HEH.be
Sciences
et technologies

Avenue V. Maistriau 8a
B-7000 Mons
+32 (0)65 33 81 54
scitech-mons@heh.be

WWW.HEH.BE

Projet Linux

Brasseur Diego, Corsini Louliano

Année Académique 2022-2023

WALLONIE-BRUXELLES ENSEIGNEMENT
pôle hainuyer

1. Introduction

Le projet Linux présenté dans ce rapport consiste à configurer un serveur exécutant un système d'exploitation Linux. Ce projet a été réalisé par Brasseur Diego et Corsini Louliano durant l'année académique 2022-2023. L'objectif était de construire un serveur répondant à certains critères, notamment en termes de partitionnement, de services, de sécurité et de sauvegarde. Nous avons sélectionné plusieurs distributions et les avons installées sur la machine physique du

pour faciliter la configuration et les services réseau. En plus des configurations manuelles, nous avons également développé des scripts personnalisés pour automatiser certaines tâches, telles que : Scripts de sauvegarde et scripts de création d'utilisateurs avec un accès spécifique au FTP, au serveur SMB et à PHPMyAdmin, y compris les bases de données des sites et les hôtes virtuels. Ce rapport décrit les différentes phases, décisions et scénarios développés au cours de ce projet.

2. Présentation générale du projet

Cette semaine nous avons dû préparer un serveur utilisant un operating system Linux, le serveur devait correspondre à certains critères. Notamment des partitions prévue pour les différents services voulus, services que nous avons dû choisir pour faire en sorte d'avoir plusieurs serveurs de fichiers (dont deux publiques et deux privé), un serveur web permettant d'héberger différentes pages web, un serveur de base de donnée, un serveur de Système de nom de domaine qui nous permet de lier un nom et une ip, tout en ayant un accès ssh.

Mais il n'y a pas que l'installation et la configuration que nous avion du faire, il y a aussi un aspect sécurité dans ce projet, comme le partitionnement comme dit précédemment, l'installation et la configuration d'un antivirus, un système automatisé de backup à l'aide d'un script, la sécurisation de la connexion ssh à l'aide d'un système de clé combiné un framework de prévention contre les intrusions, ainsi qu'un firewall permettant de configurer les ports et services voulus.

En n'oubliant pas la création de scripts permettant de créer un utilisateur ainsi que ses accès à son serveur de fichier et son serveur web avec sa base de donnée.

3. Distribution et type d'installation

Peu de temps avant le projet, nous avons discuté des différentes options qui s'offraient à nous. De prime abord, nous avons été tenté par un Ubuntu serveur car c'est une distribution largement documentée et avec une grande communauté d'utilisateurs donc de nombreux forums et aide en tous genres sur internet. Mais par habitude nous sommes partie sur une distribution basée sur RHEL (RedHat Enterprise Linux), nous avions plusieurs choix face à nous tels qu'Alma Linux, RHEL, Rocky Linux ou encore d'autres. La balance a fini par pencher du côté de Rocky Linux pour son bon support communautaire. Nous sommes partie sur une base RHEL pour plusieurs raisons. Tout d'abord son installeur permet l'installation facile de certains services et version de la distribution très facilement. Il permet aussi à l'installation de configurer des profils de sécurité (que nous n'avons malheureusement pas activé par un manque de familiariser avec "fapolicyd"). ainsi que pour la multitude de documentation disponible.

L'installation s'est faite sur une machine physique pour éviter tout problème relié à l'utilisation de machine virtuelle, Diego étant assez familier avec l'installation sur des machines physiques il nous a été plus facile de procéder comme ceci. Il nous a donc été très facile de mettre les deux machines en réseau.Nos deux machines était en effet dans un sous réseau nous permettant de ne pas devoir configurer les services lors du changement de salle.

