

Rapport de projet fin d'étude

Marwa Haoudi Rihab Atbir

27 April 2024

Résumé

Abstract

Dédicaces

A nos chers parents, pour tous leurs sacrifices, leur amour, leur tendresse, leur soutien et leurs prières tout au long de nos études. Votre confiance m'a permis de poursuivre mes rêves et de travailler dur pour atteindre mes objectifs.

À Mme. Latifa Oufkir pour son accompagnement, ses conseils et sa patience tout au long de ce projet de fin d'études. Grâce à votre soutien, on a réalisé ce projet avec succès.

À mes amis, pour leur présence et leur soutien dans les bons comme dans les mauvais moments. Vous avez rendu cette aventure inoubliable.

Enfin, à tous ceux qui nous ont aidé et soutenu tout au long de cette expérience, nous vous sommes reconnaissantes pour votre soutien, votre encouragement et vos précieux conseils. Merci du fond du cœur.

Remerciements

Avant tout développement sur cette expérience académique, il apparaît opportun de commencer ce rapport de projet de fin d'étude par des remerciements, à ceux qui m'ont beaucoup appris au cours de ce projet, et même à ceux qui ont eu la gentillesse de faire de ce PFE un moment très profitable.

Je tiens à exprimer mes sincères remerciements dans un premier temps à notre encadrante Mme. Latifa Oufkir pour son accompagnement tout au long du projet ainsi pour le temps et l'attention qu'il a consacrés à nos questions et à nos préoccupations, ainsi d'exprimer ma profonde gratitude à toute l'équipe pédagogique de notre faculté des sciences ; département informatique qui a contribué à notre réussite pendant cette année.

J'adresse aussi mes vifs remerciements aux membres des jurys pour avoir bien voulu examiner et juger notre travail.

Enfin, je tiens à exprimer ma gratitude envers ma famille et mes amis pour leur soutien inconditionnel tout au long de cette expérience.

À tous ces intervenants, je présente mes remerciements, mon respect et ma gratitude.

Liste des abréviations

List of Figures

2.1	Description de l'image	17
3.1	Architecture générale de la solution Ansible	21
3.2	Exemple du fichier d'inventary	22
4.1	Exemple du fichier d'inventary	27
5.1	Installation du package APT.	34
5.2	Mise à jour des packages.	35
5.3	Mise à jour système.	35
5.4	Vérification d'une dépendance.	35
5.5	Installer les dépendances nécessaires	36
5.6	Installer Ansible via pip	36
5.7	la Configuration des paramètres d'eveng.	37
5.8	l'interface web d'EVE-NG	38
5.9	Connexion à EVE-NG avec PuTTY	40
5.10	le répertoire d'images /opt/unetlab/addons/qemu d'EVE	41
5.11	Connexion WinSCP pour se connecter à notre appareil EVE-NG	42
5.12	Transfert de l'image avec WinSCP	42

List of Tables

Contents

Résumé	1
Abstract	2
Dédicace	3
Remerciements	4
Liste des abréviations	5
Introduction générale	10
1 Présentation de la faculté et cadre général du projet	11
1.1 Présentation de la faculté :	11
1.2 Cadre général du projet :	11
1.2.1 Contexte du projet :	11
1.2.2 Problématique :	11
1.2.3 Objectifs du projet :	12
1.2.4 Démarche :	12
1.2.5 Déroulement et Planification du projet :	12
2 Etude comparative des différentes solutions	14
2.1 Benchmarking des outils disponibles :	14
2.1.1 Ansible :	14
2.1.2 Chef :	15
2.1.3 Puppet :	15
2.1.4 Choix des critères :	16
2.1.5 Comparaison et synthèse :	16
2.1.6 Pourquoi Ansible ?	17
2.2 Présentation générale de Ansible:	18
2.2.1 Présentation générale :	18
2.2.2 Le terme Ansible :	18
2.2.3 Notions et Concepts	19
2.2.4 Progression de Ansible avec le Réseau	19

3	: Etude théorique d'Ansible	21
3.1	Architecture de Ansible :	21
3.1.1	Inventory :	22
3.1.2	Playbook :	22
3.1.3	Modules :	22
3.1.4	Facts :	23
3.1.5	Rôles :	23
3.2	Composants supplémentaires :	23
3.2.1	Modèles Jinja2 :	23
3.2.2	Ansible Vault :	23
3.2.3	Mode ad-hoc :	24
3.2.4	Langage YAML :	24
4	Virtualisation	25
4.1	Introduction à la virtualisation :	25
4.1.1	Les types principaux de virtualisation :	25
4.1.2	Les avantages :	26
4.2	Mise en place de l'environnement virtuel :	26
4.2.1	VMware :	26
4.2.2	Avantages de VMware :	27
4.3	Mise en place d'un simulateur :	28
4.3.1	Présentation EVE-NG (Emulated Virtual Environment - Next Generation):	28
4.3.2	Avantages de EVE-ng:	28
4.3.3	Présentation Penetlab :	28
4.3.4	Avantages de Panetlab:	28
4.4	Conclusion :	29
5	Environnement de travail	30
5.1	Installation de VMWare Workstation Pro :	31
5.2	Installation des Packages APT :	34
5.3	Installation d'Ansible sous Ubuntu :	35
5.4	Installation d'EVN-EG dans VMware	36
5.5	Installation d'une image d'appareil réseau dans EVE-NG en	39

Introduction générale

Chapter 1

Présentation de la faculté et cadre général du projet

1.1 Présentation de la faculté :

1.2 Cadre général du projet :

1.2.1 Contexte du projet :

Le contexte du projet d'automatisation de configuration est ancré dans la nécessité croissante des entreprises de simplifier et d'accélérer le déploiement ainsi que la gestion de leurs infrastructures informatiques. Dans un environnement où la demande de flexibilité et de rapidité est primordiale, les méthodes traditionnelles de configuration manuelle deviennent inefficaces et coûteuses. Ainsi, l'automatisation de la configuration émerge comme une solution incontournable pour répondre à ces défis, en permettant aux entreprises d'automatiser et de standardiser les processus de déploiement, de configuration et de gestion des ressources informatiques.

1.2.2 Problématique :

La problématique principale que ce sujet vise à résoudre est l'automatisation et la gestion efficace des configurations réseau qui rentre dans le domaine Net-Devops pour un environnement composé de routeurs, de commutateurs et de serveurs. La configuration manuelle de ces équipements est une tâche sujette aux erreurs, surtout lorsque le nombre d'équipements augmente. De plus, la diversité des technologies et des plateformes utilisées complique davantage la gestion et la coordination des configurations. L'automatisation des configurations réseau à l'aide d'un outil comme Ansible permet de rationaliser et de standardiser ce processus, réduisant ainsi les risques d'erreurs humaines.

1.2.3 Objectifs du projet :

Les objectifs du projet comme suit :

Réduire les délais de déploiement : L'automatisation des processus de configuration vise à accélérer le déploiement des infrastructures informatiques en éliminant les étapes manuelles fastidieuses. L'objectif est de réduire les délais de mise en service des nouveaux systèmes, des applications et des services.

Améliorer la cohérence et la fiabilité : En remplaçant les processus manuels par des scripts et des modèles automatisés, le projet vise à garantir une cohérence totale dans la configuration des systèmes. Cela permet d'éliminer les erreurs humaines et de garantir la fiabilité des environnements informatiques.

Accroître la flexibilité et l'adaptabilité : L'automatisation de la configuration permet aux entreprises de s'adapter plus rapidement aux changements en facilitant la mise à jour et la modification des configurations. Cela inclut la capacité à scaler les infrastructures en fonction des besoins fluctuants et à intégrer de nouvelles technologies sans perturber l'existant.

Renforcer la sécurité : En appliquant des standards de configuration et des politiques de sécurité de manière automatisée, le projet vise à renforcer la sécurité des infrastructures informatiques. Cela inclut la mise en place de configurations sécurisées par défaut et la surveillance constante de la conformité aux normes de sécurité.

Optimiser les ressources : En réduisant la dépendance à l'égard des processus manuels, le projet cherche à optimiser l'utilisation des ressources humaines et matérielles. Cela permet de libérer les équipes IT pour des tâches à plus haute valeur ajoutée et de réduire les coûts opérationnels liés à la gestion manuelle des configurations.

1.2.4 Démarche :

La démarche pour atteindre ces objectifs implique plusieurs étapes :

- Proposition et étude de la solution Ansible
- Etude des technologies utilisées dont les IOS (switch et routeur)
- Installation d'Eve-ng et création du lab
- Mise en place de la solution Ansible et déploiement
- Installation d'Ansible sur Ubuntu

1.2.5 Déroulement et Planification du projet :

a Diagramme de Pert :

Ce diagramme va nous permettant de visualiser dans le temps les diverses tâches composant le projet. Il s'agit d'une représentation d'un graphe, qui permet de représenter graphiquement l'avancement de notre projet.

b Diagramme de Gantt :

Ce diagramme va nous permettant de visualiser dans le temps les diverses tâches composant le projet. Il s'agit d'une représentation d'un graphe, qui permet de représenter graphiquement l'avancement de notre projet.

