Rapport TP4 : Configuration complète d'un serveur

DRISSI Mohamed Reda
reda-mohamed@isty.uvsq.fr

9 novembre 2018

Table des matières

I Intégration dans l'infrastructure

Pour obtenir la machine *server* nous allons simplement cloner la machine *client* puisque la configuration à faire pour se connecter au routeur sera la même.
Cependant il faudra :

- Changer le nom d'hôte.
- Nous changeons le fichier `/etc/network/interfaces` pour lui attribuer une adresse ip fixe. Nous allons choisir l'adresse "192.168.0.11" qui est en dehors de la plage prise par le serveur DHCP.
- Puis il faudra redémarrer les services réseaux avec la commande

```
$ systemctl restart networking
```

- Nous rajoutons dans /etc/hosts

```
192.168.0.11    server.istycorp.fr
```

- Nous redémarrons **dnsmasq**.

II Service SSH

II.1 Exercice 1

Nous rajoutons les nouvelles règles dans les paramètres de la VM :

Name	Protocol	Host IP	Host Port	Guest IP	Guest Port
Rule 1	TCP	127.0.0.1	9223	0.0.0.0	22
Rule 2	TCP	127.0.0.1	9224	0.0.0.0	2222

Le package **openssh-server** est déjà installé.
Nous rajoutons la nouvelle règle dans nos *iptables* :

```
$ iptables -t nat -A PREROUTING -p tcp -i enp0s3 --dport 2222
-j DNAT --to-destination 192.168.0.11:22
$ iptables-save > /etc/iptables/rules.v4
```

II.2 Exercice 2

Il faut créer un nouvel utilisateur **isty**. Puis luis créer une clé ssh et la copier vers la serveur, nous allons garder le nom de la clé par défaut : id_rsa.
Puis on se connecte par ssh à la VM serveur.

```
$ adduser isty
$ usermod -aG sudo isty
$ su isty
$ ssh-keygen
$ ssh-copy-id -p 9224 isty@localhost
$ ssh -p 9224 isty@localhost
```

II.3 Exercice 3

Pour plus de sécurité Nous allons modifier le sshd_config (fichier de configuration du serveur ssh). Nous allons interdire la connexion par mot de passe, et la connexion direct au compte root.
Nous pouvons aussi changer d'autres options comme le fichier contenant les clés publiques des clients autorisés, ou bien forward le serveur X vers le client, etc.
Nous décommentons puis modifions les lignes :

```
PasswordAuthentication no
PermitRootLogin no
$ systemctl restart ssh
```

II.4 Exercice 4

Nous allons installer fail2ban puis le tester, donc nous allons commenter la ligne `PasswordAuthentication` durant ce test, pour ensuite la remettre.

```
$ sudo aptitude install fail2ban
$ ssh -p 9224 adminsys@localhost
```

Nous remarquons qu'au bout du 5ème essai nous sommes banni pendant 10 minutes.

II.5 Exercice 5

La ligne suivante a été ajoutée :

```
-A f2b-sshd -s 10.0.2.2/32 -j REJECT --reject-with icmp-port-unreachable
```

— iptables permet de mettre en place des règles de traitement de paquets
— -A pour "add" qui signifie ajouter en anglais permet de rajouter une règle.
— f2b-sshd est le nom de la chaîne.

— 10.0.2.2 est l'adresse de l'hôte sur le subnet de VirtualBox. Elle est bannie puisque nous. avons tenté de nous connecter 4 fois sans succès.
— -j REJECT signifie que la règle demande de rejeter les paquets en provenance de cette adresse.
— –reject-with icmp-port-unreachable indique le type d'erreur à renvoyer.

