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 AWS

Rapport Final

Année Académique
2024 - 2025

Auteurs
Adam Bougma
Samet Catakli

HEH be Sciences et technologies
FÉDÉRATION WALLONIE-BRUXELLES
pôle hainuyer

Table des matières

Table des configurations

Table des figures

1 Introduction

Dans le cadre du projet Linux AWS, nous avons mis en place une infrastructure sécurisée et complète afin de proposer un service d'hébergement web multi-utilisateur. Ce rapport technique présente les choix technologiques, les configurations effectuées ainsi que les mesures de sécurité mises en uvre pour assurer un service fiable et sécurisé.

2 Plan de l'infrastructure

2.1 Architecture initiale prévue

Notre conception initiale prévoyait une infrastructure modulaire composée de plusieurs serveurs spécialisés :

- **Serveur Réseau** : Dédié aux services DNS (BIND9) et synchronisation temporelle (NTP), assurant ainsi les fonctions fondamentales d'infrastructure.
- **Serveur Web** : Hébergeant Apache, PHP, MariaDB et les services de partage de fichiers (Samba, FTP, NFS).
- **Serveur de Monitoring** : Dédié à la surveillance de l'infrastructure via Netdata.
- **Solution de Sauvegarde** : Initialement prévue sur AWS S3/Glacier pour stocker les sauvegardes de manière économique et durable.

2.2 Architecture effective déployée

En raison des contraintes de temps et des difficultés rencontrées avec l'API AWS pour accéder à S3, nous avons adapté notre infrastructure :

- **Instance principale** (10.42.0.109) : Regroupant les services réseau (DNS, NTP) et les services web (Apache, PHP, MariaDB, partage de fichiers).
- **Instance secondaire** (10.42.0.129) : Combinant les fonctions de monitoring (Netdata) et servant de destination pour les sauvegardes via SCP.

Les deux instances sont déployées dans un VPC (Virtual Private Cloud) sur le réseau 10.42.0.0/24, accéssible uniquement via vpn, offrant une isolation réseau et une communication sécurisée entre les composants de l'infrastructure.

Cette architecture simplifiée nous a permis de fournir tous les services requis tout en contournant les problèmes d'accès à S3, offrant une solution robuste qui respecte les exigences du projet.

3 Choix technologiques

3.1 Distribution Linux

Dans le cadre de ce projet, la distribution imposée était `Amazon Linux 2023` (al2023), développée par Amazon pour une intégration optimale avec l'infrastructure AWS. Elle repose sur un noyau léger, sécurisé et optimisé pour l'environnement AWS, offrant ainsi des performances élevées tout en garantissant un haut niveau de sécurité. Par ailleurs, cette distribution est volontairement minimaliste : très peu de paquets sont installés par défaut, ce qui permet de limiter la surface d'attaque en réduisant l'exposition à d'éventuelles vulnérabilités liées à des services non nécessaires. Basée sur RHEL, elle supporte aussi nativement SELINUX, qui est un module critique pour la sécurité renforcée.

3.2 Méthode d'installation

Le déploiement des instances a été réalisé de manière automatisée via la console AWS, permettant une mise en service rapide et standardisée. Une phase de configuration manuelle complémentaire a ensuite été effectuée, soit par accès SSH depuis un terminal, soit directement via l'interface de la console AWS, afin d'adapter finement les paramètres système et de sécurité aux besoins spécifiques du projet.

4 Plan de partitionnement

Les technologies LVM et RAID ont été utilisées pour la gestion des disques et le partitionnement. Bien qu'un RAID5 soit mis en place par défaut sur AWS, nous avons configuré un RAID1 avec les disques `/dev/nvme1n1` et `/dev/nvme2n1` attachés à l'instance et montés sur `/srv`. Un partitionnement LVM a été configuré sur ce RAID pour bénéficier des fonctionnalités de gestion avancées, telles que les snapshots et le redimensionnement à chaud.

Le partitionnement choisi permet d'optimiser sécurité, performance et gestion des données :

- **/** : système d'exploitation, configurations globales.
- **/boot** : fichiers de démarrage, noyaux Linux.
- **/tmp** : pour les fichiers temporaires.
- **/etc** : configurations spécifiques des services.
- **/var** : fichiers dynamiques tels que logs système et fichiers temporaires d'applications.
- **/home** : espaces utilisateurs. (très peu utilisé dans notre système)
- **/srv** : structuré spécifiquement en RAID1 et LVM, contenant :
  - `/srv/web` : fichiers des sites web hébergés (avec partage ftp/samba privé) .
  - `/srv/database` : stockage des bases de données (mariadb datadir).
  - `/srv/share` : données partagées non privé(NFS, Samba, FTP).

5 Description des services déployés

5.1 Services réseau

Serveur DNS (BIND9) Le serveur DNS a été mis en place à l'aide de BIND9, en s'appuyant sur le service named pour assurer la résolution des noms de domaine. Il prend en charge à la fois la résolution directe (DNS) et inverse (reverse DNS), ce qui permet d'héberger une zone DNS locale ainsi que les enregistrements associés aux noms de domaine utilisés par les clients du réseau. Pour garantir l'accès aux domaines externes notamment pour les mises à jour logicielles ou les accès internet un mécanisme de forwarding a été configuré : les requêtes en dehors de la zone locale sont redirigées vers les serveurs DNS publics de Cloudflare. Cette approche assure à la fois une résolution rapide, fiable et sécurisée des noms externes, tout en gardant un contrôle total sur la zone DNS interne.

```
options {
    listen-on port 53 { any; };
    directory "/var/named";
    recursion yes;
    allow-query { any; };
    allow-transfer { none; };
    dnssec-validation auto;
};

zone "." IN {
    type hint;
    file "named.ca";
};

zone "heh.cloud" IN {
    type master;
    file "forward.heh.cloud";
    allow-update { none; };
};

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

Configuration 1 – Configuration DNS BIND9 /etc/named.conf