4. Plan de partitionnement

Notre fstab se présente de cette façon

```
1
2  #
3  # /etc/fstab
4  # Created by anaconda on Tue May  2 08:16:55 2023
5  #
6  # Accessible filesystems, by reference, are maintained under '/dev/disk/'.
7  # See man pages fstab(5), findfs(8), mount(8) and/or blkid(8) for more info.
8  #
9  # After editing this file, run 'systemctl daemon-reload' to update systemd
10 # units generated from this file.
11 #
12 /dev/mapper/rl-root     /                       xfs     defaults        0 0
13 /dev/mapper/rl-back     /back                   xfs     defaults,noauto,nosuid,noexec        0 0
14 /dev/mapper/rl-back_log /back/log               xfs     defaults,noauto,nosuid,noexec        0 0
15 UUID=7096dbb1-ddaa-48fd-8869-f927d98621df /boot                   xfs     defaults        0 0
16 /dev/mapper/rl-home     /home                   xfs     defaults,nosuid        0 0
17 /dev/mapper/rl-share    /share                  xfs     defaults,nosuid,noexec        0 0
18 /dev/mapper/rl-var_log  /var/log                xfs     defaults        0 0
19 /dev/mapper/rl-www      /www                    xfs     defaults,nosuid,noexec,gquota,uquota       0 0
20 /dev/mapper/rl-swap     none                    swap    defaults        0 0
21
```

Comme vous pouvez le constater, nous avons pris la décision d'utiliser le système de fichier Xfs et non pas Ext4 comme vu au cours. Plusieurs éléments font que nous avons choisis ce système de fichier à la place de Ext4. Xfs est meilleur sur une série de point, notamment une taille de partition et de fichier plus grande, des inodes allouées dynamiquement, plus d'espace pour les attributs étendus, un groupes d'allocation pour gérer les inodes et l'espace libre séparément et surtout un utilitaire pour les backup efficient.

En ce qui concerne les partitions en temps que tel, nous avons une configuration de "*base*" avec un "**/**", "**/boot**", "**/home**" et une swap. A côté de ces partitions de base nous pouvons y trouver le "**/back**" et "**/back/log**" qui nous sert à gérer toutes les backups (nous avons également une partition qui n'a pas de point de montage, celle ci est utilisé par timeshift pour backuper le système excluant le /www, le /share et le /home). Ensuite nous trouvons le "**/www**" qui sert au serveur web (c'est là où les virtualhosts d'apache vont chercher les fichiers à les afficher sur le web). Finalement nous avons le "**/share**" qui est utilisé pour les différents serveurs de fichiers.

Point de montage	type	taille	rôle
/back	XFS (RL)	50GB	backup de données utilisateur.
/back/log	XFS (RL)	20GB	backup des log
aucun	XFS (RL)	100GB	backup du système avec timeshift
/	XFS (RL)	70GB	racine
/share	XFS (RL)	10GB	partage publique
/www	XFS (RL)	10GB	site web
/var/log	XFS (RL)	10GB	log
/home	XFS (RL)	5GB	dossier utilisateur
aucun	SWAP	8GB	swap
aucun	LVM	183GB	espace LVM RL

5. Description des services

5.1. *SSHD*

Le service SSHD est le service permettant un accès ssh au serveur.

Une des premières choses que l'on a fait est de mettre en place le ssh pour pouvoir travailler en parallèle. Nous avons donc commencé par générer un clef ssh avec un passphrase sécurisé. Une fois la clef copier nous avons donc désactiver la connexion par mot de passe et sous les conseils de l'outil Lynis avons modifié quelque configuration listé ci dessous. Une bannière a également été configurée.

```
# /etc/ssh/sshd_config
# set different port
Port 2222
SyslogFacility AUTH # activate verbose logging
LogLevel VERBOSE
MaxAuthTries 6 # set max try to 6
MaxSessions 3 # set max 3 concurrent session
HostbasedAuthentication no
PasswordAuthentication no # disable password auth
AllowAgentForwarding no
AllowTcpForwarding no
X11Forwarding no
TCPKeepAlive no # disconnect when idle
```

il nous a fallu également spécifier a selinux notre port comme ci dessous:

```
semanage port -a -t ssh_port_t -p tcp 2222
```

ainsi que modifier les règles du firewall.

```
firewall-cmd --add-port=2222/tcp --permanent
firewall-cmd -remove-service=ssh -permanent
firewall-cmd -reload
```

5.2. *Apache*

Pour héberger des sites web, nous avons dû installer httpd. Ce service fonctionne sur le port 80 pour l'http et sur le port 443 pour l'https (que nous avons dû ouvrir dans firewalld). Ensuite, une fois le service activé nous avons dû mettre en place un système de virtualhost qui permet d'afficher plusieurs pages web simultanément. Dans ce virtualhost on retrouve le nom du serveur, un alias, le chemin vers le fichier index, l'emplacement des logs et d'autres options.