Chapter 2

Etude comparative des différentes solutions

Introduction :

Dans ce chapitre, nous examinerons plusieurs solutions d'automatisation de configuration afin de déterminer laquelle répond le mieux aux besoins spécifiques de notre projet. Cette étude comparative est essentielle pour prendre une décision éclairée et choisir la solution la plus adaptée à nos exigences en termes de flexibilité, de performance et de facilité d'utilisation.

2.1 Benchmarking des outils disponibles :

2.1.1 Ansible :

a Qu'est-ce qu'ansible ?

Ansible est une plateforme d'automatisation de configuration open-source qui se distingue par sa simplicité et sa facilité de déploiement. Il utilise une architecture agentless et fonctionne sur SSH, ce qui le rend particulièrement adapté aux environnements hétérogènes. Ansible offre une syntaxe YAML claire et facile à comprendre pour décrire les tâches à exécuter, ce qui le rend accessible même aux débutants.

b Avantages d'ansible :

- Automatisation simplifiée Ansible est une plateforme simple d'utilisation, facile à installer, à configurer et à maîtriser. En moins de 30 minutes, il est possible d'installer et de configurer le système et d'exécuter des commandes ad hoc pour les serveurs pour résoudre un problème spécifique : réglage de l'heure d'été, synchronisation de l'heure d'été, modification du

mot de passe racine, mise à jour des serveurs, redémarrage des services, etc.

- Courbe d'apprentissage basse Ansible est facile à déployer, car il n'utilise aucun agent ni aucune infrastructure de sécurité personnalisée supplémentaire. Ansible s'appuie également sur YAML, un langage simple pour décrire votre travail d'automatisation via les playbooks. Les playbooks poussent les paramètres souhaités sur les hôtes définis dans l'inventaire et peuvent même être exécutés ad hoc (via la ligne de commande, sans définition dans les fichiers).
- Automatisation immédiate Dès le moment où vous pouvez envoyer une requête ping aux hôtes via Ansible, vous pouvez commencer à automatiser votre environnement. Commencez par de petites tâches, suivez les bonnes pratiques, hiérarchisez les tâches sources de valeur pour l'entreprise, résolvez des problèmes majeurs, gagnez du temps et améliorez la productivité.

2.1.2 Chef :

Créé en 2009, Chef est un logiciel de gestion de configurations pour le développement informatique. Il permet d'automatiser des infrastructures serveur comme les systèmes d'exploitation, à partir de "recettes". Elles entrent en ligne de compte pour le packaging et le provisioning. Il utilise un modèle client-serveur avec un agent installé sur les nœuds à gérer. Chef propose une grande flexibilité grâce à son langage de configuration DSL (Domain Specific Language), mais peut être plus complexe à mettre en place et à gérer que d'autres solutions.

2.1.3 Puppet :

Conçu comme un projet open-source et développé en langage Ruby, Puppet est un outil de gestion de configuration et d'orchestration de serveurs. On l'utilise pour automatiser la configuration et la gestion de systèmes informatiques, en assurer la cohérence et réduire les erreurs.

Il est notamment exploité pour installer et configurer des logiciels, gérer des utilisateurs et des groupes, configurer des services réseau, surveiller l'état des systèmes et bien plus encore.

Un cas d'usage courant est la gestion de configuration de grands parcs de serveurs. En effet, Puppet permet de déployer des configurations cohérentes à grande échelle et garantit que les systèmes répondent aux normes de l'entreprise.

Au fil des années, Puppet est devenu très populaire auprès des administrateurs système et des entreprises de toutes les tailles. L'entreprise Puppet Labs (désormais appelée Puppet Inc) fondée par Kanies pour son développement a

levé 5 millions de dollars en 2011 en financement de série B.

L'outil a évolué pour s'enrichir de fonctionnalités d'orchestration, permettant de gérer des tâches à travers plusieurs serveurs. Il est disponible en version open source et en version entreprise.

2.1.4 Choix des critères :

Dans la catégorie des outils de gestion de configuration trois outils sont les plus connus et utilisés par les entreprises : Ansible, Puppet et Chef.

Les critères qui vont orienter notre choix d'outil de gestion de configuration et automatisation sont :

- Sans Agent : Pouvoir contrôler des machines sans configuration apriori ou installation d'agent.
- Langage
- Dépendance client : les logiciels pré requis pour configurer une machine.
- Mécanisme : de partage de configuration, push ou pull.
- Installation : Complexité de mettre en place l'outil pour un environnement de production.

2.1.5 Comparaison et synthèse :

Dans cette section, nous allons comparer les performances des trois outils d'automatisation de la gestion des configurations - Ansible, CHEF et Puppet - en fonction des critères définis précédemment. Nous évaluerons également leur adéquation par rapport à nos besoins spécifiques en matière d'automatisation de la gestion des configurations.

- CHEF et Puppet ont une courbe d'apprentissage légèrement plus abrupte en raison de leur approche basée sur l'état désiré du système et de la nécessité d'installer des agents sur les nœuds cibles. Cependant, ils offrent un contrôle plus précis sur les configurations.

- CHEF et Puppet offrent également de bonnes performances, bien que leur utilisation d'agents sur les nœuds cibles puisse entraîner une légère surcharge réseau et des temps de déploiement légèrement plus longs.

- CHEF et Puppet offrent également une grande flexibilité, mais peuvent être plus complexes à configurer pour certains cas d'utilisation spécifiques en raison de leur approche plus axée sur l'état.

- CHEF et Puppet nécessitent l'installation d'agents sur chaque nœud, ce qui peut poser des défis de gestion à grande échelle.




<div> <div>Outil</div> <div>Critère</div> </div>			
Sans Agent	Oui	Non	Non
Langage	Python	Ruby	Ruby
Dépendance client	Python, sshd, bash	Ruby, sshd, bash	Ruby
Mécanisme	Push	Pull	Pull
Approche	Procédural	Procédural	Déclarative
Installation	Facile	Peu facile	Difficile
Contributeur	3432	522	492

Figure 2.1: Description de l'image

- CHEF et Puppet nécessitent une configuration et une gestion plus complexes, ce qui peut augmenter les efforts de maintenance, en particulier dans les environnements distribués.
- CHEF et Puppet ont également des communautés actives, bien que légèrement moins étendues que celle d'Ansible.

2.1.6 Pourquoi Ansible ?

Après une analyse approfondie des différentes solutions d'automatisation de la gestion des configurations disponibles, nous avons conclu que Ansible est la solution la plus adaptée à nos besoins.

L'avantage d'Ansible par rapport aux autres outils de gestion de configuration est son aspect sans agent, c'est-à-dire, il n'a pas besoin d'une configuration a priori pour contrôler une machine. Donc, Ansible reste le plus adapté et facile à mettre en œuvre en comparaison avec Chef et Puppet.

Conclusion :

En conclusion, Ansible offre une combinaison unique de simplicité, de flexibilité, de performance et de support communautaire qui en fait la solution idéale pour automatiser la gestion des configurations dans notre environnement. Son déploiement facile, son architecture agentless et sa grande adaptabilité en font un choix solide qui répond à nos besoins présents et futurs en matière

d’automatisation.

2.2 Présentation générale de Ansible:

2.2.1 Présentation générale :

Ansible est un outil open-source de provisionnement de logiciels, de gestion des configurations et de déploiement d’applications qui permet de faire coder une infrastructure IT en ce compris le support de ses applications. Il permet d’automatiser la plupart des tâches de gestion d’infrastructures. Il fonctionne sur de nombreux systèmes de type Unix, et il peut configurer aussi bien des systèmes de type Unix que Microsoft Windows ou autres. Il comprend son propre langage déclaratif pour décrire la configuration du système. Ansible est sans agent, se connectant temporairement à distance via SSH ou Windows Remote Management (permettant l’exécution à distance de PowerShell) par exemple pour effectuer ses tâches.

Ansible gère les différents noeuds avec un accès à distance natif (tels que les protocoles SSH ou Remote PowerShell ou encore des APIs natives, et bien d’autres). Il ne nécessite l’installation d’aucun logiciel supplémentaire à distance. Il offre des capacités de parallélisation, de collecte de métadonnées et de gestion des états. Cet aspect de conception “sans agent” installé sur le périphérique est important car il réduit les besoins sous-jacents d’infrastructure pour démarrer une gestion. Les modules fonctionnent grâce à JSON et à la sortie standard et ils peuvent être écrits dans n’importe quel langage de programmation. Ansible utilise notamment YAML pour exprimer des descriptions réutilisables de systèmes, il fournit des sorties en JSON, il traite les variables grâce à des modèles Jinja2.