```
$TTL 86400
@   IN  SOA     ns.heh.cloud. root.heh.cloud. (
              2024052101 ; Serial
              3600       ; Refresh
              1800       ; Retry
              604800     ; Expire
              86400 )    ; Minimum TTL

@       IN  NS      ns.heh.cloud.
ns      IN  A       10.42.0.109
@       IN  A       10.42.0.109
*       IN  A       10.42.0.109
```

Configuration 2 – Configuration DNS BIND9 /var/named/forward.heh.cloud

Des mesures de sécurité spécifiques ont été appliquées :

— Limitation des transferts de zone (`allow-transfer { none; }`)

— Configuration du service pour écouter uniquement sur les interfaces nécessaires

— Utilisation de `dnssec-validation auto` pour renforcer la sécurité des réponses DNS

Serveur NTP (Chrony) Chrony a été configuré pour assurer une synchronisation précise de l'horloge système sur le fuseau horaire `Europe/Bruxelles`. Dans cette infrastructure, un serveur NTP interne a été mis en place pour servir de référence temporelle principale. Les autres machines du réseau sont configurées pour se synchroniser avec ce serveur, ce qui garantit une cohérence horaire uniforme à l'échelle de l'ensemble du système. Ce choix permet non seulement de limiter la dépendance à des serveurs externes, mais aussi de réduire la latence et d'améliorer la sécurité des échanges temporels au sein du réseau local.

```
# Server directives specify NTP servers to use
server 0.pool.ntp.org iburst
server 1.pool.ntp.org iburst
server 2.pool.ntp.org iburst
server 3.pool.ntp.org iburst

# Allow NTP client access from local network
allow 10.42.0.0/24

# Record the rate at which the system clock gains/losses time
driftfile /var/lib/chrony/drift
```

Configuration 3 – Configuration NTP Chrony /etc/chrony.conf

5.2 Partage de fichiers

Samba Samba a été mis en place pour offrir un service de partage de fichiers sécurisé entre systèmes Windows et Linux. Dans la configuration, on trouve à la fois des répertoires publics, accessibles à tous les utilisateurs, et des répertoires privés, protégés par des droits d'accès individuels. Des comptes Samba spécifiques, associés à des utilisateurs du système, assurent une gestion fine des permissions, ce qui permet un contrôle complet des lectures, écritures et modifications des fichiers partagés.

```
[unauth_share]
   path = /srv/share/
   browsable = yes
   writable = yes
   guest ok = yes
   guest only = yes
   force user = nobody
   force group = nobody
   create mask = 0777
   directory mask = 0777
   read only = no
```

Configuration 4 – Configuration Samba /etc/samba/smb.unauth.conf

```
[global]
    workgroup = SAMBA
    security = user
    passdb backend = tdbsam
    include = /etc/samba/usershares.conf

[username]
    path = /srv/web/username
    valid users = username
    read only = no
```

Configuration 5 – Configuration Samba /etc/samba/smb.conf

FTP Pour garantir des transferts de fichiers sécurisés, le serveur FTP a été configuré en utilisant le chiffrement TLS (FTPS). Ce chiffrement assure la confidentialité et l'intégrité des données échangées entre le client et le serveur, en protégeant les identifiants et les fichiers contre toute interception. Les ressources sont accessibles uniquement aux utilisateurs authentifiés, et les droits d'accès sont strictement définis pour chaque utilisateur lié aux comptes système. Cette solution garantit la compatibilité avec les clients FTP traditionnels tout en répondant aux exigences de sécurité actuelles.

```
anonymous_enable=YES
local_enable=YES
write_enable=YES
anon_upload_enable=YES
anon_mkdir_write_enable=YES

dirmessage_enable=YES
xferlog_enable=YES
connect_from_port_20=YES
xferlog_std_format=YES
listen=YES
listen_ipv6=NO

pam_service_name=vsftpd
userlist_enable=YES
anon_root=/srv/share/ftp
anon_umask=022

pasv_enable=YES
pasv_min_port=30000
pasv_max_port=31000
```

Configuration 6 – Configuration FTP vsftpd /etc/vsftpd/vsftpd.conf

NFS (public uniquement) NFS, un système de partage de fichiers disponible uniquement sur Linux, est utilisé uniquement pour le partage public de fichiers. Disposant uniquement de contrôle d'accès basé sur les adresses IP (ACL), il ne convient pas aux partages privés nécessitant une authentification. Il est donc limité dans ce projet aux répertoires accessibles à tous les utilisateurs du réseau local.

```
/srv/share *(rw,sync,no_root_squash)
```

Configuration 7 – Configuration NFS /etc/exports

5.3 Serveur web (Apache et PHP)

Le serveur web Apache a été choisi pour sa popularité, sa robustesse et sa flexibilité (modules). Couplée au langage PHP, elle permet d'héberger des sites dynamiques avec code backend.

