République Algérienne Démocratique et Populaire

Ministère de l'Enseignement Supérieure et de la Recherche Scientifique

Université Abderrahmane Mira

Faculté de la Technologie

Département d'Automatique, Télécommunication et d'Electronique

Projet de Fin d'Etudes

Pour l'obtention du diplôme de Master

Filière : Télécommunication

Spécialité : Réseaux et Télécommunications

Thème

Automatisation Avec Ansible

Cas : Entreprise RTC Sonatrach Bejaia

Préparé par :

HAMMOU Ines

OUIZI Sonia

Dirigé par :

Mme GAGAOUA.M MCB UAM

Mr BEKNADJ.D Doctorant

Mr TIAB.A cadre à Sonatrach

Examiné par :

Mr BELLAHSSEN.H MCA UAM

Mr BESSAAD.O MCB UAM

Année universitaire : **2021/2022**

- REMERCIEMENTS –

Avant tout, nous remercions Dieu tout puissant de nous avoir donné le Courage et la patience de terminer ce travail.

Nous tenons à exprimer notre gratitude à nos promoteurs Mme GAGAOUA.M, Et Mr BEKNAJ.D, pour leurs encouragements, et leurs conseils Afin de mener à bien ce travail.

Nos sincères remerciements au personnel de l'entreprise Sonatrach de Bejaïa, et spécialement à Mr TIAB notre encadreur de stage pour sa Patience, ses orientations, et sa grande disponibilité tout au long de notre stage au Sein de l'entreprise.

Nous remercions également, les membres du jury qui ont accepté D'examiner et de juger ce modeste travail.

Sans oublier de remercier tous les enseignants et enseignantes qui, pendant Notre cursus universitaire, ont veillé à notre formation et réussite.

-Dédicace-

Je dédie ce travail et ma profonde gratitude à mon père et ma Mère pour

L'éducation qu'ils m'ont prodiguée, au prix de tous Les sacrifices qu'ils ont

Consentis à mon égard, pour le sens du devoir qu'ils m'ont Enseigné depuis mon enfance,

Pour leur amour, leur soutien et leurs prières tout au long de Mes études.

Puisse Dieu, le Très Haut, vous accorder santé, bonheur et longue vie et Faire en sorte que jamais je ne vous déçoive.

À mon cher futur époux Karim. M qui m'a toujours soutenu, orienté et qui a partagé Avec moi les bons et mauvais moments durant tout mon parcours.

Merci à vous tous et que Dieu vous protège.

Sonia

-Dédicace-

Je dédie ce travail à mes parents auxquels je manifeste ma gratitude pour leur soutien et la confiance qu'ils ont placé en moi, et particulièrement à mon père, qui en tant qu'enseignant m'a toujours guidée et orientée durant mon cursus.

A mes deux sœurs Lina et Lydia pour leur aide et leurs encouragements

A mes amis qui ont rendu mon quotidien à l'université plus agréable.

mon neveu Mylou pour la joie et le bonheur qu'il a apporté au sein de notre famille

Merci à vous tous et que Dieu vous garde et vous protège.

Ines

Table Des Matières

Table Des Figures

Liste Des Tableaux

Liste des Acronymes

A :

API : Application Programmation Interface
APT : Advanced Packages Tool
AWS : Amazon Web Service

B :

BMC : Baseboard Management Controller

C :

CAT : Concatenate
CFG : Configure
CFengine : The Configuration Engine
CLI : Command Line Interface
CMDB : Configuration Management Database

D :

DevOps : Developpement Operations
DHCP : Dynamic Host Configuration Protocol
DNS : Domain Name System
DRGB : Direction Régionale de transport Bejaia
DSA : Digital Signature Algorithm

E :

ECDSA : Elliptic Curve Digital Signature Algorithm

G :

GNOME : GIS Neutral Object Manipulation Engine
GNS3 : Graphical Network Simulator-3

H :

HSRP : Hot Standby Router Protocol
HTTP : Hypertext Transfer Protocol

I :

IA : Intelligence Artificielle
IBM : International Business Machine
INI : INItialize
IOS : Internetwork Operating System
IP : Internet Protocol

J :

JSON : JavaScript Object Notation

K :

KVM : Keyboard, Video and Mouse

L :

LAB : LABoratory
LS : List

M :

MKDIR : MaKe DIRectory
ML : Machine Learning

P :

PING : Packet Internet Groper
PPA : Personal Package Archive

R :

RSA : Rivest, Shamir, Adleman

S :

SLA : Service Level Agreement
SONATRACH : Société Nationale de Transport par Canalisation des Hydrocarbures
SSH : Secure SHell
SU : Super User
SUDO : Super User DO

T :

TelNet : Terminal Network
TRC : Transport par Canalisation

V :

VM : Virtual Machine

W :

WINRM : Windows Remote Management

Y :

YAML : Yet Another Markup Language

INTRODUCTION GENERALE

Introduction générale

AU cours de l'histoire du développement informatique, le déploiement ainsi que la gestion des serveurs de manière fiable, efficace et rapide fut toujours un défi.

Les administrateurs système étaient isolés des développeurs et des utilisateurs qui interagissaient avec les systèmes qu'ils administraient. Ils géraient les serveurs manuellement, installaient des logiciels, modifiaient les configurations et administraient les services sur chaque machine de manière individuelle.

Au fur et à mesure du développement des centres de données et de la complexification des applications hébergées, les administrateurs ont pris conscience de l'intérêt d'une collaboration dans la gestion des systèmes. C'est ainsi que les outils de provisionnement et de gestion de configuration ont commencé à prospérer.

Au lieu de déployer, corriger ou gérer chaque serveur manuellement, les administrateurs ont priorisé l'automatisation de la gestion des serveurs. C'est ainsi que les développeurs d'applications eux-mêmes ont commencé à interagir avec le personnel d'exploitation.

Aujourd'hui les développeurs de logiciels et les professionnels des opérations informatiques sont réunis sous le nom de DevOps. Lorsque les administrateurs travaillent en étroite collaboration avec les développeurs, la vitesse de développement est améliorée et plus de temps est consacré à des activités « attrayante » tel que la mise au point des performances et l'expérimentation.

L'une des premières étapes vers un environnement DevOps consiste à choisir des outils qui peuvent être utilisés à la fois par les développeurs et les ingénieurs d'exploitation.

Quand on parle d'automatisation avec des DevOps, il y a de fortes chances que les noms de Chef et Puppet surgissent. Mais, récemment, le moteur d'automatisation open source le plus cité est celui d'Ansible.

Cette plate-forme peut être utilisée à diverses fins : de la gestion de la configuration au déploiement d'applications procédurales, en passant par l'orchestration multi-composants et

multi-systèmes complexes interconnectés. Elle est sans agent, facile à installer, à configurer et à entretenir.

Les développeurs d'Ansible ont exploité son adoption croissante dans le monde « Open Source » et ont créé des outils pour combler les lacunes dans l'espace d'automatisation informatique. Les prés existants étant compliqués, sujets aux erreurs et difficiles à maîtriser.

Dans de nombreuses entreprises, le travail de développement et d'exploitation est intégré. Cette intégration est une exigence pour la conception des applications modernes pilotées par les tests.

Durant notre stage au sein de l'entreprise RTC Sonatrach de Bejaia et après avoir étudier les différents aspects et besoins de cette entreprise, nous constatâmes que leur architecture réseau est constituée d'une centaine d'équipements. Notre objectif sera donc de concevoir une architecture avec un nœud d'automatisation, qui aura le rôle d'un serveur de gestion, de configuration, d'orchestration et de déploiement.

Au cours de la réalisation de ce mémoire nous allons présenter l'outil d'Automatisation Ansible : son principe de fonctionnement, ses domaines d'application, son intégration et sa contribution au sein d'un réseau informatique.

Ce mémoire est structuré autour de quatre chapitre.

Le chapitre 1, intitulé ''Généralités sur Ansible'', portera sur l'automatisation en général et l'étude de l'outil Ansible et son fonctionnement d'une manière détaillée.

Le chapitre 2, ''Présentation de l'entreprise'', concernera l'entreprise Sonatrach au sein de laquelle nous avons effectué notre stage. Nous allons y présenter sa structure hiérarchique, son réseau informatique ainsi que les différents protocoles utilisés.

Dans le chapitre 3, ''Environnement de travail'', nous allons mettre en place un système d'exploitation adéquat, et y installer les outils dont nous auront besoin pour réaliser la maquette.

Le chapitre 4, « Simulation », sera consacré à l'automatisation des taches à l'aide d'Ansible et nous y expliquerons les divers exemples traité.

Enfin, nous finirons par une conclusion générale récapitulative des points essentiels de notre travail et nous aborderons les perspectives futures.

Chapiter 1:

Généralités sur ANSIBLE

I Généralités sur Ansible

Introduction

Les équipements réseau ainsi que leurs natures hétérogènes ne cesse d'augmenter au fil du temps. Les administrateurs utilisaient des méthodes traditionnelles pour gérer le réseau, ce qui entraînait une configuration plus complexe, moins rapide, exposée aux erreurs humaines et qui nécessite beaucoup de moyens financiers.

L'Automatisation vient résoudre tous ces problèmes pour des centaines de milliers d'utilisateurs dans le monde entier.

DevOps automatise le processus de livraison de logiciel, les changements d'infrastructure, ainsi que la configuration des serveurs et des équipements réseaux.

Un ingénieur DevOps choisit les outils qui peuvent être utilisés par les développeurs et les ingénieurs d'opérations. Ansible est l'outil de choix pour réunir.

La virtualisation est une technique informatique consistant à faire fonctionner plusieurs environnements logiques indépendants séparément sur une même machine. Afin de pourvoir simuler un cas d'automatisation en utilisant l'outil Ansible, il est indispensable de passer par cette technique.

I.1 Automatisation du réseau

I.1.1 Définition

L'automatisation en terme général correspond à l'utilisation de technologies pour effectuer certaines taches avec une intervention humaine réduite.

L'automatisation du réseau quant à elle est le processus d'automatisation de la configuration, de la gestion, des tests, du déploiement et du fonctionnement des périphériques physiques et virtuels au sein d'un réseau. Ce processus permet en outre d'exécuter ces tâches de façon fiable et répétée afin d'en améliorer l'efficacité, de réduire les erreurs humaines, et de diminuer les frais d'exploitation [1].

I.1.2 Utilité de l'automatisation

L'utilité de l'automatisation peut être résumée par les points suivants :

- La planification et la conception du réseau, y compris la planification de scénarios et la gestion des inventaires.
- Le test des équipements et la vérification de la configuration.
- Le provisionnement des équipements et services physiques déployés, ainsi que le déploiement et le provisionnement des équipements virtuels.
- La collecte de données réseaux relatifs aux équipements, systèmes, logiciels, topologies réseau, trafic et services en temps réel.
- L'analyse des données, y compris l'analyse prédictive basée sur l'IA et le ML, pour déterminer le comportement actuel et futur du réseau.
- La conformité de la configuration, qui garantit le bon fonctionnement de tous les équipements et services réseau.
- La mise à jour des logiciels, y compris la restauration si besoin.
- La résolution en boucle fermée des incidents réseau, y compris des pannes et des défaillances masquées (« grises »).
- La production de données de rapports, de tableaux de bord, d'alertes et d'alarmes
- L'application des règles de sécurité.
- La surveillance du réseau et de ses services pour garantir le respect des SLA et la satisfaction des clients.

I.1.3 Les outils d'automatisation

Il existe 4 domaines d'application dans le monde de l'automatisation comme illustré dans la Figure I.1, chaque domaine utilise ses propres outils cités ci-dessous.

- L'orchestration
  - BMC
  - Mcollective
  - Chef Metal
  - Ansible
- Déploiement d'application
  - Fabric

- Capistrano
- Nolio
- Ansible

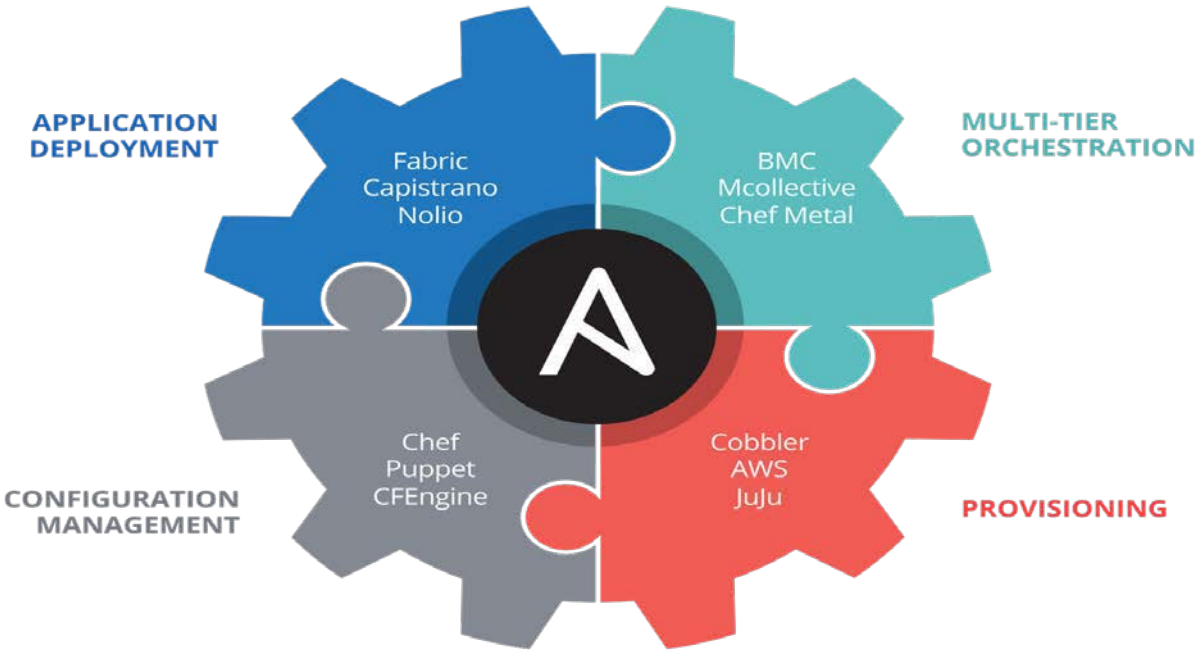

Figure I.1 : Domaines d'applications de l'automatisation [F2]

- Provisionnement
  - Cobbler
  - AWS
  - JuJu
  - Ansible
- Gestion des configurations
  - Puppet
  - Chef
  - CFengine
  - Ansible

Ansible est l'outil qui automatise tous les domaines d'application, il est utilisé dans de nombreuses entreprises connues tel que Udemy, Alibaba Travel, Tokopedia…

I.1.4 Avantage de l'automatisation

L'automatisation présente plusieurs avantages dont on peut citer :

- La réduction des couts.
- La réduction du nombre d'incident.
- L'amélioration des analyses et du contrôle du réseau.
- La réduction du temps d'arrêt du réseau.

I.2 Ansible

I.2.1 Définition

Ansible est un outil d'automatisation informatique Open Source qui automatise le provisionnement, la gestion des configurations, le déploiement des applications, l'orchestration et bien d'autres processus informatiques.

Il fonctionne sur de nombreux systèmes de type Unix et peut configurer aussi bien des systèmes de type Unix que Microsoft Windows. Il comprend son propre langage déclaratif pour décrire la configuration du système.

Ansible est sans agent, c'est-à-dire qu'il ne nécessite l'installation d'aucun logiciel supplémentaire sur les nœuds à gérer. Il se connecte temporairement à distance via SSH ou Windows Remote Management (permettant l'exécution à distance de PowerShell) pour effectuer ses tâches.[3]

Bien qu'il existe de nombreux outils d'automatisation, Ansible a réussi à se démarquer par sa simplicité, sa sécurité et surtout sa courbe d'apprentissage fluide.

I.2.2 Historique d'ansible

‘’Ansible” est un dispositif théorique permettant de réaliser des communications à une vitesse supraluminique (supérieure à la vitesse de la lumière) imaginé en 1966 par Ursula K.

Son créateur est Michael DeHaan ; la première version de Ansible date de 2012. Le nom Ansible est tiré d'un roman de science-fiction écrit par Ursula Le Guin, qui désigne un moyen de communication plus rapide que la lumière.

Entre-temps, Ansible a été racheté en 2015 par Red Hat qui a été racheté par IBM (international business machines) en 2018. Donc, Ansible appartient désormais à IBM.[4]

I.2.3 Domaine d'application

Ansible est capable d'automatiser divers processus informatiques, tels que :

- **La gestion des configurations**

  La gestion des configurations est un processus qui permet de maintenir les systèmes informatiques, les serveurs et les logiciels dans l'état souhaité et d'en préserver la cohérence. C'est une façon de s'assurer qu'un système fonctionne comme prévu au fil des changements effectués.

  Par exemple elle peut configurer un serveur, puis créer ses systèmes et en assurer le bon fonctionnement.

  Ansible nous permet d'accélérer les changements et les déploiements, de nous éviter les risques d'erreur humaine et de rendre la gestion des systèmes plus prévisible et évolutive. De plus, il permet de suivre l'état des ressources et évite de répéter des tâches, telles que l'installation d'un même paquet deux fois.

- **L'orchestration**

  L'orchestration permet de décrire comment automatiser un processus constitué de nombreuses étapes réalisées sur plusieurs systèmes différents.

  Ansible permet d'orchestrer le déploiement en exécutant les tâches du playbook dans l'ordre dans lequel elles ont été rédigées, tout en nous garantissant que le processus se déroule comme prévu.

- **Le déploiement d'application**

  L'automatisation du déploiement permet de déplacer les logiciels entre les environnements de test et de production à l'aide de processus automatisés. Ainsi, elle assure la reproductibilité et la fiabilité des déploiements tout au long du cycle de distribution.

- **Le provisionnement**

  La première étape de l'automatisation du cycle de vie des applications consiste à automatiser le provisionnement des infrastructures. Avec Ansible, on peut provisionner des plates-formes cloud, des hôtes virtualisés, des périphériques réseau et des serveurs barre métal [5]**.**

I.2.4 Langage et protocole

Techniquement, Ansible se base sur les langages ou protocoles suivants :

- La plate-forme et les modules sont développés en langage Python
- Le langage YAML (Yet Another Markup Language) est un langage de sérialisation des données qui est souvent utilisé pour coder des fichiers de configuration. Ce qui signifie qu'il s'utilise pour représenter des données plutôt que des documents. Il est utilisé pour la partie déclarative notamment le playbook et l'inventaire.
- Le protocole SSH (secure shell) est enfin utilisé pour la communication avec les machines-cibles.

I.2.5 Principe de fonctionnement

Le fonctionnement d'ansible se base principalement sur deux types de machines :

- Le serveur Ansible : couramment appelé la machine de contrôle ou control Node, c'est la machine ansible.
- Les Hôtes : appelés aussi nœuds ce sont les machines sur lesquelles Ansible effectuera des taches.

