2021

Projet REVERSE-PROXY SSL sous Nginx

Réalisé par :

Winness RAKOTOZAFY

AVANT – PROPOS

La société BeTheFirst dispose d'un site de vente en ligne ainsi que d'un site pour la gestion du SAV, aucun des serveurs n'a de certificat SSL, la gestion du SI ne respecte pas les bonnes pratiques et les moteurs de recherche ne référencent plus les sites du client.

Pour régler cette problématique, nous proposons de remettre à plat l'infrastructure afin de retrouver une architecture homogène et sécurisée, qui permettra l'intégration d'un nouvel ERP/CRM.

Pour informations, cette documentation est établie en validation d'un projet académique lors de ma formation chez Webforce3, et est adopté comme une proposition de solution à la problématique en place et une description des différents services utilisés.

SOMMAIRE

AVANT – PROPOS i

INFRASTRUCTURE INFORMATIQUE 1

1- Firewall pfsense 2

2- Reverse proxy 2

3- Serveur DNS 3

4- Site vente et SAV (sous Apache) 4

5- Site ERP et CRM (sous Nginx) 4

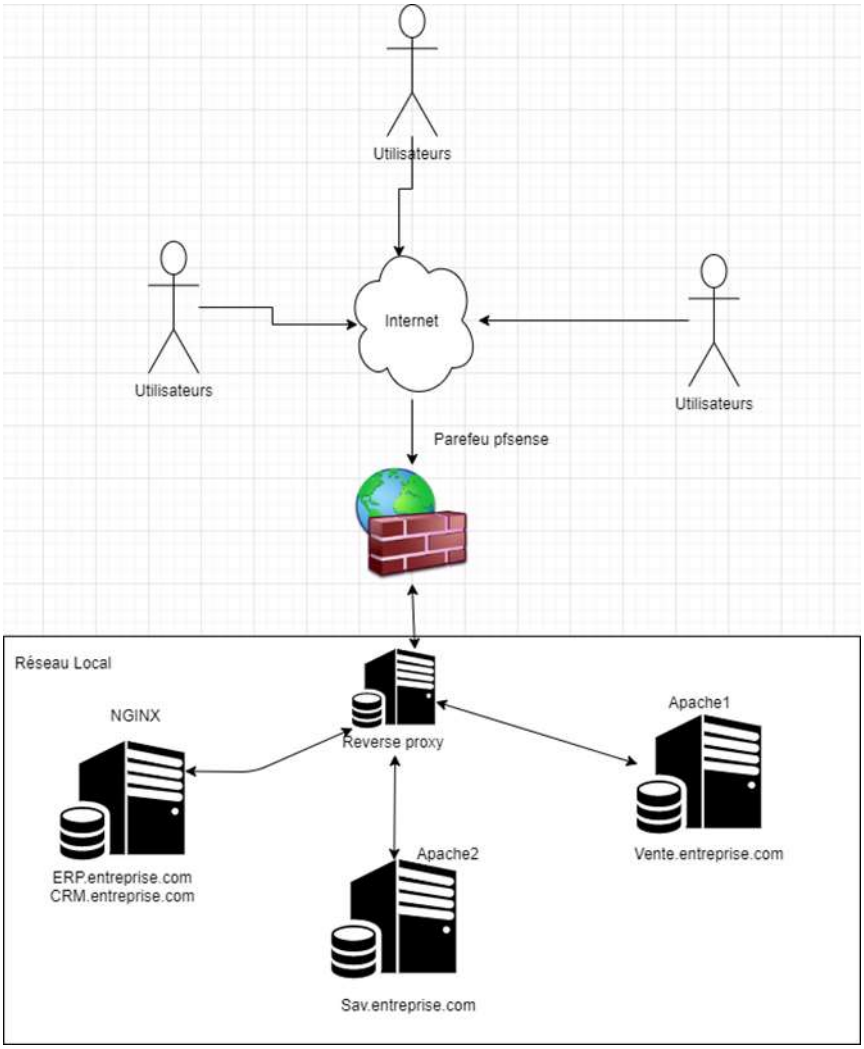
6- Iptables 4

7- Fail2Ban 8

INFRASTRUCTURE INFORMATIQUE

Le nouveau système d'information (SI) se composera de :

- 1 serveur reverse proxy SSL (nginx)
- 2 serveurs web apache (sav/ventes.entreprise.com)
- 1 serveur web nginx (erp.entreprise.com et crm.entreprise.com)