```
<VirtualHost *:80>
    ServerName heh.cloud
    ServerAlias *.heh.cloud
    DocumentRoot /srv/web/root
    <Directory /srv/web/root>
        AllowOverride All
        Require all granted
    </Directory>
    DirectoryIndex index.php

    # Journalisation
    ErrorLog /var/log/httpd/root_error.log
    CustomLog /var/log/httpd/root_access.log combined

    # Redirection vers HTTPS
    Redirect "/" "https://heh.cloud/"
</VirtualHost>
```

Configuration 8 – Configuration Apache /etc/httpd/conf.d/heh.cloud.conf

```
<VirtualHost *:443>
    ServerName heh.cloud
    ServerAlias *.heh.cloud
    DocumentRoot /srv/web/root

    <Directory /srv/web/root>
        AllowOverride All
        Require all granted
    </Directory>

    DirectoryIndex index.php

    # Configuration SSL
    SSLEngine on
    SSLCertificateFile /etc/httpd/ssl/heh.cloud.crt
    SSLCertificateKeyFile /etc/httpd/ssl/heh.cloud.key

    # Journalisation
    ErrorLog /var/log/httpd/root_ssl_error.log
    CustomLog /var/log/httpd/root_ssl_access.log combined
</VirtualHost>
```

Configuration 9 – Configuration Apache SSL /etc/httpd/conf.d/heh.cloud-ssl.conf

MariaDB, un fork de MySQL offrant davantage de fonctionnalités et une meilleure performance dans certains cas, a été choisie comme système de gestion de base de données. Elle a été configurée pour stocker ses fichiers de données sur la partition en RAID 1/LVM `/srv/database`, garantissant ainsi une redondance et une meilleure tolérance aux pannes.

```
[mysqld]
datadir=/srv/database
socket=/var/lib/mysql/mysql.sock

[client]
socket=/var/lib/mysql/mysql.sock
```

Configuration 10 – Configuration MariaDB /etc/my.cnf.d/server.cnf

Phpmyadmin, une application web qui permet une gestion simplifiée des bases de données, a été choisie et mise à la disposition des utilisateurs pour gérer leurs bases de données. Les paramètres suivants ont été mis en place :

```
<VirtualHost *:443>
    ServerName phpmyadmin.heh.cloud
    DocumentRoot /srv/web/phpmyadmin

    <Directory /srv/web/phpmyadmin/>
        AddDefaultCharset UTF-8
        Options FollowSymLinks
        AllowOverride All
        Require all granted

        # Block root user access
        <IfModule mod_rewrite.c>
            RewriteEngine On
            RewriteCond %{REQUEST_URI} ^/.*
            RewriteCond %{REQUEST_METHOD} ^POST$
            RewriteCond %{REQUEST_URI} !server-status
            RewriteCond %{THE_REQUEST} pma_username=root [NC]
            RewriteRule .* - [F,L]
        </IfModule>
    </Directory>

    <Directory /srv/web/phpmyadmin/libraries/>
        Require all denied
    </Directory>

    SSLEngine on
    SSLCertificateFile /etc/httpd/ssl/heh.cloud.crt
    SSLCertificateKeyFile /etc/httpd/ssl/heh.cloud.key
</VirtualHost>
```

Configuration 11 – Configuration phpMyAdmin /etc/httpd/conf.d/phpmyadmin.conf

```
<?php
// Secret key pour l'authentification par cookie
$cfg['blowfish_secret'] = 'BLOWFISH_SECRET';

$i = 0;
$i++;
$cfg['Servers'][$i]['auth_type'] = 'cookie';
$cfg['Servers'][$i]['host'] = 'localhost';
$cfg['Servers'][$i]['socket'] = '/var/lib/mysql/mysql.sock';
$cfg['Servers'][$i]['compress'] = false;
$cfg['Servers'][$i]['AllowNoPassword'] = false;
$cfg['Servers'][$i]['AllowRoot'] = false; // Désactivation de l'accès root

$cfg['UploadDir'] = '';
$cfg['SaveDir'] = '';
$cfg['SendErrorReports'] = 'never';
?>
```

Configuration 12 – Configuration phpMyAdmin /srv/web/phpmyadmin/config.inc.php

5.4 Monitoring

L'application `Netdata` est utilisée pour surveiller en temps réel les performances et l'état de santé du système ainsi que des différents services. Elle permet notamment de surveiller les ressources du serveur, le trafic réseau et http, et l'état de santé des différents services configurés tels que le serveur web, Samba, MariaDB, DNS, etc. Un dashboard personnalisé et des alertes ont été configurés pour une meilleure prise en main et des interventions rapides.

FIGURE 1 – Dashboard Netdata montrant l'état de santé du système et des services.

```
[stream]
  enabled = yes
  api key = API_KEY
  default memory mode = ram
  health enabled by default = auto
  allow from = *
```

Configuration 13 – Configuration Netdata Serveur /etc/netdata/stream.conf

```
[stream]
  enabled = yes
  destination = 10.42.0.129:19999
  api key = API_KEY
```

Configuration 14 – Configuration Netdata Client /etc/netdata/stream.conf

6 Plan de sauvegarde

Un plan de sauvegarde robuste vers S3 Glacier avait initialement été conçu pour effectuer des sauvegardes économiques vers un bucket AWS. Une sauvegarde incrémentielle quotidienne et une complète hebdomadaire avaient été pensées. Cependant, en raison de problèmes d'accès avec les clés API refusées, nous avons opté pour une solution alternative avec des sauvegardes envoyées vers la partition RAID/LVM `/srv/backup` de l'instance secondaire sur laquelle se trouve l'application de monitoring.

Voici une liste des partitions sauvegardées :

- **/etc** : configurations spécifiques des services.
- **/var** : fichiers dynamiques tels que logs système.
- **/home** : espaces utilisateurs. (très peu utilisé dans notre système)
- `/srv/web` : fichiers des sites web hébergés.
- `/srv/database` : fichiers de base de données.
- `/srv/share` : données partagées non privé (NFS, Samba, FTP).

