Nom : Lorente

Prénom : Loïc

N° Candidat : 02442761534

Réalisation professionnelle n°1

Ansible

BTS Services Informatiques aux Organisations

Option Solutions d'Infrastructure Systèmes et Réseaux

Session 2025

Ecole supérieure Aristée – La Valette (83)

BTS SERVICES INFORMATIQUES AUX ORGANISATIONS SESSION 2025

ANNEXE 9-1-A : Fiche descriptive de réalisation professionnelle (recto)

Épreuve E6 - Administration des systèmes et des réseaux (option SISR)

DESCRIPTION D'UNE RÉALISATION PROFESSIONNELLE	**N° réalisation : 1**
Nom, prénom : Lorente, loïc	**N° candidat : 02442761534**
Épreuve ponctuelle ☒ **Contrôle en cours de formation** ☐	**Date :** ……/…..…/…….

Organisation support de la réalisation professionnelle

La réalisation professionnelle s'appuie sur une organisation fictive : le laboratoire pharmaceutique GSB, né de la fusion de deux grandes entreprises du secteur. À la suite de cette union, GSB cherche à optimiser les performances et l'efficacité de ses activités.

Intitulé de la réalisation professionnelle

Mise en place d'un serveur de configuration et de déploiement

Période de réalisation : 16/09/2024 au 20/12/2024 **Lieu :** Ecole Aristee, 83160 La Valette-du-Var
Modalité : ☒ **Seul(e)** ☐ **En équipe**

Compétences travaillées

- ☒ Concevoir une solution d'infrastructure réseau
- ☒ Installer, tester et déployer une solution d'infrastructure réseau
- ☒ Exploiter, dépanner et superviser une solution d'infrastructure réseau

Conditions de réalisation[1] (ressources fournies, résultats attendus)

Ressources fournies :
Description générale GSB
Description du système informatique
Schéma réseau GSB et attendu
Cahier des charges

Résultats attendus :
Une solution d'infrastructure opérationnelle conformément au cahier des charges.
Documentation produite conformément aux règles et référentiels en vigueur dans l'organisme.

Description des ressources documentaires, matérielles et logicielles utilisées[2]

Documentation officielle Ansible : https://docs.ansible.com/
Documentation officielle Cisco : https://www.cisco.com/c/en/us/support/switches/category.html
Matériels : Dell PowerEdge R210 II, plusieurs switches CISCO, rtrout, proxylab, proxmox delta, Ordinateur Portable
Logiciels utilisés : Ansible, mRemoteNG
Solution d'Hypervision exploitée : Proxmox
OS utilisé pour Ansible : Debian 12

[1] En référence aux *conditions de réalisation et ressources nécessaires* du bloc « Conception et développement d'applications » prévues dans le référentiel de certification du BTS SIO.

[2] Les réalisations professionnelles sont élaborées dans un environnement technologique conforme à l'annexe II.E du référentiel du BTS SIO.

Modalités d'accès aux productions[3] et à leur documentation[4]

Access JURY EXTERNE via Teamviewer au WINSERV19-LOIC

ID : 1 155 801 935
PW : JURY.2025?

Depuis le WINSERV19-LOIC, sur le bureau mRemoteNG pour accéder à toutes les machines de l'infrastructure.
(Toutes les connexions sont déjà préconfigurées avec le bon protocole)
Sur le bureau se trouve aussi les raccourcis web vers PVE-DELTA, PBS-DELTA, PVE-LOIC, PROXYLAB.

Accès à leur documentation :

https://cloud.aristeecampus.org/index.php/s/AlP5I2PUVAz2Fds
Mot de passe : JURY.2025
Puis sélectionner RP N°1

Liste des équipements et accès INTERNE aux productions (via mRemoteNG)

Infrastructure principale (Machines physique)

Nom	Adresse IP	Identifiant	Mot de Passe
PVE-DELTA	192.168.110.235:8006	root	Aristee.2025
PBS-DELTA	192.168.110.231:8007	root	Aristee.2025
SW-RS-DELTA	192.168.110.251	admin	Aristee.2025
MUTLAB	192.168.110.1	admin	Aristee.2025
SWITCH-BDS	192.168.110.254	admin	Aristee.2025
PROXYLAB(pfSense)	192.168.110.100	admin	Aristee.2025

Serveurs GSB (Proxmox VMs, OS : Windows Server et Debian)

Nom	Adresse IP	Identifiant	Mot de Passe
SRV-DC	192.168.110.101	GSB\Administrateur	Aristee.2025
SRV-DHCP	192.168.110.111	GSB\Administrateur	Aristee.2025
INTRALAB	172.16.0.105	ansible	Aristee.2025
BDMED	172.16.60.100	ansible	Aristee.2025
BDMEDOCLAB	172.16.70.100	ansible	Aristee.2025
BDPHARMA	172.16.70.110	ansible	Aristee.2025
MESSAGELAB	172.16.0.20	ansible	Aristee.2025
JURILAB	172.16.30.100	ansible	Aristee.2025
NOTICELAB	172.16.40.100	ansible	Aristee.2025

Environnement PVE-LOIC (Proxmox VMs, OS : Windows Server et Debian)

Nom	Adresse IP	Identifiant	Mot de Passe
PVE-LOIC	192.168.110.232:8006	root	Aristee.2025
WINSERV19-LOIC	192.168.110.20	Administrateur	Aristee.2025
Ansible	192.168.110.23	ansible	Aristee.2025
Zabbix	192.168.110.24	ansible	Aristee.2025

[3] Conformément au référentiel du BTS SIO « *Dans tous les cas, les candidats doivent se munir des outils et ressources techniques nécessaires au déroulement de l'épreuve. Ils sont seuls responsables de la disponibilité et de la mise en œuvre de ces outils et ressources. La circulaire nationale d'organisation précise les conditions matérielles de déroulement des interrogations et les pénalités à appliquer aux candidats qui ne se seraient pas munis des éléments nécessaires au déroulement de l'épreuve.* ». Les éléments peuvent être un identifiant, un mot de passe, une adresse réticulaire (URL) d'un espace de stockage et de la présentation de l'organisation du stockage.

