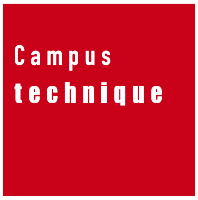
Rapport de projet

2ème Bachelier en Informatique

Linux

Auteur :
Timothée Simon
Florian Giarrusso
Fabio Cumbo

Enseignant :
Antoine Malaise



Année académique 2017 - 2018

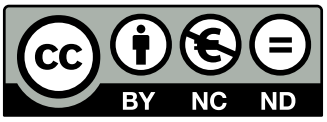


Table des matières

1 Introduction

1.1 Choix de la distribution

Pour le choix de la distribution nous avons commencé pas mettre en place certains critères de recherches :

— La gratuité

— La stabilité

— La légèreté

Nos recherches nous montrent plusieurs choix possibles :

— RHEL : Payante

— Arch Linux : Pas la plus stable car elle fonctionne en rolling release

— Ubuntu Server et Debian : Ne possèdant pas de version core ce qui les rend plus lourdes

— CentOS : Répond au mieux à tout nos critères

Nous allons donc faire notre serveur sous CentOS car celui ci répond à tous nos critères dans sa version Core.

1.2 Mode d'installation

Pour l'installation nous avons opté pour l'écriture de scripts pour chacune des fonctionnalités de notre serveur. Cela nous permet de nous rappeller de nos procédures et de toujours les comprendre dans quelques années. Il nous suffit maintenant de copier nos scripts sur une machine réelle ou virtuelle pour pouvoir immédiatement commencer à configurer Linux. Bien évidement nos scripts ne gèrent pas beaucoup d'erreurs et ne vérifient pas ce que l'utilisateur a entré, ils ne sont donc pas vraiment prêt pour tous les usages mais sont très utiles comme notes.

1.3 Organisation du groupe

Pour nous organiser nous avons utilisé les outils de GitHub. Nous avons donc commencé par créer un repo (privé pour le moment mais nous le passerons surement en publique après les examens) nous permettant de travailler ensemble sur notre code. Nous avons aussi utilisé la ToDo list de GitHub pour nous organiser dans notre développement.

1.4 Machine physique

Nous avons eu l'occation d'utiliser une machine physique pour mettre en pratique nos scripts. Celle-ci est bien évidement une machine de récupération, elle est composée de Intel Pentium 4, de 1Go de DDR2 et de deux disques dur en RAID 1.

2 Installation

Pour l'installation nous avons suivi les étapes suivantes :

- Mise en place de la machine avec l'iso bootable (sur machine virtuelle ou physique)
- Sélection des locale (English US)
- Modification de l'heure (Bruxelles)
- Partitionnement manuel (LVM avec encryption AKA LUKS, sauf pour la machine réelle, faute de puissance) :
  - /dev/sda1 monté sur /boot : 512 Mo en EXT4
  - LVM (encryptée)
    - centos-root monté sur / : 25Go en EXT4
    - centos-swap monté comme partition de swap : 7.8Go
- Activation de la NIC (enp0s3)
- Nom de domaine
- Lancement de l'installation
- Mise en place du mot de passe root

3 Serveur NTP

Le protocole NTP (***N**etwork **T**ime **P**rotocol*) va permettre de synchroniser les horloges des ordinateurs connectés au même réseau local que celle du serveur de temps. Celui-ci devra synchroniser sa propre horloge en contactant un serveur de temps de référence à distance donc par internet. Cette synchronisation des heures permettra entre autres de ne pas perturber certaines applications utilisant l'horloge du système mais aussi pour donner plus de cohérence en cas de comparaison des messages de « logs » de plusieurs ordinateurs sur le réseau.

```
#!/bin/bash

source ../Common.sh

RootCheck

#Installation de NTP
Installe ntp ntpdate ntp-doc

#On stop le service ntpd
systemctl stop ntpd

#On met le serveur à la bonne heure au préalable
ntpdate be.pool.ntp.org

#Configuration du serveur ntp
cp ntp.conf /etc/ntp.conf

#Démarrage et activation du service
Service ntpd

echo "Le service NTP est maintenant installé et activé."
```