Le logiciel Ansible a été conçu par un ancien employé Red Hat, Michael DeHaan, également auteur de l’application de serveur de “provisionnement” Cobbler et co-auteur du framework Func pour l’administration à distance. Le code source du logiciel est sous licence GNU General Public v3.0. Red Hat a racheté la société Ansible, Inc. en octobre 2015. 1

2.2.2 Le terme Ansible :

Une “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. Le Guin dans son roman de science-fiction, *Le Monde de Rocannon*. Elle en détaillera plus tard le concept dans *Les Dépossédés* (1974). L’idée est notamment reprise par d’autres auteurs de livres de science-fiction

et des jeux vidéos, la communication étant basée sur l'état d'énergie réciproque de deux particules jumelles. Par ailleurs, Ansible est le titre d'un magazine anglo-saxon consacré à la science-fiction. Enfin, le terme "ansible" peut faire référence à un système de communication hyperspace instantané fictif 2.

Comme une "ansible", une tâche Ansible se déploie simultanément sous forme de module sur les cibles désignées.

2.2.3 Notions et Concepts

Ansible repose sur plusieurs notions et concepts clés :

- **Playbooks** : Les playbooks sont des fichiers YAML qui décrivent les tâches à effectuer par Ansible sur un ensemble de nœuds cibles. Ils définissent l'état désiré du système et les actions à entreprendre pour atteindre cet état.
- **Modules** : Les modules sont des morceaux de code exécutables par Ansible sur les nœuds cibles pour réaliser des actions spécifiques. Ils peuvent être utilisés dans les playbooks pour effectuer des opérations telles que l'installation de logiciels, la configuration de services, la gestion des fichiers, etc.
- **Inventaire** : L'inventaire est un fichier qui répertorie les hôtes sur lesquels Ansible doit agir. Il peut être statique ou dynamique et permet de regrouper les nœuds en fonction de différents critères tels que leur rôle, leur emplacement géographique, etc.
- **Adhoc commandes** : Les commandes ad-hoc permettent d'exécuter des tâches rapidement sans avoir à créer de playbook. Elles sont utiles pour des actions ponctuelles telles que la vérification de l'état d'un service ou la mise à jour d'un package sur un ensemble de nœuds.

2.2.4 Progression de Ansible avec le Réseau

Au fil du temps, Ansible a évolué pour inclure des fonctionnalités spécifiques au réseau, lui permettant de gérer efficacement les équipements réseau. Ces fonctionnalités comprennent :

- **Modules réseau** : Ansible fournit une collection de modules spécifiques au réseau qui permettent de configurer, surveiller et gérer des équipements réseau tels que des commutateurs, des routeurs, des pare-feu, etc.
- **Intégration avec les API réseau** : Ansible peut interagir avec les API exposées par les équipements réseau pour automatiser des tâches de configuration

et de gestion. Cela permet d'étendre les capacités d'automatisation d'Ansible pour répondre aux besoins spécifiques des environnements réseau.

- Gestion centralisée : Ansible permet de gérer l'infrastructure réseau à partir d'une plateforme centralisée, ce qui facilite la configuration et la coordination des équipements réseau à grande échelle.

En résumé, Ansible offre une solution complète et flexible pour l'automatisation des tâches de gestion des configurations, avec des fonctionnalités spécifiques au réseau qui lui permettent de s'intégrer parfaitement dans les environnements informatiques modernes.

Chapter 3

: Etude théorique d'Ansible

Introduction

Pour bien mener ce projet, il est nécessaire de faire un étude théorique détaillé sur ansible. Alors dans ce chapitre, il sera composé de cinq parties. La première va porter sur des Composants principaux d'Ansible, la deuxième sur composants supplémentaires, la troisième sur les éléments de Configuration d'Ansible, la quatrième sur Graphical User Interface et puis la dernière partie concerne Documentation d'Ansible. Composants principaux de Ansible

3.1 Architecture de Ansible :

L'architecture Ansible est facile à comprendre

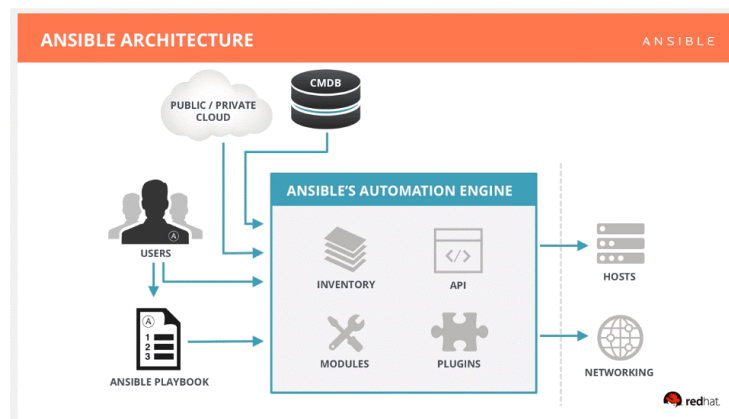


Figure 3.1: Architecture générale de la solution Ansible

3.1.1 Inventory :

Le fichier inventory contient une liste des hôtes (hosts:) (généralement leurs adresses IP ou DNS) que vous souhaitez configurer ou gérer, il est au format INI par défaut. Les hôtes d'un inventory peuvent être divisés en groupes plus petits pour faciliter la gestion et la configuration, il y'a aussi des variables (vars:) attachées à ces hôtes et à ces groupes. Chaque groupe peut exécuter différentes tâches.

```
all:
  hosts:
    server1:
    server2:
      host_var: valueZ
  vars:
    group_all_var: valueY
  children:
    production_server:
      database_server:
      oracle_server:
        hosts:
          server4
      mysql_server:
        hosts:
          server5:
          server6:
      vars:
        g2_var2: valueX
```

Figure 3.2: Exemple du fichier d'inventory

L'option `-i` suivie du chemin du fichier d'inventory, elle est utilisée avec `ansible-playbook` pour préciser l'emplacement du fichier inventory. On peut aussi préciser le chemin du fichier d'inventory par défaut dans le fichier de configuration `ansible.cfg` avec l'entrée `inventory` sous la section `[defaults]`.

3.1.2 Playbook :

Un playbook est un fichier script de configuration d'Ansible qui écrit en langage YAML pour exécuter l'ensemble des tâches et d'étapes de configuration à effectuer sur les hôtes de manière séquentielle sur tout ou partie de l'inventory càd un groupe déjà défini dans l'inventory.

Chaque playbooks est composé d'un ou plusieurs "playbook" exposés sous forme d'une liste. Un "playbook" organise des tâches (tasks) en play. Mais il est de bonne pratique de L'organiser en "rôles". Un "rôle" ajoute un niveau d'abstraction dans l'exécution des tâches d'un playbook.

3.1.3 Modules :

Les modules Ansible sont des programmes utilisés pour exécuter des tâches, ils écrivent principalement en Python. Aussi, chaque tâche utilise un module qui peut prendre des paramètres pour être exécuté de manière personnalisée. Ansible exécute ensuite ces modules (via SSH par défaut) grâce au format JSON

sur la sortie standard et les supprime lorsque l'action est terminée. [3]

3.1.4 Facts :

Les Facts sont des informations collectées par Ansible et que l'on peut appeler via des variables ansible et pour les périphériques de Cisco appeler via des variables ansible net. Par défaut, Ansible collecte les Facts des cibles avant l'exécution des tâches du playbook. Ce comportement se contrôle sur le playbook avec la directive `gather facts` une valeur booléenne.

3.1.5 Rôles :

Les rôles sont des playbooks génériques, qui peuvent être intégrés dans d'autres playbooks. Ce concept est essentiel pour créer des tâches complexes. En fait, pour rendre les playbooks lisibles, il vaut mieux construire des rôles qui peuvent être partagés, distribués et assemblés, plutôt que créer un playbook qui sera difficile à maintenir et complexe à utiliser et peu lisible.

3.2 Composants supplémentaires :

3.2.1 Modèles Jinja2 :

Ansible utilise un module de "Templates" ou modèles, permettant de créer des fichiers scripts pour la gestion des équipements, et met en forme des données, il s'appuie sur Jinja2 qui permet de gérer des listes ou des variables, des boucles, des tests logiques, et donc d'étendre la programmabilité dans l'automatisation du réseau. Donc Jinja2 est un langage de gestionnaire des templates écrit pour Python.

Les modèles de données sont transformés en configuration de périphériques. -Les sorties des périphériques peuvent être transformées en documentation (dynamique). - On doit mettre l'extension `j2` qui désigne ce fichier comme étant au format Jinja2 .

