# Serveur LAMP - 3WA

v1.0.0 | 23/01/2025 | Auteur : Bauer Baptiste

Livre

# Table des matières

# 1. Configuration complète d'un serveur LAMP sous UBUNTU 24.04

*Version 1.0.0 | Dernière mise à jour : 20/01/2025*
*Durée de réalisation : 4*
*Auteur : Bauer Baptiste*

**Compétences ciblées**

- ☐ Installer et configurer un serveur web (Apache).
- ☐ Installer et configurer PHP avec Apache.
- ☐ Installer et sécuriser MySQL.
- ☐ Installer phpMyAdmin pour administrer MySQL via une interface web.
- ☐ Installer et configurer NodeJS.
- ☐ Installer MongoDB et créer une base de données simple.
- ☐ Sécuriser un serveur Linux (firewall, gestion des permissions).
- ☐ Créer et gérer des utilisateurs FTP avec accès restreint.

## 1.1. Etape 1 : Mise à jour du système

Pour garantir le bon fonctionnement de votre machine virtuelle, il est essentiel de procéder à une mise à jour régulière des paquets logiciels. Cette opération permet de bénéficier des dernières corrections de bugs et des améliorations de sécurité.

**Q1)** Quelles commandes permettent de mettre à jour l'ensemble des paquets logiciels d'une machine Linux ?

- Réponse :

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

## 1.2. Étape 2 : Installation Apache, PHP et MySQL

Pour configurer un serveur web complet, il est nécessaire d'installer les composants suivants :

- Apache : serveur web open source, largement utilisé pour héberger des sites web.
- PHP : langage de programmation côté serveur, indispensable pour la création de sites

dynamiques.

- MySQL : système de gestion de base de données relationnelle, permettant de stocker et d'interroger des données.

**Q2)** Quelles commandes permettent d'installer Apache, PHP et MySQL sur une machine Linux ? Vérifier ensuite l'installation d'apache en ouvrant un navigateur web et en saisissant l'adresse IP de la machine virtuelle.

*Correction de Q2*

```
sudo apt install -y apache2 php libapache2-mod-php mysql-server php-mysql
```

Installe plusieurs paquets sur une distribution Linux basée sur Debian (comme Ubuntu). Voici en détail ce qu'elle fait :

- `apache2`

  Installe le serveur web Apache, qui permet d'héberger et de servir des sites web statiques ou dynamiques.

  Après installation, Apache écoute par défaut sur le port 80.

- `php` :

  Installe l'interpréteur PHP permettant au serveur web Apache d'exécuter du code PHP.

  Ceci rend possible l'utilisation de scripts dynamiques côté serveur pour générer des pages web interactives.

- `libapache2-mod-php` :

  Installe le module d'intégration de PHP directement au sein du serveur Apache, permettant à Apache d'exécuter directement les scripts PHP.

- `mysql-server` :

  Installe le système de gestion de base de données relationnelle MySQL.

  MySQL est utilisé pour stocker, gérer et interagir avec des données structurées en SQL.

- `php-mysql` :

  Installe l'extension PHP nécessaire à la communication entre PHP et la base de données MySQL.

  Permet à PHP d'envoyer des requêtes SQL et de gérer les résultats issus de la base MySQL.

## 1.3. Etape 3 : Configuration avancée d'Apache pour un projet Symfony et un site web classique

Pour personnaliser la configuration d'Apache, il est nécessaire de modifier certains fichiers de configuration. Ces fichiers sont situés dans le répertoire `/etc/apache2/` ou d'activer des modules spécifiques.

**Q3)** Quelles sont les modules Apache nécessaires pour exécuter un projet Symfony ? Comment les activer ?

*Correction de Q3*

Symfony requiert ces modules :

- `mod_rewrite` : permet de réécrire les URL pour les rendre plus lisibles et plus SEO-friendly.
- `mod_php` : permet d'exécuter du code PHP directement dans les fichiers HTML.
- `mod_ssl` : permet de sécuriser les échanges entre le serveur et le client via le protocole HTTPS.
- `mod_headers` : permet de manipuler les en-têtes HTTP pour ajouter des informations supplémentaires.