../scripts/ntp/NTP.sh

```
# For more information about this file, see the man pages
# ntp.conf(5), ntp_acc(5), ntp_auth(5), ntp_clock(5), ntp_misc(5),
    ntp_mon(5).

driftfile /var/lib/ntp/drift

#On ajoute un directory pour les logs
logfile /var/log/ntp.log

# Permit time synchronization with our time source, but do not
# permit the source to query or modify the service on this system.
#restrict default nomodify notrap nopeer noquery

# Permit all access over the loopback interface.  This could
# be tightened as well, but to do so would effect some of
# the administrative functions.
#restrict 127.0.0.1
#restrict ::1

```

```
#On établis les règles pour l'accès au service, et sécurisation
#On empêche les machines distantes de modifier la configuration du serveur, protection ddos,..
restrict default nomodify nopeer notrap noquery

#On fait confiance à la machine elle-même
restrict 127.0.0.1 mask 255.0.0.0
#IpV6
restrict ::1
# Hosts on local network are less restricted.
#restrict 192.168.1.0 mask 255.255.255.0 nomodify notrap

# Use public servers from the pool.ntp.org project.
# Please consider joining the pool (http://www.pool.ntp.org/join.html).
#server 0.centos.pool.ntp.org iburst
#server 1.centos.pool.ntp.org iburst
#server 2.centos.pool.ntp.org iburst
#server 3.centos.pool.ntp.org iburst

#Synchronisation des horloges avec la pool zone de Belgique
server 3.be.pool.ntp.org
server 1.europe.pool.ntp.org
server 0.europe.pool.ntp.org

#On indique au serveur qu'il doit se synchroniser sur l'horloge locale
server 127.127.1.0

#serveur “”bidon en guise 'dIP fallback, au cas où la source de temps extérieure deviendrait momentanément indisponible. En cas 'dindisponibilité du serveur distant, NTP continuera à tourner en se basant sur ce fonctionnement-là.
fudge 127.127.1.0 stratum 10


#broadcast 192.168.1.255 autokey  # broadcast server
#broadcastclient      # broadcast client
#broadcast 224.0.1.1 autokey    # multicast server
#multicastclient 224.0.1.1    # multicast client
#manycastserver 239.255.254.254   # manycast server
#manycastclient 239.255.254.254 autokey # manycast client

#Enable public key cryptography.
#crypto

includefile /etc/ntp/crypto/pw

# Key file containing the keys and key identifiers used when operating
# with symmetric key cryptography.
#
keys /etc/ntp/keys

# Specify the key identifiers which are trusted.
#trustedkey 4 8 42

# Specify the key identifier to use with the ntpdc utility.
#requestkey 8
```

```
# Specify the key identifier to use with the ntpq utility.
#controlkey 8

# Enable writing of statistics records.
#statistics clockstats cryptostats loopstats peerstats

# Disable the monitoring facility to prevent amplification attacks using ntpdc
# monlist command when default restrict does not include the noquery flag. See
# CVE-2013-5211 for more details.
# Note: Monitoring will not be disabled with the limited restriction flag.
disable monitor
```

../scripts/ntp/ntp.conf

On déclare d'abord le fichier de dérive « driftfile ». Il va permettre de corriger les dérives de l'horloge système en l'absence de connexion réseau au serveur de référence. On déclare ensuite le répertoire et le fichier pour stocker les « logs » du service ntpd. On permet la synchronisation avec notre source de temps mais on interdit à la source de modifier ou d'interroger le service sur ce système. On autorise les accès sur l'interface de bouclage Ensuite, on se synchronise avec les serveurs NTP belge de référence. Pour résoudre les problèmes de charges on entre plusieurs adresses. Il s'agit d'un groupement de serveur et la redistribution se fait à l'aide du Round Robin DNS (association de plusieurs adresses IP à un FQDN) Enfin, on précise au serveur de se synchroniser sur l' « Undisciplined Local Clock » et on indique un faux « pilote » destiné à la sauvegarde de l'heure dans le cas où aucune source externe d'heure synchronisée n'est disponible.

4 Connexion SSH

Pour l'administration à distance nous avons décicé d'utilier SSH (***S**ecure **SH**ell*) car se protocol est plus sécurisé que Telnet. Pour la connexion nous n'autorisons que l'authentification avec une clé RSA ou avec un mot de passe et un facteur de double autentification. Pour la double authentification nous avons décidé d'utiliser une clé Yubikey, ce facteur étant physique cela limite fortement l'accès aux personnes non-authorisées. Nous avons choisi de permettre une autre méthode d'authentification car l'utilisation d'une clé RSA seule est un risque en cas de perte du fichier et nous voulions toujours pouvoir être capable d'administer notre serveur même si ce fichier est perdu ou corrompu.

```
#!/bin/bash

source ../Common.sh

RootCheck

ARG=`Argument $1 "22"`

#Installation du paquet open-ssh (vérification) permettant de partager
    un service ssh, par défaut il est normalement déjà installé
Installe openssh-server

echo "Port $PORT" >> sshd_config

# Utilsation du fichier de config
cp sshd_config /etc/ssh/sshd_config
cp sshbanner /etc/ssh/banner

#démarrage service sshd
Service sshd

iptables -A INPUT -p tcp --dport $PORT -j ACCEPT
iptables -A OUTPUT -p tcp --sport $PORT  -j ACCEPT
```