Ansible fonctionne en se connectant aux hôtes ou aux réseaux en SSH et en y poussant de petits programmes, appelés modules. Ces modules sont définis dans un fichier nommé le Playbook. Le Noeud de contrôle se base sur un fichier d'inventaire qui fournit la liste des hôtes sur lesquels les modules Ansible doivent être exécutés.

Selon le diagramme de la figure I.2, le moteur d'automatisation Ansible a une interaction directe avec les utilisateurs qui écrivent des Playbooks pour exécuter le moteur d'automatisation Ansible. Il interagit également avec les services cloud et la base de données de gestion de configuration (CMDB) dans le cas d'un inventaire dynamique.

1. Users

Représente la machine de contrôle. C'est la machine ou est installé ansible et sur laquelle est créer l'inventaire et le Playbook, pour y pousser des taches aux hôtes via une connexion SSH. Cette machine doit être une machine linux où le langage python 3 (ou ultérieur) y est installé.

2. Hosts

Appelé aussi nœuds gérés ce sont les machines sur lesquelles Ansible viendra pousser les tâches d'automatisation. Il n'est pas nécessaire d'installer Ansible sur les nœuds gérés le seul prérequis est d'installer SSH sur les équipements ainsi que le langage python 3.

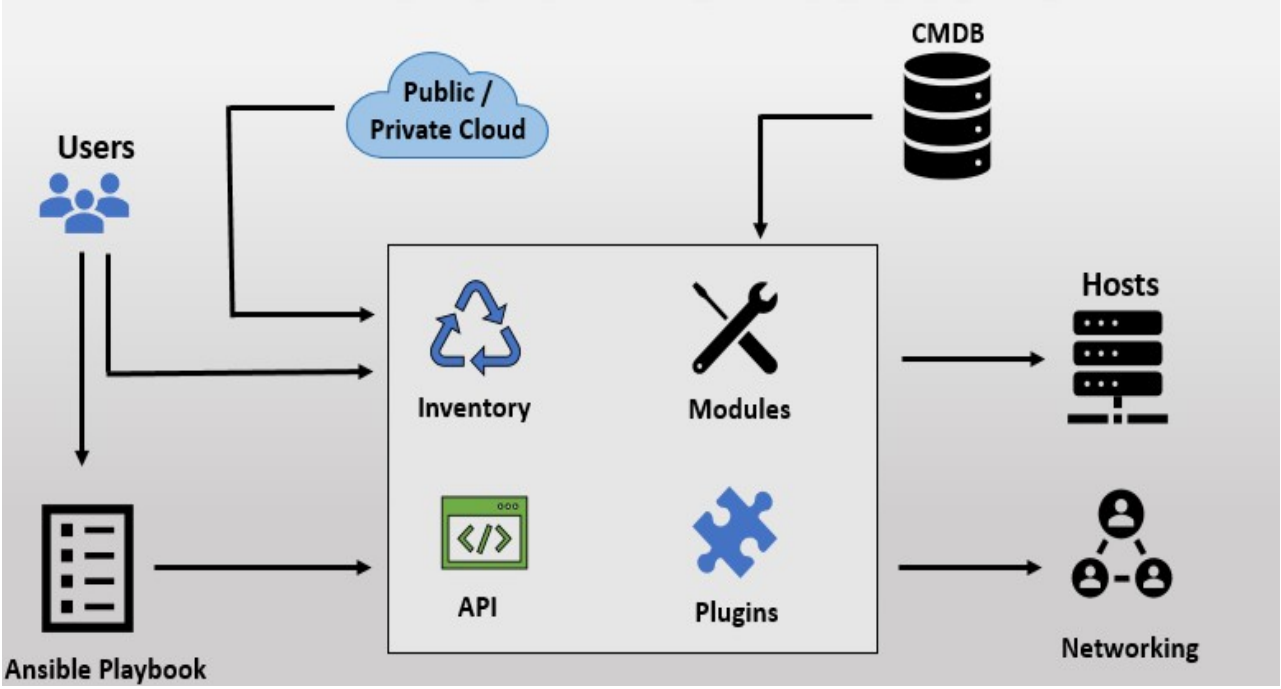

Figure I.2 : Principe de fonctionnement d'Ansible

3. Networking

Ansible peut également être utilisé pour automatiser différents réseaux. Ansible utilise le même cadre d'automatisation simple, puissant et sans agent que les opérations et le développement informatiques utilisent déjà. Il utilise un modèle de données (un Playbook) distinct du moteur d'automatisation Ansible qui couvre facilement différents matériels réseau.

4. Cloud

C'est un réseau de serveurs distants hébergés sur Internet pour stocker, gérer et traiter des données, plutôt qu'un serveur local. Il est possible de lancer les ressources et instances sur le cloud pour se connecter aux serveurs.

5. Inventaire

C'est un fichier écrit par défaut en format INI mais il peut aussi être écrit en format YAML. Il contient principalement le nom des hosts et leurs adresses IP afin de les identifier pour y faire appel dans un Playbook ou des commandes ad-hoc.

Un inventaire peut être créer statiquement avec une collection d'hôtes définis comme expliqué auparavant, ou créer dynamiquement grâce à un script dynamique interrogeant le CMDB. Le fichier de l'inventaire se trouve par défaut dans /etc/ansible/hosts.

Il est possible d'ordonner un inventaire par groupes, par alias ou simplement avec des adresses IP, comme le montre les captures d'écran ci-dessous.

- Exemple d'inventaire établi uniquement avec des adresses IP et des noms d'hôtes.

Figure I.3 : Création d'un inventaire avec adresse IP.

- Exemple d'inventaire organisé en groupes

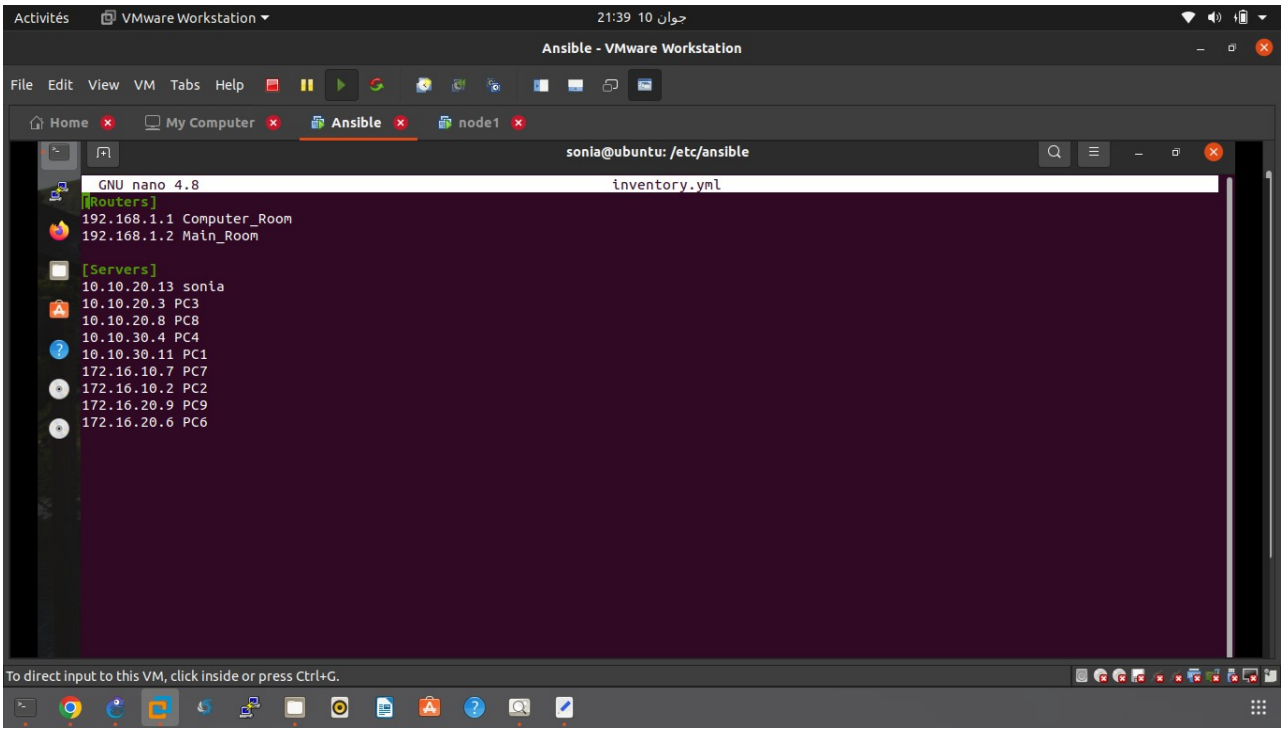

Figure I.4: Création d'un inventaire par groupe.

- La figure I.5 représente un inventaire avec alias. Lors de l'utilisation des alias, il faut suivre une certaine syntaxe telle que : nom-d'hote ansible_host = adresse IP

Figure I.5 : Création d'un inventaire par alias

6. CMDB

Est un référentiel qui agit comme un entrepôt de données pour les installations informatiques. Il contient des données relatives à une collection d'actifs informatiques (communément appelés éléments de configuration CI), et décrit les relations entre ces actifs.

7. Module

Les modules forment les outils de la boite à outils. Ceux sont des programmes ou des ensembles de programmes Ansible similaires destinés à être exécutés côté client. Un module est généré sur la machine de contrôle, il est copié, exécuté et effacé sur les nœuds gérés (dans le cadre de l'automation des serveurs). Par contre, quand il s'agit de gérer des périphériques du réseau, comme des commutateurs (switches) ou des routeurs, un module est généré et exécuté localement sur le contrôleur pour agir sur la cible.

Il est possible de faire appel à un module ansible directement à partir de l'interface en ligne de commande, sans passer par un playbook. On appelle ceci une « commande ad-hoc »

Ansible exécute ces modules en SSH grâce au protocole JSON sur la sortie standard et les supprime une fois terminer.

8. Tasks

C'est une instruction écrite en YAML qui fait appel à un module ansible.

9. Roles

Un rôle est une structure arborescente constituée de répertoires et de fichiers de configuration YAML, ayant pour fonction l'installation d'un système. Les rôles peuvent être imbriqués et interdépendants les uns des autres. Ils contiennent plusieurs taches (Tasks) et permettent de les organiser.

10. Plugins

Les plugins permettent d'exécuter des tâches Ansible en tant qu'étape de génération de travail. Les plugins sont des morceaux de code qui augmentent les fonctionnalités de base d'Ansible.

Il existe plusieurs types de plugins, qui sont des sortes de "modules" spécialisés. On peut en citer :

- connexions : ssh, winrm, …
- inventory : ini, yaml, scripts, liste, …
- become : su, sudo, enable, runas

11. Playbook

Un playbook décrit une suite de Taches ou de Rôles écrits dans un fichier en format YAML. Il constitue la convergence des hôtes et des tâches. Enfaite, c'est ici que nous définissons les actions que nous devons accomplir avec ansible. Il contient principalement le groupe d'hôtes concerné en faisant appel à l'inventaire, ainsi qu'un jeu de rôle (une suite de taches ou bien de rôles) avec les modules ansible. L'exécution des taches dans un playbook se fait selon leurs ordres de déclaration.

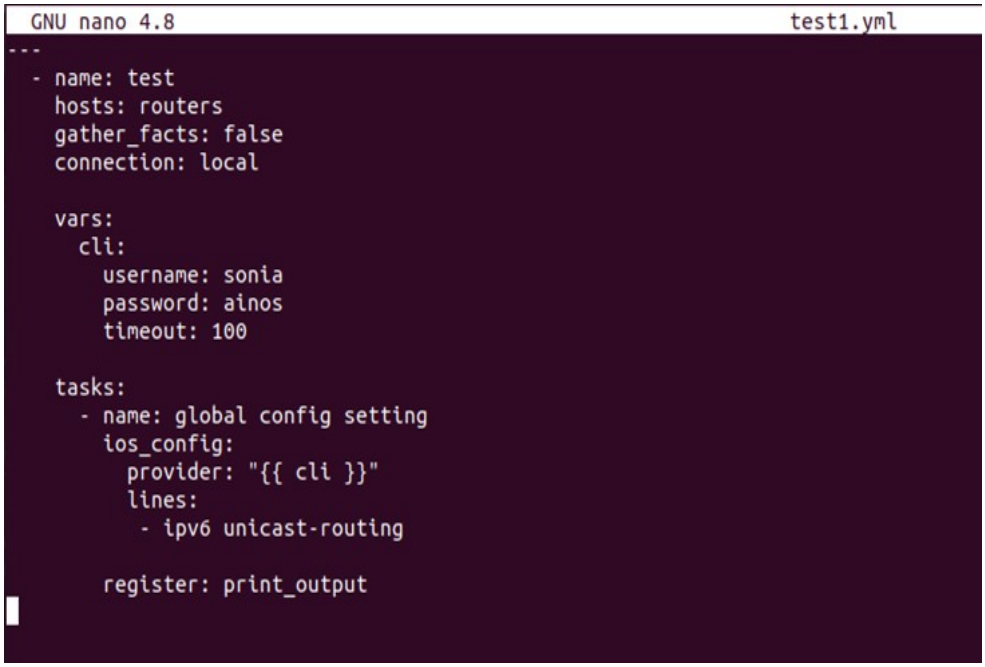

Figure I.6: Représentation d'un exemple de playbook

12. API

L'interface de programmation d'application (API) est utilisée dans Ansible comme solution de communication avec les services Cloud, publics ou privés.

13. Variable

Les variables dans Ansible sont exactement les mêmes que dans les autres langages. On considère une variable comme un 'nom' attaché à un 'objet' spécifique. Il est possible de définir ces variables dans des playbooks, dans un inventaire, dans des fichiers ou rôles réutilisables, ou en ligne de commande. Pour créer une variable, il suffit de lui attribuer une valeur, par exemple, syslog_ip=10.10.5.20 pour ensuite se référer à une adresse IP n'importe où dans Ansible en utilisant la clé syslog_ip plutôt que l'adresse IP.

14. Collection

Les "collections" sont un format de distribution pour du contenu Ansible qui peut inclure des livres de jeu, des rôles, des modules et des plugins. Les "collections" sont distribuées par Ansible Galaxy via ansible-Galaxy collection Install
<namespace.collection>*

I.2.6 Option de base pour l'utilisation d'Ansible :

Lorsqu'on lance une instruction en ligne de commande on fait appel à certain mots clé à connaître.
Le tableau I.1 résume les plus utilisés d'entre eux :

-u	User distant utilisée
-b	Passer les commandes en élévation de privilèges (sudo)
-k / --ask-pass	Password SSH
-K / --ask-become-pass	Password pour élévation de privilèges
-C / --check	Faire un dry run
-D /-diff	Avoir un outpout de la différence
--key-file	Lien direct vers la clef privée
-e / --extra-vars	Définir des variables
--ask-vault-pass	Déchiffrer un secret vault
--vault-password-file	Fichier pour déchiffrer
-m	Module

-i	Inventaire
-f	Paralléliser
-vvv	verbose

Tableau I.1: Option de base d'ansible

I.2.7 Avantages et inconvénients

I.2.7.1 Avantages

Parmi les points forts d'Ansible on peut citer les qualificatifs suivants : [6]

- **Gratuit** : Ansible est un outil open-source.
- **Très simple à configurer et à utiliser** : Aucune compétence particulière en codage n'est nécessaire pour utiliser les playbooks d'Ansible.
- **Puissant** : Ansible permet de modéliser des flux de travail informatique même très complexes.
- **Flexible** : il a la possibilité d'orchestrer l'ensemble de l'environnement applicatif, quel que soit l'endroit où il est déployé. Comme il est possible de le personnaliser en fonction des besoins.
- **Sans agent** : il n'est pas nécessaire d'installer d'autres logiciels ou ports de pare-feu sur les systèmes clients à automatiser ainsi que de mettre en place une structure de gestion distincte.
- **Langage** : le langage python est très facile à apprendre et à comprendre par l'homme, ainsi que l'utilisation du format YAML dans l'inventaire et le playbook.

I.2.7.2 Inconvénients

Pour les inconvénients on peut relever :

- **Prise en charge limitée de Windows** : Dans le cas de Windows, Ansible utilise une communication à distance PowerShell native plutôt que SSH. Par conséquent, une machine de contrôle Linux est obligatoire pour la gestion des hôtes Windows.
- **Nouveau sur le marché de l'automatisation** : en effet ansible est un nouvel outil d'automatisation comparé à son principal concurrent. En conséquence nous risquons de rencontrer de nouveaux problèmes non traités auparavant.

I.3 La virtualisation

I.3.1 Définition

La virtualisation est l'ensemble des techniques matérielles et logicielles permettant de fournir un ensemble ou un sous-ensemble de ressources informatiques de manière qu'elles puissent être utilisées, indépendamment de la plateforme matérielle [7]. Il existe cinq types de virtualisation comme le montre la figure I.7 ci-dessous :

- La virtualisation des serveurs.
- La virtualisation d'application.
- La virtualisation de postes de travail.
- La virtualisation de stockage.
- La virtualisation du réseau.

Figure I.7 : Les différents types de virtualisation [F8].

I.3.2 La virtualisation des serveurs

La virtualisation des serveurs/systèmes d'exploitation permet d'exécuter sur une seule et même machine plusieurs systèmes d'exploitation différents. L'hyperviseur crée un environnement virtuel complet simulant littéralement un nouvel ordinateur complet, avec du "matériel fictif". À quelques rares exceptions, le système d'exploitation invité (installé dans la machine virtuelle) ne communique qu'avec ce faux matériel simulé, rendant étanche l'environnement virtualisé.[9]

La virtualisation des systèmes d'exploitation permet de lancer une application nécessitant un autre système d'exploitation que celui de la machine physique, ou une version antérieure. La figure I.8 résume le principe de la virtualisation des serveurs.

