# MISE EN PLACE D'UN SERVEUR LAMP
# Linux – Apache – MySQL - PHP

**Contexte :**

Actuellement en stage, vous évoluez au sein de la société SCIS (Saint Charles Informatique Service) qui propose à ses clients des solutions tant dans le domaine applicatif et web que système et réseau.
Un client de SCIS souhaite disposer d'un site web propulsé par le CMS Wordpress. Sans compétence technique, il demande à la société SCIS d'héberger dans ses propres locaux les services qui permettront d'accéder à sa page web. Il souhaite la création d'une solution clé en main dont il n'aura qu'à gérer le contenu via l'interface de gestion de Wordpress.
Vous êtes en charge de la mise à jour du service DNS ainsi que de l'installation et la configuration du ou des serveurs répondant au besoin du client.

**Objectif :**

Mettre à disposition une solution sécurisée permettant l'utilisation d'applications web nécessitant l'accès à une base de données.

Les outils utilisés seront :
- Linux Debian : Système d'exploitation installé sur le serveur
- Apache : Serveur WEB
- MariaDB : SGBD
- PHP : Langage de développement

Cet objectif final sera atteint au travers de 3 objectifs intermédiaires :
1) Serveur WEB, SGBD, et outil d'administration PHPmyAdmin
2) CMS Wordpress
3) Séparation des services

**Prérequis :**

Vous disposez d'un service DNS fonctionnel (principal uniquement ou principal – secondaire)

Afin d'obtenir un serveur LAMP, créez une nouvelle machine virtuelle à partir d'un modèle existant.

Configuration de la machine :
Nom du poste : XX-Srv_LAMP
Mémoire vive : 512Mo
Réseau : configuration permettant l'accès à Internet pour l'installation des paquets

**Réalisation :**

## Objectif 1 : Serveur WEB, SGBD et outil d'administration PHPmyAdmin

### Etape 1 : Création d'un utilisateur disposant des droits root

Il est préférable d'utiliser un utilisateur différent de root pour administrer notre serveur LAMP. Vous pouvez utiliser un compte précédemment créé ou bien créer un nouveau compte (exemple : lampuser) avec cette commande :

```
# adduser lampuser
```
*(notez soigneusement le mot de passe de cet utilisateur)*

Attribuez ensuite les droits root à cet utilisateur :

```
# gpasswd -a lampuser sudo
```

Si sudo n'est pas installé sur votre système, il sera nécessaire de l'installer au préalable :

```
# apt install sudo
```

Authentifiez-vous ensuite en lampuser :

```
# su lampuser
```

### Etape 2 : Installation d'Apache

Pour installer le serveur WEB Apache, mettez à jour la liste des paquets disponibles puis installez le paquet apache2 :

```
$ sudo apt update
$ sudo apt install apache2
```

Les paquets dont dépend le paquet apache2 seront également installés, vous pourrez ensuite accéder à votre page index.html contenu dans `/var/www/html/` via l'url http://ip.de.votre.serveur/ depuis un poste client du même réseau.

### Etape 3 : Installation de Mariadb

```
$ sudo apt install mariadb-server php7.4-mysql ~~php7.4-mcrypt~~
```

### Etape 4 : Installation de PHP

Pour installer le langage de développement PHP, exécutez la commande suivante :

```
$ sudo apt install php7.4 libapache2-mod-php7.4
```

Etape 5 : Installation de phpMyAdmin

Afin d'administrer plus aisément votre SGBD, installez phpMyAdmin. Vous disposerez ainsi d'une interface web pour l'administration de vos bases de données.

```
$ sudo apt install phpmyadmin apache2-utils
```

Renseignez les paramètres suivants lors de l'installation :

Serveur WEB : Apache2
Configuration de base de phpMyAdmin : oui
Renseigner un **nouveau** mot de passe root phpMyAdmin *(n'oubliez pas de le noter)*

Modifiez la configuration du serveur Apache pour la prise en compte de l'installation de phpMyAdmin :

```
$ sudo nano /etc/apache2/apache2.conf
```

Ajoutez à la fin du fichier la ligne :

```
Include /etc/phpmyadmin/apache.conf
```