```
0 2 * * * /path/to/backup.sh >> /var/log/backup.log 2>&1
```

Configuration 15 – Configuration Sauvegarde /etc/crontab

Cette tâche cron exécute quotidiennement à 2h du matin un script de sauvegarde qui transfère les données critiques vers l'instance de backup via rsync et SSH.

7 Sécurité avancée du serveur

7.1 Configuration SSH

Afin de garantir un accès sécurisé au serveur, une configuration SSH renforcée a été mise en uvre. La connexion directe en tant qu'utilisateur root a été désactivée, tout comme l'authentification par mot de passe, afin de réduire les risques liés aux attaques par force brute ou à l'utilisation de mots de passe faibles. Seules les connexions via des paires de clés SSH sont autorisées, assurant ainsi une méthode d'authentification plus robuste. De plus, l'activation des journaux de connexion permet un suivi précis des accès, tandis que l'intégration de Fail2Ban contribue à bloquer automatiquement les tentatives suspectes, ajoutant une couche supplémentaire de protection à l'interface SSH.

```
# Authentification par clé uniquement
PasswordAuthentication no

# Blocage de l'accès root
PermitRootLogin no

# Journalisation avancée
LogLevel VERBOSE

# Autres options de sécurité
PubkeyAuthentication yes
Protocol 2
X11Forwarding no
```

Configuration 16 – Configuration SSH /etc/ssh/sshd_config

7.2 Firewalls

Deux niveaux de protection par pare-feu ont été déployés afin de sécuriser les communications réseau tout en optimisant les performances du système. Les `Security Groups` d'AWS agissent comme un pare-feu périmétrique en filtrant le trafic au niveau de l'infrastructure cloud, ce qui permet de bloquer les connexions non autorisées avant même qu'elles n'atteignent l'instance. Ce filtrage en amont contribue à éviter une surcharge inutile des ressources du serveur, en empêchant le traitement de requêtes malveillantes ou superflues. En complément, `firewalld` est utilisé au niveau de l'instance pour affiner la politique de sécurité interne, assurant un contrôle granulaire des flux autorisés selon les services et les zones définies. Cette approche à deux niveaux renforce à la fois la sécurité et l'efficacité de la gestion du trafic réseau.

```
# Configuration des zones
firewall-cmd --set-default-zone=public

# Services autorisés
firewall-cmd --permanent --zone=public --add-service=ssh
firewall-cmd --permanent --zone=public --add-service=http
firewall-cmd --permanent --zone=public --add-service=https
firewall-cmd --permanent --zone=public --add-service=dns
firewall-cmd --permanent --zone=public --add-service=ftp
firewall-cmd --permanent --zone=public --add-service=mysql
firewall-cmd --permanent --zone=public --add-service=samba
firewall-cmd --permanent --zone=public --add-service=nfs

# Ports spécifiques
firewall-cmd --permanent --zone=public --add-port=19999/tcp  # Netdata
firewall-cmd --permanent --zone=public --add-port=30000-31000/tcp  # FTP passif
```

Configuration 17 – Configuration Firewalld Commandes de configuration

7.3 SELinux

SELinux (Security-Enhanced Linux), un module de sécurité développé par la NSA pour fournir un contrôle d'accès renforcé, a été activé en mode `enforcing`, ce qui signifie que toutes les politiques de sécurité sont strictement appliquées. Des règles strictes ont été mises en place, afin de n'autoriser que les actions strictement nécessaires au bon fonctionnement du système. Cette approche permet de limiter les risques liés aux attaques et d'atteindre une sécurité de niveau entreprise.

```
# Activation de SELinux en mode enforcing
setenforce 1
sed -i 's/^SELINUX=.*/SELINUX=enforcing/' /etc/selinux/config

# Configuration des  booleans SELinux
setsebool -P httpd_can_network_connect_db 1
setsebool -P httpd_can_network_connect 1
setsebool -P samba_export_all_rw 1
setsebool -P ftpd_full_access 1
setsebool -P ssh_sysadm_login 1

# Configuration du contexte pour les répertoires web
semanage fcontext -a -t httpd_sys_content_t "/srv/web(/.*)?"
restorecon -Rv /srv/web
```

Configuration 18 – Configuration SELinux /etc/selinux/config

```
#  Pour autoriser PHP-FPM à se connecter au socket MySQL
module php_mysql_fix 1.0;

allow httpd_t mysqld_t:unix_stream_socket connectto;
allow httpd_t mysqld_var_run_t:sock_file write;
```

Configuration 19 – Configuration SELinux Politique PHP-FPM - MySQL

7.4 Options de montage sécurisées

Des options de montage spécifiques ont été définies dans le fichier `/etc/fstab` afin de renforcer la sécurité des différentes partitions du système. Ces options permettent, par exemple, d'empêcher l'exécution de fichiers binaires (`noexec`), l'écriture sur certaines partitions (`ro, nosuid, nodev`), ou encore de limiter les privilèges d'accès. Ce type de configuration réduit la surface d'attaque en restreignant les comportements potentiellement dangereux sur certaines zones du système de fichiers.

