Sadek

Adel

Sio2

Reverse proxy

Tuto : https://rdr-it.com/nginx-configuration-reverse-proxy/

Je vais installer un reverse proxy avec nginx

Pour ceci je vais installer une nouvelle machine debian 11

Le reverse proxy c'est l'inverse du proxy sa permet de filtrer le trafic entrant (niveau 7) et non sortant et de le rediriger

Si je veux que 2 machines soit accessible sur le port 80 je dis à mon routeur (regle de PAT) que tout le trafic entrant sur son port 80 sera rediriger vers le reverse proxy qui lui en fonction du site demander le renverra vers le PC qui heberge ce site

J'installe nginx

-apt-get install nginx

Je vais dans */etc/nginx/sites-available/ et je crée un fichier de conf par site comme apache2*

Je crée un fichier de configuration qui s'appelera booktic.conf

Dans le fichier de configuration j'utiliserai des arguments voici leur definition

***Upstream <FQDN>** = C'est pour définir vers quel serveur web seront envoyer les requêtes http envoyer au serveur reverse proxy avec cet un FQDN*

Server = Dans ce bloc il y'aura toute les options du serveur web que l'on aura affecter plus haut location du site,à partir d quand est t'il considerer injoignable

- L'option "**location**" permet de gérer où ces requêtes seront redirigées et dans quelle condition. le "/" après location indique que c'est à la racine du serveur web cible que le site se trouve.
- **proxy_pass** : permet d'affecter un ensemble de serveur à cette configuration, le nom à indiqué ici est celui du bloc "*upstream*", qui, vous vous en doutez, peut contenir plusieurs serveurs cible si notre site web est réparti de façon égale sur plusieurs serveurs.
- **proxy_set_header** : permet d'ajouter un header à la requête qui va être passée au serveur web par notre RP. Ici nous ajoutons le header "*Host*" par exemple, le nom de domaine visé par la reqête initiale
- **proxy_connect_timeout** et **proxy_send_timeout** : plus standard, ces directives permettent de gérer le temps (en seconde) au delà duquel le serveur web (backend) sera considéré comme injoignable/down. Autrement dit si la requête passée par notre RP n'a pas de réponse sous 30 secondes, alors un message d'erreur sera renvoyé à l'utilisateur.

Mes deux fichiers de conf ressemble à sa

Je les ai fais comme sa finalement

Tout sera dans le meme fichier

```
Server {
        #Le serveur proxy ecoute sur ce port
        Listen 8080 ;
        #FQDN sur lequel ce bloc s'applique
        Server_name  projet.booktic.info ;
        #Ou est situer le site web sur le serveur web par défaut c'est la racine
        location / {
        #Vers quel machine + port est rediriger le trafic qui correspond au bloc
        Proxy_pass  htt://ipDuServeurWeb :port/ ;
}
}
```

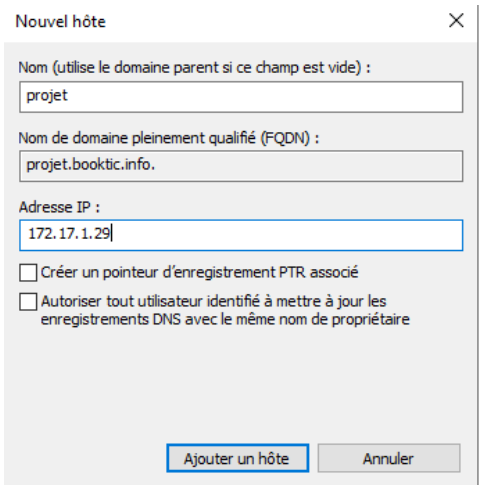
Le fichier de conf ressemble à sa

```
server{
        listen 8080 ;

        server_name projet.booktic.info ;
        location / {
        proxy_pass http://172.17.1.83:80 ;
        }
}
```