../scripts/ssh/SSH.sh

Après avoir vérifié que l'utilisateur qui lance les script est bien root et installé le serveur openSSH si celui-ci n'était pas déjà installé, nous copions simplement le fichier de configuration et la bannière au bon endroit. Pour finir nous lançons et activons au demarrage le daemon.

5 Serveur DNS

Le serveur DNS (***D**omain **N**ame **S**ystem*) permet de tranformer un nom de domaine en adresse IP.

```
#!/bin/bash

source ../Common.sh

RootCheck

DOMAIN=`Argument $1 "linux"`
SUBDOMAIN=`Argument $2 "toto"`

# Génération des variables utilisées dans la configutation
IP=`ip addr | grep 'state UP' -A2 | tail -n1 | awk '{print $2}' | cut -f1 -d'/'`
NETID=`ip addr | grep 'state UP' -A2 | tail -n1 | awk '{print $2}' | cut -f1 -d'/' | cut -f1-3 -d'.'`
HN=`hostname`

# Installation des paquet
Installe "bind" "bind-utils"

Service named
systemctl restart network.service

# Création du fichier named
echo "
options {
  listen-on port 53 { 127.0.0.1;$IP; }; /*ajout de l'ip du serveur*/
  listen-on-v6 port 53 { ::1; };
  directory   \"/var/named\";
  dump-file   \"/var/named/data/cache_dump.db\";
  statistics-file \"/var/named/data/named_stats.txt\";
  memstatistics-file \"/var/named/data/named_mem_stats.txt\";
        allow-query    { $NETID.0; localhost; }; /*ajout du reseau autorisé à query*/
  recursion yes;   /*mis sur no */

  dnssec-enable yes;
  dnssec-validation yes;

  /* Path to ISC DLV key*/
  bindkeys-file \"/etc/named.iscdlv.key\";

  managed-keys-directory \"/var/named/dynamic\";

  pid-file \"/run/named/named.pid\";
  session-keyfile \"/run/named/session.key\";
};

logging {
        channel default_debug {
                file \"data/named.run\";
```

```
                severity dynamic;
        };
};


/*Si domaine inconnu, aller voir dans named.ca*/

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

/*AJOUT DES ZONES*/

zone \"$DOMAIN.lan\" IN {
type master;
file \"forward.$DOMAIN\";
allow-update { none; };
};

zone \"2.0.10.in-addr.arpa\" IN {
type master;
file \"reverse.$DOMAIN\";
allow-update { none; };
};


include \"/etc/named.rfc1912.zones\";
include \"/etc/named.root.key\";
" > /etc/named.conf

echo "
\$TTL 86400
@   IN  SOA     $HN.$DOMAIN.lan. root.$DOMAIN.lan. (
        2011071001  ;Serial
        3600        ;Refresh
        1800        ;Retry
        604800      ;Expire
        86400       ;Minimum TTL
)
@       IN  NS  $HN.$DOMAIN.lan.
projet  IN  A   $IP
$SUBDOMAIN    IN CNAME    $HN
" > /var/named/forward.$DOMAIN

echo "
\$TTL 86400
@   IN  SOA     $HN.$DOMAIN.lan. root.$DOMAIN.lan. (
        2011071001  ;Serial
        3600        ;Refresh
        1800        ;Retry
        604800      ;Expire
        86400       ;Minimum TTL
)
@        IN  NS      $HN.$DOMAIN.lan.
```

```
104 @           IN   PTR      $DOMAIN.lan.
    projet      IN   A        $IP
106 15          IN   PTR      $HN.$DOMAIN.lan.
    " > /var/named/reverse.$DOMAIN
108
    chown -v root:named /etc/named.conf
110 systemctl restart named.service

112 named-checkconf -v /etc/named.conf
    named-checkzone $DOMAIN.lan /var/named/forward.$DOMAIN
114 named-checkzone $DOMAIN.lan /var/named/reverse.$DOMAIN
```

../scripts/dns/DNS_server.sh

Après avoir mis en place tout ce qui est nécéssaire pour utiliser le DNS, nous copions le fichier de config de Named en modifiant l'adresse IP. Ensuite nous copions les fichiers de zone reverse et forward customisées avec les arguments entrés par l'utilisateur. Finalement nous chargeons tout cela dans bind.