Puis redémarrer le serveur Apache :

```
$ sudo systemctl restart apache2
```

Vous pouvez maintenant accéder à phpMyAdmin via l'URL : http://ip.de.votre.serveur/phpmyadmin

utilisateur : phpmyadmin
mot de passe : celui-créé-précédemment *(soigneusement inscrit dans vos notes)*

S'il vous est notifié l'absence d'un paquet lors de l'accès à phpMyAdmin, il vous suffit de l'installer via la commande :

```
$ sudo apt install php7.4-nomdupackage
```

Etape 6 : Prioriser la lecture des fichiers php

Mettez en application la configuration vue lors du TP02 pour favoriser depuis un navigateur client l'accès aux fichiers `index.php` et non `index.htm` ou `index.html`

Editez le fichier de configuration du module concerné :

```
$ sudo nano /etc/apache2/mods-enabled/dir.conf
```

Modifiez la directive DirectoryIndex en renseignant « `index.php` » en première position comme ceci :

```
DirectoryIndex index.php index.html . . .
```

N’oubliez pas de relancer la lecture de la configuration du serveur Apache :

```
$ sudo systemctl reload apache2
```

Etape 7 : Sécuriser l’accès à phpMyAdmin

Mettez en application la configuration vue lors du TP05 pour sécuriser l’accès à phpMyAdmin.

Editez le fichier de configuration de phpmyadmin :

```
$ sudo nano /etc/phpmyadmin/apache.conf
```

Rajouter après la directive DirectoryIndex la directive :

```
AllowOverride All
```

Créez ensuite un fichier .htaccess dans le dossier phpMyAdmin :

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

Et renseignez :

```
AuthType Basic
AuthName « Acces reserve aux administrateurs »
AuthUserFile /etc/apache2/.phpmyadmin.htpasswd
Require user lampuser
```

Enfin, créez le fichier htpasswd :

```
$ sudo htpasswd -c /etc/apache2/.phpmyadmin.htpasswd lampuser
```

Renseignez le mot de passe de lampuser puis redémarrer le service Apache :

```
$ sudo systemctl restart apache2
```

**Quel est le rôle de ce fichier .htaccess ?**

Etape 8 : Sécuriser le SGBD Mariadb

Mariadb fournit des utilitaires pour « nettoyer » l’installation, et ainsi retirer des bases et/ou tables inutiles et potentiellement dangereuses.

Lancez l’utilitaire :

```
$ sudo mysql_secure_installation
```

Saisissez votre mot de passe root Mariadb, ne modifiez pas la connexion par le socket unix comme proposé et ne modifiez pas votre mot de passe root comme proposé, puis validez tous les choix proposés par défaut.

Modifiez maintenant la manière de se connecter à votre SGBD en root :

```
$ sudo mysql -u root -p
```

```
USE mysql;
ALTER USER root@localhost IDENTIFIED VIA mysql_native_password USING PASSWORD("votremotdepasse");
FLUSH PRIVILEGES;
exit;
```

Redémarrez ensuite votre serveur Mariadb :

```
$ sudo systemctl restart mariadb
```

Vous pouvez dorénavant vous connecter à phpMyAdmin avec le login root:votremotdepasse

**Quels sont les ports à l'écoute sur votre serveur ? Associez chaque port à son service.**
**Quelle est la particularité de l'écoute du démon Mariadb ?**

## Objectif 2 : CMS Wordpress

### Etape 9 : Installer le CMS Wordpress sur votre machine LAMP

Récupérez le fichier compressé comprenant les fichiers d'installation du CMS Wordpress :

```
$ sudo wget --no-check-certificate https://fr.wordpress.org/latest-fr_FR.tar.gz
```

Décompressez le fichier ainsi récupéré :

```
$ sudo tar -xvzf latest-fr_FR.tar.gz
```

Renommez le dossier extrait en `SISR` et déplacez-le à la racine du serveur WEB :

```
$ sudo mv wordpress/ /var/www/html/SISR
```

Modifiez le propriétaire du dossier `/var/www/html/SISR` :
*(l'utilisateur* `www:data` *est celui qui lance les processus Apache2)*

```
$ sudo chown -R www-data:www-data /var/www/html/SISR
```