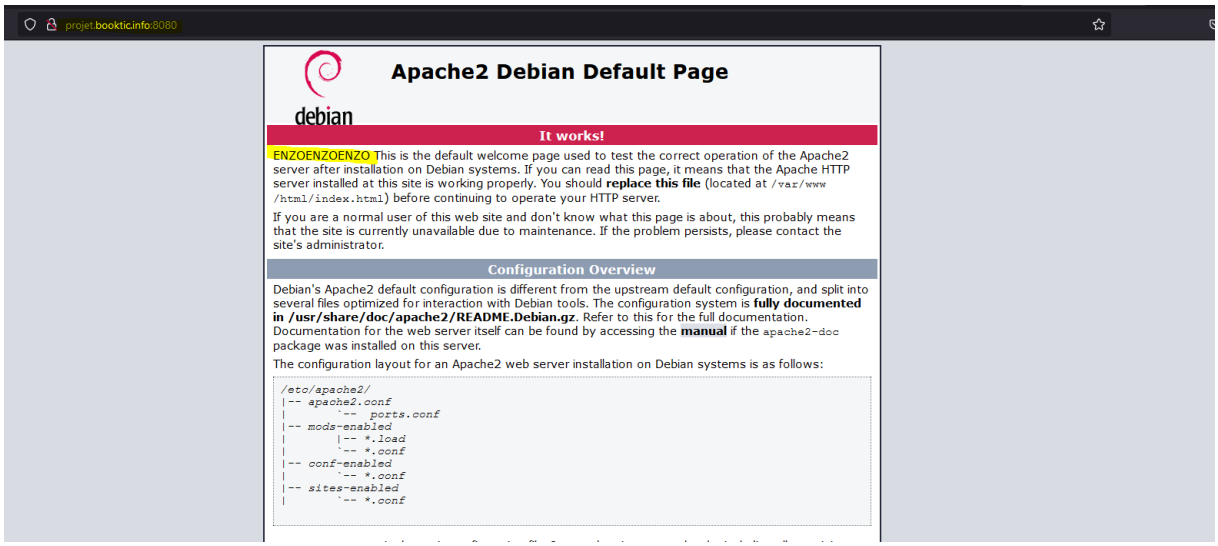
Il faut que je crée un enregistrement DNS

Projet.booktic.info IN A @ipReverseProxy

Comme ceci



Ensuite je fais un test pour le test sa redirige vers le serveur WEB d'Enzo



C’est parfait sa fonctionne

Gabriel s’occupe de créer les deux serveur LAMP sur lesquels seront héberger OWNCLOUD

Je prépare le terrain dans le fichier de conf pour le deuxième bloc

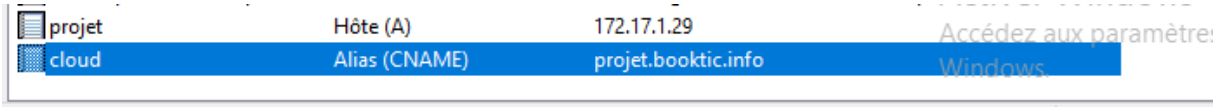
Je modifie les IP pour mettre l’IP des 2 serveurs web dans la DMZ

SRV_LAMP_1=172.18.0.6

SRV_LAMP_2=172.18.0.7

```
server{
        listen 8080 ;

        server_name projet.booktic.info ;
        location / {
        proxy_pass http://172.18.0.6:80 ;
        }
}

server{
        listen 8080 ;

        server_name cloud.booktic.info ;
        location / {
        proxy_pass http://172.18.0.7:80 ;
        }
}
```

Pour l'instant le fichier de conf ressemble à sa je vais faire un enregistrement CNAME entre projet.booktic.info et cloud.booktic.info, comme ceci



Je modifie l'ip du reverse proxy une fois que j'ai verifier qu'il marche bien pour le mettre dans la DMZ (vlan30) son ip sera 172.18.0.8