5.3. *PHPMyAdmin*

Pour PHPMyAdmin, nous avons utilisé le paquet dans la repositories EPEL et avons dû modifier cette ligne dans le fichier de configuration d'apache de phpMyAdmin pour avoir un accès via l'endpoint */phpMyAdmin* pour chaque nom de domaine:

```
<Directory /usr/share/phpMyAdmin/>
  AddDefaultCharset UTF-8

  Require all granted #before edit “denied”
</Directory>
```

5.4. *MariaDB*

Nous avons utilisé MariaDB plutôt que MySQL pour notre base de données. Nous avons donc fait la configuration initiale via *mysql_secure_install.* Pour chaque base de site web un nouvel utilisateur ayant accès à une base de données du nom de son domaine est créé.

5.5. *VSFTPD*

Pour ce qui est du FTP nous l'avons configuré en suivant le tutoriel Secure FTP Server . Ce serveur ftp devait nous permettre d'avoir un accès distant sécurisé au serveur pour chaque hébergement web.

5.6. *CHRONY*

Chrony est un simple serveur de temps qui permet de donner l'heure aux machines du réseau et ainsi synchroniser nos services.

5.7. *SAMBA*

Ce service de partage de fichier nous a permis d'effectuer un partage public de */share* mais sera également utilisé pour partager le dossier stockage des site web dans */www/<domain>/* .

la configuration globale contient plusieur point intéressant ci dessous:

```
[global]
security = user
map to guest = bad user
# require for cockpit to manage share
include = registry


# Disallow privileged accounts
invalid users = root @wheel bin daemon adm sync shutdown halt mail news uucp

# set SMB client to use SMB 3 as a minimum version
server signing = mandatory
server min protocol = SMB3

# Use PAM if available
pam password change = yes

[Public]
# set user as nobody
guest ok = yes
guest only = yes
force user = nobody
force group = nobody
# activate optional module full_audit for advanced logging, fruit for macos
# support , catia for fixing special character and streams_xattr enables
# storing of NTFS  alternate data streams
vfs objects = full_audit catia fruit streams_xattr
full_audit:priority = notice
full_audit:facility = local5
full_audit:success = connect disconnect mkdir rmdir read write rename
full_audit:failure = connect
full_audit:prefix = %u|%I|%S
fruit:encoding = native
```

```
fruit:metadata = stream
fruit:zero_file_id = yes
fruit:nfs_aces = no
```

il a également fallu configurer SELINUX pour samba comme ci dessous

```
chcon -t samba_share_t /share
```

il a ensuite fallu configurer le firewall :

```
firewall-cmd --add-service=samba -permanent
firewall-cmd -reload
```

5.8. *BIND*

Nous avons également mis en place un serveur DNS en utilisant BIND.
Nous avons donc créé une zone inverse dans /etc/named.conf ainsi que modifier les options ci dessous

```
// /etc/named.conf
options {
   listen-on port 53 { any; };
   listen-on-v6 port 53 { any; };
   allow-query     { any; };
};

zone "0.42.10.in-addr.arpa" IN {
   type master;
   file "/etc/named/reverse";
   allow-update { none; };
};
```

```
; /etc/named/reverse
$TTL  86400
@     IN    SOA   random.lan. www.random.lan. (
                  2023050151        ;Serial
                  3600              ;Refresh
                  1800              ;Retry
                  604800            ;Expire
                  86400             ;Minimum TTL
              )
      IN    NS    ns1.random.lan.
      IN    NS    ns2.random.lan.
```

Pour ce qui est des zones forward sont ajoutées via le script d'ajout de site.

5.9. *NFS*

Grâce aux partage nfs toutes personnes dans le réseau (ayant l'ip du partage nfs), peut accéder en lecture et en écriture à cet espace de stockage. Bien évidemment, il n'est pas possible de se connecter se connecter en root.

6. Scripting

6.1. *add_user.sh*

Tout d'abord le script attend que l'administrateur entre un subdomain et un mot de passe. Une fois le subdomain et le mot de passe est acquis . Ce script commence par vérifier si l'utilisateur a les droits requis et sinon prompt en utilisant *sudo -v*.le script utilisant sudo pour ces actions va mettre en cache l'accès au droit root tout au long du script. Pour chaque nom de domaine un utilisateur sera créé puis il créera une zone forward dans le DNS pour mettre en place le nom de domaine ainsi qu'ajoute ce domaine a la zone reverse. une fois fait le script met en service le DNS puis configure samba sur le dossier correspondant au site dans */www/<domain>/ .* Une fois effectuer le script configure apache ainsi que mariaDB et aura ainsi donc accès à sa DB via *<domain>/phpMyAdmin*.

