**– TP virtualisation —**

[**Introduction 2**](#_jbpez18w6bfk)

[Schéma Infrastructure 3](#_60udrbts35vu)

[**Partie 1 4**](#_4xyljktjdgqu)

[H.A 4](#_vvd3ots5wllz)

[Serveur master UP 4](#_vmlrm0509w8p)

[Serveur master DOWN 4](#_mjcyljwcvxp3)

[Fichier de configuration de corosync.conf 6](#_cv6c5kpz51kd)

[Psense 7](#_fqqrqvhdhs6x)

[**Partie 2 8**](#_opmbrrn9x4pa)

[Choix entre TrueNAS et OpenMediaVault 8](#_xbyb5r90qin8)

[TrueNas 9](#_be7qaw360p10)

[Configuration de VM TrueNAS 9](#_o4v8wlfpnlz4)

[Configuration RAID 10](#_roqjeviwc3lc)

[Avantages et inconvénients : 10](#_w6t2qti9tsls)

[Configuration du stockage : 11](#_7w4hkd9338mn)

[Creation d’un dossier partager et configuration accès 12](#_asiwk3whfw55)

[Configuration du partage TrueNAS (Serveur client) 13](#_t7pfvo7kaxnh)

[Script de sauvegarde 14](#_bwrkdywo6tu9)

[**Automatisation de sauvegarde 15**](#_cx9n5tjjdl2u)

[**Partie 3 16**](#_kuq3mwpvdrau)

[Dockerfile 16](#_t261btgx3uh6)

[Virtualisation d’un serveur Apache : 17](#_2b8ttr59yde6)

[1. Créez les fichiers Dockerfile : 17](#_a6hai6mvekg7)

[file1.dockerfile (PHP 7.4) : 17](#_y1cejlt55px7)

[file2.dockerfile (PHP 8.2) : 17](#_8l5sl5csfgb4)

[2. Modifiez les fichiers de configuration Apache : 18](#_37cxalwjwkyc)

[3. Construisez vos images Docker : 18](#_xy2754lga17v)

[4. Lancez vos conteneurs Docker : 18](#_1182dvtbnjw)

[5. Accédez à vos sites Web : 19](#_exhx3whqs0ye)

[**Conclusion 20**](#_5giow69ga29q)

### 

### 

# Introduction

Dans ce TP , nous mettrons en place la redondance du serveur Web Apache à travers un cluster multi-sites composé de trois serveurs virtualisés, avec un maître et deux esclaves. Nous avons choisi d'utiliser VirtualBox pour cette configuration en raison de contraintes de ressources. Notre démarche comprend la création des machines virtuelles, l'installation des serveurs Web, la configuration du réseau et la mise en place du cluster entre les trois serveurs sur un même réseau. De plus, nous avons ajouté un cluster de pare-feu avec pfSense afin de renforcer la sécurité de l'infrastructure.

Dans les parties 2 et 3 du TP, nous avons intégré un serveur NAS virtualisé à notre infrastructure pour la mise en place de sauvegardes automatisées des données de notre serveur web. De plus, nous avons configuré un serveur Apache capable de faire tourner deux sites avec des versions PHP différentes. Ces ajouts ont pour objectif d'améliorer la robustesse et la flexibilité de notre infrastructure

### 

## Schéma Infrastructure

### 

# Partie 1

## H.A

Tout d'abord, les serveurs sont configurés pour former un cluster en utilisant Corosync et Pacemaker. Chaque serveur 3 : serv web master -serv web slave1-serv web master2 est configuré pour communiquer avec les autres nœuds du cluster via Corosync.

