UNIVERSITE SAAD DAHLAB BLIDA 01

Faculté des Sciences

Département d'Informatique

Mémoire du Projet de Fin d'Études

Présenté pour l'obtention du diplôme

MASTER

Filière : Systèmes informatiques et Réseaux.

Thème

Mise en œuvre d'un système d'optimisation réseau à l'aide de l'automatisation

Présenté Par :

Allani Youcef

Chergui Affane

Soutenu le :10/07/2023, Devant le jury composé de :

Mr. Douga Yassine	Président
Mme. Ghebghoub	Examinateur
Mme. Bey Fella	Encadrante

Année universitaire

2022/2023

Remerciements

Nous commençons par remercier Allah tout-puissant pour les bienfaits qu'Il nous a accordés en termes de patience, de conscience, d'intelligence et de dignité, qui nous ont tous été donnés et nous ont permis d'atteindre cette étape de la vie en compagnie de nos proches.

Nous souhaitons exprimer notre gratitude envers notre promoteur, Mr ZAFOUNE Youcef, ainsi un grand merci aux ingénieurs de la DCDSI de Sonelgaz service particulièrement Mme DEBBAGH Souhila et Mr BELHOUSSINE Fadhel.

De plus, nous tenons à exprimer notre reconnaissance envers Mme BEY Fella, notre encadreur, pour son orientation et sa confiance en acceptant de nous encadrer pour ce projet de fin d'études.

Dédicace

À nos chers parents, pour toutes leurs sacrifices, leur amour, leur tendresse, leur soutien et leurs prières tout au long de nos études,

À nos chers frères et sœurs, pour leur encouragement constant et leur soutien moral,

À tous les membres de la famille, pour leur soutien tout au long de notre parcours universitaire,

Et à tous ceux qui ont contribué à notre réussite jusqu'à ce jour,

Ce travail peut être considéré comme la réalisation de vos aspirations dont vous êtes si fiers, et comme le fruit de votre soutien inépuisable,

Merci d'être toujours présents pour nous.

Abstract

The increasing complexity of computer networks due to rapid technological advances requires the development of efficient and automated configuration solutions. Manual configuration methods are slow and prone to errors, resulting in inefficiencies in network management. To solve this problem, this master memory focuses on the implementation of a network optimization system with the help of automation, which aims to set up a communication system between separate network protocols. The project also aims to streamline the network administration process by providing a user-friendly graphical interface for configuring VLANs, DHCP and retrieving the different states of STP and HSRP protocols. By leveraging automation and intuitive interfaces, the proposed application maximizes network efficiency, minimizes errors and reduces the need for human intervention in network configuration tasks.

Keywords: Computer networks, Automatic configuration, Complexity, Network management, Graphical interface, VLAN, HSRP, Automation.

Résumé

La complexité croissante des réseaux informatiques due aux avancées technologiques rapides nécessite le développement de solutions de configuration efficaces et automatisées. Les méthodes de configuration manuelles sont lentes et sujettes aux erreurs, ce qui entraîne des inefficacités dans la gestion de réseau. Pour résoudre ce problème, ce mémoire de master se concentre sur la mise en œuvre d'un système d'optimisation réseau a l'aide de l'automatisation qui vise à mettre en place un système de communication entre des protocoles réseau distinct. Le projet vise aussi à rationaliser le processus d'administration de réseau en fournissant une interface graphique conviviale pour configurer les VLAN, DHCP et récupérer les différents états des protocoles STP et HSRP. En exploitant l'automatisation et les interfaces intuitives, l'application proposée maximise l'efficacité du réseau, minimise les erreurs et réduira le besoin d'intervention humaine dans les tâches de configuration de réseau.

Mots-clés : Réseaux informatiques, Configuration automatique, Complexité, Gestion de réseau, Interface graphique, VLAN, HSRP, Automatisation.

ملخص

يتطلب التعقيد المتزايد لشبكات الحاسوب بسبب التقدم التكنولوجي السريع تطوير حلول تشكيلية فعالة وآلية. أساليب التكوين اليدوي بطيئة وعرضة للأخطاء، مما يؤدي إلى عدم الكفاءة في إدارة الشبكة. لحل هذه المشكلة، تركز هذه الذاكرة الرئيسية على تنفيذ نظام تحسين الشبكة بمساعدة الأتمتة التي تهدف إلى إنشاء نظام اتصال بين بروتوكولات الشبكة المنفصلة. يهدف المشروع أيضًا إلى تبسيط عملية إدارة الشبكة من خلال توفير واجهة رسومية سهلة الاستخدام لتكوين VLANs و DHCP واسترداد الحالات المختلفة لبروتوكولات STP و HSRP. من خلال الاستفادة من الأتمتة والواجهات البديهية، يزيد التطبيق المقترح من كفاءة الشبكة، ويقلل من الأخطاء ويقلل من الحاجة إلى التدخل البشري في مهام تكوين الشبكة.

الكلمات الرئيسية: شبكات الكمبيوتر، التكوين التلقائي، التعقيد، إدارة الشبكة، الواجهة الرسومية، VLAN، HSRP، الأتمتة.

Table des matières

Table des figures

Liste des tableaux

Introduction Générale

Les réseaux informatiques jouent un rôle essentiel dans les infrastructures technologiques d'aujourd'hui. Avec l'évolution rapide des avancées technologiques, les réseaux deviennent de plus en plus complexes, ce qui nécessite une gestion et une configuration efficaces pour assurer leur bon fonctionnement. Cependant, les méthodes traditionnelles de configuration manuelle des réseaux se révèlent souvent lentes et sujettes à des erreurs.

Dans ce contexte, l'automatisation apparaît comme une solution technologique pertinente pour répondre à la complexité croissante des réseaux. Elle permet de réduire considérablement la dépendance à l'intervention humaine, ce qui maximise l'efficacité du réseau. De plus, elle offre une grande flexibilité dans la gestion des réseaux en évitant les tâches fastidieuses associées aux configurations manuelles via les entrées en ligne de commande.

Notre projet se situe dans ce contexte précis, avec pour objectif principal d'examiner l'automatisation dans le domaine des réseaux et de sélectionner l'outil le mieux adapté. Ce choix nous permettra de proposer une solution pour résoudre un problème important qui est l'établissement d'une connexion entre des protocoles réseau distincts, afin d'améliorer les performances et l'efficacité.

Ainsi, ce projet vise à mettre en œuvre un système d'optimisation réseau en utilisant l'automatisation. Ce système sera équipé d'une interface graphique conviviale qui facilitera l'administration du réseau et apportera une assistance précieuse dans la configuration de différents éléments, tels que les VLANs. De plus, il permettra à l'utilisateur de visualiser l'état des interfaces du protocole STP sur chaque commutateur, ainsi que du protocole HSRP sur les commutateurs de distribution. Le système offrira également la possibilité de configurer plusieurs tâches, notamment la création de VLANs sur l'ensemble du réseau. L'utilisateur aura la possibilité de sélectionner un commutateur et de le configurer pour fonctionner en tant que serveur DHCP.

En résumé, l'objectif de ce projet est de répondre au défi posé par la complexité croissante des réseaux informatiques et les solutions de configuration manuelle traditionnelles, en proposant un système d'optimisation réseau basé sur l'automatisation. Ce système fournira aux administrateurs réseau une interface conviviale et des fonctionnalités avancées pour simplifier et accélérer la configuration des réseaux informatique, tout en réduisant les risques d'erreurs humaines. Ainsi, il permettra de combler le fossé entre la complexité des réseaux actuels et les méthodes de configuration manuelle, en offrant une approche plus efficace et fiable pour gérer et optimiser les performances du réseau.

Ce mémoire est organisé en quatre chapitres, suivant la structure suivante :

- Chapitre 1 : Nous soulignerons les différents protocoles et technologies présents dans l'univers des réseaux contemporains.
- Chapitre 2 : Nous aborderons l'introduction de l'automatisation du réseau ainsi que les divers outils associés.
- Chapitre 3 : Nous examinerons en détail la conception de notre réseau, la problématique qui nous a incités à entreprendre ce projet, et enfin, nous présenterons notre solution envisagée.
- Chapitre 4 : Nous exposerons les outils et les différentes configurations que nous avons mis en place pour concrétiser notre conception.

Chapitre I : Présentation générale des réseaux informatiques

I.1 Introduction

Les réseaux informatiques sont devenus indispensables pour le bon fonctionnement des entreprises modernes, car ils permettent le partage de ressources et de données, la gestion des performances et la collaboration et la communication efficaces. Dans ce chapitre, nous aborderons les différents aspects des réseaux informatiques, tels que les protocoles et les technologies de mise en œuvre.

I.2 Typologie des réseaux informatiques privés

On peut classifier les réseaux informatiques privés en fonction de leur taille (c'est-à-dire le nombre de machines), leur vitesse de transfert et leur étendue. La classification générale est basée sur l'étendue et comprend quatre types de réseaux : les réseaux personnels (PAN), les réseaux locaux (LAN), les réseaux métropolitains (MAN) et les réseaux étendus (WAN).

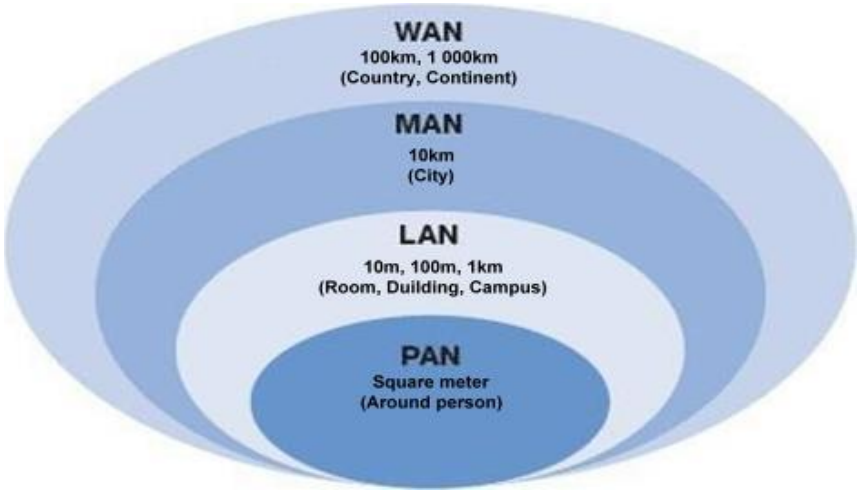


Figure I.1 – Types des réseaux informatique privés. [1]

I.3 L'Architecture Hiérarchique de Réseau :

Dans le domaine des réseaux, la conception hiérarchique divise le réseau en couches distinctes. Chaque couche, ou niveau, de la hiérarchie offre des fonctions spécifiques qui définissent son rôle dans le réseau. Ceci permet au concepteur et à l'architecte du réseau de sélectionner le matériel et les logiciels réseau adaptés, ainsi que les fonctionnalités nécessaires aux rôles de cette couche réseau. Les modèles hiérarchiques appliquent à la fois les conceptions LAN et WAN.

L'architecture hiérarchique de réseau se compose de trois couches principales :

I.3.1 Couche d'accès :

Cette couche est la plus basse de l'architecture hiérarchique et comprend les périphériques réseau auxquels les utilisateurs finaux se connectent, tels que les commutateurs d'accès et les points d'accès sans fil.

Les commutateurs d'accès fournissent des connexions aux périphériques finaux (par exemple, les ordinateurs, les imprimantes) et segmentent généralement le réseau en différents VLAN (Virtual Local Area Network) pour des raisons de sécurité et de gestion du trafic.

Les points d'accès sans fil permettent aux utilisateurs de se connecter au réseau via des connexions Wi-Fi.

I.3.2 Couche de distribution :

Cette couche assure l'agrégation des connexions des commutateurs d'accès et fournit une connectivité aux différents segments du réseau.

Les commutateurs de distribution jouent un rôle clé dans la couche de distribution et fournissent une connectivité aux commutateurs d'accès. Ils peuvent également effectuer des opérations de filtrage, de contrôle de bande passante et de routage inter-VLAN.

À cette couche, on peut également trouver des services réseau tels que les serveurs DHCP (Dynamic Host Configuration Protocol) pour l'attribution automatique des adresses IP.

I.3.3 Couche cœur :

Cette couche est la plus haute de l'architecture hiérarchique et assure la connectivité entre les différents segments du réseau.

Les commutateurs cœurs sont responsables de l'acheminement efficace du trafic entre les différents commutateurs de distribution et permettent la connectivité entre les sites distants ou les réseaux externes.

Ils utilisent des protocoles de routage dynamique pour établir les chemins optimaux du trafic dans le réseau et peuvent également implémenter des fonctionnalités avancées de sécurité et de qualité de service.

Cette conception d'architecture hiérarchique de réseau est basée sur des principes généraux, mais les détails spécifiques peuvent varier en fonction des besoins et des contraintes de chaque réseau. [2]

Les réseaux hiérarchiques présentent plusieurs avantages importants, notamment :

- **Évolutivité :** L'architecture hiérarchique permet une évolutivité efficace en séparant le réseau en couches distinctes. Cela permet d'ajouter de nouveaux périphériques ou segments de réseau sans perturber l'ensemble du réseau. L'évolutivité est essentielle pour répondre aux besoins croissants en matière de connectivité et de capacité du réseau.

