Olivier Philippe 2018

Script de contrôle des modèles de serveur

Réalisé dans le cadre du cours Cnam NSY103, Linux : principes et programmation de M. Olivier Roussel

Table des matières

A Introduction

Dans une infrastructure de grande taille, entre les zones de production, d'expérimentation et de qualification le besoin d'un nouveau serveur est quotidien. Afin de répondre rapidement à la demande et d'introduire une standardisation, le service d'infrastructure se constitue un stock de modèles de machines virtuelles. Ces dernières, rapidement configurées sont livrées aux services demandeurs.

Pour garder la main sur la gestion des possibles erreurs, la configuration des modèles reste en partie manuelle. Par contre, son contrôle peut être automatisé avec un outil de conception évolutive permettant ainsi son utilisation pour d'autres applications.

En sachant que l'outil est destiné à des initiés, nous utilisons un script avec les contraintes suivantes :

- Être adaptable à d'autres projets ;
- Être adaptée à différente distributions, Debian Red Hat ;
- Faciliter les modifications ;
- Garder la vision sur la formulation du contrôle afin de juger facilement de sa pertinence ;
- Visualiser tous les contrôles (positifs et négatifs), puis synthétiser les défauts ;

B Structure du script

Les lignes de contrôle placées dans la partie principale, font appel à des fonctions. Celles-ci traitent uniquement l'affichage.

C'est-à-dire que la complexité du code (recherche des chaînes, traitement des lignes commentées, des espaces/tabulations…) n'est pas déportée sur les fonctions. Cette solution évite la multiplication des fonctions et permet de garder la finesse des contrôles dans la partie évolutive.

1) Sortie du script, affichage

Affichage sur une ligne et code code couleur, en gardant la vision sur la formulation du contrôle pour juger sa pertinence.

Exemple de sorties :

```
systemctl is-enabled ssh     ...  enabled
systemctl is-active ssh     ...  inactive
grep -P "^[ \t]*only_from[ \t]*=[ \t]*127.0.0.1 10.0.20.1 10.0.20.2"
/etc/xinetd.d/check_mk     ... Vrai
```

2) Partie fonction du script

Les fonctions prennent en charge l'affichage et le traitement.

```
Titre_P ()      # Argument (Titre)
# Affiche l'argument en titre principal centré
{
echo
#echo "_____________________________________"
echo -e "\033[34m      $1\033[0m"
#echo "_____________________________________"
}
```

```
F_exe ()       # Argument (ligne de commande bash à exécuter)
{
eval $1 >/dev/null 2>&1
if [[ $? = 0 ]]
     then
          echo "$1     ... Vrai"
     else
          echo -n "$1     ... "  | tee -a Test_Template.log
          echo -e "\033[31mFaux\033[0m" | tee -a Test_Template.log
fi
}
```

```
F_Service ()      # Argument (Service)
#1) Affiche la sortie de la commande: systemctl is-enabled $1
#2) Affiche la sortie de la commande: systemctl is-active $1
{
var="systemctl is-enabled $1     ... "
systemctl is-enabled $1 >/dev/null 2>&1
if [[ $? = 0 ]]
     then echo  "${var}" `systemctl is-enabled $1`
     else echo -e "${var} \033[31m" `systemctl is-enabled $1` "\033[0m" | tee -a
Test_Template.log
fi
var="systemctl is-active $1     ... "
systemctl is-active $1 >/dev/null 2>&1
if [[ $? = 0 ]]
     then echo  "${var}" `systemctl is-active $1`
     else echo -e "${var} \033[31m" `systemctl is-active $1` "\033[0m" | tee -a
Test_Template.log
fi
}
```

3) Partie principale du script

Prise en compte des différentes distributions :

Au début du programme, il est demandé à l'opérateur de choisir la distribution.

```
echo "1: Debian"
echo "2: RHEL"
echo -n "Quelle distribution ? " #>&2
read distrib
```

Traitement du résumé :

Un fichier tampon est initialisé, rempli par les contrôles négatifs et affiché en fin de programme.

```
> Test_Template.log
…
Traitement des contrôles avec erreurs  | tee -a Test_Template.log
…
Titre_P "** Résumé des erreurs **"
cat Test_Template.log
rm Test_Template.log
```

Exemple de contrôles :

```
Titre_P "Mise à jour de sécurité : unattended-upgrades"
     F_Service "unattended-upgrades"
     F_exe 'ls /etc/cron.daily/yumsecurity'
case "$distrib" in
1)   # Debian
     F_exe 'grep -P "^[ \t]*Acquire::http::proxy \"http://x.x.x.x:x\";"
/root/tmp_t/50proxy'
     ;;
2)   # RHEL
     F_exe 'grep "^proxy_hostname =x.x.x.x" /etc/rhsm/rhsm.conf'
     F_exe 'grep "^proxy_port =x" /etc/rhsm/rhsm.conf'
     ;;
esac
```

Sortie obtenue :

Mise à jour de sécurité : unattended-upgrades

systemctl is-enabled unattended-upgrades ... enabled

systemctl is-active unattended-upgrades ... inactive

grep -P "^[\t]*Acquire::http::proxy \"http://x.x.x.x:x\";" /root/tmp_t/50proxy ... Vrai

C Contrôles attendus

Liste des contrôles à effectuer pour une distribution Debian.