[Créer un cluster HA avec corosync et pacemaker | memo-linux.com](https://memo-linux.com/creer-un-cluster-ha-avec-corosync-et-pacemaker/)

## Serveur master UP

### 

### 

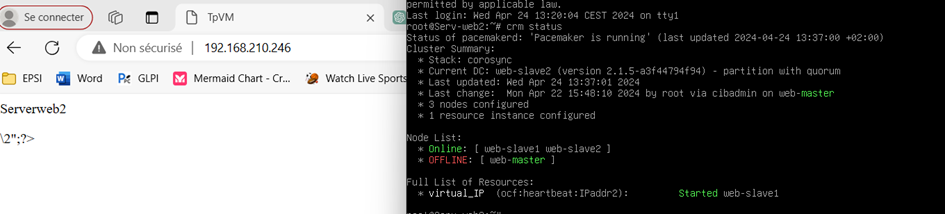
### 

### 

## Serveur master DOWN

Si une ressource tombe en panne sur un nœud dans notre exemple serv web master , Pacemaker peut automatiquement basculer cette ressource vers un autre nœud sain donc içi sur le serv web slave1. Cela garantit que le service reste disponible même en cas de panne matérielle ou logicielle.

Une fois que la panne est résolue ou que le nœud défaillant est réparé, Pacemaker peut automatiquement reprendre les ressources sur ce nœud, ramenant le cluster à son état normal.