Il existe de nombreux hyperviseurs de virtualisation :

- VirtualBox
- VMWare Workstation Pro/Player
- GNOME machine
- KVM

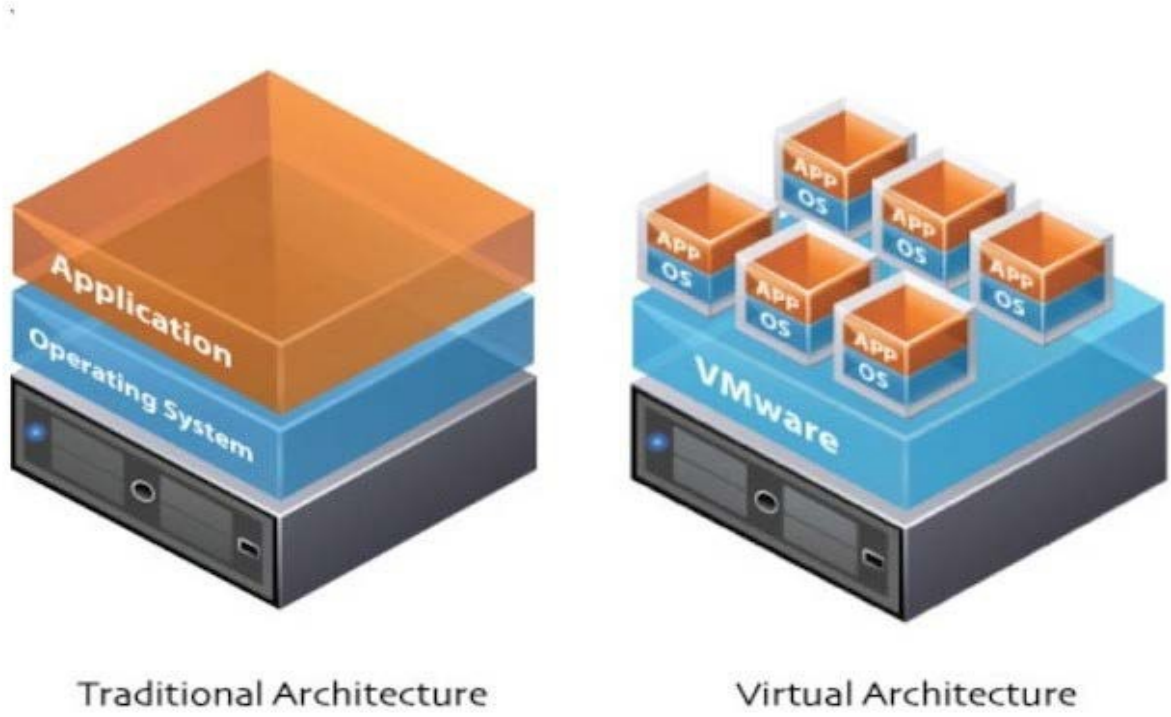

Figure I.8: Principe de la virtualisation des systèmes d'exploitation

I.3.3 Présentation de VMWare Workstation Pro

VMWare Workstation Pro est un outil de virtualisation créé par la société VMware, il met en place un environnement de test pour développer de nouveaux logiciels, ou pour tester l'architecture complexe d'un système d'exploitation avant de l'installer réellement sur une machine physique. Il Est la norme de l'industrie pour l'exécution de plusieurs systèmes d'exploitation sur un seul PC Linux ou Windows [10].

I.4 Conclusion

Dans ce chapitre, nous avons d'abord établi une vue générale sur les points essentiels de notre projet. Ensuite, une vue globale sur les différents domaines et outils de l'automatisation a été présenté. Enfin, nous nous sommes approfondis sur le cœur de notre sujet ; Ansible, en expliquant son principe de fonctionnement et pourquoi nous avons fait le choix de cet outil.

CHAPITRE II :

PRESENTATION DE L'ENTREPRISE

II Présentation de l'entreprise

Introduction

Afin d'enrichir nos connaissances et de pouvoir mettre en œuvre tout ce que nous avons eu l'occasion de voir durant notre cursus dans le domaine des réseaux et télécommunications. Nous avons suivi un stage pratique au département système informatique au sein de l'entreprise Sonatrach Bejaia.

Premièrement, nous ferons une présentation globale de l'entreprise tout en exposant leur architecture.

Ensuite, nous allons citer les protocoles utilisés au sein de l'entreprise ainsi que la définition de ceux qui feront l'objet de notre travail.

Enfin, nous proposerons une contribution à l'amélioration et au développement du réseau.

II.1 Présentation de l'organisme d'accueil

SONATRACH qui est l'acronyme de **SO**ciété **NA**tionale pour le **TR**Ansport par **C**analisation des **H**ydrocarbures, est une entreprise publique algérienne et un acteur majeur de l'industrie pétrolière. C'est une compagnie nationale d'envergure internationale, c'est le pilier de l'économie algérienne. Le groupe pétrolier et gazier SONATRACH intervient dans l'exploration, la production, le transport par canalisation, la transformation et la commercialisation des hydrocarbures et de leurs dérivés.

II.1.1 Activités de base et missions

Les activités de base de SONATRACH ont été fixé en 1992 afin qu'elle atteigne ses objectifs constitués par :

- L'exploitation et la recherche.
- L'exploitation des gisements d'hydrocarbures.
- La liquéfaction et la transformation du gaz.
- Le transport par canalisation.
- La commercialisation.

Pour la réalisation de ces objectifs, l'entreprise est divisée en 5 branches représentées sur la figure II.1.

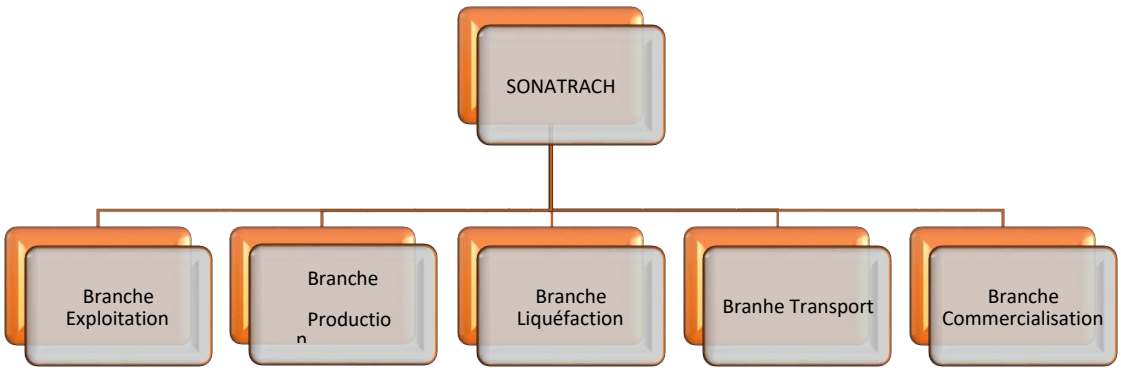

Figure II.1: Organigramme de la SONATRACH en Algérie.

A travers cette transformation structurelle et fractionnelle, le schéma du groupe a évolué en constituant des branches d'activités autonomes avec leurs filiations. Dans la branche « Activité de transport par canalisation » se trouve la Direction Régionale de Bejaia (DRGB) où s'est déroulé notre stage pratique.

II.1.2 Activités de la branche de transport par canalisation

L'activité de transport par canalisation (TRC) est en charge de l'acheminement des hydrocarbures pétroles brut, gaz et condensat vers les ports pétroliers, les zones de stockages et les pays d'exploitation.

Les missions affectées à la branche transport par canalisation sont :

- La gestion et l'exploitation des ouvrages et canalisations de transport d'hydrocarbures.
- La coordination et le contrôle de l'exécution des programmes de transport arrêtés en fonction des impératifs de production et de commercialisation.
- La maintenance, l'entretien et la protection des ouvrages et canalisation.
- L'exécution des révisions générales, des machines tournantes et équipements.
- Les installations de pompage et de stockage pour répondre aux besoins de SONATRACH dans les meilleures conditions d'économie, de qualité, de sécurité et de respect de l'environnement.

- La conduite des études, la réalisation et la gestion des projets de développement des ouvrages et canalisations.
- Gère l'interface transport des projets internationaux du groupe ou en partenariat.

❖ **Organigramme TRC**

La SONATRACH possède cinq directions régionales de transport des hydrocarbures :

- La direction régionale Est (Skikda).
- La direction régionale Centre (Bejaïa)
- La direction régionale Ouest (Arzew).
- La direction régionale de Haoud-EL-Hamra
- La direction régionale d'Ain Amenas.

II.1.3 Présentation de la direction régionale de transport de Bejaia (DRGB)

La DRGB est l'une des cinq directions régionales de transport des hydrocarbures de la SONATRACH (TRC). Elle a pour mission de transporter, stocker et livrer les hydrocarbures liquides et gazeux. Elle est chargée de l'exploitation de deux oléoducs, d'un gazoduc et d'un port pétrolier.

❖ **Structure de la DRGB**

La direction régionale de Bejaia comporte plusieurs constituants illustrés par l'organigramme de la figure II-2.

❖ **Présentation des services**

- **Direction régionale** : Elle est dirigée par un directeur régional aidé par des assistants et un secrétariat.
- **Assistant de sécurité interne** : Sa mission est de protéger et de sauvegarder le patrimoine humain et matériel de la DRGB.
- **Centre informatique** : Il regroupe les moyens d'exploitation et de développement des applications informatiques pour l'ensemble des structures de la DRGB, ainsi que la gestion du réseau informatique interne.

- **Sous-direction Technique** : Elle a pour mission d'assurer la maintenance et la protection des ouvrages. Elle est organisée en quatre départements : département maintenance, département protection des ouvrages, département approvisionnement et transport et département des travaux neufs.

- **Sous-direction Exploitation** : Elle est chargée de l'exploitation des installations de la région, et de maintenir le fonctionnement des trois ouvrages en effectuant des réparations en cas de fuite, de sabotage ou de panne pour les stations de pompage. Elle est composée de deux départements : le département exploitation liquide et le département exploitation gaz.

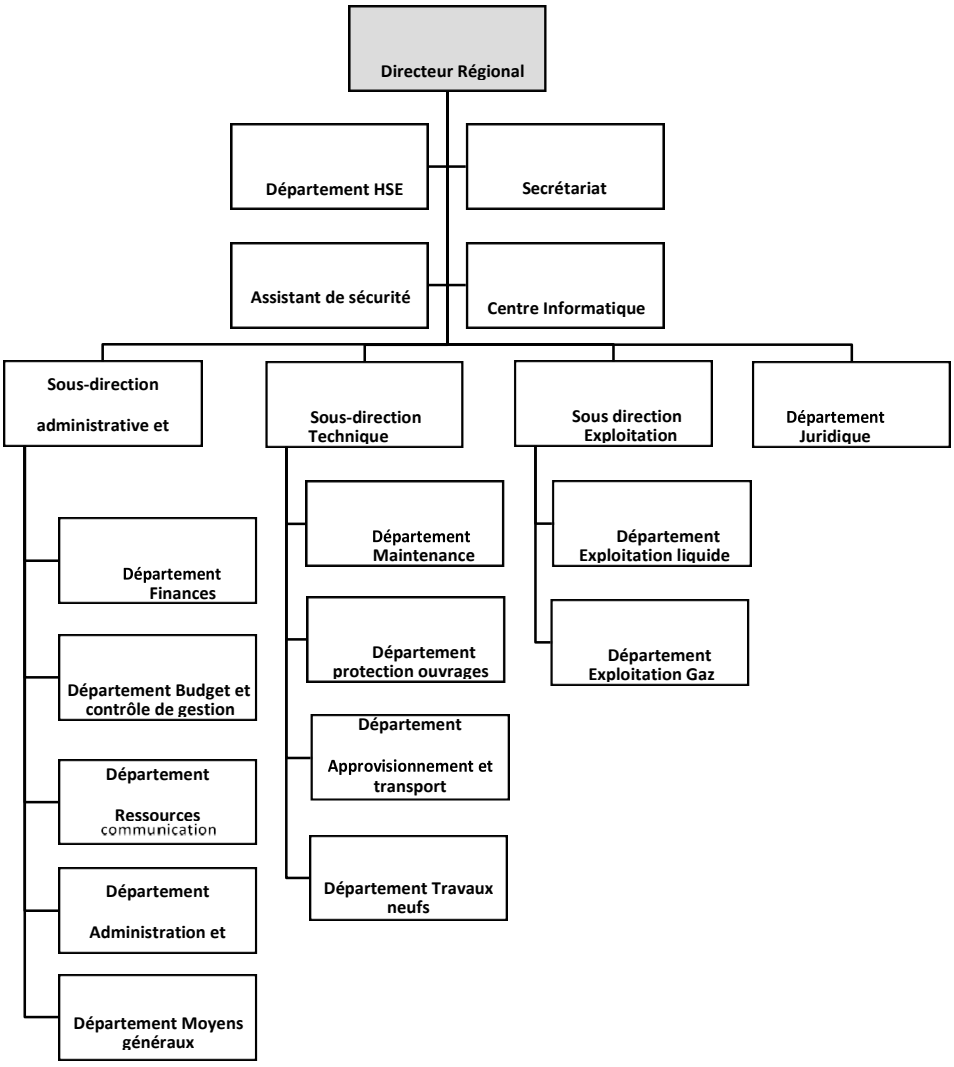

Figure II.2: Structure de la DRGB.

- **Sous-direction Administrative et Finances** : Elle a pour mission la gestion des ressources humaines et les moyens généraux, ainsi que d'effectuer la gestion financière, le budget et le contrôle de gestion. Elle est organisée en 5 départements : département administration et social, département ressources humaines et communication, département moyens généraux, département finances, département budget et contrôle de gestion.
- **Département juridique** : Il prendre en charge les affaires juridiques de la DRGB.

II.1.4 Présentation du centre informatique

Le centre informatique est chargé du développement et de l'exploitation des applications informatiques de gestion pour le compte de la direction régionale de Bejaïa (DRGB) et des autres régions.

II.1.5 Organisation structurelle

L'organisation du centre ne cesse de subir des changements et l'évolution rapide de l'informatique, pousse le centre à adopter des actions nouvelles à chaque fois afin de subvenir aux nouveaux besoins de l'entreprise.

Le centre informatique se constitue de trois services gérés par un chef de centre. Ces derniers sont illustrés par le diagramme de la figure II.3.

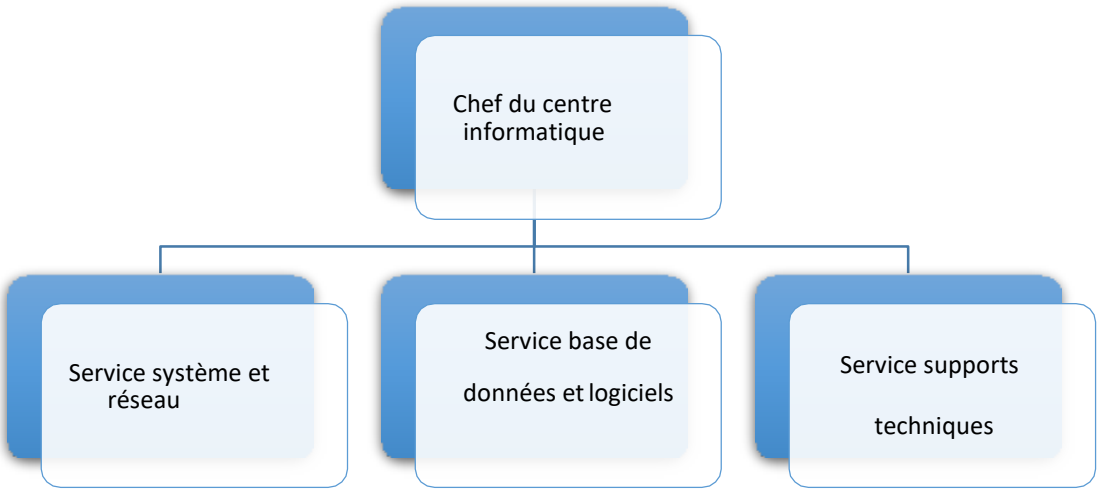

Figure II.3: Organigramme du centre informatique.

II.1.5.1 Organisation fonctionnelle

Chaque service a une mission spécifique. Ces différentes tâches peuvent être présentées de la manière suivante :

❖ **Service système et réseaux**

- Système
  - Choix des équipements informatique et logiciel de base.
  - Mettre en œuvre les solutions matériels et logiciels retenues.
  - Installation et configuration des systèmes.
  - Orienter les travaux et l'équipe de développement par une bonne utilisation des ressources de l'ordinateur.
  - Mise en œuvre des nouvelles versions de logiciels.
- Réseau
  - Conduite de l'étude pour le choix de l'architecture du réseau à installer.
  - Participer à la mise en place des réseaux.
  - Définir les droits d'accès à l'utilisation du réseau.
  - Assure la surveillance permanente pour détecter et prévenir les pannes.
  - Traitement des dysfonctionnements et incidents survenant sur le réseau.
  - Assure le bon fonctionnement, la fiabilité des communications, l'administration du réseau et organise l'évolution de sa structure.

❖ **Service base de données et logiciels**

- Base de données
  - Conçoit les bases de données et assure l'optimisation et le suivi de la gestion des données informatiques.
  - Met en œuvre et gère les procédures de sécurité (accès, integrité).
  - Installe, configure et exploite le SGBD et ses bases.
  - Gère la sauvegarde, la restauration et la migration des données.
  - Assure la cohérence et la qualité des données introduites par les utilisateurs.
- Logiciels
  - Etude et conception des systèmes d'information.
  - Développement et maintenance de l'application informatique pour TRC.
  - Déploiement des applications et formation des utilisateurs.

- Service supports techniques
  - Assistance aux utilisateurs en cas de problèmes software et hardware
  - Installation des logiciels technique et bureautique.
  - Formation aux nouveaux produits installés

II.2 Le réseau informatique de l'entreprise

II.2.1 Principaux protocoles utilisés

- **DNS :** le protocole et système DNS (Domaine Name System) a été inventé pour faciliter la recherche d'un site donné sur internet. Il permet d'associer un nom compréhensible, à une adresse IP. On associe donc une adresse logique, le nom de domaine, à une adresse physique l'adresse-IP.

  Le nom de domaine et l'adresse IP sont uniques. Le DNS permet d'atteindre le destinataire voulu. Il permet par exemple de taper www.google.com sans saisir une longue adresse IP.

- **DHCP :** DHCP (Dynamic Host Configuration Protocol) est un protocole de communication. Les administrateurs réseau l'utilisent pour gérer et automatiser de manière centralisée la configuration des appareils sur un réseau IP (Internet Protocol). Le protocole DHCP permet à des appareils nécessitant une adresses IP d'en recevoir au démarrage. Cette approche évite la configuration manuelle d'une adresse IP pour chaque appareil.

- **SSH** : La sécurité joue toujours un rôle majeur sur Internet. C'est pourquoi le protocole de sécurité SSH (secure shell) est fermement intégré dans la pile de protocoles TCP/IP. SSH permet à deux ordinateurs d'établir une connexion directe et sécurisée au sein d'un réseau potentiellement non sécurisé, tel qu'Internet. Ceci est nécessaire pour que des tiers ne puissent pas accéder au flux de données et que les données sensibles ne tombent entre de mauvaises mains. Il chiffre la connexion entre deux ordinateurs et permet d'utiliser un deuxième ordinateur à partir d'un seul.

  SSH possède de nombreuses applications possibles. On peut citer à titre d'exemple :
  - La gestion des serveurs auxquels il n'est pas possible d'accéder localement.
  - Le transfert de fichiers sécurtisé.
  - La création sécurisée de sauvegardes.
  - La connexion entre deux ordinateurs utilisant le chiffrement de bout en bout.
  - La télémaintenance d'autres ordinateurs.