6.2. *remove_user.sh*

Tout d'abord le script attend que l'administrateur entre un subdomain et un mot de passe. Une fois le subdomain et le mot de passe est acquis . Ce script commence comme le script ci-dessus par vérifier les droits de l'utilisateur. Ensuite il retire les diverses configuration effectuées pour ce sous domaine.

6.3. *back_incr_log.sh et back_incr_data.sh*

le script commence par monter la partition de backup associer au type de donnée rl-back pour back_incr_data.sh et rl-back_log pour back_incr_log.sh
une fois monté on utilise rsync pour copier les données plusieur argument sont spécifier pour pouvoir créer des backups incrémentales tout d'abord -a permettant d'être en mode archive, -v pour être en mode verbose , -z pour activer la compression, -h pour que les logs lisible par un humain, –link-dest pour lier à une backup précédente (a ohmmètre si la backup était full), –delete pour supprimer les fichier qui existerait dans le dossier de destination mais pas d'envois.

Un lien symbolique est également fait sur la dernière backup pour une plus grande facilité. Un total de 30 backup de log est gardé et un total de 15 backup de donnés est stocker au maximum.

Une fois le backup fini la partition de backup est alors démonter.

7. Plan de sauvegarde

Le plan de sauvegarde se divise en 3 partie distincte :

- les données utilisateurs (/www , /share , /home)
- les logs (/var/log)
- le système

les backups des données utilisateur et des logs sont fait via 2 script montant la partition de backups correspondante et en utilisant rsync pour y faire un backup incrémental ces script sont lancé via con tout les jours à 4h et 4h30 respectivement.

Pour les backups du système nous avons décidé d'utiliser timeshift pour une utilisation plus simple lors de ce projet.

Idéalement un full backup serait fait toute les semaines en offsite. pour ce projet nous nous somme permet de ne pas faire le script permettant de faire les full backups en offsite. Néanmoins une des approches possible serait d'utiliser rsync pour copier les données via la fonction de transfert ssh de rsync. nous permettant de suivre la règle du 3-2-1.

- 3 copies des données (l'original et deux copies des données)
- 2 moyens de stockages séparés
- 1 copies off site (dans un lieux de stockage externe)

8. Sécurité du serveur

8.1. *ClamAV*

Il nous a été demandé d'avoir un antivirus sur notre serveur, notre choix s'est dirigé vers ClamAV qui est assez renommé. Nous avons donc setup cet antivirus de manière à ce qu'il analyse les fichiers de la machine ponctuellement à l'aide d'un *crontab -e*. Mais, nous n'avons pas arrêté là, nous avons créé un service nommé freshclam servant à mettre à jour la base virale. Nous avons fait en sorte que le service check les fichiers à chaque modification de ceux-ci. En bonus nous avons resize la taille des fichiers maximum que ClamAV analysait. et nous avons également activer l'analyse en temps réel.

il nous a fallu configurer selinux pour autoriser ClamAV à effectuer des scan du système .

```
sudo setsebool -P antivirus_can_scan_system 1
```

8.2. *fail2ban*

Pour prévenir des intrusions sur notre serveur nous avons décidé d'installer le service fail2ban. Fail2ban est un service qui analyse les logs des services qu'il surveille (à activer dans le /etc//fail2ban/jail.conf) et il effectue une ou plusieurs actions. Dans notre cas on surveillait les intrusions sur le ssh et si fail2ban observait 5 essais infructueux à la suite, il bloque via le notre firewall (Firewalld) l'ip de l'intru pendant 1h.

9. Problèmes rencontrés

Pendant, nous avons de nombreux problèmes. L'organisation est la chose qui nous a posé le plus de problèmes, le manque d'assiduité sur une tâche précise a provoqué la création de plusieurs bogues et donc nous a fait perdre un bon nombre de temps.

A l'activation de fail2ban, le service à banni l'ordinateur de diego sur le serveur apache, ce qui nous a valu un bon bout de temps avant de trouver l'erreur.

Vers la fin du projet, à des moments aléatoires, le ping entre le serveur et l'ordinateur de diego passait de 0,1 ms à 50 000 ms et aucune raison apparente n'a pu être trouvée, ce qui provoquaient des déconnexions et une perte de temps.

Nous avions butté sur l'application de quota aux utilisateurs par le moyen de script alors que la même commande entrée dans le terminal s'exécute parfaitement.