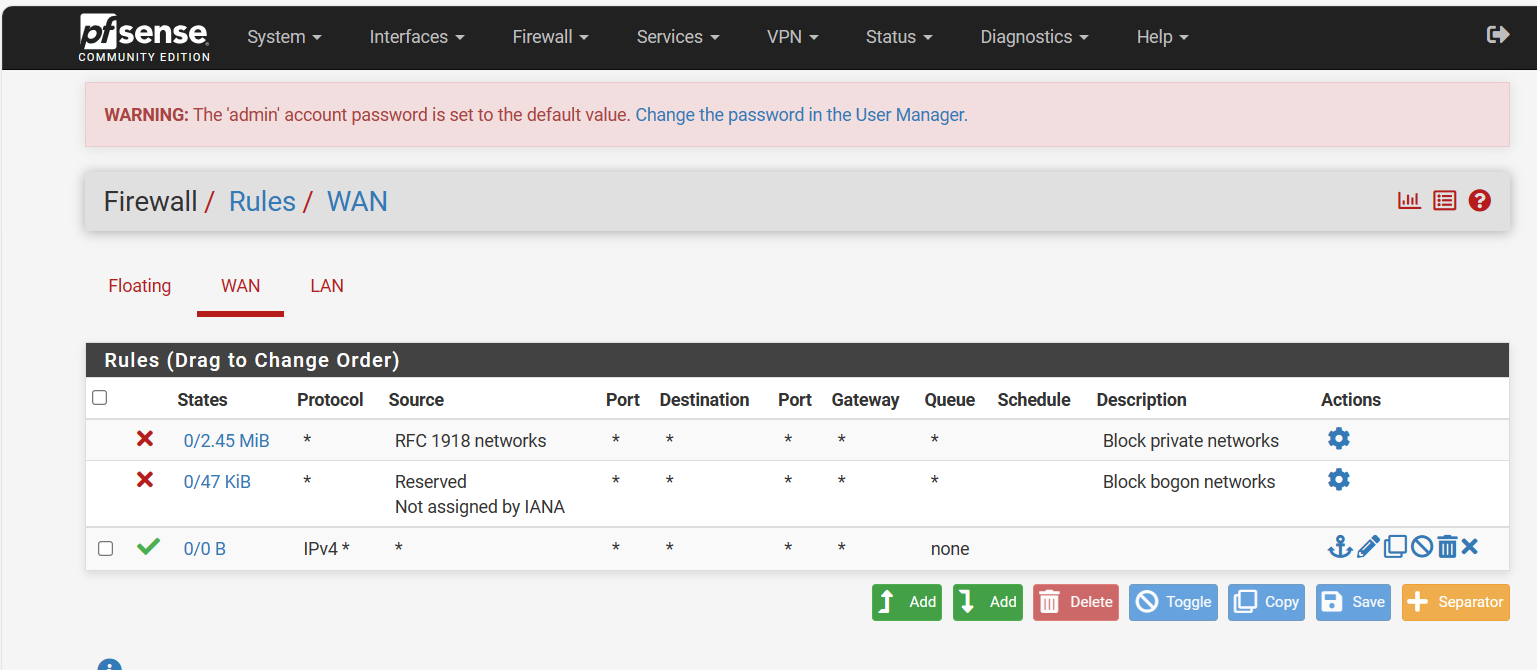


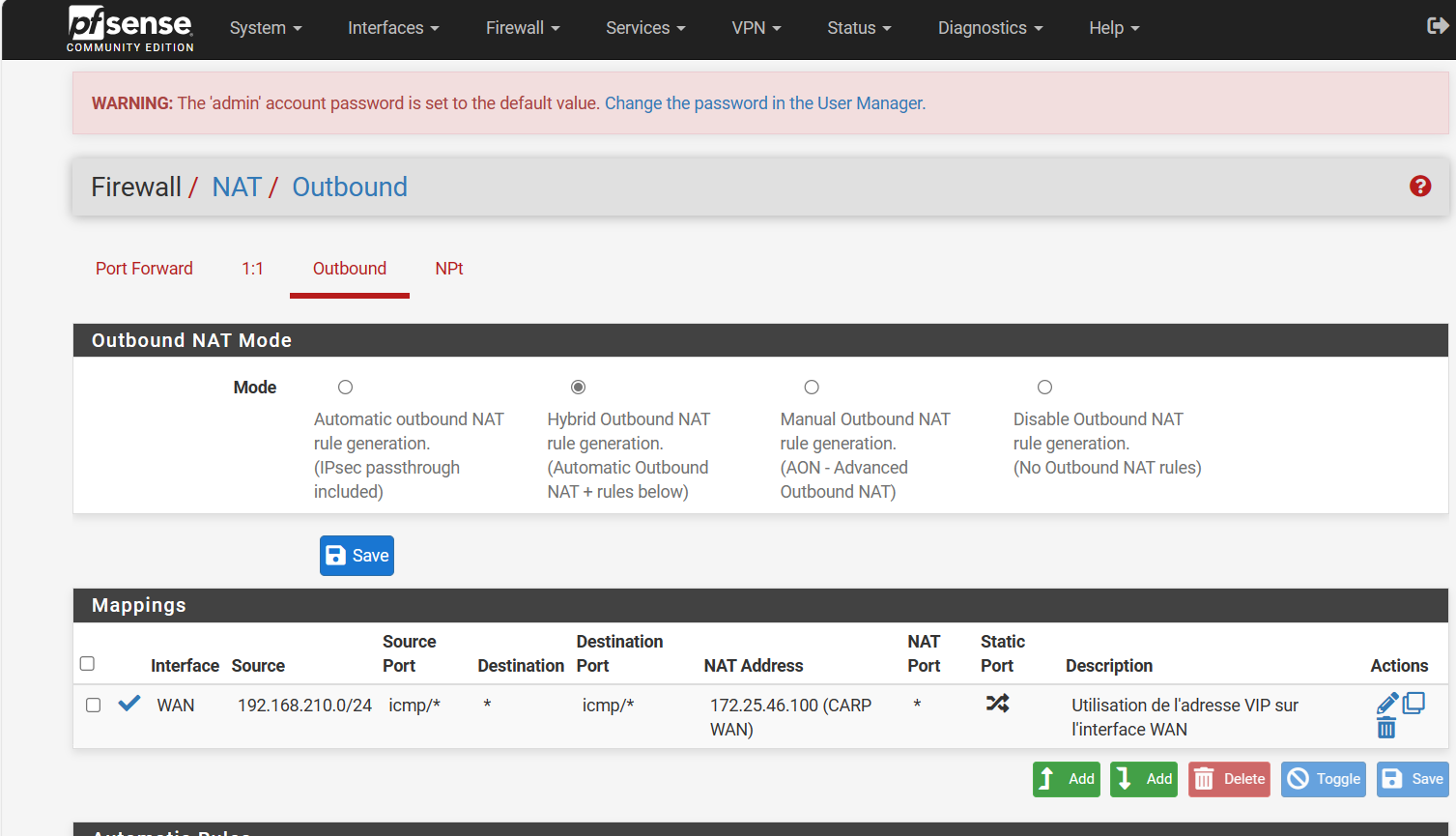
## Fichier de configuration de corosync.conf