- **Gestion simplifiée :** La structure hiérarchique facilite la gestion du réseau en fournissant des points de contrôle et de configuration clairs à chaque couche. La séparation des tâches entre les couches permet de simplifier les opérations courantes telles que la configuration, la surveillance et le dépannage. Cela permet également de limiter l'impact des modifications ou des problèmes à une couche spécifique sans affecter l'ensemble du réseau. [3]
- **Performances optimisées :** L'architecture hiérarchique permet d'optimiser les performances du réseau. Les commutateurs de distribution agissent comme des points d'agrégation pour le trafic provenant des commutateurs d'accès, ce qui réduit la charge sur les liens montants vers les commutateurs cœurs. Cela permet une utilisation efficace de la bande passante et réduit les goulots d'étranglement potentiels.
- **Sécurité renforcée :** L'architecture hiérarchique facilite la mise en place de politiques de sécurité cohérentes. Les commutateurs de distribution peuvent être configurés pour appliquer des règles de sécurité spécifiques à chaque VLAN, contrôlant ainsi l'accès et la communication entre les différents segments du réseau. La séparation des VLAN contribue également à limiter la propagation des attaques potentielles. [3]
- **Fiabilité et redondance :** Les réseaux hiérarchiques permettent la mise en place de redondance pour améliorer la fiabilité du réseau. Les commutateurs de distribution peuvent être interconnectés par plusieurs chemins pour assurer une redondance des liens. De plus, les protocoles de routage dynamique utilisés au niveau des commutateurs cœurs permettent de rétablir automatiquement les chemins de communication en cas de panne. [3]
- **Isolation des problèmes :** L'architecture hiérarchique facilite l'identification et l'isolation des problèmes de réseau. Grâce à la séparation des couches et à la localisation précise des équipements, il est plus facile de déterminer où se situe un problème spécifique et de le résoudre rapidement sans perturber le reste du réseau.

En résumé, les réseaux hiérarchiques offrent une évolutivité, une gestion simplifiée, des performances optimisées, une sécurité renforcée, une fiabilité accrue et une isolation des problèmes. Ces avantages en font une approche couramment utilisée dans la conception de réseaux d'entreprise. [3]

I.4 La technologie VLAN

Les VLAN sont un outil permettant de subdiviser un réseau physique en plusieurs domaines de diffusion logiques distincts, ce qui permet de segmenter le trafic pour des raisons de performances, de sécurité ou de logistique. Les VLAN sont identifiés par un ID de VLAN et chaque port d'un commutateur ou d'un routeur peut être affecté pour être membre d'un VLAN. Ainsi, le trafic envoyé à un port membre d'un VLAN particulier peut être transféré vers n'importe quel autre port appartenant au même VLAN sur le même commutateur ou sur un autre commutateur via des connexions de type Trunk. Toutefois, le trafic ne sera pas transféré vers des ports qui appartiennent à un VLAN différent.

Les VLAN permettent aux administrateurs réseau de diviser logiquement un commutateur, tout en conservant les avantages d'isolement, de sécurité et de performances de l'utilisation de commutateurs complètement séparés. Pour permettre la communication entre les VLAN, le

routage de couche 3 est nécessaire, tout comme dans le cas de sous-réseaux différents sur différents commutateurs. Certains commutateurs de couche 3 prennent en charge le routage entre les VLAN, ce qui peut améliorer les performances en évitant d'envoyer du trafic via le routeur.[4]

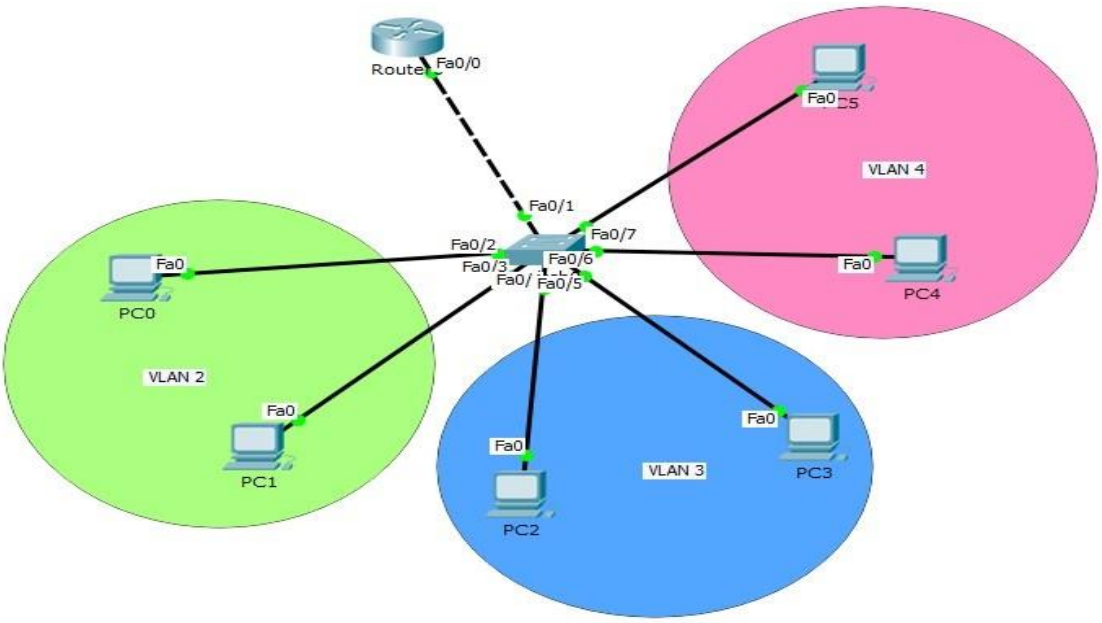


Figure I.2 – Exemple de la topologie Vlan.

Cependant, Il existe différents types de VLANs, notamment :

- VLAN de gestion : Il s'agit d'un VLAN distinct configuré pour le trafic de gestion, comme la journalisation système/application, la surveillance et d'autres tâches sensibles liées à la gestion. Quand on voit les avantages en dehors de la sécurité, la bande passante est élevée lorsque le nombre d'utilisateurs gêner le trafic en même temps. [5]

- VLAN de données : Le VLAN de données est également appelé VLAN utilisateur car il est conçu uniquement pour les données générées par l'utilisateur. Le réseau peut être conçu sur la base d'un groupe d'utilisateurs ou de groupes de travail. Prenons l'exemple d'un institut, le groupe de travail du réseau serait configuré en fonction des départements. De même, cela peut se produire dans toutes les unités commerciales et lorsque nous mettons en place ce réseau, nous devons passer du temps à comprendre le paysage pour mieux regrouper les utilisateurs. [5]

- VLAN voix : L'organisation utilise la voix sur IP (VoIP) à l'aide d'un VLAN vocal distinct. Il préservera la bande passante des autres applications et devra garantir la qualité de la voix.

- VLAN par défaut : Le VLAN par défaut peut être désigné sous deux types. Le premier fait référence à tous les ports de l'appareil et appartient à un seul lorsque le commutateur est activé. Sur le second se réfère, un gestionnaire de réseau est configuré avec le VLAN basé sur tous les ports sont attribués même lorsque le commutateur n'est pas utilisé.

- VLAN natif : Le VLAN natif est celui que le trafic non étiqueté accepter lorsqu'il est reçu sur le port Trunk. Ceci est le plus souvent utilisé pour les appareils hérités ou non

pris en charge qui n'ont pas été marqués. Il s'agit probablement d'un point d'accès sans fil au réseau. [5]

I.5 Les protocoles réseau

Les protocoles réseau sont essentiels pour permettre la communication entre les appareils connectés dans les réseaux informatiques. Certains des protocoles couramment utilisés incluent IP, ICMP, TCP et DHCP, qui sont fondamentaux pour le bon fonctionnement des réseaux modernes.

I.5.1 Le protocole IP (Internet Protocol)

Internet Protocol (IP) est un protocole de couche réseau (couche 3) qui contient des informations d'adressage et certaines informations de contrôle permettant le routage des paquets. IP est documenté dans la RFC 791 [6] et est le principal protocole de couche réseau dans la suite de protocoles Internet. Avec TCP, IP représente le cœur des protocoles Internet. IP a deux responsabilités principales : fournir une livraison sans connexion et au mieux des datagrammes via un interréseau, et la fourniture d'une fragmentation et d'un réassemblage de datagrammes pour prendre en charge des liaisons de données avec différentes tailles d'unité de transmission maximale (MTU). [7]

I.5.2 Le protocole ICMP (Internet Control Message Protocol)

L'ICMP (Internet Control Message Protocol) est un protocole de signalement d'erreurs que les appareils de réseau comme les routeurs utilisent pour générer des messages d'erreur à l'adresse IP source lorsque des problèmes de réseau empêchent la livraison de paquets IP. L'Internet Control Message Protocol crée et envoie des messages à l'adresse qui IP source indiquant qu'une passerelle vers l'internet, qu'un routeur, qui un service ou un hôte ne peut pas être atteint pour la livraison de paquets. Tout dispositif de réseau IP a la capacité d'envoyer, de recevoir ou de traiter des messages ICMP. [8]

I.5.3 Le protocole TCP (Transmission Control Protocol)

Transmission Control Protocol (TCP) [9] est une norme de communication qui permet aux programmes d'application et aux dispositifs informatiques d'échanger des messages sur un réseau. Il est conçu pour envoyer des paquets sur Internet et assurer la livraison réussie des données et des messages sur les réseaux.

TCP est l'une des normes de base qui définissent les règles d'Internet et fait partie des normes définies par l'Internet Engineering Task Force (IETF) [10]. C'est l'un des protocoles les plus couramment utilisés dans les communications de réseau numérique et il assure la livraison des données de bout en bout.

TCP organise les données afin qu'elles puissent être transmises entre un serveur et un client. Il garantit l'intégrité des données communiquées sur un réseau. Avant de transmettre des données, TCP établit une connexion entre une source et sa destination, dont il s'assure qu'elle reste active jusqu'au début de la communication. Il divise ensuite de grandes quantités de

données en paquets plus petits, tout en garantissant l'intégrité des données tout au long du processus. [11]

I.5.4 Le protocole DHCP (Dynamic Host Configuration Protocol)

DHCP signifie Dynamic Host Configuration Protocol. Il s'agit d'un protocole qui permet à un ordinateur qui se connecte sur un réseau local d'obtenir dynamiquement et automatiquement sa configuration IP. Le but principal étant la simplification de l'administration d'un réseau. On voit généralement le protocole DHCP comme distribuant des adresses IP, mais il a été conçu au départ comme complément au protocole BOOTP (Bootstrap Protocol) qui est utilisé par exemple lorsque l'on installe une machine à travers un réseau (on peut effectivement installer complètement un ordinateur, et c'est beaucoup plus rapide que de le faire en à la main). Cette dernière possibilité est très intéressante pour la maintenance de gros parcs machines. Les versions actuelles des serveurs DHCP fonctionnent pour IPv4 (adresses IP sur 4 octets). Une spécification pour IPv6 (adresses IP sur 16 octets) est en cours de développement par l'IETF. [12]

Le schéma ci-dessous (Figure 1.3) illustre le processus de configuration dynamique des adresses IP à l'aide du protocole DHCP. Il comprend les étapes de diffusion (DISCOVER), proposition (OFFER), demande (REQUEST) et réponse (ACK/NACK).

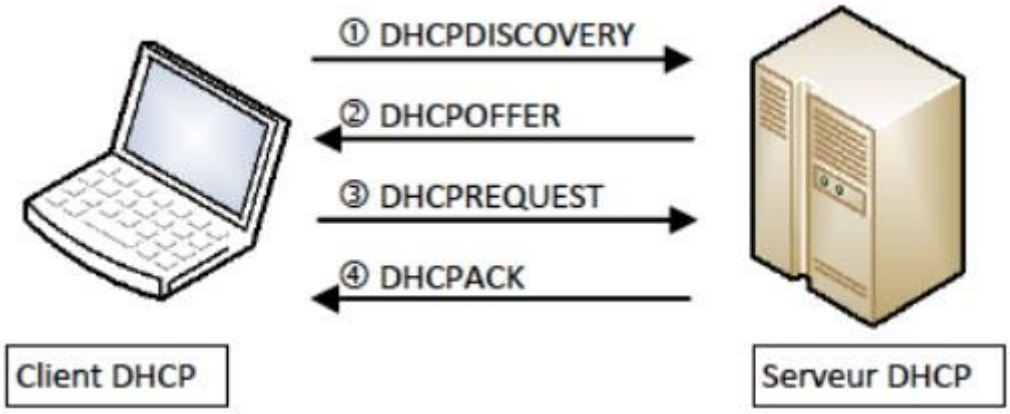


Figure I.3 – Phases d'attribution de l'adresse IP. [13]

I.6 Protocole de routage

Un protocole qui facilite l'échange d'informations de routage entre les routeurs, supportant un protocole routé. Les messages de protocole de routage sont transmis entre les routeurs, leur permettant d'échanger des informations entre eux pour mettre à jour et maintenir les tables. Plusieurs protocoles de routage TCP/IP sont disponibles, notamment :

- Routing Information Protocol (RIP)

- Interior Gateway Routing Protocol (IGRP)

- Enhanced Interior Gateway Routing Protocol (EIGRP)

- Open Shortest Path First (OSPF)

Le schéma ci-dessous (Figure I.4) représente le flux de données dans un réseau, en mettant l'accent sur les couches 1, 2 et 3 du modèle OSI. Il illustre le trajet des données après leur transmission par l'hôte émetteur et avant leur arrivée à l'hôte récepteur.