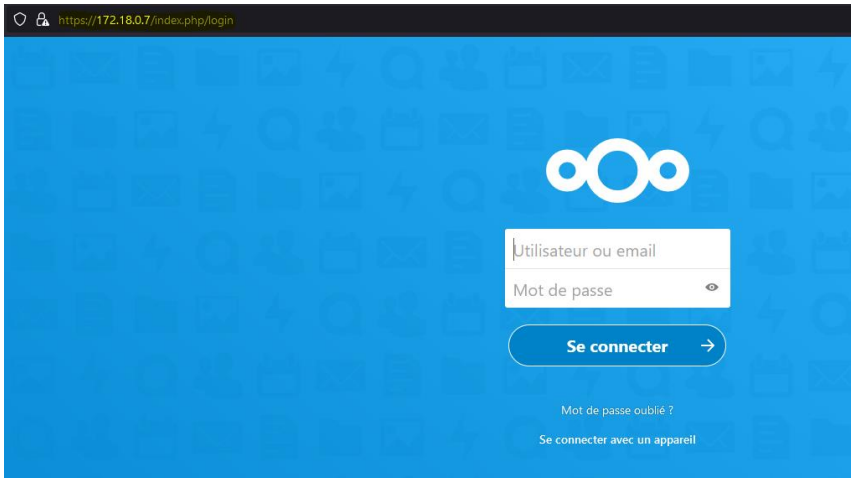
Je modifie l'enregistrement DNS

J'ai modifié les IP des enregistrement DNS pour mettre des IP du vlan DMZ et j'ai commenter ma conf dans NGINX ce dernier écoute désormais sur le port 80 car j'ai supprimer le paquet apache qui avait déjà pris le port 80

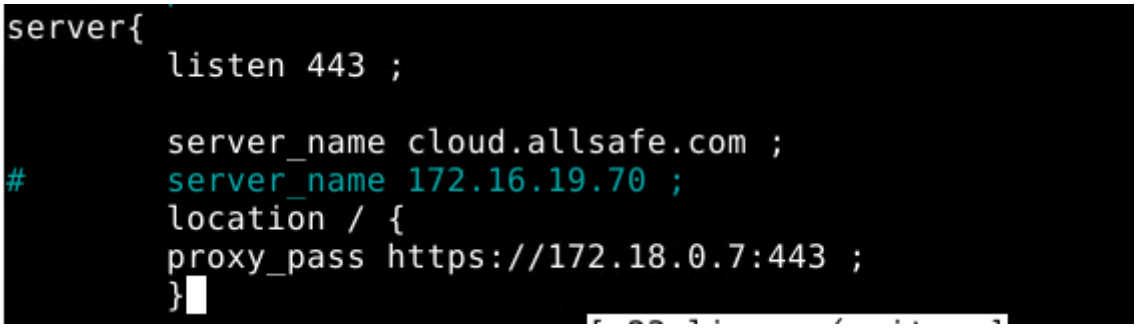
```
  GNU nano 5.4                        booktic.conf
server{
#Toutes les requetes venant sur le port 80 du reverse proxy
        listen 80 ;
#A destination du fqdn projet.booktic.info
        server_name projet.booktic.info ;
#Le contenue du serveur web ce trouve à la racine de ce dernier
        location / {
#L'adresse ip vers laquelle les requetes seront rediriger
        proxy_pass http://172.18.0.6:80 ;
        }
}
#Pareil pour le deuxieme bloc
server{
        listen 80 ;

        server_name cloud.booktic.info ;
        location / {
        proxy_pass https://172.18.0.7:443 ;
        }
}
```

J’ai redémarré le proxy et je test cloud.booktic.info

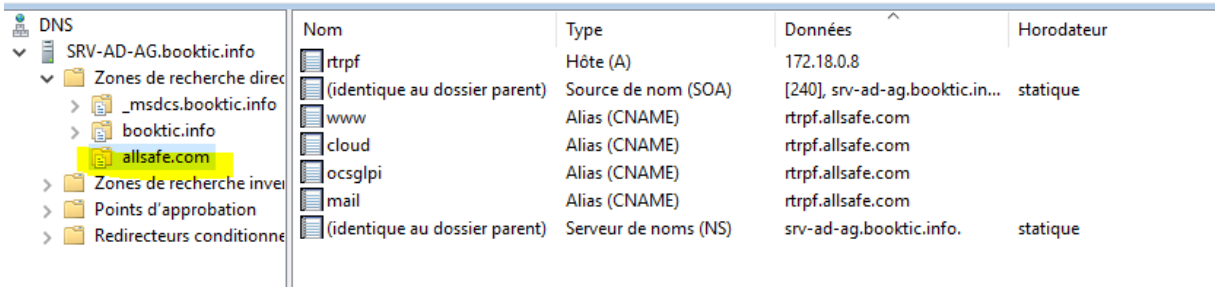
Voilà la redirection est effective



Le FAI nous a attribuer le domaine allsafe.com et à mis comme enregistrement A,MX,CNAME l’ip de notre pfsense de sorte à ce que cloud.allsafe.com redirige vers l’IP de notre routeur et ensuite avec une règle de PAT on renvoie la requête vers le serveur reverse-proxy qui regarde le FQDN demander et renvoie vers la machine paramétrer



Ensuite il faut créer une zone DNS interne de allsafe.com et mettre les enregistrements ip interne dedans les même que l’on a sur booktic pratiquement



Je modifie NGINX pour écouter sur le port 80 et non 443

Dans la variable proxy_pass je mets l'enregistrement DNS interne du cloud à la place je mets cloud.allsafe.com pour le reverse proxy ce FQDN dirige vers l'IP du cloud

SSL AVEC NGINX

Pour éviter l'erreur SSL RX RECORD et pouvoir rediriger vers des sites en https il faut que fasse écouter mon serveur nginx sur le port 443 et que à la fin de la directive listen 443 je mets « ssl ».