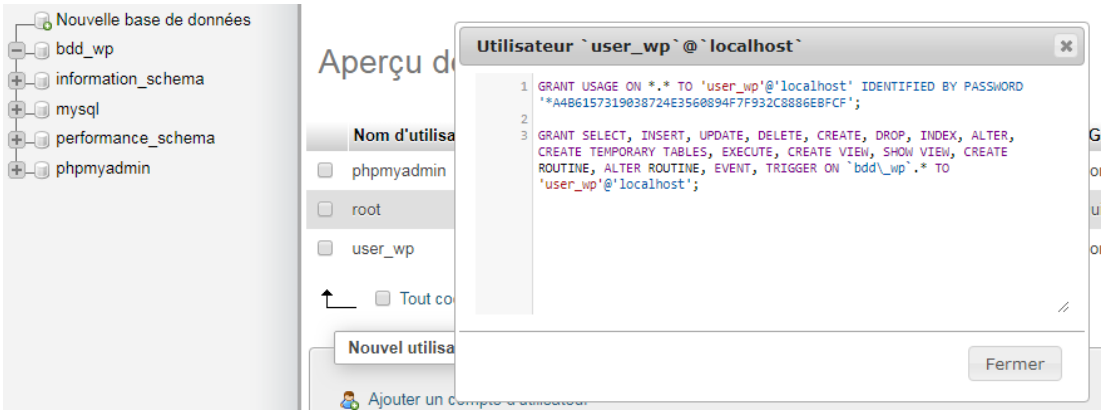
Rendez accessible par n'importe qui le dossier SISR et son contenu pour assurer l'installation correcte du CMS Wordpress :

```
$ sudo chmod -R 777 /var/www/html/SISR/
```

**Quel est le rôle de l'option '`-R`' dans les deux commandes précédentes ?**

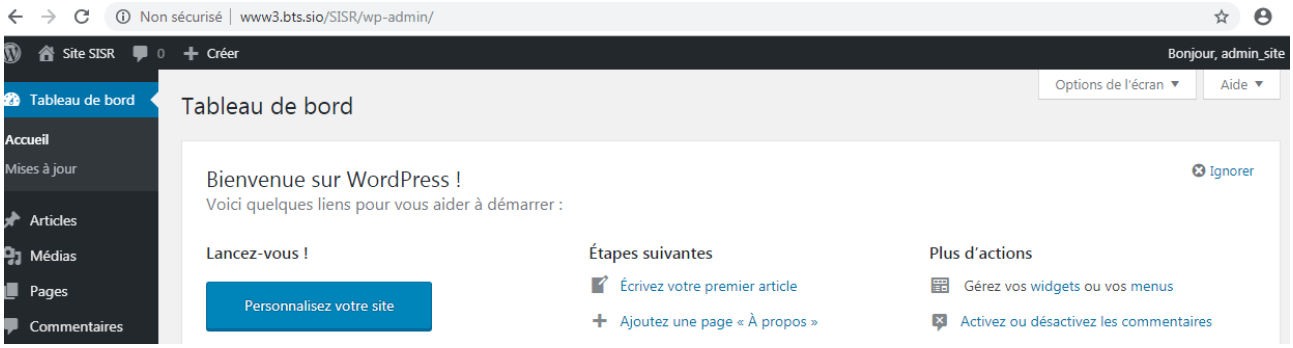
Vous pouvez à présent passer à l'installation du CMS Wordpress grâce à l'interface web de ce dernier via l'URL http://www3.bts.sio/SISR : *(mettez à jour votre serveur DNS)*


N'oubliez pas cependant de créer la base de données ainsi que l'utilisateur ayant les droits sur cette base. Utilisez l'application *phpmyadmin* pour réaliser ces tâches :



Vous renseignerez ces informations lors de l'installation de Wordpress :


Après avoir créé l'administrateur de votre site, vous pouvez accéder à son tableau de bord, prenez soin de noter l'identifiant que vous avez choisi :




## Etape 10 : Sécurisation de CMS Wordpress : *(pas nécessaire pour ce TP, absolument obligatoire en production)*

Modifiez les accès aux dossiers et fichiers contenus dans `/var/www/html/SISR` :

```
$ su  (utilisez le compte root pour la réalisation de cette étape)
# cd /var/www/html/SISR
# find ./ -type f -exec chmod 640 {} \;
# find ./ -type d -exec chmod 750 {} \;
# chmod 660 .htaccess
# find ./wp-content -type f -exec chmod 640 {} \;
# find ./wp-content -type d -exec chmod 770 {} \;
```

**Expliquez le rôle de chacune de ces commandes.**

## Objectif 3 : Séparation des services

Pour des raisons de sécurité, il est nécessaire de séparer le service WEB du service de base de données, et idéalement de placer le serveur WEB dans une DMZ publique et le SGBD dans une DMZ privée.
Vous effectuerez dans ce TP la séparation des services au niveau système. Vous allez ainsi installer le CMS Wordpress sur une autre machine Serveur WEB, qui utilisera la base de données présente dans votre SGBD. Une modification de votre SGBD sera nécessaire, en effet, par défaut, le serveur Mariadb n'accepte des connexions qu'à partir de lui-même (localhost).

