RICHARD Yves

Projet NSY103

Création d’un script Bash pour automatiser l’Installation de Cerberus (Ubuntu)

Github de Cerberus : https://github.com/cerberustesting/cerberus-source

git clone [https://github.com/yun59000/cerber.git**Introduction**](https://github.com/yun59000/cerber.gitIntroduction)**:**

Mon choix s’est porté naturellement vers une préparation de machine linux à l’installation d’un Framework : **Cerberus**

Cerberus est un Framework de tests automatisés qui utilise entre autres sélénium.

Son installation n’étant pas évidente j’ai donc décidé d’écrire mon script Bash à partir d’installation neuve d’Ubuntu.

Le script effectue les installations des prérequis mais aussi l’installation des packages nécessaires à l’installation d’un serveur Cerberus sous docker.

But : installer tous les prérequis et configurer pour avoir accès au Framework Cerberus, dans un premier temps sur un distribution de type Debian (Ubuntu, Debian, Mint)

**1 - Installations des prérequis** :

Pour chaque installation on vérifie dans un premier temps si le package est déjà installé

**A – Algorithme**

On récupère le nom de la distribution -> déterminer quelle commande on utilise pour l’installation

On lance la préparation a l’installation : (on redirige en fonction de la distribution)

On vérifie si parmi la liste des arguments si chaque package est déjà installé

Si c’est le cas on skip l’installation

Sinon on installe

Cas particulier des installations : docker, docker compose et docker machine

On vérifie si le serveur ssh s’est bien installé

S’il n’est pas installé : erreur

Sinon on le démarre et on indique les informations de connexion locales sur le terminal et dans le fichier log

On vérifie l’installation de Docker-machine, on installe si besoin (redondant mais double sécurité tout le lancement est basé dessus)

On vérifie aussi la bonne création du group docker (normalement via l’installation du package Docker)

Mais surtout on vérifie que l’utilisateur courant fasse parti de se groupe pour lancer le server docker-compose

Pour bypasser le redémarrage nécessaire pour que l’appartenance du groupe soit prise en compte on force la connexion au groupe via la commande **sg**

On créer un dossier cerberus dans lequel on va stocker les fichiers de configuration pour le lancement de docker avec la configuration des différents composants

On vérifie si les fichiers de configuration pour lancer le serveur via docker sont déjà présent et le cas échéant on télécharge un fichier exemple a partir du repository de Cerberus-testing et on le renomme

Pour les distributions de type redHat on doit absolument redémarrer pour la prise en compte des droits de groupe docker :

On l’indique a l’utilisateur et on attend sa réponse pour redémarrer, on indique aussi qu’il est nécessaire de relancer le script une fois redémarré (-on utilise grubby pour efectuer cette tache)

Si ce n’est pas le premier démarrage alors on est prêt et on propose a l’utilisateur de lancer le serveur

Lorsque le serveur est lancé on indique un temps moyen de préparation et on vérifie toutes les 10 sec si le serveur est up et on affiche un message pour indiquer son état.

**B - détails**

**--------------------------------PROCESS--------------------------------------**

Installation de **docker**

sudo apt install docker

Installation de **docker-compose**

sudo apt install docker-compose

Installation de **curl** pour récupérer l’installation de de docker-machine et tester la disponibilité du serveur une fois l’installation terminée

sudo apt install curl

Installation de **docker-machine**

base=https://github.com/docker/machine/releases/download/v0.14.0 && curl -L $base/docker-machine-$(uname -s)-$(uname -m) >/tmp/docker-machine && sudo install /tmp/docker-machine /usr/local/bin/docker-machine

Installation de **net-tools** pour avoir accès à ifconfig #optionnel dans le cadre de se script, il est présent pour des modifications futures

sudo apt-get install net-tools

Installation de **wget** pour télécharger le fichier docker-compose\_with\_selenium

sudo apt-get install wget

Installation **openssh-server** pour communiquer avec le serveur par la suite (intéressant dans le cas d’une installation sur serveur

#comme pour le net-tools cette étape est optionnelle dans la première version de ce script

#il sera utile pour communiquer avec le serveur à distance si nécessaire

sudo apt-get install openssh-server

**--------------------------------PARAMETRAGE--------------------------------------**

**On vérifie si le groupe docker est bien installé**, sinon on le créer

Si le groupe docker n’existe pas :

groupadd docker

**On ajoute l'utilisateur courant au groupe "docker"**

sudo adduser nom\_du\_user\_que\_vous\_utiliser docker

Pour que l'ajout au groupe soit pris en compte on doit **fermer la session et la réouvrir** (redémarrer ne fonctionnera pas) ou **newgrp docker** (provoque un soucis de fermeture du sub shell) **ou sg** (equivalent de su mais pour les droit de groupe)

Ce qui nous permet de bypass la déconnexion-reconnexion

**Création** d’un dossier pour stocker le fichier de configuration pour lancer le docker-compose

Pour éviter de se mélanger les pinceaux, **créer un dossier cerberus** sur le bureau

mkdir /home/USER/Bureau/Cerberus

Dedans **copier le fichier docker-with-selenium.yml et le renommer docker-compose.yml**

Se positionner dans le dossier Cerberus et utiliser la commande :

wget https://raw.githubusercontent.com/cerberustesting/cerberus-source/master/docker/compositions/cerberus-glassfish-mysql/default.yml

et mv source destination pour renomer le fichier

Pour les noyaux **redhat** on doit redémarrer la vm pour la prise en compte du groupe docker.

On utilise la commande read pour demander si l’utilisateur souhaite bien redémarrer. Et on indique qu’il faudra relancer le script une fois redémarré.

On demande si l’utilisateur est prêt pour lancer le serveur

**Lancer le compose :**

docker-compose up -d &>>/dev/null

On lance le compose en arrière-plan pour éviter de kill les processus qui lancent le serveur en fermant le terminal

Attendre quelques minutes que le serveur se lance. (3 min) on check toutes les 10 sec si le serveur est up et on affiche un message le cas échéant.

**--------------------------------PROCESS--------------------------------------**

Chaque étape est monitorée par un fichier log ainsi que sur le terminal grâce aux tail -nbligne dufichierlog.txt

**Problèmes rencontrés :**

Lancer un script en sudo donne des droits root à l’exécuteur du script, soucis pour récupérer l’utilisateur d’origine

Systèmes d’installation différents : yum et apt

Nécessite une V2 pour adapter à n’importe quelle distribution

La structure des if mais surtout de respecter les espaces

Assigner de nouvelles informations à une variable : variable = à la place d’un $variable

Afficher à la fois sur le terminal et dans un fichier log les informations voulues en redirigeant aussi les erreurs

Paralléliser le lancement du docker-compose et du check d’état du serveur en même temps (subshell et sous processus)

**Fichiers à exécuter :**

**main.sh #lance la V1**

**mainV2.sh #lance la V2**