3.2.2 Ansible Vault :

Ansible vault est une fonctionnalité d'Ansible qui nous permet de crypter les fichiers qui contiennent des données sensibles telles que les mots de passe ou les clés, plutôt que de les laisser en clair dans les Playbooks ou les rôles.

Pour exécuter un Playbook avec des fichiers de données cryptés par Vault, nous devons fournir le mot de passe Vault, on peut stocker dans un fichier (dans ce

exemple : `pass vault.txt`).

3.2.3 Mode ad-hoc :

Mode Ad-Hoc est la commande ansible en une ligne qui exécute une tâche sur l'hôte cible. Elle vous permet d'exécuter une tâche simple en une ligne sur un hôte ou un groupe d'hôtes définis dans la configuration du fichier d'inventary. Une commande Ad-Hoc n'aura que deux paramètres, le groupe d'hôtes sur lequel vous souhaitez effectuer la tâche et le module Ansible à exécuter.

La commande Ad-Hoc vous donne plus d'avantages pour l'exploration d'Ansible. Vous pouvez d'effectuer de tâche à manière rapide sans créer un playbook.

Dans cette partie, je vais montrer l'utilisation de base du mode Ad-Hoc, je vais l'utiliser pour effectuer une tâche simple :

Ansible -i inventorycisco -c local -m ios command -a "commands='show running-config' .

3.2.4 Langage YAML :

YAML, ou YAML Ain't Markup Language, est un langage de sérialisation de données couramment utilisé dans Ansible pour définir les configurations et les tâches à exécuter. Voici quelques points clés sur l'utilisation de YAML dans Ansible :

- Syntaxe YAML : YAML utilise une syntaxe claire et lisible. Il est basé sur l'indentation pour définir la structure des données. Les listes sont représentées par des tirets (-) et les dictionnaires (ou "maps") sont représentés par des paires clé-valeur.
- Fichiers YAML Ansible : Les fichiers YAML dans Ansible sont utilisés pour définir les playbooks, les rôles, les inventaires, etc.
- Playbooks : Les playbooks Ansible sont des fichiers YAML qui décrivent une série de tâches à exécuter sur des hôtes distants.

Conclusion

Chapter 4

Virtualisation

4.1 Introduction à la virtualisation :

L'introduction à la virtualisation consiste à comprendre les concepts et les principes de base de cette technologie. La virtualisation est une méthode qui permet de créer des environnements virtuels, indépendants les uns des autres, sur une seule machine physique. Cela signifie qu'une machine physique peut héberger plusieurs machines virtuelles (VM) qui fonctionnent de manière isolée les unes des autres.

L'objectif principal de la virtualisation est d'optimiser l'utilisation des ressources matérielles, telles que le processeur, la mémoire et le stockage, en les partageant entre plusieurs machines virtuelles. Chaque VM a son propre système d'exploitation et ses applications, ce qui donne l'impression qu'il s'agit de machines physiques distinctes.

La virtualisation est réalisée à l'aide d'un logiciel appelé hyperviseur, également connu sous le nom de moniteur de machine virtuelle (VMM). L'hyperviseur permet de créer, gérer et exécuter les machines virtuelles. Il fournit une interface entre le matériel physique et les machines virtuelles, permettant ainsi à chaque VM d'accéder aux ressources matérielles de manière contrôlée.

4.1.1 Les types principaux de virtualisation :

Virtualisation de type 1 (native) : Dans ce type de virtualisation, l'hyperviseur s'exécute directement sur le matériel physique, sans système d'exploitation hôte. Les machines virtuelles sont créées et exécutées directement sur l'hyperviseur. Cela offre une performance élevée et une isolation maximale entre les machines virtuelles. Exemples d'hyperviseurs de type 1 : VMware ESXi, Microsoft Hyper-V, Xen.

Virtualisation de type 2 (hébergée) : Dans ce type de virtualisation, l'hyperviseur

s'exécute comme une application sur un système d'exploitation hôte. Les machines virtuelles sont créées et exécutées sur l'hyperviseur, qui utilise les ressources du système d'exploitation hôte. Cela offre une flexibilité et une facilité d'utilisation, mais peut entraîner une légère perte de performance. Exemples d'hyperviseurs de type 2 : VMware Workstation, Oracle VirtualBox.

4.1.2 Les avantages :

- Consolidation des serveurs physiques : La virtualisation permet de consolider plusieurs serveurs physiques en un seul serveur physique, réduisant ainsi les coûts d'achat, de maintenance et de consommation d'énergie.
- Isolation des environnements : Chaque machine virtuelle fonctionne de manière isolée, ce qui signifie que les problèmes sur une VM n'affectent pas les autres. Cela améliore la sécurité et la stabilité du système.
- Flexibilité et évolutivité : Les machines virtuelles peuvent être créées, supprimées ou déplacées facilement, ce qui offre une grande flexibilité et une évolutivité rapide en fonction des besoins.
- Gestion simplifiée : Les outils de gestion de la virtualisation permettent de gérer efficacement les machines virtuelles, y compris leur déploiement, leur surveillance et leur sauvegarde.
- Test et développement : La virtualisation offre un environnement idéal pour le test et le développement d'applications, permettant de créer des environnements isolés.

4.2 Mise en place de l'environnement virtuel :

4.2.1 VMware :

VMware est une société de logiciels de virtualisation et de cloud computing basée à Palo Alto, en Californie. Fondée en 1998, VMware est une filiale de Dell Technologies. EMC Corporation a initialement acquis VMware en 2004, et EMC a été par la suite acquise par Dell Technologies en 2016. Les technologies de virtualisation de VMware sont basées sur son hyperviseur bare-metal ESX/ESXi dans l'architecture x86.

Avec la virtualisation de serveur de VMware, un hyperviseur est installé sur le serveur physique pour permettre à plusieurs machines virtuelles (VMs) de fonctionner sur le même serveur physique. Chaque VM peut exécuter son propre système d'exploitation (OS), ce qui signifie que plusieurs OS peuvent fonctionner sur un seul serveur physique. Toutes les VMs sur le même serveur physique partagent des ressources, telles que la mise en réseau et la RAM. En 2019, VMware a ajouté la prise en charge à son hyperviseur pour exécuter des charges

de travail conteneurisées dans un cluster Kubernetes de la même manière. Ces types de charges de travail peuvent être gérés par l'équipe d'infrastructure de la même manière que les machines virtuelles, et les équipes DevOps peuvent déployer des conteneurs comme ils étaient habitués.

VMware a été fondée par Diane Greene, Scott Devine, Mendel Rosenblum, Edward Wang et Edouard Bugnion, et a lancé son premier produit, VMware Workstation, en 1999. La société a publié son deuxième produit, VMware ESX, en 2001. Le PDG actuel de VMware est Patrick Gelsinger, nommé en 2012.

Les produits VMware incluent des outils de virtualisation, de gestion de réseau et de sécurité, des logiciels de centre de données à logiciel défini et des logiciels de stockage.

4.2.2 Avantages de VMware :

En tant que leader du domaine de la virtualisation, VMware offre un certain nombre de produits et services spécialement conçus pour virtualiser un data-center. Avantages d'utiliser VMware pour la virtualisation :

Des avantages économiques grâce à la division d'un seul serveur physique en plusieurs machines virtuelles Une agilité et une flexibilité informatiques accrues pour répartir votre charge de travail dans l'ensemble de votre infrastructure selon vos besoins

Déploiement, gestion et maintenance des machines virtuelles rationalisés Virtualisation granulaire des réseaux de stockage (SAN) et des matrices de stockage en réseau (NAS) avec volumes virtuels (vVols)



Figure 4.1: Exemple du fichier d'inventory

4.3 Mise en place d'un simulateur :

4.3.1 Présentation EVE-NG (Emulated Virtual Environment - Next Generation):

EVE-NG est une plate-forme de virtualisation réseau qui permet de créer des environnements réseau virtuels complexes en utilisant des machines virtuelles pour simuler des équipements réseau tels que des routeurs, des commutateurs, des pare-feu, etc. Il offre une interface graphique conviviale pour la conception et la gestion de ces topologies virtuelles.

4.3.2 Avantages de EVE-ng:

Flexibilité : EVE-NG prend en charge une large gamme d'appareils réseau virtuels et offre une grande flexibilité pour créer des topologies personnalisées.

Interface graphique conviviale : Son interface utilisateur intuitive permet de créer, configurer et gérer facilement des topologies réseau complexes.

Communauté active : EVE-NG bénéficie d'une communauté active qui partage des topologies prêtes à l'emploi et des connaissances.

Évolutivité : Il peut être déployé sur des serveurs locaux ou sur des plates-formes cloud, offrant ainsi une évolutivité pour répondre aux besoins des entreprises de toutes tailles.