Synchronisation heure par le réseau, NTP :

Contrôle	Commande bash
Service : systemd-timesyncd	a)
Configuration : /etc/systemd/timesyncd.conf	grep "^NTP= *ntp.obspm.fr *ntp1.jussieu.fr" /etc/systemd/timesyncd.conf
Contrôle, commande : timedatectl	timedatectl | grep "Time zone: Europe/Paris" timedatectl | grep "NTP synchronized: yes"

Mise à jour automatique, unattended-upgrades :

Service : unattended-upgrades	a)
Configuration, derrière un proxy : /etc/apt/apt.conf.d/50proxy	Grep "^[\t]*Acquire::http::proxy \"http://x.x.x.x:x\";" /etc/apt/apt.conf.d/50proxy
Présence du cron : /etc/cron.daily/apt-compat	ls /etc/cron.daily/apt

Politique d'accès aux processus, SELinux :

Configuration : /etc/selinux/config	grep "SELINUX= *disabled" /etc/selinux/config
Contrôle : commande sestatus	sestatus | grep disabled

Serveur de mail, Postfix :

Service : postfix	a)
Configuration, dépersonnalisation : /etc/postfix/main.cf	grep "^myhostname =[ ]*$" /etc/postfix/main.cf grep "^myorigin = \$myhostname$" /etc/postfix/main.cf grep "^mydestination = \$myhostname, localhost$" /etc/postfix/main.cf grep "^relayhost =[ ]*$" /etc/postfix/main.cf

Contrôle des connexion internet, xinetd :

Service : xinetd (ne prends pas en charge systemd)	service xinetd status | grep "Active: active" service xinetd status | grep "Loaded: loaded"

Supervision, check-mk :

Afficher la version paquet de l'agent check-mk-agent pour contrôl	echo "Agent check-mk-agent : " `apt-cache policy check-mk-agent | grep Install`
Configuration, insertion dans xinetd et déclaration du serveur Nagios :	grep -P "^[\t]*only_from[\t]*=[\t]*127.0.0.1 10.0.20.1 10.0.20.2" /etc/xinetd.d/check_mk

/etc/xinetd.d/check_mk	
Présence des scripts locaux et plugins de check_mk : /usr/lib/check_mk_agent/	ls /usr/lib/check_mk_agent/local/check_monit.py Idem autres plugins

Gestion des configurations, etckeeper et mercurial :

Service : etckeeper	Uniquement chargé
Configuration : /etc/etckeeper/etckeeper.conf	grep "^VCS=\"hg\"" /etc/etckeeper/etckeeper.conf
Paquet mercurial installé	dpkg --get-selections mercurial | grep install
Initialisation du dépôt	ls -a /etc | grep .hg
Fichiers de cache vide, absence du fichier /var/run/mercurial-depot	! ls /var/run/mercurial-depot

Connexion des administrateurs, ssh :

Service : ssh	a)
Configuration : /etc/ssh/sshd_config	grep "^PermitRootLogin without-password" /etc/ssh/sshd_config
Clés publiques des administrateurs : fichier authorized_keys présent	ls -a /root/.ssh/authorized_keys
Clés du serveur effacées : Pas de fichiers /etc/ssh/ssh_host_*	! ls /etc/ssh/ssh_host_*

Autres :

Vérifier que les outils suivants sont installés : Client de messagerie : mutt et l'outil swaks Visualiser/gerrer les processus : htop Visualiser les connexion réseau : iftop, bmon Visualiser les accès disque : iotop	dpkg --get-selections swaks | grep install dpkg --get-selections mutt | grep install dpkg --get-selections htop | grep install dpkg --get-selections iftop | grep install dpkg --get-selections iotop | grep install dpkg --get-selections bmon | grep install

a) systemctl is-enabled « service » et systemctl is-active « service »

D Configuration serveur Debian

Résumé des configurations réalisées sur le modèle.

