TP 4 : Interactions automatisées entre des processus de gestion d'infrastructure

Objectifs

- mettre en relation des processus ITIL comme la gestion de l'inventaire, la gestion de la configuration et la gestion de la disponibilité.
- découvrir un outil d'inventaire
- utiliser une API dans le cadre de l'administration système
- découvrir une plate-forme de supervision
- utiliser SNMP

1) Introduction

GLPI est un outil d'inventaire basé sur un serveur central. Il s'appuie sur un agent (via le plugin *fusioninventory*). L'agent est installé sur chaque machine à inventorier.
Lorsque l'agent se lance, il va prendre toute les informations de la machine (cpu, mémoire, logiciels installés, interfaces réseau, etc.) et alimenter les référentiels des Ordinateurs sur le serveur GLPI.

NB : La VM fourni contient une infrastructure complète sous forme de conteneurs.GLPI est déjà installé sur le conteneur nommé *ops* (pour *operations*, la machine à partir de laquelle sont gérées les autres machines)

L'accès à la machine *ops* se fait en ssh sur l'adresse suivante :
`ssh tprli@192.168.57.98  (mot de passe : tprli)`

L'accès à GLPI se fait via l'URL suivante :
`http://192.168.57.98/glpi/  (identifiant : glpi/glpi)`

Au départ, l'inventaire est vide. Il faut le remplir en deployant des agents.

2) Installation de l'agent fusioninventory

Attention : les opérations de déploiement (installation de paquet, configuration) doivent être faites via un playbook ansible.

- paramétrer Ansible (*/home/tprli/.ansible.cfg) :* ignorer les clés machines (*host_key_checking = False*) et utiliser root comme login de connexion ssh (*remote_user = root)*
- préparer un fichier d'inventaire au format "ini" contenant les pc1, pc2 et pc3.
- déployer l'agent fusion (paquet "fusioninventory-agent")
- paramétrer l'agent (fichier /etc/fusioninventory/agent.cfg) : indiquer l'URL de l'API fusioninventory du serveur GLPI : server = http://_ip_de_la_machine_ops/glpi/plugins/fusioninventory/
- exécuter l'agent sur chaque machine : fusioninventory-agent

Vérifier que les machines et leur configuration ont été ajoutées dans l'inventaire GLPI.

3) Utilisation de l'API de GLPI

GLPI propose une API pour accéder aux données de l'inventaire. L'accès à cette API se fait en deux étapes :

- authentification
- requête à l'API

L'authentification nécessite un login, un mot de passe **et** une clé (*APPTOKEN*). La clé APPTOKEN se crée via Home > Setup > General / API. Il faut activer l'API et créer un Client d'API. C'est ce dernier qui correspond à l'APPTOKEN.

Une fois l'authentification réussie, un identifiant de session est envoyé. Cet identifiant sera ensuite utilisé dans la requête à l'API.

Exemple d'authentification (en shell) :

```
user=glpi ; password=glpi ; authtoken=$(echo -n "$user:$password" | base64)
APPTOKEN='supersecret'
APIURL='http://localhost/glpi/apirest.php'
curl -s -X GET \
  -H 'Content-Type: application/json' -H "Authorization: Basic $authtoken" \
  -H "App-Token: $APPTOKEN" "$APIURL/initSession"
```

L'identifiant de session reçu doit être placée dans un en-tête "Session-Token" :

```
  curl -s -X GET \
    -H 'Content-Type: application/json' -H "Session-Token: $sessiontoken" \
    -H "App-Token: $APPTOKEN" "$APIURL/Computer
```

Écrire dans le language de votre choix un script qui s'authentifie et renvoit la liste de tous les ordinateurs de la base d'inventaire (point d'entrée */Computer* après l'URL de l'API)

4) Inventaire dynamique

En plus d'un fichier statique, Ansible peut fonctionner avec un script d'inventaire. Le but est d'utiliser l'API de GLPI pour servir de source d'inventaire à Ansible.

Le format de sortie pour l'inventaire dynamique est JSON. La sortie doit contenir des listes de machines sous des noms de groupes : "servers", "workstation" etc. (de la même manière que dans le fichiers INI).

Adapter le script précédent pour produire la sortie suivante, compatible avec Ansible. L'ensemble des postes est rangé dans un groupe *all* :

```
{
    "all": {
        "hosts": [ "pc1", "pc2", "pc3" ]
    }
}
```

Ansible exécute le fichier en paramètre du l'option -i à condition que ce fichier soit exécutable et contienne un *Shebang.*

Tester votre script d'inventaire dynamique avec la commande suivante :

```
ansible -i invscript -m ping all
```

NB: Dans cet exemple, le module Ansible "ping" se contente de tester si le host répond.

5) Supervision avec Nagios

Nagios est une plate-forme de supervision. Elle va surveiller des machines et des services. L'objectif est d'intégrer automatiquement les machines listées dans l'inventaire.

Nagios a été préinstallé sur l'infrastructure fournie , dans le conteneur *ops*. L'url de Nagios dans le cadre du TP est :
`http://192.168.57.98/nagios4/`

Les hosts et les services surveillés sont décrits dans des fichiers sous le répertoire */etc/nagios4/objects*. Chaque objet s'appuie sur un modèle (template).

Nagios définit également des services à surveiller.

Chaque service s'appuie sur un script de vérification (check). Le check le plus élémentaire est un ping pour vérifier la connectivité. Des checks plus élaborés sont possibles pour connaître le débit réseau ou le remplissage du disque.