```
# Montage en lecture seule du bootloader
/dev/nvme0n1p1 /boot/efi vfat umask=0077,ro 0 2

# Protection du répertoire temporaire
tmpfs /tmp tmpfs defaults,noexec,nosuid,nodev,size=2G 0 0

# Protection des répertoires utilisateurs
/dev/mapper/vg-home /home xfs defaults,nodev 0 2

# Protection des partages
/dev/mapper/vg_raid-share /srv/share ext4 defaults,usrquota,grpquota,nodev ↩
   ↪ 0 2
/dev/mapper/vg_raid-web /srv/web ext4 defaults,usrquota,grpquota,nodev 0 2
```

Configuration 20 – Configuration Système /etc/fstab

7.5 Fail2Ban

Fail2Ban est un outil de sécurité permettant de protéger le système contre les attaques par force brute en surveillant les fichiers journaux de plusieurs services critiques (comme SSH, FTP, SMTP, Apache, etc.). Lorsqu'il détecte des tentatives de connexion répétées échouées, il bloque automatiquement l'adresse IP de l'attaquant en configurant temporairement le pare-feu, réduisant ainsi les risques d'intrusion ou d'exploitation.
Voici les configuration fail2ban déployé :

```
[sshd]
enabled = true
port = ssh
filter = sshd
logpath = /var/log/secure
maxretry = 10
bantime = 3600
```

Configuration 21 – Configuration Fail2Ban SSH /etc/fail2ban/jail.d/sshd.local

```
[vsftpd]
enabled = true
port = ftp,ftp-data,ftps,ftps-data
filter = vsftpd
logpath = /var/log/secure
maxretry = 3
bantime = 3600
```

Configuration 22 – Configuration Fail2Ban FTP /etc/fail2ban/jail.d/ftp.local

```
[apache-auth]
enabled = true
port = http,https
filter = apache-auth
logpath = /var/log/httpd/*error_log
maxretry = 3
bantime = 3600
```

Configuration 23 – Configuration Fail2Ban Apache /etc/fail2ban/jail.d/apache.local

```
[samba]
enabled = true
port = samba,samba-ds,samba-ds-port
filter = samba
logpath = /var/log/samba/log.smbd
maxretry = 3
bantime = 3600
```

Configuration 24 – Configuration Fail2Ban Samba /etc/fail2ban/jail.d/samba.local

```
[cockpit]
enabled = true
port = http,https
filter = cockpit
logpath = /var/log/secure
maxretry = 3
bantime = 3600
```

Configuration 25 – Configuration Fail2Ban Cockpit /etc/fail2ban/jail.d/cockpit.local

8 Protection Antivirus et Anti-rootkit

8.1 ClamAV

ClamAV, un antivirus, a été configuré pour renforcer la sécurité du système en assurant une protection avec des scans quotidien contre les menaces virales. Avant chaque analyse, la base de données des signatures virales est automatiquement mise à jour afin de garantir que les derniers malwares connus soient détectés. Une fois cette mise à jour effectuée, un scan complet est lancé quotidiennement sur les partitions les plus exposées aux fichiers utilisateurs, telles que `/srv/web`, `/srv/share` et `/database`. Ces répertoires, qui contiennent des contenus accessibles ou téléversés par des utilisateurs ou via des services web, représentent des vecteurs d'entrée potentiels pour des fichiers malveillants. Les analyses sont programmées à différents moments de la nuit pour répartir la charge et minimiser l'impact sur les performances globales du système. Cette stratégie assure une protection efficace sans compromettre la disponibilité des services.

```
0 2 * * * /usr/bin/freshclam --quiet && /usr/bin/clamscan -r --quiet ↩
   ↪ --infected /srv/database > /var/log/clamav/database.log 2>&1
0 3 * * * /usr/bin/freshclam --quiet && /usr/bin/clamscan -r --quiet ↩
   ↪ --infected /srv/web > /var/log/clamav/web.log 2>&1
0 4 * * * /usr/bin/freshclam --quiet && /usr/bin/clamscan -r --quiet ↩
   ↪ --infected /srv/share > /var/log/clamav/share.log 2>&1
```

Configuration 26 – Configuration ClamAV /etc/crontab

Chaque tâche combine la mise à jour des signatures et le scan d'un répertoire spécifique, avec journalisation des résultats vers des fichiers dédiés pour faciliter l'analyse en cas de détection.

8.2 RKHunter

RKHunter (Rootkit Hunter) a été configuré pour réaliser un scan quotidien du système à la recherche de rootkits, de portes dérobées et autres modifications suspectes pouvant indiquer une compromission. Cet outil analyse les fichiers système, les permissions, les modules noyau, ainsi que les signatures de binaire critiques, afin de détecter toute anomalie par rapport à l'état attendu. Avant chaque analyse, la base de données de détection est mise à jour pour inclure les signatures les plus récentes. Les résultats sont enregistrés dans des journaux consultables pour un suivi régulier. Grâce à cette surveillance proactive, RKHunter peut rapidement repérer des comportements anormaux ou des altérations invisibles à l'utilisateur, ce qui renforce l'intégrité et la sécurité du système.

