

**ECOLE SUPERIEURE PRIVEE DES TECHNOLOGIES
D'INFORMATION ET DE MANAGEMENT DE NABEUL**



Mémoire de Fin d'Etudes

SUJET

***Mise En Place D'une Plateforme De Configuration Et
D'automatisation Des Architectures Réseaux***

**Présenté en vue de l'obtention du titre
D'INGENIEUR EN INFORMATIQUE SPECIALITE SERVICES ET
RESEAUX**

**Elaboré par
*Ben Mansour Ahmed***

**Encadrant Académique
Mr Hichem Nouaari**

**Encadrant Professionnel
Mr Tarek Ben Romdhane**

DEDICATION

Je dédie ce travail à :

À vous, mes chers parents tawfik et raja, source de mon bonheur et secret de ma force. Aucune dédicace ne saurait exprimer pleinement le respect, l'amour éternel et la gratitude que j'éprouve pour vos sacrifices en faveur de mon éducation et de mon bien-être. Puissent mes réalisations être l'accomplissement de vos souhaits tant espérés, le fruit de vos innombrables sacrifices, bien que je ne puisse jamais suffisamment vous remercier. Je prie Dieu de vous garder à mes côtés, de vous accorder une vie longue et heureuse, et de faire en sorte que je ne vous déçoive jamais.

A tous ceux qui me sont chers...

REMERCIEMENTS

Il est essentiel de s'acquitter des dettes de reconnaissance accumulées au fil de ce travail. C'est donc avec plaisir que nous dédions ces quelques lignes en signe de gratitude et de profonde reconnaissance à tous ceux qui, de près ou de loin, ont contribué à la réalisation de ce projet.

Tout d'abord, je tiens à exprimer ma profonde gratitude envers Monsieur Nouaari Hichem. Votre soutien constant, vos orientations stratégiques et votre expertise approfondie dans le domaine ont été d'une aide inestimable pour moi. Votre capacité à guider et à inspirer vos étudiants a été une source de motivation exceptionnelle. Je suis extrêmement reconnaissant d'avoir eu l'opportunité de bénéficier de vos enseignements et de votre expertise.

Nous tenons à exprimer nos plus sincères remerciements à Monsieur Ben Romdhane Tarek, notre tuteur de stage, pour nous avoir offert l'opportunité d'intégrer son équipe au sein de OneTech Business Solutions. Sa coopération, ainsi que ses précieux conseils et informations, ont grandement contribué à faire de ce stage une expérience des plus enrichissantes.

Enfin, nous souhaitons également exprimer notre gratitude à toute l'équipe de OneTech Business Solutions pour leur accueil chaleureux, leur soutien constant et leur esprit d'équipe exemplaire.

TABLE DES MATIÈRES

Table des figures	vii
Liste des tableaux	x
Introduction Générale	1
1 CADRE GÉNÉRAL DU PROJET	3
1.1 Présentation de l'organisme d'accueil	4
1.1.1 OneTech Holding	4
1.1.2 OneTech Business Solutions (OTBS)	4
1.1.3 Les activités de OneTech Business Solution	5
1.1.4 Services	5
1.1.5 Organigramme	6
1.2 Présentation du projet	6
1.2.1 Diagramme de Gantt	7
1.2.2 Etude de l'existant	7
1.2.3 Description de l'existant	8
1.2.4 Problématique	8
1.2.5 Solution Proposée	8
2 Etat de art	10
2.1 Virtualisation	11
2.1.1 Virtualisation des serveurs	11
2.1.2 Virtualisation des réseaux	11
2.1.3 Type d'hyperviseurs	11
2.1.4 Les Pare-feux	13
2.1.5 Pare-feu de nouvelle génération (NGFW)	14

2.1.6	Palo Alto Networks	15
2.2	Central Management (Panorama)	16
2.3	La méthodologie DevOps	18
2.3.1	Les phases de DevOps	18
2.3.2	Les méthodologies DevOps	20
2.3.3	Principe de conteneurisation	22
2.3.4	L'équilibrage de charge	23
2.3.5	Les types de technologie d'équilibrage de charge	23
2.3.6	Outils de gestion de code source	24
2.3.7	Environnement de travail	26
2.3.8	Outil d'infrastructure en tant que code (IaC)	28
2.3.9	Automatisation	29
2.3.10	Comparaison des Outils CI/CD	31
3	implémentation de la solutionn choisie	33
3.1	Architecture reseau	34
3.2	Environnement de travail	35
3.2.1	Environnement matériel	35
3.2.2	Environnement logiciel	35
3.2.3	Configuration des serveurs	36
3.2.4	Configuration du routeur	37
3.2.5	Configuration des commutateur	38
3.2.6	Configuration du pare-feu PALOALTO	40
3.3	Automatisation de la configuration des équipements	46
3.3.1	Installation d'Ansible Automation Platform	46
3.3.2	Configuration d'Ansible Automation Platform	48
3.4	Automatisation du proccess de project	55
3.5	Creation des machines virtuelles	59
3.5.1	Installation et configuration du Terraform	59
3.5.2	Activation VMware Workstation Pro API REST	60
3.5.3	Creation du code Terraform	62
4	Test et validation de la solution	64
4.1	Test et validation de la configuration	65
4.1.1	Test ping vlans	65
4.1.2	Test web serveur	66
4.1.3	Test serveur dhcp	66
4.1.4	Test IPsec	67

4.2 Test Automatisation et la création de virtuelle machine	67
4.2.1 Test la configuration des équipement avec ansible	67
4.2.2 Test configuration des équipement avec jenkins	68
4.2.3 Test de la création de machine virtuelle	68
Conclusion générale et perspectives	70
Annexes	72
A Les playbooks	72
A.1 Les playbook pour les commutateurs	72
A.2 Les playbooks pour les pare-feux	74
Webographie	80

TABLE DES FIGURES

1.1	Logo de la société OneTech [1].	5
1.2	Organigramme OneTech Business Solutions	6
1.3	Diagramme de Gantt	7
2.1	Architecture hyperviseur de type 1 [8]	12
2.2	Architecture hyperviseur de type 2 [8]	13
2.3	Logo Palo Alto Networks [9].	15
2.4	La mise en place de Panorama [10].	17
2.5	Cycle de vie du mouvement DevOps [11].	18
2.6	Les principes de DevOps [12].	20
2.7	L'approche CI/CD [13].	21
2.8	Principe de Conteneurisation [14].	22
2.9	Architecture d'un load balancer matériel [2].	24
2.10	Architecture d'un load balancer logiciel [15].	24
2.11	LOGO GitHub [3]	25
2.12	LOGO emulateur EVE-NG [4]	27
2.13	LOGO Terraform [5]	29
2.14	LOGO ansible [6]	30
2.15	LOGO jenkins [7]	32
3.1	Architecture proposée	34
3.2	Configuration du ssh	36
3.3	Configuration de l'adresse IP sur une interface	37
3.4	Configuration du ospf	37
3.5	Configuration du ssh	38
3.6	Configuration du vlan pour commutateur 1	38

3.7	Configuration du vlan pour commutateur 2	39
3.8	Configuration du ssh pour commutateur 1	39
3.9	Configuration du ssh pour commutateur 2	39
3.10	Configuration initiale d'une interface de Paloalto	40
3.11	Connexion à l'interface graphique de Paloalto	40
3.12	Configuration les interfaces et les subinterface en Paloalto	41
3.13	Configuration l'interface tunnel en Paloalto	41
3.14	Configuration des zones du sécurité en Paloalto	41
3.15	Configuration de virtuelle routeur en Paloalto	42
3.16	Configuration de virtuelle routeur pour le VPN	43
3.17	Configuration du protocole l'ospf en Paloalto	44
3.18	Configuration IKE Gateway en Paloalto	44
3.19	Configuration ipsec tunnel en Paloalto	45
3.20	Configuration DHCP serveur	45
3.21	Ansible version 2.15.9	46
3.22	Téléchargement d'ansible automation platform	46
3.23	Modification dans le fichier d'inventaire	47
3.24	Installation d'Ansible Automation platform	47
3.25	Interface graphique d'ansible Automation platform	48
3.26	Creation des inventaire	48
3.27	Création d'hote pour le commutateur	49
3.28	Exemple du playbook pour la configuration de routeur	50
3.29	Configuration des projets	51
3.30	Configuration des informations d'identification (credential)	51
3.31	Configuration de l'environnement d'exécution	52
3.32	Création du fichier des packages	52
3.33	Construire une collection de contenu Ansible.	53
3.34	Création de conteneur.	53
3.35	Configuration d'environnement d'execusion	54
3.36	Configuration d'un modèle de tâche	54
3.37	Visualisation d'un modèle Workflow	55
3.38	Notre Workflow modèle	55
3.39	Plugin utilisé.	56
3.40	Creation de l'identifiant.	56
3.41	Ajout URL ansible	56
3.42	Ajout URL github projet	57
3.43	Ajout de workflow projet	57
3.44	Installation de ngrok	58

3.45 Ajout token pour ngrok	58
3.46 Création d'un tunnel en local sur le port 8080	58
3.47 Ajout de workflow projet	59
3.48 Verification de l'installation	59
3.49 Activation vmware workstation Pro API REST.	60
3.50 Ouvrir le vWare workstation REST API sur le navigateur	60
3.51 L'accès sur http://127.0.0.1:8697/api/vms pour trouver l'ID de machine virtuelle	61
3.52 Création du fichier "main.tf"	62
3.53 Création du fichier "variable.tf"	63
3.54 Création du fichier "output.tf"	63
4.1 Test ping dans les vlans	65
4.2 Test l'accès au serveur web	66
4.3 Test le serveur dhcp	66
4.4 Test le protocole IPsec sur le pare-feu	67
4.5 Test la configuration avec ansible	67
4.6 Test la configuration avec jenkins	68
4.7 Test de la création d'une machine virtuelle.	68
A.1 Création des vlans.	72
A.2 Configuration des vlans.	73
A.3 Configuration de l'accès sur le pare-feu et de la règle de sécurité.	74
A.4 Création des profil de gestion et des zones sécurité.	75
A.5 Création des interfaces.	76
A.6 Création des sous-interface.	77
A.7 Création de route static.	77
A.8 Création de IKE(Internet Key Exchange)	78
A.9 Configuration de IPsec profil et création d'un canal	79

LISTE DES TABLEAUX

2.1	Tableau comparatif des fonctionnalités de GitHub, GitLab et Bitbucket	25
2.2	Tableau comparatif de GNS3 et EVE-NG	26
2.3	Tableau comparatif des fonctionnalités de Terraform, AWS CloudFormation et Google Cloud Deployment Manager	28
2.4	Tableau comparatif des fonctionnalités de Ansible, Puppet et SaltStack	29
2.5	Tableau comparatif des fonctionnalités de Jenkins, GitLab CI/CD et CircleCI	31
3.1	Caractéristiques de l'ordinateur DELL	35

INTRODUCTION GÉNÉRALE

Dans un monde où la cybersécurité est devenue un enjeu primordial pour les entreprises, la formation continue et la mise à jour des compétences en sécurité réseau sont essentielles. OneTech Business Solutions, consciente de cette réalité, s'engage dans un projet ambitieux de déploiement d'une architecture réseau sécurisée. Ce projet repose sur deux piliers fondamentaux : l'architecture et l'automatisation, visant à créer des laboratoires de formation avancés pour les certifications PCNSE (Palo Alto Networks Certified Network Security Engineer) et PCNSA (Palo Alto Networks Certified Network Security Administrator).

La cybersécurité est une priorité stratégique dans le contexte actuel où les menaces sont omniprésentes et en constante évolution. Les entreprises doivent non seulement se défendre contre ces menaces, mais aussi se préparer à réagir efficacement en cas de cyberattaque. Dans cette optique, la formation continue et l'actualisation des compétences sont cruciales pour assurer une protection optimale des infrastructures réseau. OneTech Business Solutions, avec son projet de déploiement d'une architecture réseau sécurisée, entend répondre à ces défis en proposant une solution innovante et adaptée aux besoins actuels.

C'est dans ce contexte que notre projet de fin d'études prend forme. Il vise à mettre en place une solution de laboratoire automatisé, offrant une plateforme d'apprentissage pratique pour le personnel de l'entreprise et ses partenaires. Cette plateforme permettra de simuler des environnements réseau complexes, facilitant ainsi l'acquisition et le perfectionnement des compétences nécessaires à la maîtrise des technologies de sécurité de Palo Alto Networks. En combinant des aspects théoriques et pratiques, cette initiative permettra d'améliorer significativement la préparation des utilisateurs aux certifications PCNSE et PCNSA, renforçant ainsi la sécurité globale des infrastructures informatiques.

Le présent rapport est structuré en quatre chapitres principaux. Le premier chapitre, intitulé Cadre général du projet, présente l'organisme d'accueil qui nous a accompagnés tout au long de notre stage,

ainsi qu'une description détaillée de l'étude de l'existant. Nous y aborderons également le contexte de notre intervention, les objectifs fixés et les attentes de l'entreprise. Cette section fournira une vue d'ensemble des défis et des opportunités identifiés au début du projet.

Le deuxième chapitre, nommé État de l'art, analyse et développe les notions théoriques liées à notre sujet et les outils choisis pour le déploiement de notre solution. Nous y explorerons les différentes technologies et méthodologies utilisées en cybersécurité, ainsi que les caractéristiques et avantages des produits Palo Alto Networks. Une revue de la littérature et une analyse comparative des solutions existantes seront également effectuées pour situer notre travail dans le contexte des pratiques actuelles.

Le troisième chapitre, Présentation et implémentation de la solution, décrit l'environnement matériel, les logiciels utilisés, l'architecture proposée pour notre solution, ainsi que la configuration manuelle et automatisée de certains équipements. Cette section sera dédiée à la conception technique de notre laboratoire, incluant les choix d'architecture réseau, la configuration des pare-feu, et l'automatisation des tâches de gestion de la sécurité. Des schémas et des explications détaillées illustreront les étapes d'implémentation.

Le quatrième chapitre, Réalisation, tests et évaluation de la solution, détaille les différents tests effectués sur la configuration réalisée. Nous y discuterons des résultats obtenus, des performances du système, et des ajustements apportés pour optimiser la solution. Une évaluation critique des tests permettra de mesurer l'efficacité de notre approche et d'identifier les axes d'amélioration.

Enfin, nous concluons le rapport par une conclusion générale et les perspectives futures du projet. Cette partie synthétisera les principales réalisations et les enseignements tirés de notre travail, tout en ouvrant des pistes pour des développements ultérieurs. Nous y aborderons également l'impact potentiel de notre projet sur l'entreprise et sur la formation en cybersécurité en général.

CHAPITRE 1

CADRE GÉNÉRAL DU PROJET

Introduction

Ce projet a été réalisé dans le cadre de l'obtention du diplôme d'ingénierie en réseau et service à IT Business School (ITBS). Le stage s'est déroulé entièrement à « Onetech Business Solution » dans le but de créer une architecture sécurisée basée sur Palo Alto vers une architecture virtuelle automatisée en utilisant des technologies modernes et des outils d'automatisations.

1.1 Présentation de l'organisme d'accueil

1.1.1 OneTech Holding

OneTech Holding se positionne en tant que groupe privé et leader technologique, offrant des solutions diversifiées dans plusieurs secteurs d'activités tels que la mécatronique, le câblage, l'électronique et l'informatique. Son rayonnement s'étend à l'échelle nationale et internationale.

1.1.2 OneTech Business Solutions (OTBS)

OTBS, filiale du Groupe One Tech, est un pionnier dans l'intégration de solutions IT et Télécom en Tunisie. Bénéficiant de l'expertise du Groupe dans des secteurs clés tels que l'automobile, l'énergie et les télécommunications, OTBS accompagne ses clients dans leur transformation digitale. Avec une équipe de 120 collaborateurs expérimentés, OTBS a réalisé plus de 400 projets par an, générant un chiffre d'affaires annuel de 23 millions de dinars tunisiens. Forte de ses 35 années d'expérience, OTBS offre une gamme complète de services, de la consultation à la mise en œuvre de solutions personnalisées, pour répondre aux besoins évolutifs de ses 300 clients et dans la figure A.9 le logo de la société OneTech.



FIGURE 1.1 – Logo de la société OneTech [1].

1.1.3 Les activités de OneTech Business Solution

L'entreprise One Tech Business Solution se positionne au cœur du secteur des technologies de l'information (IT) en offrant une gamme complète de services et solutions adaptés aux besoins évolutifs de ses clients. Son domaine d'activité englobe trois principaux axes :

- Intégration : Fourniture de solutions technologiques sur mesure, de la consultation à la mise en œuvre, visant à améliorer les processus métiers des clients.
- Distribution : Collaboration avec un réseau de partenaires pour répondre aux besoins spécifiques des clients en matière de technologies de l'information.
- Ingénierie logicielle et services : Offre de services numériques incluant la gestion de réseaux, la sécurité informatique, la mise en place de Data Centers, etc. grâce à des équipes d'experts expérimentées.

Grâce à ses équipes expérimentées, One Tech Business Solution s'engage à accompagner ses clients dans leur transformation numérique et à répondre efficacement à leurs besoins IT les plus complexes.

1.1.4 Services

Avec une approche personnalisée et flexible, cette entreprise propose une large gamme de services informatiques, allant de la consultation à la mise en œuvre et à la maintenance. Son équipe d'experts s'engage à fournir des solutions adaptées aux besoins spécifiques de chaque client, en mettant en avant l'innovation, la fiabilité et l'efficacité opérationnelle. Que ce soit pour des projets de développement logiciel sur mesure, la gestion de réseaux complexes ou la sécurisation des données, One Tech Business Solution se positionne comme un partenaire de confiance pour accompagner ses clients dans leur parcours numérique avec succès.

1.1.5 Organigramme

L'organisation de l'OTBS est présentée par l'organigramme ci-après :



FIGURE 1.2 – Organigramme OneTech Business Solutions

1.2 Présentation du projet

Le projet de déploiement d'une architecture réseau sécurisée chez OneTech Business Solutions est ancré sur deux piliers fondamentaux : l'architecture et l'automatisation. Il s'agit d'établir des laboratoires de formation PCNSE et PCNSA, fournissant une plateforme d'apprentissage pratique pour le personnel de l'entreprise et ses partenaires. Le premier pilier du projet est l'architecture réseau sécurisée. Nous travaillons à concevoir un environnement robuste basé sur Palo Alto, garantissant une simulation précise des scénarios réels pour une expérience d'apprentissage immersive et efficace. Le deuxième pilier est l'automatisation. Notre objectif est de simplifier et d'accélérer les processus de déploiement et de gestion de l'infrastructure des laboratoires. En automatisant ces tâches, nous visons à garantir une efficacité opérationnelle maximale, tout en minimisant les risques d'erreurs humaines.

1.2.1 Diagramme de Gantt

Le diagramme de Gantt montre la planification temporelle de trois phases principales de ton projet, représentées par différentes couleurs :

- **Vert** : Choix de la technologie
- **Rouge** : Implémentation
- **Jaune** : Test

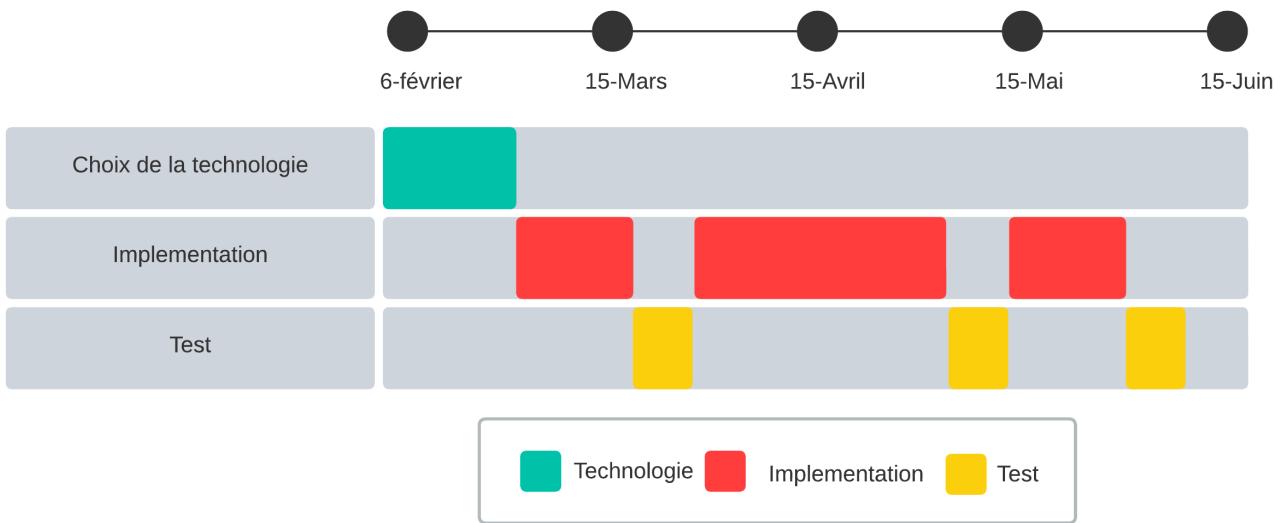


FIGURE 1.3 – Diagramme de Gantt

Dans la phase d'implémentation (rouge), elle est divisée en trois parties distinctes :

- Configuration de l'architecture réseau : Cette étape consiste à définir et mettre en place l'infrastructure réseau nécessaire pour le projet.
- Configuration de Red Hat et de la plateforme d'automatisation Ansible : Ici, l'accent est mis sur la mise en place de Red Hat ainsi que sur la configuration et l'intégration de la plateforme d'automatisation Ansible.
- Création des machines virtuelles avec Terraform : Cette partie du processus implique la création et la gestion des machines virtuelles à l'aide de l'outil Terraform.