II.6 Exercice 6

Nous allons examiner les fichiers de log pour voir les tentatives de connexion :

— /var/log/auth.log

```
Nov  5 18:35:48 server sshd[891]: Failed password for isty from 10.0.2.2 port 58374 ss
Nov  5 18:35:50 server sshd[891]: Failed password for isty from 10.0.2.2 port 58374 ss
Nov  5 18:35:50 server sshd[891]: Connection closed by 10.0.2.2 port 58374 [preauth]
Nov  5 18:36:52 server sshd[893]: Failed password for isty from 10.0.2.2 port 58375 ss
Nov  5 18:36:52 server sshd[893]: Failed password for isty from 10.0.2.2 port 58375 ss
Nov  5 18:36:52 server sshd[893]: Connection closed by 10.0.2.2 port 58375 [preauth]
Nov  5 18:36:54 server sshd[895]: Failed password for isty from 10.0.2.2 port 58376 ss
Nov  5 18:36:54 server sshd[895]: Failed password for isty from 10.0.2.2 port 58376 ss
Nov  5 18:36:54 server sshd[895]: Connection closed by 10.0.2.2 port 58376 [preauth]
```

— var/log/fail2ban.log

```
2018-11-05 18:35:48,868 fail2ban.filter         [830]: INFO    [sshd] Found 10.0.2.2
2018-11-05 18:35:50,329 fail2ban.filter         [830]: INFO    [sshd] Found 10.0.2.2
2018-11-05 18:36:52,537 fail2ban.filter         [830]: INFO    [sshd] Found 10.0.2.2
2018-11-05 18:36:52,702 fail2ban.filter         [830]: INFO    [sshd] Found 10.0.2.2
2018-11-05 18:36:54,187 fail2ban.filter         [830]: INFO    [sshd] Found 10.0.2.2
2018-11-05 18:36:54,353 fail2ban.filter         [830]: INFO    [sshd] Found 10.0.2.2
2018-11-05 18:36:54,371 fail2ban.actions        [830]: NOTICE  [sshd] Ban 10.0.2.2
2018-11-05 18:46:54,906 fail2ban.actions        [830]: NOTICE  [sshd] Unban 10.0.2.2
```

On remarque que seulement deux tentatives apparaissent dans les logs alors qu'on a trois chances d'entrer notre mot de passe à chaque commande ssh.

II.7 Exercice 7

— Supprimer les logs ne peut pas affecter les adresses ip bannies, et ceci est normal.
— Nous pouvons utiliser la commande suivante pour voir l'état des adresses bannies, puis les autoriser à se connecter à nouveau :

```
$ fail2ban-client status sshd
$ fail2ban-client set sshd unbanip 10.0.2.2
```

III Apache / php / MySQL / WordPress

III.1 Exercice 9

Nous allons installer apache php et mysql pour php, (aptitude trouvera le reste des packages dont nous avons besoin) puis rediriger le port 8080 de la machine hôte vers le port 80 de la machine client, ainsi nous pourrons accèder à notre serveur installé dans le client via le navigateur de l'hôte.

```
$ aptitude install apache2 php php-mysql php-pear
Add VM rule : TCP Rule 3 127.0.0.1  8080  0.0.0.0  80
$ iptables -t nat -A PREROUTING -p tcp -i enp0s3 --dport 80
-j DNAT --to-destination 192.168.0.11:80
$ iptables-save > /etc/iptables/rules.v4
```

Il est actuellement possible de voir la page d'acceuil par défaut d'apache sur le navigateur avec localhost :8080.
Nous pouvons ajouter une page d'infos php dans la racine du serveur apache (par défaut /var/www/).
Puis nous activons les modules apache nécessaires avec `aen2mod`.

```
$ echo -e "<?php\nphpinfo();\n?>" > /var/www/html/info.php
$ a2enmod mpm_prefork && a2enmod php7.0
```

Nous pouvons voir la page d'infos sur l'adresse `localhost:8080/info.php`

III.2 Exercice 9

Installation du package `mysql-server`

III.3 Exercice 10

Configuration de MySQL avec `mysql_secure_installation`

III.4 Exercice 11

Installation du package `phpmyAdmin` puis choisir les bonnes options.
Puis redémarrer le serveur apache et accèder à l'application web phpMyAdmin sur le lien :
`localhost:8080/phpmyadmin`