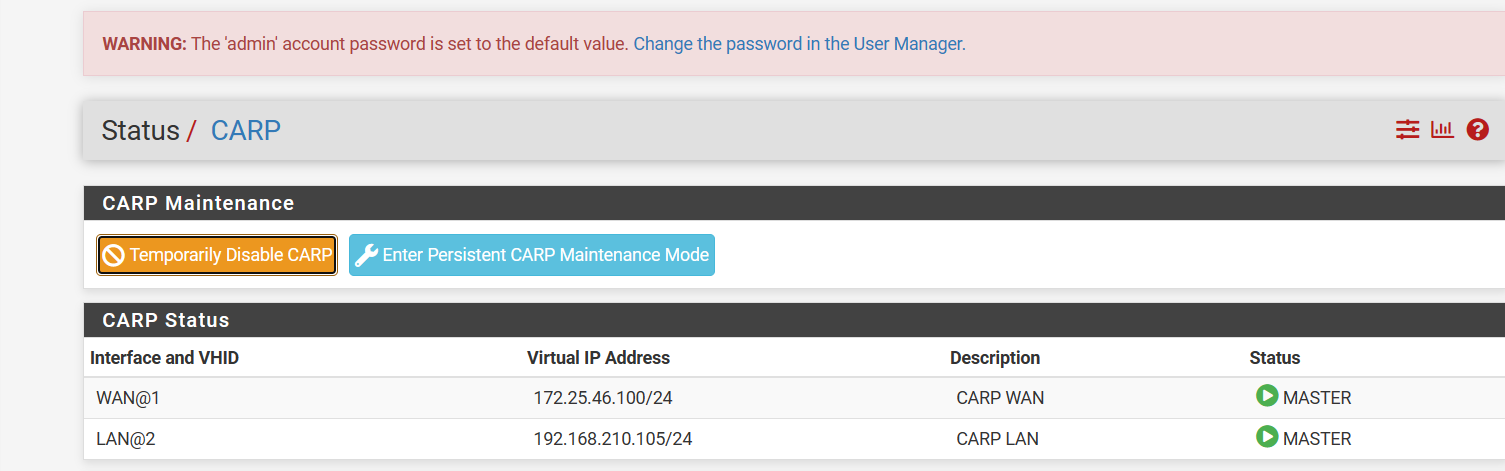
web-master 192.168.210.101- web-slave2 192.168.210.102 -web-slave3 192.168.210.103



## Psense







# Partie 2

Mettre en place un serveur NAS SMB/NFS pour sauvegardes automatisées

##### Choix entre TrueNAS et OpenMediaVault

|  | Avantages: | Inconvénients: |
| --- | --- | --- |
| TrueNAS (anciennement FreeNAS): | Interface utilisateur conviviale.  Fonctionnalités avancées de stockage comme ZFS.  Bonne gestion des sauvegardes et de la sécurité. | Peut nécessiter plus de ressources matérielles.  Interface peut sembler complexe pour les débutants. |
| OpenMediaVault: | Léger et adapté aux petites configurations.  Facile à installer et à configurer.  De nombreux plugins sont disponibles pour étendre les fonctionnalités. | Moins de fonctionnalités avancées par rapport à TrueNAS.  Moins adapté pour les environnements de stockage complexes. |

## TrueNas

Nous avons choisie trueNAS car c’est une plateforme de stockage réseau open source et puissante, conçue pour répondre aux besoins de stockage de données

## Configuration de VM TrueNAS

* 2 processeurs
* 2 Go de mémoire
* 1 disque dur de 20 Go dynamique
* 2 disques durs de 20 G0 (ada1 et ada2)

### 

## Configuration RAID

Des différentes configurations RAID :

* RAID 0 : Striping sans redondance. Améliore les performances mais aucun tolérance de panne.
* RAID 1 : Miroir. Duplication des données sur deux disques pour une redondance maximale.
* RAID 5 : Striping avec parité distribuée. Bon équilibre entre performance et redondance avec un disque de parité réparti sur tous les disques.
* RAID 6 : Comme RAID 5 mais avec double parité pour une meilleure tolérance aux pannes.
* RAID 10 : Combinaison de RAID 1 et RAID 0. Offre à la fois redondance et performances.