Après chaque partie d'implémentation, il y a une phase de test pour vérifier et valider les résultats obtenus.

1.2.2 Etude de l'existant

En combinant une architecture réseau sécurisée et une automatisation avancée, notre projet vise à créer des laboratoires de formation de haute qualité. Ces laboratoires joueront un rôle essentiel dans le renforcement des compétences en sécurité réseau du personnel de OneTech Business Solutions et de ses stagiaires.

1.2.3 Description de l'existant

L'état actuel de notre système de laboratoires pour leur formation PCNSE et PCNSA repose sur une approche traditionnelle où chaque étudiant se voit attribuer un laboratoire individuel à configurer et à utiliser pour ses études. Cette méthode, bien qu'efficace dans une certaine mesure, présente plusieurs limitations. Premièrement, elle demande un investissement de temps considérable de la part des instructeurs pour configurer et maintenir chaque laboratoire. Deuxièmement, elle peut entraîner des incohérences dans les configurations et les expériences d'apprentissage, car chaque étudiant peut avoir des niveaux de compétence différents en matière de configuration réseau.

1.2.4 Problématique

La problématique majeur de notre projet réside dans la nécessité de fournir des environnements de laboratoire PCNSE et PCNSA préconfigurés et prêts à l'emploi pour les étudiants. Plutôt que de créer individuellement des laboratoires pour chaque étudiant, notre objectif est de centraliser les ressources en créant des labs bien configurés et accessibles à tous. Cette approche vise à optimiser l'utilisation des ressources tout en offrant une expérience d'apprentissage homogène et efficace pour tous les apprenants. Cependant, cela soulève des défis quant à la mise en place et à la maintenance de ces environnements, ainsi que la gestion des configurations pour répondre aux besoins spécifiques des étudiants.

1.2.5 Solution Proposée

- La solution proposée consiste à préconfigurer et mettre en place des laboratoires PCNSE et PCNSA, prêts à l'emploi, pour permettre aux étudiants d'accéder rapidement à des environnements d'apprentissage pratiques et fonctionnels.
- Centralisation des ressources en créant des laboratoires bien configurés et accessibles à tous les étudiants, ce qui permettra d'optimiser l'utilisation des ressources disponibles.
- Expérience d'apprentissage homogène et efficace pour tous les apprenants en fournissant des environnements de laboratoire standardisés et cohérents.
- Réduction du temps nécessaire à l'étude en fournissant des laboratoires préconfigurés, ce qui permettra aux étudiants de se concentrer sur l'apprentissage des concepts et des compétences sans avoir à passer du temps sur la configuration des équipements.
- Réponse aux besoins spécifiques des étudiants en matière d'apprentissage en proposant des environnements de laboratoire flexibles et adaptables à différents scénarios d'apprentissage.

Conclusion

Au cours de ce chapitre, nous avons présenté l'organisme d'accueil (OTBS), puis nous sommes passé au cadre général du projet en réalisant une étude de l'existant ainsi qu'un cahier de charge afin de proposer la solution adéquate.

CHAPITRE 2

ETAT DE ART

Introduction

Afin de bien mener notre projet, nous devons étudier certaines notions nécessaires pour réaliser notre travail. Nous consacrons donc ce chapitre pour la présentation des différents concepts et technologies.

2.1 Virtualisation

La virtualisation se présente comme une technologie fondamentale permettant de créer des environnements virtuels à partir de composants physiques tels que des serveurs, des systèmes d'exploitation et des ressources réseau. Cette abstraction physique des ressources informatiques offre de nombreux avantages pratiques. Par exemple, elle permet d'allouer efficacement des ressources matérielles aux différentes machines virtuelles utilisées dans les laboratoires de formation, en les extrayant de leurs équivalents physiques. Cette approche offre une flexibilité accrue dans la gestion des ressources, permettant ainsi d'optimiser leur utilisation et de réduire les coûts associés à l'acquisition et à la maintenance de matériel informatique. En somme, la virtualisation joue un rôle essentiel dans la création et la gestion des environnements de formation PCNSE et PCNSA, en offrant une infrastructure flexible et économique pour la mise en œuvre de ces laboratoires.

2.1.1 Virtualisation des serveurs

La virtualisation des serveurs est une technique qui permet de faire fonctionner et d'exécuter plusieurs serveurs virtuels simultanément sur un seul hôte physique, comme s'ils étaient installés sur des serveurs physiques distincts. La virtualisation des serveurs fonctionne grâce à un noyau système très léger appelé hyperviseur. L'hyperviseur joue le rôle d'arbitre entre les systèmes invités, attribuant du temps processeur et des ressources à chacun, et redirigeant les requêtes d'entrées/sorties vers les ressources physiques. En effet, il existe deux types d'hyperviseur.

2.1.2 Virtualisation des réseaux

La virtualisation de réseau désigne l'ensemble des outils utilisés pour créer une vue artificielle de l'environnement réseau de l'entreprise et consiste à reproduire un réseau physique et ses composants : ports, interrupteurs, pare-feu, équilibriseurs de charges, routeurs, etc. Ceci permet d'exécuter des applications sur un réseau virtuel comme sur un réseau physique. Cette technique permet d'utiliser moins de matériel, augmente la flexibilité et la portabilité des charges de travail.

2.1.3 Type d'hyperviseurs

L'un des principaux composants de la virtualisation est l'hyperviseur. Il permet à plusieurs systèmes d'exploitation de s'exécuter en même temps sur une même machine physique, il permet aussi d'allouer alternativement à chaque système d'exploitation les ressources nécessaires en termes de RAM

(Random Access Memory), CPU (Central Processing Unit) et Stockage, s'assurer que ces machines virtuelles n'interfèrent pas les unes avec les autres. Il existe principalement deux types d'hyperviseurs :

Hyperviseur de type 1

Un hyperviseur de type 1, également appelé hyperviseur natif, est un logiciel de virtualisation qui s'exécute directement sur la couche matérielle, sans nécessiter de système d'exploitation hôte supplémentaire. Il offre un accès direct aux ressources matérielles, telles que le processeur, la mémoire et les périphériques, pour chaque machine virtuelle.

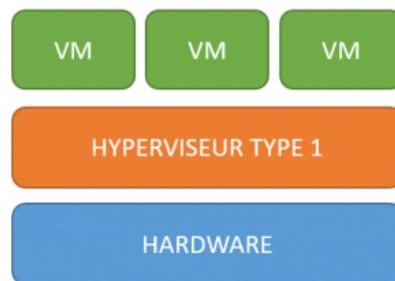


FIGURE 2.1 – Architecture hyperviseur de type 1 [8]

Les principaux produits d'hyperviseurs de type 1 :

Société	Produit
VMware	ESX/ESXi
Microsoft	Hyper-V
Red Hat	KVM
Oracle	Oracle VM
Open source	Xen

Hyperviseur de type 2

Un hyperviseur de type 2, également appelé hyperviseur hébergé, est un logiciel de virtualisation qui s'exécute sur un système d'exploitation hôte existant tel que Windows, macOS ou Linux. Il permet la création et la gestion de machines virtuelles, mais nécessite un système d'exploitation hôte pour fonctionner. Il offre une simplicité d'installation et d'utilisation, permettant l'exécution de plusieurs systèmes d'exploitation invités sur un même hôte. Cependant, il peut présenter une légère surcharge de performances par rapport à l'hyperviseur de type 1. Il est souvent utilisé pour des besoins de test ou de développement.

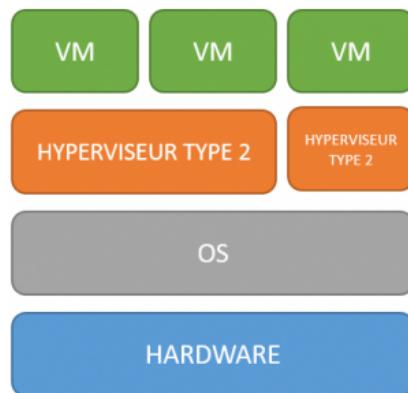


FIGURE 2.2 – Architecture hyperviseur de type 2 [8]

Les principaux produits d'hyperviseurs de type 2 :

Société	Produit
Oracle Corporation	Oracle VirtualBox
VMware	VMware Workstation
VMware	VMware Fusion
Open source	QEMU

2.1.4 Les Pare-feux

Un pare-feu/Firewall est un dispositif matériel ou logiciel conçu pour surveiller, filtrer et contrôler le trafic réseau, en fonction de règles prédéfinies. L'objectif principal du pare-feu est de protéger un réseau informatique ou un système informatique contre les menaces potentielles en régulant le flux de données entrant et sortant.

Le Firewalling

Le "firewalling" se réfère à la mise en place et à l'utilisation de pare-feu (firewalls) dans le domaine de la sécurité informatique. Les pare-feu peuvent être mis en œuvre à différents niveaux, notamment au niveau des périphériques réseau, des serveurs ou même des postes de travail individuels. Ils peuvent

fonctionner en inspectant les paquets de données et en autorisant ou en bloquant le trafic en fonction de règles de sécurité prédéfinies. Les pare-feu sont essentiels pour prévenir les intrusions, les attaques malveillantes, les logiciels malveillants et d'autres risques liés à la sécurité informatique.

Les différentes type de Pare-feux :

- **Politiques de sécurité** : Configurez les politiques de sécurité pour définir comment le pare-feu doit traiter le trafic entrant et sortant. Cela inclut les règles de filtrage basées sur des critères tels que les adresses IP source et destination, les ports, les protocoles, etc.
- **Filtrage d'application** : Si le pare-feu prend en charge le filtrage d'application, configurez les règles pour autoriser ou bloquer spécifiquement certaines applications ou services.
- **Prévention d'intrusion (IPS)** : Si le pare-feu intègre des fonctionnalités d'IPS, configurez les règles de détection et de prévention des intrusions pour identifier et bloquer les activités malveillantes.
- **Filtrage URL** : Configurez le filtrage d'URL pour contrôler l'accès à des sites web spécifiques ou à des catégories de contenu en fonction des politiques de l'organisation.
- **NAT (Network Address Translation)** : Configurez la translation d'adresse réseau pour permettre à plusieurs périphériques d'accéder à Internet via une seule adresse IP publique.
- **VPN (Virtual Private Network)** : Si le pare-feu prend en charge les VPN, configurez les tunnels VPN pour permettre des communications sécurisées entre des réseaux distants.
- **Logging et surveillance** : Configurez les paramètres de journalisation pour enregistrer les événements de sécurité et effectuez une surveillance régulière des journaux pour détecter des activités suspectes.
- **Gestion des utilisateurs** : Certains pare-feu permettent la gestion des utilisateurs, ce qui peut inclure l'authentification des utilisateurs et la définition de politiques basées sur les utilisateurs.
- **Haute disponibilité** : Pour assurer la continuité des opérations, configurez des options de haute disponibilité si elles sont prises en charge, telles que la redondance des pare-feu.
- **Mises à jour** : Assurez-vous de maintenir à jour le pare-feu avec les dernières signatures de sécurité, les mises à jour du système d'exploitation et les correctifs de sécurité.

2.1.5 Pare-feu de nouvelle génération (NGFW)

NGFW, ou "Next-Génération Firewall" (pare-feu de nouvelle génération), est une évolution des pare-feu traditionnels. Les NGFW intègrent des fonctionnalités avancées au-delà des capacités de filtrage de paquets et des règles de sécurité de base. Ces pares-feux de nouvelle génération combinent des fonctionnalités de sécurité réseau traditionnelles avec des technologies plus avancées, offrant une protection plus complète contre les menaces actuelles.

Dans ce projet on a choisi le NGFW de la société Palo Alto Networks .

2.1.6 Palo Alto Networks

Palo Alto Networks est un leader mondial en cybersécurité, dédié à rendre chaque jour plus sûr et plus sécurisé que le précédent. En tant que partenaire privilégié en cybersécurité, notre mission est de protéger notre mode de vie numérique. Nous nous distinguons par notre capacité à faciliter la transformation cybernétique grâce à nos plateformes innovantes, notre intelligence des menaces de premier ordre et nos services experts. Notre culture d'entreprise repose sur nos valeurs, faisant de Palo Alto Networks le choix numéro un pour les professionnels de la cybersécurité désireux de travailler dans un environnement dynamique et axé sur l'excellence.



FIGURE 2.3 – Logo Palo Alto Networks [9].

La relation entre PCNSE & PCNSA et Palo Alto Networks

La technologie de Palo Alto Networks est hautement intégrée et automatisée. Le portefeuille de produits de Palo Alto Networks comprend plusieurs technologies distinctes travaillant de concert pour prévenir les cyberattaques réussies. La certification Palo Alto Networks Certified Network Security Engineer (PCNSE) atteste que les ingénieurs peuvent déployer et configurer correctement les pare-feux de nouvelle génération de Palo Alto Networks tout en tirant parti du reste de la plateforme.

La certification PCNSE

La certification PCNSE couvre les aspects liés à la conception, au déploiement, à l'exploitation, à la gestion et à la résolution des problèmes des pare-feux de nouvelle génération de Palo Alto Networks.

La certification PCNSA

La certification PCNSA aborde les compétences relatives à l'exploitation et à la gestion des pare-feux de nouvelle génération de Palo Alto Networks.

Palo Alto NGFW

Palo Alto Networks est un fournisseur majeur de solutions de sécurité informatique, et son pare-feu de nouvelle génération (NGFW) est connu sous le nom de "Palo Alto Networks Next-Generation Firewall."

Les caractéristiques de Palo Alto NGFW

Approche basée sur les applications : L'un des aspects distinctifs de la solution Palo Alto Networks NGFW est son approche basée sur les applications. Au lieu de simplement contrôler le trafic en fonction des adresses IP et des ports, le pare-feu analyse le trafic à un niveau applicatif, permettant une visibilité granulaire sur les applications utilisées. Prévention des menaces avancée : Le NGFW de Palo Alto Networks intègre des fonctionnalités de prévention des menaces avancée, y compris la détection d'intrusion, la prévention des logiciels malveillants, la protection contre les exploits et d'autres mécanismes de défense contre les attaques. Contrôle d'accès et visibilité réseau : La solution offre un contrôle d'accès flexible basé sur l'utilisateur, l'application et le contenu. Elle fournit également une visibilité approfondie sur le trafic réseau, permettant aux administrateurs de surveiller et d'analyser les activités. Filtrage URL : Le filtrage d'URL permet de contrôler l'accès à Internet en fonction des catégories de sites Web, aidant ainsi à prévenir l'accès à des contenus potentiellement nuisibles. Intégration avec WildFire : Palo Alto Networks WildFire est un service de protection contre les menaces basé sur le cloud qui s'intègre avec le NGFW pour identifier et bloquer les menaces inconnues en utilisant l'analyse comportementale. VPN (Virtual Private Network) : La solution prend également en charge les connexions VPN, permettant aux utilisateurs distants de se connecter de manière sécurisée au réseau de l'organisation. Gestion centralisée : Palo Alto Networks propose une plateforme de gestion centralisée appelée Panorama, qui permet la gestion unifiée de plusieurs dispositifs NGFW à partir d'une seule interface. L'approche holistique de Palo Alto Networks, combinant la visibilité avancée sur les applications, la prévention des menaces, le contrôle d'accès et d'autres fonctionnalités, en fait un choix populaire pour les organisations cherchant à renforcer leur posture de sécurité informatique.

2.2 Central Management (Panorama)

Pour Palo Alto Networks, le terme *Central Management* est étroitement associé à la plateforme Panorama. Panorama est une solution de gestion centralisée développée par Palo Alto Networks, permettant aux administrateurs de gérer plusieurs pare-feu de nouvelle génération (NGFW) à partir d'une seule interface utilisateur. Les principales fonctionnalités de la gestion centralisée avec Panorama incluent :

- **Configuration unifiée** : Les administrateurs peuvent définir et appliquer des politiques de sécurité cohérentes sur tous les pare-feu gérés à partir d'une console unique.

- **Surveillance et reporting centralisés** : Panorama offre une visibilité centralisée sur le trafic réseau, les journaux d'événements et les rapports de sécurité, facilitant ainsi la détection des activités suspectes et la génération de rapports unifiés.
- **Mises à jour centralisées** : Les mises à jour du système d'exploitation, des signatures de menaces et d'autres logiciels peuvent être déployées de manière centralisée sur tous les pare-feu gérés.
- **Gestion des licences et des utilisateurs** : Les licences et les utilisateurs peuvent être gérés à partir d'une interface centrale, simplifiant ainsi le suivi des ressources et des droits d'accès.
- **Automatisation des tâches** : Les tâches de gestion répétitives peuvent être automatisées, ce qui permet de gagner du temps et de réduire les risques d'erreurs humaines.

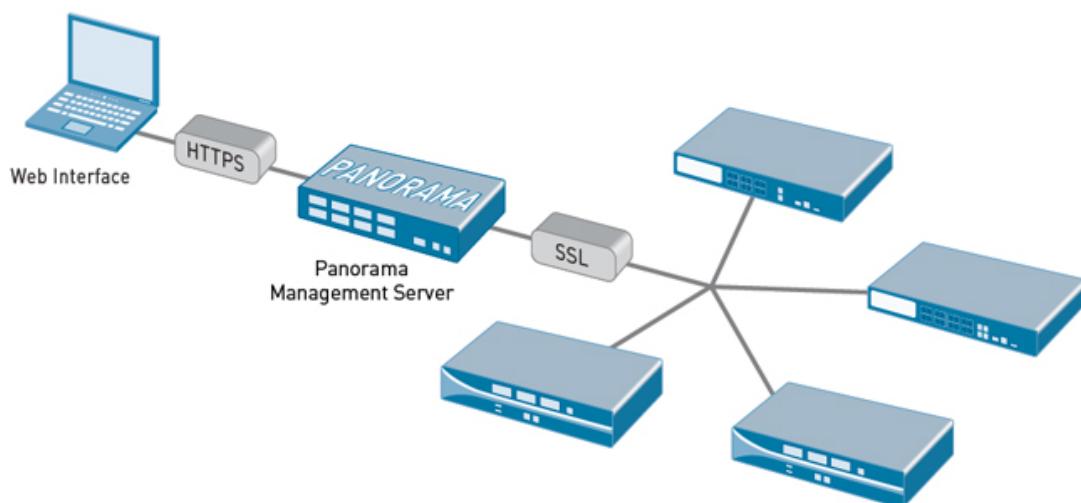


FIGURE 2.4 – La mise en place de Panorama [10].

2.3 La méthodologie DevOps

DevOps est une approche intégrée dans le domaine du développement logiciel et de l'informatique, visant à améliorer l'efficacité et la qualité du cycle de vie des systèmes. En réunissant les équipes de développement (Dev) et des opérations informatiques (Ops), DevOps facilite une collaboration étroite et harmonieuse. L'objectif principal est de rendre les processus de développement plus rapides et plus fiables.

Grâce à DevOps, les équipes de développement et opérationnelles travaillent main dans la main, ce qui améliore la communication et la coordination. Cela permet également d'automatiser de nombreuses tâches répétitives, réduisant ainsi les pertes de temps. Les pratiques clés incluent les tests continus, l'utilisation d'outils de gestion de configuration, le déploiement automatisé et la surveillance constante, tout cela pour garantir un avancement plus rapide des projets avec moins d'obstacles.

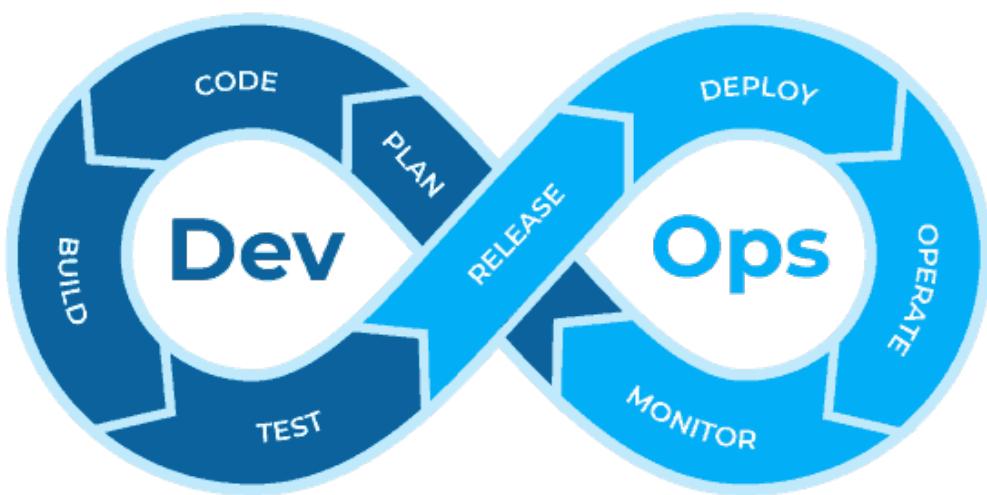


FIGURE 2.5 – Cycle de vie du mouvement DevOps [11].