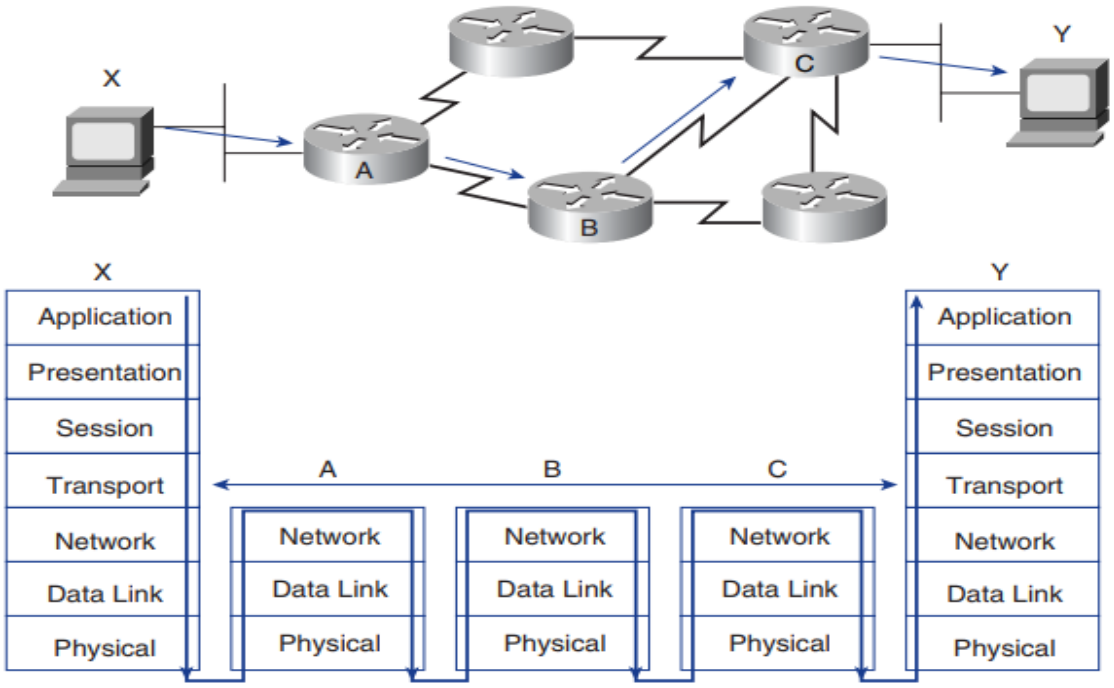


Figure I.4 – Services de routeur. [14]

I.6.1 Le protocole Routing Information Protocol (RIP)

Le protocole d'information de routage (RIP) est un protocole de routage a vecteur de distance qui envoie la table de routage complète hors de toutes les interfaces actives toutes les 30 secondes. Son principe de fonctionnement consiste à compter les sauts pour déterminer la route optimale vers un réseau distant, avec un nombre de sauts maximum autorisé de 15, rendant toute destination au-delà de cette inaccessible. Sur les petits réseaux, RIP fonctionne correctement, mais s'avère inefficace sur les grands réseaux avec des liaisons WAN lentes et une abondance de routeurs installés. En outre, RIP est inutile sur les réseaux avec des liens de bande passante variés. RIP version 1 est restrictive dans la mesure où il ne prend en charge le routage classful. Par conséquent, tous les périphériques du réseau doivent utiliser le même masque de sous-réseau. RIP version 1 n'inclut pas les informations de masque de sous-réseau dans les mises à jour qu'il envoie. Cependant, RIP version 2 offre le routage préfixe, transmettant des informations de masque de sous-réseau avec des mises à jour de route réseau. Cette version de protocole fournit ce qu'on appelle le routage sans classe. [15]

I.6.2 Le protocole Interior Gateway Routing Protocol (IGRP)

Au milieu des années 1980, un protocole alternatif au protocole d'information de routage version 1 (RIPv1) appelé Interior Gateway Routing Protocol (IGRP) a été développé par Cisco. L'objectif principal était d'éliminer les limites de la mesure du nombre de sauts (hops)et du diamètre du réseau du RIP, qui ont été jugés rudimentaires. L'IGRP a utilisé une métrique

composite plus complexe comprenant diverses variables de route et a même fourni la flexibilité de peser des variables spécifiques plus lourdement que d'autres pour répondre aux particularités des différents réseaux.

Comparé au RIP, le IGRP présentait de nombreux avantages, notamment un diamètre de réseau plus large pouvant atteindre 255 sauts (hops), une métrique multivariée plus complexe, un partage de charge inégal, une période de mise à jour de 90 secondes, trois fois plus longue que le RIP, et un format de paquet de mise à jour plus efficace. IGRP a été conçu pour interagir avec plusieurs protocoles routés tels que IPv4, ISO Connectionless Network Protocol (CLNP), Novell IPX ou AppleTalk. Comme pour le RIP, IGRP a envoyé un paquet Request à partir de toutes les interfaces compatibles IGRP au démarrage et a effectué des vérifications de contrôle sur les paquets Update reçus pour s'assurer que l'adresse source appartenait au même sous-réseau que le paquet Update. La transmission périodique des paquets de mise à jour a eu lieu toutes les 90 secondes. [16]

I.6.3 Le protocole Enhanced Interior Gateway Routing Protocol (EIGRP)

Le protocole EIGRP (Enhanced Interior Gateway Routing Protocol) de Cisco a été développé pour surmonter les limites de son prédécesseur, IGRP. EIGRP est un protocole de routage vecteur de distance qui présente des améliorations significatives par rapport à IGRP, dans la mesure où il est plus qu'une version "améliorée". Alors que IGRP s'est appuyé sur les minuteries et a annoncé toute sa base de données de réseaux connus à chaque expiration d'intervalle de mise à jour, EIGRP sépare le processus de construction et de maintien des adjacences routeur d'échanger des informations de routage. Pour ce faire, le EIGRP utilise un protocole Hello pour établir et maintenir les adjacences voisines, similaire à Open Shortest Path First (OSPF) et d'autres protocoles. Le protocole Hello prend en charge l'envoi périodique de paquets Update pour annoncer la présence continue d'un routeur sur un réseau. EIGRP utilise également le Reliable Transport Protocol (RTP) pour annoncer les informations de routage en cas de changement, sans nécessiter de publicité périodique. RTP permet aux routeurs d'échanger initialement des informations de routage complètes lors de la synchronisation pour la première fois et d'annoncer uniquement les itinéraires modifiés par la suite. L'utilisation de Hellos pour établir et maintenir les adjacencies de routeur et RTP pour transporter toutes les mises à jour permet EIGRP de fonctionner de manière fiable et progressive.[17]

I.6.4 Le protocole Open Shortest Path First (OSPF)

Open Shortest Path First (OSPF) est un IGP (Interior Gateway Protocol) de type routeur intérieur ou frontalier utilisé pour le routage d'état de liaison. Il annonce l'état des liens entre les périphériques réseau et actualise sa base de données d'état de liaison (LSDB) chaque fois qu'un routeur commence à fonctionner sur une liaison réseau. L'OSPF envoie des messages Hello sur ses liens opérationnels pour déterminer si d'autres routeurs Link State fonctionnent également sur les interfaces. Contrairement au RIP, l'OSPF n'a pas de limites de sauts sur un trajet de transmission ; il utilise un algorithme plus complexe pour déterminer le meilleur chemin. Chaque routeur exécutant OSPF maintient une base de données des liens des autres routeurs et utilise cette information pour déterminer les trajectoires alternatives en cas de défaillance de liaison. L'OSPF permet une convergence rapide et empêche les boucles de routage, c'est pourquoi il est couramment utilisé sur les systèmes autonomes qui dépendent d'un mélange de

routeurs de divers fabricants. OSPF fonctionne directement sur IP en utilisant le protocole IP numéro 89 et est capable de coexister avec le RIP (ou RIPv2) sur un réseau. [18]

I.7 Le protocole STP (Spanning Tree)

Une bonne conception de réseau offre une redondance des périphériques et déliaisons réseau afin d'éliminer les risques de pannes du réseau provoquée par un composant unique et offrir d'avantage, une haute disponibilité et une tolérance aux pannes. La solution la plus simple consiste à ajouter une deuxième liaison entre les commutateurs pour surmonter une panne de liaison réseau ou s'assurer qu'un commutateur est connecté à au moins deux autres commutateurs dans une topologie. Cependant, de telles topologies posent des problèmes lorsqu'un commutateur doit transmettre des diffusions ou lorsqu'une inondation monodiffusion inconnue se produit. Le réseau diffuse en boucle continue jusqu'à ce que le lien devienne saturé.

Le schéma ci-dessous (Figure I.5) présente un scénario de liaison redondante où, par exemple, lorsqu'un poste (P1) envoie une trame de diffusion, celle-ci est diffusée sur tous les ports de SW2, atteignant ainsi SW2 (en bleu) et SW3 (en vert). De même, SW2 rediffuse la trame vers PC2 et SW3, puis SW3 vers PC3 et SW1, créant ainsi une boucle. Elle peut même se poursuivre (en orange).

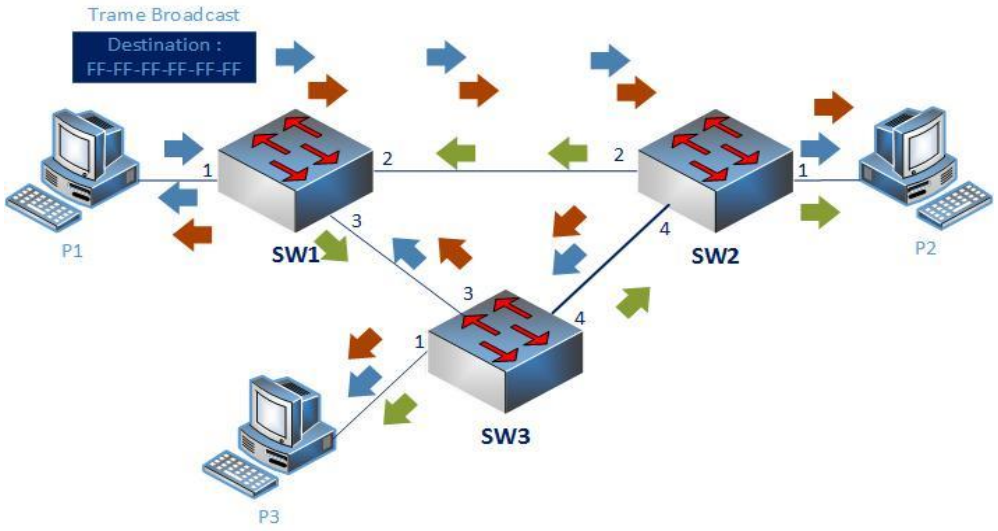


Figure I.5 – Sans STP, La Boucle Provoquera une Tempête de Broadcast. [19]

Le protocole Spanning Tree (STP) est utilisé pour créer un réseau sans boucle en surveillant le réseau pour suivre tous les liens et fermer les moins redondants par la désactivions automatiquement des ports qui génèrent des boucles.

Définition

Le protocole Spanning Tree (STP) est un protocole de réseau informatique qui permet de gérer les boucles dans les réseaux locaux (LAN). Il a été créé par Radia Perlman en 1985 pour éviter les boucles d'interconnexion de commutateurs et de ponts (bridges) dans les réseaux Ethernet.

Le STP utilise un algorithme qui permet de désigner un chemin unique pour chaque paquet de données à travers le réseau, en bloquant certains ports pour éviter les boucles. Si un lien entre

deux commutateurs ou ponts tombe en panne, le STP réorganise le réseau en temps réel pour rétablir la connectivité.

Le protocole Spanning Tree est normalisé par l'IEEE sous la norme 802.1D. Depuis sa création, de nombreuses variantes ont été développées pour améliorer les performances et la résilience du protocole, comme Rapid Spanning Tree Protocol (RSTP) et Multiple Spanning Tree Protocol (MSTP). [20]

Fonctionnement du protocole STP

1) **Sélection d'un commutateur Root :** un seul par topologie, qui sera le commutateur racine de la topologie, tous ses ports transfèrent le trafic (ports Designated). Le commutateur avec l'identifiant "Bridge ID" (BID) le plus faible remporte l'élection.
2) **Sélection d'un seul port Root :** sur les (autres) commutateurs non-Root, qui dispose de la liaison dont le coût vers le commutateur Root est le plus faible. Il est le seul à transférer du trafic.
3) **Sélection d'un port Designated :** pour chaque segment physique qui connecte deux commutateurs quand c'est nécessaire. C'est le port qui a le coût vers le commutateur Root le plus faible qui est sélectionné, il est le seul à transférer le trafic.
4) **Sélection du port Bloqué :** Les ports Root et Designated transfèrent du trafic (état "Forwarding") et les autres ports coupent la liaison (état "Blocking"). [21]

La Figure ci-dessous illustre le fonctionnement du protocole Spanning-Tree :