- **HSRP :** Le protocole HSRP (Hot Standby Routing Protocol) est le protocole propriétaire Cisco permettant d'obtenir une continuité de service LAN sur les routeurs. Il est utilisé pour assurer une disponibilité accrue de la passerelle d'un réseau. L'adresse IP de la passerelle est configurée sur deux routeurs différents. Une seule de ces deux interfaces est active. Si l'interface active n'est plus accessible, l'interface passive devient active.

II.2.2 Architecture réseau de l'entreprise

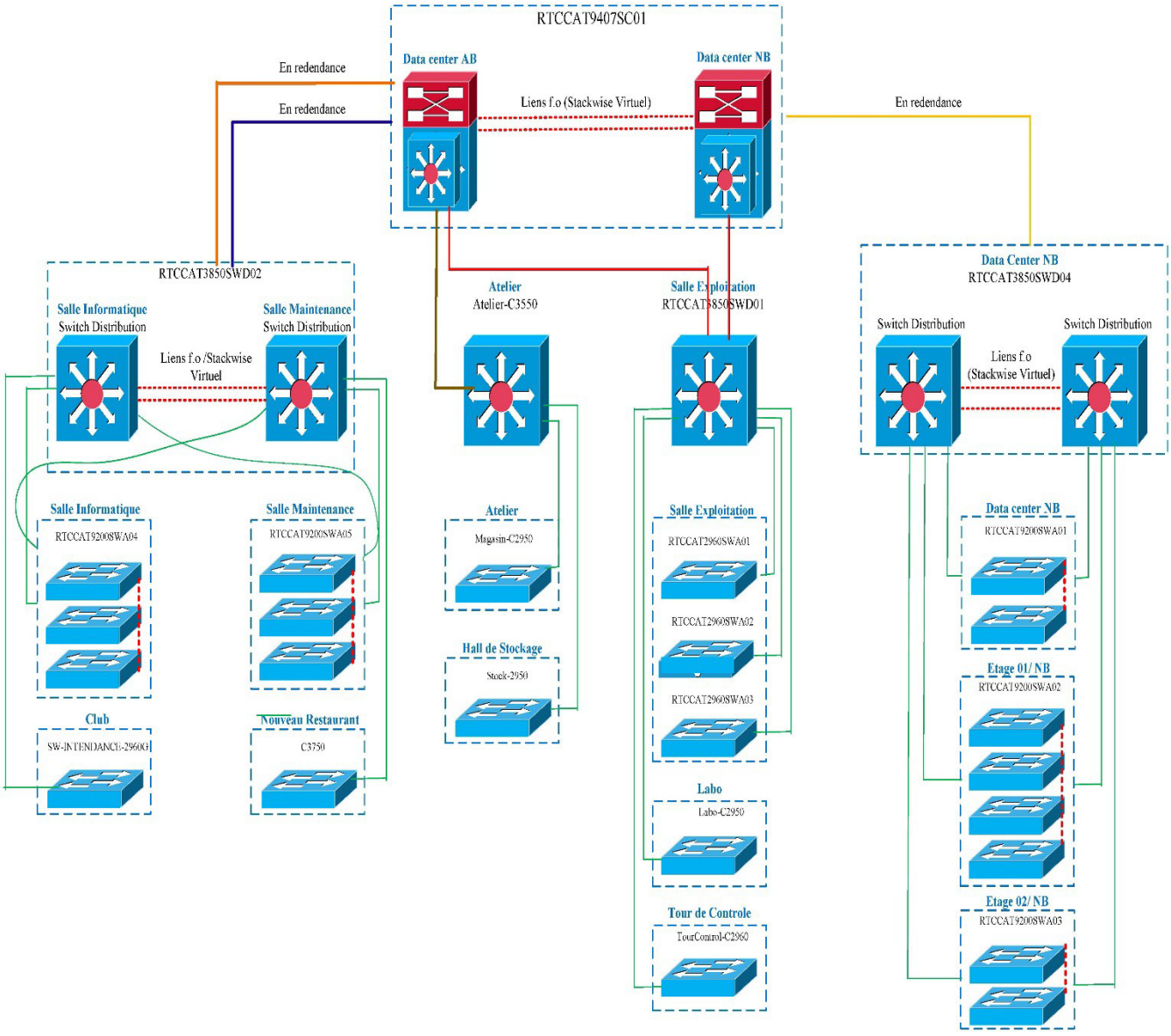

Figure II.4: Architecture réseau de l'entreprise.

II.3 Automatisation avec Ansible

La mise en œuvre de l'architecture physique et logique du réseau de Sonatrach subit constamment des changements afin d'être plus efficace.

L'automatisation de ce dernier semble une solution durable pour l'entreprise afin de pouvoir effectuer diverses taches sur le réseau d'une manière plus rapide, plus fiable et avec moindre cout.

Cette solution fera l'objet de notre travail durant notre stage pratique au sein de l'entreprise Sonatrach de Bejaia.

CHAPITRE III :

ENVIRONNEMENT DE TRAVAIL

III Environnement de travail

Introduction

Ce chapitre consistera en la préparation de notre environnement, en expliquant toutes les étapes qui contribuerons au bon fonctionnement de notre projet.

Ubuntu qui est un système d'exploitation de type Unix, très connu dans le monde de la programmation sera notre système d'exploitation principal.

D'abord, nous aurons besoin d'y installer principalement VMWare Workstation pro qui nous permettra d'installer des machines virtuelles, afin de pouvoir gérer différents nœuds. Ainsi que GNS3 (Graphical Network Simulator 3) qui est utilisé pour émuler plusieurs systèmes d'exploitation dans un environnement virtuel à l'aide des systèmes d'exploitation Cisco Inter-network. Nous pourrons donc y architecturer notre réseau et le configurer.

Ensuite, nous expliquerons toutes les configurations de bases de notre réseau, les logiciels ainsi que les commandes nécessaires afin de connecter toutes les machines et tous les équipements entre eux.

Enfin, nous montrerons comment installer l'outil d'automatisations de notre projet.

III.1 Installation des Packages APT

Lorsque nous débutons sur UBUNTU il est important de mettre à jour et d'installer les packages APT qui nous seront utiles dans nos prochaines opérations. **A**dvanced **P**ackaging **T**ool est un système complet et avancé de gestion de paquets, permettant une recherche facile et efficace, une installation simple et une désinstallation propre de logiciels et utilitaires.

Il permet également de faciliter la mise à jour de la distribution Ubuntu avec les paquets en versions les plus récentes et de passer à une nouvelle version de Ubuntu, lorsque celle-ci est disponible.

Les commandes à implémenter pour charger APT se présentent comme suit :

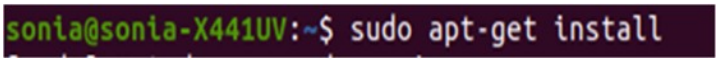

Figure III.1 : Installation du package APT

Figure III.2 : Mise à jour des packages.

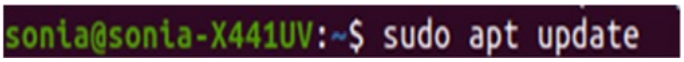

Figure III.3 : Mise à jour système.

III.2 Installation de VMWare Workstation Pro

Pour installer VMware Workstation Pro sous UBUNTU il faut tout d'abord télécharger le fichier Vmware-Workstation-Full-16.2.3-19376536.x86_64.bundle sur le site officiel de VMware, la version la plus récente actuellement est la 16eme édition.
Ensuite il faut exécuter la commande montrée sur la figure III.4.

Figure III.4 : Installation de VMWare Workstation Pro.

Une fois l'installation terminée, VMWare se trouvera dans la liste des activités. Nous pouvons y créer maintenant des machines virtuelles. Il est recommandé de suivre la configuration « typical » afin de ne pas compliquer l'installation.
Pour la réalisation de notre projet nous aurons besoin d'installer deux machines virtuelles :

- Une machine administratrice Ansible : vu que ansible fonctionne sur des systèmes de type Unix nous avons fait le choix de travailler sous Ubuntu.
- Une machine à gérer Node1 : node1 utilise aussi Ubuntu comme système d'exploitation.

Il est important d'attribuer le même nom d'utilisateur à nos machines pour des raisons de connectivité.

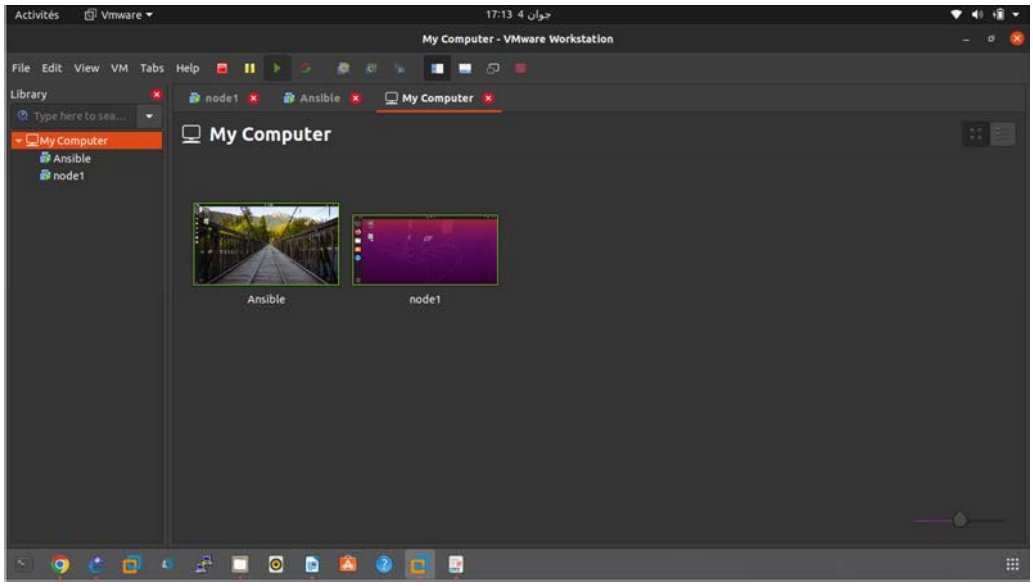

Figure III.5 : Liste des machines virtuelles installé.

III.3 GNS3

III.3.1 Installation de GNS3

Afin de pouvoir installer GNS3 sur notre machine locale nous devons exécuter les commandes des figures III.6 et III.7

- Ajout du référentiel GNS3 PPA après avoir accédé au mode super utilisateur (root) :

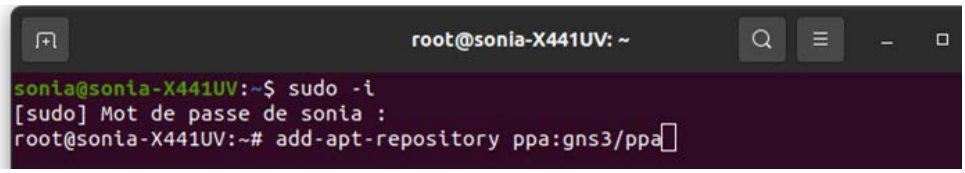

Figure III.6 : Installation du référentiel GNS3.

- Installation de l'interface graphique GNS3 et du Serveur GNS3 toujours en mode root:

Figure III.7 : Installation du GNS3 Gui et Server.

Une fois l'installation terminée, nous lançons GNS3 sur notre machine locale afin de le configurer et d'y ajouter les équipements dont nous aurons besoin pour la réalisation de notre maquette ; des routeurs Cisco ainsi que nos machines virtuelles ansible et node1.

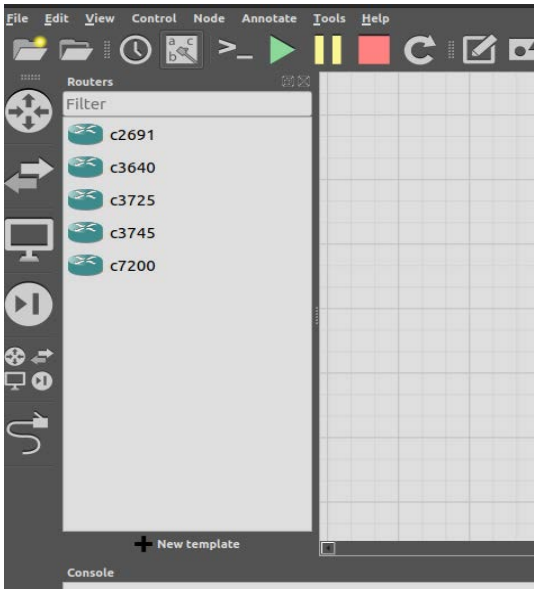

Figure III.8 : Liste des routeurs téléchargés.

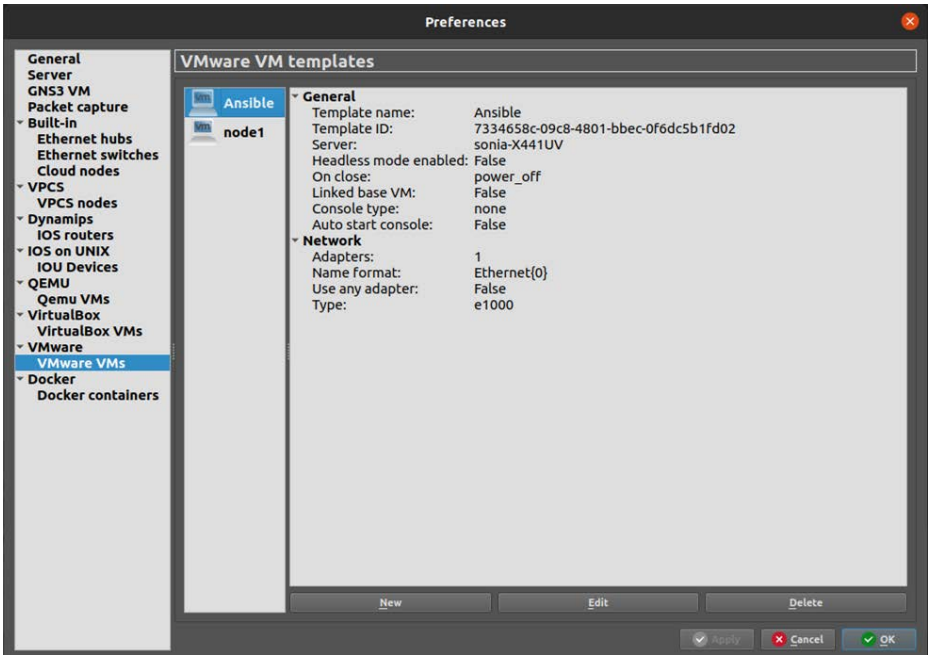

Figure III.9 : Liste des machines virtuelles créés dans GNS3.

III.3.2 Réalisation de l'architecture réseau

Une fois nos équipements prêts, nous allons réaliser cette architecture réseau sur notre LAB, comme l'indique la figure III.10

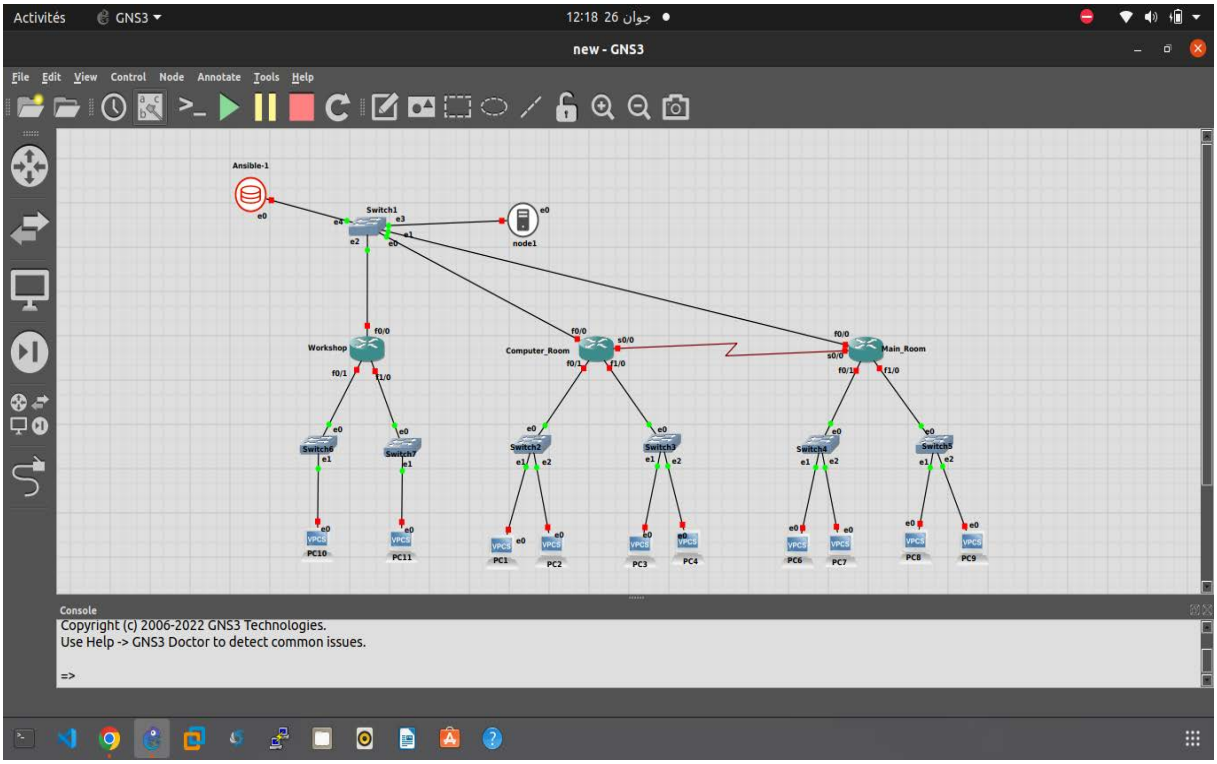

Figure III.10: Topologie du réseau.