2.3.1 Les phases de DevOps

Les phases de DevOps peuvent varier selon les méthodologies et les organisations, mais voici une description générale des phases courantes :

- **Planification** : Cette phase implique de définir les objectifs, les exigences et la feuille de route du projet. Les équipes collaborent pour déterminer les fonctionnalités à développer, établir les priorités, fixer les délais et allouer les ressources nécessaires.
- **Codage** : Durant cette phase, les développeurs écrivent le code source de l'application en utilisant les langages de programmation appropriés. Ils suivent des normes de codage, des bonnes pratiques et utilisent des systèmes de contrôle de version pour collaborer efficacement et gérer les modifications du code.

- **Build** : Après avoir écrit le code, la phase de build transforme le code source en un exécutable ou en une version prête pour le déploiement. Cela inclut la compilation du code source, la résolution des dépendances, la création d'artefacts binaires et leur empaquetage dans un format approprié.
- **Tests** : Les tests sont effectués pour vérifier la qualité du code et l'absence de bugs. Cela comprend les tests unitaires, les tests d'intégration et les tests fonctionnels. L'automatisation des tests est souvent utilisée pour accélérer le processus et assurer une couverture complète.
- **Release** : Dans cette phase, le code est préparé pour la mise en production. Les fonctionnalités et les corrections de bugs sont regroupées en versions cohérentes, prêtes à être déployées.
- **Déploiement** : Le déploiement consiste à mettre en production les versions du logiciel sur les serveurs cibles. Des techniques d'automatisation, telles que le déploiement continu (CD) et l'infrastructure en tant que code (IaC), sont utilisées pour rendre le processus plus efficace et fiable.
- **Exploitation** : Une fois le logiciel déployé, il est exploité en production. Les équipes de support et d'exploitation surveillent l'application, gèrent les incidents et assurent les performances. Des outils de surveillance et de journalisation sont utilisés pour détecter les problèmes et garantir la disponibilité du système.
- **Surveillance** : Cette phase consiste à collecter et analyser les données en temps réel sur les performances, les erreurs, les utilisateurs et les ressources du système. Cela permet de détecter les problèmes potentiels, d'optimiser les performances et d'améliorer la qualité du logiciel.

Les principes de DevOps

Les principes de DevOps sont souvent résumés par l'acronyme CALMS, qui représente les cinq aspects clés de cette approche. Voici une explication de chaque principe :

- **Culture (C)** : La création d'un environnement de travail collaboratif et le favoritisme de la communication ouverte, de la confiance et de la responsabilité entre les équipes de développement, d'exploitation et de sécurité.
- **Automatisation (A)** : L'automatisation consiste à utiliser des outils et des processus automatisés pour réduire les tâches manuelles, minimiser les erreurs et accélérer la mise en production.
- **Lean (L)** : Élimination des gaspillages et optimisation des processus, en identifiant les activités non essentielles et en améliorant le flux de travail.
- **Mesure (M)** : Mise en place de métriques et d'indicateurs pour mesurer les performances, la qualité, les délais de livraison, etc., afin d'identifier les améliorations potentielles et de prendre des décisions éclairées.
- **Partage (S)** : Le partage de connaissances, d'expériences et de bonnes pratiques entre les membres de l'équipe favorise l'apprentissage continu, l'amélioration collective et la résolution

des problèmes de manière collaborative.

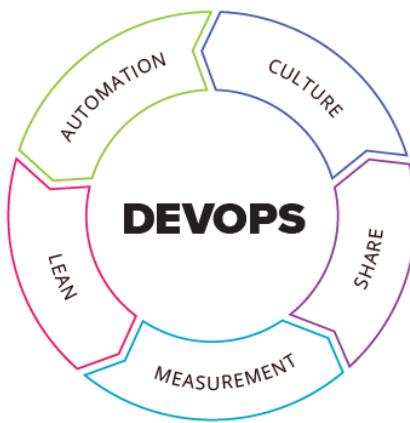


FIGURE 2.6 – Les principes de DevOps [12].

Les atouts de DevOps

Les avantages de DevOps sont nombreux et peuvent varier en fonction de l'organisation et de la mise en œuvre spécifique. Voici cinq avantages courants associés à DevOps :

- **Rapidité** : Augmentez le rythme des innovations pour vos clients, améliorez votre réactivité face aux évolutions du marché et optimisez votre efficacité et votre croissance. Le modèle DevOps permet à vos équipes de développement et d'opérations d'atteindre ces objectifs.
 - **Fiabilité** : Assurer la qualité des développements grâce au dialogue entre les développeurs et les administrateurs système.
 - **Évolution** : Faciliter l'évolution des projets en minimisant les risques.
 - **Collaboration améliorée** : Le partage d'information est primordial lors des développements pour assurer une qualité de code et une rapidité de livraison.
 - **Sécurité** : La collaboration des équipes permet de minimiser les risques en matière de sécurité.
- En général, l'adoption de DevOps offre des améliorations significatives en termes de rapidité, de qualité, de collaboration et de satisfaction des utilisateurs.

2.3.2 Les méthodologies DevOps

Infrastructure as a Code (IaC)

L'infrastructure en tant que code (IaC) est une approche de configuration informatique qui permet aux développeurs et aux opérateurs de gérer et de configurer automatiquement l'infrastructure informatique à l'aide de code, évitant ainsi les processus manuels. L'IaC peut être utilisée dans divers environnements informatiques, mais elle revêt une importance particulière dans le Cloud Computing, notamment pour l'infrastructure en tant que service (IaaS). De plus, dans le cadre de DevOps, qui

privilégie les flux de travail agiles et automatisés, une infrastructure informatique disponible en permanence est essentielle. Cette infrastructure joue un rôle crucial dans l'exécution et le test du code développé.

Approche CI/CD

L'approche CI/CD (Continuos Integration/Continuous Delivery) permet d'automatiser et d'accélérer le processus de développement et de déploiement des logiciels, garantissant ainsi des livraisons régulières et fiables des nouvelles fonctionnalités. Dans l'expression "CI/CD", le "CI" fait référence à l'"intégration continue", un processus automatisé pour les développeurs. Le "CD" désigne la "distribution continue" et/ou le "déploiement continu", des concepts très similaires, parfois utilisés de manière interchangeable.

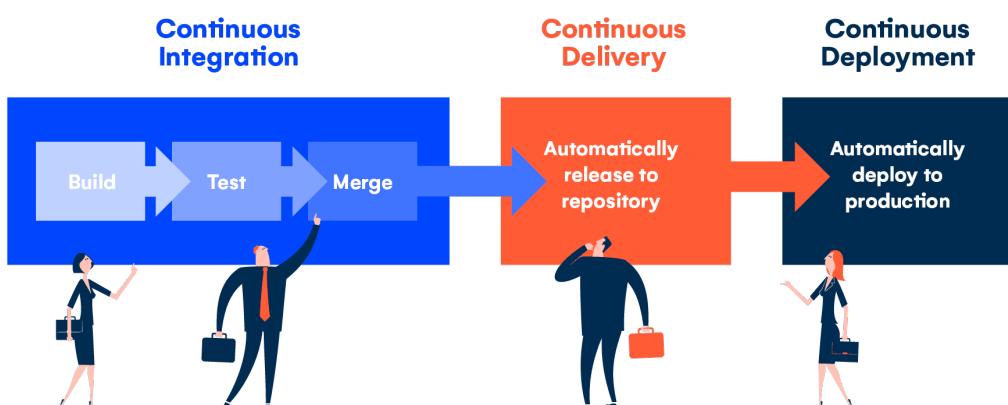


FIGURE 2.7 – L'approche CI/CD [13].

Intégration continue (CI)

L'intégration continue implique que les développeurs effectuent des modifications fréquentes sur le code de leur application, les testent ensuite, puis les fusionnent dans un référentiel partagé. Cette approche permet d'éviter de travailler sur plusieurs éléments de l'application simultanément, ce qui pourrait provoquer des conflits.

Distribution continue et déploiement continu (CD)

La distribution continue et le déploiement continu sont deux concepts concernant l'automatisation d'étapes plus avancées du pipeline, mais ils sont parfois dissociés pour illustrer le haut degré d'automatisation.

- **La distribution continue :** La distribution continue résout les problèmes de visibilité et de communication entre l'équipe de développement et l'équipe métier. Son objectif est de simplifier au maximum le déploiement de nouveau code.
- **Le déploiement continu :** Ce processus soulage les équipes d'exploitation surchargées par les tâches manuelles qui ralentissent la distribution des applications. Il s'appuie sur la distribution continue et automatise l'étape suivante du pipeline.

2.3.3 Principe de conteneurisation

La conteneurisation est un principe de virtualisation des applications qui vise à simplifier et accélérer leur développement et leur déploiement. Elle repose sur la création d'espaces utilisateurs isolés les uns des autres, tout en partageant un noyau commun. Le terme "conteneur" est utilisé pour désigner des instances spécifiques. De plus, la gestion des versions et des mises à jour est simplifiée, ce qui permet aux applications de s'adapter rapidement aux évolutions technologiques.

L'orchestration des conteneurs est un processus essentiel pour la gestion efficace des applications conteneurisées. Elle permet de coordonner et de contrôler automatiquement le déploiement, la mise à l'échelle, la réplication et la gestion des conteneurs dans un environnement distribué. Grâce à l'orchestration, les administrateurs système peuvent gérer efficacement les ressources, optimiser les performances et assurer une disponibilité continue des applications, tout en réduisant la complexité opérationnelle.

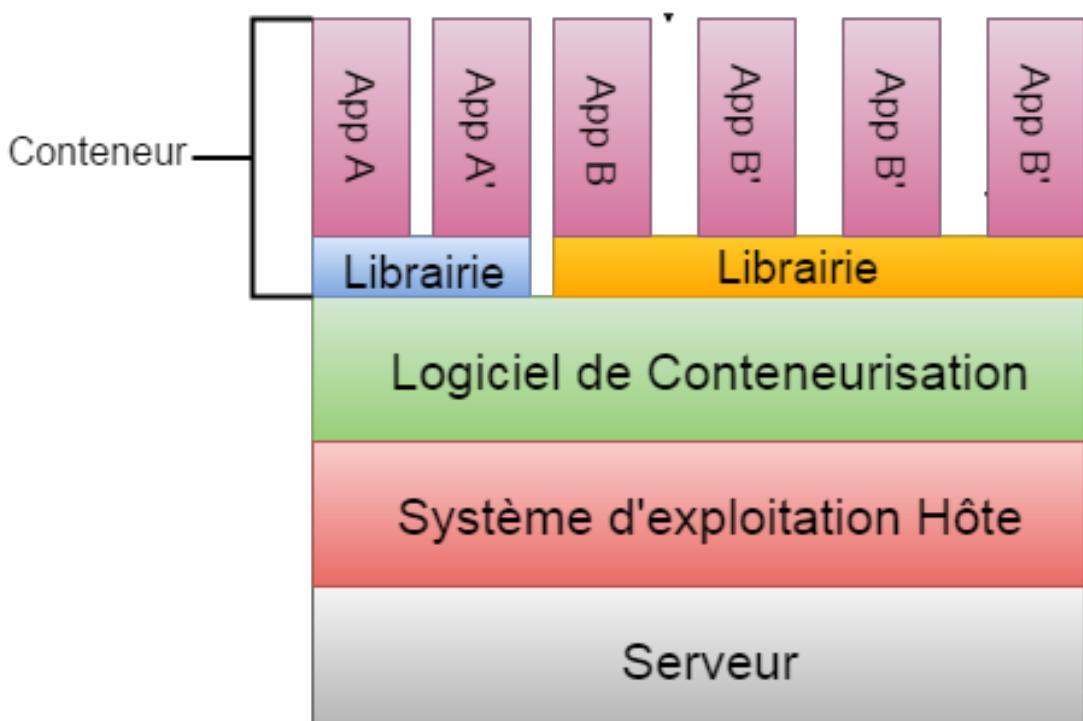


FIGURE 2.8 – Principe de Conteneurisation [14].

2.3.4 L'équilibrage de charge

Le load balancing (ou équilibrage de charge) est une technique utilisée dans les systèmes informatiques pour répartir la charge de travail entre plusieurs ressources (serveurs, machines virtuelles, etc.) afin d'optimiser les performances et garantir la disponibilité des services.

Les objectifs courants lors de la mise en place d'un équilibrEUR de charge sont les suivants :

- Garantir une répartition équitable de la charge de travail entre plusieurs serveurs. En effet, lorsqu'un grand nombre d'utilisateurs accèdent à un service en ligne, il peut y avoir une forte demande sur les serveurs sous-jacents. Le load balancing permet de répartir cette charge entre plusieurs serveurs, évitant ainsi la surcharge et garantissant des temps de réponse plus rapides pour les utilisateurs.
- Améliorer les performances. En distribuant la charge de manière équilibrée, le load balancing permet d'optimiser l'utilisation des ressources disponibles, évitant ainsi les points de congestion.
- Offrir une haute disponibilité. Le load balancing permet de créer un environnement redondant : si l'un des serveurs tombe en panne, le load balancer peut rediriger automatiquement le trafic vers d'autres serveurs disponibles, assurant ainsi la continuité du service sans interruption.
- Faciliter la scalabilité. Le load balancing facilite l'ajout de ressources au fur et à mesure de la croissance de la demande.
- Assurer la tolérance de panne. En cas de défaillance d'un serveur, le load balancing peut automatiquement rediriger le trafic vers d'autres serveurs fonctionnels, minimisant ainsi les interruptions de service et améliorant la résilience du système.

2.3.5 Les types de technologie d'équilibrage de charge

Les équilibrEURS de charge sont de deux types :

ÉquilibrEUR de charge matériel

Un équilibrEUR de charge matériel est un dispositif matériel qui intègre un système d'exploitation spécialisé et assure la répartition du trafic des applications Web au sein d'un cluster de serveurs d'applications.

Traditionnellement, les équilibrEURS de charge matériels et les serveurs d'applications sont déployés dans des centres de données locaux, et le nombre d'équilibrEURS de charge nécessaires dépend de la prévision du trafic de pointe anticipé. Pour assurer une continuité du service, les équilibrEURS de charge sont généralement déployés en paires, prêts à prendre le relais en cas de défaillance de l'un d'entre eux.

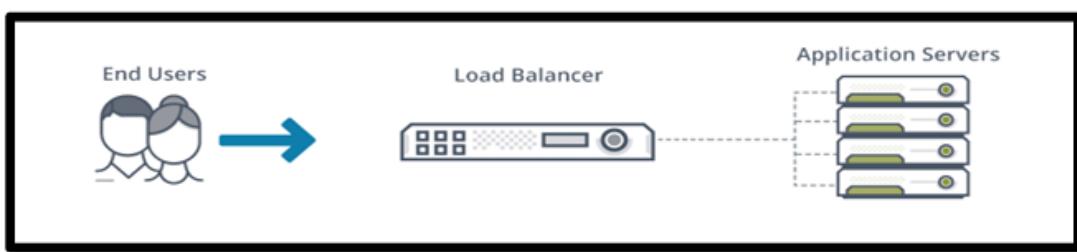


FIGURE 2.9 – Architecture d'un load balancer matériel [2].

Équilibrleur de charge logiciel

Un load balancer logiciel est une application logicielle qui répartit la charge entre les serveurs. Il peut être exécuté sur des serveurs standards, des machines virtuelles ou des conteneurs.

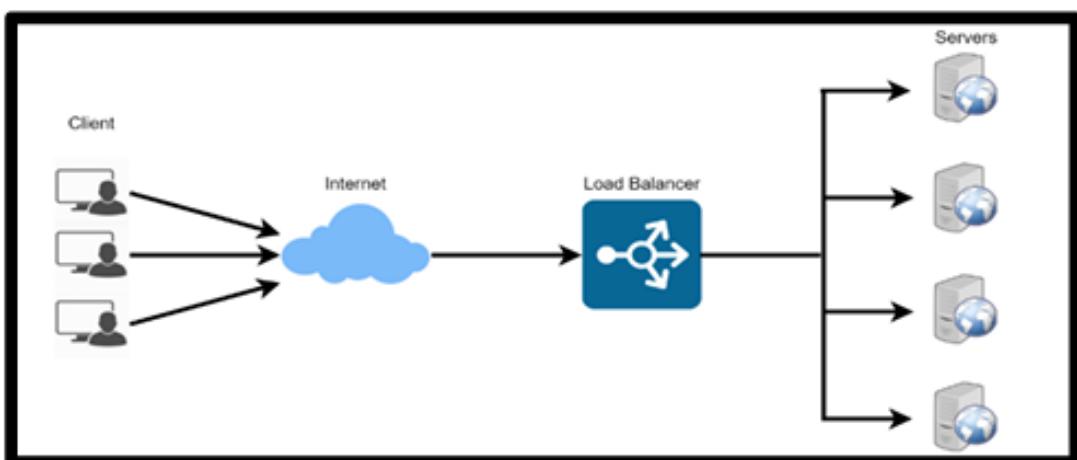


FIGURE 2.10 – Architecture d'un load balancer logiciel [15].

2.3.6 Outils de gestion de code source

Les outils de gestion de code source (SCM) sont des logiciels utilisés pour gérer les différentes versions du code source d'un projet informatique. Ils facilitent la collaboration entre les développeurs en permettant le suivi des modifications, la gestion des branches de développement, la fusion des changements et le contrôle des versions du code.

Quelques exemples populaires d'outils de gestion de code source incluent GitLab, GitHub et Bitbucket. Dans le tableau ci-dessous, nous avons présenté les différences entre ces trois outils.

Caractéristique	GitLab	GitHub	Bitbucket
Répertoires de travail	Privés payants	Privés gratuits	Privés gratuits
Hébergement	Pas d'hébergement	Gratuit sur un serveur privé	Gratuit possible sur un serveur privé
Intégration continue	Uniquement avec des outils tiers	Incluse gratuitement	Incluse gratuitement
Déploiement	Aucune plateforme de déploiement intégrée	Déploiement logiciel avec Kubernetes	Déploiement logiciel avec Kubernetes
Tableau de bord	Personnel pour suivre les problèmes et les demandes d'extraction	Analytique pour planifier et suivre le projet	Analytique pour planifier et suivre le projet

TABLE 2.1 – Tableau comparatif des fonctionnalités de GitHub, GitLab et Bitbucket

Nous avons choisi GitHub plutôt que GitLab et Bitbucket en raison de sa large adoption par la communauté des développeurs et de ses fonctionnalités avancées spécifiquement adaptées aux projets de développement logiciel. GitHub offre une plateforme bien établie pour la gestion des dépôts de code, le suivi des problèmes, la collaboration entre les équipes, et l'intégration continue, entre autres. En optant pour GitHub, nous bénéficions d'une infrastructure familière et largement utilisée, ce qui facilite la collaboration avec d'autres développeurs, la contribution à des projets open source et l'accès à une vaste communauté de développeurs et de ressources.

De plus, GitHub propose des fonctionnalités robustes et une intégration transparente avec un large éventail d'outils de développement et de déploiement, ce qui nous permet d'optimiser notre flux de travail de développement.



FIGURE 2.11 – LOGO GitHub [3] .

2.3.7 Environnement de travail

Voici le tableau des caractéristiques des comparaisons entre GNS3 et EVE-NG :

Caractéristiques	GNS3	EVE-NG
Type d'installation	Peut être installé localement, sur une VM locale ou sur un serveur distant	Sans client, installé en tant que VM
Licence	Open source, gratuit	Version gratuite avec certaines limitations, version pro moyennant des frais supplémentaires
Images logicielles	Utilise des fichiers IOS .bin ou des images Cisco VIRL	Utilise des fichiers IOS .bin ou des images Cisco VIRL
Systèmes d'exploitation pris en charge	Windows, Linux, Mac, ESXi	VMware, machine virtuelle Google Cloud, AMD Ryzen 3900
Architecture	Nécessite le téléchargement et l'installation d'une application indépendante	Emulateur virtuel autonome sans client
Simulation ou émulation ?	Émulation de la plupart des appareils, simulation des commutateurs	Émulation
Idéal pour...	Étude des certifications CCNA, CCNP et CCIE et utilisé comme banc d'essai pour les réseaux de production	Étude des certifications PCNSE, PCNSA et CCNA et utilisé comme banc d'essai pour les réseaux de production

TABLE 2.2 – Tableau comparatif de GNS3 et EVE-NG

EVE-NG est le choix idéal pour notre projet en raison de sa prise en charge des certifications PCNSE et PCNSA. Grâce à sa capacité d'émulation avancée et à sa compatibilité avec les images IOS et VIRL de Cisco, EVE-NG offre un environnement parfait pour simuler des configurations Palo Alto Networks, ce qui est essentiel pour préparer efficacement les certifications PCNSE et PCNSA. Sa flexibilité, sa convivialité et ses fonctionnalités avancées en font l'outil privilégié pour nos besoins de formation et de test dans le domaine des pare-feu Palo Alto Networks.



