Migration WordPress

NAS Synology → VM Proxmox (Debian 13)

Rémi Albertucci

Avril 2026

Table des matières

I Contexte

I.1 Architecture

Le portfolio WordPress était hébergé sur un NAS Synology via Web Station.

Cette solution était pratique car le nas était alors mon seul appareil réseau, un serveur multitâches. Mais le fait d'avoir une solution proposée par un grand constructeur et un logiciel propriétaire était rigide…

Pour plus de flexibilité, nous allons migrer le portfolio vers une VM Debian 13 sur Proxmox, tout en conservant les fichiers sur le NAS via un montage SMB.

De cette manière, l'ajout ou la modification de fichiers peut être géré de la même manière par un glisser-déposer sur l'interface du NAS, tout en reprenant le contrôle de wordpress.

I.2 Stack technique

OS : Debian 13 (Trixie)

Serveur web : Apache 2.4

Base de données : MariaDB 11.x

Langage : PHP 8.4

CMS : WordPress avec Elementor

Reverse proxy : Nginx Proxy Manager (NPM)

Partage fichiers : CIFS/SMB depuis NAS

II Préparation

II.1 Export de la base de données

Depuis phpMyAdmin sur le NAS (http://IP_NAS/phpMyAdmin) :

Sélectionner la base de données WordPress

Onglet Exporter => Format SQL => Télécharger

On conserve le fichier .sql sur son poste

Le nom de la base et les identifiants MariaDB sont visibles dans wp-config.php sur le NAS. Noter DB_NAME, DB_USER et DB_PASSWORD — ils seront réutilisés à l'identique sur la VM pour éviter de modifier wp-config.php.

Valeurs à noter :

- define('DB_NAME', '<NOM>');
- define('DB_USER', '<USER>');
- define('DB_PASSWORD', 'mot_de_passe');
- define('DB_HOST', 'localhost');

II.2 Création de la VM Proxmox

Dans l'interface Proxmox, créer une VM avec les caractéristiques suivantes :

- OS : Debian 13 (Trixie) — ISO depuis debian.org
- CPU : 1 vCPUs
- RAM : 2 Go minimum
- Disque : 8 Go
- Réseau : bridge sur l'interface principale

Pendant l'installation Debian :

Cocher SSH server uniquement

Ne pas cocher « serveur web » — Apache sera installé manuellement

Créer un utilisateur standard avec accès sudo

Prendre un snapshot Proxmox à la fin de l'installation Debian, avant toute configuration.

II.3 IP fixe

Configurer une IP fixe pour la VM pour ne pas dépendre du DHCP. Éditer /etc/network/interfaces :

```
allow-hotplug ens18
iface ens18 inet static
    address 192.168.11.200
    netmask 255.255.255.0
    gateway 192.168.11.1
    dns-nameservers 8.8.8.8 1.1.1.1
```

II.4 Reverse Proxy — NPM

Mis dans la préparation, il faudra effectuer cette étape à la fin. On peut par contre préparer le reverse proxy mais ne pas le lancer, pour éviter l'interruption de service ou que notre machine non configurée soit accessible depuis l'extérieur...

Sur Nginx Proxy Manager, modifier le proxy host existant pour pointer vers la nouvelle VM

- Forward Hostname/IP : 192.168.11.200 (nouvelle IP fixe de la VM)
- Forward Port : 80
- SSL Let's Encrypt déjà configuré — rien à changer

III Préparation de la VM

III.1 Mise à jour et installation de paquets

Au besoin, on peut ajouter l'utilisateur au groupe sudo, depuis root :

```
usermod -aG sudo <USER>
sudo apt update && sudo apt upgrade -y
sudo apt install cifs-utils sudo -y
```

III.2 Utilisateur SMB dédié sur le NAS

Sur le NAS, créer un utilisateur dédié (ex: wordpress_smb) sans accès aux applications DSM, avec accès lecture/écriture uniquement sur le dossier portfolio. Cela limite l'impact en cas de fuite des credentials.