BTS SERVICES INFORMATIQUES AUX ORGANISATIONS **SESSION 2025**

ANNEXE 9-1-A : Fiche descriptive de réalisation professionnelle (verso, éventuellement pages suivantes)

Épreuve E6 - Administration des systèmes et des réseaux (option SISR)

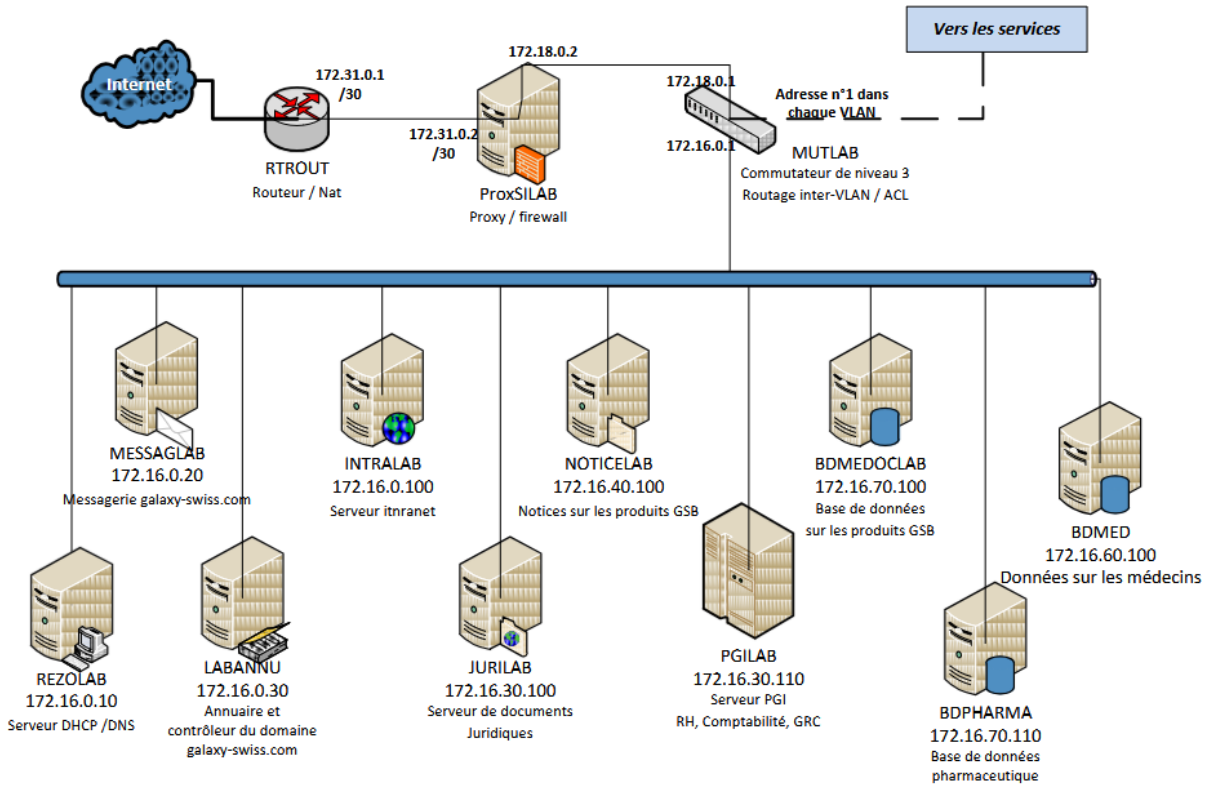
Descriptif de la réalisation professionnelle, y compris les productions réalisées et schémas explicatifs

Contexte GSB

Le laboratoire Galaxy Swiss Bourdin (GSB) est issu de la fusion entre le géant Américain Galaxy (spécialisé dans le secteur des maladies virales dont le SIDA et les Hépatites) et le conglomérat européen Swiss Bourdin (travaillant sur des médicaments plus conventionnels), lui-même déjà union de trois petits laboratoires.

Après deux années de réorganisations internes, tant au niveau du personnel que du fonctionnement administratif, l'entreprise GSB souhaite moderniser l'activité de visite médicale.

L'infrastructure actuelle :



Quel est le besoin ?

GSB doit gérer un parc informatique croissant, avec des configurations variées sur plusieurs sites. L'administration manuelle devient complexe, augmentant les risques d'erreurs et de non-conformité. Une solution centralisée et automatisée de gestion des configurations est nécessaire pour standardiser les configurations des équipements présents dans le parc informatique, sécuriser le SI et optimiser la gestion des infrastructures.

[4] Lien vers la documentation complète, précisant et décrivant, si cela n'a été fait au verso de la fiche, la réalisation professionnelle, par exemples service fourni par la réalisation, interfaces utilisateurs, description des classes ou de la base de données.

