Mission A.3 – Gestion des serveurs avec le logiciel de gestion de configuration *Ansible*

Suite à votre passage dans le service réseau de Metz, vous avez rapidement pris conscience de l'intérêt que présente le logiciel *Ansible* pour la gestion de la configuration des serveurs. Un nouveau responsable informatique du site de Trêve vient d'être recruté, il utilisait principalement des *scripts* et des images disques pour réaliser les tâches de configuration de ses serveurs. Votre mission d'assistant(e) de la DSI est de le convaincre de l'intérêt d'utiliser un outil de gestion de configuration.

Question A.3.1

Citer quatre arguments en faveur de l'utilisation d'un logiciel de gestion de configuration comme *Ansible* pour configurer les serveurs.

L'utilisation du logiciel *Ansible* est décidée, il est maintenant nécessaire de préparer l'intervention pour configurer les serveurs du site de Trèves.

Question A.3.2

Détailler les étapes nécessaires au déploiement de la configuration des serveurs du site de Trèves via le logiciel *Ansible*.

L'ensemble des serveurs est accessible via le protocole SSH sécurisé par clé asymétrique. Cette méthode d'authentification permet notamment d'éviter à tout moment le passage d'un mot de passe sur le réseau. L'administrateur voudrait donc désactiver l'authentification classique (couple login/mot de passe) sur les serveurs des zones *Out* de l'ensemble des sites.

C'est le fichier de configuration SSH (*/etc/ssh/sshd_config*) présent sur chaque serveur qui permet l'authentification par mot de passe via la ligne « PasswordAuthentication yes ». Pour empêcher cette méthode d'authentification, Il est nécessaire de passer cette directive à « no » et de relancer le service SSH.

Question A.3.3

a) Écrire le fichier d'instructions (*playbook*) « *securSSH.yml* » permettant de répondre aux contraintes de sécurité quant au service SSH.

b) Écrire la commande qui exécute le fichier d'instructions (*playbook)*.

Documents spécifiques au dossier A

DOCUMENT A1 : Extraits de la table de filtrage du commutateur S3X-M1 de Metz

N° de règle	*Adresse source*	*Port source*	*Adresse dest*	*Port dest*	**Action**
1	172.18.224.0/24	*	172.18.145.0/24	22/TCP(SSH)	Autoriser
2	*	*	172.18.31.1/32	53/UDP(DNS)	Autoriser
…					
Défaut	Toutes	Tous	Toutes	Tous	Refuser

NB : il s'agit de filtrage en mode « stateful », les règles de retour sont donc implicites.

DOCUMENT A2 : Présentation du logiciel Ansible

Pour homogénéiser et faciliter la maintenance de son parc, l'équipe réseau de DMat utilise le logiciel libre *Ansible* qui permet de centraliser la configuration système de machines au sein d'un référentiel unique, puis de déployer cette configuration sur l'ensemble ou une partie du parc informatique. *Ansible* permet le déploiement d'applications à distance, la gestion des services (lancement, arrêt, redémarrage, etc.), la copie ou la génération de fichiers de configuration (*template*), la gestion des paramètres système (interfaces, routes, montage, etc.) et l'interaction avec les principales plateformes des opérateurs de services en ligne (*cloud*).

Ansible gère de multiples systèmes d'exploitation et ne nécessite l'installation d'aucun logiciel ou agent spécifique car il se connecte grâce au protocole SSH via un couple de clés. Sur chaque hôte cible, le service SSH doit être activé et la clé publique du serveur *Ansible* doit être déployée.

DOCUMENT A3 : Intégration des machines dans le dispositif

Étape 1 - Elle consiste à faire figurer sur le serveur *Ansible*, les noms des hôtes (individuellement ou par groupe) qui sont administrés dans le fichier de configuration « */etc/ansible/hosts* » dont voici un extrait simplifié, présenté en colonnes (entre crochets, on trouve les groupes d'hôtes, avec les hôtes concernés en dessous) :

[site_metz] SrvPubM.dmat.net SrvGCM.dmat.net ProxyM.dmat.net SrvMailM.dmat.net SrvDnsM.dmat.net …. [site_esch] ProxyEs.dmat.net SrvMailEs.dmat.net SrvDnsEs.dmat.net ….	[Dmat-Out] SrvPubM.dmat.net SrvGCM.dmat.net ProxyM.dmat.net ProxyEs.dmat.net … [Dmat-In] SrvMailM.dmat.net SrvDnsM.dmat.net SrvMailEs.dmat.net SrvDnsEs.dmat.net ...	[serv_Mail] SrvMailM.dmat.net SrvMailEs.dmat.net … [serv_DNS] SrvDnsM.dmat.net SrvDnsEs.dmat.net …	À noter : • qu'il existe un groupe par défaut « all » qui définit tous les serveurs ; • que de nombreux autres groupes spécifiques ont également été créés.

