PROJET INFRASTRUCTURE

Systèmes, base de données et scripting



# Sommaire

[AVANT-PROPOS 3](#_Toc5289319)

[PREREQUIS 3](#_Toc5289320)

[OBJECTIFS 3](#_Toc5289321)

[3…2…1 : Attendez et lisez ça avant 3](#_Toc5289322)

[Partie 1 : Machine Virtuelle 4](#_Toc5289323)

[Partie 2 : Environnement 4](#_Toc5289324)

[Partie 3 : Premier site 5](#_Toc5289325)

[Partie 4 : Accès au site & sécurisation 5](#_Toc5289326)

[Partie 5 : Configuration de la base de données 6](#_Toc5289327)

[Partie 6 : Installation d’un nouveau site 6](#_Toc5289328)

[Partie 7 : Serveur de Pré-Prod / Serveur de Production 7](#_Toc5289329)

[Partie 8 : Automatisation 8](#_Toc5289330)

[Partie 9 : Sécurisation du site et disponibilité 8](#_Toc5289331)

[Partie 10 : Disponibilité du site – Bonus 9](#_Toc5289332)

[Annexes 10](#_Toc5289333)

[Annexe 1 - Résultat du fichier info.php 10](#_Toc5289334)

[Annexe 2 – Schéma simplifié d’un système de Load Balancing 10](#_Toc5289335)

[Ressources 11](#_Toc5289336)

# AVANT-PROPOS

Ce « mini-projet » consiste à vous faire manipuler des outils et services systèmes afin de monter une infrastructure web viable et fiable. Pour un développeur moderne, il est essentiel de ne pas être effrayé par la mise en place d’une infra web et surtout, de nos jours il est indispensable d’acquérir plus de polyvalence pour gagner du temps et maitriser l’ensemble de son environnement.

# PREREQUIS

Afin de réussir ce projet, il est important d’avoir suivi des **modules** de formations **systèmes** (*Linux, Windows*) et **Réseaux**. Néanmoins, il est probable que certains aspects ne soient pas abordés en formation, il sera ainsi nécessaire de pousser la réflexion et vos recherches personnelles afin d’atteindre les objectifs de ce projet.

# OBJECTIFS

* Mettre en place un serveur et le configurer en vue d’héberger une solution web
* Installer, configurer et utiliser des outils reconnus pour permettre d’héberger des solutions web en toute fiabilité
* Déployer sur son serveur un site web type WordPress
* Automatiser des backups des fichiers du site et de sa base de données
* Tracer dans un fichier de log les différentes actions réalisées
* Rendre accessible la solution web sur un réseau local via son adresse IP mais aussi via un nom de domaine personnalisé
* Permettre une haute disponibilité du service dans son ensemble (cluster / redondance)
* Sécuriser l’ensemble des services et transactions par le biais de protocole reconnus

# 3…2…1 : Attendez et lisez ça avant

Dans le cadre de ce projet, il est préférable d’avoir une **seconde machine** *(deuxième VM par exemple)*, sur le même réseau ; Cela permettra à différentes étapes du travail demandé de tester certaines commandes, les différents accès mais aussi **pour répondre au chapitre « disponibilité du serveur »**

Par ailleurs, si vous le souhaitez (*et c’est conseillé*), vous pouvez **réaliser ce projet en binôme**. Cependant, si tel est le cas, **celui qui ne pratique pas devra prendre des notes** en temps réel afin de réaliser ce qui se rapproche le plus d’une **procédure**. Cela aura pour bénéfice de vous permettre de garder des traces de vos différentes manipulations et permettra également de bien valider la compréhension des actions menées.

**Cependant, comme mentionné plus haut, il sera nécessaire à certaines étapes du projet d’avoir 2 machines sur le même réseau, il est donc recommandé de pratiquer en même temps !**

## Partie 1 : Machine Virtuelle

Déployer une machine virtuelle **Linux Server** (distribution de votre choix).

Vous utiliserez une version Linux sans interface graphique. Cela permet de diminuer la surface d’attaque, certaines options ne sont accessibles qu’en ligne de commande et cela permet également d’optimiser l’utilisation des ressources (*affichage gourmand*).

**INFO** : La réalisation de ce TP sur Windows Server est envisageable, mais vous serez limité dans certaines actions et bridés sur certains paramétrages.

Il faudra créer un utilisateur « cesi » avec le mot de passe de votre choix.

