

# Hébergement apache/nginx + fpm

dimanche 22 décembre 2024, par [manu](#)

**Un nouveau type d'hébergement : serveur web nginx, php en "fast-cgi php manager", pour aller plus vite ! Attention, article très technique, si vous ne comprenez pas tout ce n'est pas grave.... mais lisez tout de même la dernière partie !**

## Apache ou nginx ?

[Apache \[1\]](#) est un logiciel serveur http qui existe depuis les années 1995... c'est-à-dire presque depuis les débuts du web. Il est très versatile et puissant, et surtout très répandu. Mais depuis ces années, d'autres logiciels sont apparus, entre autres [nginx \[2\]](#) Tous deux sont bien entendu des logiciels open source.

### Un peu de vocabulaire..

Voici quelques précisions sur les noms de logiciels :

**apache** = Le logiciel serveur http le plus utilisé au monde, très puissant et versatile, grâce à ses nombreux modules (extensions). Il comporte un module lui permettant d'interpréter du code [php](#) (voir ci-dessous). Mais du coup il consomme beaucoup de mémoire vive et comme il fait plein de choses, il ne va pas toujours aussi vite qu'on le souhaiterait. *Sauf que*, pour compliquer encore un peu plus les choses, on peut aussi utiliser apache avec [fpm](#) (voir ci-dessous), et dans ce cas on a (presque) l'argent du beurre en plus du beurre : moins de consommation de mémoire qu'avec le module php et plus de rapidité, mais possibilité de lire les fichiers [.htaccess](#) (voir ci-dessous).

**nginx** = Un logiciel concurrent d'apache qui ne fait que lire les requêtes d'un client web (le navigateur d'un internaute par exemple) et lui renvoyer la réponse, gérer les restrictions d'accès, etc. Il ne fait que ça, mais il le fait très bien : très vite et en prenant pas trop de mémoire vive.

**php** = Un langage de programmation, très pratique pour développer des applications web. Beaucoup de CMS ([spip](#), [wordpress](#), [yeswiki](#), [joomla](#), [drupal](#)) ou d'applications web ([galette](#), [dolibarr](#), [paheko](#)) sont écrits dans ce langage.

**fpm** = Un logiciel qui exécute le code écrit en [php](#) et qui communique avec le serveur web ([nginx](#) ou [apache](#)) : lorsque le serveur web voit un fichier [.php](#) il passe la patate chaude à [fpm](#). Celui-ci interprète alors le code [php](#) et renvoie le résultat à [nginx](#), qui le transmet à son client.

**mod\_php** = Un module php intégré à apache, c'est la manière d'utiliser php au Pic jusqu'en 2024. Cela fonctionne parfaitement, mais avec moins de performances, et en utilisant plus de mémoire sur le serveur, que [fpm](#).

## Pourquoi nginx ?

[nginx](#) présente quelques intérêts, en particulier il est très léger et très rapide. Sa conception lui permet de répondre à plusieurs requêtes en même temps. Apache aussi bien sûr, mais de manière moins efficace. Par ailleurs [nginx](#) n'a pas besoin de démarrer en "root", il est donc simple de lancer

une instance de `nginx` par utilisateur, ce qui est plus sûr que d'avoir une instance de serveur commune. A l'inverse, `apache` démarre en tant que `root`, puis change d'utilisateur lorsqu'il reçoit la requête (à condition qu'on utilise un module spécialisé appelé `itk`) : Cela fonctionne très bien mais c'est plus lourd, et là encore, plus lent. Enfin, `nginx` n'a pas de module `php`, à l'inverse d'`Apache`. Il faudra donc utiliser une installation de `php` extérieure à `nginx`, elle implémentera le protocole `fpm` (Fast-cgi php manager) [3]

## Fichiers `.htaccess`

Tous les concepteurs de sites web connaissent les fichiers `.htaccess`, qui peuvent être déposés dans chaque répertoire du site web. Le fichier `.htaccess` contient des directives permettant de modifier la configuration du serveur pour ce répertoire et pour ceux qui sont en-dessous de lui. Par exemple protéger certains fichiers contenant des secrets (mots de passe ou autre). C'est très pratique car cela permet au concepteur du site d'agir sur la configuration du serveur : après tout c'est lui ou elle qui connaît les fichiers à protéger ! Mais, revers de la médaille, le serveur perd du temps en recherchant et en lisant tous les fichiers `.htaccess` ! C'est pourquoi les concepteurs d'`nginx` ont délibérément refusé d'implémenter cette fonctionnalité, qui nuit aux performances. Mais cela complique un peu le travail des concepteurs du site, d'autant plus que beaucoup de CMS utilisent cette fonctionnalité, partant du principe qu'`Apache`, de part sa position hégémonique, sera toujours le serveur web utilisé...

## Au PIC : Apache ou nginx ? module php ou fpm ?

Depuis 2025, nous avons trois possibilités :

1. `apache+mod_php`
2. `apache+fpm`
3. `nginx+fpm`

Ce qui est utilisé :

1. La plupart des sites web utilisent toujours en 2025 `apache`, avec :
  - `mod_php`, hébergement traditionnel.
  - ou (le plus souvent) `fpm` pour exécuter le code `php`
2. Nous avons également la possibilité de mettre en place un hébergement `nginx-fpm`, afin d'accélérer le site web tout en chargeant moins notre serveur.

### Avec `nginx` : les répertoires `/priv/htaccess` :

`nginx` est le serveur web le plus rapide actuellement, il est donc très tentant de l'utiliser. Nous avons mis en place le serveur `nginx` de sorte qu'une partie de sa configuration soit accessible par le concepteur de site web... donc par les adhérents du PIC :

Tout fichier situé dans le répertoire `/priv/htaccess` et dont le nom a l'extension `.conf` sera lu au démarrage de `nginx` et influera sur sa configuration. Mais attention, **les directives de configuration de `nginx` ne sont pas les mêmes que celles d'`apache`** ! Donc si votre CMS ne prévoit pas de directives `nginx`, vous devrez les écrire à partir des directives `apache` fournies par le CMS !

## **Redémarrage requis !**

**Attention si vous modifiez la configuration** dans un fichier de `/priv/htaccess` **le serveur nginx devra être redémarré** pour que ce soit pris en compte ! Il faudra nous le demander car vous ne pouvez pas le faire vous-même.

### **Avec SPIP et WordPress :**

Pour vous aider nous avons déjà écrit les directives de configuration qui vont bien pour les CMS **SPiP** et **WordPress**. Attention tout de même, lorsque vous installez des extensions elles peuvent venir avec des fichiers `.htaccess` spécifiques, peut-être faudra-t-il alors modifier quelque chose dans `/priv/htaccess`

### **Avec les autres CMS :**

Nous vous laissons le soin de faire le même travail avec vos CMS préférés, merci de partager le cas échéant votre configuration !

### **Avec apache-fpm : Pas de problème .htaccess !**

S'il y a de nombreux et complexes fichiers `.htaccess`, nous conservons l'usage d'apache, mais avec fpm pour avoir de meilleurs performances. Tout fonctionne de la même manière qu'avec `apache+mod_php` : c'est bien sûr très pratique, mais il faut garder en tête que les performances seront un peu moins bonnes qu'avec `nginx` !

---

## **Notes**

[1] <https://httpd.apache.org>

[2] <https://www.f5.com/go/product/welcome-to-nginx>

[3] <https://php-fpm.org/>