Étape 2 - Elle consiste à copier la clé publique du serveur *Ansible* sur chaque machine.

DOCUMENT A4 : Fichier d'instructions (playbook) Ansible

Ansible fournit un ensemble de modules permettant d'effectuer les tâches les plus courantes comme installer un paquet, redémarrer un service ou bien modifier un fichier de configuration. Les actions à effectuer peuvent être exécutées individuellement en mode commandes ou bien depuis des fichiers d'instructions YAML (*Yet Another Markup Language*) appelés *playbook.* Ces derniers permettent de configurer une machine et de la faire évoluer ; ils décrivent les tâches que le logiciel *Ansible* doit accomplir sur les machines (installation d'un paquet si celui-ci n'est pas encore installé, copie d'un fichier s'il n'existe pas déjà et configuration de ce dernier avec telle ou telle directive, etc.).

Les fichiers *playbooks* utilisent une syntaxe très simple : on définit les hôtes, les variables éventuelles puis les tâches. Chaque tâche possède un nom et appelle des modules.

De nombreux fichiers *playbooks* pour l'installation d'un serveur de base, l'installation de services spécifiques (baseServers.yml, webServers.yml, dbServers.yml, etc.) ont été rédigés.

Extrait simplifié d'un fichier *playbook* qui initialise les serveurs sur un système Linux *Debian*

--- […] - name: Installation des paquets de base hosts: "{{ nomsHotes }}"	## les documents YAML commencent toujours par « --- » # Description du fichier playbook. # Utilisation du module hosts pour appliquer le playbook aux machines définies par la variable « nomsHotes » (option « -e » lors de l'exécution du playbook). Il est possible d'écrire directement ici un nom ou un groupe de machines plutôt qu'une variable.
tasks: - name: Mise à jour liste paquets et installation apt: update_cache: yes name={{ item }} state= present with_items: - vim - unzip - fail2ban […]	# Définition des tâches. # Description de l'action. # Utilisation du module *apt* pour mettre à jour (update_cache: yes et upgrade: dist) et installer les paquets s'ils ne sont pas déjà installés (state=present) qui sont définis dans la boucle {{ item }}. {{ item }} correspond à une variable qui utilise les valeurs présentes dans la directive with_items. Ici, elle va itérer sur les valeurs permettant d'installer la liste de paquets. À noter que si un seul paquet doit être installé, l'utilisation d'une variable n'est pas nécessaire (name=nom_paquet state=present).
- name: Configuration de fail2ban copy: src: fail2ban.conf dest: /etc/fail2ban/jail.d/defaults-debian.conf […]	# Utilisation du module *copy* qui va copier un fichier source sur les machines.
- name: Configuration de zabbix-agent lineinfile: dest: /etc/zabbix/zabbix_agentd.conf regexp: '^ServerActive=127.0.0.1' line: 'ServerActive=172.18.145.1' regexp: '^Hostname=' line: 'Hostname={{ inventory_hostname }}'	# Utilisation du module *linefile* qui permet de modifier les lignes d'un fichier. # Fichier qui doit être modifié. # Recherche de la ligne qui commence par « ServerActive=127.0.0.1 ». # Modification par ServerActive= adresse IP du serveur de supervision ici 172.18.145.1. # Recherche de la ligne qui commence par « Hostname= » # Modification par Hostname=contenu de la variable « inventory_hostname ». Cette variable contient le nom de la machine sur laquelle s'applique le playbook.
- name: Redémarrage de zabbix-agent service: name: zabbix-agent state: restarted	# Utilisation du module service pour redémarrer zabbix-agent après modification de la configuration.

Exécution d'un fichier *playbook*

Ansible est généralement exécuté en mode *push* : un poste de commande lance le fichier *playbook* sur tout ou partie des machines cibles (décrites dans le fichier d'inventaire */etc/ansible/hosts*) :

ansible-playbook baseServers.yml -e "nomsHotes=serv_DNS"

Ansible va se connecter à tous les hôtes compris dans le groupe « serv_DNS » et jouer les différentes étapes du scénario. Ici serv_DNS est la valeur donnée à la variable nomsHotes déclarée dans le script.

Si le nom d'une machine ou d'un groupe de machines est spécifié directement dans le fichier *playbook* au niveau du module *hosts*, il suffit d'appeler la commande **ansible-playbook baseServers.yml** sans option.