6 Serveur web

Pour le serveur web nous avons opté pour apache pour sa forte communauté.

```
source ../Common.sh

RootCheck

Installe httpd

Service httpd

DOMAIN=`Argument $1 "toto.linux.lan"`
PORT=`Argument $2 "2187"`

# Ajout des configurations au fichier de config
echo "ServerName $DOMAIN:$PORT" >> httpd.conf
echo "Listen $PORT" >> httpd.conf

# Déplacement du fichier de config
cp httpd.conf /etc/httpd/conf/httpd.conf

# Création d'une page web stupide permettant de tester la bonne
    configuration
echo "coucou" > /var/www/html/index.html

# Ajout du port aux iptables
iptables -A INPUT -p tcp --dport $PORT -j ACCEPT
iptables -A OUTPUT -p tcp --sport $PORT  -j ACCEPT
```

../scripts/web/apache.sh

Après avoir tout installé et customisé le fichier de config, nous le copions au bon endroit et créons une page web contenant "coucou". Pour finir nous ajoutons le bon port au firewall.

7 Serveurs de fichiers

Nous avons mis en place deux serveurs de fichier, un serveur NFS et un serveur Samba.

7.1 Serveur NFS

Le serveur NFS (***N**etwork **F**ile **S**ystem*) permettant à un utilisateur d'accéder à des fichiers sur un serveur de la même façon qu'il accède à des fichiers stoqués en local.

```
#!/bin/bash
source ../Common.sh

RootCheck

s="./NFS.sh [DOSSIER] [ARGUMENTS] [IP]
DOSSIER: Dossier de partage (Defaut: /Partage)
ARGUMENTS: Arguments du partage nfs (Defaut: (rw,sync,no_root_squash,
    no_subtree_check))
IP: Adresse ip du serveur (Defaut: adresse IP local de la machine)
"

Aide $1 $s
# Défault du dossier de partage
DossierPartage=`Argument $1 "/Partage"`
ARG=`Argument $2 "(rw,sync,no_root_squash,no_subtree_check)"`
IP=`Argument $3 \`ip addr | grep 'state UP' -A2 | tail -n1 | awk '{
    print $2}' | cut -f1  -d'/'\``

# Installation du serveur nfs
Installe nfs-utils

#création du dossier partagé si celui-ci n'existe pas encore
mkdir -p $DossierPartage

#Modification des permissions d'accès
chmod 755 $DossierPartage

echo "Le dossier $DossierPartage est maintenant crée"


#Activation et démarrage des services nfs au boot
Service rpcbind
Service nfs


#Configuration du fichier /etc/exports
echo "$DossierPartage $IP$ARG" >> /etc/exports

#On exporte le partage
exportfs -a
showmount -e
```

../scripts/nfs/NFS.sh

Après avoir vérifié que l'utilisateur qui lance le script est bien root et importé les variables en argument, nous installons le serveur nfs si celui-ci n'est pas déjà installé. Ensuite nous créons le dossier de partage avec les bons droits et activons les processus requis pour le serveur. Finalemement nous ajoutons les dossier partagé ainsi que les arguments au fichier, partage et nous en informons le nfs.