Solutions envisageables :

Pour automatiser la gestion de son infrastructure, GSB a étudié plusieurs outils reconnus dans le domaine de l'orchestration et de la gestion de configuration :

- **Puppet** : robuste et largement adopté, mais nécessitant un agent sur chaque machine et une infrastructure dédiée. Open Source / Payant (Puppet Enterprise)
- **Chef** : puissant et flexible, mais avec une courbe d'apprentissage plus complexe en raison de son langage Ruby. Open Source / Payant (Chef Automate)
- **SaltStack** : rapide et efficace, mais reposant sur une architecture maître-minion qui peut être plus lourde à déployer. Open Source / Payant (SaltStack Enterprise, racheté par VMware)
- **Ansible** : simple et agentless, basé sur SSH. 100% Open Source / Payant (Ansible Automation Platform via Red Hat)

Solution retenue :

Après analyse, **Ansible** (sur VM Debian) a été choisi pour sa simplicité et son efficacité. Son architecture **agentless** simplifie le déploiement en évitant l'installation et la gestion d'agents sur les machines cibles, réduisant ainsi les contraintes de maintenance. De plus, son approche basée sur l'**idempotence** garantit que les configurations sont appliquées de manière prévisible et sans effets indésirables, évitant les modifications inutiles. L'utilisation de fichiers YAML et son intégration fluide avec l'environnement existant permettent une gestion standardisée, évolutive et sécurisée de l'infrastructure.

Qu'est-ce qu'Ansible ?
Ansible est un outil open-source d'automatisation permettant de gérer la configuration des systèmes, le déploiement d'applications et l'orchestration d'infrastructures de manière simple et efficace. Il repose sur une approche déclarative et utilise des fichiers YAML pour la configuration de rôles, tâches et les exécuter via des fichiers appelés playbooks.

Histoire d'Ansible
Créé en 2012 par Michael DeHaan, Ansible a rapidement gagné en popularité grâce à sa simplicité et son architecture sans agent. Il a été racheté par Red Hat en 2015, puis intégré dans l'écosystème IBM après l'acquisition de Red Hat en 2019.

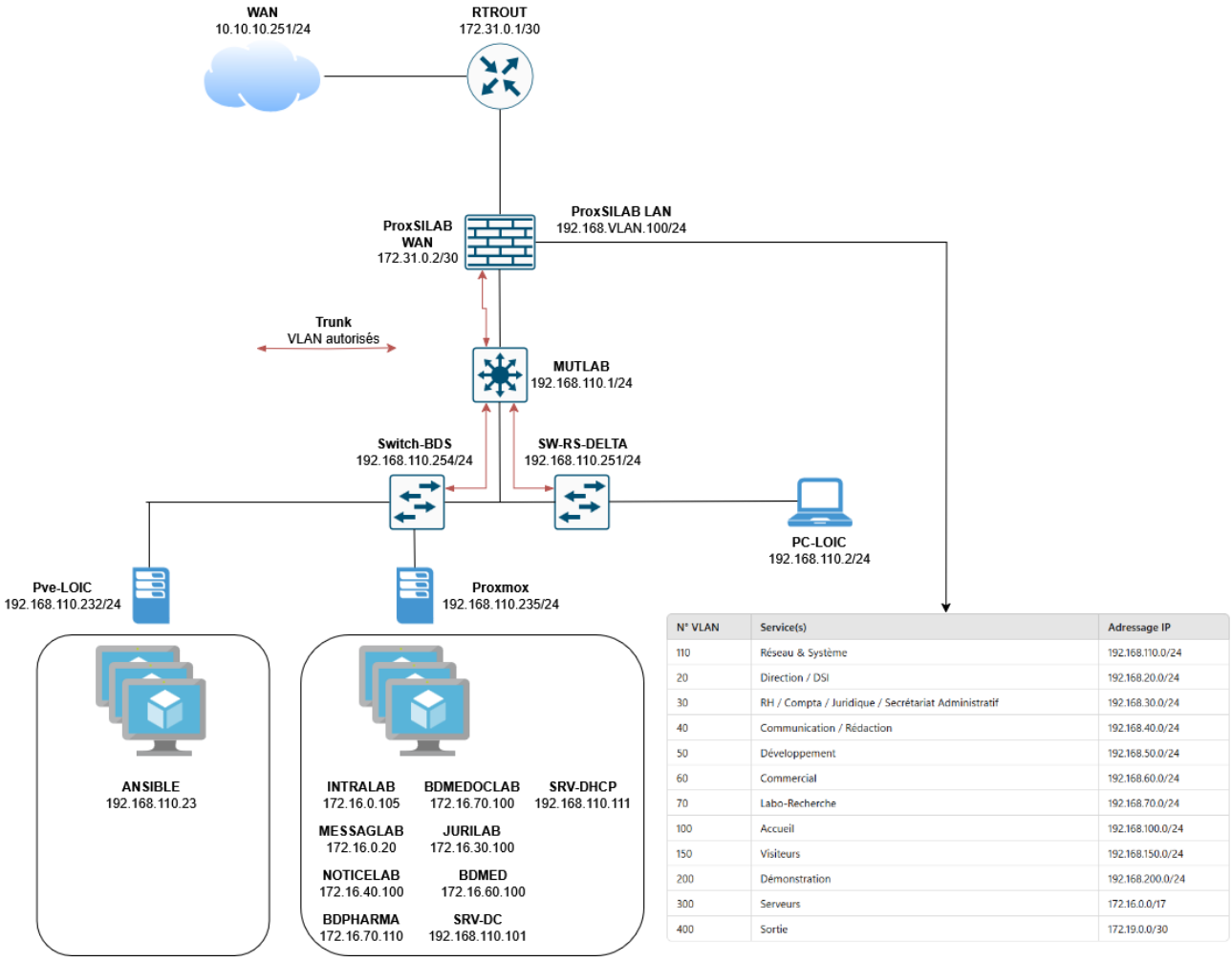
Fonctionnement d'Ansible
Ansible fonctionne en mode agentless en se connectant aux machines cibles via SSH ou WinRM. Il applique des tâches définies dans des playbooks en suivant le principe d'idempotence, garantissant que l'état final du système soit toujours conforme aux attentes, sans modifications inutiles.
Ansible utilise aussi des modules natif pour pouvoir par exemple, connaître les adresses IP des machines cible et/ou connaître leur disponibilité sur le réseau.

