HÉBERGER SON PROPRE SITE AVEC APACHE

Résumé

- « Vous avez la main sur votre serveur, vos configurations et vos fichiers. »

TRXTRZ.INFO

Table des matières

Héberger son Propre Site avec Apache

Héberger son propre site web avec Apache peut être une excellente idée dans plusieurs contextes. Voici les principaux avantages :

Contrôle total

- Vous avez la main sur votre serveur, vos configurations et vos fichiers.
- Pas de dépendance directe à un fournisseur tiers d'hébergement.
- Vous pouvez personnaliser les réglages (sécurité, performances, modules, logs) exactement comme vous le souhaitez.

Gratuit et open source

- Apache est open source et gratuit.
- Il est largement utilisé et très bien documenté.
- Vous bénéficiez d'une communauté mondiale active et d'innombrables tutoriels.

Apprentissage et compétences

- En installant et en configurant Apache, vous développez des compétences en :
  - Administration système,
  - Gestion de serveurs Linux,
  - Sécurité et réseaux,
  - Hébergement web.
- C'est une excellente expérience pratique pour les étudiants, développeurs ou administrateurs systèmes.

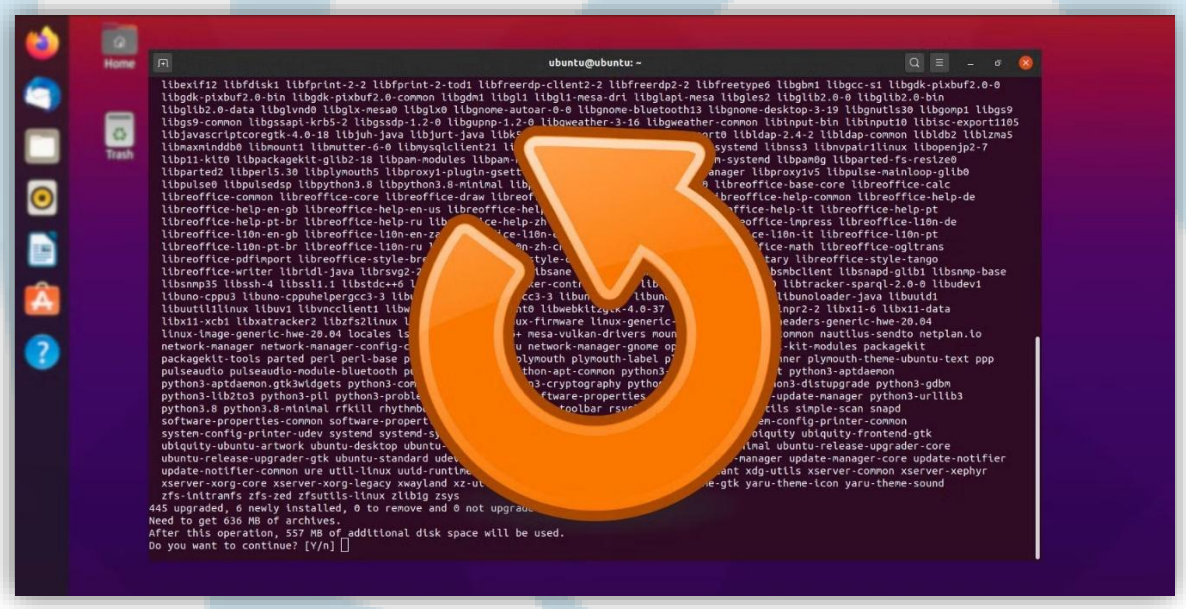
Installation et configuration d'Apache2 sur Ubuntu

Voici les instructions pour installer et configurer Apache2 sur Ubuntu

Mettre à jour le système

Assurez-vous que votre système est à jour :