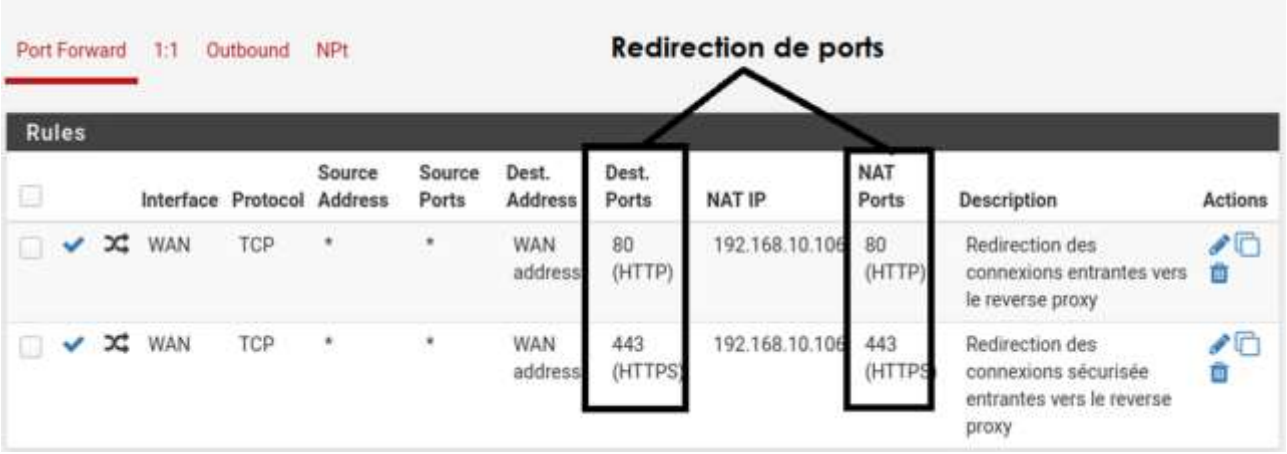
1- Firewall pfsense

Le pfsense est un pare-feu open-source (libre de droit) permettant de sécuriser votre SI des tentatives d'intrusion externe.

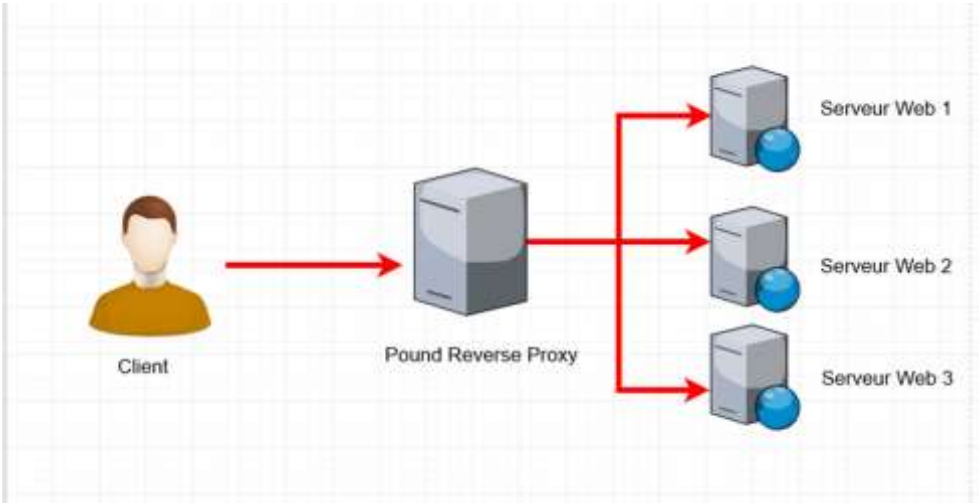
Il est le premier outil mis en place afin de protéger les serveurs du trafic externe.

Il va également nous être utile à assurer la sécurisation de vos sites web en appliquant une règle permettant l'accès sécurisé de votre SI.

⇨ Paramètres appliqués



2- Reverse proxy



Il remplit le rôle de passerelle entre internet et le réseau local, il donne l'accès via internet aux serveurs dans votre réseau de façon que lorsque l'on cherchera à joindre vos sites, les utilisateurs n'interrogeront que le reverse proxy.