```
0 5 * * * /usr/bin/rkhunter --check --skip-keypress --quiet >> ↩
   ↪ /var/log/rkhunter/daily-scan.log 2>&1
```

Configuration 27 – Configuration RKHunter /etc/crontab

Ce scan est programmé quotidiennement à 5h du matin et génère un rapport détaillé dans le journal dédié, permettant une vérification régulière de l'intégrité du système.

9 Description et explication des scripts

L'ensemble des scripts forme un programme géré par le script principal `main.sh`, qui joue un rôle central dans la gestion et le déploiement automatisé de nouvelles instances. Ce script principal simplifie la gestion et le déploiement des différents services. Grâce aux entrées utilisateurs, une configuration sur mesure est possible sur toutes instances avec sa propre adresse IP, nom d'hôte, etc.

L'ensemble du programme est disponible sur un dépôt GitHub, dont l'URL est fournie en annexe.

L'architecture générale des scripts est structurée comme suit :

- **`main.sh`** : script central qui appelle les modules suivants :
  - **`hardware.sh`** : configure le système de partitions RAID et LVM, et met en place le mécanisme de sauvegarde à l'aide d'un script planifié via `cron`.
  - **`network.sh`** : gère la configuration du nom d'hôte, des serveurs DNS et NTP, ainsi que du service SSH avec toutes leur configuration.
  - **`filesharing.sh`** : installe et configure les services de partage de fichiers tels que FTP, Samba et NFS, en incluant les répertoires partagés accessibles publiquement.
  - **`webservices.sh`** : installe et configure les services web, notamment le serveur Apache, la base de données MariaDB (dans la partition dédiée), ainsi que l'interface d'administration `phpMyAdmin`.
  - **`usermanagement.sh`** : permet d'ajouter ou de supprimer un utilisateur en configurant simultanément ses accès aux services (utilisateurs système, Samba, FTP, MariaDB) et son domaine associé dans le serveur DNS.
  - **`security.sh`** : met en uvre la politique de sécurité du serveur à travers la configuration de modules tels que le pare-feu, SELinux, les options de montage, ainsi que l'installation d'un antivirus et d'un anti-rootkit. Ce script est conçu pour offrir une certaine flexibilité d'adaptation aux besoins spécifiques.
  - **`monitoring.sh`** : installe et configure le client ou le serveur Netdata, en utilisant l'adresse IP et la clé API pour l'interconnexion.