Ensuite il faut que je crée une clef publique et privée pour chiffrer ma connexion entre le client qui souhaite SE connecter et le reverse proxy.

Pour cela je les crée avec openssl et je rajouter les directives :

SSL_CERTIFICATE

SSL_CERTIFICATE_KEY

Le fichier de configuration ressemblera à sa :

```
server{
        listen 443 ssl;
#       listen 443 ssl;
        ssl_certificate /ssl/nginx.pem ;
        ssl_certificate_key /ssl/nginx.key ;
        server_name cloud.allsafe.com ;
#       server_name 172.16.19.70 ;
        location / {
        proxy_pass https://172.18.0.7:443 ;
        }
}
```

Pour rediriger http vers https dans la même directive que https il faut écrire ceci

If ($scheme != https) {

return 301 https://$host$request_uri;

}

Pour des CMS comme joomla pour que la connexion soit transparent entre le client et le serveur et que le serveur ne cherche pas à charger les pages en local et reçoit les requêtes en direction de son FQDN et non de son IP il faut changer l'entête des requêtes http dans le fichier de configuration et rajouter ceci

Location / {

Proxy_set_header HOST $host ;

}

Return 301 = Sa permet de renvoyer une requête http 301 pour dire au client d'aller sur l'adresse préciser

```
server{
        listen 100 ;
        listen 443 ssl;
        ssl_certificate /ssl/nginx.pem ;
        ssl_certificate_key /ssl/nginx.key ;
        server_name cloud.allsafe.com;
        location / {
        proxy_set_header HOST $host;
        if ($scheme != https){
        #Je le redirige vers l'ip de mon serveur reverse proxy sur le port 443
        return 301 https://$host$request_uri ;
}
        proxy_pass https://172.18.0.7:443 ;

        }
}
```

Le processus est tel quel :

Le client se connecte via le protocole « http » sur le port 80 de mon haproxy ensuite le Haproxy renvoie vers un des deux serveur nginx sur le port 100 ensuite si la requête est en http le serveur nginx va demander au client de ce connecter à l'URL du HAproxy en https et le processus va recommencer HAproxy ensuite NGINX et comme la requête est en https elle ne rentre pas dans la condition et la nouvelle connexion (https) est redirigée vers un autre site (https)

La requête proxy_pass est après la condition « IF » pour la logique

Nginx avec plusieurs apaches sur le srv reverse-proxy

Je veux héberger plusieurs serveur apache sur la même machine et rendre le port 8080 accessible pour un site qui est réellement hébergé sur un autre port le 6500 pour que plus tard je rajoute d'autre serveur web qui sauront héberger sur d'autre port via des Virtual host mais qui demeuront tous accessible via le port 8080 de ma machine.

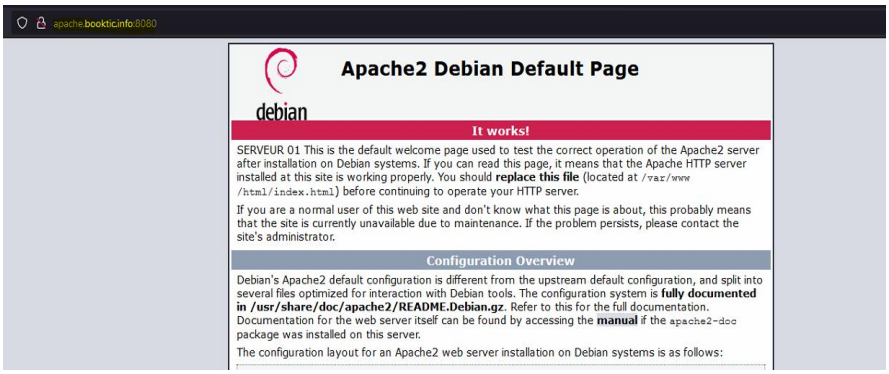
Tout d'abord après avoir installé apache2 je vais dans le fichier par défaut et je modifie le port d'écoute par défaut je le règle sur le 6500

