Principe de fonctionnement d'un serveur web :

- Apache
- Php
- Base de données

Comment transformer un PC en serveur web :

- Installer un service web

De quoi ai-je besoin pour rendre mon serveur dynamique :

- Site dynamique
- Site web statique

- Un service web
- Fichiers html/php
- Fichier d'applications php pour mon serveur
- Base de données (avec [phpMyAdmin] ou sans interface graphique [MariaDB, MySQL, Oracle])

Pourquoi choisir Linux plutôt que Windows pour mon serveur web :

- Open source
- Plus léger
- Moins sujet aux attaques
- Le plus utilisé dans le monde (même chez les prestataires)
- Plus sécurisé, fiable

Pourquoi un OS sans interface graphique :

- GUI = Graphical user interface
- Interface graphique = des icônes cliquables

- Performance élevée
- Sécurité
- Accessible à distance grâce au protocole SSH, SCP, sans besoin de validation depuis le poste

10 clés à connaître pour commencer Linux :

- Super utilisateur (su -), nommé « root » = « administrateur » Windows
- Chemin d'accès avec des « / » et non des « \ » :
    - Windows : **C:\tmp\fichier.txt**
    - Linux : **/tmp/fichier.txt**
- La racine du disque est « / ». Si on ne précise pas le chemin d'accès à un fichier c'est que le fichier se trouve dans le répertoire courant. Commande pour connaitre le répertoire courant : **pwd**
- Les répertoires de bases sont :
    - **/etc/** -> les configurations systèmes : adresse IP, nom de la machine
    - **/sbin/** -> Exécutables pour le système : redémarrage et autres
    - **/root/** -> Dossier personnel du super administrateur
    - **/etc/var/www/html/** -> les pages web dans ce dossier seront publiées en lignes

- Par défaut, un user standard (tous sauf root) n'a le droit d'écrire et de supprimer que dans son répertoire perso **/home/toto**. On peut voir les droits avec la commande **chmod** et le propriétaire avec la commande **chown**.
- Le logiciel se télécharge depuis un magasin d'application (comme sur les smartphones). Ils s'appellent « miroirs » ou « dépôts ». On peut voir ces dépôts dans le fichier **/etc/apt/sources.list**
- Installer un logiciel : **apt nomdulogiciel**. Pour mettre à jour : **apt update**
- Pour voir un logiciel ou un service est lancé, il faut utiliser la commande **systemctl status nomdulogiciel**
    - **systemctl status apache2** ; pour voir tous les logiciels lancés actuellement : **ps -ef**
- Pour ouvrir ou modifier un fichier texte ou de paramétrage il faut utiliser un éditeur de texte (comme Word ou NotePad++). Nous allons utiliser « nano ».

Comment accéder à mon serveur web à distance ?

- Activer le protocole SSH dédié – port 22
- Commande : **apt install openssh-server**
- Les utilisateurs pourront utiliser SSH à distance
- On peut ranger le serveur dans une salle climatisée sans écran.

Installation Pratique :

NGINX :

- Serveur web open source depuis 2004
- Performant et stable
- 400M d'utilisateurs (1er ou 2^{e} mondial)
- Fonctions d'équilibrage de charge : répartition efficace du trafic entre plusieurs serveurs.

Pourquoi NGINX plutôt qu'apache2 :

- Performance (connexions simultanées)
- Utilisation efficace des ressources (consomme moins pour plus de performance)

D'où vient asynchrone et non bloquant :

- Modèle événementielle, traitant plusieurs connexions sans bloquer le processus principal
- Utilisation de workers pour la gestion des connexions

5 étapes d'installation (statique) :

- Téléchargement et Installation :
    - sudo apt update
    - sudo apt install nginx
- Configuration de base dans **/etc/nginx/nginx.conf**
- Démarrage du service et activation au démarrage du serveur :
    - sudo systemctl start nginx
    - sudo systemctl enable nginx
- Déposer son site web dans : **/var/www/html**
- Tester l'accès dans son navigateur en tapant l'IP du serveur web

Installation dynamique :

- Installation du moteur **PHP**

- Installation d'une base de données : **MARIADB MySQL**
- Déposer un site web PHP
- Avoir un nom de domaine DNS : www.site.com

Protection :

- Bloquer les IP attaquantes : **firewall**
- Gérer les droits d'accès aux fichiers : **CHMOD**, **CHOWN**
- Gérer les comptes utilisateurs : **root**, **sudoers**
- Crypter les connexions : **HTTPS**, **SSL/TLS**
- Le mettre dans une salle sécurisée
- Faire des sauvegardes
- Prévoir un serveur redondant : **Haute Disponibilité**
- Prévoir un ondulateur