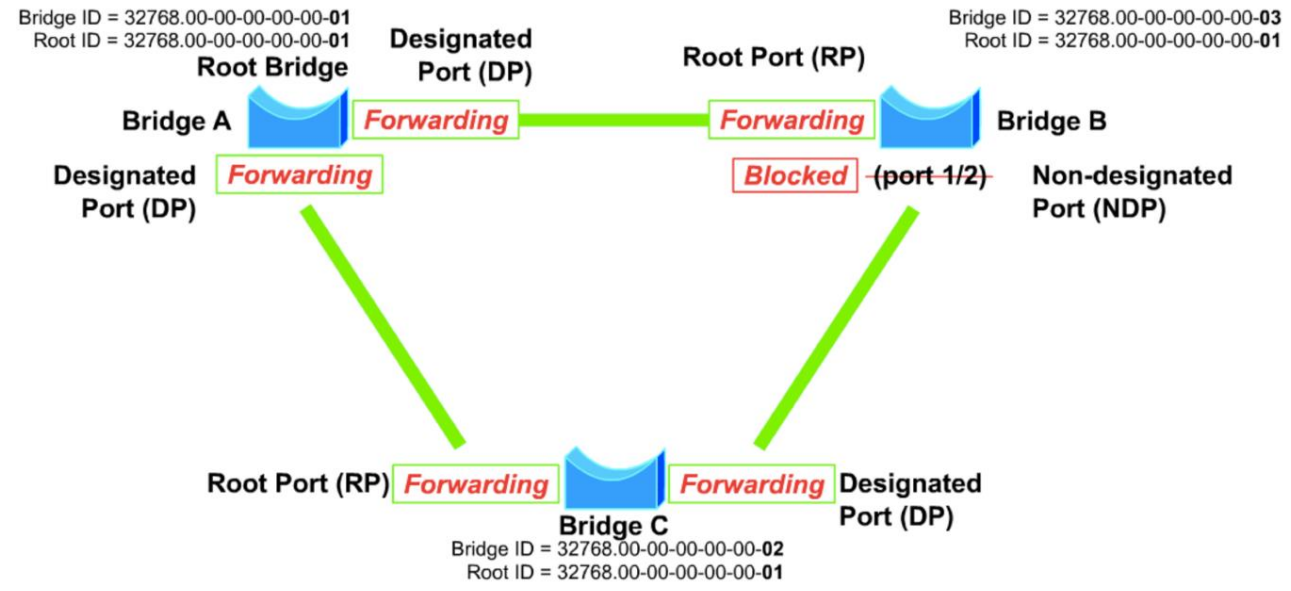


Figure I.6 – Exemple de Fonctionnement du Spanning Tree. [22]

Variantes STP

Plusieurs variétés de protocoles Spanning Tree ont émergé depuis l'IEEE 802.1D [23] d'origine. Les variétés de protocoles Spanning-Tree comprennent les éléments suivants :

- **RSTP :** également connu sous le nom de IEEE 802.1w, est une variante du protocole Spanning Tree Protocol (STP) utilisée pour éviter les boucles dans les réseaux locaux (LAN). RSTP a été développé pour améliorer les performances et la réactivité du protocole STP, en réduisant le temps de convergence après une défaillance de liaison et en optimisant l'utilisation de la bande passante. [24]

- **PVST et PVST+ :** Per-VLAN Spanning Tree (PVST) est un protocole propriétaire de Cisco qui permet à un appareil Cisco d'avoir plusieurs arbres couvrants, le périphérique Cisco peut interagir avec les arbres couvrants sur d'autres périphériques PVST mais ne peut pas interagir avec les périphériques IEEE 802.1Q. un périphérique IEEE 802.1Q a tous ses ports exécutant un seul arbre couvrant. PVST+ est une extension de PVST qui permet à un appareil Cisco d'interagir également avec des appareils exécutant un seul arbre couvrant (IEEE 802.1Q). [25]

- **MSTP :** également connu sous le nom de IEEE 802.1s, est une variante du protocole Spanning Tree Protocol (STP) utilisée pour éviter les boucles dans les réseaux locaux (LAN). MSTP a été développé pour améliorer les performances et la scalabilité du protocole STP, en permettant à plusieurs instances de STP de coexister sur un même réseau.

MSTP utilise une méthode plus efficace pour réduire les temps de convergence et de propagation dans le réseau, en permettant de regrouper plusieurs VLANs en une seule instance de STP. Cela permet de réduire la charge de calcul et de transmission sur les commutateurs et d'améliorer les performances globales du réseau. [26]

Le tableau I.1 présente une comparaison entre les différentes variantes du protocole Spanning Tree (STP, RSTP, MSTP, PVST). Il met en évidence les différences en termes d'efficacité, de scalabilité, de complexité, de vitesse de convergence, du nombre d'instances de protocole par VLAN et des rôles de port. Ce tableau offre une vue d'ensemble concise des caractéristiques distinctives de chaque protocole.

Comparaison entre les protocoles Spanning Tree :

Caractéristiques	**STP**	**RSTP**	**MSTP**	**PVST**
Efficacité	Faible	Moyenne	Élevée	Moyenne
Scalabilité	Faible	Moyenne	Élevée	Faible à Moyenne
Complexité	Faible	Moyenne	Élevée	Élevée
Convergence	Lente	Rapide	Rapide	Moyenne
Nombre d'instances de protocole par VLAN	1	1	Multiple	N/A (1 instance par VLAN)
Port Roles	3 (Root Port, Designated Port, Blocked Port)	3 (Root Port, Designated Port, Alternate Port)	3 (Root Port, Designated Port, Alternate Port)	3 (Root Port, Designated Port, Blocked Port)

Table I.1 – Comparaison entre les variantes du protocole STP.[27]

PortFast

PortFast est une fonctionnalité de Cisco Spanning Tree Protocol (STP) qui permet à un port sur un commutateur d'aller directement de l'état de blocage à l'état de transmission sans passer par l'état de transition, ce qui accélère la convergence du réseau. Sans PortFast, les ports d'un commutateur prennent du temps à entrer dans l'état de transmission après la mise en place de la connexion, car STP doit d'abord passer par une série d'états de transition pour vérifier la topologie du réseau. [28]

I.8 Le Protocole FHRP (First Hop Redundancy Protocol)

Les protocoles FHRP permettent d'établir une redondance pour la passerelle par défaut en utilisant plusieurs passerelles qui agissent comme une seule passerelle virtuelle. Les ordinateurs du réseau envoient leurs paquets à cette passerelle virtuelle, qui redirige ensuite les paquets vers leur destination finale.

Les protocoles FHRP les plus couramment utilisés sont HSRP (Hot Standby Router Protocol), VRRP (Virtual Router Redundancy Protocol) et GLBP (Gateway Load Balancing Protocol). Chacun de ces protocoles fonctionne de manière légèrement différente, mais l'objectif est le même : offrir une haute disponibilité pour la passerelle par défaut. [29]

HSRP (Hot Standby Router Protocol)

Le protocole HSRP (Hot Standby Routing Protocol) est un protocole propriétaire de Cisco qui fournit un basculement transparent du périphérique de premier saut, qui agit généralement comme une passerelle vers les hôtes. HSRP fournit une redondance de routage pour les hôtes IP sur un réseau Ethernet configuré avec une adresse IP de passerelle par défaut. Un minimum de deux appareils est requis pour l'activer : un appareil agit en tant qu'appareil actif et s'occupe de la transmission des paquets, et l'autre agit en tant qu'appareil de secours prêt à prendre le rôle d'appareil actif en cas de panne. Sur un segment de réseau, une adresse IP virtuelle est configurée sur chaque interface compatible HSRP appartenant au même groupe. HSRP sélectionne l'une des interfaces pour agir en tant que routeur actif. En plus de l'adresse IP virtuelle, une adresse MAC virtuelle est attribuée au groupe. Le routeur actif reçoit et acheminées paquets destinés à l'adresse MAC virtuelle du groupe. Lorsque le routeur actif tombe en panne, le routeur de secours prend le contrôle de l'adresse IP virtuelle étude l'adresse MAC virtuelle du groupe. L'élection HSRP sélectionne le routeur avec la priorité la plus élevée (la valeur par défaut est 100). En cas d'égalité de priorité, le routeur avec l'adresse IP la plus élevée pour le segment de réseau est préféré. [29]

VRRP (Virtual Router Redundancy Protocol)

Le protocole VRRP (Virtual Router Redundancy Protocol) est un protocole standard ouvert et fonctionne de manière similaire à HSRP. Il est utilisé pour assurer la redondance dans un réseau. VRRP est un protocole de couche réseau. Il utilisée concept de routeur maître (Master) et de secours (Standby), lorsque le routeur maitre tombe en panne, l'un des routeurs de secours assumera les responsabilités du routeur maître, c'est-à-dire que le routeur de secours sera responsable de la transmission du trafic jusqu'à ce que le routeur maître arrive de nouveau. [30]

GLBP (Gateway Load Balancing Protocol)

Le protocole GLBP (Gateway Load Balancing Protocol) fournit une redondance de passerelle et une capacité d'équilibrage de charge à un segment de réseau. Il offre une redondance avec une passerelle Active/Standby et offre une capacité d'équilibrage de charge en garantissant que chaque membre du groupe GLBP prend soin de transférer le trafic vers la passerelle appropriée. Le GLBP assume deux rôles:

- **Active virtuel gateway (AVG) :** Les routeurs participants élisent un AVGpar groupe GLBP pour répondre aux requêtes ARP initiales du VIP.

- **Active Virtual forwarder (AVF) :** L'AVF achemine le trafic reçu des hôtes affectés. Une adresse MAC virtuelle unique est créée et attribuée par l'AVGaux AVF. L'AVF est attribué à un hôte lorsque l'AVG répond à la requête ARP avec l'adresse MAC virtuelle de l'AVF attribuée. Les réponses ARP sont unicast et ne sont pas entendues par les autres hôtes sur ce segment de diffusion.
  Lorsqu'un hôte envoie du trafic au MAC AVF virtuel, le routeur actuel est chargé de l'acheminer vers le réseau approprié. Les AVF sont également reconnus comme des instances de transfert sur les routeurs.
  GLBP prend en charge quatre AVF actifs et un AVG par groupe GLBP. UN routeur peut être un AVG et un AVF en même temps. En cas de panne d'AVG, il n'y a pas d'interruption du trafic en raison du transfert du rôle AVG vers un appareil AVG de secours. En cas de défaillance d'un AVF, un autre routeur prend en charge les responsabilités de transfert pour cet AVF, qui inclut l'adresse MAC virtuelle de cette instance. [29]

I.9 La Technologie Etherchannel (Aggregation Des Liens)

EtherChannel est une technologie de Cisco qui permet de regrouper plusieurs ports physiques d'un commutateur en une seule connexion logique. Cette connexion logique utilise une seule adresse MAC et une seule adresse IP, ce qui facilite la gestion et la configuration du réseau.

L'EtherChannel permet d'augmenter la bande passante, la redondance et la tolérance aux pannes en agrégeant plusieurs connexions physiques en une seule connexion logique.

Il existe deux modes de fonctionnement pour l'EtherChannel : le mode LACP (Link Aggregation Control Protocol) et le mode PAgP (Port Aggregation Protocol). Ces deux protocoles permettent de négocier la création d'un EtherChannel entre les commutateurs connectés. [31]

LACP (Link Aggregation Control Protocol)

Le protocole LACP permet de faciliter la configuration des liens agrégés entre les commutateurs en négociant automatiquement la création et la configuration de l'agrégation de liens. Lorsque LACP est activé sur deux ports Ethernet reliés entre deux commutateurs, ces derniers négocient automatiquement la création d'un lien agrégé en fonction des paramètres de configuration définis.

L'agrégation de liens LACP peut être configurée selon différents modes de fonctionnement, tels que le mode actif/passif, le mode dynamique ou encore le mode statique. Le mode actif/passif permet d'activer un port de liaison agrégée pour envoyer des paquets alors que

l'autre port est utilisé en tant que backup. Dans le mode dynamique, les ports peuvent être activés ou désactivés en fonction de la charge du réseau. Enfin, dans le mode statique, les ports sont activés de manière permanente et ne peuvent être désactivés que manuellement. [31]

PAGP (Port Aggregation Protocol)

PAgP est un protocole propriétaire de Cisco utilisé pour la création de liens agrégés entre deux commutateurs. Comme LACP, ce protocole permet de combiner plusieurs connexions Ethernet physiques en une seule liaison logique de haute capacité et de haute disponibilité. [31]

PAgP fonctionne en mode actif/passif. Lorsqu'un port Ethernet est configuré en mode actif, il envoie des requêtes PAgP aux autres ports Ethernet reliés au même commutateur ou à un commutateur distant. Lorsqu'un port est configuré en mode passif, il n'envoie pas de requêtes PAgP, mais peut répondre aux requêtes envoyées par un port en mode actif.

Lorsqu'un port Ethernet est configuré en mode actif, il envoie des messages PAgP pour négocier la création d'un lien agrégé avec les ports Ethernet voisins. Les ports voisins répondent également avec des messages PAgP pour indiquer leur état de disponibilité et leur mode de fonctionnement. [31]

En fonction de ces messages PAgP, le commutateur peut créer automatiquement un lien agrégé entre les ports disponibles. Si l'un des ports agrégés tombe en panne, PAgP peut détecter la défaillance et mettre automatiquement le port en mode inactif, permettant ainsi une redondance et une haute disponibilité. [31]

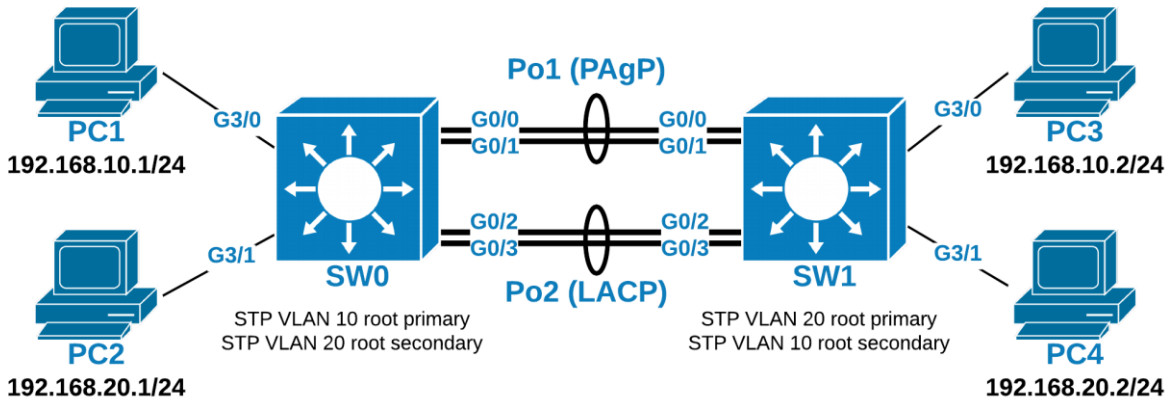


Figure I.7 – Exemple d'Etherchannel. [32]

I.10 Le Protocole SSH (Secure Shell)

Le protocole SSH (Secure Shell) est un protocole de communication sécurisé utilisé pour accéder à des équipements réseau ou à des ordinateurs distants. Contrairement à Telnet, qui transmet les données en clair, SSH utilise une connexion cryptée pour garantir la confidentialité et la sécurité des données. [33]

Le protocole SSH fonctionne en utilisant une paire de clés cryptographiques publique/privée pour authentifier l'utilisateur et pour établir une session cryptée entre l'utilisateur et l'hôte distant. Une fois la connexion établie, l'utilisateur peut entrer des commandes ou des données

à partir de son terminal local qui seront transmises de manière cryptée à l'hôte distant. Les résultats de ces commandes ou données seront également renvoyés de manière cryptée à l'utilisateur via la connexion SSH.