Pour activer ces modules, utilisez la commande `a2enmod` suivie du nom du module.

```
sudo a2enmod rewrite
sudo a2enmod headers
sudo systemctl restart apache2
```

## 1.4. Etape 4 : Configuration des VirtualHost pour héberger plusieurs sites web

Nous pouvons modiifier les fichiers appelés "VirtualHost" pour configurer les paramètres spécifiques à chaque site web hébergé sur le serveur Apache.

Imaginons que nous ayons deux sites web à héberger sur notre serveur :

- `projet1.symfony.3wa` : Projet 1 développé avec Symfony.
- `projet2.legacy.3wa` : Projet 2 développé avec PHP et HTML sans framework.

**Q4)** Comment configurer les **VirtualHost** pour héberger ces deux sites web sur un même serveur Apache ?

*Correction de Q4*

Créez un fichier de configuration pour chaque site web dans le répertoire `/etc/apache2/sites-available/`.

Voici un exemple de configuration pour le site `projet1.symfony.3wa` :

*Fichier* `/etc/apache2/sites-available/projet1.conf`

```
<VirtualHost *:80>
    ServerName projet1.symfony.3wa
    DocumentRoot /var/www/projet1/public
    <Directory /var/www/projet1/public>
        AllowOverride None
        Require all granted
        # Activer mod_rewrite et gérer le .htaccess
        <IfModule mod_rewrite.c>
            Options -MultiViews
            RewriteEngine On
            RewriteCond %{REQUEST_FILENAME} !-f
            RewriteRule ^(.*)$ index.php [QSA,L]
        </IfModule>
        # Sécuriser les accès aux fichiers sensibles
        <FilesMatch ".+\.php$">
            Require all granted
        </FilesMatch>
    </Directory>
    # Logs d'erreurs et d'accès
    ErrorLog ${APACHE_LOG_DIR}/symfony_error.log
    CustomLog ${APACHE_LOG_DIR}/symfony_access.log combined
</VirtualHost>
```

La configuration pour le site `projet2.legacy.3wa` sera similaire, mais adaptée aux besoins spécifiques du projet.

*Fichier* `/etc/apache2/sites-available/projet2.conf`

```
<VirtualHost *:80>
    ServerName projet2.legacy.3wa
    DocumentRoot /var/www/projet2
    <Directory /var/www/projet2>
        AllowOverride None
        Require all granted
    </Directory>
    ErrorLog ${APACHE_LOG_DIR}/legacy_error.log
    CustomLog ${APACHE_LOG_DIR}/legacy_access.log combined
</VirtualHost>
```

Ensuite, activez les sites avec la commande `a2ensite` et redémarrez Apache.

```
sudo a2ensite projet1
sudo a2ensite projet2
sudo systemctl restart apache2
```

- **ServerName** : votre domaine local ou public.
- **DocumentRoot** : le répertoire public de Symfony.
- Le bloc `<Directory>` active la réécriture d'URL nécessaire à Symfony (tout est réécrit vers `index.php` s'il n'existe pas physiquement).

## 1.5. Etape 5 : Droit d'accès aux fichiers et répertoires

Pour des raisons de sécurité, il est essentiel de définir les bons droits d'accès aux fichiers et répertoires de votre serveur web. Cela permet de limiter les risques.

Assurez-vous que le serveur web peut accéder aux fichiers de cache et log de Symfony.

**Q5)** Quels droits d'accès sont recommandés pour les fichiers et répertoires d'un projet Symfony ?

Quelle commande permet de les appliquer ?

*Correction de Q5*

Les droits d'accès recommandés pour un projet Symfony sont les suivants :

- Les répertoires doivent avoir les droits `755` (`rwxr-xr-x`).
- Les fichiers doivent avoir les droits `644` (`rw-r--r--`).
- L'utilisateur www-data doit avoir les droits en lecture et écriture sur les répertoires `var` et `public`.