Ensuite sur mon reverse proxy je rajoute un bloc pour mon nouveau serveur web :

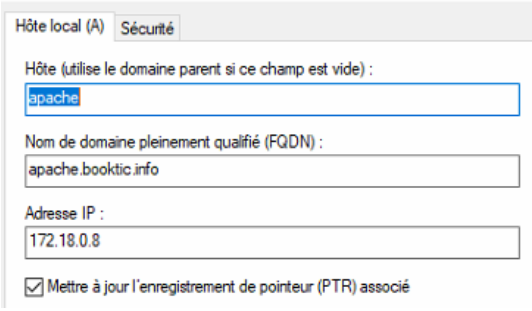
```
server{
        listen 8080;
        server_name apache.booktic.info ;
        location / {
        proxy_pass http://172.18.0.8:6500 ;
        }
}
```

Sur le serveur DNS je fais le mappage entre l'hôte apache.booktic.conf et l'IP du serveur reverse proxy comme sa sa sera à lui que seront adressé les requêtes http pour l'hôte apache.booktic.conf et il pourra renvoyer sa de manière transparente vers le serveur web hébergé sur un autre de ses ports.

Test :



Enregistrement DNS :



Je n'ai aucun serveur sur le port 8080 il n'y a que le serveur reverse proxy qui écoute je conclus que le reverse proxy fonctionne correctement.

Suite sur un autre projet comment faire si j'ai une machine derrière mon reverse proxy qui cherche à communiquer avec elle-même via son IP dans le réseau local pour chercher des templates avec Joomla par exemple ?

- Par défaut une fois qu'on a communiqué avec le reverse proxy pour nous rediriger vers le serveur web si on veut recommuniquer avec le serveur web une fois qu'on est connecté pour chercher une page par exemple sa sera l'adresse IP ou fqdn qui est dans la directive « proxy_pass » qui nous répondra mais si on cherche à communiquer encor une fois avec le FQDN de la machine il faut utiliser la directive « proxy_set_header Host $host ; » pour que si il y'a d'autre requête vers le serveur web sa soit fait au nom du FQDN qui est dans la directive « server_name »

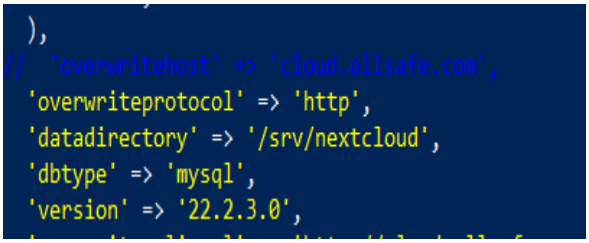
- Cette configuration nous permet de faire en sorte que le proxy transfère les cookies, le nom de domaine le Referer ainsi que la plus part des attributs de l'entête du protocole HTTP ou HTTPS au serveur web qui s'exécute au niveau du contenaire WEB. Ainsi le Reverse proxy sera totalement transparent

- Sa va permettre de rendre le proxy transparent sa sera comme si on communique vraiment directement avec le serveur web

CORRECTION D'UNE PROBLEMATIQUE AVEC OWNCLOUD

Je ne pouvais pas accéder directement à l'IP de mon serveur OWNCLOUD car le fichier de configuration de owncloud mon camarade avait forcer la redirection des connexion vers le serveur de web de sorte si on y accède via un autre « FQDN ou via son IP » la réécriture d'URL ce fait automatiquement

J'ai donc commenté cette directive



Répartition de charges entre plusieurs serveurs web

Pour faire en sorte que ça soit un groupe de serveur qui soit solliciter il faut au préalable définir un upstream qui ressemble à sa

Upstream planning{

Server 172.17.1.99 :80 weight=5 ;

Server 172.17.1.82 :8080 weight=1 ;

}

Il faut définir les IP et port dedans la directive weight sert à définir au bout de combien de requêtes le serveur web ne sera plus solliciter et le serveur en dessous sera solliciter sa fonctionne en boucle ici par exemple au bout de 5 requêtes le premier serveur web renvoie au second serveur web qui au bout de 1 requête, sa sera le premier serveur web qui sera de nouveau solliciter.

Sa sera l'algorithme weight roundrobin qui sera utilisé.