Le serveur ftp nous donnait un accès à l'arborescence de fichier mais nous ne pouvions pas télécharger ni uploader de fichier.

l'utilisation de sed dans nos script de suppression de site web nous a également poser de gros problème empêchant le script de fonctionner.

Lors de nos phases de test, nous avons eu des problèmes sur les test de l'http car le navigateur de diego forcait l'https et vu que nous n'avons que l'http (et pas de redirection vers l'http), il fallut désactiver la redirection automatique http vers https du navigateurs et ajouter les dns personnalisés de ce même navigateur .

10. Amélioration et Conclusion

10.1. *Conclusion*

Dans l'ensemble, ce projet Linux nous a permis d'approfondir nos connaissances en configuration de serveur et en administration système. Nous avons pu configurer divers services requis tels que serveur de fichiers, serveur Web, serveur de base de données, serveur DNS avec les distributions Linux respectives. Nous avons également développé des scripts personnalisés qui ont grandement simplifié la configuration et automatiser certaines tâches.

Ces scripts nous ont permis de mettre en place des sauvegardes automatiques régulières et de créer des utilisateurs avec un accès spécifique aux serveurs FTP, SMB et PHPMyAdmin, incluant la base de données et l'hébergeur virtuel dédié. Malgré les défis que nous avons rencontrés lors du développement et du débogage des scripts, nous avons pu les terminer avec succès, augmentant ainsi les performances et la cohérence de notre système.

Ce projet nous a donné une occasion précieuse d'améliorer nos compétences en rédaction de scripts et de les appliquer dans un environnement réel. Nous sommes convaincus que ces compétences seront d'une grande utilité dans notre futur travail professionnel. Nous sommes fiers des résultats et de la qualité de notre serveur Linux personnalisé.

10.2. *Piste Amélioration*

10.2.1. Apache

- désactivation des identifiants de versions (non effectuer dû à un oubli)

```
#/etc/httpd/conf/httpd.conf
ServerTokens Prod
ServerSignature Off
```

- installation et configuration de **Mod_security** et **Mod_evasive**
- configuration du HTTPS pour l'endpoint /phpMyAdmin
- automatisation de la configuration de HTTPS pour les nouveaux site créé via le site donnant un certificat ssl au nouvel utilisateur

10.2.2. BIND

- utilisation de dnssec pour chaque site web et leurs dns (une partie non tester dans le script est présent en commentaire)
- configurer et vérifier le dns pour fonctionner correctement avec samba et Active Directory

10.2.3. Installation

- une plus grande isolation des partitions aurait pu être faite. (*/var* dans sa propre partition, etc)

10.2.4. Sécurité

- mise en place d'une liste blanche de logiciel en utilisant Fapolicyd

10.2.5. Organisation

- Manque de rigueur dans l'organisation du progrès. Nous avons été un peu trop laxiste dans la façon de faire une tâche (tout en négligeant pas la qualité mais la duré à effectuer la tâche)

11. Bibliographie

Linda. (2022). XFS vs Ext4 : Which One Is Better ? *MiniTool*.

https://www.partitionwizard.com/partitionmanager/xfs-vs-ext4.html

Shrivastava, T. (2016, 14 novembre). *13 Apache Web Server Security and Hardening Tips*. 13 Apache Web Server Security and Hardening Tips.

https://www.tecmint.com/apache-security-tips/

Spencer, S. (s. d.). *Apache Multisite - Documentation*.

https://docs.rockylinux.org/guides/web/apache-sites-enabled/

Spencer, S. (s. d.-a). *Apache Hardened Webserver - Documentation*.

https://docs.rockylinux.org/guides/web/apache_hardened_webserver/

mysql_secure_installation. (s. d.). MariaDB KnowledgeBase.

https://mariadb.com/kb/en/mysql_secure_installation/

vfs_streams_xattr. (s. d.).

https://www.samba.org/samba/docs/current/man-html/vfs_streams_xattr.8.html

vfs_fruit. (s. d.). https://www.samba.org/samba/docs/current/man-html/vfs_fruit.8.html

vfs_catia. (s. d.).

https://www.samba.org/samba/docs/current/man-html/vfs_catia.8.html

vfs_full_audit. (s. d.).

https://www.samba.org/samba/docs/current/man-html/vfs_full_audit.8.html

Maurya, H. (2023). How to Install ClamAV on Rocky Linux | AlmaLinux 9 or 8. *Linux Shout*. https://linux.how2shout.com/install-clamav-on-rocky-linux-8-almalinux/

12. Annexe

voir fichier joint