7.2 Serveur SMB

Le serveur SMB (***S**erver **M**essage **B**lock*) permet à des ordinateurs de partager des fichiers et des imprimantes entre eux.

```
#/bin/bash

source ../Common.sh

RootCheck
s=" ./SAMBA.sh DOSSIER UTILISATEUR NOM GROUPE
DOSSIER: Dossier de partage (Defaut: /Partage)
UTILISATEUR: Utilisateur possédant les droits (Defaut: Utilisateur
    courant)
NOM: Nom du partage (Defaut: DOSSIER)
GROUPE: Nom du groupe utilisé pour le partage (Defaut: GroupePartage)
"

Aide $1 $s

#On crée le dossier de partage
DossierPartage=`Argument $1 "/Partage"`
UserP=`Argument $2 $USER`
NameP=`Argument $3 $DossierPartage`
GroupePartage=`Argument $4 "GroupePartage"`

#Installation samba
Installe samba

#démarrage et activation du démon au démarrage
Service smb

#création du dossier partagé si celui-ci n'existe pas encore
mkdir -p $DossierPartage


#On gère les permissions du dossier partagé
chmod 770 $DossierPartage
chown -R $UserP:$GroupePartage $DossierPartage

echo "Veuilez choisir un mot de passe pour l'utilisateur $UserP"
smbpasswd -a $UserP
```

```
38 # Copie des réglages de samba
   echo "
40 [${NameP}]
   path=${DossierPartage}
42 comment=Partage crée par un script
   public=yes
44 force directory mode=777
   force create mode=777
46 writeable=yes
   browseable=yes" >> smb.conf
48 cp smb.conf /etc/samba/smb.conf

50 # Désavtivation de SELinux (car autrement impossible de se connecter
       depuis le client, il faudra configurer tout ça )
   setenforce 0
52
   #Redémarrage du service smb
54 sudo systemctl restart smb.service

56 #quota
   echo "veuillez rajouter l'option defaults,usrquota,grpquota à la place
       de defaults sur la partition voulu dans le fichier /etc/fstab"
58
   #La modification de /etc/fstab doit préalablement être déjà effectué
60 Installe quota
   mount -o remount /home
62 quotacheck -cufgv /home
   quotaon /home
64 edquota -g  $GroupePartage
   #On affiche les quotas
66 repquota -as
   #Configuration periode de grace
68 echo "Voulez-vous modifier la période de grâce ? (N/y)"
   read P
70 if [ "$P" = "y" ] || [ "$P" = "Y" ]
       then
72     edquota -t
   fi
```

../scripts/samba/SAMBA.sh

Tout d'abord nous intégrons un menu d'aide et nous importons les variables passées en arguments. Ensuite nous installons le serveur samba si celui-ci n'est pas encore installé et nous démarrons ses services.Nous créons le groupe relatif au partage et nous y ajoutons l'utilisateur. Maintenant il nous faut demander le mot de passe du partage à l'utilisateur et créer la configuration. Finalement nous redémarrons le daemon.

8 Utilisateurs

Pour les utilsateurs nous avons opté pour un script intéractif. Nous demandons à l'utilsateur si il a une clé Yubikey, il doit pouvoir se connecter en SSH et il faut lui appliquer un quota. Bien évidement le script n'est lançable que par l'utilisateur root.

```
#!/bin/bash
source ../Common.ssh

RootCheck

echo "Login :"
read LOGIN
echo "Mot de passe :"
read -s PASS
useradd $LOGIN -p $PASS

echo "Voulez-vous configurer une Yubikey pour cet utilisateur ? (N/y)"
read C
if [ "$C" = "y" ] || [ "$C" = "Y" ]
then
  echo "Veuillez connecter votre Yubikey"
  read
  echo "Dans quel slot est configuré votre Challenge ?(1/2)"
  read S
  if [ "$S" = "1" ] || [ "$C" = "2" ]
  then
    SERIAL=`ykinfo -s`
    mkdir /home/$LOGIN/.yubico
    ykpamcfg -$S
    mv /root/.yubico/challenge-* /var/yubico/$LOGIN-"${SERIAL: -7}"
  else
    echo "Dans quel slot voulez-vous configurer votre Cahllenge ? (1/2)
    "
    read S

    echo "y" | ykpersonalize -$S -ochal-resp -ochal-hmac -ohmac-lt64 -
    oserial-api-visible
    SERIAL=`ykinfo -s`
    mkdir /home/$LOGIN/.yubico
    ykpamcfg -$S
    mv /root/.yubico/challenge-* /var/yubico/$LOGIN-"${SERIAL: -7}"
  fi
fi

echo "Permettre la connexion en ssh sur cet utilisateur ? (N/y)"
read C
if [ "$C" = "y" ] || [ "$C" = "Y" ]
then
  usermod -g sshallow $LOGIN
fi

echo "Voulez-vous mettre un quota sur la partition home de cet
    utilisateur ? (N/y)"
```

```
read Q
echo "Avez-vous rajouté l'option defaults,usrquota,grpquota à la place de defaults sur la partition voulu dans le fichier /etc/fstab ? (N/y) "
read F

if ([ "$Q" = "y" ] || [ "$Q" = "Y" ]) && ([ "$F" = "y" ] || [ "$F" = "Y" ])
  then
  Installe quota

  mount -o remount /home
  quotacheck -cugv /home
  quotaon /home/
  edquota -u $LOGIN
  #On affiche les quotas
  repquota -as
  #Configuration periode de grace
  echo "Voulez-vous modifier la période de grâce ? (N/y)"
  read P
  if [ "$P" = "y" ] || [ "$P" = "Y" ]
  then
  edquota -t
  fi
fi

echo "L'utilisateur  $LOGIN a bien été créé"
```

