-get update
```

BASH

1. **Installation des prérequis** : Installez les paquets nécessaires pour télécharger et importer la clé GPG de Nodesource.

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

BASH

1. **Création du répertoire pour la clé GPG** : Créez un répertoire pour stocker la clé GPG de Nodesource.

BASH

```
sudo mkdir -p /etc/apt/keyrings
```

1. **Téléchargement et importation de la clé GPG** : Téléchargez la clé GPG de Nodesource et importez-la.

BASH

```
curl -fsSL https://deb.nodesource.com/gpgkey/nodesource-repo.gpg.key | sudo gpg --  
dearmor -o /etc/apt/keyrings/nodesource.gpg
```

1. **Création du dépôt deb** : Créez le dépôt deb pour Node.js. Vous pouvez spécifier la version majeure de Node.js que vous souhaitez installer en changeant la valeur de `NODE_MAJOR`.

BASH

```
NODE_MAJOR=20  
echo "deb [signed-by=/etc/apt/keyrings/nodesource.gpg]  
https://deb.nodesource.com/node_${NODE_MAJOR}.x nodistro main" | sudo tee  
/etc/apt/sources.list.d/nodesource.list
```

Notez que vous pouvez changer `NODE_MAJOR` en fonction de la version de Node.js que vous souhaitez installer. Par exemple, vous pouvez choisir parmi les versions 16, 18 ou 20.

1. **Mise à jour des paquets et installation de Node.js** : Mettez à jour à nouveau la liste des paquets disponibles pour votre système, puis installez Node.js.

BASH

```
sudo apt-get update  
sudo apt-get install nodejs -y
```

Félicitations, vous avez installé Node.js et npm sur votre système. Vous pouvez vérifier que l'installation a réussi en exécutant `node -v` et `npm -v` dans votre terminal. Ces commandes afficheront les versions de Node.js et npm que vous avez installées.

Ajout de l'authentification

Laravel Breeze est un package minimaliste qui vous permet d'implémenter facilement l'authentification dans votre application Laravel. Vous pouvez l'installer en utilisant Composer :

BASH

```
sudo composer require laravel/breeze --dev  
sudo php artisan breeze:install
```

Choisissez les options suivantes:

```
Which Breeze stack would you like to install? _____
```

```
Vue with Inertia
```

```
Would you like any optional features? _____
```

- Dark mode
- Inertia SSR
- > ■ TypeScript (experimental)