SSH est largement utilisé pour accéder à des équipements réseau tels que des routeurs, des commutateurs ou des serveurs, ainsi que pour se connecter à des ordinateurs distants pour accéder à des applications ou des services. Il est considéré comme plus sûr que Telnet car il garantit la confidentialité et la sécurité des données transmises entre l'utilisateur et l'hôte distant. [33]

I.11 Conclusion

Avec les avancées technologiques dans le monde des réseaux, la configuration manuelle de ces derniers devient de plus en plus complexe et laborieuse. Les réseaux modernes nécessitent désormais la configuration de multiples protocoles fondamentaux tels que les protocoles de redondance et de routage. Les tâches manuelles quotidiennes telles que les mises à jour et les dépannages sont coûteux en temps et en ressources humaines, et peuvent entraîner des erreurs humaines à l'origine de 80% des pannes réseau. Face à ces problèmes, l'automatisation de la configuration du réseau est une solution idéale pour éliminer les problèmes liés à la configuration manuelle et augmenter l'efficacité du réseau tout en réduisant les coûts.

Chapitre II : Automatisation du réseau

II.1 Introduction

De nos jours, de nombreuses entreprises adoptent l'automatisation de leurs réseaux, car elle présente de nombreux avantages, notamment en ce qui concerne l'établissement d'une connexion entre des protocoles réseau distincts, ce qui est essentiel pour assurer un fonctionnement fluide et efficace. Dans cette section, nous aborderons l'automatisation du réseau, ses améliorations et les outils utilisés pour y parvenir.

II.2 Définition de l’automatisation du réseau

L'automatisation du réseau est le processus d'application de technologies et de processus de gestion automatisés pour simplifier, accélérer et rationaliser la gestion des réseaux informatiques. Les technologies d'automatisation du réseau permettent d'effectuer des tâches telles que la configuration, la surveillance et la résolution des problèmes sans intervention humaine, ce qui réduit les erreurs et améliore l'efficacité opérationnelle. [34]

II.3 Importance de l’automatisation du réseau

L'automatisation du réseau est devenue un élément clé pour les entreprises souhaitant rester compétitives. Elle permet de répondre rapidement aux besoins des clients et aux changements du marché en fournissant des services réseau agiles, fiables et sécurisés. En automatisant les tâches courantes, telles que la configuration, la surveillance et la gestion des politiques de sécurité, les équipes informatiques peuvent se concentrer sur des tâches plus stratégiques et à plus haute valeur ajoutée, ce qui améliore l'efficacité opérationnelle et réduit les coûts. De plus, l'automatisation du réseau permet de minimiser les risques d'erreurs humaines et de renforcer la sécurité du réseau en garantissant la cohérence et la conformité des configurations. [35]

Les principaux avantages de l'automatisation du réseau :

Amélioration de l'efficacité opérationnelle : L'automatisation du réseau permet de simplifier et de rationaliser les tâches de gestion du réseau, ce qui permet de gagner du temps et de l'argent. Selon une étude de 2018 réalisée par Gartner, "l'automatisation du réseau peut réduire le temps nécessaire pour déployer des services de 40 % à 90 %". [36]

Permet la communication entre protocoles : Chaque protocole réseau est conçu principalement pour opérer individuellement, alors que parfois la communication entre les différents protocoles est requise pour que le réseau fonctionne de façon optimale. L’automatisation fournit le privilège de pouvoir consulter et analyser les états des différents protocoles ce qui permet d’ajuster la configuration qui assure le meilleur fonctionnement possible du réseau.

Réduction des erreurs humaines : L'automatisation du réseau permet de minimiser les risques d'erreurs humaines, qui peuvent causer des temps d'arrêt et des problèmes de sécurité

importants. Selon une étude de 2018 réalisée par le Ponemon Institute, "les erreurs humaines sont la cause de 22 % des temps d'arrêt non planifiés". [36]

Amélioration de la sécurité : L'automatisation du réseau permet de garantir la cohérence et la conformité des configurations de sécurité, ce qui améliore la sécurité du réseau. Selon une étude de 2019 réalisée par Fortinet, "l'automatisation du réseau peut aider les entreprises à améliorer la conformité aux politiques de sécurité de 60 %". [38]

Réduction des coûts : L'automatisation du réseau permet de réduire les coûts d'exploitation en minimisant le temps et les ressources nécessaires pour gérer le réseau. Selon une étude de 2020 réalisée par IDC, "l'automatisation du réseau peut réduire les coûts d'exploitation de 30 % à 50 %". [39]

II.4 Outils d'automatisation du réseau

L'automatisation du réseau utilise des outils conçus pour simplifier la configuration des ressources d'infrastructure distribuées, ce qui permet une rapidité d'exécution tout en garantissant la fiabilité et la conformité. Ces outils se divisent généralement en deux catégories : ceux avec des agents et ceux sans agents. Les outils basés sur des agents impliquent l'installation d'un logiciel de monitoring sur chaque périphérique de l'architecture réseau. Les agents collectent ensuite des données à partir de différentes sources telles que les API, les appels système et les fichiers logs. Les informations collectées sont ensuite envoyées à un serveur de monitoring central qui compile les résultats, vérifie les avertissements et les alarmes, et fournit les résultats aux utilisateurs.

Cependant, les environnements sans agent utilisent un seul agent de gestionnaire de configuration pour surveiller l'ensemble de l'infrastructure. Par conséquent, le terme « sans agent » peut être un peu trompeur. Le serveur central monitor les périphériques d'infrastructure via des API, SSH, et des interfaces système pour déterminer les performances et la disponibilité. Cet agent peut être utilisé pour gérer les opérations du serveur sur l'ensemble de votre réseau. [40]

Il existe plusieurs applications que les administrateurs réseau peuvent utiliser pour gérer différents équipements, mais les outils les plus répandus dans ce contexte sont: Puppet, Chef et Ansible.

II.4.1 Puppet

Puppet est un outil open-source d'automatisation du réseau qui permet la gestion centralisée des configurations des systèmes et des applications. Il utilise une approche déclarative pour spécifier les états désirés des systèmes et des applications, plutôt que de définir explicitement les actions à exécuter.

Le fonctionnement de Puppet se base sur un modèle client-serveur, où un agent Puppet est installé sur chaque système cible et communique avec un serveur Puppet pour récupérer et appliquer les configurations requises. Les configurations peuvent être écrites en utilisant le langage de description de ressources de Puppet, qui permet de spécifier les propriétés et les états souhaités pour chaque ressource (fichier, utilisateur, service, etc.).

Dans la figure II.1, le processus commence avec le logiciel Agent qui fournit des faits au serveur de marionnettes (1). Le Serveur de Marionnettes distribue ensuite le Catalogue, qui contient les Modèles de Marionnettes, à l'Agent de Marionnettes (2). Le Puppet Agent signale son état au serveur primaire (3) de manière continue. Enfin, lorsque le catalogue correspond au modèle Puppet, le flux de travail Agent passe à l'état de rapport (4), qui fonctionne comme un état de surveillance du système.

Le schéma ci-dessous montre comment fonctionne l'architecture serveur-agent d'une exécution Puppet.

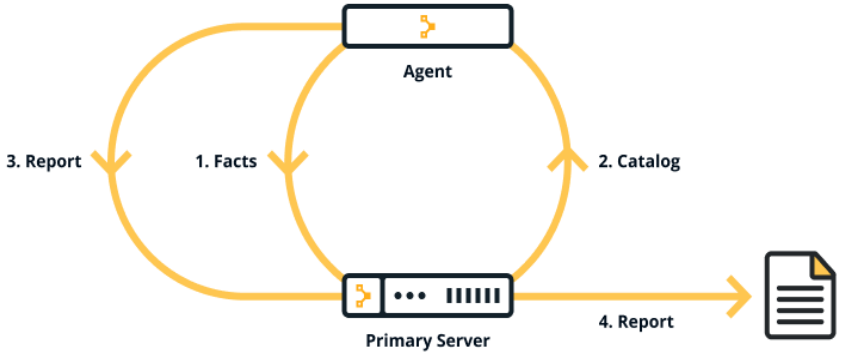


Figure II. 1 – Fonctionnement Puppet. [41]

Les avantages de l'utilisation de Puppet incluent :

- Automatisation des tâches répétitives et fastidieuses de configuration et de maintenance des systèmes.
- Gestion centralisée des configurations, ce qui facilite la mise à jour et la cohérence des configurations sur l'ensemble des systèmes cibles
- Suivi de l'état des systèmes et des applications, ce qui permet de détecter rapidement les écarts par rapport aux configurations requises.
- Flexibilité et extensibilité grâce à une large bibliothèque de modules préconstruits pour la gestion de différents types de ressources. [42]

Cependant, Puppet peut causer des inconvénients, tels que :

- Apprentissage initial nécessaire pour comprendre le langage de description de ressources de Puppet.
- Complexité accrue pour la configuration initiale de l'infrastructure et la mise en place du modèle client-serveur.
- Possibilité de ralentissement du système lors de l'application de configurations importantes.
- En l'absence d'un système push, il n'y a pas d'action immédiate sur les changements, le processus d'extraction suit un calendrier spécifié pour les tâches. [42]

II.4.2 Chef

Il s'agit d'un outil de gestion de configuration open source conçu pour l'automatisation de la configuration et la gestion des réseaux et des serveurs. Un agent doit être installé sur les périphériques réseau que le chef gère. Le client de Chef extraites configurations du serveur, et ces configurations sont en Ruby DSL (Domain Specific Language). Ruby est le langage de programmation/configuration utilisé dans Chef. Le serveur chef est l'appareil qui gère tous les nœuds du réseau, tandis que le client chef est le nœud que le serveur chef gère. [43]

La figure ci-dessous illustre les composants de Chef :

Figure II. 2 – Composant de Chef. [44]

La Figure II.2 présente Chef Infra, une plateforme d'automatisation qui transforme l'infrastructure en code. Le schéma illustre le processus de développement, de test et de déploiement du code Chef Infra. Il met en évidence les rôles de Chef Workstation, Chef Infra Server et Chef Infra Client. Chef Workstation permet aux utilisateurs de créer et de tester des cookbooks, tandis que Chef Infra Client configure les systèmes selon l'état souhaité. Chef Infra Server agit comme un hub centralisé pour les données de configuration, stockant les cookbooks et gérant les métadonnées des nœuds. Le schéma démontre l'interaction transparente entre ces composants pour automatiser la gestion de l'infrastructure.

Toutefois, Chef, en tant qu'outil d'automatisation, présente des avantages et des inconvénients, comme tout autre outil. Voici quelques-uns de ces avantages et inconvénients :

Les avantages :

- Le langage Chef Infra permet de définir précisément et en détail l'état souhaité des systèmes et applications.
- Les Cookbooks peuvent être partagés et réutilisés, ce qui permet de gagner du temps et de l'argent lors de la configuration de l'infrastructure informatique.
- Chef peut être utilisé dans des environnements divers, ce qui le rend flexible et adaptable à différents besoins et exigences. [45]

Les inconvénients :

- L'apprentissage du langage Chef Infra peut être difficile pour les non-programmeurs.
- L'installation et la configuration peuvent être complexes, surtout pour les installations distribuées à grande échelle.
- Les coûts liés à l'utilisation de Chef peuvent être élevés, surtout pour les organisations de petite à moyenne taille. [45]

II.4.3 Ansible

Il s'agit d'une plate-forme d'automatisation capable de déployer des applications, de gérer la configuration et le monitoring, de gérer la sécurité et d'automatiser le provisionnement et le déploiement du cloud. Il fonctionne avec le langage de programmation Python et le format de données YAML. Ansible est un outil sans agent, ce qui signifie qu'aucun logiciel ou agent ne doit être installé sur les machines clientes. Le fait d'être sans agent permet à l'utilisateur de pusher la configuration vers l'un des périphériques réseau du réseau.