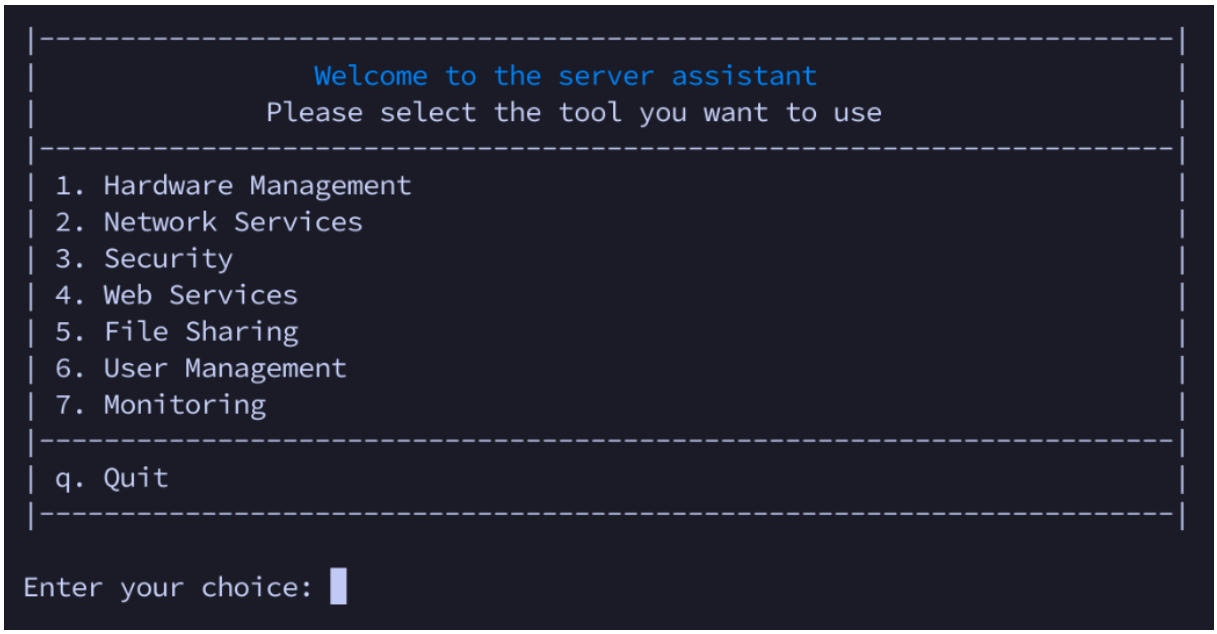

FIGURE 2 – Aperçu du script principal `main.sh` gérant les déploiements automatisés.

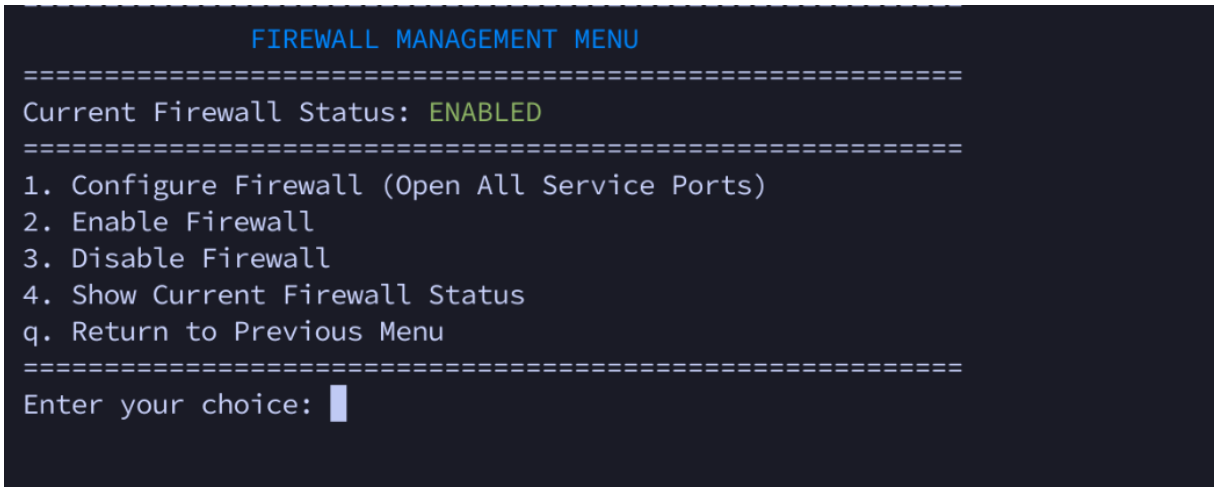

FIGURE 3 – Aperçu du script `security.sh` gérant les règles de firewalls.

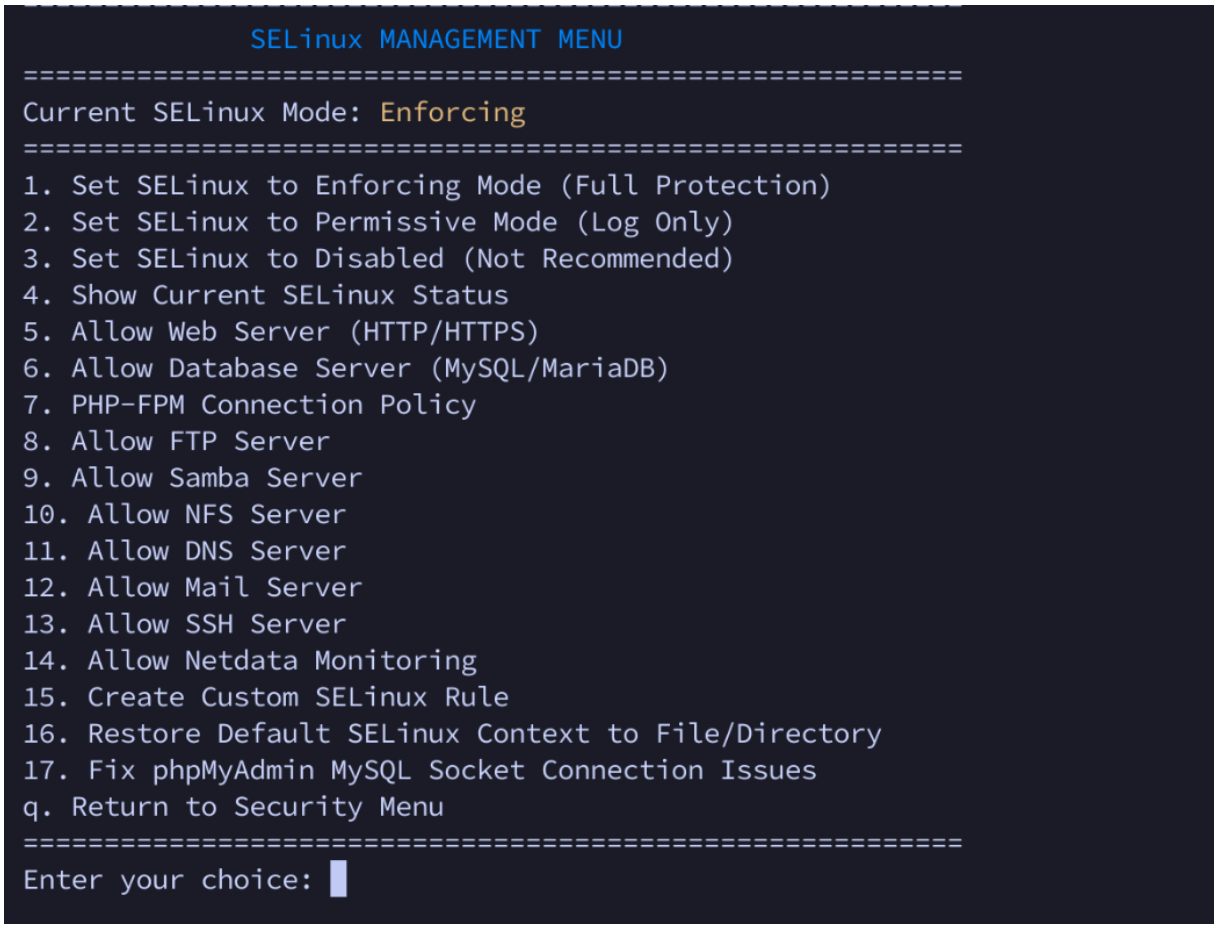

FIGURE 4 – Aperçu du script `security.sh` gérant les règles selinux.

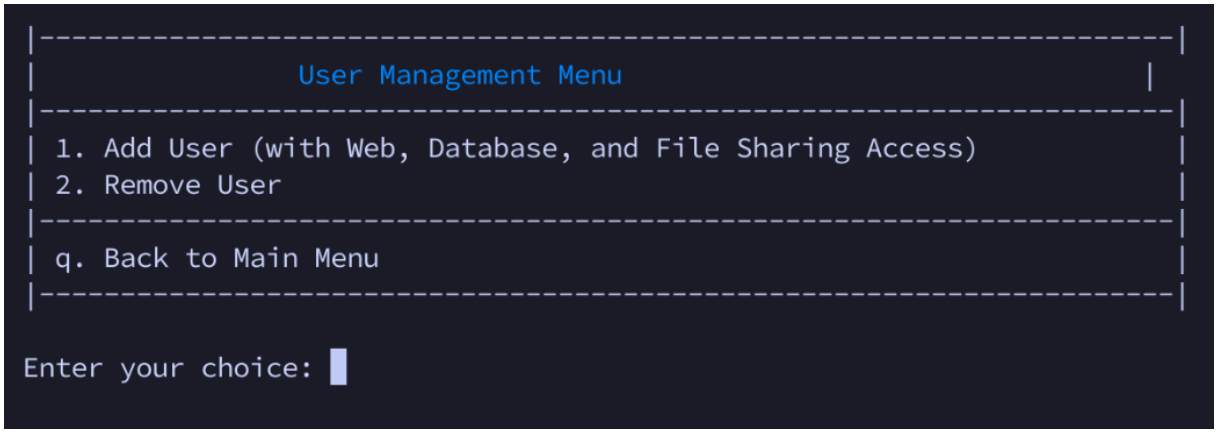

FIGURE 5 – Aperçu du script `usermanagement.sh` gérant les utilisateurs.

Ce programme de script permet la configuration complète de A à Z des différents services et toutes les configurations matérielles et de sécurité du système, facilitant ainsi la gestion et le déploiement.

10 Problèmes rencontrés

— **Problèmes d'authentification AWS** : difficultés persistantes avec l'authentification à AWS S3 et S3 Glacier malgré l'utilisation des bonnes clés API et secrets pour accéder aux buckets. Malgré plusieurs tentatives avec différentes permissions IAM, les clés d'accès étaient systématiquement refusées, nous obligeant à adopter une solution alternative.

— **Instabilité réseau** : qualité médiocre de la connexion rendant la communication extrêmement lente, compliquant significativement les opérations de déploiement et maintenance. Les temps de latence importants ont rendu difficile le débogage en temps réel.

— **Dysfonctionnements de l'infrastructure AWS** : gel inexpliqué de plusieurs instances qui ont repris leur fonctionnement plusieurs heures après, sans qu'aucun problème de configuration ne soit détecté (les mêmes scripts de configuration fonctionnant parfaitement sur un serveur Linode). Ces problèmes nous ont contraints à reconfigurer trois instances vierges le jour même de l'évaluation, limitant notre capacité à implémenter des configurations de sécurité encore plus avancées.

11 Conclusion

Malgré les difficultés techniques rencontrées, ce projet nous a permis d'acquérir des compétences solides en gestion et sécurisation d'infrastructures Linux dans un contexte professionnel. Les solutions mises en oeuvre offrent un compromis équilibré entre sécurité, performance et facilité d'administration.

Nous avons réussi à atteindre les objectifs principaux du projet en déployant une infrastructure complète d'hébergement multi-utilisateur avec :

— Une gestion centralisée des utilisateurs, simplifiant l'administration

— Des services web sécurisés avec Apache, PHP, PhpMyAdmin et MariaDB