III.3 Fichier de credentials SMB sécurisé

```
sudo mkdir -p /root/.smb
sudo nano /root/.smb/creds

# Contenu :
username=wordpress_smb
password=mot_de_passe_nas

# Sécuriser le fichier (lecture root
uniquement)
sudo chmod 600 /root/.smb/creds
sudo chown root:root /root/.smb/creds
```

III.4 Montage SMB dans /etc/fstab

Le dossier portfolio est monté depuis le NAS. L'option uid=www-data permet à Apache d'écrire dedans (plugins, uploads, thèmes). file_mode et dir_mode forcent les permissions à 755 sur le montage CIFS — sans ces options, l'installation de plugins échoue.

```
sudo mkdir -p /home/<USER>/portfolio
sudo nano /etc/fstab
```

```
# Ajouter cette ligne :
//IP_NAS/nom_partage
/home/<USER>/portfolio cifs
credentials=/root/.smb/creds,rw,uid=www-
data,gid=www-
data,file_mode=0755,dir_mode=0755,_netdev
0 0

# Monter le partage
sudo mount -a && sudo systemctl daemon-
reload
```

'_netdev' indique à Debian d'attendre que le réseau soit disponible avant de monter le partage. Sans cette option, le boot peut échouer si le NAS n'est pas encore joignable.

III.5 Transfert du fichier SQL

```
# Depuis son poste local
scp /chemin/local/export.sql
<USER>@IP_VM:/home/<USER>/
```

Faire un snapshot Proxmox ici, avant l'installation de LAMP.

III.6 Mises à jour automatiques

Unattended-upgrades applique automatiquement les mises à jour de sécurité sans intervention manuelle. Seules les mises à jour de sécurité sont appliquées, pas les mises à jour de version qui pourraient casser des choses.

```
sudo apt install -y unattended-upgrades
sudo dpkg-reconfigure --priority=low
unattended-upgrades
# Répondre Oui à la question

# Vérifier que c'est actif :
cat /etc/apt/apt.conf.d/20auto-upgrades
# Doit afficher :
# APT::Periodic::Update-Package-Lists "1";
# APT::Periodic::Unattended-Upgrade "1";
```

IV Migration

IV.1 Installation de la stack LAMP

Paquets

```
sudo apt install -y apache2 mariadb-server
php php-mysql php-curl \
  php-gd php-mbstring php-xml php-zip
libapache2-mod-php
```

Activation de mod rewrite

Indispensable pour les permaliens WordPress :

```
sudo a2enmod rewrite
sudo systemctl restart apache2
```

IV.2 Configuration de MariaDB

```
sudo mariadb-secure-installation

# Répondre aux questions :
# Switch to unix_socket authentication → N
(déjà actif par défaut sur Debian 13)
# Change the root password → Y
# Remove anonymous users → Y
# Disallow root login remotely → Y
# Remove test database → Y
# Reload privilege tables → Y
```

Sur Debian 13, MariaDB utilise l'authentification unix_socket par défaut : root ne peut se connecter que via 'sudo mariadb', sans exposition réseau possible. C'est plus sécurisé qu'un mot de passe traditionnel.

IV.3 Création de la base de données

Utiliser les mêmes identifiants que sur le NAS pour que wp-config.php soit directement compatible :

```
sudo mariadb -u root
```

```
CREATE DATABASE portfolio;
CREATE USER '<USER>'@'localhost'
IDENTIFIED BY
'mot_de_passe_identique_au_nas';
GRANT ALL PRIVILEGES ON portfolio.* TO
'<USER>'@'localhost';
FLUSH PRIVILEGES;
EXIT;
```

IV.4 Import de la base de données

```
sudo mariadb portfolio <
/home/<USER>/export.sql

# Vérifier que les tables sont bien importées
sudo mariadb -u root -e "USE portfolio;
SHOW TABLES;"
```

Si une erreur 'table already exists' apparaît, recréer la base proprement : DROP DATABASE portfolio; CREATE DATABASE portfolio; puis relancer l'import.

IV.5 Configuration d'Apache

VirtualHost

```
sudo nano
/etc/apache2/sites-available/portfolio.conf

<VirtualHost *:80>
    DocumentRoot ~/portfolio
    ServerName <NDD>

    <Directory ~/portfolio>
        AllowOverride All
        Require all granted
    </Directory>

    # Bloquer xmlrpc.php
    <Files xmlrpc.php>
        Require all denied
```

```
    </Files>

    # Limiter wp-login.php au réseau interne
et VPN
    <Files wp-login.php>
        Require ip 192.168.11.0/24
        Require ip 10.8.0.0/24
    </Files>

    # Bloquer wp-admin depuis l'extérieur
(nécessite mod_remoteip)
    RemoteIPHeader X-Forwarded-For
    RemoteIPTrustedProxy 192.168.11.252

    <Directory ~/portfolio/wp-admin>
        Require ip 192.168.11.0/24
        Require ip 10.8.0.0/24
    </Directory>

    # Headers de sécurité
    Header always set X-Frame-Options
"SAMEORIGIN"
    Header always set X-Content-Type-Options
"nosniff"
    Header always set Referrer-Policy "strict-
origin-when-cross-origin"
    Header always set Permissions-Policy
"geolocation=(), microphone=(), camera=()"
</VirtualHost>
```

