



# 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 .....	2
Contrôle total.....	2
Gratuit et open source .....	2
Apprentissage et compétences .....	2
Installation et configuration d'Apache2 sur Ubuntu .....	3
Voici les instructions pour installer et configurer Apache2 sur Ubuntu.....	3
Mettre à jour le système.....	3
Changer le hostname de votre machine:.....	4
Installer Apache2 : .....	5
Vérifier l'installation .....	5
Autoriser Apache dans le pare-feu (si UFW est actif) .....	6
Tester dans un navigateur .....	6
Gérer le service Apache au besoin.....	7
Créer un hôte virtuel (site web).....	8
Utiliser Filezilla pour transférer des fichiers vers votre serveur Web .....	9
Accès via SFTP .....	9
Installer OpenSSH sur Ubuntu (si ce n'est pas déjà fait) .....	9
Donner l'accès au dossier /var/www/html/exemple.com/public_html/ .....	10
Installation de FileZilla si nécessaire.....	10
Connection dans FileZilla .....	10
Configurer un Cloudflare Tunnel sur votre serveur Ubuntu .....	11



## 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.

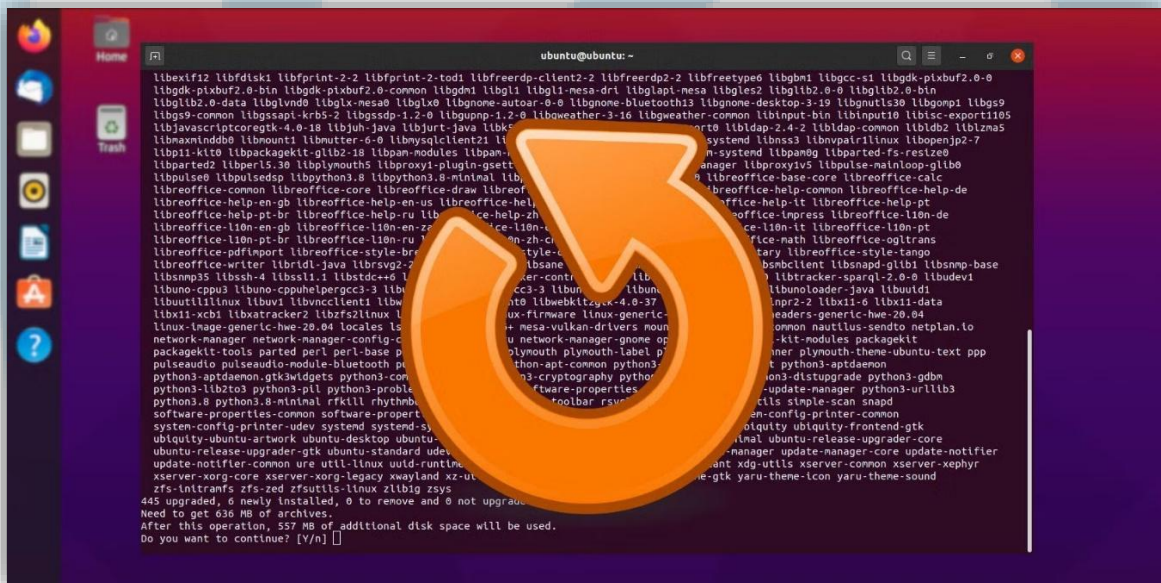
# Installation et configuration d'Apache2 sur Ubuntu

## 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
```



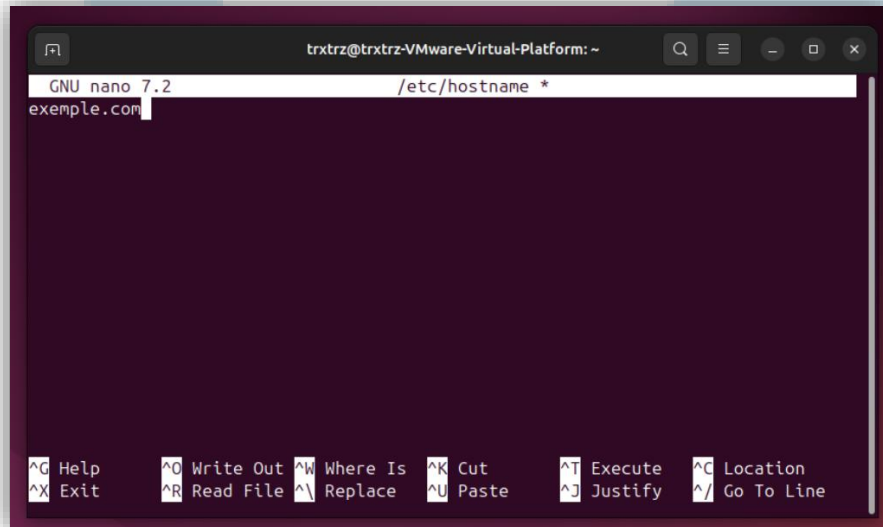
**Note:** Faire `sudo apt update && apt upgrade` après l'installation d'**Ubuntu** permet d'avoir un système plus sécurisé, stable, et à jour, prêt à accueillir vos logiciels et services (comme Apache, MySQL, PHP, etc.).



## Changer le hostname de votre machine:

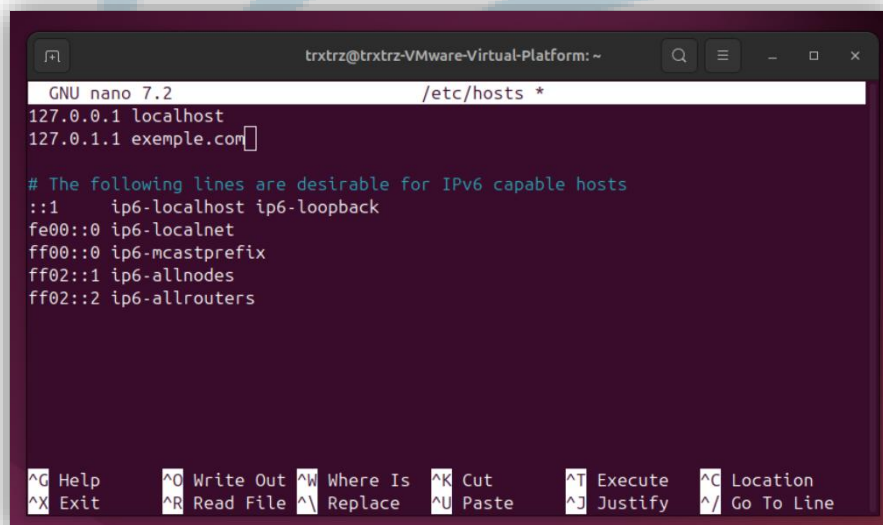
Il est important de changer le **hostname** de notre machine par le nom de domaine :