4.3.3 Présentation Penetlab :

Penetlab est une plate-forme de formation à la cybersécurité qui offre des environnements virtuels prêts à l'emploi pour la pratique de divers scénarios d'attaque et de défense. Il propose une série de laboratoires où les utilisateurs peuvent pratiquer des techniques de piratage éthique, des analyses de vulnérabilités et des tests de sécurité.

4.3.4 Avantages de Panetlab:

Scénarios prêts à l'emploi : Penetlab fournit des environnements de laboratoire complets avec des scénarios de formation prêts à l'emploi, ce qui facilite la mise en place rapide de simulations de cyberattaques et de défense.

Orientation vers la sécurité : Contrairement à EVE-NG qui se concentre sur la virtualisation réseau en général, Penetlab est spécifiquement axé sur la cybersécurité, offrant ainsi des outils et des exercices ciblés sur la sécurité informatique.

Didactique : Il est conçu pour être utilisé dans un contexte pédagogique, avec des guides d'utilisation et des instructions détaillées pour aider les apprenants à tirer le meilleur parti des laboratoires.

Accessibilité : Penetlab peut être facilement déployé localement ou sur des plates-formes cloud, offrant ainsi une grande accessibilité aux utilisateurs.

4.4 Conclusion :

EVE-NG et Penetlab sont deux outils complémentaires, mais avec des objectifs différents. EVE-NG est idéal pour la simulation et la formation en matière de réseaux, tandis que Penetlab se concentre spécifiquement sur la cybersécurité et offre des environnements de laboratoire prêts à l'emploi pour la pratique de scénarios d'attaque et de défense. Le choix entre les deux dépendra des besoins spécifiques de l'utilisateur en termes de formation et de simulation. Mais pour nos objectifs, nous utiliserons Eve-ng.

Chapter 5

Environnement de travail

Introduction:

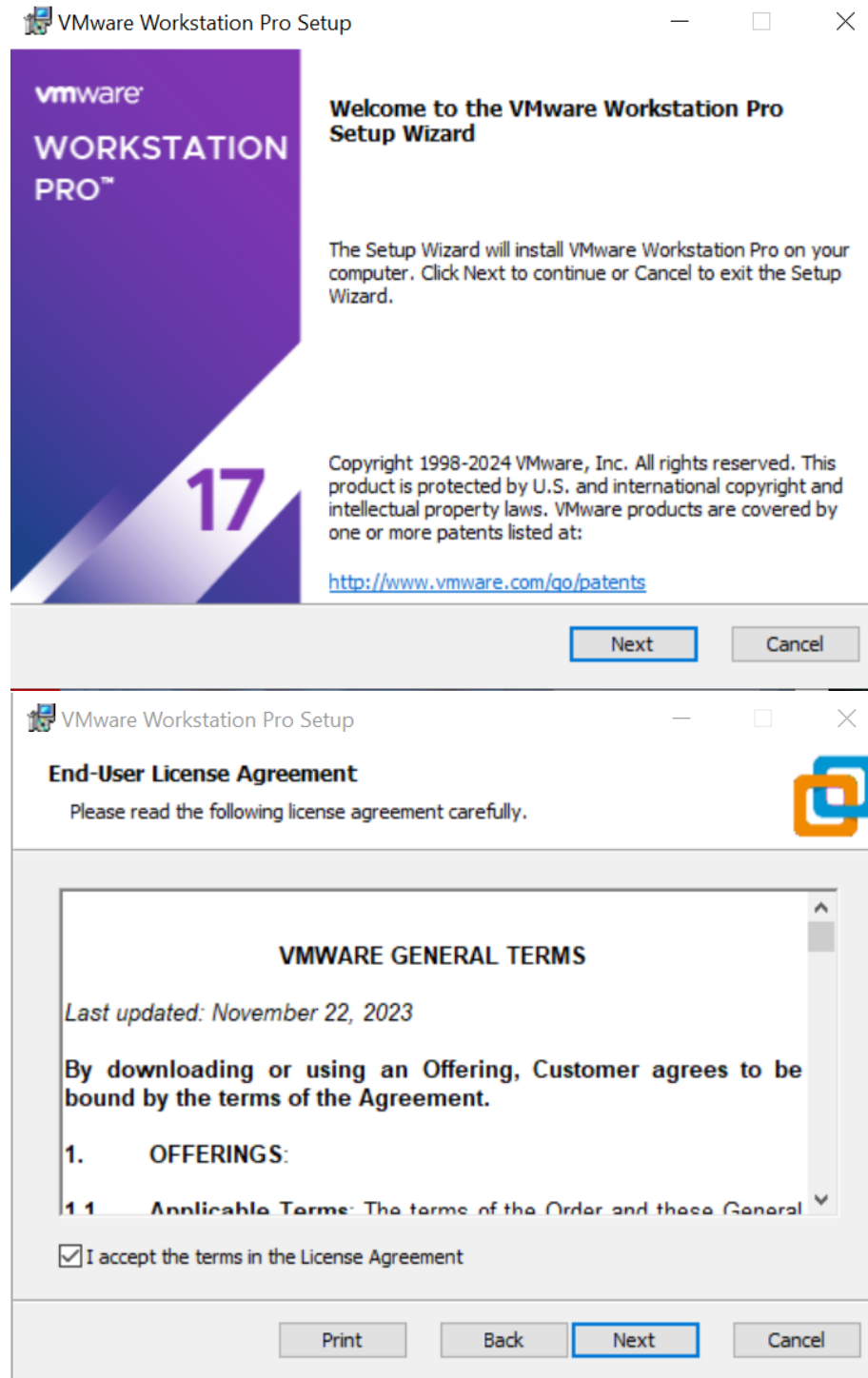
Ce chapitre consistera en la préparation de notre environnement, en expliquant toutes les étapes qui contribueront au bon fonctionnement de notre projet.

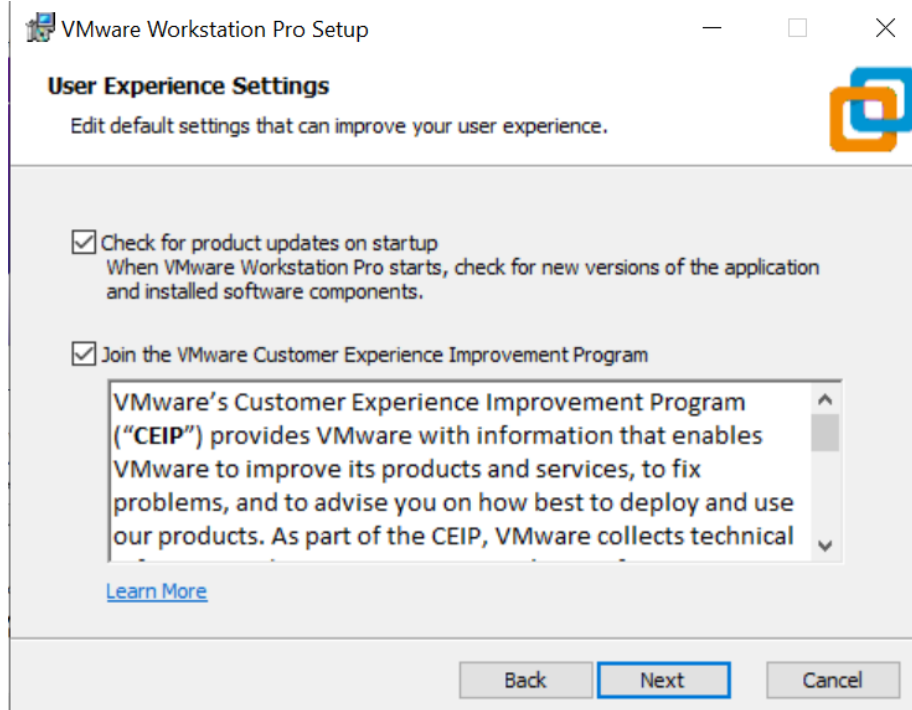
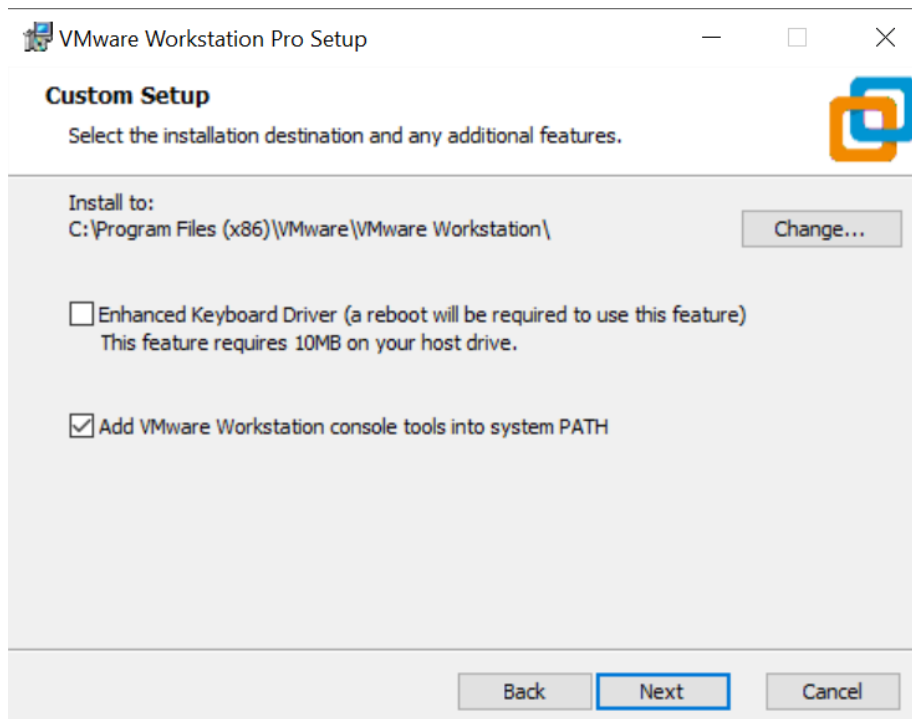
Ubuntu qui est un système d'exploitation de type Unix, très connu dans le monde de la programmation sera notre système d'exploitation principal.