Ansible utilise SSH comme protocole de communication à distance et peut également prendre en charge la gestion à distance de Windows. Il utilise des modèles push pour obtenir les configurations des périphériques réseau. Le modèle push signifie qu'aucun logiciel agent n'est installé sur les nœuds. Nous pouvons gérer n'importe quel appareil en utilisant Ansible. Comme il est sans agent, n'importe quel appareil peut être un contrôleur Ansible sur le réseau. [43]

La figure ci-dessous illustre l'architecture d'Ansible :

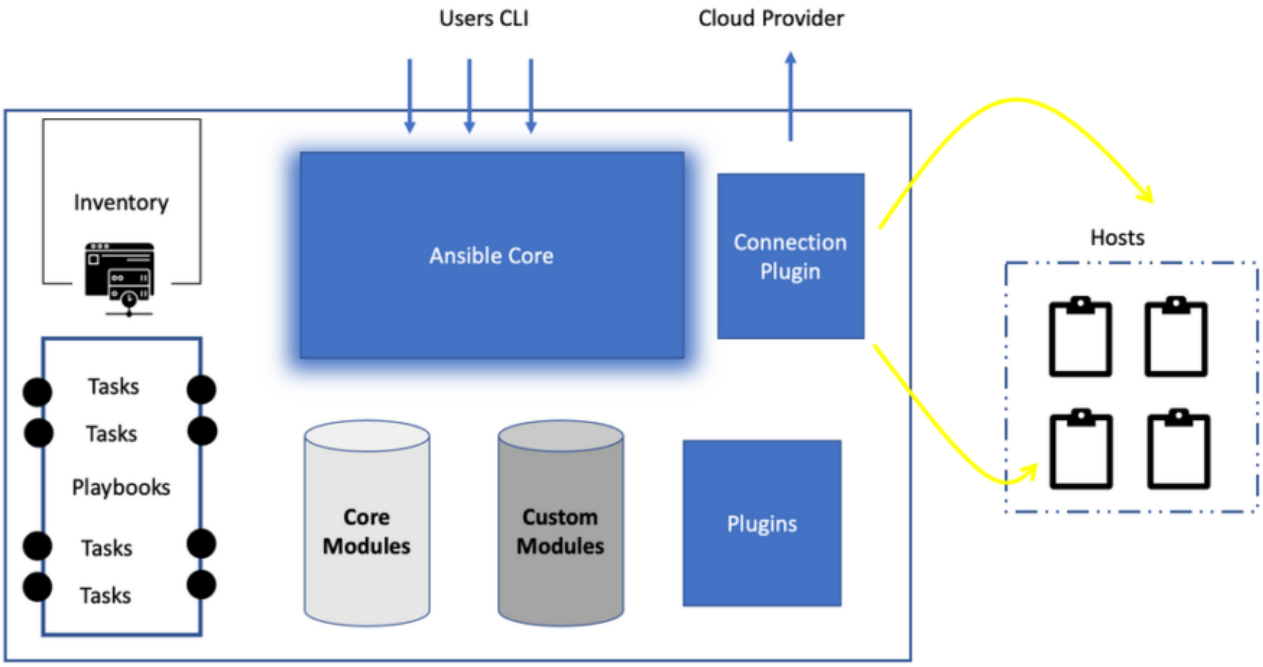


Figure II. 3 – Composant et architecture Ansible. [46]

Les avantages d'Ansible sont nombreux. Parmi eux, on peut citer :

- **Facilité d'utilisation :** Ansible est facile à apprendre et à utiliser. Il utilise un langage de description simple qui permet aux utilisateurs de définir des configurations et des tâches de manière efficace.

- **Automatisation complète :** Ansible peut automatiser l'ensemble du cycle de vie de l'infrastructure, de la configuration initiale à la gestion des changements et des mises à jour.

- **Sécurité :** Ansible utilise le protocole SSH pour la communication et ne nécessite pas d'installation d'agents sur les machines distantes, ce qui permet de garantir la sécurité du système.

- **Grande communauté :** Ansible dispose d'une grande communauté d'utilisateurs et de contributeurs, ce qui permet d'accéder à une vaste gamme de modules et de ressources.

Cependant, Ansible présente également quelques inconvénients, tels que :

- **Limitations de performance :** bien qu'Ansible soit rapide, il peut être moins performant que certains outils d'automatisation plus spécialisés pour les tâches les plus intensives en ressources.

- **Limitations de complexité :** Ansible est moins adapté aux tâches les plus complexes qui nécessitent une programmation avancée.

- **Documentation limitée :** bien qu'Ansible dispose d'une documentation complète, elle peut parfois être difficile à naviguer en raison de sa complexité. [47]

Puppet vs Chef vs Ansible – Quelles sont les différences ?

La différence distinctive entre Ansible, Chef et Puppet est le script sous-jacent. Alors que Puppet est codé en Ruby, Ansible utilise des scripts Python et YAML, et Chef est construit sur Ruby comme Puppet. Cependant, dans les Puppet Vs. Chef bout, Chef est différent de Puppet car il est programmé sur Ruby combiné avec Domain Spécifique Langage. Les autres différences entre Chef Vs. Puppet Vs. Ansible est décrit en fonction de différents facteurs, notamment la disponibilité, la langue de configuration, la configuration et l'installation, la facilité de gestion, l'évolutivité, l'interopérabilité.

- **Disponibilité**

Outil DevOps	Disponibilité (en cas de panne de serveur)
Puppet	Maître alternatif
Chef	Serveur de sauvegarde
Ansible	Instance secondaire

Table II. 1 – Disponibilité des outils DevOps dans les scénarios de défaillance de serveur. [47]

- **Langue de Configuration**

Outils DevOps	Langue de config	Convient a	Niveau d'apprentissage
Puppet	Ruby, Puppet DSL, ERB, DSL	Administrateurs Système	Difficile
Chef	Ruby DSL	Développeurs	Difficile
Ansible	Python, YAML	Administrateurs Système	Simple

Table II. 2 – Langage de configuration et difficultés d'apprentissage pour les outils DevOps. [47]

- **Configuration et installation**

Outil DevOps	Architecture	Facilité de configuration et installation
Puppet	Maitre-Agent	Difficile en raison de la signature du certificat entre le maitre et l'agent
Chef	Maitre-agent	Difficile et complexe en raison de chef Workstation
Ansible	Maitre uniquement (sans agent)	Facile

Table II. 3 – Configuration et facilité d'installation des outils DevOps. [47]

- **Facilité de gestion**

La gestion des services d'automatisation DevOps dépend du langage et de la configuration des outils. Il existe deux types de configurations, y compris 'pull' et 'push'. La configuration Pull implique de tirer toutes les configurations d'un serveur central vers les nœuds esclaves sans aucune commande. Alors que dans une configuration push, toutes les configurations du serveur seront poussées vers les nœuds avec des commandes spécifiques.

En termes de langage de configuration, YAML est considéré comme le plus simple car il est similaire à l'anglais et est lisible par l'homme. Alors que les langages Puppet DSL et Ruby DSL créent des revers pour la gestion.

Une fois de plus, Ansible dépeint sa domination sur les autres en termes de gestion car il prend en charge le langage YAML et suit les configurations push et pull. [47]

Outil DevOps	Configuration	Facilité de Gestion
Puppet	Tirer	Difficile
Chef	Tirer	Difficile
Ansible	Pousser et tirer	Facile

Table II. 4 – Configuration Gestion de la configuration et facilité d'administration pour les outils DevOps. [47]

- **Evolutivité**

Outil DevOps	Evolutivité
Puppet	Haut
Chef	Haut
Ansible	Très Haut

Table II. 5 – Évolutivité des outils DevOps. [47]

- **Interopérabilité**

Outil DevOps	Interopérabilité
Puppet	Puppet Master Doit être sous linux/Unix ; L'agent ou le client Puppet prend en charge Windows
Chef	Chef server doit être sous Linux/Unix ; Workstation et chef Client prennent en charge Windows
Ansible	Ansible Server doit être sous Linux/Unix ; les machines clientes prennent en charge Windows

Table II. 6 – Interopérabilité des outils DevOps avec les systèmes d'exploitation. [47]

II.5 Préférer Ansible à Puppet et Chef

Ansible Automation Platform fournit aux administrateurs système, aux opérateurs et aux décideurs informatiques tous les outils nécessaires pour mettre en œuvre l'automatisation à l'échelle de l'entreprise. Avec Ansible, vous pouvez automatiser la plupart des tâches informatiques, telles que la gestion de la configuration, le déploiement d'applications, l'orchestration, le provisionnement et la sécurité. Et ce qui distingue le plus Ansible, c'est qu'il est aussi puissant que facile à utiliser. Avec Ansible, nous pouvons simplifier la gestion de structures extrêmement complexes plus rapidement qu'avec tout autre outil de gestion de configuration. [48]

II.6 Présentation générale d'Ansible

Ansible est un outil open-source de provisionnement de logiciels, de gestion des configurations et de déploiement d'applications qui permet de faire coder une infrastructure IT en ce compris le support de ses applications. Il permet d'automatiser la plupart des tâches de gestion d'infrastructures. Il fonctionne sur de nombreux systèmes de type Unix, et il peut configurer aussi bien des systèmes de type Unix que Microsoft Windows ou autres. Le logiciel Ansible a été conçu par un ancien employé RedHat, Michael DeHaan. Le code source du logiciel est sous licence GNU General Public v3.0. RedHat a racheté la société Ansible, Inc. en octobre 2015. [49]

II.6.1 Les Composants d'Ansible

Les composants d'Ansible s'associent pour former un système de gestion très puissant. Comprendre chaque composant et les relations entre les composants est essentiel pour réaliser leur puissance :

- **Nœud de contrôle** : La machine à partir de laquelle vous exécutez les outils Ansible CLI (**ansible-playbook**, **ansible** et **ansible-vault** autres). Vous pouvez utiliser n'importe quel ordinateur qui répond aux exigences logicielles comme nœud de contrôle - les ordinateurs portables, les bureaux partagés et les serveurs peuvent tous exécuter Ansible. Plusieurs nœuds de contrôle sont possibles, mais Ansible lui-même ne se coordonne pas entre eux. [49]

- **Nœuds gérés :** Également appelés "hôtes", il s'agit des périphériques cibles (serveurs, appareils réseau ou tout ordinateur) que vous souhaitez gérer avec Ansible. Ansible n'est normalement pas installé sur les nœuds gérés, sauf si vous utilisez Ansible-pull, mais cela est rare et n'est pas la configuration recommandée. [49]

- **Inventaire :** Une liste de nœuds gérés fournie par une ou plusieurs « sources d'inventaire ». Votre inventaire peut spécifier des informations spécifiques à chaque nœud, comme l'adresse IP. Il est également utilisé pour affecter des groupes, qui permettent tous deux la sélection de nœuds dans la lecture et l'affectation de variables en masse. Parfois, un fichier source d'inventaire est également appelé "fichier hôte". [49]

- **Playbooks :** Ils contiennent des lectures (qui sont l'unité de base de l'exécution d'Ansible). Il s'agit à la fois d'un « concept d'exécution » et de la manière dont nous décrivons les fichiers sur lesquels **ansible-playbook** opère. Les playbooks sont écrits en YAML et sont faciles à lire, écrire, partager et comprendre. [49]

- **Plays :** Contexte principal de l'exécution d'Ansible, cet objet de playbook mappe les nœuds gérés (hôtes) aux tâches. Le play contient des variables, des rôles et une liste ordonnée de tâches et peut être exécuté à plusieurs reprises. Il consiste essentiellement en une boucle implicite sur les hôtes et les tâches mappés et définit comment les parcourir. [49]

- **Roles :** Une distribution limitée de contenu Ansible réutilisable (tâches, gestionnaires, variables, plugins, modèles et fichiers) à utiliser dans un Play. Pour utiliser une ressource de rôle, le rôle lui-même doit être importé dans le Play. [49]

- **Tasks :** La définition d'une 'action' à appliquer à l'hôte géré. Les tâches doivent toujours être contenues dans un Play, directement ou indirectement (Rôle, ou fichier de liste de tâches importé/inclus). Vous pouvez exécuter une seule tâche une seule fois avec une commande ad hoc en utilisant **ansible** ou **ansible-console** (les deux créent un jeu virtuel). [49]

- **Handlers :** Une forme spéciale d'une tâche, qui ne s'exécute que lorsqu'elle est notifiée par une tâche précédente qui a entraîné un statut "modifié". [49]

- **Modules :** Le code ou les fichiers binaires qu'Ansible copie et exécute sur chaque nœud géré (si nécessaire) pour accomplir l'action définie dans chaque tâche. Chaque module a une utilisation particulière, de l'administration des utilisateurs sur un type spécifique de base de données à la gestion des interfaces VLAN sur un type spécifique de périphérique réseau. Vous pouvez invoquer un seul module avec une tâche ou invoquer plusieurs modules différents dans un Playbook. Les modules Ansible sont regroupés en collections. Pour avoir une idée du nombre de collections incluses dans Ansible. [49]

- **Plugins :** Morceaux de code qui étendent les capacités de base d'Ansible, ils peuvent contrôler la façon dont vous vous connectez à un nœud géré (plugins de connexion), manipuler les données (plugins de filtrage) et même contrôler ce qui est affiché dans la console (plugins de rappel). Voir Travailler avec des plugins pour plus de détails. [49]

- **Collections :** Un format dans lequel le contenu Ansible est distribué et qui peut contenir des Playbooks, des rôles, des modules et des plugins. Vous pouvez installer et utiliser des collections via Ansible Galaxy. Les ressources de la collection peuvent être utilisées indépendamment et discrètement les unes des autres. [49]

- **PAA :** Abréviation de 'Ansible Automation Platform'. Il s'agit d'un produit qui inclut des fonctionnalités de niveau entreprise et intègre de nombreux outils de l'écosystème Ansible : ansible-core, awx, galaxyNG, etc. [49]

II.6.2 Terminologie d'Ansible

Les termes les plus utilisés d'Ansible sont indiqués dans le tableau suivant :

Terme	Description
Playbook	un fichier YAML qui décrit un ensemble de tâches à exécuter sur un ensemble de machines distantes.
Tâches	une instruction à exécuter sur une machine distante, telle que l'installation d'un package ou la création d'un fichier.
Module	une unité de code qui peut être utilisée dans une tâche pour effectuer une action sur un appareil géré.
Inventory	un fichier ou une source de données qui répertorie les machines distantes à gérer par Ansible.
Play	une exécution d'un Playbook sur un ensemble de machines distantes.
Rôles	un ensemble de fichiers, de variables et de tâches organisés de manière cohérente et réutilisable, qui peut être inclus dans un Playbook pour simplifier sa gestion et sa maintenance.
Handler	une tâche qui est déclenchée lorsqu'un événement spécifique se produit, telle que la relance d'un service après une modification de configuration.