En reprenant le script précédent, créer un fichier "pc.cfg" dans */etc/nagios4/objects* pour toutes les machines de l'inventaire au format suivant :

```
define host{
        use                     linux-server
        host_name               pc1
        check_interval          1
        }
...
```

NB: Dans l'exemple ci-dessus, les pc sont associé au modèle "linux-server" (décrit dans */etc/nagios4/objects/template.cfg*

Activer le fichier *pc.cfg* en l'ajoutant dans les fichiers de configuration chargés par Nagios (cf. */etc/nagios4/nagios.cfg*). Relancer Nagios.

Vérifier que la liste des hosts est modifiée dans l'interface de Nagios.

Se connecter sur un des PC via ssh et arrêter le système

Il est possible de se connecter sur la VM en SSH pour arrêter/relancer les containers avec les commands *lxc-start nom_du_conteneur* et *lxc-stop nom_du_conteneur*

Observer la détection de changement d'état dans Nagios.

Redémarrer les containers que vous avez arrêté.

6) Check Nagios

Nagios utilise des commandes externes pour vérifier le fonctionnement ou mesurer une valeur. Un certain nombre de commandes sont installés dans /usr/lib/nagios/plugins; exemples : check_disk, check_dns, check_ldap, check_smtp, check_ping etc.

Les commandes ont tous les mêmes conventions d'appel et de sortie (cf. http://nagios-plugins.org/doc/guidelines.html) :

- des paramètres en ligne de commande
- la valeur du code retour du programme (0 -> Ok, 1 -> Avertissement, 2 -> Erreur Critique etc.)
- la sortie standard

Exemple (relativement limité au regard des conventions complètes) :
my_check_ssh :

```
#!/bin/sh
OK=0 ; ERROR=2
if nc -q 0 -w 2 -v "$1" 22 </dev/null ; then exit "$OK" ; else exit "$ERROR"; fi
```

Pour superviser un élément sur une machine, il faut créer plusieurs objects dans Nagios :

- un host
- un service
- une commande

Exemple :

```
define host{
        use             linux-server
        host_name       pc1
        check_interval  1
        }

define command{
        command_name    ssh-active
        command_line    /local/nagios/plugins/my_check_ssh $HOSTADDRESS$
        }

define service{
      use                     generic-service
      service_description     Verify if SSH is responding
      host_name               pc1
      check_interval          1
      check_command           ssh-active
      }
```

Ajouter un service qui vérifier si SMTP fonctionne sur localhost (commande *check_smtp*)

7) SNMP

SNMP (Simple Network Management Protocol) permet d'interroger à distance des machines pour obtenir des informations sur leur état. Le modèle de gestion est centralisé : une "Network Management Station" (NMS) va requêter périodiquement l'agent SNMP de chaque machine pour présenter le résultat aux administrateurs.

Nagios peut utiliser des scripts de vérification (check) qui s'appuient sur SNMP.

Toujours via ansible, effectuer les opérations suivantes (pour tous les PC) :

- Déployer l'agent SNMP sur chaque PC via ansible et l'inventaire dynamique
- modifier le fichier de configuration /etc/snmp/snmpd.conf
- activer SNMP pour que l'agent soit à l'écoute sur toutes les interfaces,

- mettre la communauté "example" (au lieu de "public")
- redémarrer l'agent snmpd

Les variables interrogeables par SNMP sont organisées dans une structure de données hiérarchiques.

Ces structures sont spécifiées dans des MIB (Management Information Base). Chaque branche de l'arborescence sera décrite dans une MIB spécifique. Exemple: la MIB la plus couramment utilisé (essentiellement pour fournir des statistiques et des informations concernant la pile réseau), la MIB-II, est détaillée dans la RFC1213.

Dans une MIB, on trouvera des groupes, et dans ces groupes, on aura soit des variables individuelles, soit des tables contenant plusieurs clés, et pour chaque clé, plusieurs variables.

Chaque variable a un chemin (l'OID, Object Identifier), qu'on peut représenter sous forme de chaîne ou sous forme numérique.

Exemple : dans la MIB-II, la variable *sysDescr* qui indique la marque/le modèle de la machine aura comme chemin .iso.org.dod.internet.mgmt.mib-2.system.sysDescr.0 ou .1.3.6.1.2.1.1.1.0
(NB: le 0 terminal correspondant à l'instance de la valeur)

Déterminer l'OID numérique complet du la variable hrSystemProcesses (groupe *hrSystem*) de la MIB 'HOST-RESOURCES-MIB', RFC 2790

Interroger cette variable hrSystemProcesses avec SNMP sur pc1 (la communauté correspond à un mot de passe en clair; sa valeur par défaut est *public*) :

```
snmpget -v2c -c communauté machine_cible OID.0
```

8) Check SNMP dans Nagios

Première étape : modifier la configuration de l'agent SNMP pour qu'il détecte un processus spécifique (sleep) .

Utiliser le module ansible ansible *copy* pour mettre en place un fichier */etc/snmp/snmpd.conf* contenant uniquement 3 lignes :

- l'agent doit être à l'écoute sur l'adresse IP 0.0.0.0 (InAddrAny, toutes les adresses IP) sur le port udp 161
- un communauté "example" en lecture seule, sans restriction d'OID ("default")
- une directive "proc" dans /etc/snmp/snmpd.conf pour le processus "sleep"

Déployer ce fichier et relancer snmpd avec Ansible

Vérifier que cela fonctionne en interrogeant l'agent SNMP :

```
snmpwalk  -v2c -c example pc1 UCD-SNMP-MIB::prTable
snmptable -v2c -c example pc1 UCD-SNMP-MIB::prTable
```

Deuxième étape : créer un service de supervision

Ajouter un "service" dans Nagios associé à un groupe de machine comprenant tous les PC (le groupe sera généré à partir de l'inventaire).

Le service doit vérifier le nombre d'occurence : le seuil d'alerte est de 10 occurences du processus "sleep"

Déclencher artificiellement l'alerte en lançant sur un pc des processus "sleep" en tâche de fond