Apache

Sur Ubuntu

Apache est souvent combiné à des systèmes Linux. Ubuntu est tout particulièrement indiqué pour l'installation du serveur Web, en raison de sa communauté passionnée et de son importante documentation. Découvrez avec nous l'ensemble des étapes d'installation et de configuration d'Apache sous Ubuntu.

Apache sous Ubuntu : prérequis

Apache compte parmi les serveurs Web les plus anciens et les plus stables. Il est très populaire, notamment grâce à son évolutivité et à la facilité de sa configuration. Il n'existe aucun prérequis spécifique en ce qui concerne le processeur pour l'installation d'Apache sous Ubuntu 22.04. La plupart des processeurs modernes sont généralement suffisants pour faire fonctionner Apache sous Ubuntu. Les éléments fondamentaux demandés sont **la mémoire vive disponible et la capacité de stockage sur le disque dur**.

Apache consomme peu de ressources et peut être configuré sur différents types de systèmes : ordinateurs de bureau, ordinateurs portables, serveurs, machines virtuelles, etc. Afin d'héberger un site Web ou une application efficace, davantage de ressources seront peut-être requises pour profiter de performances optimales. Attention : n'oubliez pas que **l'utilisation de modules est susceptible de modifier la configuration système requise**. Si vous souhaitez par exemple intégrer un module destiné à améliorer les performances d'un serveur Web Apache, une plus grande mémoire sera nécessaire pour la mise en cache et d'autres optimisations.

Pour configurer un serveur Web Apache, vous devez respecter les prérequis suivants :

- **Mémoire vive (RAM)** : 4 Go
- **Système d'exploitation** : Ubuntu (utilisateur avec droits d'accès sudo)
- **Espace en disque dur** : 5 Go
- **Pare-feu** : suffisant pour réguler le trafic HTTP et bloquer les ports qui ne sont pas nécessaires
- **Connexion Internet** : suffisante pour télécharger des paquets

Conseil

Avec l'offre d'hébergement Linux proposée par IONOS, profitez d'un trafic illimité, des meilleures performances en matière de rapidité et des normes de sécurité les plus élevées. N'attendez plus pour améliorer votre visibilité en ligne avec les serveurs Linux proposés par IONOS !

Instructions étape par étape : installation d'Apache sous Ubuntu

Ubuntu 22.04 fait appel à l'**outil de gestion des paquets APT** pour installer Apache. Commencez par actualiser l'index des paquets sur votre système Ubuntu, de manière à vérifier que toutes les dépendances nécessaires sont bien à jour.

Si l'installation n'est pas locale, utilisez le protocole SSH pour vous connecter à votre serveur Ubuntu.

Étape 1 : mettre à jour la liste des paquets

Commencez par appeler le terminal, puis procédez à une mise à jour.

```
$ sudo apt update
```

Étape 2 : installer le paquet Apache

Installez ensuite le paquet Apache ainsi que toutes ses dépendances en utilisant la commande APT « **install** ».

```
$ sudo apt install apache2
```

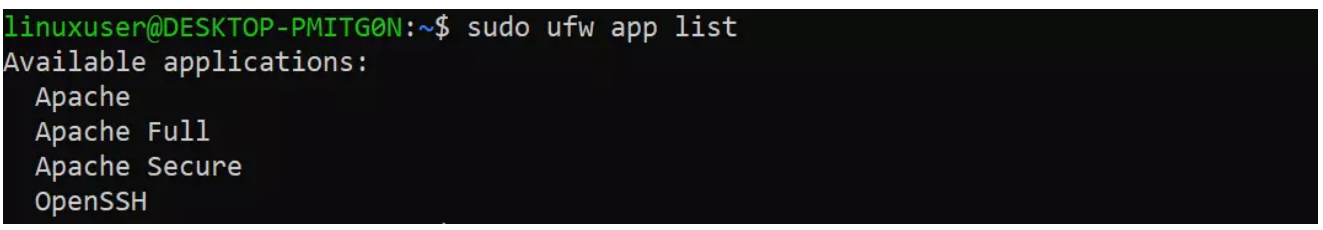
Étape 3 : modifier les paramètres du pare-feu

Pour configurer Apache, vous devez activer le programme Uncomplicated Firewall (UFW) dans Ubuntu. Une fois Apache installé sous Ubuntu, il configure des **profils d'application** dans UFW, ces derniers permettant de réguler le trafic de données vers les ports Web.