### Etape 11 : Ajouter un nouveau serveur WEB à votre maquette

Utilisez une machine Serveur WEB existante ou créez une nouvelle machine virtuelle à partir d'un modèle.

Configuration de la machine :

- Nom du poste : XX-Srv_WEB
- Mémoire vive : 512Mo
- Réseau : configuration permettant l'accès à Internet pour l'installation des paquets

### Etape 12 : Création d'un utilisateur disposant des droits root

Il est préférable d'utiliser un utilisateur différent de root pour administrer notre serveur WEB. Vous pouvez utiliser un compte précédemment créé ou bien créer un nouveau compte (exemple : webuser) avec cette commande :

```
# adduser webuser
```
*(notez soigneusement le mot de passe de cet utilisateur)*

Attribuez ensuite les droits root à cet utilisateur :

```
# gpasswd -a webuser sudo
```

Si sudo n'est pas installé sur votre système, il sera nécessaire de l'installer au préalable :

```
# apt install sudo
```

Authentifiez-vous ensuite en webuser :

```
# su webuser
```

Etape 13 : Installation d'Apache et PHP

Pour installer le serveur WEB Apache, mettez à jour la liste des paquets disponibles puis installez le paquet apache2 :

```
$ sudo apt update
$ sudo apt install apache2 php7.4-mysql ~~php7.4-mcrypt~~
$ sudo apt install php7.4 libapache2-mod-php7.4
```

S'il vous est notifié l'absence d'un paquet lors de l'accès à votre application, il vous suffit de l'installer via la commande :

```
$ sudo apt install php7.4-nomdupackage
```

Etape 14 : Prioriser la lecture des fichiers php

Mettez en application la configuration vue lors du TP02 pour favoriser depuis un navigateur client l'accès aux fichiers `index.php` et non `index.htm` ou `index.html`

Editez le fichier de configuration du module concerné :

```
$ sudo nano /etc/apache2/mods-enabled/dir.conf
```

Modifiez la directive DirectoryIndex en renseignant « `index.php` » en première position comme ceci :

```
DirectoryIndex index.php index.html . . .
```

N'oubliez pas de relancer la lecture de la configuration du serveur Apache :

```
$ sudo systemctl reload apache2
```

Etape 15 : Autoriser le SGBD à recevoir des connexions depuis d'autres serveurs

Comme vous l'avez remarqué à l'étape 8, votre SGBD n'écoute sur le port 3306 uniquement les requêtes qu'il émet lui-même. Modifiez l'option `bind-address` de votre SGBD pour lui permettre d'écouter les autres serveurs :

```
$ sudo nano /etc/mysql/mariadb.conf.d/50-server.cnf
```

Remplacez l'option `bind-address = 127.0.0.1` par `bind-address = 0.0.0.0`

N'oubliez pas de redémarrer votre SGBD :

```
$ sudo systemctl restart mariadb
```

**Quels sont les ports à l'écoute sur votre serveur ? Que remarquez-vous par rapport à l'étape 8 ?**

Etape 16 : Installer le CMS Wordpress sur votre machine Serveur WEB

Récupérez le fichier compressé comprenant les fichiers d'installation du CMS Wordpress :

```
$ sudo wget --no-check-certificate https://fr.wordpress.org/latest-fr_FR.tar.gz
```

Décompressez le fichier ainsi récupéré :

```
$ sudo tar -xvzf latest-fr_FR.tar.gz
```

Renommez le dossier extrait en `SLAM` et déplacez-le à la racine du serveur WEB :

```
$ sudo mv wordpress/ /var/www/html/SLAM
```