```
Use the space bar to select options.
```

Le script installera les dépendances PHP et JavaScript du projet et lancera le script de build des assets frontend.

Connexion à une base de données

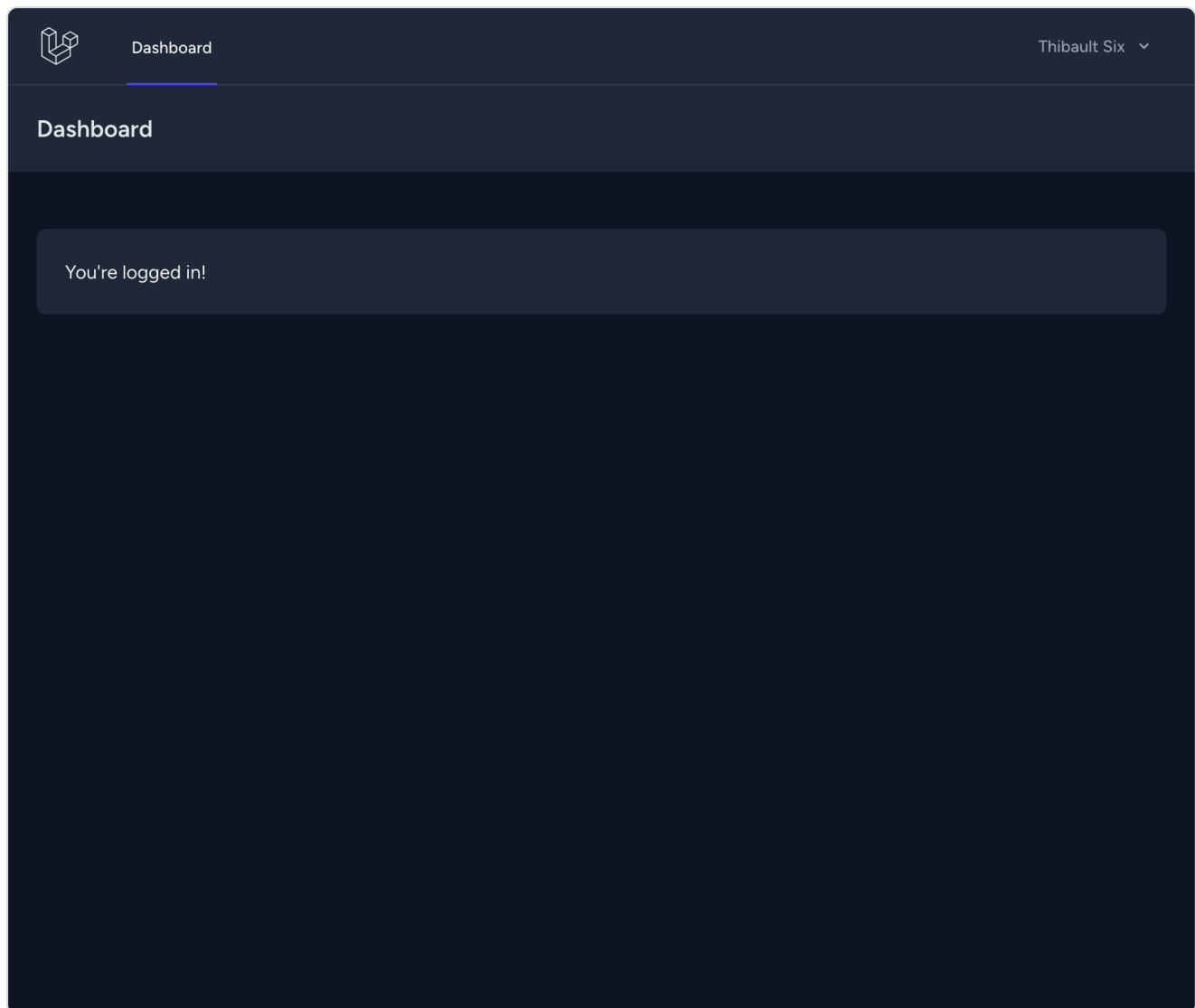
Créez d'abord une nouvelle base de données `laravel` et un utilisateur qui lui sera dédié.

Pour connecter votre application Laravel à une base de données, vous devez modifier le fichier `.env` de votre projet. Vous pouvez y spécifier le type de base de données, l'hôte, le nom de la base de données, l'utilisateur et le mot de passe.

Vous devrez ensuite lancer la commande de migration de la base de données:

```
php artisan migrate
```

Vous pouvez maintenant aller sur l'application et utiliser le formulaire dédié à la création de compte. Si tout fonctionne, vous devriez pouvoir accéder au Dashboard une fois le compte créé :



Optimisation du projet

Pour terminer l'installation, vous pouvez optimiser votre projet Laravel en exécutant les commandes suivantes :