III.3.3 Attribution des adresses IP aux équipements

Nom de la machine	Interface	Adresse IP	Masque de sous réseau	Passerelle
Ansible	/	10.10.20.6	255.255.255.0	10.10.20.1
Node1	/	10.10.20.13	255.255.255.0	10.10.20.1
Computer_Room	F0/0	10.10.20.1	255.255.255.0	/
	F0/1	10.10.10.1	255.255.255.0	/
	F1/0	10.10.30.1	255.255.255.0	/
	S0/0	192.168.1.1	255.255.255.252	/
Main_Room	F0/0	10.10.20.33	255.255.255.0	/
	F0/1	172.16.20.1	255.255.255.0	/
	F1/0	172.16.30.1	255.255.255.0	/
	S0/0	192.168.1.2	255.255.255.252	/

Workshop	F0/0	10.10.20.55	255.255.255.0	/
	F0/1	10.10.12.1	255.255.255.224	/
	F1/0	10.10.13.1	255.255.255.224	/
PC1	/	10.10.10.11	255.255.255.0	10.10.10.1
PC2	/	10.10.10.2	255.255.255.0	10.10.10.1
PC3	/	10.10.30.3	255.255.255.0	10.10.30.1
PC4	/	10.10.30.4	255.255.255.0	10.10.30.1
PC6	/	172.16.20.6	255.255.255.0	172.16.20.1
PC7	/	172.16.20.7	255.255.255.0	172.16.20.1
PC8	/	172.16.30.8	255.255.255.0	172.16.30.1
PC9	/	172.16.30.9	255.255.255.0	172.16.30.1
PC10	/	DHCP	DHCP	DHCP
PC11	/	DHCP	DHCP	DHCP

Tableau III.1 : Adressage IP

III.4 Le protocole Secure Shell

III.4.1 SSH sur les machines virtuelles

III.4.1.1 Activation du protocole SSH sur les VMs

Pour activer le protocole SSH sur une machine UBUNTU il suffit de télécharger « Open SSH Server », pour cela, nous avons besoin d'une seule commande «sudo apt install openssh-server» comme illustré sur la figure III.11. Cette commande sera exécutée sur toutes les machines UBUNTU que nous souhaitons connecter via SSH.

```
sonia@sonia-X441UV:~$ sudo apt install openssh-server
Lecture des listes de paquets... Fait
Construction de l'arbre des dépendances
Lecture des informations d'état... Fait
Les paquets supplémentaires suivants seront installés :
  ncurses-term openssh-sftp-server ssh-import-id
Paquets suggérés :
  molly-guard monkeysphere ssh-askpass
Les NOUVEAUX paquets suivants seront installés :
  ncurses-term openssh-server openssh-sftp-server ssh-import-id
0 mis à jour, 4 nouvellement installés, 0 à enlever et 17 non mis à jour.
Il est nécessaire de prendre 688 ko dans les archives.
Après cette opération, 6,010 ko d'espace disque supplémentaires seront utilisés.
Souhaitez-vous continuer ? [O/n] o
Réception de :1 http://dz.archive.ubuntu.com/ubuntu focal/main amd64 ncurses-term all 6.2-0ubuntu2 [249 kB]
Réception de :2 http://dz.archive.ubuntu.com/ubuntu focal-updates/main amd64 openssh-sftp-server amd64 1:8.2p1-4ubuntu0.5 [51.5 kB]
Réception de :3 http://dz.archive.ubuntu.com/ubuntu focal-updates/main amd64 openssh-server amd64 1:8.2p1-4ubuntu0.5 [377 kB]
Réception de :4 http://dz.archive.ubuntu.com/ubuntu focal/main amd64 ssh-import-id all 5.10-0ubuntu1 [10.0 kB]
688 ko réceptionnés en 13s (51.2 ko/s)
Préconfiguration des paquets...
Sélection du paquet ncurses-term précédemment désélectionné.
(Lecture de la base de données... 259565 fichiers et répertoires déjà installés.)
Préparation du dépaquetage de .../ncurses-term_6.2-0ubuntu2_all.deb ...
Dépaquetage de ncurses-term (6.2-0ubuntu2) ...
Sélection du paquet openssh-sftp-server précédemment désélectionné.
Préparation du dépaquetage de .../openssh-sftp-server_1%3a8.2p1-4ubuntu0.5_amd64.deb ...
Dépaquetage de openssh-sftp-server (1:8.2p1-4ubuntu0.5) ...
```

Figure III.11 : Installation de l'open SSH server

III.4.1.2 SSH User Equivalence

L'objectif de cette configuration est de pouvoir utiliser Ansible via SSH et non pas par password. L'inconvénient de la méthode « password » est que le mot de passe peut être plus difficile à retenir ; lorsqu'on sécurise une session ou une machine d'un réseau privé, on doit créer un mot de passe long et contenant des caractères d'usage inhabituel ce qui fait que l'utilisateur aura du mal à écrire à chaque fois ce mot de passe, de plus c'est une méthode moins sécurisée par rapport au SSH User Equivalence.

Le type de clé que nous allons utiliser est la clé « ECDSA » ; c'est le type le plus récent et le plus sécurisé. On peut également utiliser « RSA » ou « DSA ».

III.4.1.3 Configuration du SSH User Equivalence entre la machine Ansible et le nœud géré

Pour cela il faut que les deux machines échangent leurs clés de chiffrement. Cela se fait par la création d'une clé sur la machine Ansible afin de la copier sur le nœud géré, ensuite se connecter à ce nœud avec une seule commande pour enfin pouvoir le gérer à distance.

- **Création de la clé**

La fenêtre suivante illustre la commande à introduire et les réponses générées lors de la création de cette clé.

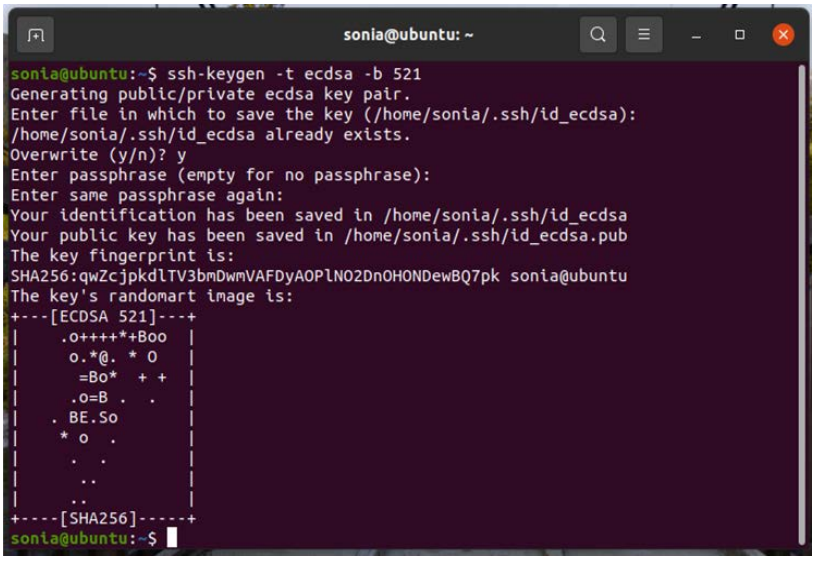

Figure III.12 : Création de la clé SSH

- **Verification de la creation de la clé**

  Afin de vérifier que la clé a été créé avec succès nous exécutons la commande suivante qui va nous lister toutes les clés générées dans cet emplacement.

« id_ecdsa » représente la clé privée

« id_ecdsa.pub » représente la clé publique

Figure III.13 : Liste des clés existantes sur le ficher .SSH

- **Copie de la clé sur le nœud géré**

En premier lieu il est préférable de faire un test de connectivité entre les deux machines en utilisant la commande ping

Figure III.14 : Ping entre la machine ansible et node1

Après cela nous ouvrons la liste des clés de chiffrement sur le nœud géré en exécutant la commande suivante :

Figure III.15 : Commande pour ouvrir le fichier des clés sur node1

Nous remarquons sur la figure suivante qu'il existe déjà 3 clés sur notre fichier

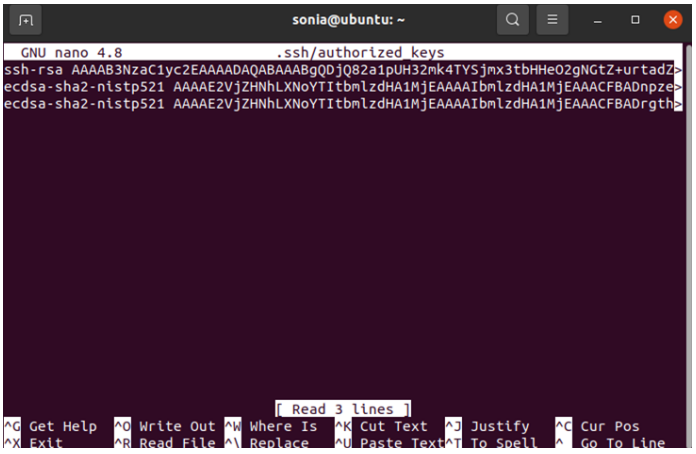

Figure III.16 : Clés existantes sur node1

Copions maintenant la clé avec la commande montrée dans la figure III.17 en introduisant le nom de la machine :

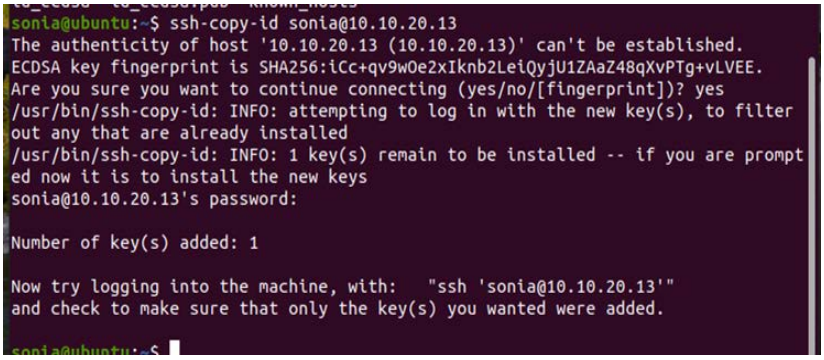

Figure III.17 : Commande pour copier la clé

Une vérification sur le nœud géré permet de montrer une nouvelle clé dans la liste des « authorized_keys »

Figure III.18 : Vérification de l'existence de la clé

❖ Connexion à distance via SSH

Dans notre cas nous avons déjà un « agent SSH » ajouté par défaut par notre système d'exploitation UBUNTU. Il suffit alors d'introduire les commandes comme illustrées sur la figure III-19.

On peut voir que nous avons réussi à accéder à notre machine via le protocole SSH. Nous pouvons aussi nous déconnecter en utilisant la commande « logout » et revenir facilement sur notre machine locale.

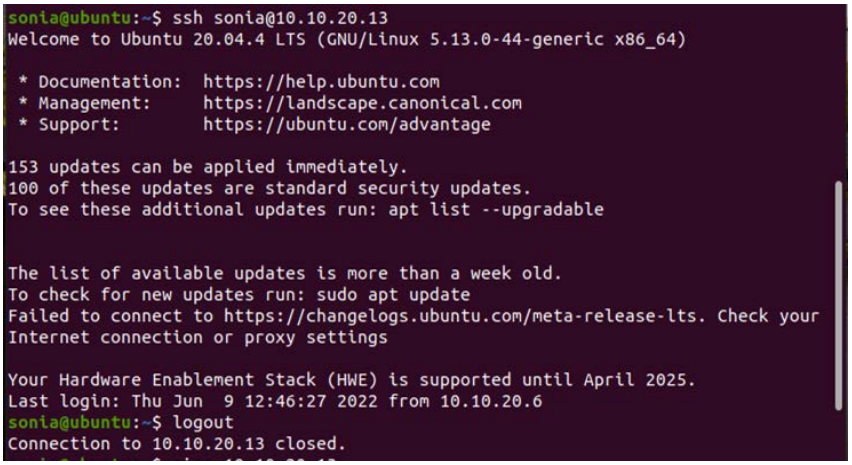

Figure III.19 : Test de connectivité via SSH

Il est à noter que la machine ne demande le mot de passe du compte utilisateur que lors de la première connexion seulement.

III.4.2 SSH sur les Equipements Cisco

III.4.2.1 Activer SSH sur les Routeurs Cisco

Ici nous allons configurer SSH sur nos routeurs, pour cela, nous allons suivre les étapes suivantes :

- Renommer notre équipement
- Créer un compte utilisateur avec le mot de passe
- Définir un nom de domaine
- Générer les clés « RSA » et choisir le nombre de bits a 1024 (longueur de clés
- Définir la version SSH que nous allons utiliser
- Configurer la ligne virtuelle VTY
- Utiliser le compte utilisateur que nous avons créé
- Désactiver le protocole Telnet en activant SSH

Les configurations sont montrées sur la figure III-20 et doivent être exécutées sur tous les routeurs du réseau.

```
Computer_Room#conf t
Enter configuration commands, one per line.  End with CNTL/Z.
Computer_Room(config)#username sonia priv 15 password 1008
Computer_Room(config)#ip domain-name c_room.com
Computer_Room(config)#crypto key generate rsa
The name for the keys will be: Computer_Room.c_room.com
Choose the size of the key modulus in the range of 360 to 2048 for your
  General Purpose Keys. Choosing a key modulus greater than 512 may take
  a few minutes.

How many bits in the modulus [512]: 1024
% Generating 1024 bit RSA keys, keys will be non-exportable...[OK]

Computer_Room(config)#
*Mar  1 02:36:56.855: %SSH-5-ENABLED: SSH 1.99 has been enabled
Computer_Room(config)#ip ssh version 2
Computer_Room(config)#line vty 0 15
Computer_Room(config-line)#login local
Computer_Room(config-line)#transport input ssh
Computer_Room(config-line)#
```

Figure III.20 : Configuration du protocole SSH sur les routeurs.

III.4.2.2 Teste de la configuration SSH avec PuTTY

PuTTY est un émulateur de terminal qui permet de se connecter à une machine distante en utilisant SSH. Nous allons l'utiliser pour tester la configuration du protocole SSH sur nos machines Cisco. Pour cela nous allons accéder à ce logiciel et nous allons entrer les informations de connexion nécessaires puis, nous allons essayer de nous connecter à l'équipement voulu :

Nous avons configuré SSH sur nos deux routeurs : Computer_Room et Main_Room.

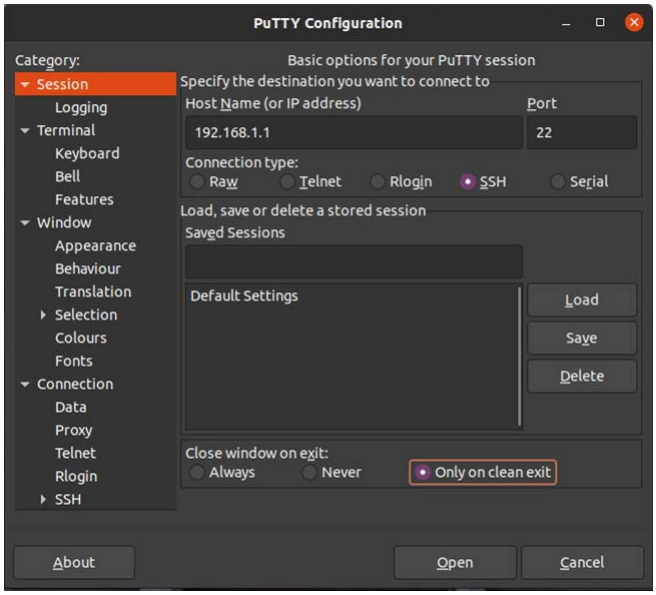

Figure III.21 : Test de connectivité avec Putty

La figure III.22 montre que nous avons réussi à acceder aux routeurs via SSH.

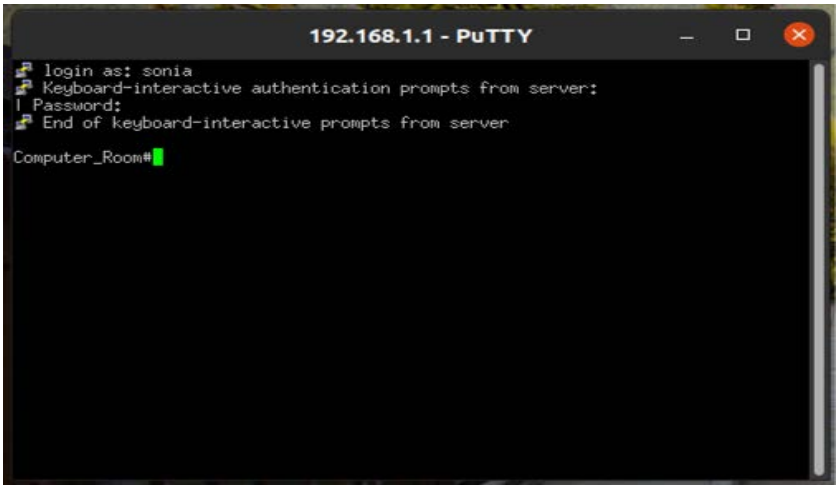

Figure III.22 : Connexion établie sur le routeur.

III.5 Installation d'Ansible

Afin d'effectuer l'installation d'Ansible sur notre VM administratrice nous allons exécuter les commandes représentées dans les figures III.23, III.24 et III.25, III.26.

❖ **Software Properties Common**

C'est un logiciel qui fournit une abstraction des référentiels APT utilisés. Il permet de gérer facilement des sources de logiciels de distribution et de fournisseurs de logiciels indépendants. En pratique, cela signifie qu'il fournit des scripts utiles pour ajouter et supprimer des PPA.

```
sonia@ubuntu:~$ sudo apt install software-properties-common
```

Figure III.23 : Installation de Software properties Common.

❖ **Personal package archive (PPA)**

À l'aide d'une archive de packages personnels (PPA), il est possible de distribuer des logiciels et des mises à jour directement aux utilisateurs d'Ubuntu. De créer notre paquet source, le télécharger et Launchpad créera des fichiers binaires, puis les hébergera dans notre propre référentiel APT.

```
sonia@ubuntu:~$ sudo add-apt-repository --yes --update ppa:ansible/ansible
```

Figure III.24 : Mise à jour du répertoire PPA ansible.

❖ **Installation de l'outil Ansible**

```
sonia@ubuntu:~$ sudo apt install ansible
```

Figure III.25 : Installation complète de ansible.

❖ **Installation de Paramiko**

Paramiko est une interface python (2.7et plus) qui implémente le protocole SSH version2.fournissant à la fois des fonctionnalités client et serveur.

```
sonia@ubuntu:~$ pip3 install paramiko
```

Figure III.26 : Installation de paramiko.

❖ **Vérification de l'installation**

Pour vérifier l'installation d'Ansible nous exécutons la commande illustrée par la figure III.27.

```
sonia@ubuntu:~$ ansible --version
ansible [core 2.12.6]
  config file = /etc/ansible/ansible.cfg
  configured module search path = ['/home/sonia/.ansible/plugins/modules', '/us
r/share/ansible/plugins/modules']
  ansible python module location = /usr/lib/python3/dist-packages/ansible
  ansible collection location = /home/sonia/.ansible/collections:/usr/share/ans
ible/collections
  executable location = /usr/bin/ansible
  python version = 3.8.10 (default, Mar 15 2022, 12:22:08) [GCC 9.4.0]
  jinja version = 2.10.1
  libyaml = True
```

Figure III.27 : Vérification de l'installation d'ansible

III.6 Conclusion

Dans ce chapitre nous avons montré toutes les étapes de la préparation de notre environnement et de la mise en place notre réseau en passant par la virtualisation, l'émulateur GNS3 et le protocole sécuriser SSH afin de pouvoir l'automatiser en y injectons des configurations via l'outil d'automatisation Ansible.