Table II. 7 – Terminologie Ansible. [49]

II.6.3 Un Playbook Ansible

Les Playbooks Ansible offrent un système de gestion de configuration et de déploiement multi-machine reproductible, réutilisable et simple, bien adapté au déploiement d'applications complexes. Si vous devez exécuter plusieurs fois une tâche avec Ansible, écrivez un Playbooks et mettez-le sous contrôle de code source. Ensuite, vous pouvez utiliser le Playbooks pour publier une nouvelle configuration ou confirmer la configuration des systèmes distants. [49]

Les Playbooks sont exprimés au format YAML avec un minimum de syntaxe. Etant donné que YAML est un fichier en texte brut.

Un Playbook est composé d'un ou de plusieurs "plays" dans une liste ordonnée. Chaque play exécute une partie de l'objectif global du playbook, exécutant une ou plusieurs tâches. Chaque tâche appelle un module Ansible. [49]

Un Playbook s'exécute dans l'ordre de haut en bas. Dans chaque play, les tâches s'exécutent également dans l'ordre de haut en bas. Les Playbooks avec plusieurs “plays" peuvent orchestrer des déploiements multi-machines, en exécutant un play sur vos serveurs Web, puis un autre play sur vos serveurs de base de données, puis un troisième play sur votre infrastructure réseau, et ainsi de suite. Au minimum, chaque jeu définit deux choses :

- Les nœuds gérés à cibler.
- Au moins une tâche à exécuter. [49]

La figure ci-dessous présente un exemple d’un Playbook Ansible :

```
---
- name: Afficher l'etats des interface des switchs
  hosts: Switches
  gather_facts: no

  tasks:
   - name: Show ip interface brief
     ios_command:
       commands:
       - show ip interface brief

     register: config

   - name: save output
     copy:
       content: "{{ config.stdout | replace('\\n', '\n') }}"
       dest: "interface_Switches.txt"
```

Figure II. 4 – Exemple d’un Playbook Ansible.

II.6.4 Exécution de la tâche

Par défaut, Ansible exécute chaque tâche dans l'ordre, une à la fois, sur toutes les machines correspondant au modèle d'hôte. Chaque tâche exécute un module avec des arguments

spécifiques. Lorsqu'une tâche s'est exécutée sur toutes les machines cibles, Ansible passe à la tâche suivante. Vous pouvez utiliser des stratégies pour modifier ce comportement par défaut. Dans chaque jeu, Ansible applique les mêmes directives de tâche à tous les hôtes. Si une tâche échoue sur un hôte, Ansible retire cet hôte de la rotation pour le reste du Playbook. [49]

Lorsque vous exécutez un Playbook, Ansible renvoie des informations sur les connexions, les namelignes de toutes vos parties et tâches, si chaque tâche a réussi ou échoué sur chaque machine, et si chaque tâche a apporté une modification sur chaque machine. Au bas de l'exécution du Playbook, Ansible fournit un résumé des nœuds ciblés et de leurs performances. Les échecs généraux et les tentatives de communication fatales "injoignables" sont séparés dans les décomptes.

Ansible peut fonctionner en parallèle sur un maximum de cinq hôtes. Si vous ajoutez d'autres hôtes, vous devez attendre plus longtemps pour que chaque tâche se termine. Vous pouvez modifier ce comportement en définissant les forks sur 20 dans la section de niveau supérieur du Playbook. [49]

Dans cet exemple, le Playbook est configuré pour s'exécuter sur les hôtes du groupe "**routers**". La tâche utilise le module « **ios_command**" pour exécuter la commande "**show ip interface brief**" sur chaque périphérique réseau. Les résultats de la commande sont stockés dans la variable "**ip_interfaces**".

Ensuite, une autre tâche "debug" est utilisée pour afficher les résultats de la commande. Dans cet exemple, nous affichons seulement la première ligne de sortie en utilisant "ip_interfaces.stdout_lines[0]". [49]

II.6.5 Fichier d'inventaire

Le fichier d'inventaire dans Ansible est un fichier qui répertorie les hôtes cibles sur lesquels les actions et les tâches doivent être exécutées. Il peut être utilisé pour regrouper les hôtes en fonction de leur rôle, de leur emplacement ou de tout autre critère pertinent. Le fichier d'inventaire peut être écrit au format INI ou YAML, selon la préférence de l'utilisateur. [50]

La figure ci-dessous présente un exemple de fichier d'inventaire :

```
[switches]
switch1 ansible_host=192.168.1.10
switch2 ansible_host=192.168.1.11

[routers]
router1 ansible_host=192.168.1.20
router2 ansible_host=192.168.1.21

[servers]
server1 ansible_host=192.168.1.100
server2 ansible_host=192.168.1.101

[all:vars]
ansible_user=admin
ansible_password=secretpassword
```

Figure II. 5 – Exemple d'un fichier inventaire.

Dans cet exemple, nous avons trois groupes d'hôtes : "**switches**", "**routers**" et "**servers**". Chaque groupe contient des hôtes spécifiques avec leurs adresses IP respectives. Par exemple, le groupe "switches" contient deux switches avec les adresses IP 192.168.1.10 et 192.168.1.11.

Le groupe "[all:vars]" définit des variables globales qui s'appliquent à tous les hôtes. Dans cet exemple, nous avons défini les variables "ansible_user" et "ansible_password" pour spécifier les informations d'identification utilisées pour se connecter aux hôtes.

Cet inventaire peut être utilisé avec des Playbooks Ansible pour exécuter des tâches spécifiques sur des groupes d'hôtes ou sur tous les hôtes.

II.6.6 Exécution d'un Playbook

L'exécution d'un Playbook Ansible se fait à l'aide de la commande Ansible Playbook suivie du nom du Playbook que vous souhaitez exécuter. Voici la syntaxe générale :
ansible-playbook playbook.yml

La figure présente un exemple d'exécution d'une playbook Ansible :

```
PLAY [network_devices] *********************************************************************************

TASK [gather_facts] ************************************************************************************
ok: [switch1.example.com]
ok: [switch2.example.com]

TASK [configure_interface] *****************************************************************************
changed: [switch1.example.com]
changed: [switch2.example.com]

PLAY RECAP *********************************************************************************************
switch1.example.com        : ok=2   changed=1   unreachable=0   failed=0   skipped=0   rescued=0   ignored=0
switch2.example.com        : ok=2   changed=1   unreachable=0   failed=0   skipped=0   rescued=0   ignored=0
```

Figure II. 6 – Exemple d'exécution d'une Playbook.

Dans cet exemple, le Playbook a été exécuté sur les équipements réseau '**switch1.example.com'** et '**switch2.example.com**'. Les tâches exécutées peuvent inclure la collecte des informations (**facts**) sur les équipements réseau et la configuration des interfaces.

La désactivation de **Gather_fact** est possible en définissant sa valeur sur faux la tache **Configure_Interface** s'exécute en premier.

Chaque tâche exécutée est répertoriée avec des informations sur son statut, indiquant si elle a été modifiée (**changed**) ou si elle a réussi (**ok**). Le nombre total de tâches exécutées, le nombre de tâches modifiées, le nombre d'équipements inaccessibles ou en échec, etc., sont récapitulés dans la section "**PLAY RECAP**".

Le résultat d'exécution dépendra des modules Ansible utilisés pour interagir avec l'équipement réseau, tels que **ios_command**, **nxos_command, ios_config**, etc. Ces modules vous permettent d'envoyer des commandes, de configurer des paramètres et d'effectuer d'autres opérations sur les équipements réseau. [48]

II.6.7 Ansible et la sécurité

Ansible propose plusieurs fonctionnalités et bonnes pratiques pour renforcer la sécurité lors de l'utilisation de l'outil. Voici quelques points de vue sur la sécurité dans Ansible :

- **Connexions sécurisées :** Ansible utilise SSH (Secure Shell) pour se connecter aux hôtes cibles. Cela permet de sécuriser les communications entre l'hôte de contrôle et les hôtes cibles. [51]

- **Gestion des informations sensibles :** Ansible offre des mécanismes pour gérer les informations sensibles, telles que les mots de passe, les clés SSH, etc. Vous pouvez

utiliser des variables chiffrées dans des fichiers de variables ou dans des outils externes tels que Ansible Valut pour protéger ces informations.

- **Gestion des autorisations :** Ansible permet de spécifier des utilisateurs et des groupes avec des autorisations spécifiques pour l'exécution des Playbooks. Cela vous permet de contrôler les droits d'accès et de limiter les opérations aux utilisateurs autorisés. [51]

- **Audits et journalisation :** Ansible propose des fonctionnalités de journalisation qui enregistrent les activités et les modifications effectuées lors de l'exécution des Playbooks. Cela vous permet de garder une trace des opérations effectuées et de détecter toute activité suspecte.

- **Vérification des états :** Ansible utilise une approche déclarative, ce qui signifie que vous spécifiez l'état souhaité du système plutôt que de décrire les étapes pour l'atteindre. Cela permet de s'assurer que les systèmes restent conformes à l'état défini, réduisant ainsi les risques liés aux configurations inattendues. [51]

- **Intégration avec les outils de sécurité :** Ansible peut être intégré à d'autres outils de sécurité tels que les scanners de vulnérabilités, les outils de conformité, les systèmes de gestion des correctifs, etc. Cela vous permet d'automatiser les tâches de sécurité et de garantir que les systèmes sont maintenus à jour et sécurisés.

Il est important de noter que la sécurité dans Ansible dépend également de la manière dont vous configurez et gérez vos environnements, de la sécurisation de votre infrastructure et des pratiques de sécurité que vous suivez lors du développement des Playbooks. [51]

II.7 Conclusion

En conclusion, nous avons exploré les différents aspects de l'automatisation du réseau et examiné certains des outils populaires tels qu'Ansible. Nous avons constaté que l'automatisation du réseau offre de nombreux avantages, tels que la simplification de la gestion, l'amélioration de la sécurité et la réduction des coûts opérationnels.

Parmi les outils d'automatisation du réseau, Ansible se démarque en raison de sa puissance et de sa facilité d'utilisation. Il utilise un langage de script simple et déclaratif, ne nécessite pas l'installation d'agents sur les hôtes cibles et peut être intégré facilement à d'autres outils et technologies.

Dans le prochain chapitre, nous détaillerons la conception de notre architecture réseau et les technologies que nous avons choisies. Nous présenterons également la problématique principale qui a motivé notre projet. Enfin, nous expliquerons comment nous avons utilisé Ansible pour concevoir notre solution d'automatisation, qui nous permettra de résoudre cette problématique et d'optimiser notre réseau.

En combinant les avantages de l'automatisation du réseau avec la puissance et la simplicité d'Ansible, nous sommes convaincus que notre solution apportera des améliorations significatives à notre infrastructure et nous aidera à atteindre nos objectifs.

Chapitre III : Optimisation de l’Architecture Réseau

III.1 Introduction

L'architecture réseau est essentielle pour assurer le bon fonctionnement des entreprises en permettant la connectivité, la communication et l'échange de données. Cependant, la gestion d'un réseau complexe peut être un défi pour les équipes informatiques. C'est là que l'automatisation du réseau intervient en améliorant l'efficacité du réseau et en simplifiant sa gestion à l'aide d'un serveur dédié à la configuration, au monitoring et à l'automatisation. Dans ce chapitre, nous détaillerons les technologies que nous avons intégrées dans notre architecture, en mettant l'accent sur les objectifs que nous souhaitons atteindre et les problèmes que nous cherchons à résoudre. Nous présenterons également le rôle clé du serveur d'automatisation et son mode de fonctionnement. Notre objectif global est de créer une infrastructure réseau plus efficace et plus facile à gérer grâce à l'utilisation de ce serveur. Nous explorerons les aspects techniques et opérationnels afin de fournir une compréhension approfondie de notre solution.

III.2 Conception d’une Architecture Réseau Hiérarchique

Les modèles hiérarchiques vous permettent de concevoir des inter réseaux qui utilisent la spécialisation des fonctions associée à une organisation hiérarchique. Une telle conception simplifie les tâches requises pour construire un réseau qui répond aux exigences actuelles et peut se développer pour répondre aux exigences futures.

Les modèles hiérarchiques utilisent des couches pour simplifier les tâches d'interconnexion de réseaux. Chaque couche peut se concentrer sur des fonctions spécifiques, vous permettant de choisir les bons systèmes et fonctionnalités pour chaque couché. Les modèles hiérarchiques s'appliquent à la fois à la conception LAN et WAN.