Utilisez la commande suivante pour afficher la liste des profils d'application :

```
$ sudo ufw app list
```

Le résultat affiche trois profils pour Apache :



Terminal : liste des profils d'application pour Apache

- **Apache** : ouvre le port TCP 80 pour HTTP (connexion non chiffrée)

- **Apache Full** : ouvre le port TCP 80 (HTTP, connexion non chiffrée) et 443 (HTTPS, connexion chiffrée avec TLS/SSL)
- **Apache Secure** : ouvre seulement le port HTTPS 443 pour une connexion chiffrée

Comme aucun SSL n'a encore été mis en place, nous ne pouvons ouvrir que le port 80.

```
$ sudo ufw allow 'Apache'
```

La commande « status » permet de vérifier que le bon réglage a bien été effectué.

```
$ sudo ufw status
```

Étape 4 : tester l'état d'Apache

Utilisez le gestionnaire de système « **systemd** » pour vérifier si le service Apache est bien actif.

```
$ sudo systemctl status apache2
```

Étape 5 : appeler la page de démarrage par défaut d'Apache

Renseignez votre adresse IP dans la barre d'adresse du navigateur pour accéder à la page par défaut d'Apache. Vous ne connaissez pas votre adresse IP ? Il vous suffit de taper « hostname » (nom d'hôte) pour l'afficher.

```
$ hostname -I
```

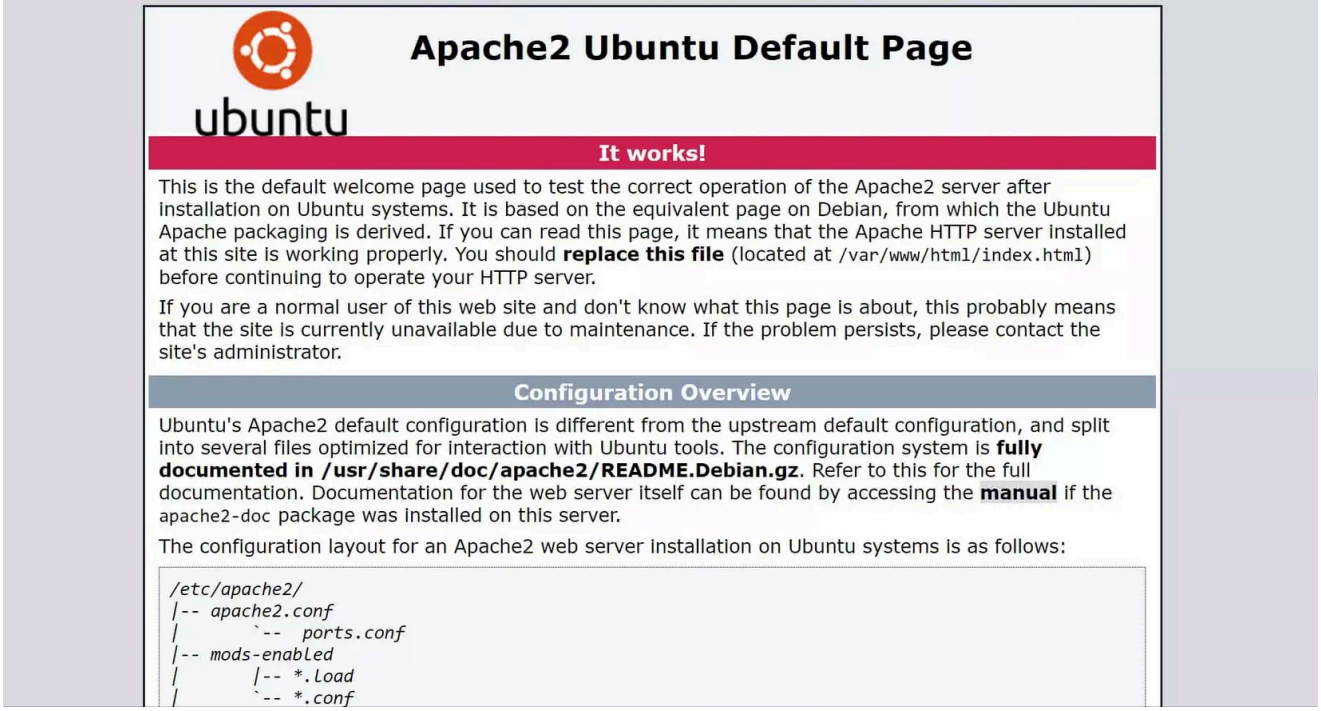
Vous avez aussi la possibilité d'utiliser l'outil « icanhazip ».

```
$ curl -4 icanhazip.com
```