FIGURE 2.12 – LOGO emulateur EVE-NG [4] .

2.3.8 Outil d'infrastructure en tant que code (IaC)

L'infrastructure en tant que code (IaC) simplifie la gestion des infrastructures informatiques en les décrivant à l'aide de fichiers de configuration ou de scripts. Les outils de IaC automatisent le déploiement et la configuration des ressources dans le Cloud. Cette approche réduit les erreurs humaines, accélère le déploiement des applications et rend l'infrastructure plus flexible et évolutive.

Voici le tableau comparatif des outils d'infrastructure en tant que code :

Caractéristique	Terraform	AWS CloudFormation	Google Cloud Deployment Manager
Langage	HCL (HashiCorp Configuration Language)	JSON or YAML	YAML
Infrastructure en tant que code	Oui	Oui	Oui
Multi-cloud	Oui	Non (lié à AWS)	Non (lié à Google Cloud)
Gestion des états	Oui (par défaut)	Oui	Oui (automatique)
Contrôle des versions	Oui	Oui	Oui
Gestion des ressources	Multi-fournisseur	AWS uniquement	Google uniquement
Flexibilité	Haute	Limitée	Limitée
Intégration avec d'autres outils	Bonne	Excellente	Bonne

TABLE 2.3 – Tableau comparatif des fonctionnalités de Terraform, AWS CloudFormation et Google Cloud Deployment Manager

Nous avons choisi Terraform pour notre projet en raison de sa polyvalence, de sa robustesse et de sa large adoption dans l'industrie. Terraform offre une approche déclarative pour la gestion de l'infrastructure, ce qui nous permet de décrire notre infrastructure cible dans un code simple et lisible. De plus, sa prise en charge multi-cloud nous donne la flexibilité d'utiliser divers fournisseurs de services cloud sans être verrouillés dans une plateforme spécifique. Avec sa communauté active et ses mises à jour régulières, Terraform garantit que nous restons à jour avec les dernières pratiques et fonctionnalités en matière d'infrastructure en tant que code.



FIGURE 2.13 – LOGO Terraform [5].

2.3.9 Automatisation

L’automatisation est un processus essentiel dans le domaine de l’informatique, permettant de simplifier et d’accélérer les tâches répétitives et laborieuses. En utilisant des outils et des scripts, l’automatisation permet d’exécuter des actions de manière programmée et cohérente, réduisant ainsi les erreurs humaines et améliorant l’efficacité opérationnelle. En automatisant les processus de déploiement, de configuration, de gestion et de surveillance des infrastructures et des applications, les équipes informatiques peuvent consacrer moins de temps à des tâches manuelles et plus de temps à des activités à valeur ajoutée, telles que l’innovation et le développement de nouvelles fonctionnalités. En résumé, l’automatisation contribue à accroître la productivité, à améliorer la qualité et à réduire les coûts dans les environnements informatiques modernes.

Voici le tableau comparatif des outils d’automatisation :

Caractéristiques	Ansible	Puppet	SaltStack
Langage	YAML	DSL (Domain Specific Language)	YAML
Architecture	Agentless	Agent-based	-
Installation	Agent-based	Simple	Complex
Configuration	Moderate	Simple	Complex
Learning Curve	Moderate	Low	High
Community Support	Large	Moderate	Large
Scalability	Moderate	Highly scalable	Scalable
Use Cases	Configuration Management, Orchestration, Automation	Configuration Management, Orchestration, Automation	Configuration Management, Orchestration, Automation
Platform Support	Cross-Platform	Cross-Platform	Cross-Platform

TABLE 2.4 – Tableau comparatif des fonctionnalités de Ansible, Puppet et SaltStack

Dans notre projet, nous avons opté pour Ansible en raison de sa simplicité d'utilisation, de son installation aisée et de sa facilité de configuration et de maîtrise. En moins de 30 minutes, il est possible d'installer et de configurer le système, et d'exécuter des commandes sur les serveurs pour résoudre divers problèmes tels que l'ajustement de l'heure d'été, la synchronisation de l'heure, la modification du mot de passe root, la mise à jour des serveurs, le redémarrage des services, et bien d'autres encore.



FIGURE 2.14 – LOGO ansible [6] .

2.3.10 Comparaison des Outils CI/CD

Pour vous aider à choisir l'outil CI/CD le mieux adapté à vos besoins, voici un tableau comparatif des fonctionnalités et caractéristiques de Jenkins, GitLab CI/CD et CircleCI :

Caractéristique	Jenkins	GitLab CI/CD	CircleCI
Intégration	Indépendant, intégration via plugins	Intégré à GitLab	Intégré avec GitHub, Bitbucket
Configuration	Fichiers de configuration XML ou GUI	Fichier <code>.gitlab-ci.yml</code>	Fichier <code>config.yml</code>
Evolutivité	Très évolutif avec plugins	Très évolutif avec pipelines Docker/Kubernetes	Très évolutif avec jobs parallèles
Facilité d'utilisation	Courbe d'apprentissage élevée	Intégré, facile si vous utilisez GitLab	Interface intuitive, facile à configurer
Support de conteneurs	Oui, via plugins	Oui, support natif Docker et Kubernetes	Oui, Docker et Orbs pour réutilisabilité
Gestion des secrets	Via plugins	Intégré avec GitLab Secrets	Intégré avec CircleCI Contexts
Intégrations	Très nombreuses via plugins	Intégration GitLab et autres via API	NOMBREUSES intégrations via Orbs
Hébergement	Auto-hébergé	Hébergement cloud et auto-hébergé	Principalement hébergement cloud
Analyse et rapports	Via plugins	Intégré avec des tableaux de bord	Analyses détaillées et rapports intégrés
Sécurité	Variable, dépend des plugins	Fort, gestion des utilisateurs et secrets	Bon, avec gestion des permissions
Communauté et support	Grande communauté, nombreux plugins	Communauté GitLab active, support intégré	Communauté active, support réactif

TABLE 2.5 – Tableau comparatif des fonctionnalités de Jenkins, GitLab CI/CD et CircleCI

Dans notre projet, Jenkins bénéficie d'une communauté active et dynamique qui contribue constamment à son amélioration et à l'ajout de nouvelles fonctionnalités. Avec une documentation complète et de nombreux forums de support, les utilisateurs peuvent facilement trouver des solutions à leurs problèmes et partager leurs expériences. De plus, la grande popularité de Jenkins signifie qu'il existe une multitude de ressources éducatives, comme des tutoriels et des guides pratiques, pour aider les nouveaux utilisateurs à se familiariser rapidement avec l'outil.



FIGURE 2.15 – LOGO jenkins [7] .

Conclusion

Après avoir expliqué les notions théoriques dans ce chapitre, le chapitre suivant sera consacré pour présenter notre solution.

CHAPITRE 3

IMPLÉMENTATION DE LA SOLUTIONN CHOISIE

Introduction

Dans ce chapitre, nous nous intéressons à l'étude technique de la solution choisie précédemment. Pour cela nous allons commencer, tout d'abord, par la présentation de notre architecture, puis nous présentons l'environnement matériel et logiciel utilisé pour développer notre application.

3.1 Architecture reseau

Cette architecture réseau représente une infrastructure segmentée en plusieurs VLANs (Virtual Local Area Networks) pour assurer une gestion efficace et sécurisée du trafic réseau. Voici une description détaillée de chaque composant et de leur rôle dans cette architecture :

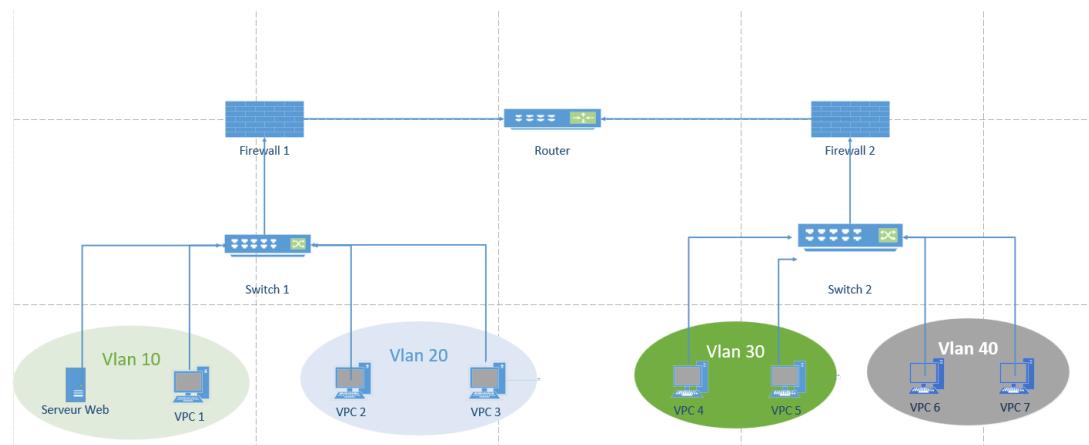


FIGURE 3.1 – Architecture proposée .

- **Routeur :**
 - Le routeur situé au centre de l'architecture permet la communication entre les différents VLANs. Il gère le routage du trafic réseau entre les sous-réseaux distincts.
- **Pare-feu 1 et Pare-feu 2 :**
 - Les deux pare-feux (Firewall 1 et Firewall 2) assurent la sécurité du réseau en filtrant le trafic entrant et sortant. Ils protègent les différents segments du réseau contre les accès non autorisés et les attaques.
- **Switch 1 et Switch 2 :**
 - Les deux commutateurs (Switch 1 et Switch 2) sont responsables de la connexion des différents dispositifs au réseau et de la segmentation en VLANs. Ils assurent une gestion efficace du trafic au sein de chaque VLAN.
- **VLAN 10 :**
 - Contient un serveur web et un PC (VPC 1). Ce VLAN est probablement dédié aux services web et applications accessibles depuis l'extérieur ou par les utilisateurs internes.
- **VLAN 20 :**

- Comprend deux PC (VPC 2 et VPC 3). Ce VLAN pourrait être utilisé pour des utilisateurs ou des départements spécifiques nécessitant un isolement pour des raisons de sécurité ou de gestion du trafic.
- **VLAN 30 :**
 - Inclut deux PC (VPC 4 et VPC 5). Similaire au VLAN 20, ce segment est dédié à un autre groupe d'utilisateurs ou à des services spécifiques.
- **VLAN 40 :**
 - Contient deux PC (VPC 6 et VPC 7). Comme les autres VLANs, il isole le trafic de ce groupe d'utilisateurs pour des raisons de sécurité et d'efficacité.

3.2 Environnement de travail

3.2.1 Environnement matériel

Caractéristiques de l'ordinateur	
Nom	Dell
Écran	1920 * 1080, 120.04 Hz
Processeur	12th Gen Intel(R) Core(TM) i7-1255U 1.70 GHz
Mémoire RAM	24 Go
Stockage	512 Go SSD
Système d'exploitation	Windows 11 Professionnel x64

TABLE 3.1 – Caractéristiques de l'ordinateur DELL

3.2.2 Environnement logiciel

Pour réaliser notre travail, nous avons choisi les logiciels suivants :

- **EVE-NG community** : est un émulateur graphique de réseau qui fournit une interface utilisateur via un navigateur et qui permet de concevoir des topologies de réseau complexes. Il permet aussi d'ajouter des nœuds de réseau à partir d'une bibliothèque de modèles, les connecter ensemble et les configurer ;
- **Putty** : est un client SSH et telnet qui permet d'exécuter les commandes à distance
- **VMware Workstation** : est un outil de virtualisation de poste de travail qui permet d'installer et exécuter des machines virtuelles au sein d'un même système d'exploitation en utilisant les ressources du système hôte et en attribuant à volonté au système virtuel de la mémoire vive, des disques durs, des cartes réseaux, etc. ;
- **Plate-forme d'automatisation informatique Ansible Tower** : est une solution Web qui rend Ansible encore plus facile à utiliser pour les équipements informatiques. Il est conçu pour être la plaque tournante de toutes nos tâches d'automatisation.

- **Terraform** : est un outil d’infrastructure en tant que code (IaC) qui permet de définir et de provisionner des datacenters via un langage de configuration déclaratif;
- **GitHub** : est une plateforme de développement collaboratif et de gestion de versions de code source qui permet aux développeurs de stocker, de suivre et de gérer les modifications de leur code source.
- **VSCode** : Visual Studio Code est un éditeur de code source gratuit développé par Microsoft, connu pour sa légèreté et ses nombreuses extensions qui permettent d’ajouter des fonctionnalités de développement avancées et de personnaliser l’interface utilisateur.
- **Jenkins** : est un serveur d’intégration continue open source qui permet d’automatiser les tâches de construction, de test et de déploiement de logiciels, facilitant ainsi le processus de développement et la livraison continue.

3.2.3 Configuration des serveurs

Dans notre architecture, nous avons implémenté un serveur Web .

Configuration du serveur Web

Au niveau du serveur Web, nous avons indiqué l’adresse IP :"192.168.10.11" d’une manière automatique (DHCP) puis nous téléchargé et install le package BusyBox HTTP daemon avec la commande "tce-load -wi busybox-httpd.tc". Cela permet de configurer un serveur web sur Tiny Core Linux.

```

Terminal Example usage:
Load local extension:
  tce-load -i /mnt/hda1/tce/optional/nano.tc
Download into tce/optional directory, updates OnBoot and installs:
  tce-load -w -i nano.tc
Download only into tce/optional directory:
  tce-load -w nano.tc
gns3@box:~$ tce-load -wi busybox-httpd.tc
busybox-httpd is already installed!
gns3@box:~$ ifconfig
eth0      Link encap:Ethernet HWaddr 00:50:00:00:10:00
          inet addr:192.168.10.11  Bcast:192.168.10.255  Mask:255.255.255.0
          inet6 addr: fe80::250:ff:fe00:1000/64 Scope:Link
          UP BROADCAST RUNNING MULTICAST  MTU:1500  Metric:1
          RX packets:13 errors:0 dropped:0 overruns:0 frame:0
          TX packets:11 errors:0 dropped:0 overruns:0 carrier:0
          collisions:0 txqueuelen:1000
          RX bytes:2213 (2.1 KiB)  TX bytes:2166 (2.1 KiB)

```

FIGURE 3.2 – Configuration du ssh .

3.2.4 Configuration du routeur

La configuration du routeur est une étape cruciale dans la mise en place d'un réseau informatique. En attribuant des adresses IP à chaque interface, telles que f0/0, f1/0, f2/0. Comme illustré dans la figure , la commande "ip address" est utilisée pour assigner à chaque interface une adresse IP en spécifiant le masque de sous-réseau.

```
interface FastEthernet0/0
  ip address 20.20.1.1 255.255.255.0
  duplex full
!
interface FastEthernet1/0
  ip address 20.20.2.1 255.255.255.0
  duplex full
!
interface FastEthernet2/0
  ip address 20.20.3.1 255.255.255.0
  duplex full
!
interface Ethernet3/0
  no ip address
  shutdown
  duplex full
!
interface Ethernet3/1
  no ip address
  shutdown
  duplex full
!
interface Ethernet3/2
  no ip address
  shutdown
  duplex full
!
interface Ethernet3/3
  no ip address
  shutdown
  duplex full
!
interface FastEthernet4/0
  ip address 192.168.1.200 255.255.255.0
  duplex full
!
```

FIGURE 3.3 – Configuration de l'adresse IP sur une interface .

pour le configuration du ospf en a utilisé la commande "router ospf" pour démarrer la configuration du protocole de routage (OSPF) avec l'ID de processus 1 La figure 3.4 représente la configuration pour ospf

```
router ospf 1
  network 0.0.0.0 255.255.255.255 area 0
!
```

FIGURE 3.4 – Configuration du ospf .

La figure 3.5 représente la configuration SSH d'un routeur Cisco est illustrée. Le commandes montrent l'activation du service SSH, la définition des versions, et la configuration des utilisateurs et des mots passe chiffré pour sécuriser l'accès distant

```
username ahmed privilege 15 password 0 ahmed
username ahmed123 privilege 15 secret 5 $1$aX05$XZxfwAullzOFN26kDg6OE.
username ben privilege 15 secret 5 $1$kh06$ygRSleOp/cVc0R9DVC5aU1
username hdhily secret 5 $1$Ute3$TzUwU420YStlh1j6t9iu6.
username mansour secret 5 $1$UN6P$nt5ElOTBG38rPKnY3CM380
!
!
!
!
!
ip ssh version 2
!
```

FIGURE 3.5 – Configuration du ssh .

3.2.5 Configuration des commutateur

Au niveau du commutateur, nous avons créé les VLANs et configuré SSH pour sécuriser l'accès : La figure 3.6 et figure 3.7 représente la configuration pour les deux vlans 10 et 20 et configuration de l'interface vlan1

```
interface Ethernet0/0
switchport trunk encapsulation dot1q
switchport mode trunk
duplex auto
|
interface Ethernet0/1
switchport access vlan 10
switchport mode access
duplex auto
|
interface Ethernet0/2
switchport access vlan 10
switchport mode access
duplex auto
|
interface Ethernet0/3
switchport access vlan 20
switchport mode access
duplex auto
|
interface Ethernet1/0
switchport access vlan 20
switchport mode access
duplex auto
|
interface Ethernet1/1
duplex auto
|
interface Ethernet1/2
duplex auto
|
interface Ethernet1/3
duplex auto
|
interface Vlan1
ip address 192.168.1.201 255.255.255.0
|
```

FIGURE 3.6 – Configuration du vlan pour commutateur 1 .

VLAN	Name	Status	Ports
1	default	active	Et1/1, Et1/2, Et1/3
30	gray	active	Et0/1, Et0/2
40	yellow	active	Et0/3, Et1/0
1002	fdmi-default	act/unsup	
1003	token-ring-default	act/unsup	
1004	fdninet-default	act/unsup	
1005	trnet-default	act/unsup	

FIGURE 3.7 – Configuration du vlan pour commutateur 2 .

La figure 3.8 et figure 3.9 représente la configuration SSH par l'interface `vlan1` des deux commutateur Cisco est illustrée. Le commandes montrent l'activation du service SSH, la définition des versions, et la configuration des utilisateurs et des mots passe chiffré pour sécuriser l'accès distant.

```
username ahmed123 privilege 15 secret 4 1L3fWDTaoudA1koR3kM2RqyRBYUPcxQs2vpCSMKMF.
no aaa new-model
!
ip cef
!
!
ip domain-name cisco1.com
no ipv6 cef
ipv6 multicast rpf use-bgp
!
!
!
!
!
!
!
!
spanning-tree mode pvst
spanning-tree extend system-id
!
!
!
!
vlan internal allocation policy ascending
!
ip ssh version 2
```

FIGURE 3.8 – Configuration du ssh pour commutateur 1 .

VLAN	Name	Status	Ports
1	default	active	Et1/1, Et1/2, Et1/3
30	gray	active	Et0/1, Et0/2
40	yellow	active	Et0/3, Et1/0
1002	fddi-default	act/unsup	
1003	token-ring-default	act/unsup	
1004	fddinet-default	act/unsup	
1005	trnet-default	act/unsup	

FIGURE 3.9 – Configuration du ssh pour commutateur 2 .

3.2.6 Configuration du pare-feu PALOALTO

Un pare-feu est un système permettant de protéger un réseau

Configuration les interfaces management

Après l'installation des images du pare-feu Paloalto, nous étions en accès local en utilisant l'accès par défaut attribué par le constructeur, à savoir le login « admin » et le mot de passe « admin », afin d'attribuer les adresses IP "192.168.1.130" et "192.168.1.132".

```
admin@PA-VM> configure
Entering configuration mode
[edit]
admin@PA-VM# set deviceconfig system type static
[edit]
admin@PA-VM# set deviceconfig system ip-address 192.168.1.130 netmask 255.255.255.0 default-gateway 192.168.1.1
[edit]
admin@PA-VM# commit force

Commit job 2 is in progress. Use Ctrl+C to return to command prompt
.....55%..70%98%....[ 1298.513693] netlink: 12 bytes leftover after parsing attributes in process `ip'.
.....100%
Configuration committed successfully
```

FIGURE 3.10 – Configuration initiale d'une interface de Paloalto .