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

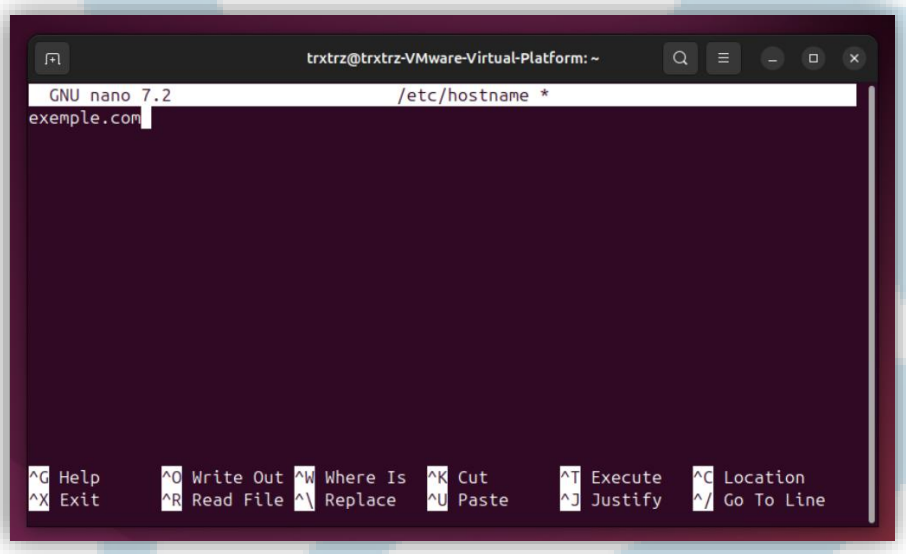


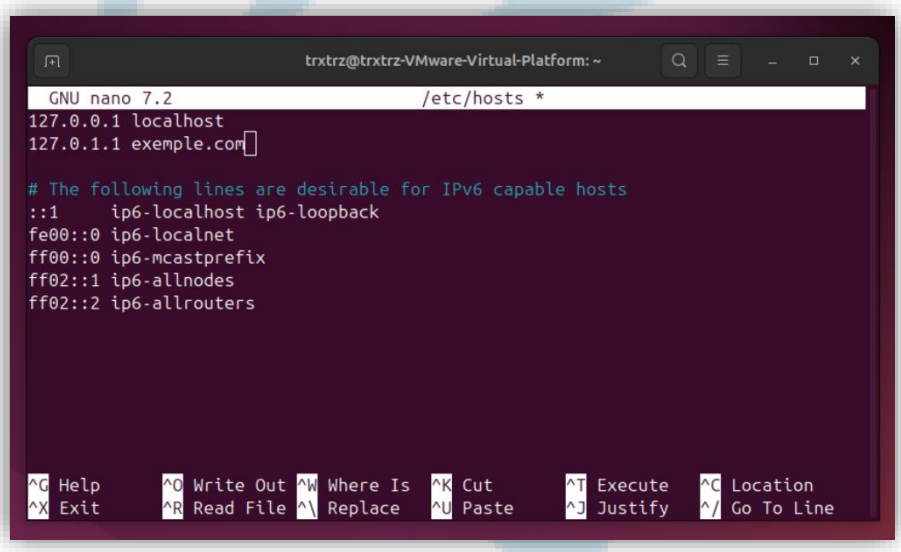
Note: Faire sudo apt update && apt upgrade après l'installation d'Ubuntu permet d'avoir un système plus sécurisé, stable, et à jour, prêt à accueillir vos logiciels et services (comme Apache, MySQL, PHP, etc.).

Changer le hostname de votre machine:

Il est important de changer le hostname de notre machine par le nom de domaine :

```
sudo nano /etc/hostname

sudo nano /etc/hosts

sudo reboot
```





Installer Apache2 :