Modifiez le propriétaire du dossier `/var/www/html/SLAM` :
*(l'utilisateur* `www:data` *est celui qui lance les processus Apache2)*

```
$ sudo chown -R www-data:www-data /var/www/html/SLAM
```

Rendez accessible par n'importe qui le dossier `SLAM` et son contenu pour assurer l'installation correcte du CMS Wordpress :

```
$ sudo chmod -R 777 /var/www/html/SLAM/
```

Suivez ensuite les configurations réalisées à l'étape 9 sur votre SGBD :

- création d'une nouvelle base de données : `bdd_wp_web`
- création d'un utilisateur ayant les droits sur cette base et pouvant se connecter depuis n'importe quel client : `user_wp_web`
  (notez qu'il est possible de remplacer la variable '%' par l'adresse IP de votre serveur WEB afin de limiter l'accès à la base de données)


- installation du CMS Wordpress grâce à l'interface web de ce dernier via l'URL http://www.bts.sio/SLAM

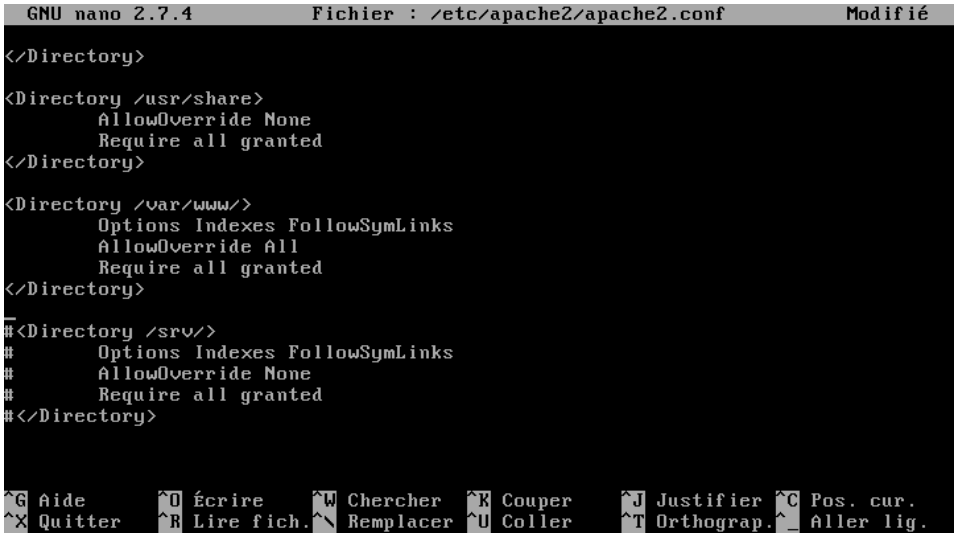
Notez l'identifiant de l'administrateur du site

Etape 17 : Sécuriser l'accès à votre site

Mettez en application la configuration vue lors du TP05 pour sécuriser l'accès à votre nouveau site.
Editez le fichier de configuration de apache2 :

```
$ sudo nano /etc/apache2/apache2.conf
```

Modifiez la directive `AllowOverride` dans la directive `<Directory /var/www/>`




Modifiez ensuite le fichier .htaccess contenu dans le dossier `SLAM` :

```
$ sudo nano /var/www/html/SLAM/.htaccess
```

Et renseignez :

```
AuthType Basic
AuthName « Acces reserve aux developpeurs »
AuthUserFile /etc/apache2/.phpmyadmin.htpasswd
Require user webuser
```

Enfin, créez le fichier htpasswd :

```
$ sudo htpasswd -c /etc/apache2/.phpmyadmin.htpasswd webuser
```

Renseignez le mot de passe de webuser puis redémarrer le service Apache :

```
$ sudo systemctl restart apache2
```

Etape 18 : Sécurisation de CMS Wordpress : *(pas nécessaire pour ce TP, absolument obligatoire en production)*

Modifiez les accès aux dossiers et fichiers contenus dans `/var/www/html/SLAM` :

```
$ su  (utilisez le compte root pour la réalisation de cette étape)
# cd /var/www/html/SLAM
# find ./ -type f -exec chmod 640 {} \;
# find ./ -type d -exec chmod 750 {} \;
# chmod 660 .htaccess
# find ./wp-content -type f -exec chmod 640 {} \;
# find ./wp-content -type d -exec chmod 770 {} \;
```