Ansible dans le contexte GSB

La DSI de GSB m'a confié la mission d'installer et déployer **Ansible** pour répondre aux besoins d'automatisation et de gestion centralisée de l'infrastructure, une solution permettant de standardiser les configurations et d'optimiser l'administration du parc informatique.
Les caractéristiques principales d'Ansible sont :

- **Agentless** : ne nécessite pas d'installation sur les machines cibles, simplifiant la gestion.
- **Idempotence** : garantit que chaque action n'est appliquée qu'en cas de besoin, évitant les modifications inutiles.
- **Déclaratif** : utilise des fichiers YAML (playbooks) pour définir l'état souhaité des systèmes.
- **Scalabilité** : permet de gérer un grand nombre de machines avec une configuration centralisée et flexible.

Infrastructure avec l'intégration d'ansible :



N° VLAN	Service(s)	Adressage IP
110	Réseau & Système	192.168.110.0/24
20	Direction / DSI	192.168.20.0/24
30	RH / Compta / Juridique / Secrétariat Administratif	192.168.30.0/24
40	Communication / Rédaction	192.168.40.0/24
50	Développement	192.168.50.0/24
60	Commercial	192.168.60.0/24
70	Labo-Recherche	192.168.70.0/24
100	Accueil	192.168.100.0/24
150	Visiteurs	192.168.150.0/24
200	Démonstration	192.168.200.0/24
300	Serveurs	172.16.0.0/17
400	Sortie	172.19.0.0/30

Installation et configuration de Ansible sur une VM Debian
Création et éxécution de playbooks

Introduction :

En tant qu'administrateur système et réseau chez GSB, j'ai mis en place Ansible sur une VM **Debian 12** pour automatiser les tâches d'administration et de maintenance de l'infrastructure.
Cette VM sera positionner dans le VLAN 110 et aura comme adresse IP 192.168.110.23/24.
Le nom de cette machine sera « ansible-loic ».
Les commandes sur la machine ainsi que l'execution des playbooks seront réalisé via l'utilisateur « ansible » qui fait partie du groupe « sudoers ».
J'effectue les mises à jour et je procède à l'installation de la solution.
Voici les étapes principales :

Installation d'Ansible

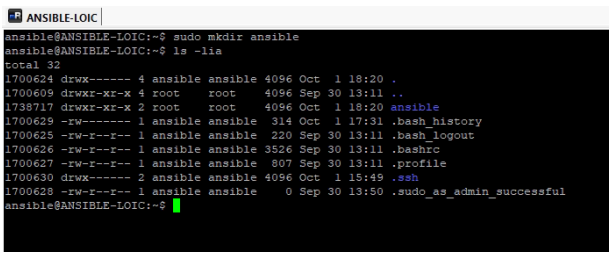
1. **Installation d'Ansible avec sudo :**
    - Commande utilisée : sudo apt install ansible
    - Permet d'installer Ansible via le gestionnaire de paquets apt.

```
ansible@ANSIBLE-LOIC:~$ pwd
/home/ansible
ansible@ANSIBLE-LOIC:~$ apt install ansible
E: Could not open lock file /var/lib/dpkg/lock-frontend - open (13: Permission denied)
E: Unable to acquire the dpkg frontend lock (/var/lib/dpkg/lock-frontend), are you root?
ansible@ANSIBLE-LOIC:~$ sudo apt install ansible
```

Création d'un répertoire et vérification des fichiers

1. **Création d'un répertoire nommé ansible :**
    - Commande utilisée : sudo mkdir ansible
    - Cette commande crée un nouveau répertoire appelé ansible dans le répertoire courant. Ce répertoire sera utilisé pour stocker les configurations Ansible.

2. **Vérification des fichiers dans le répertoire courant :**
    - Commande utilisée : ls -lia
    - La commande liste tous les fichiers et répertoires dans le répertoire courant, avec des informations supplémentaires sur les permissions, les propriétaires, et les liens. Elle montre que le répertoire ansible a bien été créé et qu'il est associé à l'utilisateur et au groupe ansible.