À ce stade, nous avons pu se connecter à les interfaces Web de ces pare-feu via les adresses : https ://192.168.1.130 et https ://192.168.1.132

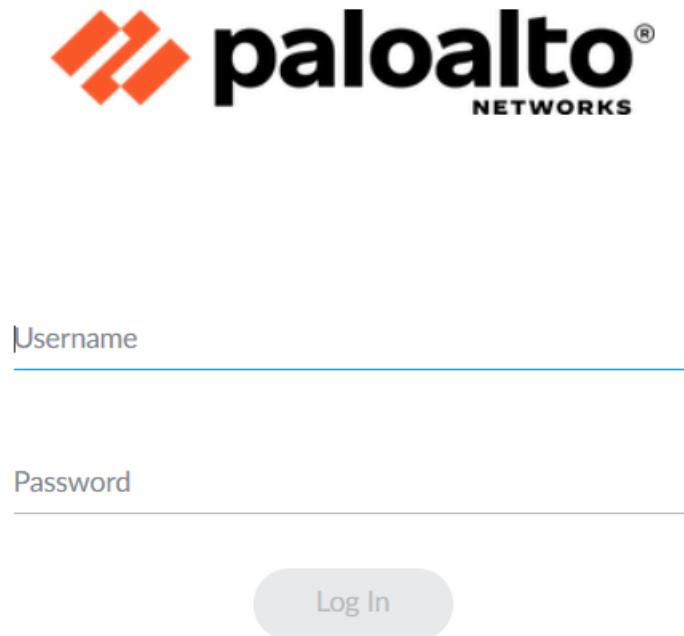


FIGURE 3.11 – Connexion à l'interface graphique de Paloalto .

Configuration les interfaces

Après la configuration des interfaces management , le système du réseau maintiennent des tables de routage pour transférer les paquets. Pour cela nous avons configuré les différentes routes dans les deux pare-feu paloalto

Ethernet VLAN Loopback Tunnel SD-WAN											
INTERFACE	INTERFACE TYPE	MANAGEMENT PROFILE	LINK STATE	IP ADDRESS	VIRTUAL ROUTER	TAG	VLAN / VIRTUAL-WIRE	SECURITY ZONE	SD-WAN INTERFACE PROFILE	UPSTREAM NAT	FEATURES
ethernet1/1	Layer3		none	none	Untagged	none	none	none		Disabled	
ethernet1/1.10	Layer3	ping-trust	green	192.168.10.1/24	default	10	none	trust		Disabled	SSL
ethernet1/1.20	Layer3		green	192.168.20.1/24	default	20	none	trust		Disabled	SSL
ethernet1/2	Layer3	ping-untrust	green	20.20.2.2/24	default	Untagged	none	untrust		Disabled	SSL

FIGURE 3.12 – Configuration les interfaces et les subinterface en Paloalto .

Pour le tunnel entre les deux pare-feu on à crée aussi l'interface tunnel

Tunnel Interface

Interface Name	tunnel	MTU	100				
Comment	Configured for internal traffic						
Netflow Profile	None						
Config	IPv4	IPv6	Advanced				
Assign Interface To <table border="1" style="width: 100%; border-collapse: collapse;"> <tr> <td>Virtual Router</td> <td>default</td> </tr> <tr> <td>Security Zone</td> <td>VPN</td> </tr> </table>				Virtual Router	default	Security Zone	VPN
Virtual Router	default						
Security Zone	VPN						
<input type="button" value="OK"/> <input type="button" value="Cancel"/>							

FIGURE 3.13 – Configuration l'interface tunnel en Paloalto .

Configuration des zone du sécurité

Apres on a la configuration des 3 zone du sécurité "trust" , "untrust" et "VPN"

NAME	TYPE	INTERFACES / VIRTUAL SYSTEMS	ZONE PROTECTION PROFILE	PACKET BUFFER PROTECTION	LOG SETTING	User-ID		Device-ID			
						ENABLED	INCLUDED NETWORKS	EXCLUDED NETWORKS	ENABLED	INCLUDED NETWORKS	EXCLUDED NETWORKS
trust	layer3	ethernet1/1.10 ethernet1/1.20		<input type="checkbox"/>		<input type="checkbox"/>	any	none	<input type="checkbox"/>	any	none
untrust	layer3	ethernet1/2		<input type="checkbox"/>		<input type="checkbox"/>	any	none	<input type="checkbox"/>	any	none
VPN	layer3	tunnel.100		<input type="checkbox"/>		<input type="checkbox"/>	any	none	<input type="checkbox"/>	any	none

FIGURE 3.14 – Configuration des zones du sécurité en Paloalto .

Configuration les virtuelles routeurs

Une adresse IP virtuelle routeur est utilisée pour mapper une adresse IP

The screenshot shows the 'Virtual Router - Static Route - IPv4' configuration dialog. The 'Name' field is set to 'route'. The 'Destination' field is '0.0.0.0/0'. The 'Interface' is 'None'. The 'Next Hop' is 'IP Address' with value '20.20.2.1'. The 'Admin Distance' is '10 - 240'. The 'Metric' is '10'. The 'Route Table' is 'Unicast'. The 'BFD Profile' is 'Disable BFD'. Below this, there is a 'Path Monitoring' section with a checkbox. Under 'Failure Condition', 'Any' is selected. The 'Preemptive Hold Time (min)' is set to '2'. A table for monitoring paths is shown, with columns: NAME, ENABLE, SOURCE IP, DESTINATION IP, PING INTERVAL(SEC), and PING COUNT. At the bottom are 'Add' and 'Delete' buttons, and 'OK' and 'Cancel' buttons.

FIGURE 3.15 – Configuration de virtuelle routeur en Paloalto .

Ensuite , dans la figure 3.16 on a la configuration de virtuelle routeur pour le VPN

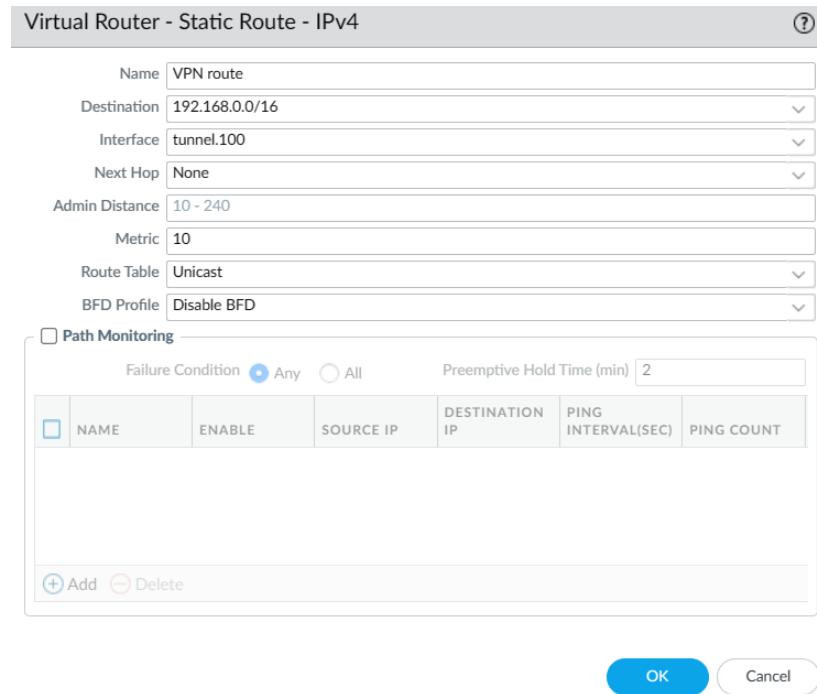


FIGURE 3.16 – Configuration de virtuelle routeur pour le VPN .

Configuration du protocole l'ospf

Ceci est la création du protocole OSPF, de l'identifiant du routeur et de la zone :

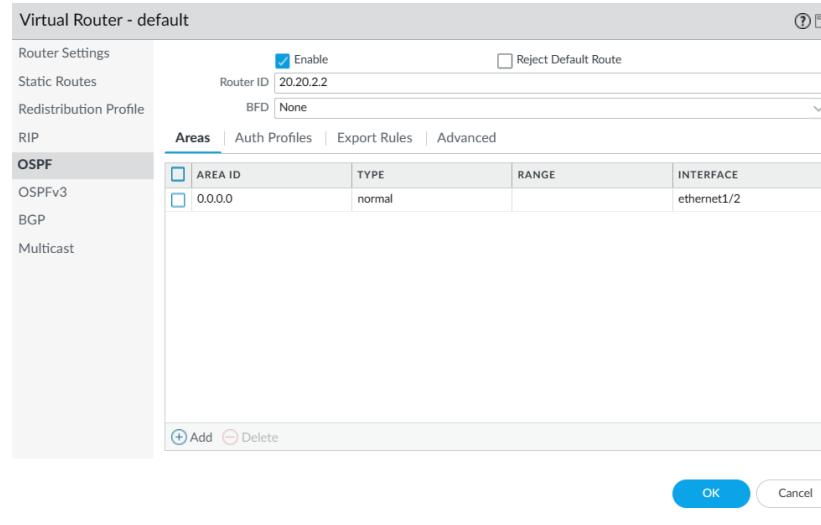


FIGURE 3.17 – Configuration du protocole l'ospf en Paloalto .

Configuration IKE Gateway

La passerelle IKE (Internet Key Exchange) est utilisée dans les VPN IPSec (Internet Protocol Security) pour faciliter la communication sécurisée entre deux ou plusieurs réseaux via Internet.

Dans notre projet, nous avons utilisé l'interface Ethernet 1/2 :

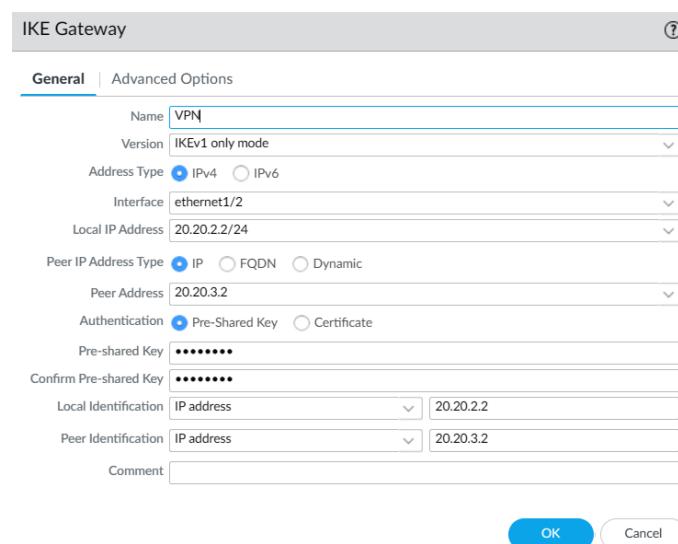


FIGURE 3.18 – Configuration IKE Gateway en Paloalto .

Configuration ipsec tunnel

Pour l'IPsec, nous configurons un tunnel IPsec avec l'interface tunnel "tunnel.100" :

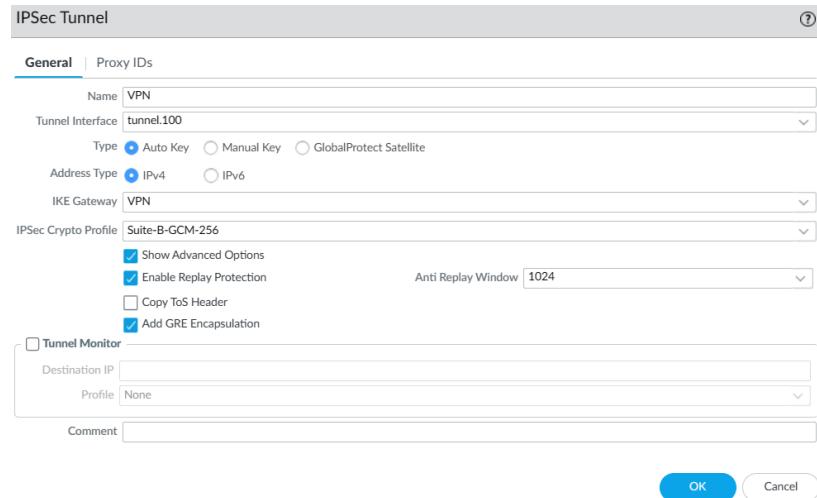


FIGURE 3.19 – Configuration ipsec tunnel en Paloalto .

Configuration DHCP serveur

Configuration de deux serveurs DHCP pour les deux sous-interfaces, incluant la définition des pools, des passerelles et des serveurs DNS.

DHCP Server DHCP Relay					
INTERFACE	MODE	PROBE IP	OPTIONS	IP POOLS	RESERVED
ethernet1/1.10	enabled	<input checked="" type="checkbox"/>	Lease: Unlimited DNS: 8.8.8.8.4.4 Gateway: 192.168.10.1	View Allocation 192.168.10.10-192.168.10.20	
ethernet1/1.20	enabled	<input checked="" type="checkbox"/>	Lease: Unlimited DNS: 8.8.8.8.4.4 Gateway: 192.168.20.1	View Allocation 192.168.20.10-192.168.20.20	

FIGURE 3.20 – Configuration DHCP serveur .

3.3 Automatisation de la configuration des équipements

Après une configuration manuelle de l'infrastructure, nous avons remarqué que la gestion des réseaux informatiques est un travail difficile car le nombre d'appareils à gérer est de plus en plus important et la configuration manuelle engendre différentes erreurs. Pour cela, nous avons implémenté une configuration automatisée au niveau des équipements routeur et commutateur et pare-feu de notre architecture.

3.3.1 Installation d'Ansible Automation Platform

Lorsqu'une entreprise souhaite déployer Ansible pour l'automatisation, elle utilise Ansible Automation Platform qui donne l'interface graphique Web pour déterminer la configuration des équipements .

Pour installer cet outil, nous avons suivi les différentes étapes au niveau d'une machine redhat. Tout d'abord , nous avons installé ansible par la commande « sudo yum-y install ansible » parce que le ansible Automation platform utilise le playbook ansible pour la configuration

```
[ahmedbenmansour@aap ~]$ ansible --version
ansible [core 2.15.9]
  config file = /etc/ansible/ansible.cfg
  configured module search path = ['/home/ahmedbenmansour/.ansible/plugins/modules', '/usr/share/ansible/plugins/modules']
  ansible python module location = /usr/lib/python3.9/site-packages/ansible
  ansible collection location = /home/ahmedbenmansour/.ansible/collections:/usr/share/ansible/collections
  executable location = /usr/bin/ansible
  python version = 3.9.18 (main, Jan 24 2024, 00:00:00) [GCC 11.4.1 20231218 (Red Hat 11.4.1-3)] (/usr/bin/python3)
  jinja version = 3.1.2
  libyaml = True
[ahmedbenmansour@aap ~]$
```

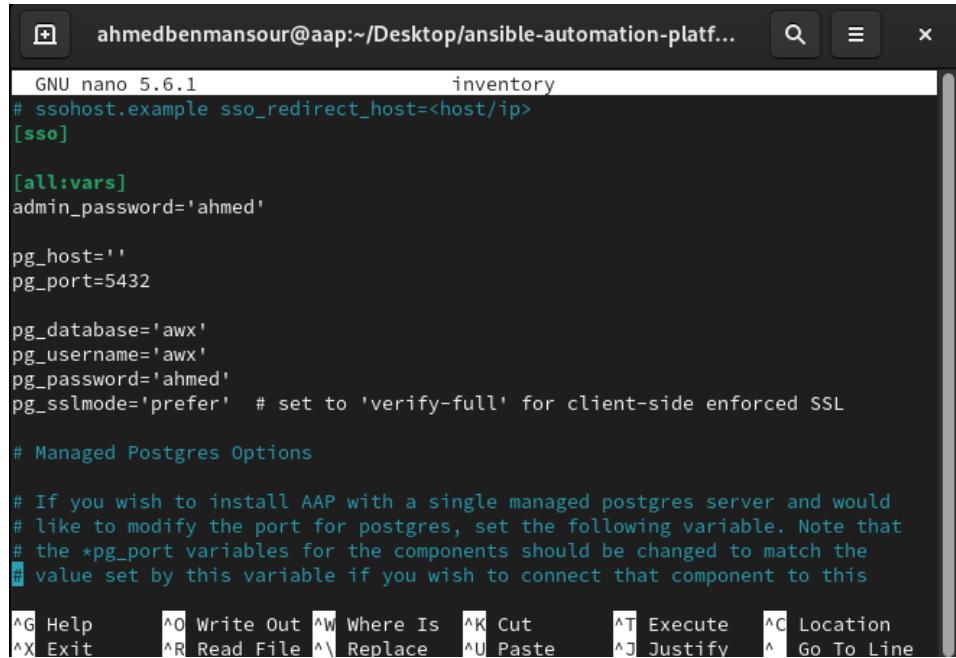
FIGURE 3.21 – Ansible version 2.15.9 .

Après cette étape, nous avons extrait l'archive téléchargée par la commande « tar xvf ansible-automation-platform-setup-bundle-2.4-5 .tar.gz »

```
[ahmedbenmansour@aap Desktop]$ cd ansible-automation-platform-setup-bundle-2.4-5-x86_64/
[ahmedbenmansour@aap ansible-automation-platform-setup-bundle-2.4-5-x86_64]$ ls
bundle  collections  group_vars  inventory  README.md  setup.sh
[ahmedbenmansour@aap ansible-automation-platform-setup-bundle-2.4-5-x86_64]$
```

FIGURE 3.22 – Téléchargement d'ansible automation platform .

Ensuite, nous avons accédé au répertoire créé par «cd ansible-automation-platform-setup-bundle-2.4-5/» et nous avons modifié le fichier d'inventaire pour définir les informations d'identification requises « "admin-password", "postgresql-password" ».



```

ahmedbenmansour@aap:~/Desktop/ansible-automation-plat...
GNU nano 5.6.1           inventory
# ssohost.example sso_redirect_host=<host/ip>
[sso]

[all:vars]
admin_password='ahmed'

pg_host=''
pg_port=5432

pg_database='awx'
pg_username='awx'
pg_password='ahmed'
pg_sslmode='prefer' # set to 'verify-full' for client-side enforced SSL

# Managed Postgres Options

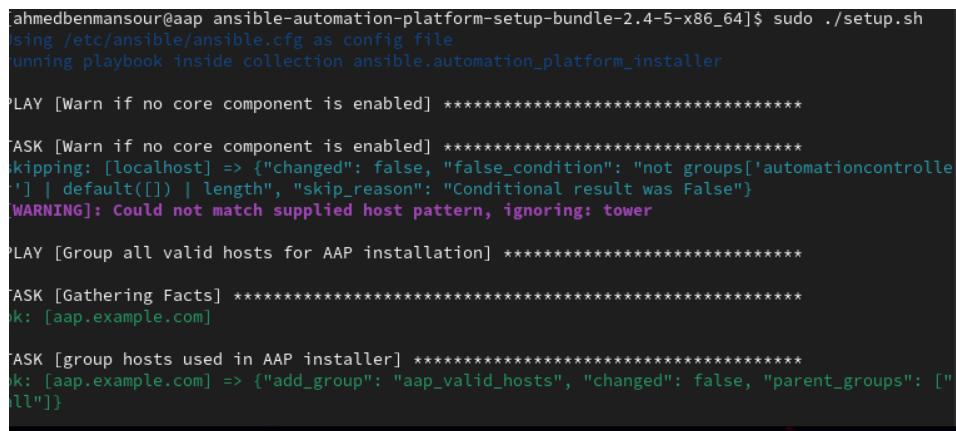
# If you wish to install AAP with a single managed postgres server and would
# like to modify the port for postgres, set the following variable. Note that
# the *pg_port variables for the components should be changed to match the
# value set by this variable if you wish to connect that component to this

^G Help      ^O Write Out ^W Where Is  ^K Cut      ^T Execute   ^C Location
^X Exit      ^R Read File ^\ Replace   ^U Paste    ^J Justify   ^L Go To Line

```

FIGURE 3.23 – Modification dans le fichier d'inventaire .

La dernière étape sera consacrée à l'installation d'Ansible Automation Platform par la commande « sudo ./setup.sh ».



```

ahmedbenmansour@aap ansible-automation-platform-setup-bundle-2.4-5-x86_64]$ sudo ./setup.sh
using /etc/ansible/ansible.cfg as config file
running playbook inside collection ansible.automation_platform_installer

PLAY [Warn if no core component is enabled] ****
TASK [Warn if no core component is enabled] ****
skipping: [localhost] => {"changed": false, "false_condition": "not groups['automationcontroller'] | default([]) | length", "skip_reason": "Conditional result was False"}
[WARNING]: Could not match supplied host pattern, ignoring: tower

PLAY [Group all valid hosts for AAP installation] ****
TASK [Gathering Facts] ****
ok: [aap.example.com]

TASK [group hosts used in AAP installer] ****
ok: [aap.example.com] => {"add_group": "aap_valid_hosts", "changed": false, "parent_groups": ["aap"]}


```

FIGURE 3.24 – Installation d'Ansible Automation platform .