CHAPITRE IV :

SIMULATION

IV Simulation

Introduction

Dans ce chapitre des exemples d'automatisation avec Ansible seront traités. L'objectif est de pouvoir créer un programme et de l'injecter en utilisant des lignes de commandes sur les nœuds de notre réseau.

Premièrement, nous allons créer un fichier qui enregistrera toutes les machines que nous devrons gérer.

Deuxièmement, nous écrirons des jeux d'instructions au format YAML qui contiendrons toutes les informations sur les nœuds à gérer et les taches qui doivent être exécutées sur ces nœuds.

Finalement, nous injecterons ces programmes sur nos machines et testerons le bon fonctionnement de ces scripts.

IV.1 Configuration de base

IV.1.1 Configuration sur le fichier ansible.cfg

- L'installation de l'outil Ansible, fournit un fichier de configuration par défaut, ansible.cfg sur le nœud de contrôle qui se situe dans le répertoire courant. Ce fichier permet de modifier certains paramètres Ansible.

  La configuration de base est très souvent suffisante, mais il se peut que certaines modifications des paramètres soient nécessaires afin d'assurer la connectivité et le bon fonctionnement de l'automatisation.

  La figure IV.1 montre le chemin du fichier et comment y accéder en mode super-utilisateur (sudo ou root).

```
sonia@ubuntu:~$ cd /etc/ansible
sonia@ubuntu:/etc/ansible$ ls
ansible.cfg  ansible.cfg.dpkg-old  hosts  hosts.dpkg-old  inventory  playbook  roles
sonia@ubuntu:/etc/ansible$ sudo nano ansible.cfg
[sudo] password for sonia:
```

Figure IV.1 : Accéder au fichier ansible.cfg

- Le fichier ansible.cfg contient des commentaires d'orientation et des paramètres désactivés comme le montre la figure IV.2.

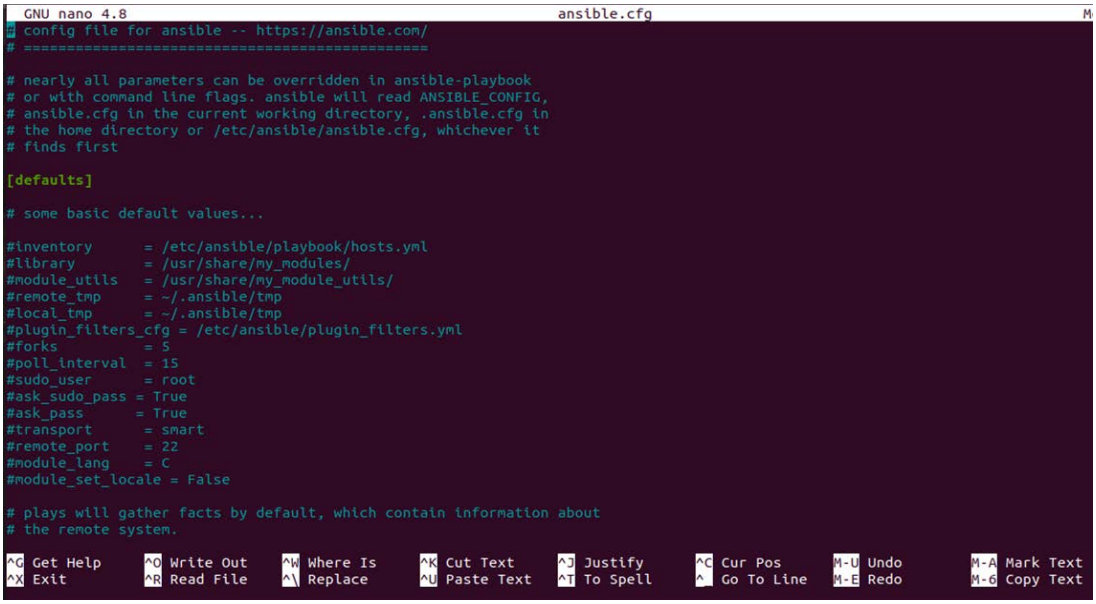

Figure IV.2 : Contenu du fichier ansible.cfg

- Lorsqu'un nouveau nœud est connecté à Ansible, une erreur de vérification des clés SSH apparaît. Afin de régler ce problème il suffit d'activer le paramètre « host_key_checking = false » en retirant le « # » du début de la ligne comme le montre la figure IV.3.

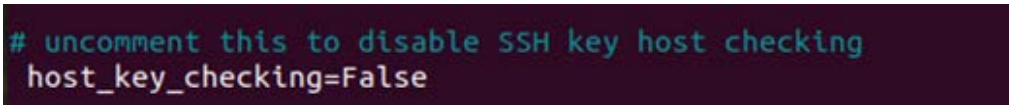

Figure IV.3 : Activation des paramètres.

IV.1.2 Enregistrement des nœuds sur le répertoire de la machine

Pour pouvoir connecter les nœuds gérés à notre machine Ansible, la première étape consiste à enregistrer tous nos nœuds sur la machine de contrôle dans le répertoire « etc » comme le montre la figure IV.4.

Figure IV.4 : Liste des machines dans le répertoire etc.

IV.1.3 Création du fichier d'inventaire Ansible

Dans le répertoire « ansible », il faut créer un sous-répertoire « Playbook », celui-ci contiendra le fichier « hosts.yml » ou seront enregistrés nos nœuds comme le montre la figure IV.5.

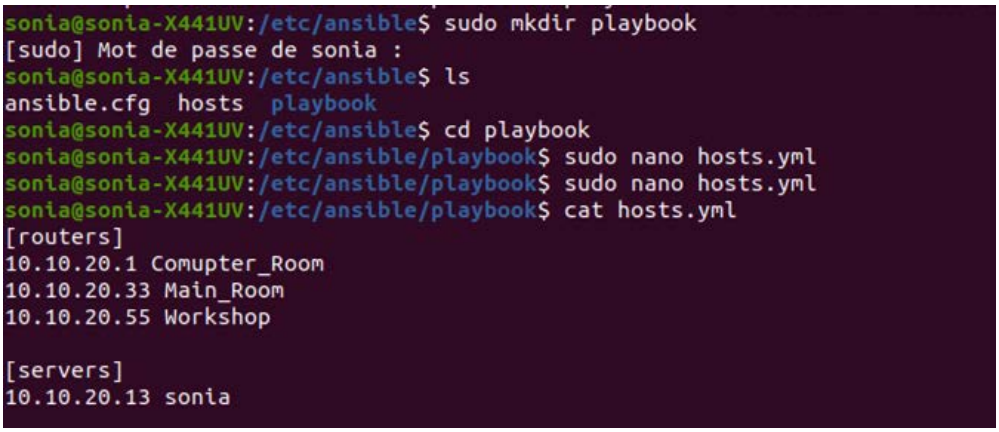

Figure IV.5 : Inventaire.

- La commande « mkdir » est utilisée pour créer un répertoire.
- La commande «ls » est utilisée pour lister le contenu du répertoire.
- La commande « nano » est utilisée pour créer ou modifier le contenu d'un fichier (.txt, .ini, .yml…).
- La commande « cat » est utilisée pour afficher le contenu du fichier dans le terminal.

IV.2 Injection des configurations sur les équipements réseau avec Ansible

IV.2.1 Structure du Playbook Ansible

Un Playbook est structuré comme suit :

- Un fichier YAML commence toujours par trois tirets.
- La première partie du Playbook contient :
  - Name : Le nom du Playbook.
  - Hosts : Les machines cibles.
  - Gather_facts : Le module ‘’des faits’’ est appelé par le Playbook afin de rassembler les variables utiles sur les nœuds distants. La valeur booléenne « true » lui est attribué lorsque nous souhaitons qu'Ansible exécute les taches automatiquement. Tant dis que la valeur booléenne « false » est utilisée lorsque nous voulons ajouter les modules manuellement.

- Connection : Ce plugin est utilisé pour exécuter un Playbook dans un programme d'installation de système d'exploitation, nous mettons la valeur « locale » lorsque notre machine Ansible est connectée dans un réseau local a la machine cible.

- La deuxième partie concerne les variables qu'utilise notre Playbook pour accéder aux machines cibles via SSH. Elle contient à son tour :
  - Username : le nom d'utilisateur
  - Password : le mot de passe lié au nom d'utilisateur
  - Timeout : il est facultatif. Il indique le temps que prend la machine Ansible avant de se déconnecter.

- La troisième et dernière partie concerne les « tasks ». Elle indique les différentes tâches que nous exécutons sur les nœuds. Chaque tache comprend :
  - Name : c'est le nom de la tache
  - Le module concerné comme par exemple « ios_config »
  - Provider : c'est l'argument qui suit chaque appel de module. Il indique que l'algorithme exige l'utilisation des paramètres « vars » dans chacune des taches. Les modules utilisés ne prennent en charge que la valeur « provider :''{{ cli }}'' ».

IV.2.2 Création du playbook

IV.2.2.1 Playbook a plusieurs taches

❖ **Creation du playbook**

Comme le montre la figure IV.6 nous avons exécuter trois taches dans un seul Playbook. Voici la signification de chaque tache :

- Tache 1 : Teste de la joignabilité d'un hôte en utilisant le module Ansible « ios_ping ».
- Tache 2 : Configuration du nom des machines sur Sonatrach_Bejaia en utilisant le module Ansible « ios_config ».
- Tache 3 : Exécution d'une commande d'affichage des interfaces et leurs évaluations en utilisant un Algorithme. Le module Ansible utilisé est « ios_command ».

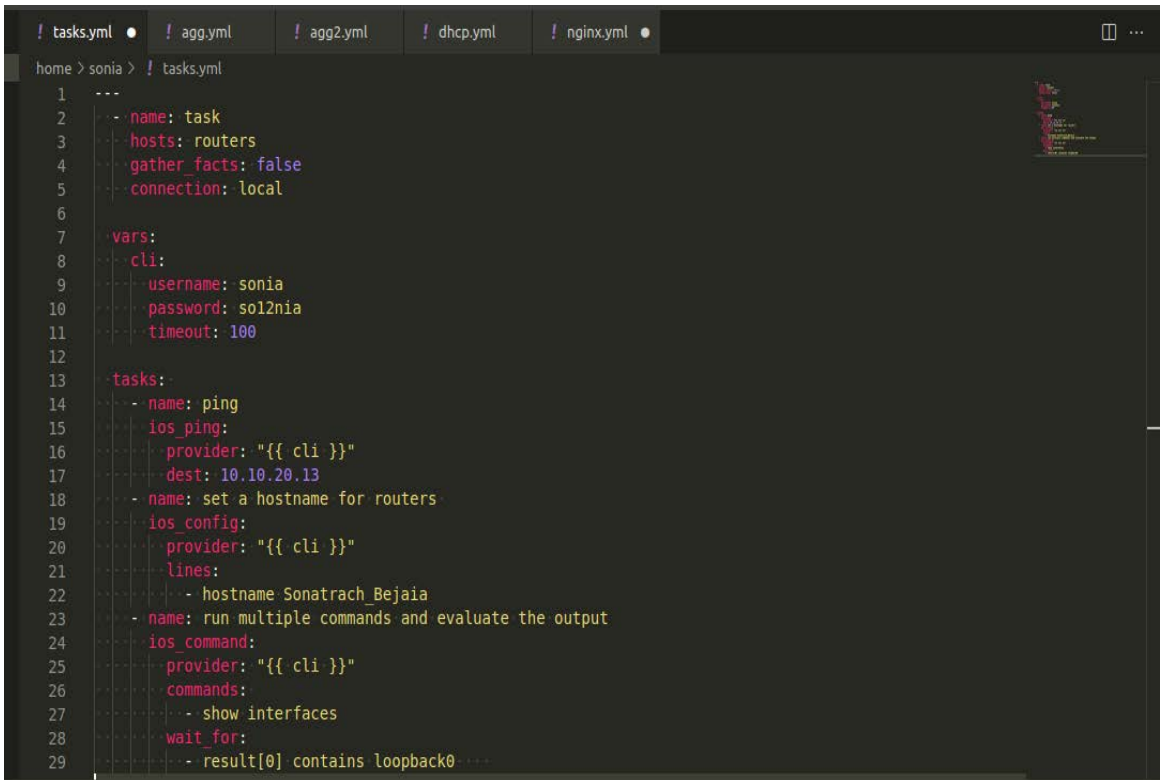

Figure IV.6 : Playbook avec plusieurs tâches

❖ Exécution et résultat du Playbook

Nous injectons le playbook «task.yml» sur un groupe de nœuds situés dans le fichier d'inventaire «hosts.yml» en utilisant la commande « ansible-playbook » comme Illustré dans la figure IV.7.

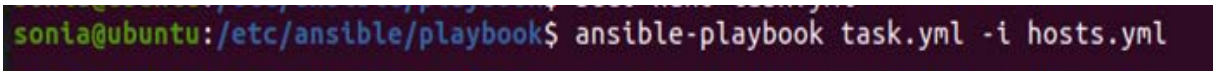

Figure IV.7 : Commande d'exécution du playbook

Les résultats de cette automatisation apparaissent quelques secondes après l'exécution de la commande dans le terminal de la machine Ansible comme l'indique la figure IV.8.

Les deux premières taches [ping] et [set a hostname for routers] ont été exécutées sur le groupe d'hôtes avec succès. Les trois routeurs arrivent à joindre la machine « 10.10.20.13 » et Ansible a pu modifier les noms des machines. Chaque machine a retourné la valeur « OK » pour la tache [ping] et la valeur « changed » pour la tache [set a hostname for routers], puisque nous avons apporté un changement aux machines. Nous pouvons vérifier cela directement sur nos routeurs.

La troisième tache [run multiple commands and evaluate the output] représente un petit Algorithme qui affiche les interfaces à Ansible et qui retourne le résultat [0] lorsqu'il trouve

une interface loopback0. Comme nous n'avons pas configuré d'interface loopback0 sur les trois routeurs, l'Algorithme nous répond par : « la condition n'est pas satisfaite ».

```
sonia@ubuntu:/etc/ansible/playbook$ ansible-playbook task.yml -i hosts.yml

PLAY [tasks] ***********************************************************************************************************************

TASK [ping] ************************************************************************************************************************
[WARNING]: ansible-pylibssh not installed, falling back to paramiko
[WARNING]: ansible-pylibssh not installed, falling back to paramiko
[WARNING]: ansible-pylibssh not installed, falling back to paramiko
[WARNING]: ['connection local support for this module is deprecated and will be removed in version 2.14, use connection
ansible.netcommon.network_cli']
ok: [10.10.20.33]
ok: [10.10.20.1]
ok: [10.10.20.55]

TASK [set a hostname for routers] **************************************************************************************************
[WARNING]: To ensure idempotency and correct diff the input configuration lines should be similar to how they appear if present in the
running configuration on device
changed: [10.10.20.55]
changed: [10.10.20.1]
changed: [10.10.20.33]

TASK [run multiple commands and evaluate the output] *******************************************************************************
fatal: [10.10.20.1]: FAILED! => {"changed": false, "failed_conditions": ["result[0] contains loopback0"], "msg": "One or more condition
al statements have not been satisfied"}
fatal: [10.10.20.33]: FAILED! => {"changed": false, "failed_conditions": ["result[0] contains loopback0"], "msg": "One or more conditio
nal statements have not been satisfied"}
fatal: [10.10.20.55]: FAILED! => {"changed": false, "failed_conditions": ["result[0] contains loopback0"], "msg": "One or more conditio
nal statements have not been satisfied"}

PLAY RECAP *************************************************************************************************************************
10.10.20.1                 : ok=2    changed=1    unreachable=0    failed=1    skipped=0    rescued=0    ignored=0
10.10.20.33                : ok=2    changed=1    unreachable=0    failed=1    skipped=0    rescued=0    ignored=0
10.10.20.55                : ok=2    changed=1    unreachable=0    failed=1    skipped=0    rescued=0    ignored=0
```

Figure IV.8 : Résultat du playbook.

A la fin du Playbook Ansible affiche un « Play récapitulatif » qui affiche le nombre de taches qui ont été exécutée, le nombre de taches qui ont apporté un changement ainsi que le nombre de taches qui retourne une ou plusieurs erreurs.

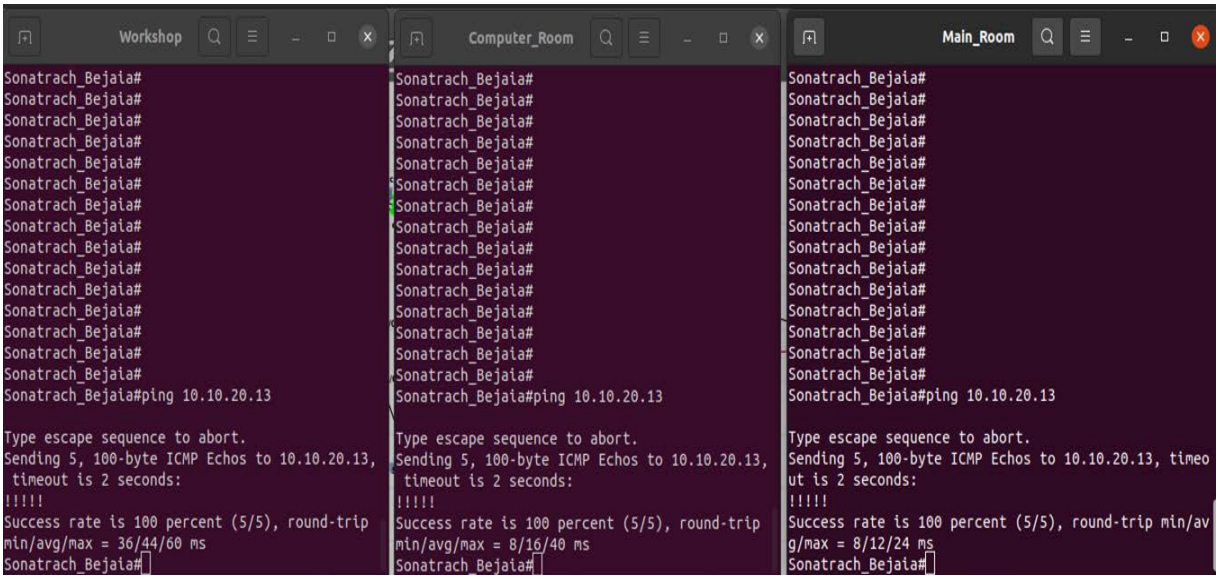

Figure IV.9 : Résultat des configurations sur les routeurs.

- Remplaçons maintenant la dernière tâche du Playbook par des interfaces fast Ethernet qui existent. Comme illustrée sur la figure IV.10

```
23      - name: run multiple commands and evaluate the output
24        ios_command:
25          provider: "{{ cli }}"
26          commands:
27            - show interfaces
28          wait_for:
29            - result[0] contains FastEthernet0/0
```

Figure IV.10 : Modification de la dernière tâche

Ansible nous retourne alors un résultat différent du précédent.