```
composer install --optimize-autoloader --no-dev  
php artisan optimize
```

BASH

Ces commandes réduisent le temps de chargement de votre application en optimisant la mise en cache et l'autoloader de Composer.

Et voilà ! Vous avez créé et configuré un nouveau projet Laravel. Vous pouvez maintenant commencer à développer votre application.

Sources

<https://laravel.com/docs/10.x/deployment>

<https://www.digitalocean.com/community/tutorials/how-to-install-and-configure-laravel-with-nginx-on-ubuntu-22-04>

<https://www.digitalocean.com/community/tutorials/how-to-install-linux-nginx-mysql-php-lemp-stack-on-ubuntu-22-04>

https://computingforgeeks.com/how-to-install-php-8-2-on-ubuntu/?expand_article=1

<https://getcomposer.org/download/>

<https://github.com/oerdnj/deb.sury.org/wiki/Managing-Multiple-Versions>

<https://linux.how2shout.com/how-to-install-7-zip-on-ubuntu-22-04-Its-jammy-linux/>

<https://github.com/nodesource/distributions>

https://www.reddit.com/r/laravel/comments/ymmvif/laravel_project_who_should_own_root_folder/

1.7 CertBot

Introduction à Certbot

Définition et utilité de Certbot

Certbot est un outil logiciel gratuit et open-source qui permet d'obtenir automatiquement des certificats SSL/TLS, fournis par Let's Encrypt, pour sécuriser les communications sur un site web. Il facilite également le processus de renouvellement de ces certificats.

Brève histoire et l'organisation Let's Encrypt

Let's Encrypt est une autorité de certification lancée en 2016 qui fournit des certificats gratuits dans le but de promouvoir un internet plus sécurisé. Certbot a été développé par l'EFF (Electronic Frontier Foundation) pour automatiser l'utilisation de ces certificats.

Installation de Certbot

Prérequis système

- Accès root ou sudo sur le serveur
- Serveur web déjà installé (Apache, Nginx, etc.)

- Noms de domaine pointant vers le serveur

Instructions d'installation

Debian/Ubuntu

<https://certbot.eff.org/instructions?ws=nginx&os=ubuntufocal>

1. Mettre à jour la liste des paquets :

```
sudo apt update
```

BASH

2. Installer Certbot :

```
sudo snap install --classic certbot
```

BASH

3. Préparation de la commande certbot :

```
sudo ln -s /snap/bin/certbot /usr/bin/certbot
```

BASH

Configuration de Certbot

Configuration initiale pour un site avec Nginx

Pour obtenir un certificat SSL/TLS et configurer automatiquement Nginx, exécutez la commande suivante :

```
sudo certbot --nginx
```

BASH

- Certbot va rechercher les noms de domaine configurés dans Nginx et vous demandera de choisir ceux pour lesquels vous souhaitez obtenir un certificat.
- Vous devrez également fournir une adresse email pour les notifications de renouvellement et accepter les termes du service.
- Certbot modifiera ensuite la configuration de Nginx pour activer HTTPS.

Configuration pour plusieurs sites

Si vous avez plusieurs sites hébergés sur le même serveur Nginx, vous pouvez répéter le processus pour chaque site en exécutant à nouveau la commande `sudo certbot --nginx` et en sélectionnant les noms de domaine appropriés lorsqu'ils vous sont présentés.

Exemple

1. Lancez la commande `sudo certbot --nginx`
2. Entrez votre email.

```
@vps: /var/www/php/ $ sudo certbot --nginx
Saving debug log to /var/log/letsencrypt/letsencrypt.log
Enter email address (used for urgent renewal and security notices)
(Enter 'c' to cancel): gmail.com
```

3. Acceptez les conditions d'utilisation.

```
-----
Please read the Terms of Service at
https://letsencrypt.org/documents/LE-SA-v1.3-September-21-2022.pdf. You must
agree in order to register with the ACME server. Do you agree?
-----
(Y)es/(N)o: Y
```

4. Refusez le partage de votre email.

```
-----
Would you be willing, once your first certificate is successfully issued, to
share your email address with the Electronic Frontier Foundation, a founding
partner of the Let's Encrypt project and the non-profit organization that
develops Certbot? We'd like to send you email about our work encrypting the web,
EFF news, campaigns, and ways to support digital freedom.
-----
(Y)es/(N)o: N
Account registered.
```

5. Sélectionnez les sites sur lesquels vous voulez activer HTTPS.

```
Which names would you like to activate HTTPS for?
We recommend selecting either all domains, or all domains in a VirtualHost/server block.
-----
1:
2:
3:
-----
Select the appropriate numbers separated by commas and/or spaces, or leave input
blank to select all options shown (Enter 'c' to cancel): 1 2 3
Requesting a certificate for and 2 more domains

Successfully received certificate.
Certificate is saved at: /etc/letsencrypt/live/ /fullchain.pem
Key is saved at: /etc/letsencrypt/live/ /privkey.pem
This certificate expires on 2024-05-04.
These files will be updated when the certificate renews.
Certbot has set up a scheduled task to automatically renew this certificate in the background.