Après l'installation, nous avons accédé à l'interface utilisateur Web par l'adresse IP https://192.168.1.140.

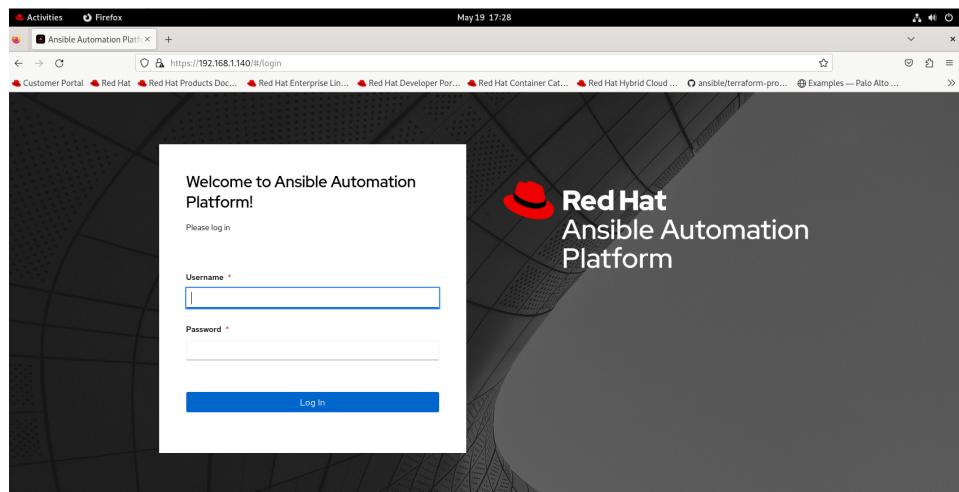


FIGURE 3.25 – Interface graphique d'ansible Automation platform .

3.3.2 Configuration d'Ansible Automation Platform

Afin d'assurer la configuration automatisée de certains équipements, nous avons procédé à la configuration d'Ansible Automation Platform.

Création d'un inventaires

Un inventaire est un ensemble d'hôtes sur lesquels des tâches peuvent être lancées. Sans inventaire défini, Ansible est inutilisable sur des machines distantes. La création d'un inventaire se fait par l'ajout des différents champs (nom, description, organisation).

Name	Sync Status	Type	Organization	Actions
192.168.1.200	Disabled	Inventory	Default	
Demo Inventory	Disabled	Inventory	Default	
paloalto 2 inv	Disabled	Inventory	Default	
palo alto inv	Disabled	Inventory	Default	
switch 2 inv	Disabled	Inventory	Default	
switch inv	Disabled	Inventory	Default	

FIGURE 3.26 – Creation des inventaire .

Création d'hôtes

Nous créons des hôtes avec des adresses IP pour permettre à Ansible Automation Platform de se connecter en SSH aux équipements :

The screenshot shows the 'Details' tab for a host named '192.168.1.10'. The host is marked as 'On'. Its activity status is shown with green and red icons. It was created on 5/2/2024 at 9:13:52 PM by 'admin'. The last modification was also on 5/2/2024 at 9:13:52 PM by 'admin'. The host is part of an inventory named 'switch 2'. Under 'Variables', there is a YAML block containing:

```

1 ---
2 ansible_connection: network_cli
3 ansible_network_os: ios

```

At the bottom, there are 'Edit' and 'Delete' buttons.

FIGURE 3.27 – Crédit d'hôte pour le commutateur .

Création du playbook

Les playbooks enregistrent et exécutent les fonctions de configuration et ils décrivent les politiques appliquées aux systèmes distants . Il existe des concepts de base importants et des mots-clés sur les playbooks représentés comme suit :

- **hosts** : le playbook va exécuter une ou plusieurs tâches sur les hôtes cibles dans Ansible Tower. Il est donc nécessaire de mentionner quel hôte ou groupe d'hôtes vont exécuter la configuration ;
- **tasks** : le playbook contient une ou plusieurs tâches qui exécutent essentiellement certaines activités sur le serveur distant. Chaque tâche est généralement appelée un module.

Ainsi que la figure 3.27 indique un exemple du playbook pour la configuration de routeur

Code Blame 54 lines (51 loc) · 1.42 KB

```
1      ---
2      - name: Configure network devices
3        hosts: all
4        gather_facts: no
5        connection: network_cli
6        tasks:
7          - name: Configure interface FastEthernet
8            ios_command:
9              commands:
10             - conf t
11             - interface FastEthernet0/0
12             - ip address 20.20.1.1 255.255.255.0
13             - no shutdown
14             - interface FastEthernet1/0
15             - ip address 20.20.2.1 255.255.255.0
16             - no shutdown
17             - interface FastEthernet2/0
18             - ip address 20.20.3.1 255.255.255.0
19             - no shutdown
20             - interface FastEthernet4/0
21             - ip address 192.168.1.200 255.255.255.0
22             - no shutdown
23           register: result_fastethernet
24           reaister: result interface config
```

FIGURE 3.28 – Exemple du playbook pour la configuration de routeur .

Configuration des projets

Après la création des playbooks, nous avons configuré les champs du projet au niveau d'Ansible Automation Platform . Un projet est une collection logique de playbooks Ansible.

Name	Status	Type	Revision	Actions
ahmed	Successful	Git	67ed2a7	
Demo Project	Successful	Git	347e44f	
firewall-project	Successful	Git	67ed2a7	
palo alto 2 project	Successful	Git	67ed2a7	
switch 2 project	Successful	Git	afa426d	
switch project	Successful	Git	afa426d	

FIGURE 3.29 – Configuration des projets .

Configuration des informations d'identification

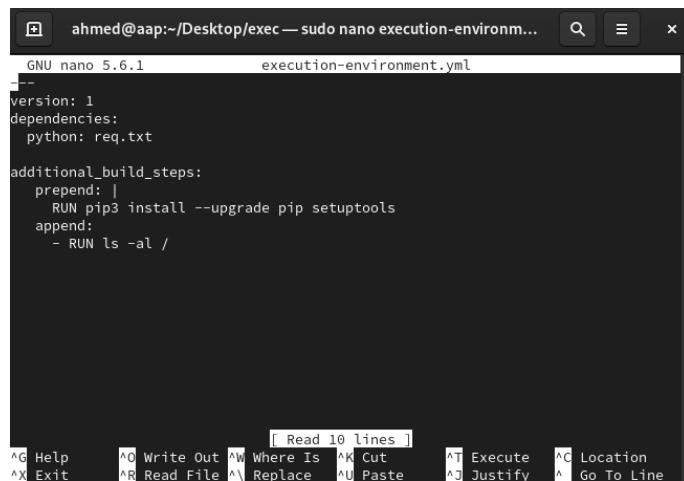
l'information d'identification est une clé pour accédé a l'équipement pour connecté avec le protocole ssh ou le lien local(url)

FIGURE 3.30 – Configuration des informations d'identification (credential) .

Configuration de l'environnement d'exécution

Pour exécuter un playbook de Palo Alto, nous devons ajouter les packages pan-os-python, pan-python et xmldict à un environnement d'exécution généré avec "ansible-builder", puis le mettre dans une image et créer un conteneur que nous pousserons avec "podman".

Création du fichier environnement d'execution



```
ahmed@aap:~/Desktop/exec — sudo nano execution-environment.yml
GNU nano 5.6.1
version: 1
dependencies:
  python: req.txt

additional_build_steps:
  prepend: |
    RUN pip3 install --upgrade pip setuptools
  append:
  - RUN ls -al /
```

FIGURE 3.31 – Configuration de l'environnement d'exécution .

Création du fichier des packages

Voici les packages pour executé le playbook de paloalto firewall



```
ahmed@aap:~/Desktop/exec — sudo nano req.txt
GNU nano 5.6.1
req.txt
pan-os-python=1.8.0
pan-python==0.17.0
xmldict==0.12.0
```

FIGURE 3.32 – Crédit du fichier des packages .

Afin de générer notre environnement d'exécution, nous utilisons la commande "ansible-builder".

```
[ahmed@aap exec]$ podman login registry.redhat.io
Username: benmansoura987
Password:
Login Succeeded!
[ahmed@aap exec]$ ansible-builder build -t py_req -v 3
Ansible Builder is generating your execution environment build context.
File context/_build/requirements.txt is already up-to-date.
File context/_build/scripts/assemble is already up-to-date.
File context/_build/scripts/install-from-bindep is already up-to-date.
File context/_build/scripts/introspect.py is already up-to-date.
File context/_build/scripts/check_galaxy is already up-to-date.
File context/_build/scripts/check_ansible is already up-to-date.
File context/_build/scripts/entrypoint is already up-to-date.
Ansible Builder is building your execution environment image. Tags: py_req
Running command:
  podman build -f context/Containerfile -t py_req context
[1/3] STEP 1/11: FROM registry.redhat.io/ansible-automation-platform-24/ee-minimal-rhel8:latest AS base
Trying to pull registry.redhat.io/ansible-automation-platform-24/ee-minimal-rhel8:latest

```

FIGURE 3.33 – Construire une collection de contenu Ansible.

Création du conteneur

Dans cette étape, nous créons un conteneur à partir de l'environnement généré, puis nous utilisons la commande "push"

```
[ahmed@aap exec]$ podman create -it --name python localhost/py_req:latest
6519b13ad155678ff976fdc458f21c27cac5774497962ad37277edbcd5731d19
[ahmed@aap exec]$ podman login quay.io
Username: benmansoura987
Password:
Login Succeeded!
[ahmed@aap exec]$ podman tag localhost/py_req:latest quay.io/benmansoura987/python-env
[ahmed@aap exec]$ podman push quay.io/benmansoura987/python-env:latest
Getting image source signatures
Copying blob 84c191436f0f done   |
Copying blob 10939ce12d40 done   |
Copying blob 6f5d6d0c9a6f done   |
Copying blob 932e25b7e78c done   |
```

FIGURE 3.34 – Crédit de conteneur.

Création un environnement d'exécution

Dans cette étape, nous créons un environnement d'exécution en Ansible Automation Platform avec une image créée :

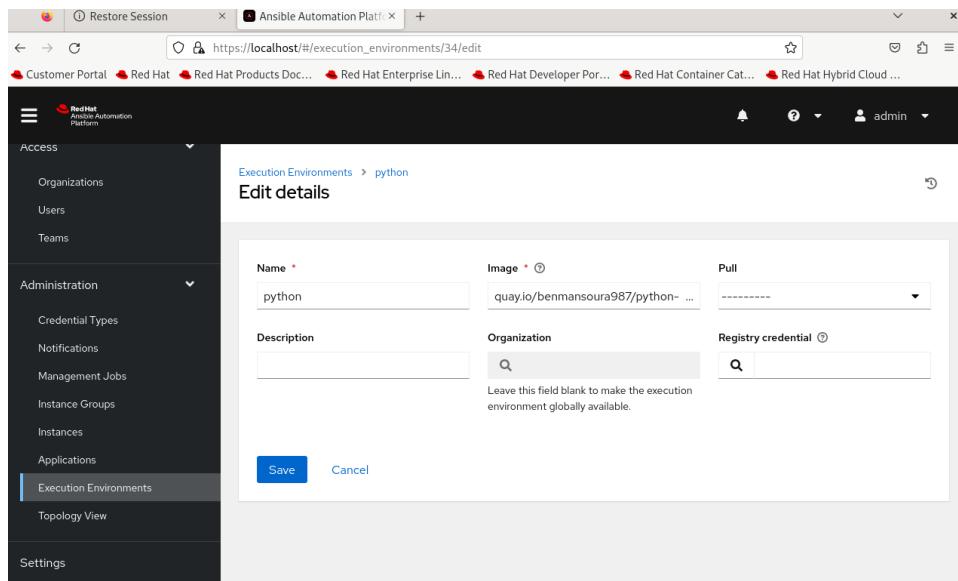


FIGURE 3.35 – Configuration d'environnement d'exécution .

Configuration d'un modèle de tâche

Un modèle de tâche est un ensemble de définitions et de paramètres permettant d'exécuter une tâche Ansible. Son utilité réside dans la possibilité d'exécuter plusieurs fois la même tâche. Il repose sur la réutilisation du contenu du playbook Ansible. Pour créer ce type de modèle, nous avons besoin de tout ce que nous avons créé précédemment. Nous avons utilisé notre inventaire et les informations d'identification pour accéder à l'appareil et le projet pour sélectionner notre playbook pour la configuration.

			Job Template	Default	Created			
>	firewall temp		Job Template	Default	5/17/2024, 2:00:54 PM			
>	palo alto 2 temp		Job Template	Default	5/17/2024, 2:00:50 PM			
>	router		Job Template	Default	5/17/2024, 2:03:01 PM			
>	switch 2 temp		Job Template	Default	5/17/2024, 1:55:37 PM			
>	switch temp		Job Template	Default	5/17/2024, 1:55:42 PM			

FIGURE 3.36 – Configuration d'un modèle de tâche .

Création d'un Workflow

Le workflow est une fonctionnalité spéciale d'Ansible Tower qui permet aux utilisateurs de créer des séquences de différentes combinaisons de modèles de tâches, de synchronisations de projets, et de les

exécuter comme une seule unité. L'utilité de cette fonctionnalité est de permettre aux utilisateurs de prendre n'importe quel nombre de playbooks et de les chaîner ensemble. Pour créer le modèle du workflow, nous avons accédé à l'option de modèle, puis au modèle de workflow, et nous avons saisi les informations nécessaires.

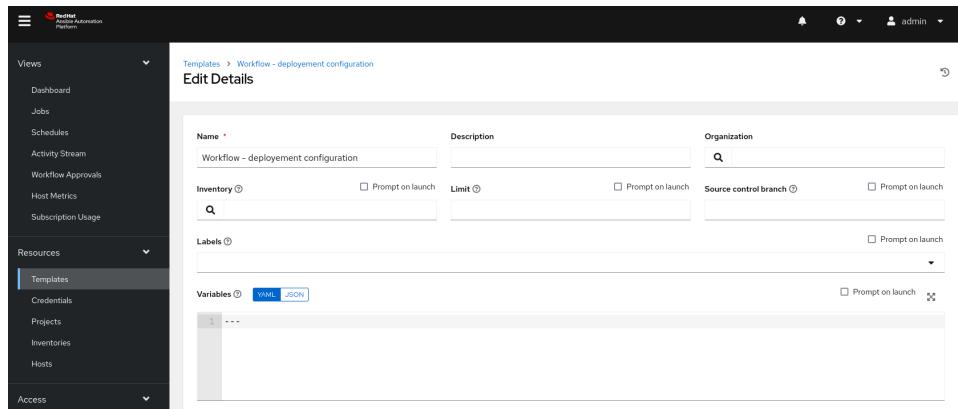


FIGURE 3.37 – Visualisation d'un modèle Workflow .

Puis, nous avons créé un Workflow pour la configuration de notre architecture

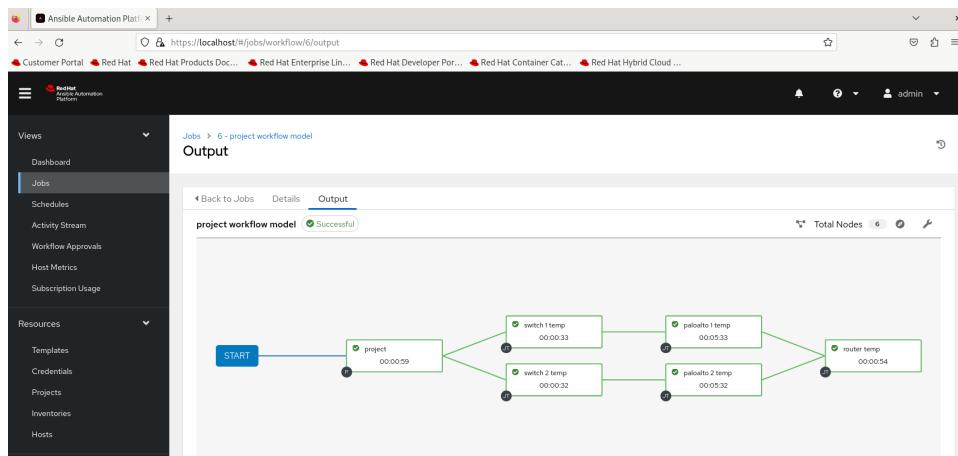


FIGURE 3.38 – Notre Workflow modèle .

3.4 Automatisation du process de project

Pour l'intégration continue et la livraison continue, nous créons un projet freestyle pour notre projet Ansible et nous le lions à GitHub.

Plugin utilisé

Premièrement, nous ajoutons le plugin Ansible Tower à Jenkins.



FIGURE 3.39 – Plugin utilisé.

Ajout identifiant et ansible URL

À cette étape, pour permettre à Jenkins d'accéder à l'ansible automation platform, nous créons un identifiant dans Jenkins.

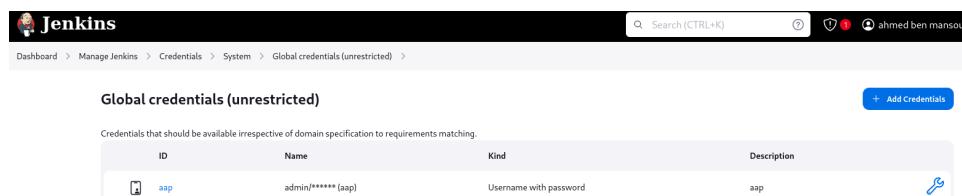


FIGURE 3.40 – Creation de l'identifiant.

Ensuite, nous ajoutons dans le système l'URL de l'ansible automation platform

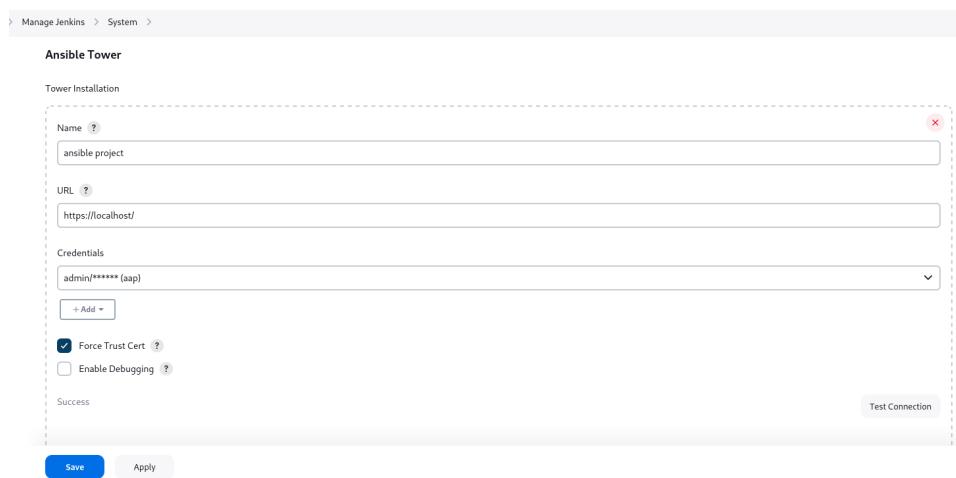


FIGURE 3.41 – Ajout URL ansible .

Création d'un projet freestyle

Ici, nous lions Jenkins à notre projet GitHub.

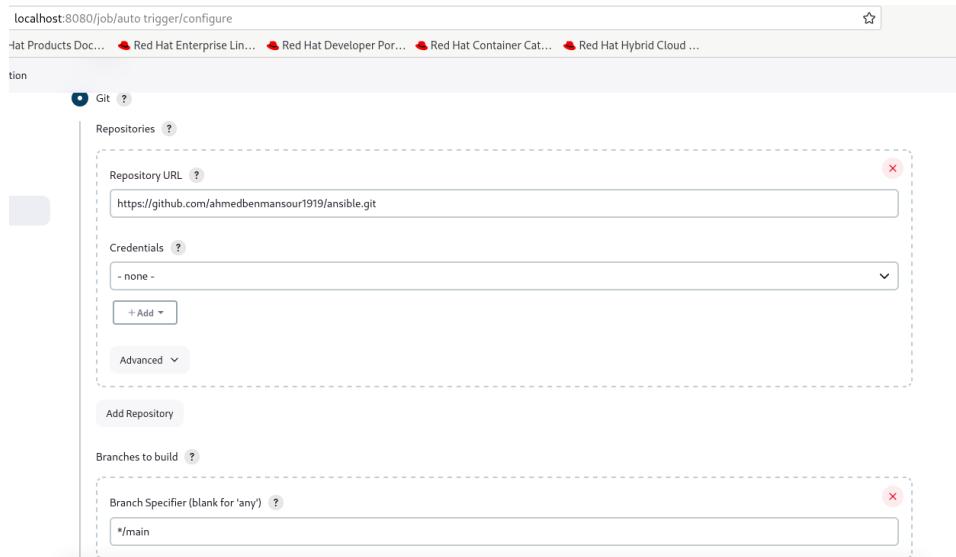


FIGURE 3.42 – Ajout URL github projet .