## Avantages et inconvénients :

|  | Avantages: | Inconvénients: |
| --- | --- | --- |
| RAID 1 | Meilleure tolérance aux pannes, faible temps de reconstruction. | Utilisation inefficace de l'espace disque (50% seulement). |
| RAID 5 | Bonne combinaison de performance et de redondance, espace disque efficace. | Temps de reconstruction long en cas de panne de disque |
| RAID 6 | Tolérance aux pannes élevée même en cas de double défaillance de disque. | Plus lent en écriture par rapport au RAID 5. |
| RAID 10 | Excellente performance en lecture/écriture, bonne tolérance aux pannes. | Coût élevé en termes d'espace disque (50%). |

## Configuration du stockage :

Pour assurer la redondance et la tolérance aux pannes de nos données, nous allons configurer nos deux disques durs de 20 Go en miroir. Cela correspond à une configuration RAID 1, où les données seront écrites simultanément sur les deux disques, offrant ainsi une redondance complète des donnée

##### 

Donc nous avons bien notre pools “Apachesync” qui est en redondance avec nos deux disques dur ada1et ada2

##### 

## Creation d’un dossier partager et configuration accès

Nous avons créé un utilisateur “santo” qui peuvent accéder au dossier “ santo “

##### 

Nous allons maintenant partager ce dossier sur un serveur client en utilisant le protocole NFS (Network File System). Grâce à ce dossier partagé, nous pourrons centraliser la sauvegarde des données de ce serveur.Le protocole NFS permet le partage de fichiers entre systèmes informatiques via un réseau. En configurant le dossier partagé avec le protocole NFS, nous permettrons à d'autres serveurs ou clients d'accéder aux fichiers et répertoires présents dans ce dossier depuis leur propre système.

##### 

## Configuration du partage TrueNAS (Serveur client)

### 

EXplication de cette commande

- mount : C'est la commande principale que nous utilisons pour monter des systèmes de fichiers sur notre système de fichiers local.

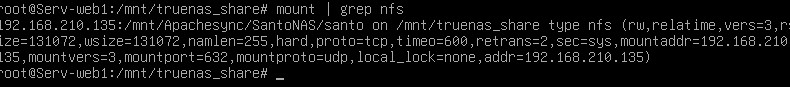
-t nfs : Cette option spécifie le type de système de fichiers à monter. Dans ce cas, nous indiquons `nfs` pour un partage NFS (Network File System).

-o defaults: Cette partie de la commande spécifie les options de montage par défaut.

- defaults : Indique que les options de montage par défaut doivent être utilisées. Cela inclut généralement des options telles que `rw` (lecture/écriture)

- 192.168.210.135:/mnt/Apachesync/SantoNAS/santo : C'est l'emplacement du partage trueNFS distant que nous voulons monter. Cela comprend l'adresse IP du serveur trueNFS (192.168.210.135) suivie du chemin absolu du dossier partagé sur ce serveur (/mnt/Apachesync/SantoNAS/santo).