```
sudo nano /etc/hostname  
sudo nano /etc/hosts  
sudo reboot
```



```
txtrtz@txtrtz-VMware-Virtual-Platform: ~  
GNU nano 7.2 /etc/hostname *  
exemple.com
```

Help Write Out Where Is Cut Execute Location  
Exit Read File Replace Paste Justify Go To Line



```
txtrtz@txtrtz-VMware-Virtual-Platform: ~  
GNU nano 7.2 /etc/hosts *  
127.0.0.1 localhost  
127.0.1.1 exemple.com  
  
# The following lines are desirable for IPv6 capable hosts  
::1 ip6-localhost ip6-loopback  
fe00::0 ip6-localnet  
ff00::0 ip6-mcastprefix  
ff02::1 ip6-allnodes  
ff02::2 ip6-allrouters
```

Help Write Out Where Is Cut Execute Location  
Exit Read File Replace Paste Justify Go To Line



## 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
```

```
trxtrrz@exemple: ~  
trxtrrz@exemple:~$ sudo systemctl status apache2  
● apache2.service - The Apache HTTP Server  
   Loaded: loaded (/usr/lib/systemd/system/apache2.service; enabled; preset: enabled)  
   Active: active (running) since Sun 2025-09-14 01:55:30 EDT; 4min 2s ago  
     Docs: https://httpd.apache.org/docs/2.4/  
   Main PID: 4209 (apache2)  
     Tasks: 55 (limit: 9377)  
    Memory: 5.6M (peak: 6.2M)  
       CPU: 53ms  
    CGroup: /system.slice/apache2.service  
            └─4209 /usr/sbin/apache2 -k start  
              └─4211 /usr/sbin/apache2 -k start  
                └─4213 /usr/sbin/apache2 -k start  
  
Sep 14 01:55:30 exemple.com systemd[1]: Starting apache2.service - The Apache HTTP Server...  
Sep 14 01:55:30 exemple.com systemd[1]: Started apache2.service - The Apache HTTP Server.  
trxtrrz@exemple:~$
```

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
```

```
trxtrz@example:~  
trxtrz@example:~$ sudo ufw app list  
Available applications:  
  Apache  
  Apache Full  
  Apache Secure  
  CUPS  
trxtrz@example:~$ sudo ufw status  
Status: inactive  
trxtrz@example:~$ sudo ufw allow 'Apache Full'  
Rules updated  
Rules updated (v6)  
trxtrz@example:~$ sudo ufw enable  
Firewall is active and enabled on system startup  
trxtrz@example:~$ sudo ufw status  
Status: active  
  
To Action From  
--  
Apache Full ALLOW Anywhere  
Apache Full (v6) ALLOW Anywhere (v6)  
  
trxtrz@example:~$
```

## Tester dans un navigateur

Ouvrez un navigateur et tapez :

[http://votre\\_ip\\_du\\_serveur](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
```

```
txtrtz@example: ~
● apache2.service - The Apache HTTP Server
   Loaded: loaded (/usr/lib/systemd/system/apache2.service; enabled; preset: enabled)
   Active: active (running) since Sun 2025-09-14 02:15:45 EDT; 2min 53s ago
     Docs: https://httpd.apache.org/docs/2.4/
   Process: 5054 ExecStart=/usr/sbin/apachectl start (code=exited, status=0/SUCCESS)
   Process: 5120 ExecReload=/usr/sbin/apachectl graceful (code=exited, status=0/SUCCESS)
  Main PID: 5058 (apache2)
    Tasks: 55 (limit: 9377)
   Memory: 5.4M (peak: 7.9M)
      CPU: 72ms
   CGroup: /system.slice/apache2.service
           └─5058 /usr/sbin/apache2 -k start
             └─5124 /usr/sbin/apache2 -k start
               └─5125 /usr/sbin/apache2 -k start

Sep 14 02:15:45 exemple.com systemd[1]: Starting apache2.service - The Apache HTTP Server...
Sep 14 02:15:45 exemple.com systemd[1]: Started apache2.service - The Apache HTTP Server.
Sep 14 02:15:59 exemple.com systemd[1]: Reloading apache2.service - The Apache HTTP Server...
Sep 14 02:15:59 exemple.com systemd[1]: Reloaded apache2.service - The Apache HTTP Server.
~
~
~
lines 1-19/19 (END)
```





**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



# CLOUDFLARE®

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.