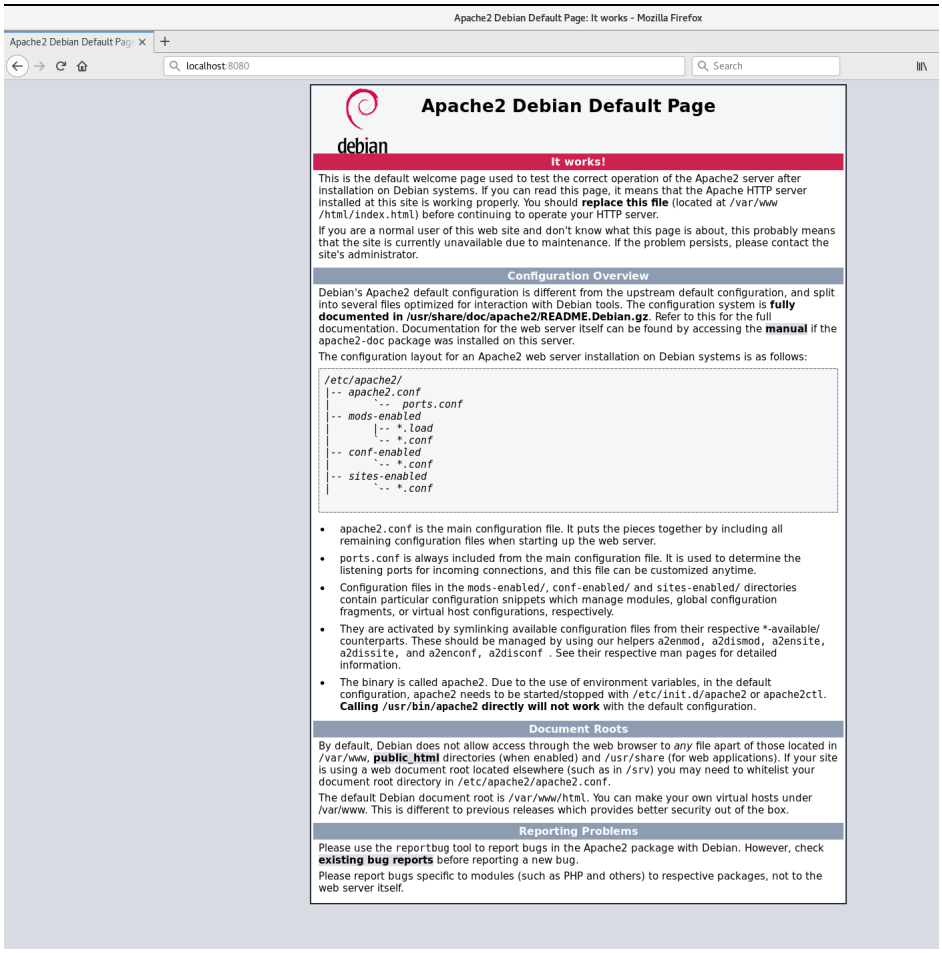

FIGURE 1 – Page d'acceuil "It Works" d'Apache

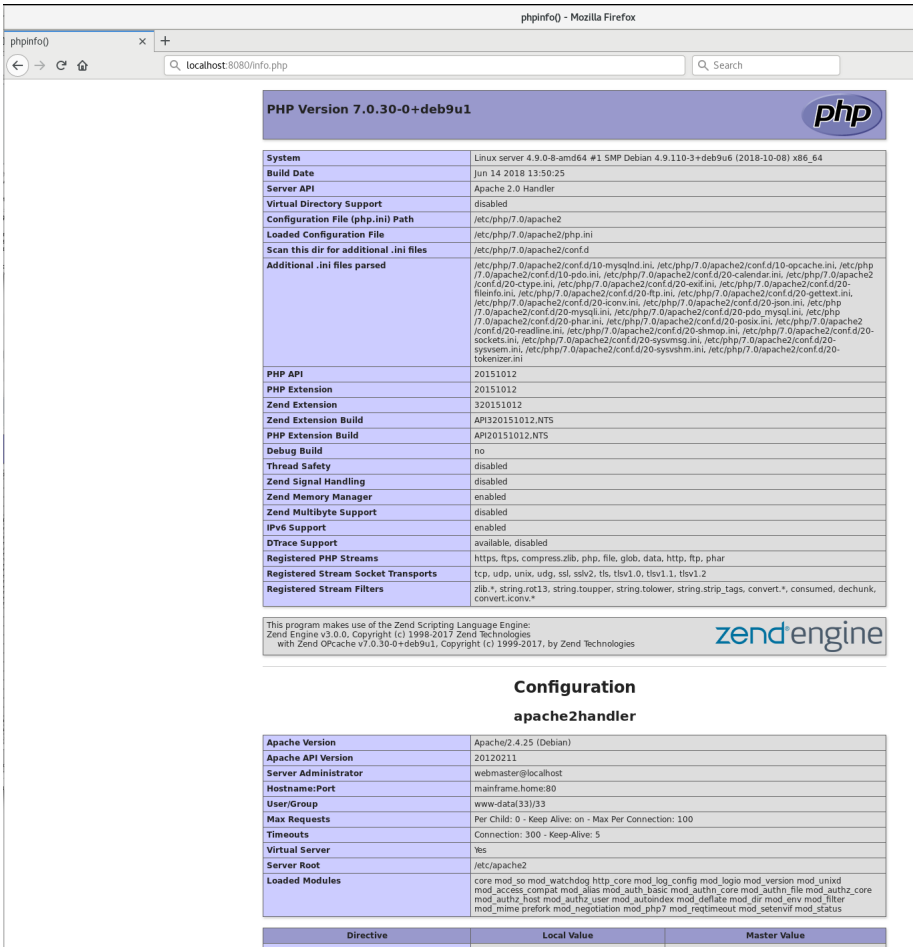

FIGURE 2 – Page d'infos sur la machine générée par PHP

III.5 Exercice 12

III.5.1 MySQL

Nous allons configurer la base de données mySQL (maintenant mariadb)

```
$ mariadb
#nouveau utilisateur wpuser
$ CREATE USER 'wpuser'@'localhost' IDENTIFIED BY 'wpuser';
#nouvelle db wpdb
$ CREATE DATABASE wpdb DEFAULT CHARACTER SET
'utf8' COLLATE 'utf8_unicode_ci';
#accorder toutes les permissions sur wpdb à wpuser
$ GRANT ALL ON `wpdb`.* TO `wpuser`@`localhost`;
#actualiser les tables de permissions
$ FLUSH PRIVILEGES;
$ exit
```

III.5.2 WordPress

Installation du package `wordpress`. Modification des fichiers `000-default.conf` et `default-ssl.conf` qui permettent de configurer les liens mis à disposition des clients par apache.
Nous rajoutons les lignes suivantes sous la directive `DocumentRoot /var/www/html` :

```
Alias /server  /usr/share/wordpress
<Directory /usr/share/wordpress>
    Options FollowSymLinks
    AllowOverride Limit Options FileInfo
    DirectoryIndex index.php
    Require all granted
</Directory>
<Directory /usr/share/wordpress/wp-content>
    Options FollowSymLinks
    Require all granted
</Directory>
-------------------------------------------------
Recharger apache avec :
$ systemctl reload apache2
```

Configurer le site pour utiliser MySQL/MariaDB dans /etc/wordpress/config-localhost.php :

```
<?php
define('DB_NAME', 'wpdb');
define('DB_USER', 'wpuser');
define('DB_PASSWORD', 'wpuser');
define('DB_HOST', 'localhost');
define('WP_CONTENT_DIR', '/usr/share/wordpress/wp-content');
?>
```