Deploying certificate
Successfully deployed certificate for to /etc/nginx/sites-enabled,
Successfully deployed certificate for to /etc/nginx/sites-enabled,
Successfully deployed certificate for to /etc/nginx/sites-enabled,
Congratulations! You have successfully enabled HTTPS on , and

-----
If you like Certbot, please consider supporting our work by:
* Donating to ISRG / Let's Encrypt: https://letsencrypt.org/donate
* Donating to EFF: https://eff.org/donate-le
-----
```

Gestion des certificats SSL/TLS

Obtention d'un certificat

- Utilisez la commande suivante pour obtenir un certificat pour un ou plusieurs domaines :

```
sudo certbot --nginx -d exemple.com -d www.exemple.com
```

BASH

- Certbot va valider la propriété du domaine et émettre un certificat.

Renouvellement automatique des certificats

- Certbot crée une tâche cron ou systemd timer pour renouveler les certificats automatiquement avant qu'ils n'expirent.
- Pour tester le processus de renouvellement, utilisez :

```
sudo certbot renew --dry-run
```

BASH

Visualisation et gestion des certificats existants

- Pour lister tous les certificats Certbot installés sur le serveur :

```
sudo certbot certificates
```

BASH

- Pour supprimer un certificat :

```
sudo certbot delete --cert-name exemple.com
```

BASH

Commandes et options utiles de Certbot

Commandes de base

- Obtenir un certificat :**

```
sudo certbot --nginx
```

BASH

- Renouveler tous les certificats :**

```
sudo certbot renew
```

BASH

- **Annuler un certificat :**

```
sudo certbot revoke --cert-path /chemin/vers/cert.pem
```

BASH

Options avancées et personnalisations

- **Changer l'email associé aux certificats :**

```
sudo certbot update_account --email nouvel-email@example.com
```

BASH

- **Modifier la configuration HTTPS :**

```
sudo certbot --nginx --preferred-challenges http -d exemple.com
```

BASH

- **Forcer le renouvellement d'un certificat :**

```
sudo certbot renew --force-renewal
```

BASH

Bonnes pratiques et sécurité

Sécurisation de la configuration SSL/TLS

- Utilisez des paramètres de chiffrement forts dans la configuration de votre serveur web.
- Mettez régulièrement à jour vos logiciels serveurs pour corriger les vulnérabilités.
- Utilisez l'option `--hsts` pour ajouter la Strict-Transport-Security header et forcer le navigateur à toujours utiliser HTTPS.

Automatisation et planification du renouvellement

- Vérifiez que la tâche cron ou systemd timer est bien en place pour automatiser le renouvellement des certificats.
- Surveillez les logs de renouvellement pour vous assurer qu'il n'y a pas d'erreurs.

Surveillance et dépannage

- Testez votre configuration SSL avec des outils en ligne comme [SSL Labs' SSL Test](#).
- Consultez les logs de Certbot en cas d'échec de renouvellement ou d'autres erreurs :

```
sudo less /var/log/letsencrypt/letsencrypt.log
```

BASH

- En cas de problème, utilisez l'option `--debug` pour obtenir plus d'informations lors de l'exécution des commandes Certbot.

Sources

- <https://certbot.eff.org>

1.8 Deployer

Dans ce chapitre, nous allons aborder un aspect crucial du développement d'applications web : le déploiement. Le déploiement est l'étape où nous prenons notre application développée localement et la mettons en ligne pour que tout le monde puisse y accéder.

Nous allons créer un nouveau projet Laravel, un framework PHP populaire, et y intégrer l'authentification grâce au package Laravel Breeze. Ensuite, nous sauvegarderons notre projet sur GitHub, un service d'hébergement pour les projets versionnés avec Git.

Ensuite, nous allons installer et configurer Deployer, un outil de déploiement pour PHP. Nous verrons comment créer une configuration de déploiement et comment déployer notre application avec un utilisateur spécifique.

Enfin, nous allons mettre en place un système de CI/CD avec GitHub Actions. Les systèmes de CI/CD sont essentiels pour la livraison continue de logiciels de qualité. Ils nous permettent d'automatiser des tâches telles que les tests et le déploiement, ce qui accélère le processus de développement et assure que notre application est toujours dans un état déployable.

Création d'un nouveau projet Laravel

Installation de Laravel

Avant de commencer, assurez-vous d'avoir installé Composer, l'outil de gestion des dépendances de PHP, sur votre machine.

Création d'un nouveau projet Laravel

Pour créer un nouveau projet Laravel, utilisez la commande suivante :