../scripts/users/users.sh

```
#!/bin/bash

source Common.sh
# Activation de l'EPEL
yum install -y https://dl.fedoraproject.org/pub/epel/epel-release-latest-7.noarch.rpm

# Installation du module PAM
Installe pam_yubico

cp ../Files/yubikey-auth /etc/pam.d/yubikey-auth
cp ../Files/sshd /etc/pam.d/sshd
```

../scripts/users/Yubico.sh

Nous avons laissé la possibilité de créer un quota sur la partition /home lors de la création d'un utilisateur. Pour que les quotas soient fonctionnels, il faut d'abord rajouter l'option usrquota et grpquota sur la partition /home comme si dessous

```
/dev/mapper/cento-home /home   ext4   default,usrquota,grpquota 1 2
```

9 Gestion des sauvegardes

Nous utilisons un programe nommé BORG qui gère les sauvegardes de façon incrémentale sur un serveur distant ou sur une autre partition (par exemple un disque USB)

```
#!/bin/bash

source ../Common.sh

RootCheck

Chemin=`Argument $1 "/Backup"` # Le chemin peut aussi être sous la forme user@hostname:/path/to/repo
PWD=`Argument $2 "Test123*"`
DATE=`date '+%Y%m%d%H%M%S'`
tmpfile="/tmp/borgcron"

# Installation de python et de pip
Installe python
pip install --upgrade setuptools

cd /tmp/
curl "https://bootstrap.pypa.io/get-pip.py" -o "get-pip.py"
python get-pip.py

# Installation de borg
pip install borgbackup

# Inistialisation du repo borg
borg init --encryption=$PWD $Chemin
borg create $Chemin::$DATE /

# Ajout de borg au crontab toute les heures
crontab -l > $tmpfile
echo "0 * * * * export BORG_PASSPHRASE='$PWD';borg create $Chemin::`date '+%Y%m%d%H%M%S'` /" >> $tmpfile
crontab $tmpfile
rm $tmpfile
```

../scripts/backups/backup.sh

Après avoir lancé la première sauvegarde on ajoute un crontab pour faire une sauvegarde toutes les heures.

10 Mot de passe Grub

Avec l'arrivée de CentOS 7.2, un nouvel utilitaire a été implémenté pour mettre facilement un mot de passe au chargement de grub. Il suffit d'utiliser la commande : `grub2-setpassword` Entrer le mot de passe et le confirmer. Enfin, il faut modifier le fichier : `/boot/grub2/grub.cfg` et supprimer le paramètre `--unrestricted`

```
### BEGIN /etc/grub.d/10_linux ###
menuentry 'CentOS Linux (3.10.0-862.3.2.el7.x86_64) 7 (Core)' --class
    centos --class gnu-linux --class gnu --class os --unrestricted
    $menuentry_id_option 'gnulinux-3.10.0-693.el7.x86_64-advanced-
    e47f19c5-8992-40a8-b004-05d4fbe5996c'

  content...
```

Il faut faire attention car la génération d'un nouveau fichier de config Grub il faut à nouveau mettre le mot de passe.

11 Serveur FTP

Le protocole FTP (File Transfer Protocol) permet de transferer des fichiers d'un serveur à un client.

```
#!/bin/bash
source ../Common.sh
RootCheck

s='./FTP.sh [DOSSIER]
DOSSIER: Dossier de partage (Defaut: /home/Partage)
'

Aide $1 $s

Installe vsftpd

DossierPartage=`Argument $1 "/home/Partage"`

Service vsftpd

mkdir -p $DossierPartage


echo "veuillez rajouter l'option defaults,usrquota,grpquota à la place
    de defaults sur la partition voulu dans le fichier /etc/fstab"

#La modification de /etc/fstab doit préalablement être déjà effectué
Installe quota
mount -o remount /home
quotacheck -cufgv /home
quotaon /home
edquota -u ftp
#On affiche les quotas
repquota -as
#Configuration periode de grace
echo "Voulez-vous modifier la période de grâce ? (N/y)"
read P
if [ "$P" = "y" ] || [ "$P" = "Y" ]
    then
    edquota -t
fi

echo "# Allow all connections \nvsftpd: ALL\n# IP adress range\nvsftpd:
    10.0.0.0/255.255.255.0" > /etc/hosts.allow
```