Activer le site, désactiver le site par défaut, activer le module headers

```
sudo a2ensite portfolio.conf
sudo a2dissite 000-default.conf
sudo a2enmod headers
sudo a2enmod remoteip
sudo systemctl reload apache2
```

ServerName global

Pour éviter le warning 'Could not reliably determine the server fully qualified domain name' dans les logs Apache, on ajoute à la fin de /etc/apache2/apache2.conf :
+ 1

```
ServerName localhost
```

```
sudo systemctl reload apache2
```

Sécurisation

Dans /etc/apache2/conf-available/security.conf :

```
ServerTokens Prod       # Cache la version d'Apache
ServerSignature Off     # Supprime la signature dans les pages d'erreur
```

Désactiver les modules inutiles

```
sudo a2dismod status autoindex
sudo systemctl reload apache2
```

IV.6 Configuration PHP

Localiser le bon php.ini :

```
php --ini | grep 'Loaded Configuration'
```

Valeurs recommandées pour WordPress avec Elementor. Attention : Elementor est très gourmand en mémoire, 1024M est recommandé pour éviter les erreurs 500 à la publication.

```
memory_limit = 1024M
upload_max_filesize = 64M
post_max_size = 64M         # Doit être >= upload_max_filesize
max_execution_time = 60
max_input_time = 60
max_input_vars = 3000       # Certains thèmes dépassent la valeur par défaut de 1000
```

```
sudo systemctl reload apache2
```

V Configuration WordPress

V.1 Mixed Content derrière NPM

NPM fait la terminaison SSL mais la VM reçoit du HTTP. Sans ce fix, WordPress génère des URLs en http:// pour les assets (CSS, JS, images), ce qui provoque des erreurs Mixed Content et un site affiché sans mise en forme.

```
# Ajouter dans wp-config.php (avant le
'That's all, stop editing!') :
define('FORCE_SSL_ADMIN', true);
if
(isset($_SERVER['HTTP_X_FORWARDED_PR
OTO']) &&
$_SERVER['HTTP_X_FORWARDED_PROTO']
=== 'https') {
    $_SERVER['HTTPS'] = 'on';
}
```

V.2 Désactiver l'éditeur de fichiers

Empêche la modification directe des fichiers PHP depuis le dashboard WordPress, même en cas de compromission du compte admin :

```
# Ajouter dans wp-config.php :
define('DISALLOW_FILE_EDIT', true);
```

V.3 Mise à jour des URLs en base

Si les URLs en base pointent encore vers l'ancienne adresse :

```
sudo mariadb -u root

UPDATE portfolio.wp_options SET
option_value = 'https://<NDD>' WHERE
option_name = 'siteurl';
UPDATE portfolio.wp_options SET
option_value = 'https://<NDD>' WHERE
option_name = 'home';
EXIT;
```

VI Sécurisation

VI.1 Firewall UFW

N'autoriser que SSH depuis le réseau local et HTTP depuis NPM uniquement. Le HTTPS est géré par NPM, la VM n'a pas besoin du port 443.

```
sudo ufw allow from 192.168.11.0/24 to any
port 22   # SSH réseau local
sudo ufw allow from 192.168.11.XXX to any
port 80    # HTTP depuis NPM uniquement
sudo ufw enable

# Vérifier les règles
sudo ufw status verbose
```

Remplacer 192.168.11.XXX par l'IP réelle de NPM et 192.168.11.0/24 par la plage réseau locale.

VI.2 Fail2ban

Fail2ban protège contre les tentatives de brute force sur SSH et sur wp-login.php. Ne jamais modifier les .conf de base — toujours créer des .local qui les surchargent.