1) Synchronisation heure par le réseau, NTP

On utilise le service de systemd installé par défaut : systemd-timesyncd

Configuration :

```
/etc/systemd/timesyncd.conf
[Time]
NTP=ntp.obspm.fr ntp1.jussieu.fr (exemple)
```

Activation de la synchronisation :

```
timedatectl set-ntp true
```

2) Mises à jour de sécurité automatiques, unattended-upgrades

Le paquet est utilisé pour installer automatiquement les mises à jour de paquets. Il peut être configuré pour mettre à jour tous les paquets ou uniquement les mises à jour de sécurité. (Autre options : périodicité, notification par mail, black list, reboot, limitation de la vitesse de téléchargement...)

Configuration :

```
/etc/apt/apt.conf.d/50unattended-upgrades
/etc/apt/apt.conf.d/20auto-upgrades
```

Test fonctionnement :

```
unattended-upgrades -v (-d debug)
```

Apt derrière un proxy :

```
/etc/apt/apt.conf.d/50proxy
Acquire::http::proxy "http://10.202.1.222:8080";
```

3) Politique d'accès aux processus, SELinux

Security-Enhanced Linux permet de définir une politique de « contrôle d'accès obligatoire » aux éléments d'un système issu de Linux.

Le contrôle d'accès obligatoire, se différencie des droits traditionnels linux en cloisonnant la portée d'un processus. Les décisions de protection sont donc imposées par le système en suivant des règles dans un contexte donné.

État attendu : Pour l'instant, la protection est installé et désactivée.

Service : selinux-basics, autre : selinux-policy-default, contient un ensemble de règles standards.

Configuration, Désactivation des règles :

```
/etc/selinux/config
SELINUX=disabled
```

Test de non fonctionnement :

```
sestatus
SELinux status:                 disabled
```

4) Serveur de messagerie, Postfix

Permet l'envoi des notification.

État attendu : Dépersonnalisation du serveur.

Configuration :

```
/etc/postfix/main.cf
```

5) Contrôle des connexion internet, xinetd

Le deamon gère les connexions basées sur l'internet, placé entre les services internes et les clients, il réalise

des contrôles avant la connexion (filtrage, temps d'accès, utilisation des ressources...)

Configuration :

```
/etc/xinetd.conf
```

- Le répertoire contenant tous les fichiers spécifiques aux services à surveiller (voir le fichier pour configurer check_mk) :

```
/etc/xinetd.d/
```

Voir le fichier pour configurer check_mk

Démarrage du service :

```
/etc/init.d/xinetd restart
```

6) Supervision, check-mk-agent

Ckeck-mk est intercalé entre le moteur de supervision Nagios et les machines à surveiller, il interroge la machine, traite les informations, puis les relais au superviseur.

État attendu :

- Dernière version de check-mk-agent soit installé, disponible uniquement auprès du fournisseur.

- Adapter la supervision dans xinetd, déposer le fichier de configuration suivant :

```
/etc/xinetd.d/check_mk
service check_mk
{
        type            = UNLISTED
        port            = 6556
        socket_type     = stream
        protocol        = tcp
        wait            = no
        user            = root
        server          = /usr/bin/check_mk_agent
# configure the IP address(es) of your Nagios server here:
       only_from       = 127.0.0.1 x.x.x.x y.y.y.y
        disable         = no
}
```

- Copie des scripts locaux et plugins de check_mk, pour Debian :

```
/usr/lib/check_mk_agent/
├── local
│   ├── check_monit.py
│   ├── localhost-check.sh
│   ├── mk-mercurial.sh
│   ├── postfix_mailqueue
│   └── sshd-rootlogin.sh
└── plugins
    ├── apt
```

7) Gestion des configurations, etckeeper et mercurial

etckeeper est un système conçu pour suivre la configuration d'une machine (répertoire /etc) à l'aide d'un gestionnaire de versions (ici mercurial). On doit donc enregistrer manuellement chacune des modifications que l'on fait sur le système, en y ajoutant un message les décrivant.

Service : mercurial

Test installation : hg version

Création d'un dossier d'accueil du dépôt et l'initialiser :

```
mkdir /var/run/mercurial-depot
```

Service : etckeeper

Configuration :

- Pointer sur le gestionnaire de version mercurial :

```
/etc/etckeeper/etckeeper.conf
VCS="hg"
```

- Initialiser le dépôt :

```
cd /etc
etckeeper init
```

8) Connexion des administrateurs, ssh

Les administrateurs établissent une connexion par authentification par clé.

Configuration :

- Les clés publiques des administrateurs sont déposée dans le fichier :

```
/root/.ssh/authorized_keys
```

- La connexion par mot de passe est interdit :

```
/etc/ssh/sshd_config
PermitRootLogin without-password
```

- les clés du serveur sont supprimées, pour provoquer la reconstruction de nouvelles.

9) Autres

Installer outils suivants :

Client de messagerie : mutt et l’outil swaks

Visualiser/gérer les processus : htop

Visualiser les connexion réseau : iftop, bmon

Visualiser les accès disque : iotop