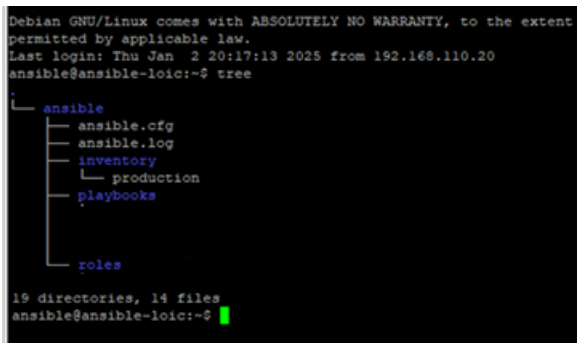
L’arborescence du projet

Dans le cadre de ce projet Ansible, j’ai mis en place une structure claire et bien organisée qui respecte les bonnes pratiques recommandées pour la gestion des projets Ansible. La structure du projet est représentée ci-dessous.
Pour visualisé l’arborescence du contenu des dossiers, j’ai besoin d’une application nommée « tree ».
sudo apt install tree

Présentation de Tree

Le tree présenté reflète l'organisation du projet Ansible, qui repose sur les principes suivants :

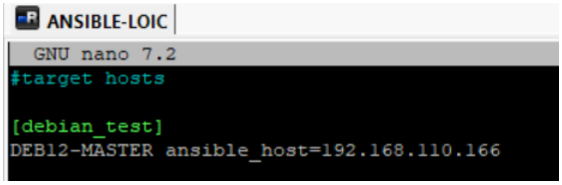
1. **Modularité** : Chaque rôle et playbook est conçu pour être indépendant et réutilisable.
2. **Séparation des responsabilités** : Les fichiers et répertoires sont divisés selon leur fonction, comme la configuration globale (fichier ansible.cfg), les playbooks, les inventaires et les rôles.



Structure détaillée avant la mise en place des playbooks :

1. **Fichier ansible.cfg :**
    - Contient les configurations globales d’Ansible, comme le chemin de l'inventaire, les options de connexion SSH et l'emplacement des rôles.

2. **Répertoire inventory :**
    - Contient les fichiers d’inventaire, qui définissent les hôtes et groupes de serveurs à gérer avec Ansible.
    - Ici, un fichier production est utilisé pour identifier les cibles dans un environnement de production.



3. **Répertoire playbooks :**
    - Contient les fichiers YAML décrivant les séries d'actions à exécuter.
4. **Répertoire roles :**
    - Ce répertoire est divisé en sous-répertoires représentant chaque rôle, tels que commons, security, et services.

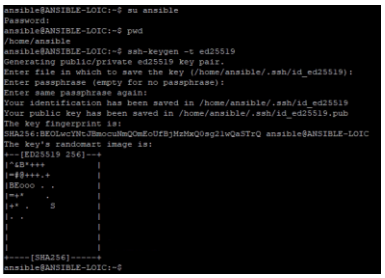
- Chaque rôle suit la structure standard d'Ansible :
  - **tasks/** : Contient les fichiers définissant les tâches principales.
  - **handlers/** : Décrit les actions déclenchées après une modification (par ex. redémarrer un service).
  - **templates/** : Stocke les modèles Jinja2 pour générer des fichiers de configuration.
  - **vars/** : Définit les variables spécifiques au rôle.

Cette organisation structurée constitue une base solide pour les parties suivantes, où je détaillerai le fonctionnement des trois playbooks principaux et leurs dépendances.

Pour pouvoir faire en sorte que ansible puisse communiquer avec les machines cibles sans utilisation de mot de passe à la connexion, il faut mettre en place un échange de clés entre le serveur **Ansible** et les machines cibles avec l'utilisateur « ansible » présent sur toutes les machines.

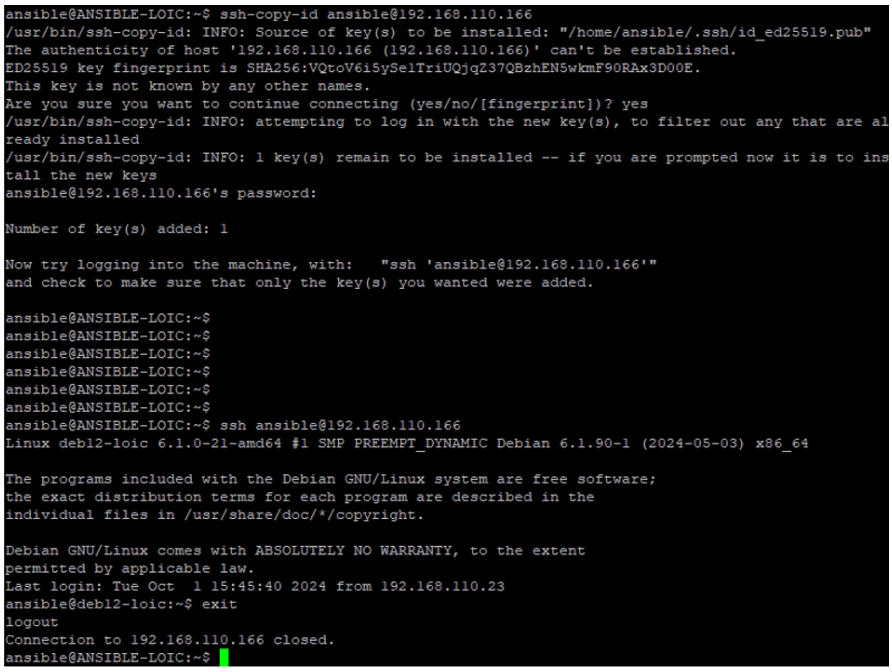
Génération d'une paire de clés SSH avec ED25519 (Serveur Ansible)

1. **Génération de la clé SSH :**
   - Commande utilisée : ssh-keygen -t ed25519
   - Cette commande génère une paire de clés SSH utilisant l'algorithme ED25519, qui est plus performant et sécurisé que RSA pour les connexions SSH.
   - Les clés sont sauvegardées dans le répertoire caché .ssh de l'utilisateur ansible sous les noms id_ed25519 (clé privée) et id_ed25519.pub (clé publique).
2. **Confirmation de la clé générée :**
   - L'identification (clé publique) est stockée et la clé privée est prête à être utilisée pour l'authentification sur des serveurs distants.