Les noms des machines n'ont pas été changés donc la valeur de « changed » devient « 0 » et l'interface « loopback0 » est remplacée par l'interface « FastEthernet0/0 », qui existe dans tous nos routeurs utilisés. L'algorithme retourne alors la valeur vraie [0]. La valeur « OK » dans le récapitulatif devient « 3 » car toutes les actions ne contiennent aucune erreur La figure IV.11 résume cela.

```
sonia@ubuntu:/etc/ansible/playbook$ ansible-playbook task.yml -i hosts.yml

PLAY [tasks] ***********************************************************************************************************************************

TASK [ping] ************************************************************************************************************************************
[WARNING]: ansible-pylibssh not installed, falling back to paramiko
[WARNING]: ansible-pylibssh not installed, falling back to paramiko
[WARNING]: ansible-pylibssh not installed, falling back to paramiko
[WARNING]: ['connection local support for this module is deprecated and will be removed in version 2.14, use connection
ansible.netcommon.network_cli']
ok: [10.10.20.55]
ok: [10.10.20.33]
ok: [10.10.20.1]

TASK [set a hostname for routers] **************************************************************************************************************
ok: [10.10.20.55]
ok: [10.10.20.1]
ok: [10.10.20.33]

TASK [run multiple commands and evaluate the output] *******************************************************************************************
ok: [10.10.20.55]
ok: [10.10.20.33]
ok: [10.10.20.1]

PLAY RECAP *************************************************************************************************************************************
10.10.20.1                 : ok=3    changed=0    unreachable=0    failed=0    skipped=0    rescued=0    ignored=0
10.10.20.33                : ok=3    changed=0    unreachable=0    failed=0    skipped=0    rescued=0    ignored=0
10.10.20.55                : ok=3    changed=0    unreachable=0    failed=0    skipped=0    rescued=0    ignored=0
```

Figure IV.11 : Résultat du playbook.

IV.2.2.2 Playbook pour la création d'une bannière login

❖ Creation du playbook

Le playbook montré dans la figure IV.12 crée une bannière login, en utilisant le module « ios_banner ». Ce playbook nous permettra d'afficher un message lorsque nous essayerons de nous connecter en mode SSH a notre équipement.

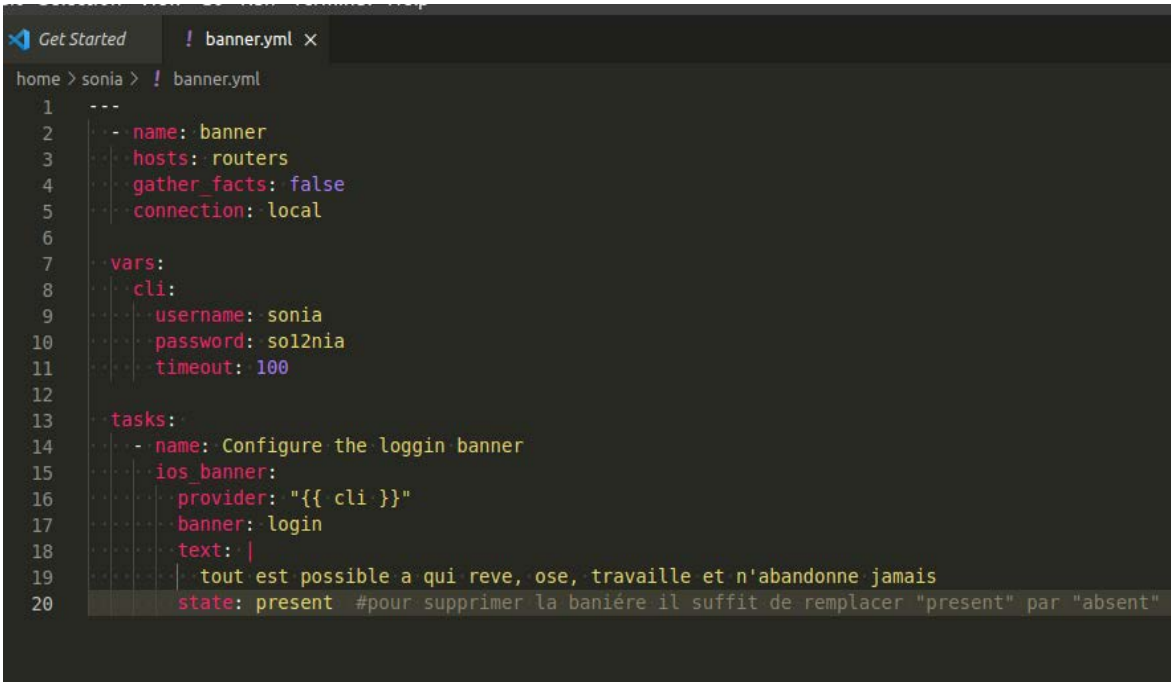

Figure IV.12 : Playbook pour la création d'une bannière

❖ Exécution et résultat du playbook

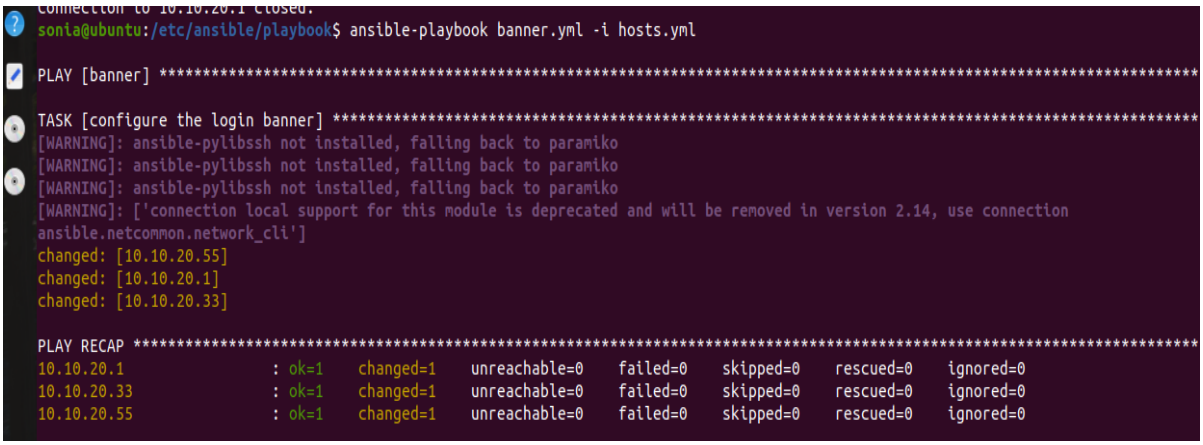

Figure IV.13 : Exécution du playbook bannière.

```
sonia@ubuntu:/etc/ansible/playbook$ ssh -oKexAlgorithms=+diffie-hellman-group1-sha1 -c aes128-cbc sonia@10.10.20.1

tout est possible a qui reve, ose, travaille et n'abandonne jamais
Password:

Computer_Room#logout
Connection to 10.10.20.1 closed.
sonia@ubuntu:/etc/ansible/playbook$ ssh -oKexAlgorithms=+diffie-hellman-group1-sha1 -c aes128-cbc sonia@10.10.20.55

tout est possible a qui reve, ose, travaille et n'abandonne jamais
Password:

Workshop#logout
Connection to 10.10.20.55 closed.
sonia@ubuntu:/etc/ansible/playbook$ ssh -oKexAlgorithms=+diffie-hellman-group1-sha1 -c aes128-cbc sonia@10.10.20.33

tout est possible a qui reve, ose, travaille et n'abandonne jamais
Password:

Main_Room#logout
Connection to 10.10.20.33 closed.
sonia@ubuntu:/etc/ansible/playbook$
```

Figure IV.14 : Vérification du bon fonctionnement sur les équipements.

La figure IV.14 montre que lorsque nous lançons la commande pour nous connecter en SSH à nos équipements depuis la machine administratrice, le message de la bannière créé sur notre Playbook s'affiche.

IV.2.2.3 Playbook pour le routage statique

❖ **Création du Playbook**

```
home > sonia > ! agg.yml
1  ---
2  - name: static routes
3    hosts: 10.10.20.1
4    gather_facts: false
5    connection: local
6
7  vars:
8    cli:
9      username: sonia
10     password: so12nia
11     timeout: 100
12
13 tasks:
14   - name: Configure static routes on Computer_Room
15     ios_static_route:
16       provider: "{{ cli }}"
17       aggregate:
18         - { prefix: 172.16.20.0, mask: 255.255.255.0, next_hop: 192.168.1.2 }
19         - { prefix: 172.16.30.0, mask: 255.255.255.0, next_hop: 192.168.1.2 }
20         - { prefix: 10.10.20.0, mask: 255.255.255.0, next_hop: 192.168.1.2 }
```

Figure IV.15 : Playbook du routage statique pour Computer_Room.

- Pour configurer le Routage Statique sur les Routeurs en utilisant Ansible, on fait appel au module « ios_static_route ». On crée ainsi un Playbook pour chaque routeur.
- La figure IV.15 montre le Playbook «agg.yml» qui a été injecté sur le routeur Computer_Room afin de lui attribuer un routage statique.
- La figure IV.16 montre le playbook « agg2.ml » qui a été injecter sur le routeur Main_Room :

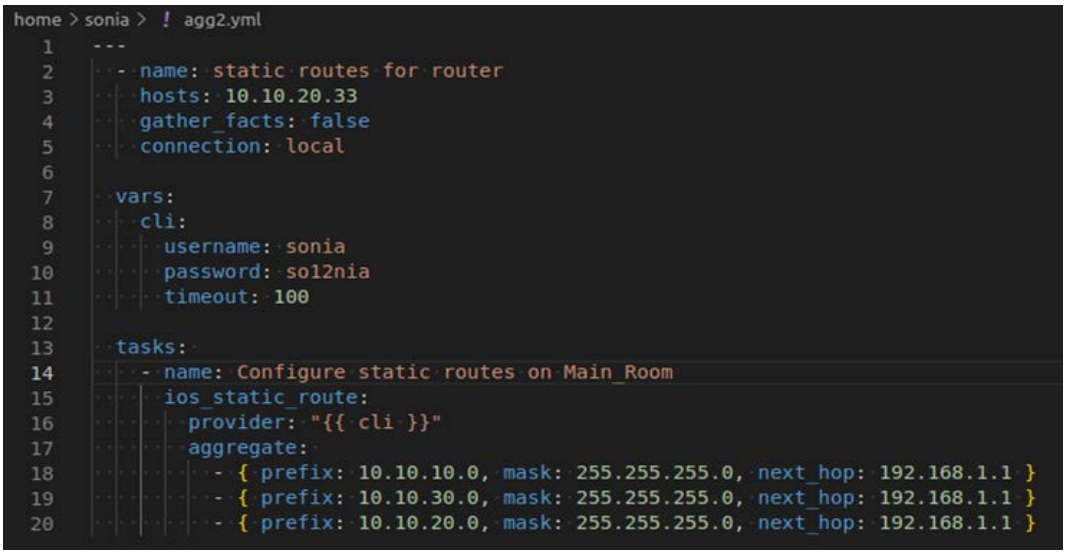

Figure IV.16 : Playbook du routage statique pour Main_Room.

- L'instruction « aggregate » dans nos deux précédents playbook signifie la réunion d'un ensemble de lignes, afin d'eviter la répétition du module « ios_static_route ».
- La ligne 20 représente une route configurée en redondance.

❖ Exécution et résultat du playbook

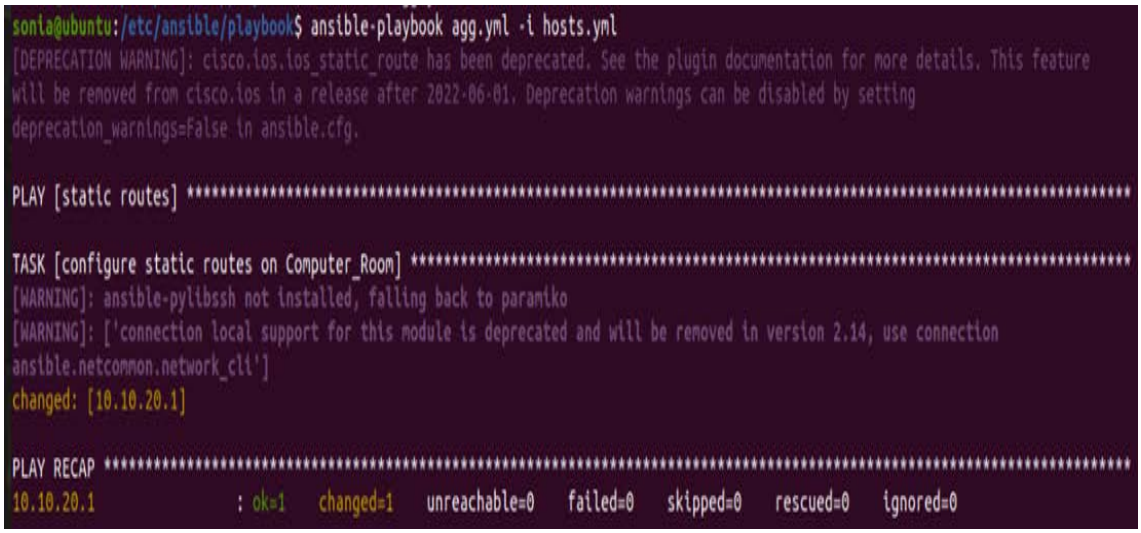

Figure IV.17 : Résultat du Playbook sur Computer_Room.

- La figure IV.17 nous montre l'exécution du playbook « agg.yml » sur notre routeur Computer_Room ainsi que le résultat qui indique qu'il a été exécuter avec succès et qu'Ansible a apporté un changement de configuration pour notre routeur (changed =1)
- La figure IV.18 montre le résultat d'exécution du playbook sur Main_Room. Les routes statiques ont été configurés avec succès :

Figure IV.18 : Résultat du playbook sur Main_Room.

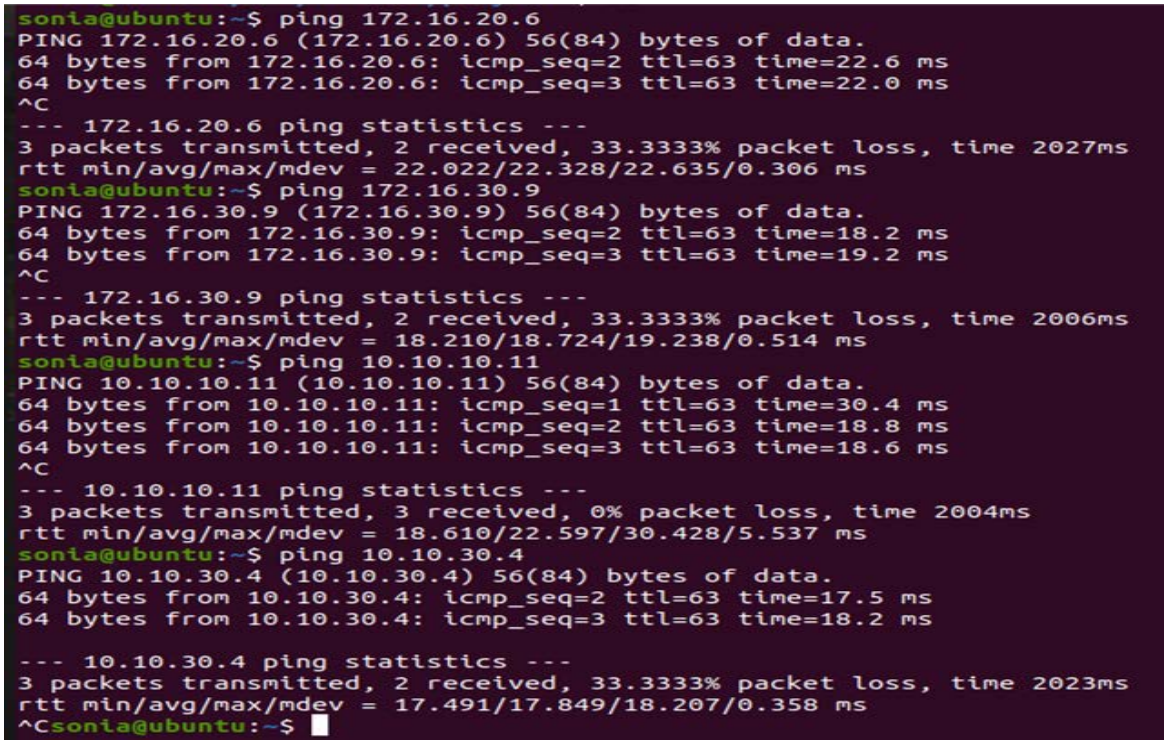

Figure IV.19 : Test du ping sur les hots VPCS.

- Pour tester et vérifier le bon fonctionnement du playbook, nous utilisons le test de connectivité « ping » à partir de notre machine Ansible vers toutes les autres machines du réseau. La figure IV.19 montre que les hôtes PC6, PC9, PC1 et PC4 sont joignables par la machine Ansible.

IV.2.2.4 Playbook DHCP

❖ **Creation du playbook**

Nous avons configuré le protocole DHCP sur le routeur « Workshop ». Le jeu d'instructions écrit est présenté dans la figure IV.20.

```
home > sonia > ! dhcp.yml
1   ---
2   - name: DHCP
3     hosts: 10.10.20.55
4     gather_facts: false
5     connection: local
6
7   vars:
8     cli:
9       username: sonia
10      password: so12nia
11      timeout: 100
12
13  tasks:
14    - name: DHCP config
15      ios_config:
16        provider: "{{ cli }}"
17        lines:
18          - interface f0/1
19          - ip add 10.10.12.1 255.255.255.224
20          - no sh
21          - exit
22          - ip dhcp pool f0/1
23          - network 10.10.12.0 255.255.255.224
24          - default-router 10.10.12.1
25          - exit
26          - interface f1/0
27          - ip add 10.10.13.1 255.255.255.224
28          - no sh
29          - exit
30          - ip dhcp pool f1/0
31          - network 10.10.13.0 255.255.255.224
32          - default-router 10.10.13.1
33          - exit
34          - do wr
```

Figure IV.20 : PPlaybook DHCP.

Il n'existe pas de module spécifique pour configurer le protocole DHCP sur les équipements ios avec Ansible. Nous avons donc utilisé le module « ios_config ». Ce module permet d'exécuter toutes les commandes Cisco sur l'équipement souhaité pour le configurer avec Ansible. Nous avons commencé par configurer les interfaces, ensuite le protocole DHCP, à la fin nous avons tout enregistré sur la machine.

❖ **Execution et résultat du Playbook**

La figure IV.21 montre la commande Ansible, ainsi que le résultat obtenu.