Ensuite la directive proxy_pass ressemble à ceci

Proxy_pass http://planning;

Configuration des serveurs web

J'ai fait écouter mon serveur second serveur web sur le port 8080 pour éviter tout problème avec les virtualhost mettre un serveur web par port et à la fin c'est totalement transparent

Les virtualhost pour ne pas avoir de problème mieux vaut les définir dans le même fichier de configuration

Au niveau des certificats

Pour avoir un certificat par serveur il faut mettre la couche TLS au niveau du nginx car sur HAproxy je n'ai pas trouvé comment faire par crainte de casser tout le reste du travail est d'être confronter à des erreurs mais sur nginx c'est totalement possible

Lorsque la couche TLS est mit en place au niveau du serveur nginx et non du serveur web la directive proxy_pass est suivi de « http » appart si il y'a aussi un certificat sur le serveur web mais ce n'est pas conseillé sa peut causer beaucoup d'erreur(trop de redirection etc..)

Mettre en place un serveur web nginx

Mettre en place un serveur web nginx est très simple

Il faut simplement rajouter ceci dans le fichier de configuration

```
server {
   listen 80;
   server_name localhost;
   location / {
      root /usr/share/nginx/html;
      index index.html index.htm;
   }
```

Root = équivalent de document root d'apache pour définir un répertoire ou seront stocker les fichiers appartenant au site

Index = Pour définir la page d'accueil afficher equivalent d'IndexDirectory sur apache2

Listen = le serveur écoute sur quel port

Server_name = FQDN du site

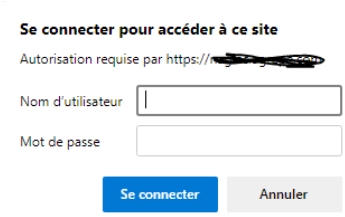
Nagios mise en place authentification

Pour mettre une place une authentification c'est un peu le même principe qu'apache 2 on définit le type d'authentification avec un fichier qui contiendra le nom d'utilisateur et le mot de passe hasher

```
        auth_basic      "Veuillez vous identifiez !";
        auth_basic_user_file    /auth/user ;
```

Ensuite créer le fichier avec la commande htpasswd comme sur apache pas besoin de le repréciser ici

Ensuite je fais un test



C'est parfait sa fonctionne correctement attention à ne pas mettre le fichier d'utilisateur dans sites-enabled/ car nginx pourrait le voir comme un fichier de conf

Résolution du problème "Lack of sufficient authorization" lors du renouvellement de certificats Let's Encrypt

Introduction :

Cette documentation vous guide à travers la résolution du problème courant "Lack of sufficient authorization" qui peut survenir lors du renouvellement de certificats SSL Let's Encrypt sur un serveur reverse proxy. Ce problème se produit généralement lorsque Let's Encrypt ne peut pas vérifier la possession du domaine par votre serveur. Pour résoudre ce problème, nous devrons configurer un répertoire web spécifique pour les vérifications ACME Challenge de Let's Encrypt en utilisant l'option --webroot avec Certbot.

Prérequis :

Un serveur reverse proxy configuré avec des certificats SSL Let's Encrypt.

Certbot installé sur votre serveur.

Accès SSH au serveur.

Étape 1 : Configuration du répertoire web pour les vérifications ACME Challenge :

Connectez-vous à votre serveur (le reverse proxy biensur tout ce fait sur ce dernier rien ne s'effectue sur le serveur web car avec le processus que nous allons mettre en place la verification de la possession du domaine par certbot s'effectuera qu'entre certbot et votre serveur reverse proxy) via SSH.

Créez le répertoire **/var/www/html/.well-known/acme-challenge/** si ce n'est pas déjà fait. C'est dans ce répertoire que Let's Encrypt placera les fichiers de vérification. Vous pouvez le créer avec la commande suivante :

```
sudo mkdir -p /var/www/html/.well-known/acme-challenge/
```

Assurez-vous que le répertoire /var/www/html/ est accessible en écriture par Certbot. Vous pouvez le faire en modifiant les permissions du répertoire :

Ensuite, ajoutez la directive suivante dans votre configuration Nginx ou Apache, en fonction du serveur web que vous utilisez. Cette directive indique au serveur web de servir les fichiers dans le répertoire spécifié pour les vérifications ACME Challenge :

Pour Nginx (ajoutez ceci à votre bloc de serveur) :