Copie de la clé publique SSH et connexion SSH au serveur distant

1. **Copie de la clé publique vers le serveur distant :**
    - Commande utilisée : ssh-copy-id ansible@192.168.110.166
    - Cette commande copie la clé publique SSH (id_ed25519.pub) générée à l'étape précédente sur le serveur distant, ici à l'adresse 192.168.110.166, dans le compte utilisateur ansible.
2. **Connexion au serveur distant :**
    - Commande utilisée : ssh ansible@192.168.110.166
    - Cette commande teste la connexion SSH à partir de l'hôte local (192.168.110.23) vers le serveur distant (192.168.110.166), en utilisant la clé SSH pour l'authentification.
    - La connexion est réussie, ce qui signifie que la clé publique a été installée correctement et qu'il n'est plus nécessaire de fournir un mot de passe.



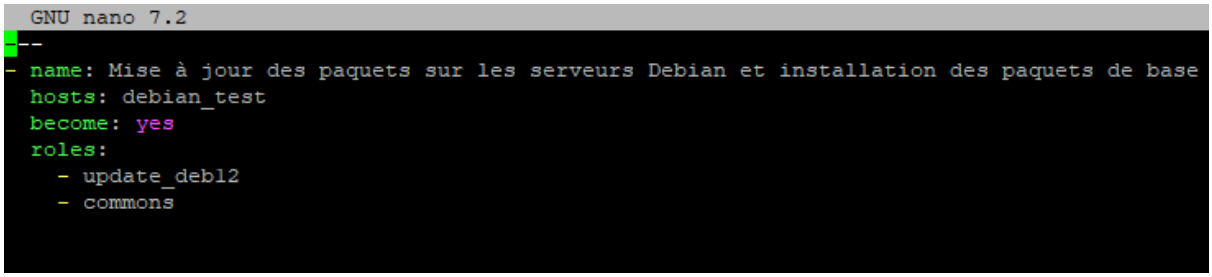
Voici plusieurs playbooks mis en place avec leur description :

Playbook init_deb12.yml et ses dépendances

Introduction
Le playbook init_deb12.yml a pour objectif de préparer les serveurs Debian 12 en effectuant des tâches essentielles telles que la mise à jour des paquets et l'installation des outils de base. Cette étape constitue une base solide pour le déploiement d'autres services.

1. Structure du Playbook

- **Nom du Playbook** : Mise à jour des paquets sur les serveurs Debian et installation des paquets de base.
- **Hôtes ciblés** : debian_test, qui représente les serveurs concernés dans l'inventaire.
- **Privilèges élevés** : Utilisation de become: yes pour exécuter les tâches avec les élévations nécessaires.
- **Rôles associés** :
    - update_deb12 : Gère la mise à jour des dépôts et des paquets.
    - commons : Installe les paquets de base nécessaires à l'environnement.