Appelez maintenant la page standard d'Apache dans votre navigateur, puis remplacez « **server_ip** » par votre propre adresse IP.

```
http://server_ip
```

Vous trouverez ci-dessous un extrait de la page sous Ubuntu :



Navigateur Web : page d'accueil d'Apache sous Ubuntu

Étape 6 : gérer le démon Apache

Vous pouvez utiliser « systemctl »pour gérer le démon (ou service) du serveur Web Apache.

Démarrez le serveur Web Apache comme suit :

```
$ sudo systemctl start apache2
```

Arrêtez le serveur Web Apache comme suit :

```
$ sudo systemctl stop apache2
```

Arrêtez et redémarrez le serveur Web Apache comme suit :

```
$ sudo systemctl restart apache2
```

Redémarrez Apache et chargez à nouveau la configuration comme suit :

```
$ sudo systemctl reload apache2
```

Si vous installez Apache sous Ubuntu, le serveur Web se lance alors automatiquement au démarrage, juste après la configuration. Vous pouvez aussi choisir de désactiver cette fonction :

```
$ sudo systemctl disable apache2
```

Si vous souhaitez réactiver le lancement automatique d'Apache au démarrage, procédez comme suit :

```
$ sudo systemctl enable apache2
```

Étape 7 : utiliser des hôtes virtuels

Apache héberge, par défaut, les documents dans */var/www/html*. Si vous voulez utiliser plusieurs domaines sur un serveur, vous pouvez alors mettre en place des hôtes virtuels. Pour ce faire, créez une structure de répertoire pour un domaine propre au sein de */var/www/*.

```
$ sudo mkdir /var/www/your_domain
```

Remplacez ensuite « **your_domain** » par le nom de votre domaine.

Attribuez la propriété du répertoire à la variable d'environnement « *$USER* » :

```
$ sudo chown -R $USER:$USER /var/www/your_domain
```

Il est également possible d'attribuer explicitement les droits de lecture, d'écriture et d'exécution en mode octal :

```
$ sudo chmod -R 755 /var/www/your_domain
```

Étape 8 : créer une page de test

Créez un *index.html* en tant que page d'accueil pour votre domaine. Vous pouvez pour cela utiliser l'éditeur de texte « **nano** », par exemple.

```
$ sudo nano /var/www/your_domain/index.html
```

Choisissez ensuite une formulation avant de l'insérer dans le fichier HTML :

```
<html>
    <head>
        <title>Welcome to your_domain!</title>
    </head>
    <body>
        <h1>Here you can see that your_domain virtual host is successfully
working!</h1>
    </body>
</html>
```