Je préfère mettre cette directive dans le bloc du port 80 et 443 plutôt que me contenter du bloc 443 (https)

```
location /.well-known/acme-challenge/ {
    alias /var/www/html/.well-known/acme-challenge/;
    try_files $uri =404;
}
```

Enregistrez les modifications dans la configuration du serveur web et rechargez le serveur pour les appliquer :

```
sudo systemctl reload nginx
```

Étape 2 : Renouvellement du certificat Let's Encrypt avec --webroot :

Maintenant que vous avez configuré le répertoire web pour les vérifications ACME Challenge, vous pouvez renouveler votre certificat Let's Encrypt en utilisant l'option --webroot avec Certbot. Assurez-vous de spécifier le chemin du répertoire web que vous avez défini dans la section précédente. Par exemple :

```
sudo certbot certonly --webroot -w /var/www/html -d votredomaine.com
```

Suivez les instructions de Certbot pour terminer le renouvellement.

Principe du Webroot :

L'option --webroot de Certbot vous permet de spécifier un répertoire web sur votre serveur où Let's Encrypt peut placer temporairement des fichiers de vérification pour prouver la possession du

domaine. Cela permet à Let's Encrypt de vérifier que vous avez le contrôle sur le domaine sans interrompre le fonctionnement normal de votre site web.

Conclusion :

En suivant ces étapes, vous avez résolu le problème "Lack of sufficient authorization" lors du renouvellement de certificats Let's Encrypt sur un serveur reverse proxy. Vous avez configuré un répertoire web spécifique pour les vérifications ACME Challenge en utilisant l'option --webroot avec Certbot, permettant à Let's Encrypt de valider votre domaine avec succès. Assurez-vous de renouveler régulièrement vos certificats pour maintenir un chiffrement SSL sécurisé sur votre serveur reverse proxy.

Tout ce joue pratiquement au niveau du reverse proxy

Nous avons commencé par discuter de la réécriture d'URL avec Nginx et comment vous pouvez utiliser Nginx pour rediriger une URL comme https://testjoomla.agrepe.com/index.php/abcd vers https://testjoomla.agrepe.com/abcd/. Pour cela, vous devez ajouter une règle de réécriture à la configuration Nginx. Voici un exemple de règle de réécriture pour Nginx :

```
location /abcd/ {
    proxy_pass http://votreserveurweb.com/index.php/abcd;
}
```

Cette règle redirigera les requêtes pour l'URL

https://testjoomla.agrepe.com/abcd/ vers le serveur web situé à l'adresse http://votreserveurweb.com/index.php/abcd.

Nous avons également discuté de la façon d'activer les URLs de réécriture dans Joomla pour supprimer "index.php" des URLs de votre site. Pour cela, vous devez activer l'option "URLs de réécriture" dans la configuration Joomla. Voici les étapes à suivre :

Connectez-vous à l'interface d'administration de Joomla

Accédez au menu "Système" et sélectionnez "Configuration"

Dans l'onglet "Site", recherchez la section "Paramètres SEO" et activez l'option "URLs de réécriture"

Enregistrez les modifications

Cela activera les URLs de réécriture pour votre site Joomla, ce qui permettra de supprimer "index.php" des URLs de votre site.

Pour faire référence à "/blog" plutôt que "/abcd", vous pouvez simplement remplacer "abcd" par "blog" dans les règles de réécriture et les URL. Par exemple :

Pour la règle de réécriture Nginx :

```
location /blog/ {
    proxy_pass http://votreserveurweb.com/index.php/blog;
}
```

