Configuration d'un Serveur Web en ligne sur Debian

Ce document détaille les étapes complètes pour transformer une machine locale sous Debian en un serveur web sécurisé et accessible depuis Internet.

SOMMAIRE :

Installation du serveur web Apache

Apache est le logiciel qui va traiter les requêtes HTTP et délivrer les pages web aux visiteurs.

Mise à jour du système : Avant toute installation, il est crucial de mettre à jour les paquets du système.
```
sudo apt update && sudo apt upgrade -y
```

Installation d'Apache :
```
sudo apt install apache2 -y
```

Vérification du service : Pour s'assurer qu'Apache est bien démarré :
```
sudo systemctl status apache2
```

À ce stade, en tapant l'adresse IP locale de la machine dans un navigateur, la page par défaut d'Apache s'affiche.

Installation de MySQL, PHP et phpMyAdmin

Pour héberger des sites web dynamiques, nous avons besoin d'une base de données et de PHP.

Installation de MariaDB (fork de MySQL) et PHP :
```
sudo apt install mariadb-server php libapache2-mod-php php-mysql -y
```

Sécurisation de la base de données : Il est impératif de sécuriser l'installation par défaut de la base de données :
```
sudo mysql_secure_installation
```

Ce script interactif permet de définir un mot de passe root fort, de supprimer les utilisateurs anonymes, de désactiver la connexion root à distance et de supprimer la base de données de test.

Installation de phpMyAdmin : phpMyAdmin offre une interface graphique pour gérer les bases de données.
```
sudo apt install phpmyadmin -y
```

Lors de l'installation, sélectionnez `apache2` avec la barre d'espace, et acceptez la configuration de la base de données avec `dbconfig-common`.

Installation et configuration d'OpenSSH

OpenSSH permet de prendre le contrôle du serveur à distance de manière sécurisée.

Installation du serveur SSH :
```
sudo apt install openssh-server -y
```

Modification du port par défaut (Sécurité) : Le port SSH par défaut (22) est la cible de nombreuses attaques automatisées.
```
sudo nano /etc/ssh/sshd_config
```

Cherchez la ligne `#Port 22`, décommentez-la.

Redémarrage du service SSH :
```
sudo systemctl restart ssh
```

Désormais, pour se **connecter**, il faudra spécifier le port :

```
ssh utilisateur@ip_du_serveur -p 2222.
```

Ouverture des ports sur la Box Internet (Redirection NAT/PAT)

Pour que le serveur soit accessible depuis l'extérieur (Internet), il faut configurer le routeur (Box Internet) pour rediriger le trafic entrant vers la machine Debian.

Récupération de l'IP locale du serveur :
```
ip a
```

Exemple : `192.168.1.50`

Configuration de la Box : Connectez-vous à l'interface d'administration de votre Box (souvent `192.168.1.1` ou `192.168.1.254`). Allez dans la section **Réseau > NAT/PAT** ou **Transfert de ports** et créez les règles suivantes :

Règle	Adresse IP	Port Interne	Port Externe	Protocole
HTTP	192.168.1.50	80	80	TCP
HTTPS	192.168.1.50	443	443	TCP
SSH	192.168.1.50	22	2222	TCP

Configuration d'un nom de domaine avec No-IP

Les connexions internet résidentielles ont souvent une IP publique dynamique (qui change régulièrement). No-IP permet d'associer un nom de domaine fixe à cette IP changeante.

1. **Création du domaine** : Créez un compte sur le site de No-IP et créez un nom d'hôte gratuit (ex: `monprojet.ddns.net`).

Installation du client No-IP (DUC) sur le serveur : Ce programme met à jour l'IP automatiquement auprès de No-IP.

```
sudo apt install make gcc -y
cd /usr/local/src
sudo wget
[http://www.no-ip.com/client/linux/noip-duc-linux.tar.gz](http://www
.no-ip.com/client/linux/noip-duc-linux.tar.gz)
sudo tar xzf noip-duc-linux.tar.gz
cd noip-2.1.9-1
sudo make
sudo make install
```

Lors de l'installation, renseignez vos identifiants No-IP. Le service s'assurera que `monprojet.ddns.net` pointe toujours vers la bonne IP.

Double sécurité de phpMyAdmin avec un `.htaccess`

L'interface phpMyAdmin est un point critique. Nous allons y ajouter une authentification HTTP supplémentaire avant même d'afficher la page de connexion.

Autoriser l'utilisation des `.htaccess` pour phpMyAdmin :

```
sudo nano /etc/apache2/conf-available/phpmyadmin.conf
```

Dans le bloc `<Directory /usr/share/phpmyadmin>`, ajoutez : `AllowOverride All.`

Création du fichier de mots de passe `.htpasswd` :

```
sudo htpasswd -c /etc/phpmyadmin/.htpasswd admin_pma
```

Saisissez un mot de passe sécurisé. Ce fichier stocke les identifiants chiffrés.

Création du `.htaccess` :

```
sudo nano /usr/share/phpmyadmin/.htaccess
```

Ajoutez le contenu suivant :

```
AuthType Basic
AuthName "Acces Restreint"
AuthUserFile /etc/phpmyadmin/.htpasswd
Require valid-user
```

Redémarrage d'Apache :

```
sudo systemctl restart apache2
```

Déploiement du site et génération du certificat SSL (HTTPS)

Maintenant que le serveur est prêt et accessible via `monprojet.ddns.net`, nous pouvons sécuriser le trafic.

1. **Déploiement des fichiers du site** : Placez les fichiers HTML/PHP de votre projet dans `/var/www/html/`.

Installation de Certbot (Let's Encrypt) : Certbot est un outil qui génère et installe automatiquement des certificats SSL gratuits.

```
sudo apt install certbot python3-certbot-apache -y
```

Génération du certificat :

```
sudo certbot --apache -d monprojet.ddns.net
```

Certbot va analyser la configuration d'Apache, communiquer avec Let's Encrypt pour valider que vous possédez le domaine (en passant par le port 80), générer le certificat, et reconfigurer Apache pour forcer la redirection vers le port 443 (HTTPS).

Conclusion

Le serveur est désormais en production :

- Le code source de l'application est déployé et accessible mondialement.
- Le trafic est chiffré via HTTPS.
- L'administration distante (SSH) utilise un port non-standard.
- La gestion de la base de données (phpMyAdmin) est protégée par une double authentification.