Configuration principale

```
sudo apt install -y fail2ban

sudo nano /etc/fail2ban/jail.local

[DEFAULT]
bantime  = 24h
findtime = 10m
maxretry = 3

[sshd]
enabled = true

[wordpress]
enabled  = true
port     = http,https
filter   = wordpress
logpath  = /var/log/wordpress/debug.log
maxretry = 3
```

```
bantime  = 24h
```

Filtre WordPress

```
sudo nano
/etc/fail2ban/filter.d/wordpress.conf

[Definition]
failregex = ^<HOST> .* "POST /wp-login.php
            ^<HOST> .* "POST /xmlrpc.php
ignoreregex =
```

Debug log WordPress et démarrage

Le fichier debug.log doit exister avant de démarrer Fail2ban. S'il est absent, la jail WordPress est ignorée silencieusement sans message d'erreur.

Dans wp-config.php :

```
define('WP_DEBUG', true);
define('WP_DEBUG_LOG',
'/var/log/wordpress/debug.log') ;
define('WP_DEBUG_DISPLAY', false);
```

On crée le dossier et le fichier de log en local sur la VM, puis on donne les droits à Apache avant de lancer Fail2ban :

```
sudo mkdir -p /var/log/wordpress
sudo touch
/var/log/wordpress/debug.log
sudo chown -R www-data:www-data
/var/log/wordpress
```

Démarrer et vérifier les deux jails :

```
sudo systemctl enable fail2ban
sudo systemctl restart fail2ban
sudo fail2ban-client status
sudo fail2ban-client status wordpress
```

VI.3 ModSecurity (WAF)

ModSecurity est un Web Application Firewall qui analyse chaque requête HTTP en temps réel avant qu'elle atteigne WordPress. Il bloque les attaques connues : injections SQL, XSS, traversée de répertoires, scanners de vulnérabilités.

Installation

```
sudo apt install -y libapache2-mod-security2
modsecurity-crs
sudo a2enmod security2
sudo systemctl restart apache2
```

Configuration

```
sudo cp /etc/modsecurity/modsecurity.conf-
recommended
/etc/modsecurity/modsecurity.conf
sudo nano
/etc/modsecurity/modsecurity.conf

# Mode détection uniquement (logge sans
bloquer) :
SecRuleEngine DetectionOnly

# Augmenter la limite pour Elementor
(chercher SecRequestBodyNoFilesLimit) :
SecRequestBodyNoFilesLimit 1310720

sudo systemctl restart apache2
```

Laisser ModSecurity en mode DetectionOnly quelques jours et surveiller les logs avant de passer en mode On. Elementor génère des requêtes complexes qui peuvent déclencher des faux positifs.

Passage en mode bloquant

Après quelques jours de surveillance, si les logs sont propres :
Surveiller les détections :

```
sudo tail -f /var/log/apache2/error.log |
grep -i modsecurity
```

Quand c'est propre, passer en mode bloquant dans modsecurity.conf :

```
SecRuleEngine On

sudo systemctl restart apache2
```

VII Récapitulatif des protections

- **Réseau (UFW) :**
  - Port 22 limité au réseau local uniquement
  - Port 80 limité à l'IP de NPM uniquement

- **Apache :**
  - Version Apache masquée (ServerTokens Prod)
  - Modules status et autoindex désactivés
  - xmlrpc.php bloqué
  - wp-login.php limité au réseau local + VPN
  - Headers : X-Frame-Options, X-Content-Type-Options, Referrer-Policy, Permissions-Policy

- **MariaDB**
  - User dédié avec droits limités à la base portfolio
  - Root uniquement via unix_socket (pas d'accès réseau possible)

- **WordPress :**
  - Éditeur de fichiers désactivé (DISALLOW_FILE_EDIT)
  - Debug log activé

- **Fail2ban :**
  - Jail SSH active
  - Jail WordPress active (wp-login.php + xmlrpc.php)
  - Ban 24h après 3 tentatives en 10 minutes

- **ModSecurity**
  - OWASP Core Rule Set actif
  - Mode DetectionOnly — à passer en On après surveillance

- **SMB :**
  - Credentials dans /root/.smb/creds (chmod 600, root uniquement)
  - Utilisateur NAS dédié avec accès limité au dossier portfolio

- **Système**
  - Mises à jour de sécurité appliquées automatiquement
  - IP fixe configurée (192.168.11.200)