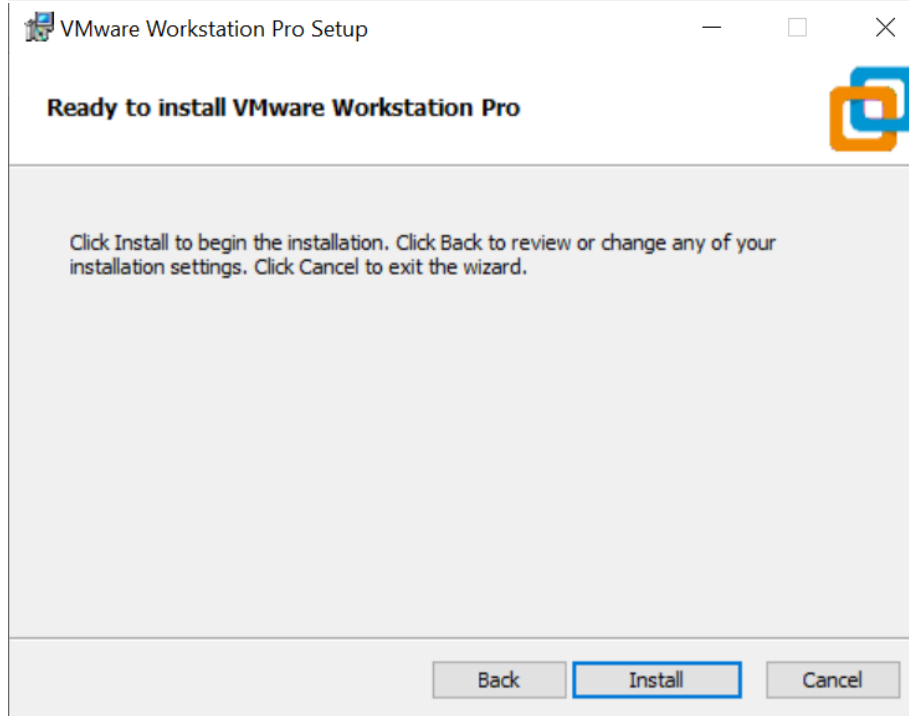
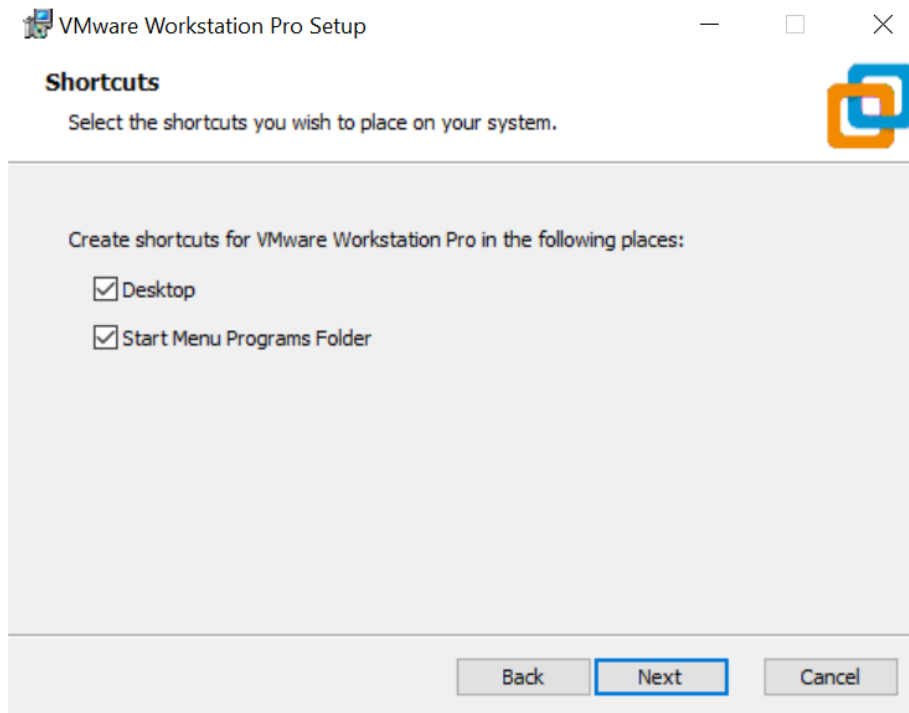
D'abord, nous aurons besoin d'y installer principalement VMWare Workstation pro qui nous permettra d'installer des machines virtuelles, afin de pouvoir gérer différents nœuds. Ainsi que EVN-EG qui est utilisé pour émuler plusieurs systèmes d'exploitation dans un environnement virtuel à l'aide des systèmes 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.

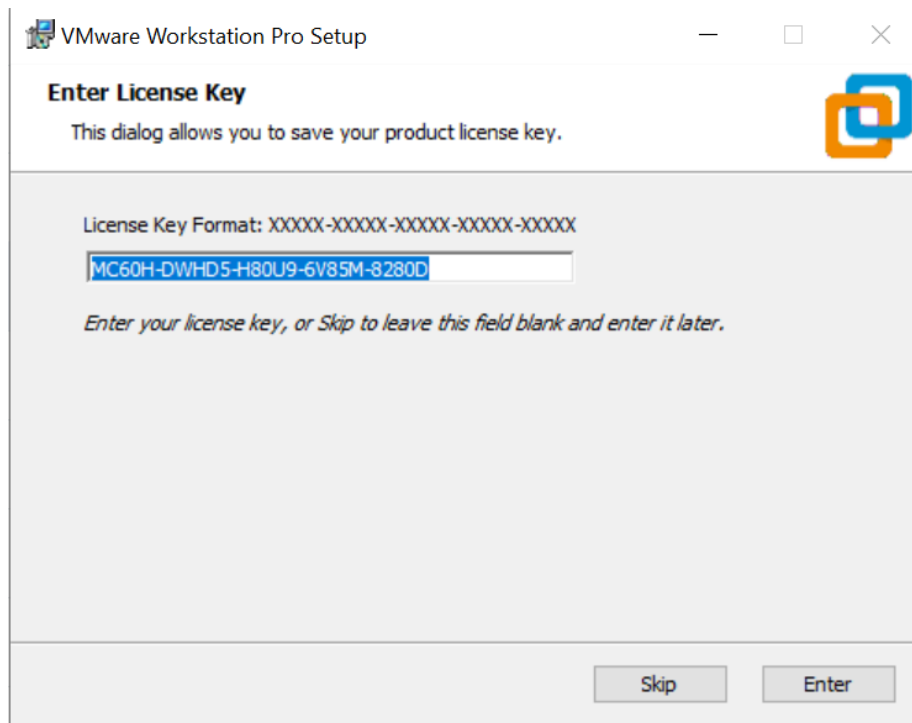
Enfin, nous montrerons comment installer l'outil d'automatisations de notre projet.

5.1 Installation de VMWare Workstation Pro :









5.2 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. Advanced Packaging Tool 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 :

Installation du package APT

```
pfe@pfe-virtual-machine:~$ sudo apt-get install
```

Figure 5.1: Installation du package APT.

Mise à jour des packages.

```
pfe@pfe-virtual-machine:~$ sudo apt-get upgrade
```

Figure 5.2: Mise à jour des packages.