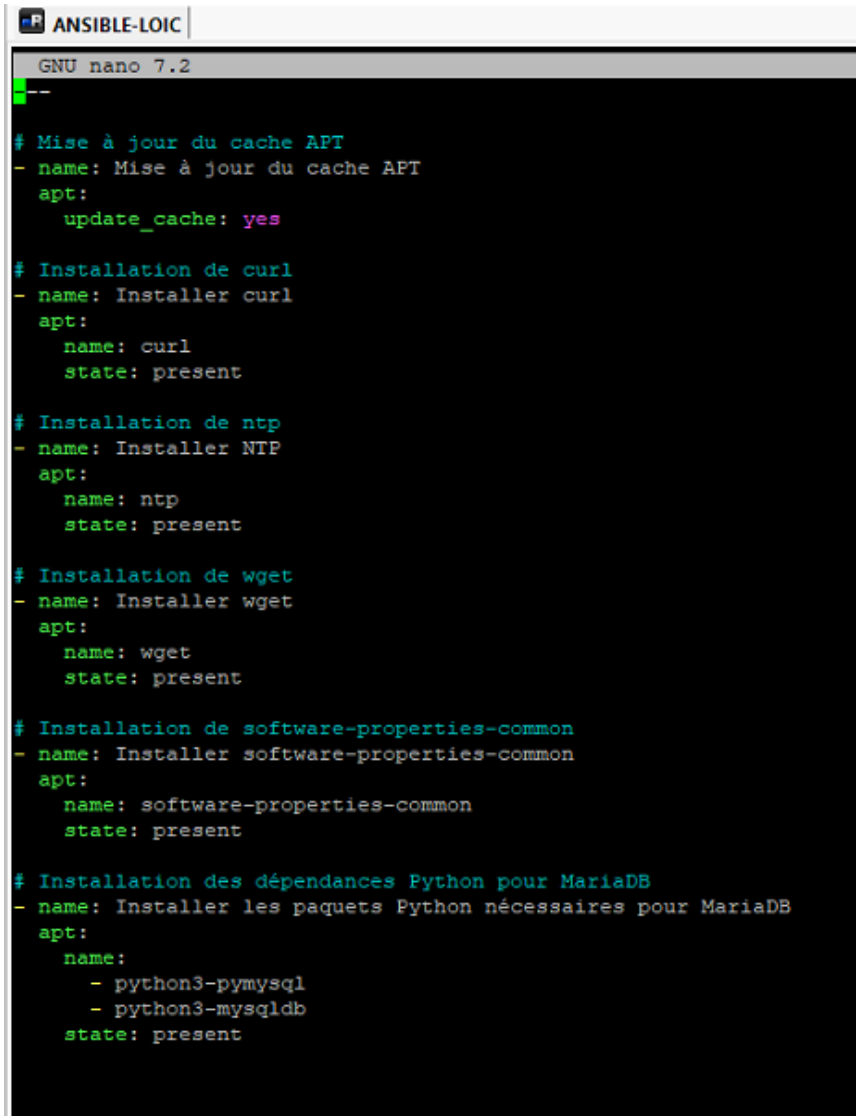


2. Rôle commons

Le rôle commons est responsable de l'installation des outils courants et des dépendances pour les serveurs Debian 12.

Principales tâches :

1. **Mise à jour du cache APT :**
    - Actualise les dépôts pour garantir l'accès aux dernières versions des paquets.
2. **Installation de paquets essentiels :**
    - **curl** : Pour le transfert de données via divers protocoles.
    - **ntp** : Synchronisation de l'horloge système.
    - **wget** : Téléchargement de fichiers.
    - **software-properties-common** : Gestion avancée des dépôts.
    - **Dépendances Python pour MariaDB** : Prépare l'environnement pour les futures bases de données.

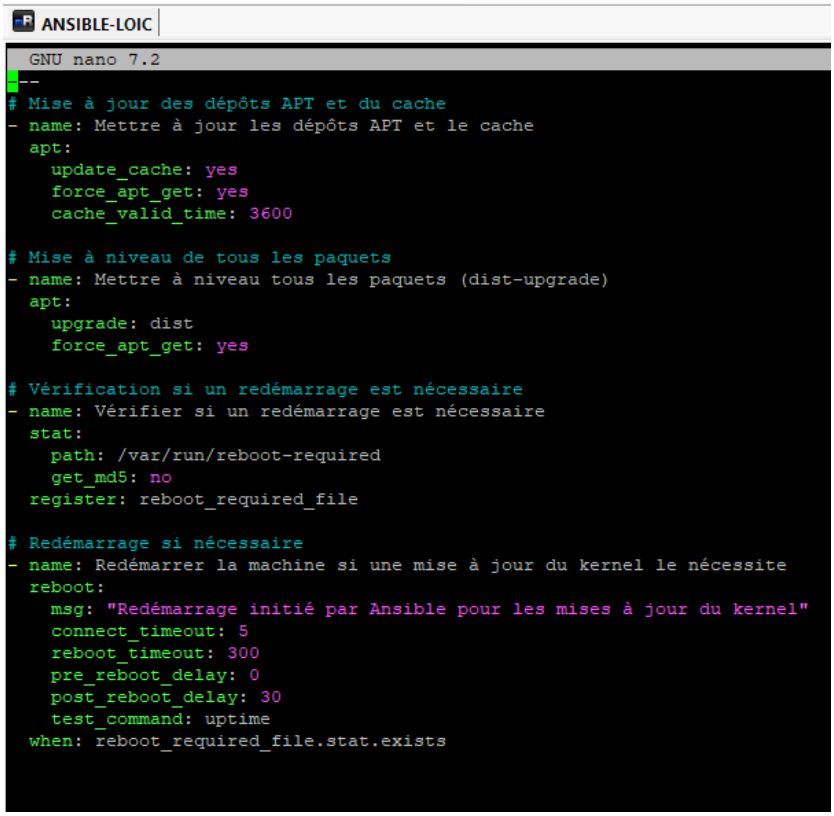


3. Rôle update_deb12

Le rôle update_deb12 garantit que les serveurs Debian 12 sont à jour, minimisant ainsi les vulnérabilités.

Principales tâches :

1. **Mise à jour des dépôts APT :**
    - Utilise update_cache: yes et force la mise à jour des paquets.
2. **Mise à niveau des paquets :**
    - Effectue une mise à jour complète du système.
3. **Gestion conditionnelle des redémarrages :**
    - Vérifie si un redémarrage est nécessaire après une mise à jour critique.
    - Redémarre le serveur si nécessaire, avec un délai configuré.



4. Synthèse

Le playbook init_deb12.yml, combiné aux rôles commons et update_deb12, permet de :

- Préparer un environnement Debian 12 entièrement à jour et fonctionnel.
- Fournir une base fiable pour les déploiements futurs.
- Automatiser les étapes critiques de maintenance, comme les mises à jour et les redémarrages conditionnels.

Playbook security_deb12.yml et ses dépendances

Introduction

Le playbook security_deb12.yml est conçu pour renforcer la sécurité des serveurs Debian 12. Il applique des configurations de protection essentielles, comme la mise en place de Fail2Ban. Cette étape garantit une défense robuste contre les intrusions.

1. Structure du Playbook

- **Nom du Playbook** : Configuration de la sécurité sur Debian 12.
- **Hôtes ciblés** : debian_test, représentant les serveurs devant être sécurisés.
- **Privilèges élevés** : Utilisation de become: yes pour appliquer les configurations.
- **Rôle associé** :
    - security : Regroupe toutes les tâches nécessaires pour la sécurisation du système (protection SSH, Fail2Ban, pare-feu, etc.).

```
 GNU nano 7.2
---
- name: Configuration de la sécurité sur Debian 12
  hosts: debian_test
  become: yes

  roles:
    - security # Configuration de la sécurité du système (pare-feu, utilisateurs, etc.)
```

2. Rôle security

Le rôle security applique des configurations essentielles pour protéger les serveurs Debian 12.