- /mnt/truenas\_share : C'est l'emplacement sur notre système de fichiers local où le partage trueNFS sera monté. C'est là que nous pourrons accéder aux fichiers et répertoires du partage trueNFS



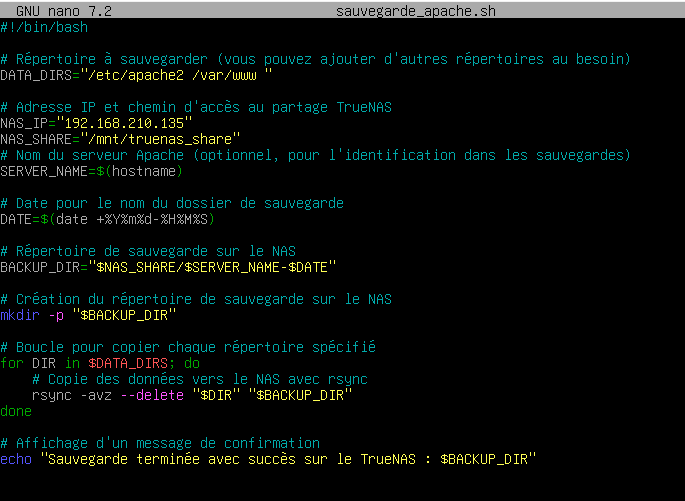
Nous utiliserons le point de montage `/mnt/truenas-share` sur le serveur client pour synchroniser avec le TrueNAS. Plus précisément, nous envisageons de sauvegarder nos données à l'aide d'un script et de les stocker dans `/mnt/truenas-share`, partagé sur nos serveurs TrueNAS. Le script de sauvegarde sera programmé pour s'exécuter toutes les 2 minutes dans notre cas, mais nous avons la possibilité de le configurer selon nos besoins.

En choisissant ce point de montage sur le serveur client, nous assurons une accessibilité aisée à notre espace de stockage sur le TrueNAS. Cette configuration permettra au script de sauvegarde d'accéder facilement aux données nécessaires et de les transférer vers notre système de stockage centralisé. Cette approche garantit une gestion efficace des sauvegardes, offrant la possibilité de centraliser et de sécuriser nos données critiques sur nos serveurs TrueNAS

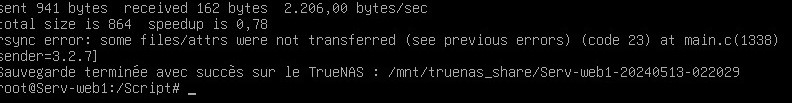
## Script de sauvegarde

Nous avons un script qui sauvegarde quelques dossiers de nos configuration de site web dans notre cas nous avons pris le dossier apache2 qui à les configurations de nos sites-available et le dossier “www “ qui à nos fichier index.php

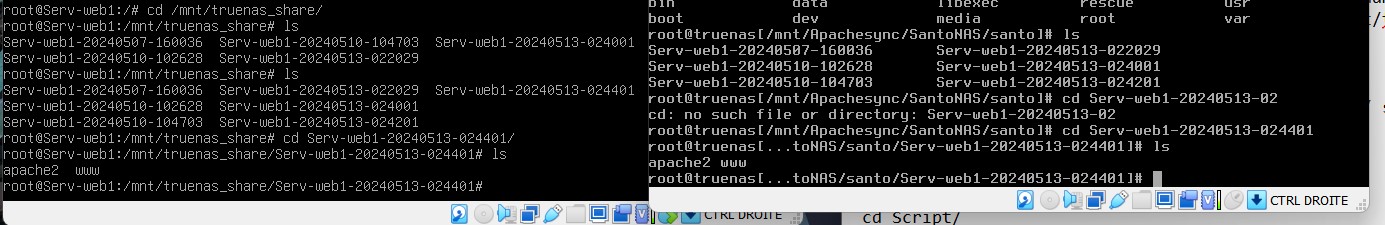
( chmod +x sauvegarde\_apache.sh rendre ce script executable ./sauvegarde\_apache.sh )



Comme nous pouvons voir les sauvegardes on été terminé avec succès



Serveur client Serveur TrueNAS



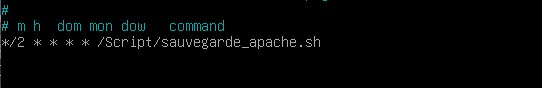
# Automatisation de sauvegarde

Dans le but de planifier l'exécution périodique de notre script de sauvegarde, nous utiliserons le service crontab, disponible sur les systèmes Unix/Linux. Crontab est un service qui permet de planifier l'exécution de tâches à des intervalles réguliers,dans notre cas nous allons enregistrer tous les 2 mns

Edite le fichier crontab



Nous allons ajouter la ligne suivant pour exécuter le script de sauvegarde tous les 2mns



# Partie 3

Dans cette partie, nous devons configurer un serveur Apache virtualisé pour tester la migration d'un applicatif de PHP 7.4 vers PHP 8.2 sur Debian 10. Pour ce faire, nous créons deux sites web qui fonctionneront chacun sous une version différente de PHP : l'un sous PHP 7.4 et l'autre sous PHP 8.2.