Mise à jour système

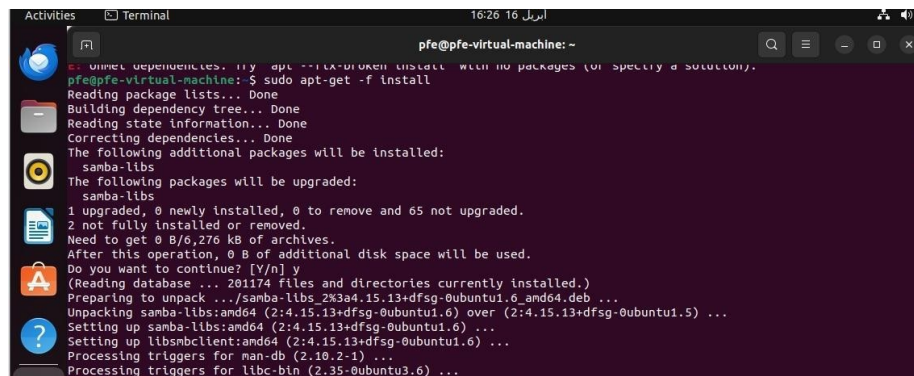
```
pfe@pfe-virtual-machine:~$ sudo apt update
```

Figure 5.3: Mise à jour système.

5.3 Installation d'Ansible sous Ubuntu :

Vérification des dépendances

Pour commencer, vérifiez et résolvez les éventuelles dépendances cassées.



```
pfe@pfe-virtual-machine:~$ sudo apt-get -f install
Reading package lists... Done
Building dependency tree... Done
Reading state information... Done
Correcting dependencies... Done
The following additional packages will be installed:
  samba-lsbs
The following packages will be upgraded:
  samba-lsbs
1 upgraded, 0 newly installed, 0 to remove and 65 not upgraded.
Need to get 0 B/6,276 kB of archives.
After this operation, 0 B of additional disk space will be used.
Do you want to continue? [Y/n] y
(Reading database ... 201174 files and directories currently installed.)
Preparing to unpack .../samba-lsbs_2:4.15.13+dfsg-0ubuntu1.6_amd64.deb ...
Unpacking samba-lsbs:amd64 (2:4.15.13+dfsg-0ubuntu1.6) over (2:4.15.13+dfsg-0ubuntu1.5) ...
Setting up samba-lsbs:amd64 (2:4.15.13+dfsg-0ubuntu1.6) ...
Setting up libsnbclient:amd64 (2:4.15.13+dfsg-0ubuntu1.6) ...
Processing triggers for man-db (2.10.2-1) ...
Processing triggers for libc-bin (2.35-0ubuntu3.6) ...
```

Figure 5.4: Vérification d'une dépendance.

l'installation via pip :

Ensuite, vous pouvez installer Ansible via pip (le gestionnaire de packages Python).

```

pfe@pfe-virtual-machine:~$ sudo apt install python3 python3-pip
Reading package lists... Done
Building dependency tree... Done
Reading state information... Done
python3 is already the newest version (3.10.6-1~22.04).
python3 set to manually installed.
The following additional packages will be installed:
  binutils binutils-common binutils-x86-64-linux-gnu build-essential dpkg-dev
  fakeroot g++ g++-11 gcc gcc-11 javascript-common libalgorithm-diff-perl
  libalgorithm-diff-xs-perl libalgorithm-merge-perl libasan6 libbinutils
  libc-dev-bin libc-devtools libc6-dev libcc1-0 libcrypt-dev libctf-nobfd0
  libctf0 libdpkg-perl libexpat1 libexpat1-dev libfakeroot
  libfile-fcntllock-perl libgcc-11-dev libitm1 libjs-jquery libjs-sphinxdoc
  libjs-underscore liblsan0 libnsl-dev libpython3-dev libpython3.10-dev
  libstdc++-11-dev libtirpc-dev libtsan0 libubsan1 linux-libc-dev
  lto-disabled-list make manpages-dev python3-dev python3-distutils
  python3-setuptools python3-wheel python3.10-dev rpcsvc-proto zlib1g-dev
Suggested packages:
  binutils-doc debian-keyring g++-multilib g++-11-multilib gcc-11-doc
  gcc-multilib autoconf automake libtool flex bison gcc-doc gcc-11-multilib

```

Figure 5.5: Installer les dépendances nécessaires .

```

pfe@pfe-virtual-machine:~$ sudo pip3 install ansible
Collecting ansible
  Downloading ansible-9.4.0-py3-none-any.whl (46.4 MB)
    46.4/46.4 MB 8.9 MB/s eta 0:00:00
Collecting ansible-core==2.16.5
  Downloading ansible_core-2.16.6-py3-none-any.whl (2.3 MB)
    2.3/2.3 MB 14.6 MB/s eta 0:00:00
Collecting resolvelib<1.1.0,>=0.5.3
  Downloading resolvelib-1.0.1-py2.py3-none-any.whl (17 kB)
Collecting packaging
  Downloading packaging-24.0-py3-none-any.whl (53 kB)
    53.5/53.5 KB 11.9 MB/s eta 0:00:00
Requirement already satisfied: cryptography in /usr/lib/python3/dist-packages (from ansible-core==2.16.6)
Collecting Jinja2>=3.0.0
  Downloading Jinja2-3.1.3-py3-none-any.whl (133 kB)
    133.2/133.2 KB 3.2 MB/s eta 0:00:00

```

Figure 5.6: Installer Ansible via pip .

5.4 Installation d'EVN-EG dans VMware

la Configuration des paramètres

la Configuration des paramètres de la machine virtuelle eve ng tels que la quantité de RAM, la capacité du disque dur, le nombre de cœurs de processeur, etc.

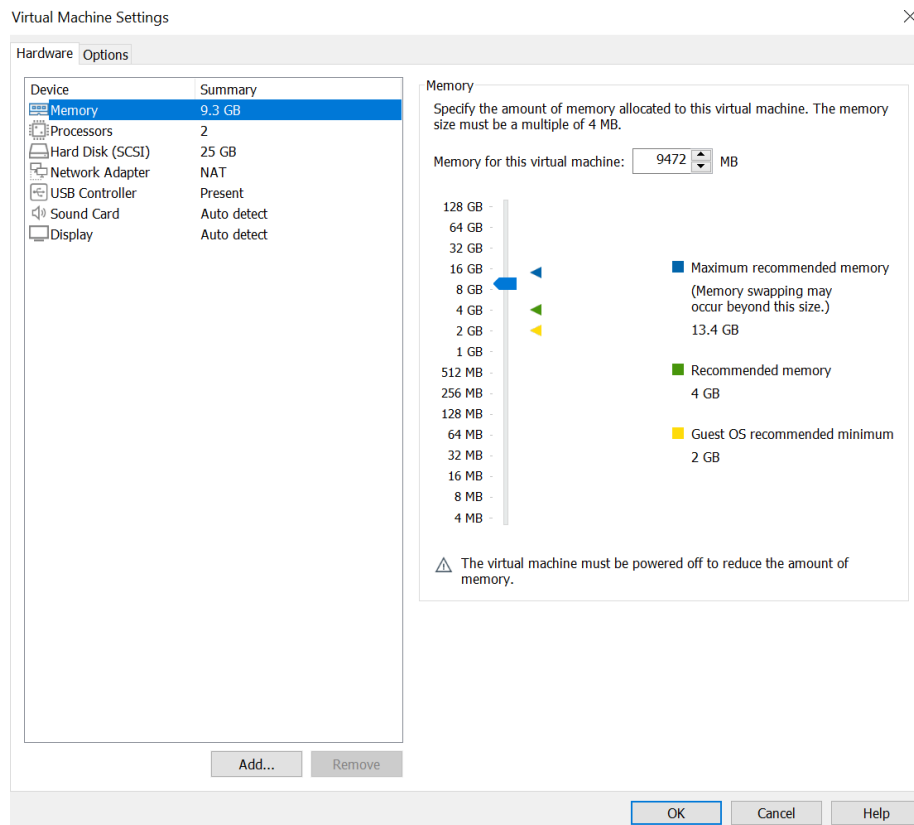


Figure 5.7: la Configuration des paramètres d'eveng.

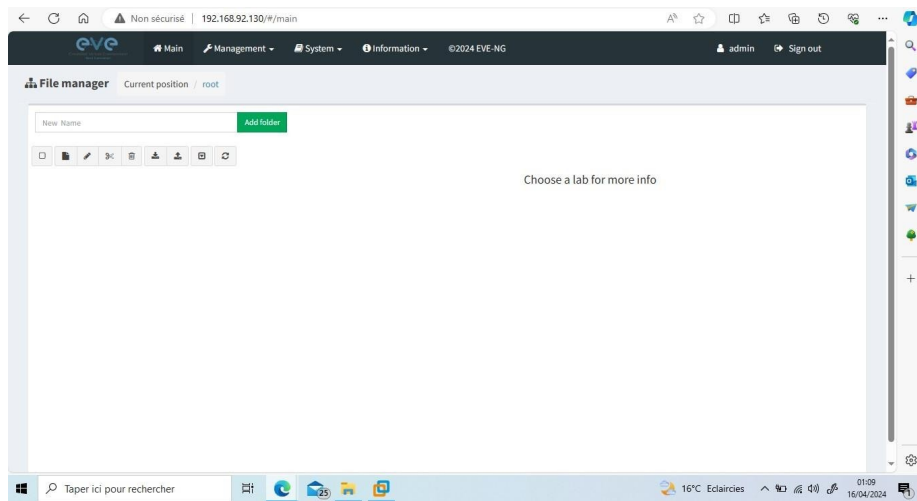


Figure 5.8: l'interface web d'EVE-NG

....