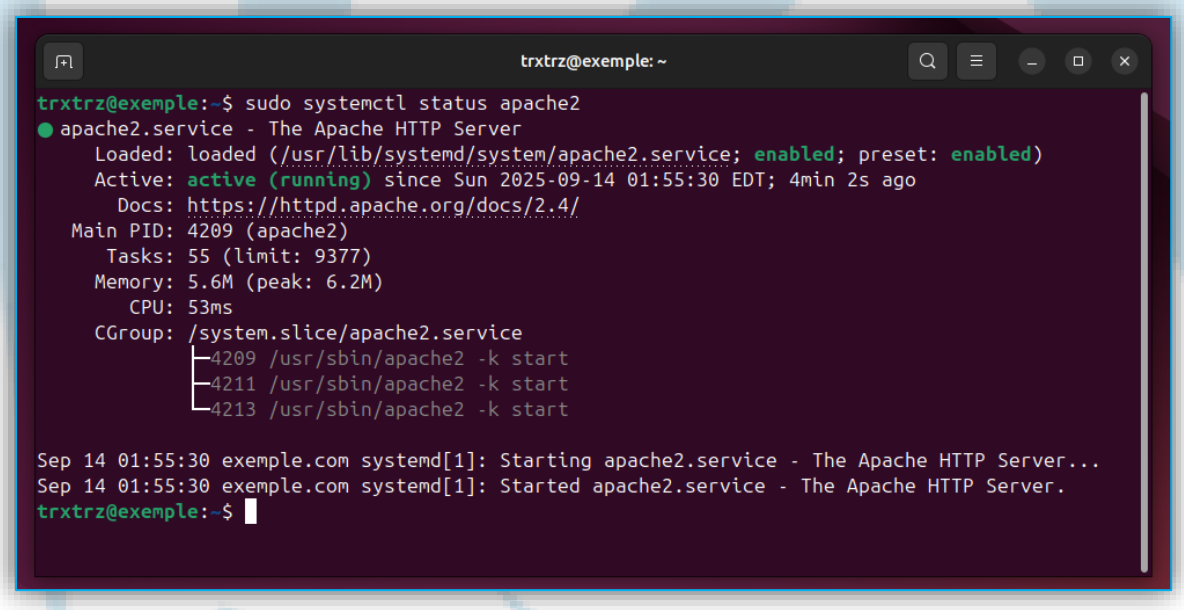
Installez les paquets Apache2, apache2-doc et apache2-utils:

```
sudo apt install apache2 apache2-doc apache2-utils -y
```

Vérifier l'installation

Vérifiez si Apache fonctionne :

```
sudo systemctl status apache2
```



Vous devriez voir active (running). Si ce n'est pas le cas, lancez-le :

```
sudo systemctl start apache2
```

Et activez-le au démarrage :

```
sudo systemctl enable apache2
```

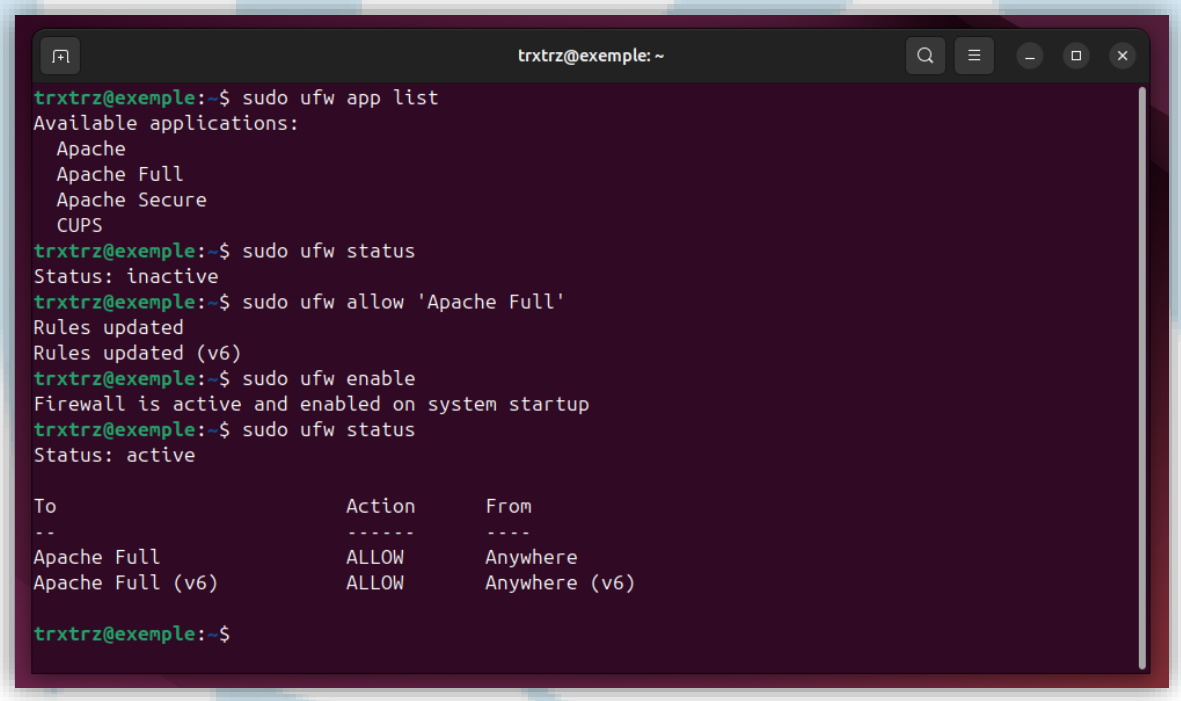
Autoriser Apache dans le pare-feu (si UFW est actif)

