

Architecture du projet Vagrant (3 machines)

Présentation générale

Le projet utilise **Vagrant** pour créer et gérer trois machines virtuelles (VM) sous Debian Bookworm. Vagrant est un outil qui simplifie la création d'environnements virtuels reproductibles ¹. Chaque VM se comporte comme un ordinateur isolé (on parle de *machine virtuelle*, un « PC simulé » par logiciel) ². Les trois VM sont nommées **nginx1**, **nginx2** et **storage**. Deux d'entre elles jouent le rôle de serveurs web, le troisième est un serveur de fichiers. Toutes ces machines sont configurées automatiquement au démarrage par **Ansible** (via le provisionnement `ansible_local` de Vagrant), c'est-à-dire qu'un « playbook » installe et configure Nginx, PHP, NFS, etc., sans intervention manuelle ³ ⁴.

Rôle de chaque machine

- **nginx1 (serveur Web)** : C'est un serveur web Debian avec **Nginx** et PHP-FPM. Nginx est le logiciel serveur qui envoie les pages web ⁵. PHP-FPM (« FastCGI Process Manager ») gère l'exécution du code PHP pour Nginx ⁶. Ce serveur écoute sur le port **8080** (depuis l'extérieur). Il monte en lecture/écriture le dossier partagé du serveur de fichiers (voir ci-dessous). Les utilisateurs qui accèdent à nginx1 peuvent déposer et consulter des PDF grâce à l'interface web.
- **nginx2 (serveur Web)** : Similaire à nginx1, il installe aussi Nginx + PHP-FPM et gère le même site web. Il est accessible sur le port **8081**. Comme nginx1, il monte le dossier partagé du serveur de fichiers (via NFS), de sorte que les deux serveurs web voient toujours les mêmes fichiers dans `/var/www/html/upload`.
- **storage (serveur de fichiers)** : C'est un serveur dédié au stockage. Il contient le répertoire `/var/www/html/upload` qui reçoit les fichiers PDF envoyés depuis les sites web. Ce dossier est **partagé via NFS** (Network File System) ⁷ pour être accessible par les deux serveurs web. Autrement dit, storage exporte (met à disposition) ce répertoire sur le réseau. Les clients (nginx1 et nginx2) peuvent alors *monter* ce dossier comme s'il faisait partie de leur propre disque.

Stockage des fichiers

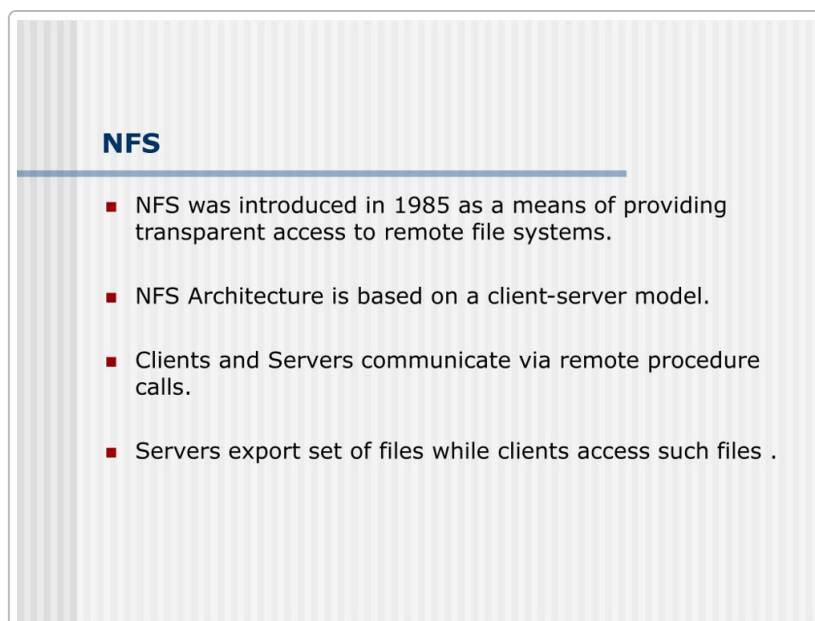


Figure 1 : Exemple de configuration NFS (modèle client-serveur).