Le reverse proxy se chargera d'aiguiller les requêtes vers les serveurs demandés.

Pour finir, on lui attribuera certificats de nos serveurs web.

```
server {
        listen 443;
        server_name vente.entreprise.com;

ssl on;
ssl_certificate /etc/ssl/ventes.crt;
ssl_certificate_key /etc/ssl/ventes.key;

access_log /var/log/nginx/access.log;
error_log /var/log/nginx/error.log;


        location / {
                proxy_pass https://192.168.10.101;
                proxy_set_header Host $host;
                        proxy_set_header                X-Forwarded-For
        $proxy_add_x_forwarded_for;
        proxy_connect_timeout 30;               proxy_send_timeout 30;
        }
}
#Quand un client effectuera une requête sur internet en cherchant vente.entreprise.com
#Le reverse proxy aiguillera le trafic vers le serveur d'apache ventes
#Toutes les requêtes destinées à une adresse publique sont transférer par le reverse-proxy
#A l'adresse interne spécifiée dans le ProxyPass.

#header X-Forwarded-For : il va nous permettre de changer l'ip source arrivant du client
#en transformant cette ip par celle du reverse proxy qui lui seul pourra atteindre le serveur web
# cela sécurise encore plus notre accès aux serveurs web.
```

3- Serveur DNS

Le DNS « Domain Name System » ou en Français le Système de noms de domaine est un service informatique permettant de traduire une adresse IP en nom de domaine.

Le DNS permet donc d'associer un nom de domaine à une adresse IP et facilite la recherche d'un site sur internet.

```
; BIND reverse data file for empty rfc1918 zone
;
; DO NOT EDIT THIS FILE - it is used for multiple zones.
; Instead, copy it, edit named.conf, and use that copy.
;
$TTL    86400
@       IN      SOA     entreprise.com. root.entreprise.com. (
                              1         ; Serial
                         604800         ; Refresh
                          86400         ; Retry
                        2419200         ; Expire
                          86400 )       ; Negative Cache TTL
;
@       IN      NS      dns1.entreprise.com.

dns1    IN      A       192.168.10.181


vente   IN      A       192.168.10.101
sav     IN      A       192.168.10.102
erp     IN      A       192.168.10.103
crm     IN      A       192.168.10.103
"db.entreprise.com" 23L, 507C                                 23,0-1        Tout
```

Nom de domaine déclaré entreprise.com

Sous-domaines déclaré associés à leurs adresse ip.

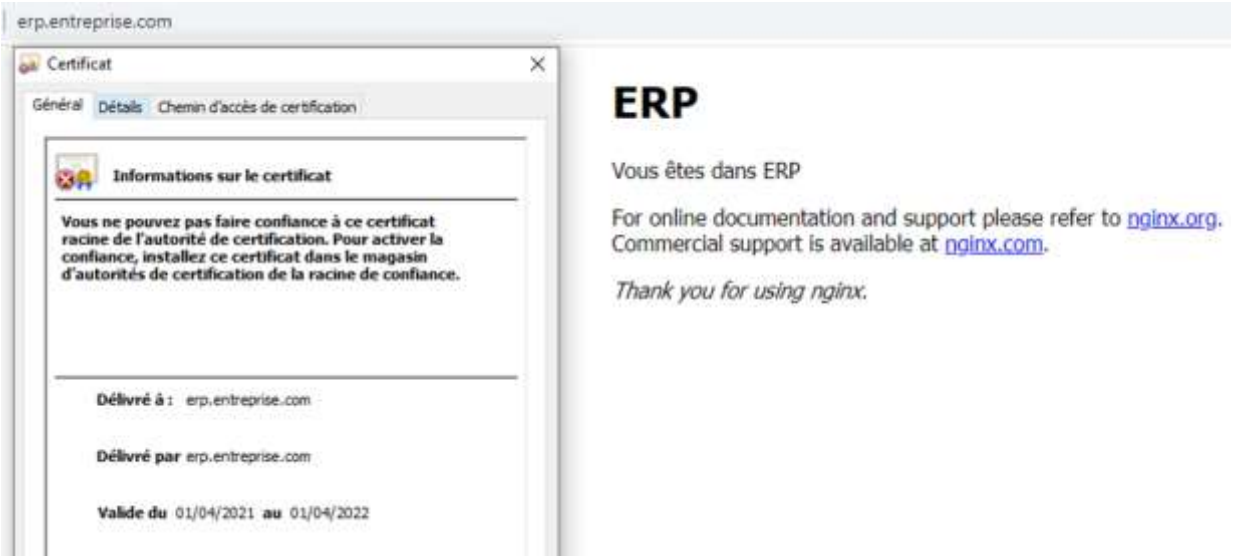
4- Site vente et SAV (sous Apache)

D'après votre infrastructure vos serveurs pour les sites Vente et SAV étaient configurés sans connexions sécurisés.