Vérifiez l'état de UFW :

```
sudo ufw status

sudo ufw app list
```

Autorisez le trafic web :

```
sudo ufw allow 'Apache Full'

sudo ufw enable

sudo ufw status
```



Tester dans un navigateur

Ouvrez un navigateur et tapez :

http://votre_ip_du_serveur (Ex. : http://192.168.2.239/)

Ou, si vous êtes sur la machine roulant Apache (L'hôte):

http://localhost

Gérer le service Apache au besoin...

Redémarrer Apache :

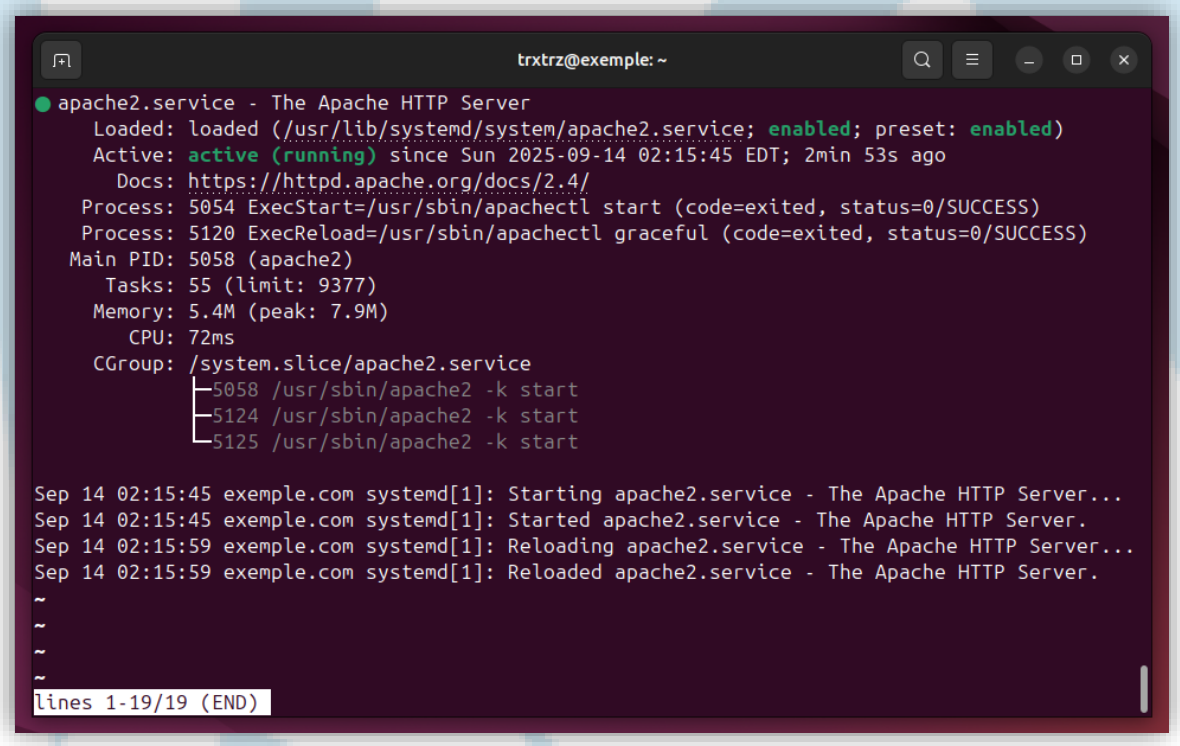
```
sudo systemctl restart apache2
```

Recharger la configuration (sans couper le service) :

```
sudo systemctl reload apache2
```

Vérifier l'état Apache2 :

```
sudo systemctl status apache2
```



Hey! On va faire les étapes de cette page en groupe... Fais signe si tu es rendu ICI!