Le partage NFS permet à plusieurs machines de partager un même dossier comme s'il était local ⁷. Dans notre cas, le serveur **storage** joue le rôle de serveur NFS et « exporte » le dossier `/var/www/html/upload`. Les deux serveurs web (nginx1 et nginx2) agissent comme clients NFS : chacun monte ce dossier exporté de storage dans son système de fichiers. Ainsi, tout fichier déposé dans `/var/www/html/upload` sur le serveur de fichiers est immédiatement visible par les deux serveurs web. Ce mécanisme permet à plusieurs utilisateurs (via nginx1 ou nginx2) de partager les mêmes documents PDF : l'upload, la consultation et la suppression des fichiers se font toujours dans ce dossier commun.

Accès web et interface utilisateur

Les utilisateurs accèdent au site web depuis leur navigateur. Ils tapent par exemple `http://<adresse_de_la_VM>:8080` pour atteindre nginx1, ou `:8081` pour nginx2. (Dans un environnement de développement Vagrant, on peut rediriger ces ports vers la machine hôte.) Le site web propose un formulaire pour **uploader des fichiers PDF**, et affiche la liste des fichiers présents. Chaque fichier peut être supprimé via l'interface. Les deux serveurs nginx1 et nginx2 servent exactement le même site web (identique copie du code) et utilisent le dossier upload partagé pour stocker les fichiers.

Sur chaque serveur web, **Nginx** reçoit les requêtes HTTP et renvoie les pages web ⁵. Quand la page contient du code PHP, Nginx délègue l'exécution à **PHP-FPM**, qui gère un pool de processus PHP pour répondre rapidement aux requêtes ⁶. En résumé, Nginx sert le site statique et demande à PHP-FPM de traiter les scripts dynamiques (formulaire d'upload, génération de liste, suppression de fichier). Les fichiers PDF sont stockés dans `/var/www/html/upload`, mais ce chemin est en réalité le point de montage NFS vers le serveur storage.

Automatisation avec Ansible

Toutes les machines sont configurées automatiquement grâce à **Ansible**, un outil d'automatisation de l'infrastructure ⁴. Dans Vagrant, on utilise le *provisioner* `ansible_local`, ce qui signifie qu'Ansible

est exécuté directement **à l'intérieur** de chaque VM pour la configurer ³. Concrètement, un playbook Ansible réalise les tâches suivantes sur chaque machine :

- Sur **storage** : installer le service NFS (nfs-kernel-server), créer le dossier `/var/www/html/upload`, configurer `/etc/exports` pour partager ce dossier et démarrer le service NFS.
- Sur **nginx1/nginx2** : installer Nginx, PHP-FPM et le client NFS (nfs-common), configurer le montage NFS du dossier `/var/www/html/upload` dans leur système de fichiers (par exemple en ajoutant une ligne dans `/etc/fstab`), copier les fichiers du site web dans `/var/www/html`, et démarrer les services nginx et php-fpm.

Grâce à Ansible, ces étapes sont automatisées : dès qu'on fait `vagrant up`, les machines se préparent toutes seules sans intervention manuelle. Cela garantit que nginx1 et nginx2 sont prêts à servir le site sur les ports 8080/8081, et que storage partage automatiquement le dossier pour l'upload des fichiers.

Sources : description de Vagrant et machines virtuelles ¹ ², définition de Nginx et PHP-FPM ⁵ ⁶, fonctionnement de NFS ⁷, et rôle d'Ansible pour l'automatisation ⁴ ³.

¹ Vagrant (software) - Wikipedia

[https://en.wikipedia.org/wiki/Vagrant_\(software\)](https://en.wikipedia.org/wiki/Vagrant_(software))

² Understanding Virtual Machines: A Beginner's Guide - RackSolutions

<https://www.racksolutions.com/news/blog/understanding-virtual-machines-a-beginners-guide/?srsltid=AfmBOoq7ZQHQe6EZWI2wJVIBiYywq2fLv5CR7hjQVTcVCOhrTVtrXhCt>

³ Ansible Local - Provisioning | Vagrant | HashiCorp Developer

https://developer.hashicorp.com/vagrant/docs/provisioning/ansible_local

⁴ Ansible : Automatisation IT et Infrastructure as Code ☸

<https://foxeet.fr/contenu/ansible-automatisation-it>

⁵ Qu'est-ce que Nginx ? Qu'est-ce que c'est et comment ça marche

<https://kinsta.com/fr/blog/qu-est-ce-que-nginx/>

⁶ How to Configure PHP-FPM with NGINX for Secure PHP Processing | DigitalOcean

<https://www.digitalocean.com/community/tutorials/php-fpm-nginx>

⁷ Configure NFS Server and Client using Ansible | by Maher Askri | Medium

<https://maheraskri.medium.com/configure-nfs-server-and-client-using-ansible-bea387a9d303>