../scripts/ftp/FTP.sh

12 Anti-virus

```
#!/bin/bash
source ../Common.sh

RootCheck

#Check installation du support EPEL
Installe epel-release

#Installation de tous les composants de ClamAV
Installe clamav-server clamav-data clamav-update clamav-filesystem clamav clamav-scanner-systemd clamav-devel clamav-lib clamav-server-systemd

#Configuration du daemon Clam
#Copie du template dans le cas où l'on a pas de fichier de configuration
cp /usr/share/clamav/template/clamd.conf /etc/clamd.d/clamd.conf

#Activation de Freshclam pour garder la DB à jour

#Création du service freshclam et configuration
echo "# Run the freshclam as daemon
[Unit]
Description = freshclam scanner
After = network.target

[Service]
Type = forking
ExecStart = /usr/bin/freshclam -d -c 4
Restart = on-failure
PrivateTmp = true

[Install]
WantedBy=multi-user.target" > /usr/lib/systemd/system/clam-freshclam.service

#Démarrage et activation du service au démarrage
Service clam-freshclam.service

#Changement des fichiers service autrement  clamd@.service ne démarre pas
#On renomme le fichier si on l'a pas déjà fais
if [ ! -e "/usr/lib/systemd/system/clamd.service" ];then
    mv /usr/lib/systemd/system/clamd@.service /usr/lib/systemd/system/clamd.service
fi

#On modifie le fichier clamd@scan.service et on change la référence vers
# /lib/systemd/system/clamd.service
echo ".include /lib/systemd/system/clamd.service

[Unit]
```

```
Description = Generic clamav scanner daemon

[Install]
WantedBy = multi-user.target" > /usr/lib/systemd/system/clamd@scan.service

#On modifie le fichier /usr/lib/systemd/system/clamd.service et on configure
echo "[Unit]
Description = clamd scanner daemon
After = syslog.target nss-lookup.target network.target

[Service]
Type = simple
ExecStart = /usr/sbin/clamd -c /etc/clamd.d/clamd.conf --foreground=yes
Restart = on-failure
PrivateTmp = true

[Install]
WantedBy=multi-user.target" > /usr/lib/systemd/system/clamd.service

#Démarrage et automatistion des services
Service clamd.service
Service clamd@scan.service
```

../scripts/antivirus/antivirus.sh

Ce script va permettre d'installer l'antivirus sur le serveur afin d'en améliorer sa sécurité. On vérifie d'abord que l'utilisateur exécute bien le script en tant que root, ce qui est nécessaire pour configurer les fichiers de ClamAV. On installe ensuite EPEL (Extra Package for Entreprise Linux) qui est un repo fournissant des packages additionnels pour les distributions de type RedHat/CentOs.

Ensuite, au cas où on ne possède pas de fichier de configuration, on copie le template clamd.conf. On peut créer le service freshclam et le configurer de sorte qu'il vérifie 4 fois par jour la présence de mise à jour. On le démarre et on l'active au démarrage ensuite.

On renomme également le fichier /usr/lib/systemd/system/clamd@.service en /usr/-lib/systemd/system/clamd.service autrement, par défaut le service ne fonctionne pas. Puis, il faut indiquer le bon chemin du clamd.service dans le fichier /usr/lib/-systemd/system/clamd@scan.service et on peut configurer le fichier clamd.service Les dernières lignes vont pemettre d'activer et de démarrer automatiquement les services clamd et clamd@scan au démarrage

```
#!/bin/bash

date=$(date +%d_%m_%Y_%H_%M)

echo Arret du service freshclam et mise à jour

service clam-freshclam stop

sudo freshclam
```

```
11 service clam-freshclam start

13 echo Service  redémarré et mise à jour effectuée
   echo Entrez le nom du repertoire ou fichier à analyser :
15 read nom

17 if [ -e "$nom" ];then
     echo Analyse en cours ....
19
     echo "
       ------------------------------------------------------------------------
       ---\n" >> $HOME/analyseVirus.log
21   echo " " >> $HOME/analyseVirus.log
     echo "Analyse du $date\n" >> $HOME/analyseVirus.log
23   echo " "  >> $HOME/analyseVirus.log

25   clamscan -l $HOME/analyseVirus.log -r $nom
   else
27   echo Vous avez entré un nom de repertoire ou fichier inexistant
   fi
```

../scripts/antivirus/analyseVirus.sh

Ce script va permettre d'exécuter une analyse virale d'un dossier ou fichier choisi par l'utilisateur qui lance le script. On met d'abord à jour la base de donnée de l'antivirus et puis on demande à l'utilisateur de spécifier la cible. Si elle existe, on procède à l'analyse.