5.5 Installation d'une image d'appareil réseau dans EVE-NG en

Téléchargement de l'image :

- Téléchargement de l'image : Tout d'abord, nous sommes assurés de télécharger une image du périphérique réseau que nous souhaitons utiliser lors de l'événement, et nous sommes également assurés que l'image était compatible avec l'événement.

Connectez-vous à EVE-NG à l'aide de PuTTY:

- Nous avons exécuté PuTTY sur notre ordinateur.
- Dans la fenêtre PuTTY, nous avons saisi l'adresse IP de notre appareil EVE-NG dans le champ Nom d'hôte (ou adresse IP) .
- Nous nous sommes assurés que le port était défini sur 22 (valeur par défaut pour SSH).
- Cliquer sur Ouvrir pour ouvrir la session SSH.

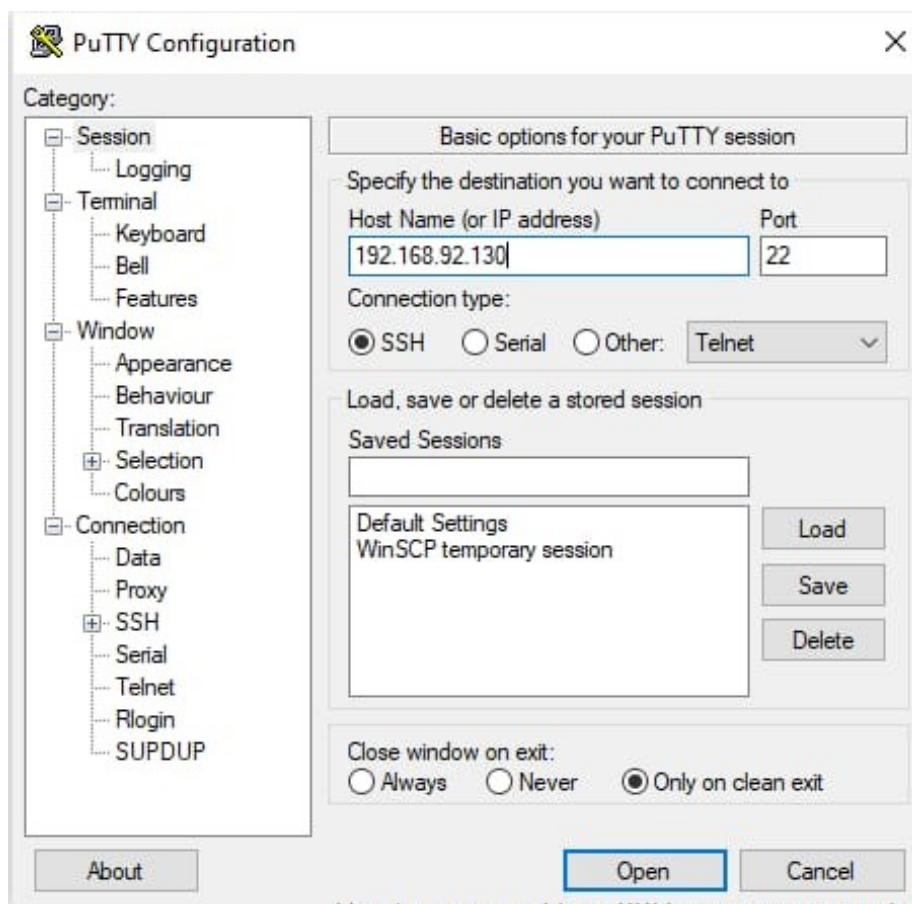


Figure 5.9: Connexion à EVE-NG avec PuTTY

Exemple d'installation d'une image de routeur

la création le répertoire d'images sur /opt/unetlab/addons/qemu d'EVE, cet exemple concerne la première image du tableau ci-dessus. Il s'agit de l'image du routeur vios L3. Selon notre table de dénomination d'image , nous devons créer un dossier d'images commençant par vios- .

```
root@eve-ng:/opt/unetlab/addons/qemu/vios-adventerprisek9-m.SPA.159-3.M6# ls
virtioa.qcow2
root@eve-ng:/opt/unetlab/addons/qemu/vios-adventerprisek9-m.SPA.159-3.M6#
```

Figure 5.10: le répertoire d'images /opt/unetlab/addons/qemu d'EVE

Transfert de l'image à l'aide de WinSCP:

- Nous avons exécuté WinSCP sur notre ordinateur.
- Dans la fenêtre de connexion WinSCP, nous avons entré l'adresse IP, le nom d'utilisateur et le mot de passe de l'appareil EVE-NG.
- Nous avons cliqué sur Connexion pour nous connecter à notre appareil EVE-NG.

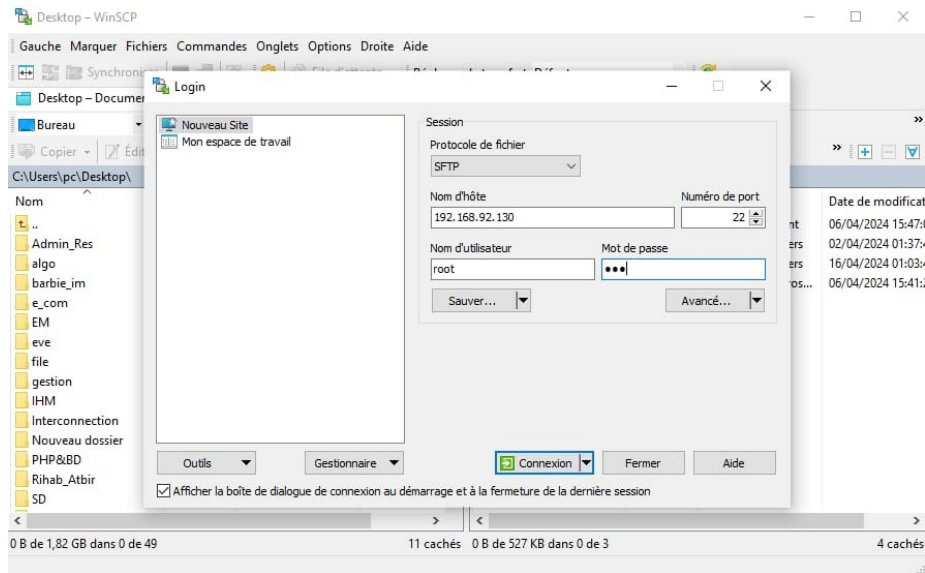


Figure 5.11: Connexion WinSCP pour se connecter à notre appareil EVE-NG

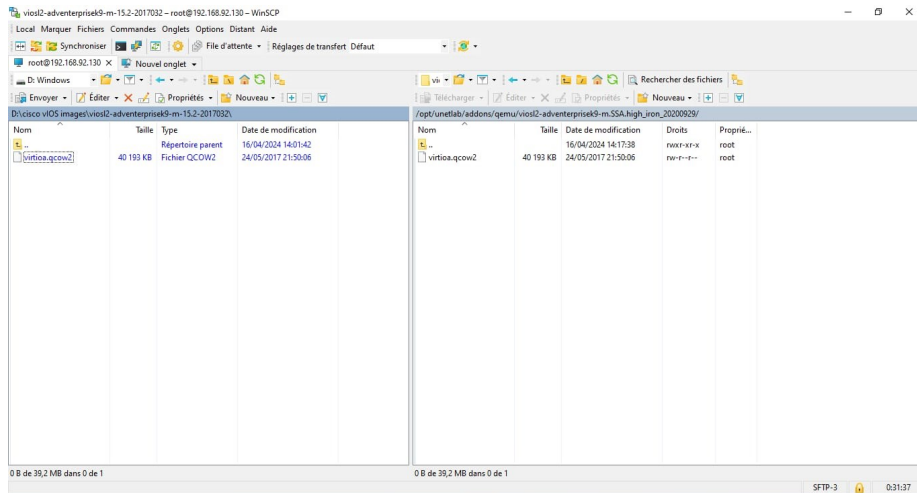


Figure 5.12: Transfert de l'image avec WinSCP