Nous avons donc apporté une sécurisation SSL à l'aide de certificats pour sécuriser vos serveurs web et ainsi les référencer sur google.

5- Site ERP et CRM (sous Nginx)

Comme convenu, nous avons ajouté un serveur web Nginx à votre infrastructure qui comporte deux sites web : erp.entreprise.com/crm.entreprise.com que nous avons sécurisé en SSL à l'aide de certificats.



6- Iptables

Pour mieux sécuriser le déploiement et l'utilisation de vos serveurs, nous avons mis en place précédemment les certificats SSL, et actuellement, nous allons mettre en fonction les Iptables sur chaque serveur, qui jouera le rôle de pare-feu sur vos serveurs sous Linux.

Pourquoi utiliser Iptables ? Par défaut, toutes les données sont envoyées sous forme de paquets sur Internet, et toutes données venant d'Internet peuvent infiltrer votre serveur. Actuellement, cela ne reflète plus la cybersécurité, et notamment avec la montée des

cyberattaques, il serait préférable de filtrer les trafics entrants et sortants d'Internet sur le serveur.

- **Les bases d'Iptables**

Iptables est une application de ligne de commande et un pare-feu Linux que vous pouvez utiliser pour configurer, maintenir et inspecter les tableaux de filtres de paquets. Plusieurs tableaux peuvent être définis. Chaque tableau peut contenir plusieurs chaînes. Une chaîne n'est qu'un ensemble de règles. Chaque règle définit ce qu'il faut faire avec le paquet, s'il correspond à ce paquet.

Dans l'iptable, il existe trois sortes de chaînes :

- **INPUT** : qui permet d'analyser les paquets entrants, et si le paquet est adressé au poste, il est confronté au filtre INPUT
- **FORWARD** qui permet d'analyser et d'autoriser les trames à passer d'une interface à une autre, seulement dans le cadre d'une interface réseau servant de passerelle.
- **OUTPUT** qui permet d'analyser les paquets sortants, et si le paquet sort du poste, il passera par la chaine OUTPUT.

A cette table, on peut affecter des politiques (règles de pare-feu) :

- **ACCEPT** où les paquets seront autorisés à passer
- **DROP** où les paquets ne seront pas autorisés à passer
- **RETURN** où l'on cesse de parcourir cette chaîne pour retourner dans la chaîne précédente (appelante) en passant à la règle suivante.

A savoir qu'il y existe deux méthodes pour activer notre iptables : la première consistera à créer un script bash qui s'exécutera à chaque démarrage des serveurs, la seconde consiste à installer les services iptables et les rendre persistantes, pour qu'ils soient toujours actives après reboot.

1ère méthode : Script Bash

Pour créer un script, il nous suffit de créer un fichier de script, et en faire un exécutable de démarrage. Pour ce faire, il nous faut créer le fichier script (moniptables) dans le dossier /etc/init.d/, et lui donner des droits d'exécution :

```
sudo vim /etc/init.d/moniptables
sudo chmod +x /etc/init.d/moniptables
```

```
sudo update-rc.d moniptables default
```