## Partie 2 : Environnement

* **Lancer le serveur** et se connecter.
* **Créer** un utilisateur « cesi » si ce n’est pas déjà fait.
* Nommer la machine
* Mettre une IP fixe sur le serveur.
* Installer les outils suivants :
  + **Apache**
  + **PHP version 7** (*php7.x*)
  + **Une base de données relationnelles**
    - *Attention à bien noter le mot de passe root*
  + **Modules PHP** nécessaires
* **Démarrer** Apache et s’assurer que le service démarre systématiquement en même temps que la machine
* **Tester** depuis la machine hôte (*votre PC*) si le **serveur Apache** (*de la machine virtuelle*) répond correctement (*en tapant l’adresse IP du serveur web*)
  + *Cela devrait afficher un message par défaut (de votre distribution) qui prouve le bon fonctionnement*
* **Tester** si PHP est bien fonctionnel également
  + **Créer** un **fichier** *info.php dans le* ***répertoire www/html*** *et* y **ajouter** :

<?php  
phpinfo();  
?>

* + **Afficher** ensuite depuis le navigateur de votre machine hôte le fichier. Ce qui affichera la configuration de votre serveur web
    - Vous devriez avoir [ça](#_Résultat_du_fichier) (*voir Annexe 1 – résultat du fichier info.php*)

**Si tout est ok, cette partie est terminée !**

**Faites un screenshot du navigateur web. Veillez à stipuler dans le nom du fichier le numéro de la partie validée.**

## Partie 3 : Premier site

Une fois Apache et PHP bien fonctionnels, vous pouvez créer une première page web.

**INFO** : *Vous aurez pris le soin d’affecter une IP fixe sur votre VM, ce qui simplifiera les différents tests, vous pouvez même créer un alias dans le fichier* ***hosts*** *de la machine hôte.*

Se rendre dans le dossier ***www*** du serveur web (VM) :

* **Créer** un nouveau dossier « **monsite** » à la racine
* **Créer** un fichier « **index.php** » dans ce nouveau dossier
* **Ajouter** du code PHP dans le fichier (*une commande* ***echo*** *par exemple*)
* **Essayer** d’atteindre le site depuis sa machine hôte

## Partie 4 : Accès au site & sécurisation

### Sécurité

* **Installer** SSH[[1]](#footnote-1) si nécessaire
* **Configurer** **SSH** si besoin
* **S’assurer** qu’il est bien possible de se **connecter** en **SSH** depuis une autre machine[[2]](#footnote-2)
  + *Vous devez générer* ***une authentification par certificat*** *si ce n’est pas déjà fait*
* Mise en place d’un mécanisme de sécurité pour éviter les attaques brutes forces (de mot de passe)
* Configurer le pare-feu en adéquation avec votre stratégie (règles / ports)

**INFO** *: Il devra être possible d’ajouter à tout moment un nouvel utilisateur « distant » pouvant se connecter en SSH avec son compte.*

### Client FTP

**Tester** depuis la machine hôte de se connecter avec un **client FTP** (exemple : WinSCP)

* **Créer** 2 utilisateurs virtuels « Dev1 » et « Dev2 » qui pourront interagir avec le serveur web
* Chaque utilisateur (développeur) aura un répertoire virtuel
* Chaque utilisateur accédera par défaut au répertoire « **monsite** » et ne pourra pas en sortir (*jail*)
* Les utilisateurs ne peuvent pas modifier les fichiers « *systèmes* »
* **Se connecter** en **SFTP** avec chaque utilisateur (tester)
* **Modifier** le fichier par défaut du site

**Tester** en affichant la page web « modifiée » dans le navigateur.

## Partie 5 : Configuration de la base de données

Se **connecter** à la base de données

Exemple :

*mysql -u root -p*

**Créer** une base de données, vous pourrez la nommer **cesibdd**

* *Cette base de données sera utile pour déployer un site (CMS) dans la suite de ce projet*

**Créer** un nouvel utilisateur « dibdd » et **attribuez**-**lui** les privilèges maximum sur cette nouvelle base de données

* *Ce nouvel utilisateur sera utilisé pour se connecter à la base et accéder aux données dans la suite de ce projet*

## Partie 6 : Installation d’un nouveau site

*Dans cette partie nous allons vous demander d’installer un nouveau site reposant sur un CMS (Wordpress ou Drupal) sur votre serveur actuel.*