13 Scripts communs

```
#!/bin/bash

# Vérification des erreurs
set -e

# Vérifie qu'un package est installé
function EstInstalle {
  if yum list installed "$@" >/dev/null 2>&1; then
    true
  else
    false
  fi
}

function Installe {
  if EstInstalle "$@"
  then
    echo "$@ est/sont déjà installé"
  else
    yum -y install "$@" &> /dev/null # Tentative d'installation du
    paquet

    if EstInstalle "$@" # Vérification que le paquet à été correctement
    installé
    then
      if [ -z "$2" ]
      then
        echo "$@ est maintenant installé"
      else
        echo "$@ sont maintenant installés"
      fi

    else
      echo "Impossible d'installer le(s) package(s) \"$@\""
      echo "Vérifiez votre connexion internet et réessayez !"
      exit
    fi
  fi
}

function Argument {
  # Si le premier argument est vide on retourne le deuxième
  if [ -z "$1" ]
  then
    echo "$2"
  else
    echo "$1"
  fi
}

function RootCheck {
  if [ "$EUID" -ne 0 ]
  then
```

```
        echo "Vous devez lancer ce script en tant que root"
53      echo "Ré-essayez avec \"sudo ./$0\""
        exit
55  fi
}
57
function Service {
59  if ! systemctl restart $1
  then
61    echo "Erreur lors du lancement du service"
    exit 1
63  else
    systemctl enable $1 &> /dev/null
65    echo "Le service à bien été lancé"
  fi
67}

69function Aide {
  if [ "$1" == "-h" ]
71  then
    printf $2
73    exit
  fi
75}
```

../scripts/Common.sh

Dans ce fichier nous déclarons plusieurs fonctions utilisées dans plusieurs scripts. Les fonctions sont les suivantes :

- Installe : Installe un paquet si celui-ci n'est pas installé.
- Argument : Retourne le string passé en argument au programme et s'il n'y en a pas retourne l'argument par défaut.
- RootCheck : Vérifie que l'utilisateur qui lance le script est bien root.
- Service : Start puis enable un service.
- Aide : Retourne un string d'aide si l'utilisateur le demande avec l'argument -h.

14 Machine réelle

Nous avons utilisé une machine réelle de récupération équipée d'un processeur Intel Pentium 4 et de 1 Go de RAM. Nous avons décidé de faire celà pour augmenter le challenge et le réalisme du projet. Nous n'avons pas eu beaucoup de problèmes causés par cette machine si ce n'est l'impossibilité d'utiliser LUKS (car la machine manque de puissance) et un problème de carte graphique réglé par le module `nomodeset` dans Grub. Nous avons utilisé le RAID1 pour augmenter la sécurité de nos données, nous avons utilisé deux disques de même capacité, ainsi si l'un des deux disque lache il est possible de récupérer l'intégralité des données. Avant de charger nos fichiers sur la machie réelle nous avons travaillé sur des machines virtuelles nous permettant simplement de gérer les snapshot et de travailler lorsque nous ne sommes pas proche de la machine. Nous avons aussi ajouté un mot de passe superviseur sur le BIOS pour empécher les modifications ce qui permettrait d'outrepasser le mot de passe sur GRUB.

Figure 1 – Machine réelle

Nous avons assimiler beaucoup de connaissances grâce à ce projet. Non seulement nous avons appris comment créer des scripts utiles permettant de configurer rapidement un serveur Linux multi-fonction, mais nous avons aussi acquis beaucoup d'expériences en gestion de projet avec des outils comme git et les services de chez Github. La manipulation d'une machine réelle à rajouté beaucoup de réalisme et nous a permis de toucher aussi à la partie matérielle. Ce projet s'est donc très bien déroulé.

15 Remerciement

Je remercie Terencio Agozzino pour avoir réalisé la mise en page de ce document en LaTeX.