Et ensuite, nous ajoutons le projet workflow au projet freestyle.

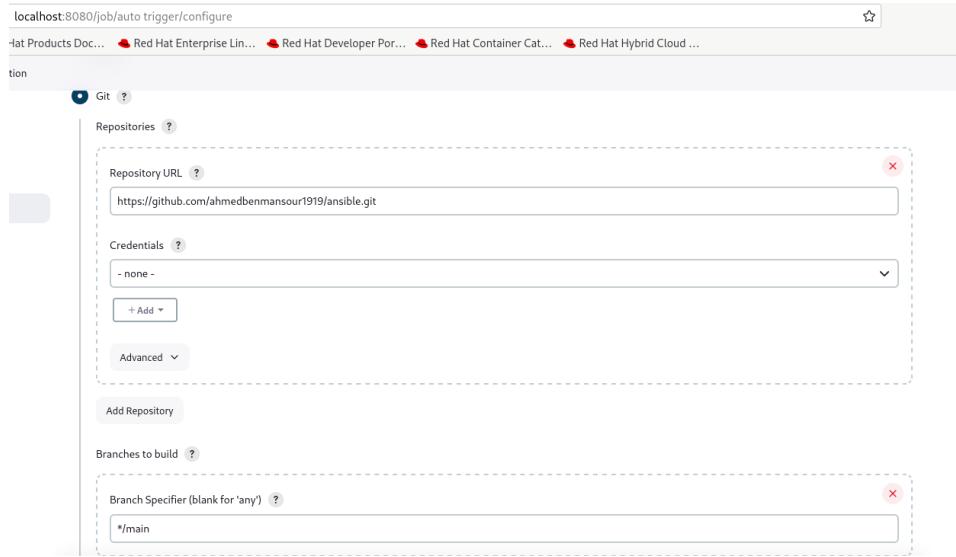


FIGURE 3.43 – Ajout de workflow projet .

Synchroniser GitHub et Jenkins

Pour synchroniser GitHub et Jenkins, nous créons un webhook sur GitHub pour notifier Jenkins. Ensuite, nous établissons un tunnel sécurisé vers notre localhost sur le port 8080 à l'aide de ngrok, permettant à GitHub de communiquer avec Jenkins pour des constructions et des déploiements automatisés.

Install ngrok

Premièrement, nous télécharge ngrok



FIGURE 3.44 – Installation de ngrok .

Créer une tunnel avec ngrok

Après cela, nous ajoutons un token(jeton) à ngrok.

```
[ahmed@aap Downloads]$ ngrok config add-authtoken 2gakwZquhWW6MjXEHpLaUWj9o9p_6GHwfkWSkFpNMD2LW9L
59
Authtoken saved to configuration file: /home/ahmed/.config/ngrok/ngrok.yml
[ahmed@aap Downloads]$ ngrok http 8080
[ahmed@aap Downloads]$
```

FIGURE 3.45 – Ajout token pour ngrok .

Après avoir ajouté le token, nous créons un tunnel pour notre URL de Jenkins, `http://localhost:8080`.

A screenshot of a terminal window titled 'ahmed@aap:~/Downloads — ngrok http 8080'. The window has two tabs: 'ahmed@aap:~/Downloads — ngrok http 8080' and 'ahmed@aap:~'. The active tab shows the output of the 'ngrok' command. It includes session details like status, account, version, region, latency, and web interface. It also shows a forwarded URL: `https://d5bd-102-156-72-46.ngrok-free.app` pointing to `http://localhost:8080`. The bottom part of the terminal shows connection statistics: ttl, open, rt1, rt5, p50, and p90.

```
ahmed@aap:~/Downloads — ngrok http 8080
ahmed@aap:~
```

```
ngrok
K8s Gateway API support available now: https://ngrok.com/r/k8sgb

Session Status      online
Account            ahmed (Plan: Free)
Version           3.9.0
Region            Europe (eu)
Latency          1713ms
Web Interface    http://127.0.0.1:4040
Forwarding        https://d5bd-102-156-72-46.ngrok-free.app -> http://localhost:8080

Connections        ttl     opn     rt1     rt5     p50     p90
                  0       0     0.00    0.00    0.00    0.00
```

FIGURE 3.46 – Création d'un tunnel en local sur le port 8080 .

ajout du url en webhook

Dans la dernière étape, nous ajoutons la nouvelle URL au webhook de GitHub.

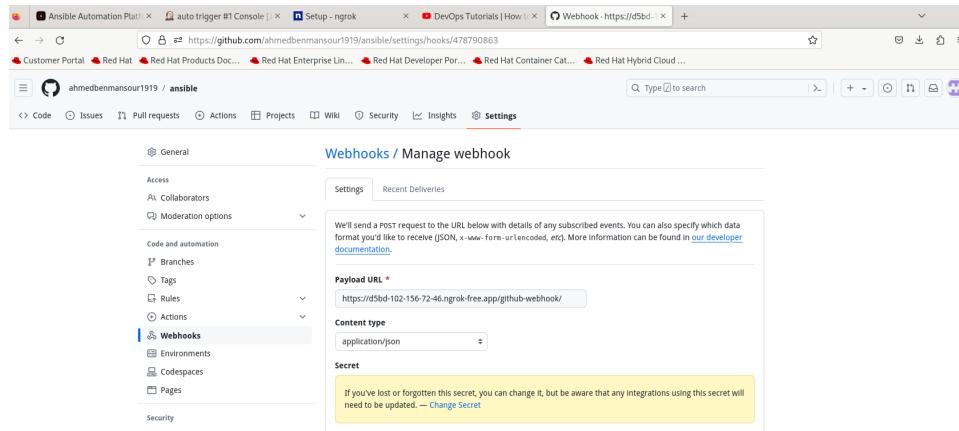


FIGURE 3.47 – Ajout de workflow projet .

3.5 Creation des machines virtuelles

Après la configuration automatisé de l'infrastructure, nous avons pu créer des machines virtuelles EVE de notre infrastructure configurée pour le nombre de candidats à la formation Palo Alto.

3.5.1 Installation et configuration du Terraform

Terraform est un outil d'infrastructure as code permettant de définir, provisionner et gérer des ressources cloud et on-premise de manière déclarative.

```
PS C:\Users\benma> terraform --version
Terraform v1.7.4
on windows_386

Your version of Terraform is out of date! The latest version
is 1.8.3. You can update by downloading from https://www.terraform.io/downloads.html
PS C:\Users\benma>
```

FIGURE 3.48 – Verification de l'installation .

3.5.2 Activation VMware Workstation Pro API REST

Tout d'abord, nous allons configurer nos identifiants pour l'API REST de "VMware Workstation Pro". Le nom d'utilisateur et le mot de passe sont définis en exécutant la commande `vmrest.exe -C`. Ensuite, nous exécuterons la commande `vmrest` pour permettre la gestion et l'automatisation des machines virtuelles via des requêtes HTTP. Cette commande retourne l'adresse IP et le numéro de port à partir desquels vous pouvez accéder au service HTTP. L'adresse IP par défaut est 127.0.0.1 :8697.

```
C:\Program Files (x86)\VMware\VMware Workstation>vmrest.exe -C
VMware Workstation REST API
Copyright (C) 2018-2024 VMware Inc.
All Rights Reserved

vmrest 1.3.1 build-23298084
Username:ahmed
New password:
Retype new password:
Processing...
Credential updated successfully

C:\Program Files (x86)\VMware\VMware Workstation>vmrest
VMware Workstation REST API
Copyright (C) 2018-2024 VMware Inc.
All Rights Reserved

vmrest 1.3.1 build-23298084
-
Using the VMware Workstation UI while API calls are in progress is not recommended and may yield unexpected results.
-
Serving HTTP on 127.0.0.1:8697
-
Press Ctrl+C to stop.
```

FIGURE 3.49 – Activation vmware workstation Pro API REST.

Ouvrez un navigateur web et allez à l'adresse `http://127.0.0.1:8697`.

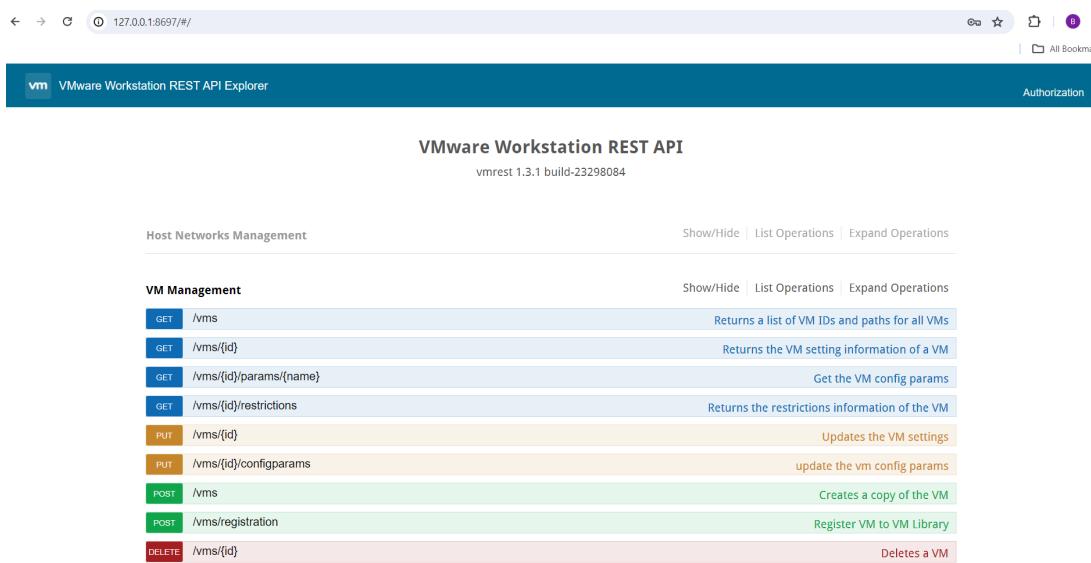


FIGURE 3.50 – Ouvrir le vWare workstation REST API sur le navigateur .

Pour obtenir l'ID de la machine virtuelle, authentifiez-vous d'abord, puis allez à l'adresse <http://127.0.0.1:8697/api/vms>.

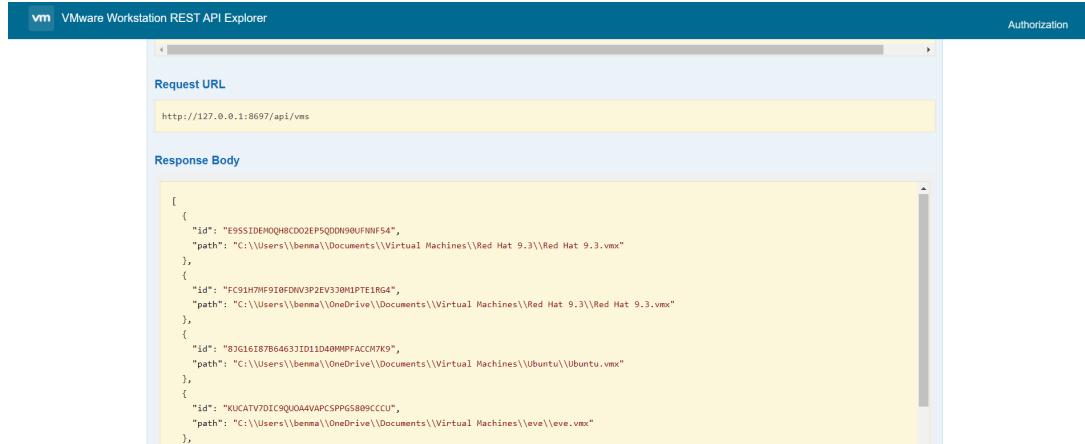


FIGURE 3.51 – L'accès sur <http://127.0.0.1:8697/api/vms> pour trouver l'ID de machine virtuelle .

3.5.3 Creation du code Terraform

Après l'activation de l'API REST de VMware Workstation Pro, nous allons créer le code Terraform de HashiCorp avec des paramètres tels que sourceid, denomination, description, path, processors, memory, et count.

```

1 terraform {
2   required_providers {
3     vmworkstation = {
4       source = "elsudano/vmworkstation"
5       version = "1.0.4"
6     }
7   }
8 }
9
10 provider "vmworkstation" {
11   user = "ahmed"
12   password = "Taraaji4!*"
13   url = "http://127.0.0.1:8697/api"
14
15   https = false
16   debug = false
17 }
18
19 resource "vmworkstation_vm" "test_machine" {
20   count = var.vmws_number
21   sourceid      = var.vmws_resource_frontend_sourceid
22   denomination = var.vmws_resource_frontend_denomination
23   description   = var.vmws_resource_frontend_description
24   path          = var.vmws_frontend_path
25   processors    = var.vmws_resource_frontend_processors
26   memory        = var.vmws_resource_frontend_memory
27 }
28

```

FIGURE 3.52 – Crédit à la création du fichier "main.tf".

Ceci sont les variables utilisées dans le fichier "main.tf" :

```

variable "vmws_resource_frontend_sourceid" {
  type    = string
  description = "The ID of the VM that to use for clone at the new"
  default   = "KUCATV7DIC9QUA4VAPCSPPG5809CCU"
}

variable "vmws_resource_frontend_denomination" {
  type    = string
  description = "The Name of VM in WS"
  default   = "vmcop_${count.index}"
}

variable "vmws_resource_frontend_description" {
  type    = string
  description = "the instance"
}

variable "vmws_frontend_path" {
  type    = string
  description = "The Path where will be our instance in VmWare"
  default   = "C:\\\\Users\\\\benma\\\\OneDrive\\\\Documents\\\\Virtual Machines\\\\vmcop_${count.index}\\\\vmcop_${count.index}.vmx"
}

variable "vmws_resource_frontend_processors" {
  type    = string
  description = "The number of processors of the Virtual Machine"
  default   = "1"
}

variable "vmws_resource_frontend_memory" {
  type    = string
  description = "The size of memory to the Virtual Machine"
  default   = "512"
}

variable "vmws_number" {
  type    = string
  description = "the number of VMs"
  default   = "1"
}

```

FIGURE 3.53 – Création du fichier "variable.tf" .

Voici le fichier "output.tf" contenant l'ID de la nouvelle machine virtuelle :

```

output "vmws_frontend_id" {
  value   = vmworkstation_vm_tf_test[*].id
  description = "This is the id of the VM"
}

```

FIGURE 3.54 – Création du fichier "output.tf" .

CHAPITRE 4

TEST ET VALIDATION DE LA SOLUTION

Introduction

Après avoir terminé la partie implémentation et mise en place du projet, nous passons maintenant à la phase de test, qui est la partie la plus essentielle du projet. elle consiste à présenter les tests nécessaires pour la vérification des connectivités entre les différents composants de l'architecture

4.1 Test et validation de la configuration

Après les phases de configuration de notre architecture proposée, nous nous proposons dans cette section de présenter les différentes connectivités entre les équipements et les résultats de cette configuration.

4.1.1 Test ping vlans

Nous avons fait des tests "ping" entre les équipements pour voir le connectivité et testé le vlans :

```
VPCS> ping 192.168.10.11  
84 bytes from 192.168.10.11 icmp_seq=1 ttl=64 time=4.280 ms  
84 bytes from 192.168.10.11 icmp_seq=2 ttl=64 time=1.427 ms  
84 bytes from 192.168.10.11 icmp_seq=3 ttl=64 time=2.081 ms  
84 bytes from 192.168.10.11 icmp_seq=4 ttl=64 time=2.044 ms  
84 bytes from 192.168.10.11 icmp_seq=5 ttl=64 time=1.895 ms  
  
VPCS> ping 192.168.20.11  
192.168.20.11 icmp_seq=1 timeout  
192.168.20.11 icmp_seq=2 timeout  
192.168.20.11 icmp_seq=3 timeout  
192.168.20.11 icmp_seq=4 timeout  
192.168.20.11 icmp_seq=5 timeout
```

FIGURE 4.1 – Test ping dans les vlans .

4.1.2 Test web serveur

Nous avons testé l'accès au serveur web avec IP adresse dhcp "192.168.10.11"

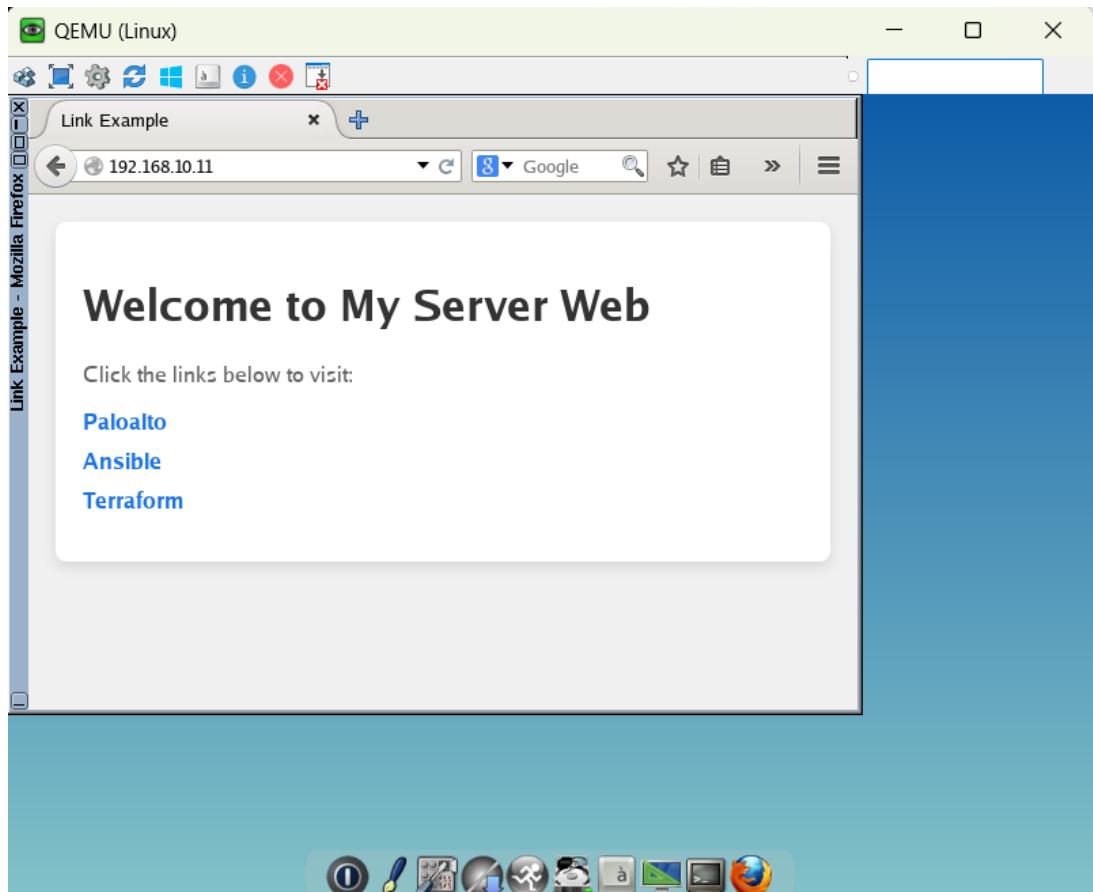


FIGURE 4.2 – Test l'accès au serveur web .

4.1.3 Test serveur dhcp

Testé le serveur dhcp dans le firewall avec le VPC

```
VPC1
VPCS> ip dhcp
DORA IP 192.168.10.12/24 GW 192.168.10.1
VPCS> 
```

A screenshot of a terminal window titled "VPC1". The prompt is "VPCS>". The user enters the command "ip dhcp", which is responded to with "DORA IP 192.168.10.12/24 GW 192.168.10.1". The terminal window has a black background with white text.

FIGURE 4.3 – Test le serveur dhcp .

4.1.4 Test IPsec

Ensuite on à testé la connectivité entre les deux pare-feu avec le IPsec

IKE Gateway/Satellite			Tunnel Interface									
NAME	STATUS	TYPE	INTERFACE	LOCAL IP	PEER ADDRESS	STATUS	INTERFACE	VIRTUAL ROUTER	VIRTUAL SYSTEM	SECURITY ZONE	STATUS	COMMENT
VPN	Tunnel Info	Auto Key	ethernet1/2	20.20.2.2/24	20.20.3.2	IKE Info	tunnel100	default (Show Route(s))		VPN		

FIGURE 4.4 – Test le protocole IPsec sur le pare-feu .