### Téléchargement, installation & configuration

* **Télécharger** sur votre serveur la dernière version du CMS.
* **Placer** le CMS dans le répertoire de destination.
  + *Vous* ***devrez*** *le placer directement dans le répertoire* ***www*** *d’Apache*
  + *Veillez à bien* ***conserver*** *votre premier dossier* **monsite** (du dossier www)
* **Installer** le CMS
  + *En cas de difficulté, pensez à lire le fichier README.txt à la racine du projet (CMS)*
* **Créer** un nouveau vhost pointant sur le dossier
* **Se connecter** en administrateur sur le CMS nouvellement installé

### Modification CMS

* **Modifier** la page d’accueil de votre site
* **Ajouter** un ou deux contenus
* **Personnaliser** le site comme bon vous semble

## Partie 7 : Serveur de Pré-Prod / Serveur de Production

*Pour réaliser cette partie, il est conseillé d’être à deux (avec votre binôme). L’un jouera le rôle du serveur pré-prod, l’autre le rôle du serveur de production. Vous pourrez changer les rôles afin d’aborder chacun les objectifs*

Idéalement, vous serez dans la même configuration. Si ce n’est pas le cas, n’hésitez pas à dupliquer (cloner) le serveur[[3]](#footnote-3) le plus abouti / fonctionnel des deux.

* *Serveur A = Serveur de pré-production*
* *Serveur B = Serveur de production*

### Test

* Dans un premier temps essayez de **pinger** le **serveur A** depuis le **serveur B** avec leur **nom** (*et non pas leur adresse IP*)
* Essayez ensuite **d’afficher** le site depuis un **navigateur** du même réseau
* Connectez-vous en **SSH** sur le **serveur** **de développement (Serveur A)**

A ce stade, vous devriez avoir quelques contenus sur votre **serveur A** (*Cf. chapitre précédent – Modification CMS*). Si ce n’est pas le cas, merci de vous y reporter avant de passer à la suite.

### Préparation à la migration du site

*Ici, nous allons migrer (en prod) le site du serveur A vers le serveur B.*

1. Faire une **extraction** de la base de données (dump SQL) dans son état actuel
2. Connectez-vous en **SFTP** sur le serveur
3. **Récupérer** la base de données « dumpée »
4. **Récupérer** si nécessaire l’archive des dossiers « personnalisé » de votre CMS
   * *Ici, nous parlons des dossiers et fichiers qui ont pu être ajouté ou qui aurait pu évoluer depuis l’installation du CMS*
   * *Lire la documentation du CMS ou suivre un article / tuto expliquant comment migrer / déployer le site*

### Serveur de production

**Entre le serveur A et le serveur B**

1. **Récupérer** les sources (*dossiers du site CMS sauvegardés*) du **serveur A** et les **copier** sur le **serveur B**
2. **Importer** la BDD « *dumpées* » du **serveur A** vers le **serveur B**
3. **Modifier** les éventuels fichiers de configuration natifs du CMS de manière à ce que le site soit opérationnel sur le **serveur B.**
   * Lire la documentation officielle du CMS
   * Suivre un article / tuto au sujet de la migration / déploiement du CMS concerné
4. Depuis la machine hôte ou depuis le **serveur A**, tester le bon fonctionnement et affichage du site nouvellement déployé sur le **serveur B**.

## Partie 8 : Automatisation

### Créer un script qui permet :

* Lorsque l’on va créer un nouvel utilisateur son site web portant son nom sera créé.
* Le répertoire par défaut de l’utilisateur sera */home/<UTILISATEUR>/public\_html*
* Créer un répertoire **logs** par utilisateur
* Stocker l’horodatage de la création du compte dans un fichier

### Création d’un nouvel utilisateur

* **Créer** un nouvel utilisateur « **cesien** » (*ce sera votre profil pour le développement*)
* **Assurez-vous** que l’on puisse **accéder** à la page web[[4]](#footnote-4) de l’utilisateur via son nom

**Il vous sera demandé de faire une démonstration lors de la correction.**

### Sauvegarde / backup

**Créer** une tâche planifiée (script) permettant **d’automatiser** la sauvegarde quotidienne de la base de données du site du serveur de production.

## Partie 9 : Sécurisation du site et disponibilité

*Dans cette partie nous allons vous demander de réaliser quelques opérations pour sécuriser votre site. Vous pourrez appliquer cela sur le dossier* ***public\_html*** *de l’utilisateur* ***cesien*** *ou sur le projet «****monsite****» du répertoire* ***www****.*

### HTTPS

* **Activer** le module SSL dans Apache

**INFO** : Par défaut, si vous activez seulement le **mod\_ssl** et que vous relancez les services Apache, vous avez un certificat « **Self-Signed** ». Le navigateur ne considère pas ce genre de certificat comme « **fiable** », ce qui a pour effet d’alerter le visiteur de votre site (*avec un message demandant s’il fait confiance au site afin d’y accéder*).

