4 Déployer sudo avec Ansible

- Nous allons apprendre à :
  - Utiliser Ansible pour automatiser le déploiement d'une application web Python
  - Utiliser Ansible pour créer une politique de sécurité (sudoers)
- Fichiers sources :
  - **Dossier "ansible/4"**
- Qu'est-ce que sudo ?
  - Permet aux utilisateurs de lancer des commandes spécifiques en tant qu'un autre utilisateur (root ou autre), tout en conservant des traces des activités effectuées (principe de non-répudiation).
  - Définition d'une politique de sécurité sudoers
    - Autoriser les membres du groupe " developers" à utiliser la commande sudo pour démarrer, arrêter, redémarrer et modifier l'application web.
    - Appliquer le principe des moindres privilèges (seulement les permissions strictement nécessaires).
- Installation d'une application web : "Salut !"(Python)
  - WebApp fournie
    - Affiche "Salut !" en visitant http://localhost:5000 sur la VM
  - Utiliser des tâches Ansible pour
    - installer les bibliothèques et fichiers nécessaires pour démarrer la webapp.
    - installer un fichier unitaire systemd pour permettre gérer facilement le lancement et l' extinction de l'application
  - Ouvrir le fichier ansible/4/web_app.yml
    - 4 tâches
      - Installer python3-flask, gunicorn3, nginx
        - Utilisation du module apt
        - Usage d'une liste YAML
          - Évite d'avoir à créer une tâche par paquet à installer.
      - Copier l'application d'exemple Flask Salut
        - Module copy
        - {{ item }} est remplacé par les valeurs contenues dans le module "loop" (fichiers salut.py et wsgi.py)
          - salut.py
            - Code Python Flask de l'appli
          - wsgi.py
            - Objet pour le serveur HTTP
        - Permission sur rx pour les membres du groupe developers
      - Copier le fichier unitaire systemd pour la webap Salut
        - Module copy
        - Ouvrir fichier salut.service
          - Lignes 8 et 9
      - Lancer et activer l'application Salut
        - Module systemd permet de lancer un service via systemd
          - Demande le nom du service et l'état
          - Enabled demande à systemd de lancer le service au démarrage du système
          - daemon_reload (= systemctl daemon-reload) permet à systemd de charger tous les fichiers de services et de découvrir notre salut.service
- Le fichier sudoers
  - Qu'est-ce que c'est ?
    - Un fichier sudoers permet de configurer les politiques de sécurité utilisées par la commande sudo. Il s'applique aux utilisateurs et groupes.
    - 3 sections :
      - Defaults
        - Modifier variables d'environnement...
      - User Specifications
        - Détermine quelles commandes les utilisateurs peuvent lancer et sur quelles machines.
      - Aliases
        - Permet de définir des objets réutilisables dans le fichier pour gagner en concision et en clarté.
        - 5 types :
          - User_Alias
            - Utilisateur ou groupe
          - Host_Alias
            - Hôte ou groupe
          - Cmnd_Alias
            - Définit une ou plusieurs commandes
          - Runas_Alias
            - Liste d'utilisateurs ou groupes au nom desquels une commande peut être lancée
    - Lu de haut en bas.
      La dernière règle l'emporte sur les précédentes.
  - Création du fichier sudoers
    - Nous allons utiliser :
      - Module template d'Ansible
      - Un fichier de template (Jinja2)
        - *ansible/templates/*
    - Ouvrir sudoers.yml
      - Module Ansible set_fact
        - Règle les variables pour l'hôte (réutilisables dans les tâches ou n'importe où dans le playbook).
        - Variable "salut_webapp_file"
      - Module template
        - Construit le sudoers à partir du fichier template Jinja2
        - validate : permet de vérifier si le template est correct (visudo -cf)
          %s est remplacé par le contenu de dest
        - Permissions 440 : permissions attendues pour un fichier sudoers
  - Template sudoers
    - ansible/templates/developers.j2
      - Pour systemd un service peut être appelé par son nom ou nom.service.
      - LOCAL_VM
        - Utilise "hostvars" (variable interne à Ansible qui récupère l'IP de la VM lors du provisionnement)
        - User specifications :
          - %developers (le "%" définit qu'il s'agit d'un groupe et non d'un utilisateur).
          - N'importe quel utilisateur du groupe " developers" sur la VM peut lancer, arrêter ou redémarrer et modifier l'application web " Salut" sans spécifier de mot de passe et en tant que root.
- Provisionnement de la VM
  - Décommenter les tâches
    - 4/web_app.yml
    - 4/sudoers.yml
  - Lancer les tâches Ansible grâce à Vagrant
    - Déplacez-vous vers le dossier contenant votre Vagrantfile et entrez la commande " vagrant provision"
- Test des permissions
  - Vérifier la politique de sécurité avec l' utilisateur "florian"
    - Récupérez le 2ème token 2FA du fichier ansible/3/google_authenticator
    - Saisissez la commande
      - ***ssh -i ~/.ssh/devsecops -p 2222 florian@localhost***
    - Saisissez le token lorsque demandé
  - Accès à l'application web
    - Vérifions que l'application web répond bien aux requêtes entrantes
      - ***curl http://localhost:5000***
  - Édition du fichier salut.py pour tester la politique des sudoers
    - La politique sudoers définie précédemment autorise les membres du groupe " developers" à modifier le fichier /opt/engineering/salut.py grâce à la commande ' sudoedit".
      sudoedit sait utiliser n'importe quel éditeur de code et crée une copie du fichier avant de le modifier, au cas où...
    - Pour utiliser un autre éditeur de code que celui défini par défaut (nano sur Ubuntu), il suffit de définir la variable EDITOR avant de lancer la commande sudoedit.
      - ***Ex. : export EDITOR=vim***
    - ***sudoedit /opt/engineering/salut.py***
    - Modifiez le message "salut" par "coucou toi !"
    - Sauvez et quittez
  - Redémarrage de la webapp avec systemctl
    - Pour que les modifications précédentes soient prises en compte, il faut redémarrer le serveur de la webapp.
    - ***sudo systemctl stop salut***
    - ***curl http://localhost:5000***
      - Aucune réponse, l'appli est bien arrêtée.
    - On relance le serveur :
      - ***sudo systemctl start salut***
    - On vérifie que tout fonctionne :
      - ***curl http://localhost:5000***
    - Si problème :
      - /var/log/syslog
      - /var/log/auth.log
- Audit des fichiers journaux
  - Comme dit plus tôt, l'intérêt de sudo est qu'il journalise toutes les actions affectuées.
    Cela peut être utile pour du monitoring ou pour retracer les actions en cas de réponse sur incident.
  - Chaque appel à la commande sudo a été journalisé dans /var/log/auth.log
    - Analysez ce fichier
    - Le fichier auth.log requiers une élévation de privilèges qui n'est pas autorisée pour " florian". Vous devrez donc utiliser sudo en tant que l'utilisateur "vagrant" pour l'ouvrir.
- Résumé du chapitre
  - Ségrégation de privilèges
  - Journalisation des actions
  - Modules Ansible :
    - template
    - systemd
    - set_fact
  - Automatisation de l'installation d'une application web simple et de mécanismes de contrôle (démarrage/arrêt/modification).