4.2 Test Automatisation et la création de virtuelle machine

4.2.1 Test la configuration des équipement avec ansible

Pour faire la configuration de notre architecture avec ansible on doit synchronisé les playbooks avec "project" et executé le playbook de chaque équipement

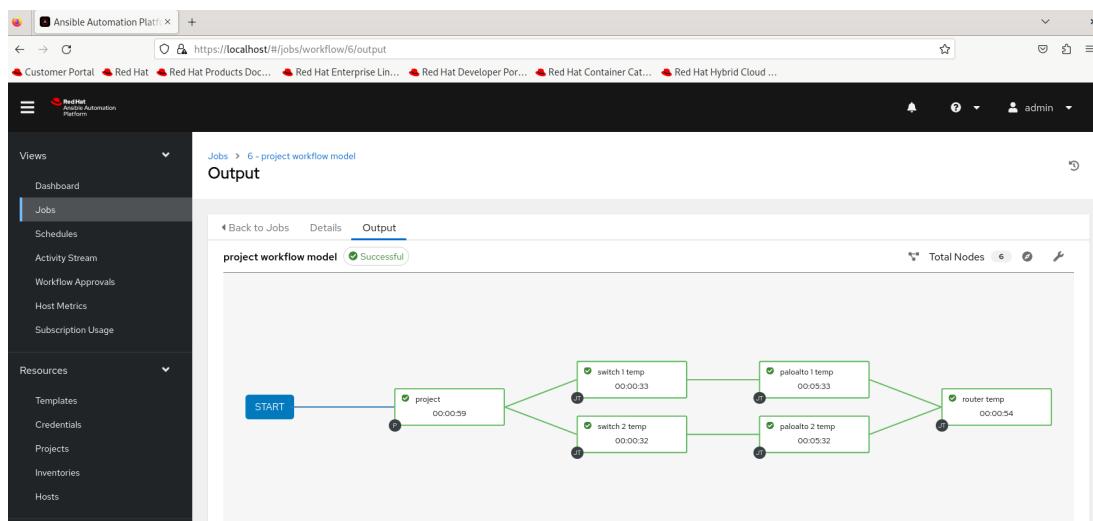


FIGURE 4.5 – Test la configuration avec ansible .

4.2.2 Test configuration des équipements avec Jenkins

Testé la configuration avec jenkins des modification dans les playbooks l'execution ce fait comme suit :

```
Activities Firefox May 27 13:43 •
Restore Session Ansible Automation Plat... auto trigger #3 Console | ansible/hello.yaml at mail... +
localhost:8080/job/auto-trigger/3/console
Customer Portal Red Hat Red Hat Products Doc... Red Hat Enterprise Lin... Red Hat Developer Por... Red Hat Container Cat... Red Hat Hybrid Cloud ...

Dashboard > auto trigger > #3 > Console Output

< Previous Build

> git rev-parse refs/remotes/origin/main^{commit} # timeout=10
Checking out Revision 17901cf2014017ebf5a905a148ba4417defebc3 (refs/remotes/origin/main)
> git config core.sparsecheckout # timeout=10
> git checkout -f 17901cf2014017ebf5a905a148ba4417defebc3 # timeout=10
Commit message: "Update hello.yaml"
> git rev-list --no-walk 6e261b2e038c7184454f289b7cdfa23cd149681 # timeout=10
Template Job URL: https://localhost/#/workflows/23
project => successful https://localhost/#/jobs/24

switch 2 temp => waiting https://localhost/#/jobs/25

paloalto 1 temp => successful https://localhost/#/jobs/27

switch 1 temp => successful https://localhost/#/jobs/26

paloalto 2 temp => successful https://localhost/#/jobs/28

Tower completed the requested job
Build step 'Ansible Tower' changed build result to SUCCESS
Finished: SUCCESS
```

FIGURE 4.6 – Test la configuration avec jenkins .

4.2.3 Test de la création de machine virtuelle

Finalement, on a executé le code terraform pour crée des autres machines virtuelles de notre virtuelle machine initial

The screenshot shows the VS Code interface with the following details:

- File Explorer (Left):** Shows files: `EXPLORER`, `MV`, `.terraform`, `.terraform.lock.hcl`, `main.tf` (selected), `output.tf`, `terraform.tfstate`, and `terraform.tfstate.backup`.
- Code Editor (Center):** Displays the `main.tf` file content:

```
resource "vmworkstation" "tf_test" {
  name = "tf-test"
  required_providers {
    vmworkstation = {
      source = "elisudano/vmworkstation"
      version = "1.0.4"
    }
  }

  provider "vmworkstation" {
    user = "ahmed"
    password = "tarajj14"
    url = "http://127.0.0.1:8697/api"

    https = false
    debug = false
  }

  resource "vmworkstation_vm" "tf_test" {
    denomination = "vcpu-vga"
    sourcecid = "K0CATV7D1C9Q00A4VAPCSPPG5809CCU"

    description = "ahmed"
    path = "%USER%\benma\OneDrive\Documents\Virtual Machines\vmcopy.vmx"
    processors = "2"
    memory = "4048"
  }
}
```

- Terminal (Bottom):** Shows the output of the Terraform command:

```
vmworkstation_vm.tf: test: Still creating... [9m11s elapsed]
vmworkstation_vm.tf: test: Creation complete after 9m03s [id=49QNATJGFJ7K0KSLK9H8LQYQULCQLKGS]

Apply complete! Resources: 1 added, 0 changed, 0 destroyed.
```
- Status Bar (Bottom):** Shows the current file is `main.tf`, line 26, column 1, with 8 spaces and 1 tab, and the file is a `Terraform` file.

FIGURE 4.7 – Test de la création d'une machine virtuelle.

Conclusion

Le présent chapitre a été consacré à la validation et le test du projet. En effet, nous nous sommes intéressées à la présentation des différents tests de configuration qui est déjà configuré au chapitre précédent et la création de machine virtuelle avec terraform

CONCLUSION GÉNÉRALE ET PERSPECTIVES

Ce travail s'est avéré extrêmement bénéfique et instructif en termes de connaissances acquises. Il nous a permis de faire le lien entre nos apprentissages académiques et le monde réel, tout en découvrant le fonctionnement de technologies avancées telles que Palo Alto, Ansible et Terraform. Ce stage nous a également offert l'opportunité de nous intégrer dans le monde professionnel, en nous confrontant aux défis réels et en nous permettant de développer des compétences pratiques indispensables dans le domaine de la cybersécurité et de l'automatisation des réseaux.

Notre Projet de Fin d'Études, réalisé au sein de la société OneTech Business Solutions sur une période de quatre mois, avait pour objectif de concevoir et d'implémenter une solution innovante consistant en des laboratoires préconfigurés. Ces laboratoires, prêts à l'emploi, permettent aux étudiants et aux professionnels d'accéder rapidement à des environnements d'apprentissage pratiques et fonctionnels. En offrant des infrastructures de formation prêtes à l'utilisation, nous avons considérablement réduit le temps nécessaire pour préparer les environnements de test, facilitant ainsi l'apprentissage et l'acquisition de compétences en cybersécurité.

Le projet nous a permis de mettre en pratique une gamme variée de technologies et d'outils. Nous avons utilisé Terraform pour la création et la gestion d'infrastructures virtualisées de manière efficace et reproductible. Grâce à Ansible, nous avons pu automatiser la configuration des équipements, ce qui a considérablement simplifié la gestion des configurations et réduit le risque d'erreurs humaines. Jenkins, en tant qu'outil d'intégration continue, nous a permis d'intégrer et de déployer nos configurations de manière fluide et régulière, assurant ainsi une mise à jour continue des environnements de laboratoire.

En outre, notre solution inclut la création de machines virtuelles EVE-NG à partir de notre machine virtuelle initiale avec Terraform. EVE-NG (Emulated Virtual Environment Next Generation) est une plateforme de virtualisation réseau avancée qui permet de simuler des environnements ré-

seau complexes, essentiels pour les formations en sécurité réseau. En automatisant la configuration de ces machines virtuelles avec Ansible, nous avons assuré une consistance et une répétabilité dans la configuration des environnements de formation, permettant ainsi une expérience d'apprentissage optimisée.

Ce projet a non seulement enrichi nos connaissances techniques, mais il a également développé nos compétences en gestion de projet, en résolution de problèmes et en travail d'équipe. Nous avons appris à travailler de manière collaborative, à gérer les contraintes de temps et à adapter nos solutions aux besoins spécifiques de l'entreprise.

Enfin, il existe plusieurs voies pour améliorer ce travail dans le futur. Nous pourrions, par exemple, explorer l'intégration de nouvelles technologies et outils pour augmenter encore l'efficacité et la flexibilité de nos laboratoires. La création de modules de formation supplémentaires, adaptés à des niveaux de compétence variés, pourrait également être envisagée pour élargir l'accessibilité et l'utilité de nos laboratoires préconfigurés.

En conclusion, ce projet a été une expérience très enrichissante qui nous a permis de mettre en œuvre nos connaissances théoriques dans un contexte pratique, de découvrir de nouvelles technologies et de nous préparer de manière significative pour notre future carrière professionnelle. Les compétences et les connaissances acquises durant ce projet seront sans aucun doute des atouts précieux pour nos futures entreprises et pour notre développement professionnel continu.

ANNEXE A

LES PLAYBOOKS

A.1 Les playbook pour les commutateurs

Les deux figure A.1 et A.2 représente le playbook de la configuration pour les deux vlans 10 et 20 et configuration de l'interface vlan1 par le ssh sur le terminal de commutateur

```
1      ---
2      - name: Configure VLANs on Switch
3          hosts: all
4          gather_facts: no
5          connection: network_cli
6          tasks:
7              - name: Create VLANs
8                  ios_command:
9                      commands:
10                         - conf t
11                         - vlan 10
12                         - name bleu
13                         - vlan 20
14                         - name green
```

FIGURE A.1 – Crédit des vlans.

```
- name: Assign VLANs to interfaces
  ios_command:
    commands:
      - conf t
      - interface Ethernet0/0
      - switchport trunk encapsulation dot1q
      - switchport mode trunk
      - interface Ethernet0/1
      - switchport access vlan 10
      - switchport mode access
      - interface Ethernet0/2
      - switchport access vlan 10
      - switchport mode access
      - interface Ethernet0/3
      - switchport access vlan 20
      - switchport mode access
      - interface Ethernet1/0
      - switchport access vlan 20
      - switchport mode access
  register: create_vlans

- name: save
  ios_command:
    commands:
      - wr

- name: Debug registered var
  debug: var=sh_vlan_br_result.stdout_lines
```

FIGURE A.2 – Configuration des vlans.

A.2 Les playbooks pour les pare-feux

La figure A.3 représente la configuration de "username" et le "password" pour accès a l'interface graphique de pare-feu . après on a commencé par la configuration du règle de sécurité et le routeur virtuelle

```
1      ---
2      - name: Create security policies
3          hosts: all
4          gather_facts: false
5          connection: local
6
7      vars:
8          device:
9              ip_address: "192.168.1.132"
10             username: "admin"
11             password: "Ahmed123"
12
13     tasks:
14         - name: Create first security rule
15             paloaltonetworks.panos.panos_security_rule:
16                 provider: "{{ device }}"
17                 rule_name: 'permit'
18                 source_zone: ['any']
19                 destination_zone: ['any']
20                 source_ip: ['any']
21                 destination_ip: ['any']
22                 application: ['any']
23                 service: ['application-default']
24                 action: 'allow'
25
26         - name: Create Virtual Router
27             paloaltonetworks.panos.panos_virtual_router:
28                 provider: '{{ device }}'
29                 name: 'vr-1'
30
31 ---
```

FIGURE A.3 – Configuration de l'accès sur le pare-feu et de la règle de sécurité.

Dans la figure A.4 on a crée deux profil de gestion "trust" et "untrust" et les trois zones sécurité "trust" , "untrust" et "VPN"

```
31      - name: ensure mngt profile trust exists
32          paloaltonetworks.panos.panos_management_profile:
33              provider: '{{ device }}'
34              name: 'trust'
35              ping: true
36              ssh: true
37              https: true
38
39      - name: ensure mngt profile untrust exists
40          paloaltonetworks.panos.panos_management_profile:
41              provider: '{{ device }}'
42              name: 'untrust'
43              ping: true
44
45      - name: create trust zone on a firewall
46          paloaltonetworks.panos.panos_zone:
47              provider: '{{ device }}'
48              zone: 'trust2'
49              mode: 'layer3'
50
51      - name: create untrust zone on a firewall
52          paloaltonetworks.panos.panos_zone:
53              provider: '{{ device }}'
54              zone: 'untrust2'
55              mode: 'layer3'
56
57      - name: create VPN zone on a firewall
58          paloaltonetworks.panos.panos_zone:
59              provider: '{{ device }}'
60              zone: 'VPN'
61              mode: 'layer3'
62
```

FIGURE A.4 – Création des profil de gestion et des zones sécurité.

Dans cette figure on a crée l'interface ethernet et l'interface tunnel pour le premier pare-feu

ansible / paloalto.yaml

Code Blame 213 lines (175 loc) · 5.78 KB

```
64      - name: Create ethernet interfaces
65          paloaltonetworks.panos.panos_interface:
66              provider: '{{ device }}'
67              if_name: 'ethernet1/2'
68              mode: 'layer3'
69              zone_name: 'untrust'
70              vr_name: 'vr-1'
71              ip: ['20.20.2.2/24']
72              enable_dhcp: false
73              management_profile: 'untrust'
74          register: result
75
76      - name: create tunnel.100
77          paloaltonetworks.panos.panos_tunnel:
78              provider: '{{ device }}'
79              if_name: "tunnel.100"
80              zone_name: 'VPN'
81              vr_name: 'vr-1'
82              management_profile: 'untrust'
83
84      - name: ensure mngt profile trust exists
85          paloaltonetworks.panos.panos_management_profile:
86              provider: '{{ device }}'
87              name: 'trust'
88              ping: true
89              ssh: true
90              https: true
91
92      - name: ensure mngt profile untrust exists
93          paloaltonetworks.panos.panos_management_profile:
94              provider: '{{ device }}'
95              name: 'untrust'
96              ping: true
97
```

FIGURE A.5 – Création des interfaces.

Ensuite, on a configuré deux sous-interface "ethernet1/1.10" et "ethernet1/1.20" et on a créé le static route pour le mapping

```

99      - name: ethernet1/1.10 as static
100     paloaltonetworks.panos.panos_l3_subinterface:
101       provider: '{{ device }}'
102       name: "ethernet1/1.10"
103       tag: 10
104       enable_dhcp: false
105       ip: ["192.168.10.1/24"]
106       zone_name: "trust"
107       vr_name: 'vr-1'
108       management_profile: 'trust'
109
110      - name: ethernet1/1.20 as static
111     paloaltonetworks.panos.panos_l3_subinterface:
112       provider: '{{ device }}'
113       name: "ethernet1/1.20"
114       tag: 20
115       enable_dhcp: false
116       ip: ["192.168.20.1/24"]
117       zone_name: "trust"
118       vr_name: 'vr-1'
119       management_profile: 'trust'
120
121      - name: Create route
122     paloaltonetworks.panos.panos_static_route:
123       provider: '{{ device }}'
124       virtual_router: 'vr-1'
125       name: 'route'
126       destination: '0.0.0.0/0'
127       nexthop: '20.20.2.1'
128       metric: "10"
129

```

FIGURE A.6 – Création des sous-interface.

Ici on a créé la route static pour le VPN et le mapping

```

- name: Create route 'vpn'
  paloaltonetworks.panos.panos_static_route:
    provider: '{{ device }}'
    virtual_router: 'vr-1'
    name: 'ROUTE VPN'
    destination: '192.168.0.0/16'
    interface: 'tunnel.100'
    nexthop_type: 'none'
    metric: "10"

```

FIGURE A.7 – Création de route static.

Dans cette étape on a créé le IKE(Internet Key Exchange) profile et on l'a ajouté au pare-feu

```
- name: Add IKE crypto config to the firewall
  paloaltonetworks.panos.panos_ike_crypto_profile:
    provider: '{{ device }}'
    state: 'present'
    name: 'vpn-ike-profile'
    dh_group: ['group2']
    authentication: ['sha1']
    encryption: ['aes-128-cbc']
    lifetime_seconds: '28800'

- name: Add IKE gateway config to the firewall
  paloaltonetworks.panos.panos_ike_gateway:
    provider: '{{ device }}'
    state: 'present'
    name: 'IKEGW-Ansible'
    version: 'ikev1'
    interface: 'ethernet1/2'
    local_ip_address_type: 'ip'
    local_ip_address: '20.20.2.2/24'
    peer_ip_value: '20.20.3.2'
    pre_shared_key: 'ahmed123'
    peer_id_type: 'ipaddr'
    peer_id_value: '20.20.3.2'
    local_id_type: 'ipaddr'
    local_id_value: '20.20.2.2'
    enable_passive_mode: false
    enable_liveness_check: false
    liveness_check_interval: '5'
    ikev1_exchange_mode: 'auto'
    ikev1_crypto_profile: 'vpn-ike-profile'
```

FIGURE A.8 – Création de IKE(Internet Key Exchange) .

Et enfin la configuration de IPsec profil et crée un canal IPsec avec l'interface "tunnel.100" et le "IKE gateway" et le "IPsec profile"

```
- name: Add IPSec crypto config to the firewall
  paloaltonetworks.panos.panos_ipsec_profile:
    provider: '{{ device }}'
    state: 'present'
    name: 'ipsec-vpn-profile'
    esp_authentication: ['sha1']
    esp_encryption: ['aes-128-cbc']
    lifetime_seconds: '3600'

- name: Add IPSec tunnel to IKE gateway profile
  paloaltonetworks.panos.panos_ipsec_tunnel:
    provider: '{{ device }}'
    name: 'IPSecTunnel-Ansible'
    tunnel_interface: 'tunnel.100'
    ak_ike_gateway: 'IKEGW-Ansible'
    ak_ipsec_crypto_profile: 'ipsec-vpn-profile'
    state: 'present'
```

FIGURE A.9 – Configuration de IPsec profil et création d'un canal .

WEBOGRAPHIE

- [1] 1. Logo otbs. <https://www.ilboursa.com/marches/onetech-holding-vers-un-nouveau-cycle-de-croissance>. Dernier accès : 25 février 2024.
- [2] 10. Architecture d'un load balancer matériel. <https://medium.com/@ethemm/load-balancers-explained-and-basic-setup-guide-8b2d3ef38bd6>. Dernier accès : 9 avril 2024.
- [3] 11. Github logo. <https://logos-world.net/github-logo/>. Dernier accès : 2 mars 2024.
- [4] 12. Eve-ng logo. <https://www.eve-ng.net/>. Dernier accès : 12 avril 2024.
- [5] 13. Terraform logo. <https://www.cleanpng.com/png-terraform-hashicorp-microsoft-azure-infrastructure-2641388/>. Dernier accès : 8 février 2024.
- [6] 14. Ansible logo. <https://commons.wikimedia.org/wiki/File:AnsibleLogo.png>. Dernier accès : 30 mars 2024.
- [7] 15. Jenkins logo. <https://commons.wikimedia.org/wiki/File:JenkinsLogo.svg>. Dernier accès : 13 avril 2024.
- [8] 2. Hyperviseur type 1 et 2. <https://www.compufirst.com/compufirst-lab/serveur/qu-est-ce-qu-un-hyperviseur/main.do?appTreeId=45692>. Dernier accès : 3 avril 2024.
- [9] 3. Palo alto logo. <https://www.paloaltonetworks.com/company/brand>. Dernier accès : 19 février 2024.
- [10] 4. Panorama architecture. <https://www.paloguard.com/Panorama.asp>. Dernier accès : 10 mars 2024.
- [11] 5. Cycle de vie devops. <https://www.plunge.cloud/blog/approche-devops>. Dernier accès : 7 avril 2024.
- [12] 6. Calms model en devops. <https://utibeabasiumanah6.medium.com/the-calms-model-in-devops-a8b86d9457f0>. Dernier accès : 21 février 2024.
- [13] 7. Approche ci/cd. <https://www.cegid.com/fr/blog/approche-cicd-definition-et-avantages/>. Dernier accès : 5 avril 2024.

- [14] 8. Principe de conteneurisation. <https://phelepjeremy.wordpress.com/2017/06/21/la-conteneurisation/>. Dernier accès : 28 mars 2024.
- [15] 9. Architecture d'un load balancer logiciel. <https://www.devskillbuilder.com/exploring-oci-load-balancer-service-a-guide-for-beginners-fc524056b868>. Dernier accès : 17 février 2024.

Résumé

Ce travail s'inscrit dans le cadre de notre Projet de Fin d'Études, réalisé au sein de la société OneTech Business Solutions. La solution proposée vise à mettre en place un laboratoire automatisé offrant une plateforme d'apprentissage pratique pour le personnel de l'entreprise et ses partenaires. Le projet consiste à concevoir une architecture réseau, à automatiser la configuration des équipements, et enfin à créer plusieurs machines virtuelles configurées.

Mots clés— automation, plateforme d'apprentissage, architecture, machines virtuelles

Abstract

This work is part of our Final Year Project, developed within the company OneTech Business Solutions. The proposed solution aims to set up an automated lab providing a practical learning platform for the company's staff and its partners. The project involves designing a network architecture, automating the configuration of equipment, and finally creating multiple configured virtual machines.

Keywords— automation, learning platform, architecture, virtual machines