Étape 9 : créer un fichier de configuration pour l'hôte virtuel

Apache doit être configuré en fonction du domaine pour que la page d'exemple soit affichée. Créez un **fichier de configuration propre à votre domaine**, sans modifier le fichier de configuration par défaut d'Apache.

```
$ /etc/apache2/sites-available/your_domain.conf
```

Insérez le bloc suivant, puis remplacez « **your_domain** » par le nom de votre domaine. Dans « ServerAdmin », il est également nécessaire d'indiquer une adresse email pour l'administrateur :

```
<VirtualHost *:80>
    ServerAdmin webmaster@localhost
    ServerName your_domain
    ServerAlias www.your_domain
    DocumentRoot /var/www/your_domain
    ErrorLog ${APACHE_LOG_DIR}/error.log
    CustomLog ${APACHE_LOG_DIR}/access.log combined
</VirtualHost>
```

Activez le fichier de configuration en utilisant « a2ensite » :

```
$ sudo a2ensite your_domain.conf
```

Désactivez ensuite l'ancienne page par défaut :

```
$ sudo a2dissite 000-default.conf
```

Testez la configuration afin de détecter toute erreur :

```
$ sudo apache2ctl configtest
```

En l'absence d'erreur, vous pouvez alors redémarrer Apache :

```
$ sudo systemctl restart apache2
```

Accédez à votre page d'accueil :

```
http://your_domain
```

Maintenant, il est normalement possible de voir votre page d'exemple :

Navigateur Web : exemple de page pour un hôte virtuel

Étape 10 : fichiers et répertoires Apache importants

Si vous voulez utiliser efficacement le serveur Web Apache, il peut s'avérer bénéfique de connaître quelques fichiers et répertoires parmi les plus fréquemment utilisés :

- **/var/www/html** : par défaut, Apache met les documents à disposition dans ce répertoire. Vous pouvez modifier ce réglage dans les fichiers de configuration.
- **/etc/apache2** : tous les fichiers de configuration d'Apache sont conservés dans ce répertoire.
- **/etc/apache2/apache2.conf** : il s'agit du fichier de configuration principal, qui permet de modifier la configuration dans son ensemble.
- **/etc/apache2/ports.conf** : les ports ouverts sont listés dans ce fichier. En règle générale, il s'agit du port 80 et/ou du port 443.
- **/etc/apache2/sites-available/** : les hôtes virtuels configurés sont stockés dans ce dossier. Pour pouvoir fonctionner, les fichiers de configuration qui s'y trouvent doivent être associés au répertoire « site-enabled ».
- **/etc/apache2/conf-available/, /etc/apache2/conf-enabled/** : des fichiers de configuration supplémentaires n'appartenant pas à des hôtes virtuels sont stockés dans ces répertoires. Pour activer la configuration, utilisez « a2enconf » ; pour la désactiver, utilisez « a2disconf ».
- **/etc/apache2/mods-available/**, **/etc/apache2/mods-enabled/** : tous les modules qui sont disponibles et activés se trouvent dans ces répertoires. Utilisez « a2enmod » pour activer un module, et « a2dismod » pour le désactiver.
- **/var/log/apache2/access.log** : toutes les demandes adressées au serveur Web sont enregistrées dans ce fichier journal.
- **/var/log/apache2/error.log** : l'ensemble des messages d'erreur est consigné dans ce fichier. Les informations « LogLevel » donnent des indications sur le degré de gravité des événements.

Conseil

Vous n'avez pas le temps d'administrer et d'entretenir un serveur ? Nous vous recommandons de louer un serveur Managed chez IONOS. Nul besoin de connaître Linux sur le bout des doigts pour gérer un serveur Managed. IONOS s'occupe à votre place de toutes les tâches importantes liées à la gestion du serveur, notamment des sauvegardes automatiques et des contrôles de sécurité.