### Conf Apache

* **Mettre en place** l’url rewriting sur votre projet
  + **Ajouter** le « www » dans l’url du site
  + **Rediriger** l’url [**http://VOTRE\_URL**](http://VOTRE_URL)vers [**https://VOTRE\_URL**](https://VOTRE_URL)
* **Créer** des redirections vers des pages d’erreurs en fonction du code d’erreur
  + *Erreurs : 401, 403, 404, 500*
  + *Un concours de la plus belle « page 404 » sera organisé !*☺

## Partie 10 : Disponibilité du site – Bonus

### Cache

**Mettre en place** un système de cache sur l’un de vos projets (*exemple : memcache*)

### Load Balancing

La répartition de charge (= load balancing en anglais, littéralement équilibrage de charge) est une technique utilisée en informatique pour distribuer un travail entre plusieurs serveurs.

### Les avantages sont nombreux

* Augmentation de la qualité des services
* Amélioration des temps de réponse des services
* Capacité à pallier la défaillance d'une ou de plusieurs machines
* Possibilité d'ajouter des serveurs sans interruption de service

### Mise en place

**Mettre en place** un système de **Load Balancing** sur l’instance de votre serveur. L’outil **haproxy** permet d’atteindre ces objectifs.

*Cf. Annexe 2 – Schéma simplifié d’un système de Load Balancing*

Vous pouvez vous informer ici : <https://linuxfr.org/forums/linux-general/posts/tuto-howto-ubuntu-debian-load-balancing-redirection-vers-plusieurs-vhost-avec-haproxy>

# Annexes

## Annexe 1 - Résultat du fichier info.php

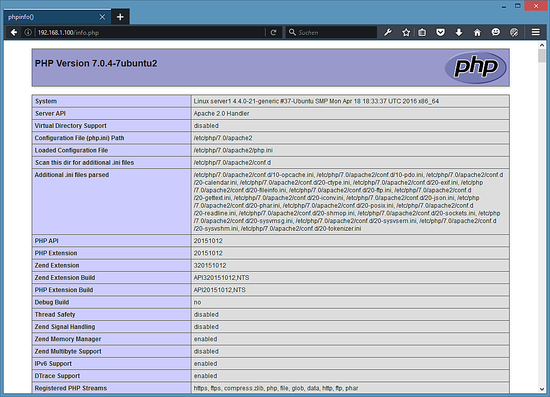


Figure 1 - résultat de la fonction phpinfo()

## Annexe 2 – Schéma simplifié d’un système de Load Balancing

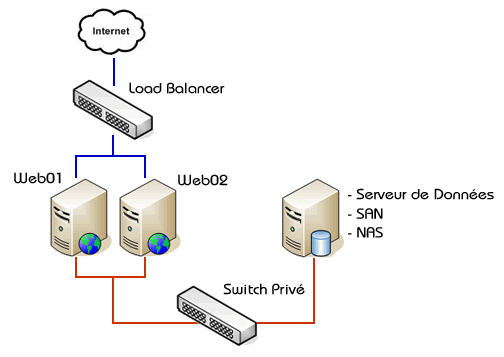


Figure 2 - schéma exemple Load Balancing

# Ressources

* **Apache** : <http://httpd.apache.org/docs/>
* **Apache sur Ubuntu :** <https://doc.ubuntu-fr.org/tutoriel/securiser_apache2_avec_ssl>
* **PHP** : <http://www.php.net/>
* **MySQL** : <https://www.mysql.com/>
* **MySQL [Fr] :** <https://wiki.debian.org/fr/MySql>
* **MariaDB** : <https://mariadb.com/>
* **Ubuntu** : https://www.ubuntu.com/
* **Wordpress :** <https://fr.wordpress.org/>
* **Drupal –** Arborescence : <https://happyculture.coop/blog/drupal-8-structure-des-fichiers>

1. *Utiliser la version libre - Documentation d’OpenSSH sur internet* [↑](#footnote-ref-1)
2. *Depuis sa machine hôte* [↑](#footnote-ref-2)
3. *Le serveur ici est très certainement une machine virtuelle sur votre machine hôte.* [↑](#footnote-ref-3)
4. *Une page web* ***ou*** *un site web – un fichier index.php avec une instruction echo à l’intérieur suffit pour tester* [↑](#footnote-ref-4)