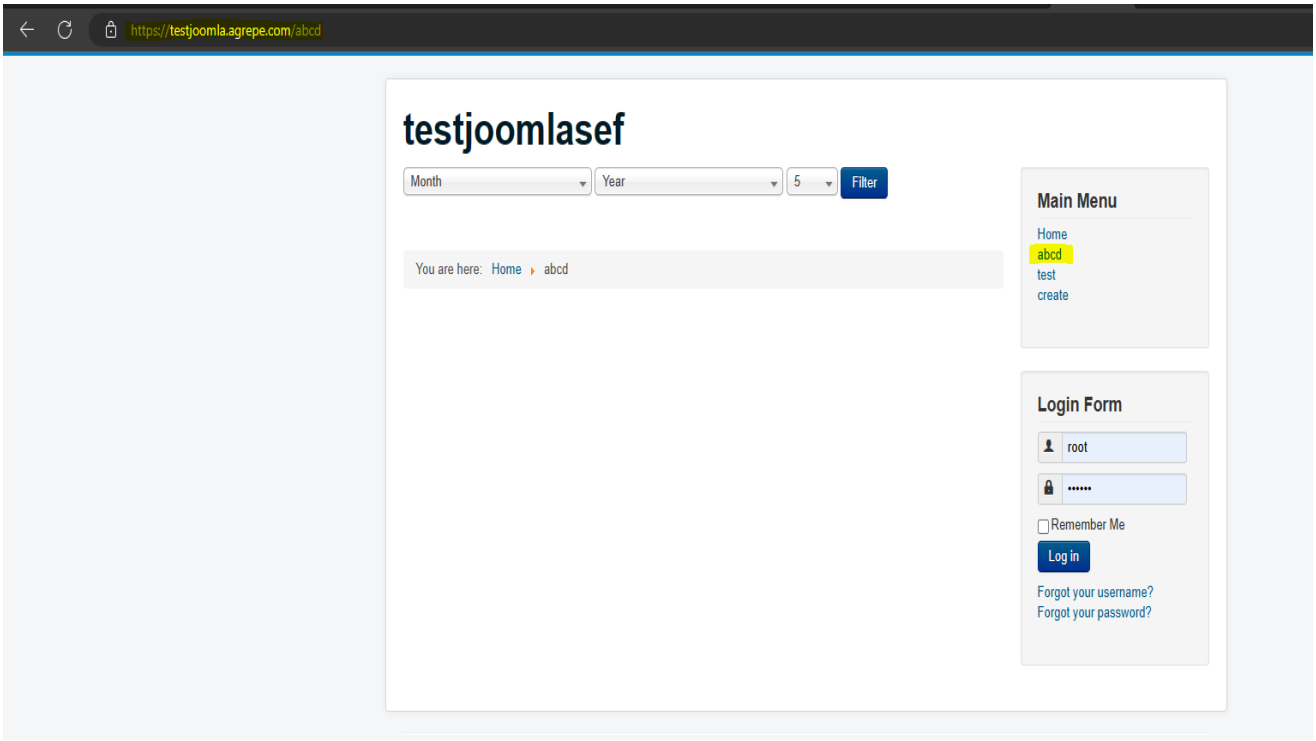
Pour l'URL de la page d'accueil Joomla :

https://testjoomla.agrepe.com/blog/

Pour la règle de réécriture Apache (si vous utilisez Apache plutôt que Nginx) :

RewriteRule ^blog/(.*)$ http://votreserveurweb.com/$1 [P]

Résultat :



Pour que la page par défaut soit index.php, vous pouvez ajouter la directive index à votre bloc de configuration nginx pour le serveur ou la location que vous utilisez. Par exemple, si votre site utilise un fichier index.php dans le

répertoire racine /var/www/html, vous pouvez ajouter la directive suivante dans votre bloc de configuration:

perl

Copy code

```
server {
    listen 80;
    server_name example.com;
    root /var/www/html;
    index index.php index.html;
    location / {
        # votre configuration de proxy
    }
}
```

Cela indique à nginx d'utiliser index.php comme page d'index par défaut, en essayant d'abord d'afficher index.php, puis index.html s'il ne peut pas trouver index.php.

Assurez-vous également que le fichier index.php existe bien dans le répertoire /var/www/html. Si vous avez modifié la configuration de la racine de votre site, vous devez vous assurer que les fichiers sont au bon endroit.

Possible solution aussi

Voici un résumé de la configuration de la réécriture pour rediriger les URL de la forme https://dvpsbookingcaraibe.agrepe.com/index.php/fr/... vers https://dvpsbookingcaraibe.agrepe.com/fr/... :

Dans la configuration du serveur web (probablement nginx), ajoutez le bloc de code suivant :

```
location /fr {
    rewrite ^/index\.php/fr(/.*)$ /fr$1 permanent;
    proxy_pass https://192.168.1.132:443/index.php/fr;
}
```

Cela redirigera les URL de la forme **/index.php/fr/...** vers **/fr/...**, puis passera la requête au serveur backend à l'adresse **https://192.168.1.132:443/index.php/fr**. Cela devrait supprimer **index.php** de l'URL.

Il est important de noter que cette configuration suppose que le serveur est derrière un reverse proxy nginx.