III.6 Exercice 13

Il faudra rajouter une nouvelle règle pour rediriger le port 443 afin d'utiliser https .
Puis utiliser iptables comme avant, et activer le module ssl d'apache.

```
$ Adding rule  Rule 4  TCP  127.0.0.1  9443  0.0.0.0  443
$ iptables -t nat -A PREROUTING -p tcp -i enp0s3 --dport 443
  -j DNAT --to-destination 192.168.0.11:443
$ iptables-save > /etc/iptables/rules.v4
$ a2ensite default-ssl
$ a2enmod ssl
$ systemctl restart apache2
```

Nous pouvons accèder à notre site d'acceuil avec https via : `https://localhost:9443`.
Cependant notre site wordpress ne fonctionne pas, il faudra changer les champs WordPress Address et Site Address en modifiant le port et en mettant *https* au lieu de *http* dans Settings > General.
Nous remarquons que l'émetteur du certificat SSL n'est pas reconnu.
Ceci peut-être corrigé en utilisant l'outil `openssl` ou bien si nous avons envie de rendre le site public, l'outil gratuit `letsencrypt`.

III.7 Exercice 14

Si un attaquant parvient à obtenir la clé privée de notre paire, il peut voler l'identité du site, puis voler les données des visiteurs du sites, et autres informations.

IV NFS

IV.1 Exercice 15

— Serveur

Nous installons les binaires du serveur NFS :

```
$ aptitude install nfs-kernel-server
```

Il faudra exporter le répertoire NFS. Ce dossier va être monté à distance, donc accessible depuis la machine client. Puis nous devrons rajouter une nouvelle entrée d'export NFS dans le fichier `/etc/exports`. Nous allons aussi rajouter un fichier afin de tester.
Cela va permettre à n'importe quelle adresse IP avec des droits de lecture/écriture d'accèder à ce répertoire. Puis nous appliquons ces changements. Nous pouvons voir quelques exemples avec la dernière commande.

```
$ mkdir /var/nfs-export
$ echo "testing nfs for sysTP4" >> /var/nfs-export/file
$ echo "/var/nfs-export *(rw,sync,no_subtree_check,\
no_root_squash)" >> /etc/exports
$ exportfs -a
$ man exports
```

Nous activons le démarrage automatique avec la commande :

```
$ systemctl enable nfs-kernel-server
```

— Client

Nous installons les binaires du client NFS

```
$ aptitude install nfs-common
```

Nous allons créer un point de montage de notre répertoire NFS, puis le monter.
Nous allons ensuite tester nos droits de lecture/écriture.

```
$ mkdir /mnt/nfs
$ mount -t nfs 192.168.0.11:/var/nfs-export /mnt/nfs/
$ cat /mnt/nfs/file
> testing nfs for sysTP4
$ touch /mnt/nfs/foo
$ ls /mnt/nfs
> file foo
```

Pour rendre ce montage automatique à chaque démarrage nous pouvons rajouter cette ligne dans le `/etc/fstab` :

```
192.168.0.11:/var/nfs-export /mnt/nfs/    nfs
```

IV.2 Exercice 16

Si une machine client est compromise, l'attaquant peut usurper l'identité de n'importe quel autre utilisateur, donc il aura accès libre au répertoire partagé.

V Jenkins

V.1 Exercice 17

Jenkins dépend de java donc il faut vérifier qu'il est présent.

```
$ java -version
$ wget -q -O - https://pkg.jenkins.io/debian/jenkins.io.key\
 | sudo apt-key add -
$ sh -c 'echo deb http://pkg.jenkins.io/debian-stable binary/ > \
/etc/apt/sources.list.d/jenkins.list'
$ aptitude update
$ aptitude install jenkins
```

V.2 Exercice 18

Puisque nous avons opté pour l'installation du .deb plutôt que l'installation manuelle, jenkins est installé sur `/var/lib/jenkins`