```
composer create-project laravel/laravel nom_du_projet
```

BASH

Remplacez "nom_du_projet" par le nom que vous souhaitez donner à votre projet. Cette commande crée un nouveau répertoire avec le nom du projet et installe une nouvelle instance de Laravel dans ce répertoire.

Installation du package Laravel Breeze

Introduction à Laravel Breeze

Laravel Breeze est un kit de démarrage minimal pour la construction d'une application Laravel avec authentification. Il fournit une bonne base pour commencer à construire votre application.

Installation de Laravel Breeze

Pour installer Laravel Breeze, utilisez la commande suivante dans le répertoire de votre projet :

```
composer require laravel/breeze --dev
```

BASH

Ensuite, installez Breeze en utilisant la commande :

```
php artisan breeze:install
```

BASH

Cette commande installe les vues d'authentification et le contrôleur d'authentification de Breeze.

Créez une base de données et renseignez les informations de connexion dans le fichier `.env`.

Exécutez la commande suivante pour créer les tables :

```
php artisan migrate
```

Sauvegarde du projet sur GitHub

Initialisation d'un dépôt Git

Pour initialiser un dépôt Git dans votre projet, utilisez la commande suivante :

```
git init
```

BASH

Cette commande crée un nouveau dépôt Git dans votre répertoire de projet.

Ajout du projet au dépôt Git

Pour ajouter votre projet au dépôt Git, utilisez les commandes suivantes :

```
git add .  
git commit -m "Initial commit"
```

BASH

Ces commandes ajoutent tous les fichiers de votre projet au dépôt Git et créent un premier commit avec le message "Initial commit".

Création d'un dépôt GitHub

Allez sur GitHub et créez un nouveau dépôt. Ne cochez pas l'option "Initialize this repository with a README", car vous avez déjà un projet existant que vous souhaitez pousser sur GitHub.

Push du projet sur GitHub

Pour pousser votre projet sur GitHub, utilisez les commandes suivantes :

```
git remote add origin url_de_votre_dépôt_github  
git push -u origin master
```

BASH

Remplacez "url_de_votre_dépôt_github" par l'URL de votre dépôt GitHub. Ces commandes poussent votre projet sur GitHub.

Installation et configuration du package Deployer

Présentation de Deployer

Deployer est un outil de déploiement qui permet d'automatiser le déploiement de vos applications PHP sur des serveurs distants.

Installation de Deployer

Pour installer Deployer, utilisez la commande suivante dans votre terminal :

```
composer require deployer/deployer --dev
```

BASH

Création de la configuration de déploiement

Après avoir installé Deployer, créez un fichier "deploy.php" à la racine de votre projet. Ce fichier contiendra les tâches de déploiement spécifiques à votre application.

Déploiement automatique avec l'utilisateur Deployer

Configuration de l'utilisateur Deployer

Dans le fichier "deploy.php", configurez l'utilisateur Deployer en définissant le serveur de déploiement et le chemin de déploiement.

Déploiement du projet avec Deployer

Pour déployer votre projet avec Deployer, utilisez la commande suivante :

```
dep deploy production
```

BASH

Mise en place d'un système de CI/CD avec GitHub Actions

Introduction à GitHub Actions

GitHub Actions est un outil de CI/CD qui permet d'automatiser votre workflow de développement. Avec GitHub Actions, vous pouvez créer des workflows personnalisés pour construire, tester, emballer, publier et déployer votre code.

Création d'un workflow de CI/CD

Pour créer un workflow de CI/CD, créez un nouveau fichier dans le répertoire ".github/workflows" de votre projet. Ce fichier définira les étapes de votre workflow de CI/CD.

Configuration du workflow pour le déploiement automatique

Dans le fichier de workflow, configurez les étapes de build, de test et de déploiement pour déployer automatiquement votre application à chaque push sur la branche master.