## Dockerfile

| FROM debian:10  RUN apt update && apt full-upgrade -y  # Install some basic tools  RUN apt-get -y install lsb-release ca-certificates curl  # Install the sury.org repository key  RUN curl -sSLo /tmp/debsuryorg-archive-keyring.deb https://packages.sury.org/debsuryorg-archive-keyring.deb  RUN dpkg -i /tmp/debsuryorg-archive-keyring.deb  RUN sh -c 'echo "deb [signed-by=/usr/share/keyrings/deb.sury.org-php.gpg] https://packages.sury.org/php/ $(lsb\_release -sc) main" > /etc/apt/sources.list.d/php.list'  RUN apt update  # Install PHP  RUN apt-get -y install php8.2  RUN apt-get -y install php7.4  RUN apt-get -y install php7.4-cli php7.4-common php7.4-curl php7.4-gd php7.4-json php7.4-mbstring php7.4-mysql php7.4-xml php7.4-zip  # Install Apache  RUN apt-get -y install apache2 libapache2-mod-php7.4 libapache2-mod-php8.2  # Enable PHP module in Apache  RUN a2enmod php8.2  COPY new\_ports.conf /etc/apache2/ports.conf  # ServerName directive  RUN echo "ServerName localhost" >> /etc/apache2/apache2.conf  # Start Apache service  CMD ["apachectl", "-D", "FOREGROUND"]  # Expose port 80  EXPOSE 8081 |
| --- |

## Virtualisation d’un serveur Apache :

### 1. Créez les fichiers Dockerfile :

##### file1.dockerfile (PHP 7.4) :

| dockerfile |
| --- |
| FROM php:7.4-apache  COPY . /var/www/html/ |

##### file2.dockerfile (PHP 8.2) :

| dockerfile |
| --- |
| FROM php:8.2-apache  COPY . /var/www/html/ |

### 2. Modifiez les fichiers de configuration Apache :

Dans chaque répertoire de votre projet, créez un fichier `info.php` contenant le code suivant :

| php |
| --- |
| <?php  phpinfo();  ?> |

Assurez-vous que ce fichier est inclus dans votre copie (`COPY`) dans les Dockerfiles.

### 3. Construisez vos images Docker :

Dans votre terminal, accédez au répertoire où se trouvent vos Dockerfiles et exécutez les commandes suivantes :

| bash |
| --- |
| docker build -t my-php74-image -f file1.dockerfile .  docker build -t my-php82-image -f file2.dockerfile . |

### 4. Lancez vos conteneurs Docker :

| bash |
| --- |
| docker run -d -p 8080:80 my-php74-image  docker run -d -p 8081:80 my-php82-image |

### 5. Accédez à vos sites Web :

- Pour PHP 7.4 : Ouvrez votre navigateur et entrez `http://localhost:8080/info.php`.

- Pour PHP 8.2 : Ouvrez votre navigateur et entrez `http://localhost:8081/info.php`.

Vous devriez voir les informations PHP correspondant à chaque version.

# Conclusion

En résumé, ce TP nous a permis de comprendre l'importance de la redondance et de la haute disponibilité pour maintenir nos services en ligne en cas de panne. Nous avons appris à configurer des clusters de serveurs pour assurer la continuité des opérations et à renforcer la sécurité avec des pare-feux. En ajoutant un serveur NAS pour les sauvegardes et en optimisant notre serveur Apache, nous avons amélioré la fiabilité et la flexibilité de notre infrastructure.