Pour appliquer ces droits, utilisez la commande `chmod` suivie des droits et du nom du fichier ou répertoire.

```
sudo chmod -R 755 /var/www/projet1
sudo chmod -R 644 /var/www/projet1/var
sudo chmod -R 644 /var/www/projet1/public
sudo chown -R www-data:www-data /var/www/projet1/var
```

## 1.6. Etape 6 : Configuration de MySQL

Pour sécuriser votre serveur MySQL, il est recommandé de définir un mot de passe pour

l'utilisateur `root` et de supprimer les utilisateurs inutiles.

**Q6)** Comment définir un mot de passe pour l'utilisateur `root` de MySQL ? Comment supprimer un utilisateur inutile ?

*Correction de Q6*

Pour définir un mot de passe pour l'utilisateur `root` de MySQL, utilisez la commande suivante :

```
sudo mysql_secure_installation
```

Cette commande vous guidera pour définir un mot de passe sécurisé pour l'utilisateur `root` de MySQL. Suivez les instructions à l'écran. (Mot de passe root, supprimer utilisateurs anonymes, désactiver accès distant root, supprimer base de test, recharger les privilèges.)

## 1.7. Etape 7 : Installation de phpMyAdmin

phpMyAdmin est une interface web permettant de gérer les bases de données MySQL de manière graphique. Cela facilite la création, la modification et la suppression de bases de données et de tables.

**Q7)** Comment installer phpMyAdmin sur une machine Linux ? Comment y accéder via un navigateur web ?

*Correction de Q7*

Pour installer phpMyAdmin, utilisez la commande suivante :

```
sudo apt install -y phpmyadmin
```

Choisissez Apache lorsque demandé, puis configurez la base de données automatiquement.

- Vérifiez l'accès à phpMyAdmin via : http://<ADRESSE_IP>/phpmyadmin

## 1.8. Etape 8 : Installation et configuration de NodeJS

Node.js est un environnement d'exécution JavaScript côté serveur, basé sur le moteur JavaScript V8 de Google Chrome. Il permet d'exécuter du code JavaScript côté serveur.

**Q8)** Comment installer NodeJS sur une machine Linux ? Comment vérifier l'installation de NodeJS et de npm ?

*Correction de Q8*

```
sudo apt install -y nodejs npm
```

Vérifiez l'installation en affichant les versions de NodeJS et de npm.

```
node -v
npm -v
```

## 1.9. Etape 9 : Installation de MongoDB

MongoDB est une base de données NoSQL orientée document, qui stocke les données sous forme de documents JSON. Elle est très utilisée pour les applications web modernes.

**Q9)** Comment installer MongoDB sur une machine Linux ? Comment vérifier que MongoDB est en cours d'exécution

*Correction de Q9*

Pour installer MongoDB, utilisez la commande suivante :

```
sudo apt install -y mongodb
sudo systemctl

start mongodb
sudo systemctl enable mongodb
```

Vérifiez que MongoDB est en cours d'exécution en utilisant la commande suivante :

```
sudo systemctl status mongodb
```

## 1.10. Etape 10 : Sécurisation du serveur Linux

Pour sécuriser votre serveur Linux, il est essentiel de mettre en place un pare-feu et de gérer les permissions des utilisateurs.

**Q10)** Comment configurer un pare-feu sur une machine Linux ? Comment gérer les permissions des utilisateurs pour renforcer la sécurité du serveur ? Lister les services à autoriser dans le pare-feu pour un serveur web comme celui que nous avons configuré.

*Correction de Q10*

Pour configurer un pare-feu sur une machine Linux, utilisez la commande suivante :

```
sudo ufw enable
sudo ufw allow ssh
sudo ufw allow http
sudo ufw allow https
```

**Q11)** Comment sécuriser au mieux l'accès SSH à votre serveur Linux ?

*Correction de Q11*

Pour sécuriser l'accès SSH à votre serveur Linux, suivez ces recommandations :

- Désactivez l'accès root via SSH.
- Utilisez des clés SSH pour l'authentification.
- Limitez les utilisateurs autorisés à se connecter via SSH.
- Changez le port SSH par défaut (22) pour un port non standard.
- Surveillez les tentatives de connexion SSH échouées.