Voici un exemple de script iptables provenant du site www.community.jaguar-network.com:

```
#!/bin/bash
echo Setting firewall rules...
#
# config de base
#

# Vider les tables actuelles
iptables -t filter -F
iptables -t filter -X
echo - Vidage : [OK]

# Autoriser SSH
iptables -t filter -A INPUT -p tcp --dport 22 -j ACCEPT
echo - Autoriser SSH : [OK]

# Ne pas casser les connexions établies
iptables -A INPUT -m state --state RELATED,ESTABLISHED -j ACCEPT
iptables -A OUTPUT -m state --state RELATED,ESTABLISHED -j ACCEPT
echo - Ne pas casser les connexions établies : [OK]

# Interdire toute connexion entrante
iptables -t filter -P INPUT DROP
iptables -t filter -P FORWARD DROP
echo - Interdire toute connexion entrante : [OK]

# Interdire toute connexion sortante
iptables -t filter -P OUTPUT DROP
echo - Interdire toute connexion sortante : [OK]

# Autoriser les requetes DNS, FTP, HTTP, NTP etc. SORTANTES
iptables -t filter -A OUTPUT -p tcp --dport 20 -j ACCEPT
iptables -t filter -A OUTPUT -p tcp --dport 21 -j ACCEPT
iptables -t filter -A OUTPUT -p tcp --dport 80 -j ACCEPT
iptables -t filter -A OUTPUT -p udp --dport 53 -j ACCEPT
iptables -t filter -A OUTPUT -p tcp --dport 53 -j ACCEPT
iptables -t filter -A OUTPUT -p udp --dport 123 -j ACCEPT
echo - Autoriser les requêtes DNS, FTP, HTTP, NTP : [OK]

# Autoriser loopback
iptables -t filter -A INPUT -i lo -j ACCEPT
iptables -t filter -A OUTPUT -o lo -j ACCEPT
echo - Autoriser loopback : [OK]

# Autoriser ping
iptables -t filter -A INPUT -p icmp -j ACCEPT
iptables -t filter -A OUTPUT -p icmp -j ACCEPT
echo - Autoriser ping : [OK]

# Gestion des connexions ENTRANTES autorisées
#
# iptables -t filter -A INPUT -p  --dport -j ACCEPT

# Autoriser HTTP, HTTPS
iptables -t filter -A INPUT -p tcp --dport 80 -j ACCEPT
iptables -t filter -A INPUT -p tcp --dport 443 -j ACCEPT
```

```
echo - Autoriser serveur Apache : [OK]

# Autoriser FTP
modprobe ip_conntrack
modprobe ip_conntrack_ftp
iptables -t filter -A INPUT -p tcp --dport 20 -j ACCEPT
iptables -t filter -A INPUT -p tcp --dport 21 -j ACCEPT
iptables -t filter -A INPUT -m state --state ESTABLISHED,RELATED -j ACCEPT
echo - Autoriser serveur FTP : [OK]

# Autoriser Mail
iptables -t filter -A INPUT -p tcp --dport 25 -j ACCEPT
iptables -t filter -A INPUT -p tcp --dport 110 -j ACCEPT
iptables -t filter -A INPUT -p tcp --dport 143 -j ACCEPT
iptables -t filter -A OUTPUT -p tcp --dport 25 -j ACCEPT
iptables -t filter -A OUTPUT -p tcp --dport 110 -j ACCEPT
iptables -t filter -A OUTPUT -p tcp --dport 143 -j ACCEPT
echo - Autoriser serveur Mail : [OK]

# Limiter le Syn-Flood à 1 seconde
iptables -A FORWARD -p tcp --syn -m limit --limit 1/second -j ACCEPT
iptables -A FORWARD -p udp -m limit --limit 1/second -j ACCEPT
echo - Limiter le Syn-Flood : [OK]

# IP à blacklister
# iptables -A INPUT -s ADRESSE_IP -j DROP
echo - Mise à jour des IP blacklistées : [OK]

# Bloquer le Spoofing
iptables -N SPOOFED
iptables -A SPOOFED -s 127.0.0.0/8 -j DROP
iptables -A SPOOFED -s 169.254.0.0/12 -j DROP
iptables -A SPOOFED -s 172.16.0.0/12 -j DROP
iptables -A SPOOFED -s 192.168.0.0/16 -j DROP
iptables -A SPOOFED -s 10.0.0.0/8 -j DROP
echo - Bloquer le Spoofing : [OK]

echo Firewall mis a jour avec succès !
```

2ème méthode : Installation des services persistantes

Pour rendre les services persistants, il nous suffit d'exécuter ces commandes depuis le prompt, et ensuite après avoir définit les règles que l'on souhaite, il faudrait **save** les arguments avant de redémarrer :

```
sudo apt-get install iptables-persistent

sudo service iptables-persistent
```

7- Fail2Ban

Fail2ban est un programme qui analyse les logs de connexions de divers services (SSH, Apache, FTP) en cherchant des correspondances entre des motifs définis dans ses filtres.

Typiquement, fail2ban cherche des tentatives répétées de connexions infructueuses et procède à un bannissement en ajoutant une règle au pare-feu Iptables.