— Des solutions de partage de fichiers avec SAMBA, FTP et NFS

— Un système de monitoring en temps réel

— Des sauvegardes automatisées et fiables

— Une sécurité multicouche protégeant contre différentes menaces

Si nous avions disposé de davantage de temps, nous aurions pu développer les aspects suivants :

— Mise en place d'une solution de haute disponibilité sur plusieurs noeuds

— Configuration de sécurité plus avancée (php no exec, options de montages correctes et poussée)

— Api entre noeud (pour ajouter un enregistrement dns sur l'instance dédiée au réseau, etc)

— Mise en place d'un monitoring plus poussé (ex : déploiement d'un Wazuh pour le monitoring de sécurité)

Néanmoins, les fondations sont solides et l'infrastructure répond aux exigences initiales du projet, tout en démontrant notre capacité à adapter notre approche face aux contraintes techniques rencontrées.

12 Bibliographie

Références

[1] Documentation officielle AWS, *Amazon Web Services Documentation*, https://docs.aws.amazon.com/

[2] Documentation Amazon Linux 2023, *Amazon Linux 2023 User Guide*, https://docs.aws.amazon.com/fr_fr/linux/al2023/ug/al2023-ug.pdf

[3] ISC BIND 9 Documentation, *BIND 9 Administrator Reference Manual*, https://kb.isc.org/docs/aa-01031

[4] The Chrony Project, *Chrony Documentation*, https://chrony.tuxfamily.org/documentation.html

[5] The Apache Software Foundation, *Apache HTTP Server Documentation*, https://httpd.apache.org/docs/

[6] MariaDB Foundation, *MariaDB Server Documentation*, https://mariadb.com/kb/en/documentation/

[7] Samba Team, *Samba Documentation*, https://www.samba.org/samba/docs/

[8] Chris Evans, *vsftpd Documentation*, https://security.appspot.com/vsftpd.html

[9] Netdata Inc., *Netdata Documentation*, https://learn.netdata.cloud/docs/

[10] Red Hat, *SELinux User's and Administrator's Guide*, https://access.redhat.com/documentation/en-us/red_hat_enterprise_linux/7/html/selinux_users_and_administrators_guide/

13 Annexes

13.1 Gestion de projet

Trello Pour la gestion des tâches et le suivi des délais du projet, nous avons utilisé un tableau Trello accessible à l'adresse suivante :
https://trello.com/invite/b/6823356ebf1cd401ea0175e8/ATTI3a52839bae87ca535511781bc7c02projet-linux

GitHub Le code source des scripts sont disponibles dans le dépôt GitHub :
https://github.com/sametcatakli/projet-linux-aws