Créer un hôte virtuel (site web)

Le répertoire par défaut est /var/www/html/

Pour héberger plusieurs sites, créez un fichier de configuration :

```
sudo nano /etc/apache2/sites-available/exemple.com.conf
```

Exemple de configuration :

```
<VirtualHost *:80>
      ServerName exemple.com
      ServerAlias www.exemple.com
      DocumentRoot /var/www/html/exemple.com/public_html/
      ErrorLog ${APACHE_LOG_DIR}/exemple.com_error.log
      CustomLog ${APACHE_LOG_DIR}/exemple.com_access.log combined
</VirtualHost>
```

Activez le site et rechargez Apache :

```
sudo a2dissite 000-default.conf
sudo a2ensite exemple.com.conf
sudo systemctl reload apache2
```

Utiliser Filezilla pour transférer des fichiers vers votre serveur Web

Accès via SFTP

Qu'est-ce que l'accès SFTP ?

SFTP signifie SSH File Transfer Protocol (ou Secure File Transfer Protocol).

C'est une méthode pour transférer et gérer des fichiers à distance, mais en utilisant la connexion sécurisée SSH.

Installer OpenSSH sur Ubuntu (si ce n'est pas déjà fait)

```
sudo apt update && sudo apt install openssh-server -y
```

Vérifier que le service SSH fonctionne :

```
sudo systemctl status ssh
```

Si ce n'est pas actif :

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

Donner l'accès au dossier /var/www/html/exemple.com/public_html/

Il faut ajouter votre utilisateur au groupe www-data :

```
sudo usermod -aG www-data trxtrz

sudo chown -R trxtrz:www-data /var/www/html/exemple.com/public_html/
```

Installation de FileZilla si nécessaire…

Rendez-vous à https://ninite.com/

Connection dans FileZilla

Simplement mettre l'IP du serveur, choisir le port 22, et FileZilla comprend que c'est une connexion sécurisée.

Hôte : 192.168.1.100 (par exemple)

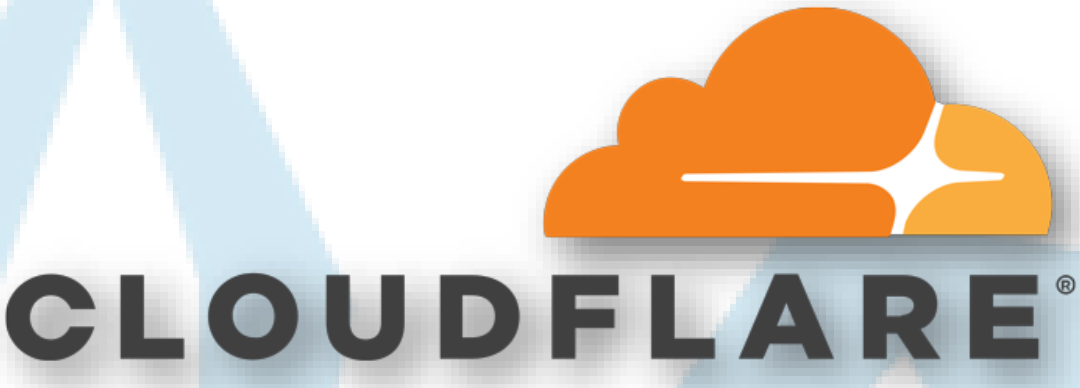
Utilisateur : votre login Ubuntu

Mot de passe : votre mot de passe Ubuntu

Port : 22

Ensuite vous pouvez naviguer dans votre serveur comme si c'était un dossier local.

Configurer un Cloudflare Tunnel sur votre serveur Ubuntu



Un tunnel géré (Cloudflare Tunnel, Tailscale Funnel, ngrok) : dans ces modèles, l'URL/domaine pointe vers le fournisseur du tunnel (ou un CNAME proxied), pas directement vers votre IP domestique/public, donc pas besoin de suivre les changements d'IP chez vous.