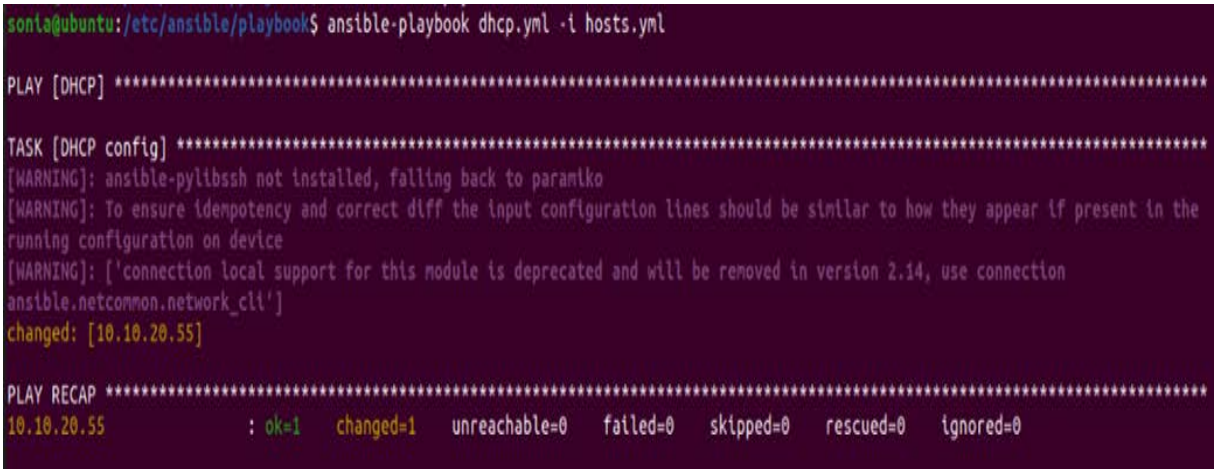

Figure IV.21 : Résultat du playbook DHCP

❖ **Vérification des configurations**

Pour s'assurer que le protocole DHCP a été configuré sur le Routeur Workshop, nous allons activer DHCP sur les Ordinateurs en utilisant la commande « ip dhcp ». Le serveur DHCP (Routeur Workshop) leurs attributs alors automatiquement des adresses IP, comme l'illustre la figure IV.22.

Figure IV.22 : Vérification des configurations.

IV.3 Injection des configurations sur des machines avec Ansible

IV.3.1 Installation d'un serveur web à l'aide d'un Playbook

❖ **Définition du serveur web NGINX**

Nginx est un serveur web open-source qui est utilisé comme cache http. Il offre une faible utilisation de la mémoire et une grande simultanéité. Plutôt que de créer de nouveaux processus pour chaque requête Web, Nginx utilise une approche asynchrone et événementielle où les requêtes sont traitées dans une seule tache.

Figure IV.23: Serveur Web Nginx

❖ Création du Playbook

Nous avons créé un Playbook pour installer le serveur web Nginx sur la machine cible UBUNTU. Le Playbook est illustré sur la figure IV.24.

```
1  ---
2  - name: nginx
3    hosts: servers
4    become: yes
5    gather_facts: false
6
7    tasks:
8      - name: install nginx
9        apt:
10         name: nginx
11         state: latest
```

Figure IV.24 : Playbook pour installer Nginx

Le Playbook injecté sur la Machine Linux ne contient pas de partie variable car il n'utilise pas de compte d'authentification SSH, mais une clé de chiffrement déjà échangé. Explication des instructions qui diffèrent du playbook injecté sur les IOS :

- Become : Ce plugin indique que les taches utilisées ont accès au mode « root » lorsqu'on lui attribue la valeur booléenne « yes ».
- Apt : Ce module est utilisé dans la partie « tasks ». Comme nous l'avons vu dans les chapitres précédents, il permet l'installation des différents packages.

- State : Peut avoir plusieurs valeurs selon le cas traité, dans le nôtre l'instruction indique a Ansible qu'il doit installer la dernière version de « Nginx ».

❖ Exécution et résultat du Playbook

Avant de lancer l'installation avec Ansible nous exécutons la commande « nginx -V » pour s'assurer qu'il n'est pas installé sur le nœud cible. La figure IV.25 indique le résultat :

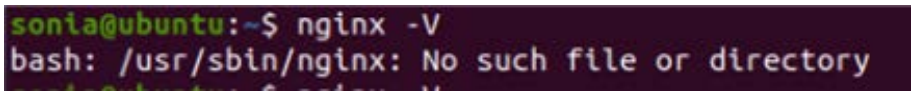

Figure IV.25 : Vérification de l'installation de Nginx

Les options de base --become et -K ont été ajouté lors de l'exécution de la commande afin d'accéder au mode privilège. La figure IV.26 illustre ce résultat.

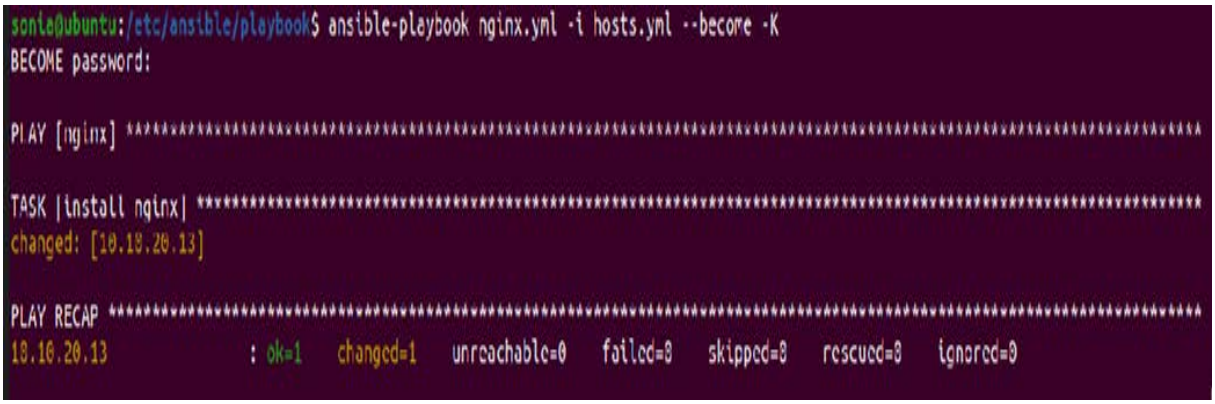

Figure IV.27 : éxecution du playbook Nginx

Pour vérifier que « Nginx » a correctement été installé sur le nœud cible, nous réexécutons la commande précédente. La figure IV.27 montre que la version 1.18.0 de Nginx a été correctement installé.

Figure IV.28 : Vérification de l'installation de Nginx

IV.3.2 Injection des configurations avec des commandes Ad-hoc

❖ **Définition des Commandes Ad-hoc**

La commande Ansible offre la possibilité d'exécuter des modules Ansible de manière "ad-hoc", c'est à dire tâche par tâche sur un groupe d'hôtes. L'intérêt est de pouvoir exécuter la même tâche en parallèle sur un inventaire ou un groupe d'hôtes. Nous n'avons pas besoin d'enregistrer ces commandes pour une utilisation ultérieure. Il existe plusieurs tâches dans Ansible pour lesquelles nous n'avons pas besoin d'écrire un Playbook. A la place, nous pouvons simplement exécuter une commande ad-hoc Ansible. Il s'agit d'une commande à une seule ligne pour effectuer une seule tâche sur l'hôte cible.

❖ **Utilisation des commandes Ad-Hoc**

- **Commande 1**

La commande Ad-Hoc suivante permet l'exécution du module ping sur les hôtes ciblées. La figure IV.28 illustre la syntaxe de la commande avec comme résultat « succes », ce qui montre que la machine cible est joignable à travers ansible.

```
ines@ubuntu:/etc/ansible/playbook$ ansible servers -i hosts.yml -m ping
10.10.20.13 | SUCCESS => {
    "ansible_facts": {
        "discovered_interpreter_python": "/usr/bin/python3"
    },
    "changed": false,
    "ping": "pong"
}
ines@ubuntu:/etc/ansible/playbook$
```

Figure IV.29 : Commande ad-hoc pour effectuer le ping

- **Commande 2**

La commande suivante nous permet l'exécution des commandes Ad-Hoc en tant qu'utilisateur non root (non super utilisateur), avec des privilèges root (super utilisateur), A condition de lui fournir d'abord le mot de passe. Nous Montrons une partie du résultat sur la figure IV.29.

```
ines@ubuntu:/etc/ansible/playbook$ ansible servers -i hosts.yml -m shell -a 'fdisk -l' -u ines --become -K
BECOME password:
10.10.20.13 | CHANGED | rc=0 >>
Disk /dev/fd0: 1.42 MiB, 1474560 bytes, 2880 sectors
Units: sectors of 1 * 512 = 512 bytes
Sector size (logical/physical): 512 bytes / 512 bytes
I/O size (minimum/optimal): 512 bytes / 512 bytes
Disklabel type: dos
Disk identifier: 0x90909090

Device     Boot      Start        End    Sectors  Size Id Type
/dev/fd0p1      2425393296 4850786591 2425393296  1.1T 90 unknown
/dev/fd0p2      2425393296 4850786591 2425393296  1.1T 90 unknown
/dev/fd0p3      2425393296 4850786591 2425393296  1.1T 90 unknown
/dev/fd0p4      2425393296 4850786591 2425393296  1.1T 90 unknown


Disk /dev/sda: 10 GiB, 10737418240 bytes, 20971520 sectors
Disk model: VMware Virtual S
Units: sectors of 1 * 512 = 512 bytes
Sector size (logical/physical): 512 bytes / 512 bytes
I/O size (minimum/optimal): 512 bytes / 512 bytes
Disklabel type: dos
Disk identifier: 0x6a0bac08

Device     Boot   Start      End  Sectors  Size Id Type
/dev/sda1  *       2048  1050623  1048576  512M  b W95 FAT32
/dev/sda2       1052670 20969471 19916802  9.5G  5 Extended
/dev/sda5       1052672 20969471 19916800  9.5G 83 Linux
```

Figure IV.30 : Commander ad-hoc pour le privilège du compte root

- **Commande 3**

Cette Commande est utilisée pour demander l'authentification par mot de passe SSH. Lorsque nous exécutons la commande, le mot de passe SSH que nous avons créé lors de l'échange des clés est demandé. La figure IV.30 montre le résultat.

```
}
ines@ubuntu:/etc/ansible/playbook$ ansible servers -i hosts.yml -m ping --ask-pass
SSH password:
10.10.20.13 | SUCCESS => {
    "ansible_facts": {
        "discovered_interpreter_python": "/usr/bin/python3"
    },
    "changed": false,
    "ping": "pong"
}
ines@ubuntu:/etc/ansible/playbook$
```

Figure IV.31 : Commande ad-hoc pour l'authentification SSH

- **Commande 4**

La commande illustrée dans la figure IV.31 permet d'afficher toutes les informations relatives aux logiciels et application présents dans l'hôte cible.

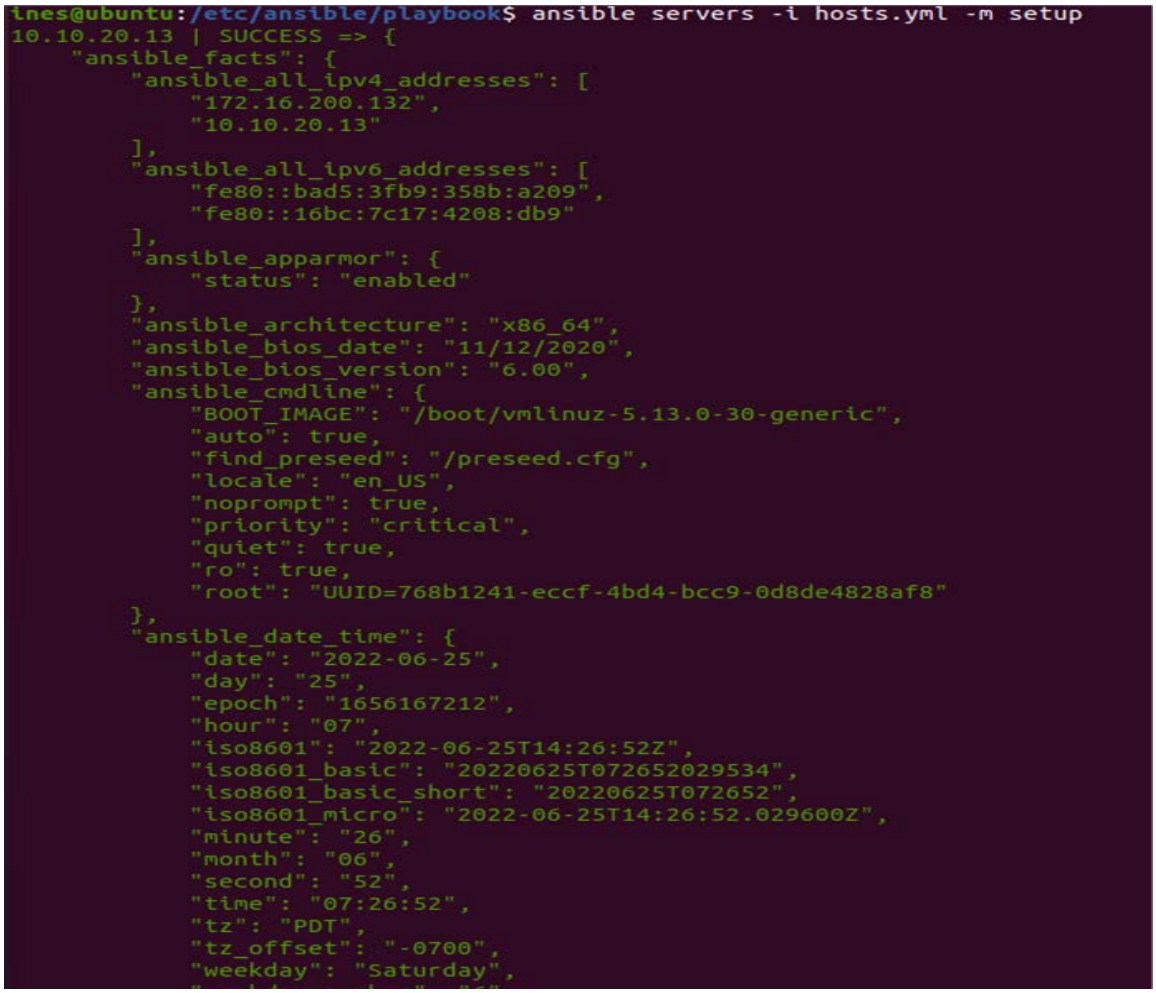

Figure IV.32 : Commande ad-hoc setup.

IV.4 Conclusion

Pour conclure, il est possible de dire que l'objectif fixé au début de ce chapitre a été atteint, nous avons montrée qu'il était possible d'injecter des configurations, d'installer des applications et de géré le réseau à travers l'outil d'automatisation ansible.

Nous avons détaillé et expliquer les différents jeux d'instructions créer ainsi que les deux méthodes qu'utilise Ansible pour injecter des modules. A savoir le Playbook et les commandes ad-hoc.

On peut donc dire que Ansible représente une excellente solution pour l'automatisation des infrastructures.

CONCLUSION GENERALE :

Conclusion générale

L’Objectif de notre projet était la réalisation d’une architecture réseau en s’inspirant du réseau de l’entreprise Sonatrach RTC de Bejaia et d’intégrer un serveur qui contrôle les différents nœuds du réseau.

Nous avons démontré l'importance de l'automatisation dans un réseau. Nous avons cité les aspects majeurs d'Ansible, rendant ainsi son utilisation très simple, intéressante et puissante. C'est un outil qui se veut simple à l'utilisation mais suffisamment puissant pour automatiser des environnements d'applications complexes à plusieurs niveaux.

Nous sommes partis sur la base d'un projet de fin d’études, et nous avons fini par augmenter la performance du réseau de l’entreprise, en étudiant l’automatisation de plus prés.

Grace a cet outil, les ingénieurs réseau n'ont plus besoin de configurer par eux-mêmes chaque appareil individuellement, ils ont juste besoin de créer l'infrastructure appropriée et exécuter l’automatisation avec la scénarisation. La contrôlabilité du réseau devient plus facile et les changements peuvent être déployés plus rapidement, certains automatiquement.

Le monde de l’automatisation est impatient de ce que l’avenir réserve. Investir du temps dans l’apprentissage des outils d’infrastructure et de code nous sera utile pour les années à venir. Des projets Ansible sont sûrement à venir dans les prochains mémoires.

Par manque de temps et de moyens matériels, nous n’avons pas eu l’occasion d’élargir notre maquette et d’automatiser encore plus de taches. En revanche nous espérons que les générations à venir se pencheront dessus.

WEBOGRAPHIE

[1] https://www.cisco.com/c/fr_ca/solutions/automation/network-automation.html.

[F2] https://linux.goffinet.org/ansible/presentation-produit-ansible/.

[3] https://docs.ansible.com/ansible/latest/index.html.

[4] https://www.ansible.com/blog/2013/12/08/the-origins-of-ansible

[5] https://medium.com/@snsavithrik1/an-introduction-to-ansible-aa62559d97de.

[6] https://www.ionos.fr/digitalguide/serveur/outils/protocole-ssh/

[7] https://www.redhat.com/fr/topics/virtualization.

[F8] http://neurones-oxygen.com/virtualisation#sthash.Mftc4fuG.dpbs.

[9] https://www.mbi85.fr/serveur-informatique-talmont-saint-hilaire.html.

[10] https://www.vmware.com/support/pubs/.

Résumé

Ce mémoire a été rédigé tout au long du semestre en vue de l’obtention du diplôme de Master, spécialité réseau et télécommunications au niveau de l’université de Bejaia. Nous avons établi un stage dans le cadre d'une acquisition de connaissances, encadré par le département des Systèmes Informatiques de Sonatrach pour but d’étudier l’architecture du réseau informatique de l’entreprise et de proposé une amélioration qui traite l’automatisation avec l’outil Open Source Ansible, dans un environnement Linux. L’automatisation est un sujet d'actualité pour les acteurs de l'industrie des systèmes et des réseaux informatiques. Ansible offre une meilleure utilisation des ressources en l’attribuant à l’innovation et fait face à la concurrence avec un déploiement des logiciels rapide et sécurisé.

Abstract

This thesis was redacted throughout the semester in order to obtain the Master's degree, specialty network and telecommunications at the University of Bejaia. We have established an internship as part of acquiring knowledge; supervised by the Computer Systems department of Sonatrach for the purpose of studying the company’s computer network architecture and also suggesting an improvement that treats or deals with automation using the Open Source Ansible tool, on a Linux environment. Automation is a topic for industry performers in the computer systems and networks. Ansible offers the best resources utilization, assigning it to innovation and faces the competition with fast and secure software deployment.