Principales tâches (tasks/main.yml) :

1. **Installation de Fail2Ban :**
    - Installe le paquet fail2ban, un outil de protection contre les attaques par force brute.
2. **Configuration de Fail2Ban :**
    - Utilise le modèle Jinja2 fail2ban.conf.j2 pour personnaliser les paramètres de sécurité.
    - Déploie ce fichier dans /etc/fail2ban/jail.local.
3. **Redémarrage du service :**
    - Redémarre le service Fail2Ban pour appliquer les nouvelles configurations.

ANSIBLE-LOIC

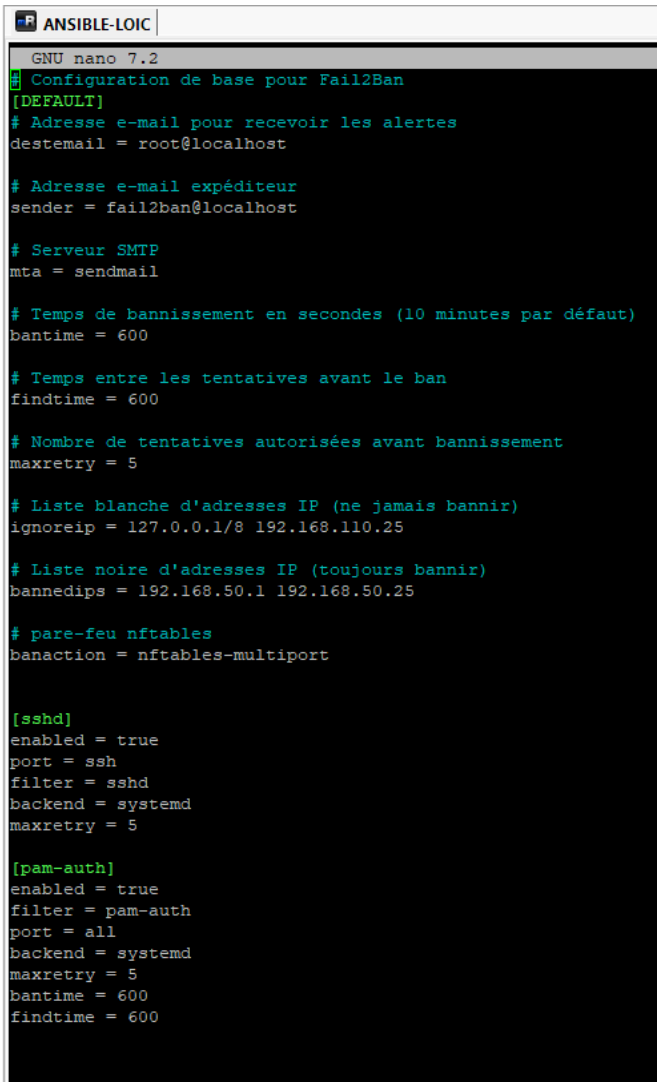
```
 GNU nano 7.2
---
- name: Installer fail2ban
  apt:
    name: fail2ban
    state: present

- name: Configurer fail2ban
  template:
    src: fail2ban.conf.j2
    dest: /etc/fail2ban/jail.local

- name: Redémarrer fail2ban
  service:
    name: fail2ban
    state: restarted
```

3. Modèle fail2ban.conf.j2

Le fichier de modèle Jinja2 est utilisé pour personnaliser la configuration de Fail2Ban, tout en permettant une réutilisation flexible.

Paramètres clés :

1. **Globaux :**
    - **E-mail d'alerte** : Notifications envoyées à root@localhost.
    - **Temps de bannissement** : Défini à 10 minutes par défaut (bantime = 600).
    - **Nombre de tentatives avant bannissement** : Limité à 5 (maxretry = 5).
2. **Listes d'adresses IP :**
    - **Blanche (ignoreip)** : Inclut 127.0.0.1 et 192.168.110.25.
    - **Noire (bannedips)** : Blocage permanent des IP 192.168.50.1 et 192.168.50.25.
3. **Services protégés :**
    - **SSH** : Protège le port 22 en activant les règles de Fail2Ban pour limiter les connexions.
    - **PAM Authentication** : Applique des règles de sécurité similaires.



4. Synthèse

Le playbook security_deb12.yml et le rôle security apportent :

- **Une sécurité renforcée** grâce à Fail2Ban, protégeant SSH et PAM contre les attaques par force brute.
- **Une configuration flexible** via le modèle Jinja2, permettant une adaptation rapide.
- **Une automatisation complète**, limitant les erreurs humaines dans la configuration de la sécurité.

Voici le Tree complet après la mise en place des playbooks et rôles mis en place :

Les trois playbooks principaux sont :

init_deb12.yml :
Initialisation des serveurs Debian 12.

security_deb12.yml :
Sécurisation des serveurs avec Fail2ban.

services_deb12.yml :
Installation et configuration des services Apache2 et MariaDB.

Evolutions possible :

Ansible peut être utilisé pour :

- **Déployer des services critiques**, comme un serveur web ou un serveur de base de données, nécessaires pour la communication et le marketing.
- **Déployer des agents**, comme des agents de serveurs de supervision par exemple.

Ce projet répond directement aux besoins du cahier des charges.
Les tests effectués sont concluants.
La solution est opérationnelle.