Pour désactiver l'accès root via SSH, éditez le fichier `/etc/ssh/sshd_config` et définissez `PermitRootLogin no`.

Pour limiter les utilisateurs autorisés à se connecter via SSH, éditez le fichier `/etc/ssh/sshd_config` et définissez `AllowUsers user1 user2`.

Pour changer le port SSH par défaut, éditez le fichier `/etc/ssh/sshd_config` et définissez `Port 2222` (par exemple).

Pour surveiller les tentatives de connexion SSH échouées, consultez les logs d'authentification SSH (`/var/log/auth.log`).

N'oubliez pas de redémarrer le service SSH après avoir modifié la configuration :

```
sudo systemctl restart sshd
```

**Q12)** Quand trop de tentative de connexion échoue, comment bloquer l'adresse IP de l'attaquant ?

*Correction de Q12*

Pour bloquer l'adresse IP d'un attaquant après un certain nombre de tentatives de connexion échouées, utilisez `fail2ban`.

## 1.11. Etape 11 : Création service FTP

**Q13)** Sur Ubuntu, nous utiliserons le service FTP `SFTP`. Comment installer SFTP et vérifier que le service fonctionne ?

*Correction de Q13*

Ubuntu utilise par défaut OpenSSH pour fournir SFTP :

`openssh-server` contient automatiquement SFTP. Aucun paquet supplémentaire requis.

Vérifiez que SSH tourne bien après l'installation :

```
sudo systemctl status ssh
```

Si le service n'est pas actif, lancez-le et activez son démarrage automatique :

```
sudo systemctl enable ssh
sudo systemctl start ssh
```

**Q14)** Comment créer un groupe et un utilisateur SFTP sécurisé ?

*Correction de Q14*

Pour sécuriser l'accès via SFTP, créez un groupe spécifique (`sftpusers`) et un utilisateur (`sftpuser`) :

```
sudo addgroup sftpusers
sudo useradd -g sftpusers -s /usr/sbin/nologin -m sftpuser
```

```
sudo passwd sftpuser
```

- `-s /usr/sbin/nologin` empêche l'utilisateur de se connecter par shell classique, limitant l'accès à SFTP uniquement.

**Q15)** Donner les droits corrects sur le dossier web afin de permettre à Apache et SFTP d'accéder au dossier web.

*Correction de Q15*

Pour permettre à Apache et SFTP d'accéder au dossier web :

- `/var/www` appartient à `root` pour permettre le `chroot` sécurisé.

```
sudo chown root:root /var/www
```

- Assurez-vous que le dossier `html` appartient à l'utilisateur `SFTP` et au groupe `www-data`

```
sudo chown -R sftpuser:www-data /var/www/html
```

- Appliquer les bonnes permissions :

```
sudo chmod 755 /var/www
sudo chmod -R 775 /var/www/html
```

- Ajoutez l'utilisateur **SFTP** au groupe Apache pour une compatibilité optimale :

```
sudo usermod -aG www-data sftpuser
```

`/var/www/html` appartient à sftpuser pour permettre l'écriture via SFTP, et au groupe `www-data` pour permettre à Apache de lire/servir les fichiers web.

**Q16)** Modifier la configuration **SSH** pour pointer vers `/var/www`

*Correction de Q16*

Modifiez le fichier suivant :

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

Ajoutez en fin de fichier :

```
Match Group sftpusers
    ChrootDirectory /var/www
    ForceCommand internal-sftp
    AllowTcpForwarding no
    X11Forwarding no
```

Le dossier défini dans `ChrootDirectory` doit impérativement appartenir à `root:root`. C'est pourquoi on définit `/var/www` en root, mais `/var/www/html` appartient à l'utilisateur SFTP.

Redémarrer SSH pour appliquer les modifications : `sudo systemctl restart ssh`