L'approche de conception d'architecture hiérarchique de réseau est couramment utilisée pour créer des réseaux évolutifs, faciles à gérer et à dépanner. [3]

Certaines entreprises implémente l’architecture a 2 niveau au lieu de l’architecture a 3 niveau en raison de son cout élevé, La notion de "collapsed core" (noyau effondré) est une approche de conception de réseau qui simplifie l'architecture hiérarchique traditionnelle en combinant les fonctions de distribution et de cœur en une seule couche. Cela réduit la complexité et le coût du réseau, mais peut présenter des limitations en termes d'évolutivité et de redondance. [2]

Compte tenu de notre objectif de concevoir une architecture réseau adaptée à une taille moyenne, nous avons opté pour une approche en deux niveaux.

Selon l'illustration fournie, notre environnement de travail est principalement composé de deux commutateurs multicouches au niveau du noyau effondré (DS1(Distribution Switch), DS2) et de trois commutateurs dans la couche d'accès (AS1 (Access Switch), AS2, AS3). Veuillez-vous référer à la Figure III pour une visualisation de cette configuration.

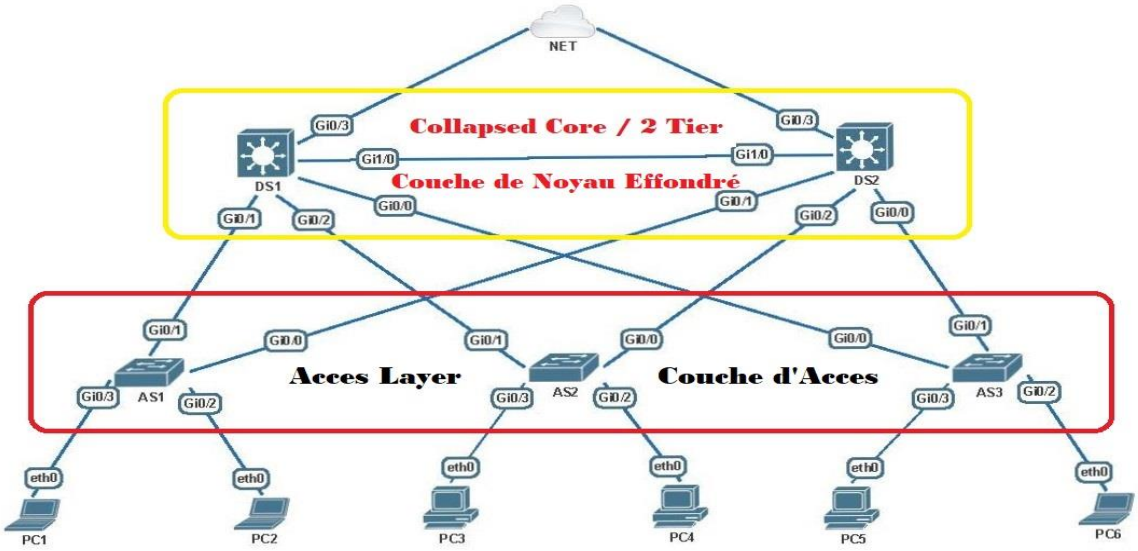


Figure III. 1 – Architecture a 2 niveaux.

III.3 Conception des protocoles et des mécanismes d'infrastructure réseau

L'architecture choisie, telle qu'illustrée dans la figure, implique la mise en œuvre de diverses technologies et protocoles réseau afin de tirer pleinement parti de chaque périphérique et de garantir un acheminement optimal des paquets de données.

III.3.1 Conception d'un EtherChannel

Les réseaux actuels sont souvent confrontés à une augmentation significative du trafic en raison du nombre croissant d'utilisateurs. Cependant, une conception réseau inadéquate peut entraîner de sérieux problèmes. Le fait de faire transiter ce trafic important à travers un lien qui ne peut le gérer peut entraîner une congestion du réseau, ce qui entraîne une dégradation sévère des performances.

La conception d'un EtherChannel, également connu sous le nom de port channel ou agrégation de liens Ethernet, vise à combiner plusieurs liens physiques entre deux équipements réseau en un seul lien logique. Cela permet d'augmenter la bande passante disponible, d'améliorer la redondance et la fiabilité du réseau.

Pour éviter la congestion sur le lien reliant les deux commutateurs multicouches, nous allons utiliser la technologie Etherchannel propriétaire de Cisco, appelée PAgP (Port Aggregation Protocol). Grâce à PAgP, nous pourrons combiner deux liens physiques entre ces deux commutateurs en un seul lien logique. En résumé, l'utilisation de l'EtherChannel permet d'augmenter la bande passante, d'équilibrer la charge, d'améliorer la redondance et la tolérance aux pannes, de simplifier la configuration et de renforcer la stabilité du réseau. Ces avantages en font une solution populaire pour améliorer les performances et la fiabilité des réseaux.

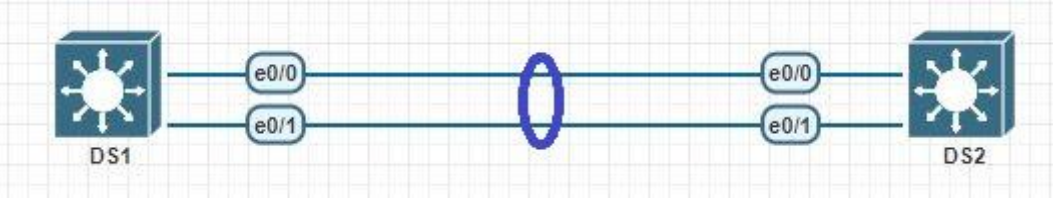


Figure III. 2 – EtherChannel (Agrégation des liens).

III.3.2 Mise en place d'une solution automatisée de gestion de configuration (DHCP)

Afin d'assurer une connectivité efficace et d'accéder aux ressources du réseau, chaque hôte dans un réseau basé sur TCP/IP doit disposer d'une adresse IP unicast unique, ainsi que d'autres paramètres nécessaires. Sans le protocole DHCP (Dynamic Host Configuration Protocol), les administrateurs réseau devraient configurer manuellement toutes les machines, ce qui serait une tâche longue, inefficace et sujette aux erreurs. Heureusement, DHCP simplifie ce processus en l'automatisant et en le gérant de manière centralisée. Un serveur DHCP gère un pool d'adresses IP et attribue une adresse à tout client compatible DHCP lorsqu'il se connecte au réseau.

Dans notre architecture, afin de maximiser l'efficacité de ce protocole, nous avons décidé de désigner les deux commutateurs de la couche du noyau effondré comme serveurs DHCP. Ainsi, nous allons créer deux pools DHCP sur chaque commutateur : DS1 aura les pools pour les sous-réseaux des VLAN 10 et 20, tandis que DS2 aura les pools pour les sous-réseaux des VLAN 50 et 70. Cette approche est optimale car elle permet de répartir la charge entre les deux serveurs, assurant ainsi une meilleure performance.

III.3.3 Conception des VLANs

La mise en œuvre de la technologie VLAN est essentielle dans les réseaux modernes. Elle permet de créer des domaines de diffusion spécifiques à chaque VLAN, ce qui permet de segmenter le réseau et d'améliorer sa sécurité, sa fiabilité et son efficacité. Dans notre cas, nous supposons que notre entreprise est composée de quatre Structures : Commercial, Info, Ressources humaines, Logistique. Afin de séparer le trafic de ces différentes structures, nous allons mettre en place un VLAN dédié à chacun d'entre eux. Ainsi, les hôtes d'une Structure ne pourront communiquer qu'avec les hôtes appartenant au même structure. La figure ci-dessous illustre notre conception des VLANs.

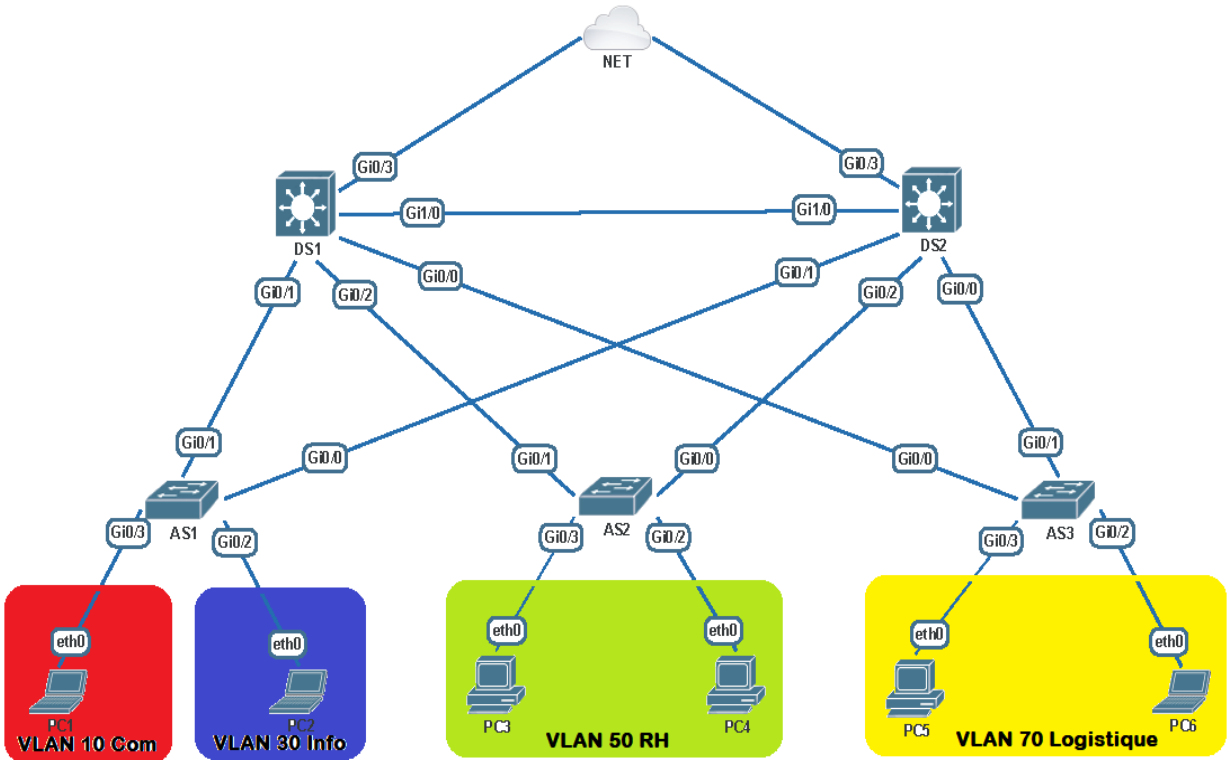


Figure III. 3 – Implémentation des VLANs.

Avec tout cela, il peut arriver qu'un hôte d'une structure spécifique ait besoin de communiquer avec un hôte d'une autre structure. Pour permettre cela, et étant donné que nous utilisons des commutateurs multicouches, nous allons mettre en place le routage inter-VLAN sur ces commutateurs.

Par ailleurs, notre architecture comprend deux commutateurs de niveau 3 (multicouches). Pour une répartition équitable de la charge de travail et des performances réseau optimales, nous allons configurer chaque commutateur de manière à router le trafic de seulement deux VLAN. Par conséquent, DS1 sera dédié au routage du trafic des VLAN 10 et 30, tandis que DS2 se chargera du routage pour les VLAN 50 et 70. Cette configuration nous permettra d'équilibrer la charge de travail entre les deux commutateurs, ce qui contribuera à optimiser les performances globales du réseau.

III.3.4 Optimisation du fonctionnement du protocole Spanning Tree

En raison de l'utilisation de la redondance des liens et des périphériques dans notre architecture, des boucles de couche 2 se sont formées. Pour résoudre ce problème, le protocole Spanning Tree (STP) intervient automatiquement en bloquant l'un des chemins redondants, ne laissant ainsi qu'un seul chemin actif de la source vers la destination.

Le protocole Rapid Spanning-Tree (RSTP) est généralement configuré par défaut sur la plupart des commutateurs Ethernet pour éviter les boucles dans le réseau. Malgré sa capacité à fournir une convergence rapide du réseau, le RSTP présente un inconvénient majeur, tous les commutateurs d'un réseau local partagent le même arbre. En conséquence, tous les paquets, quels que soient les VLAN auxquels ils appartiennent, sont acheminés le long du même chemin

(instance unique). Cette limitation empêche la possibilité de bloquer sélectivement les liens redondants en fonction des VLAN et rend l'équilibrage de charge entre les VLAN n'est pas réalisable.

Pour résoudre ce problème tout en conservant une convergence rapide, la solution la plus appropriée pour notre topologie est Rapid-PVST+. Il s'agit d'une version propriétaire de Cisco du protocole STP qui crée une instance pour chaque VLAN. Dans chaque instance, un processus STP distinct se déroule, un pont racine différent est sélectionné, des rôles de port différents sont utilisés, et ainsi de suite. Cela permet une gestion séparée et optimisée des VLANs, offrant une meilleure flexibilité et des performances améliorées pour notre réseau.

Étant donné que notre réseau est conçu pour inclure quatre VLAN, nous aurons donc quatre instances de Spanning-Tree, ce qui nous offre un moyen pratique de contrôler notre réseau. Nous allons manipuler les priorités pour que DS1 à devenir le commutateur racine des instances STP des VLAN 10 et 30, tandis que DS2 sera le commutateur racine des instances des VLAN 50 et 70. Par conséquent, les chemins seront ouverts pour les instances STP des VLAN 10 et 30 et fermés pour celles des VLAN 50 et 70, et inversement. Cela nous permettra d'équilibrer la charge, ce qui entraînera une augmentation efficace de la bande passante globale du réseau. La figure présente un résumé de cette conception. Cela permettra d'équilibrer la charge et d'optimiser l'utilisation globale de la bande passante du réseau. La figure ci-dessous illustre cette configuration.

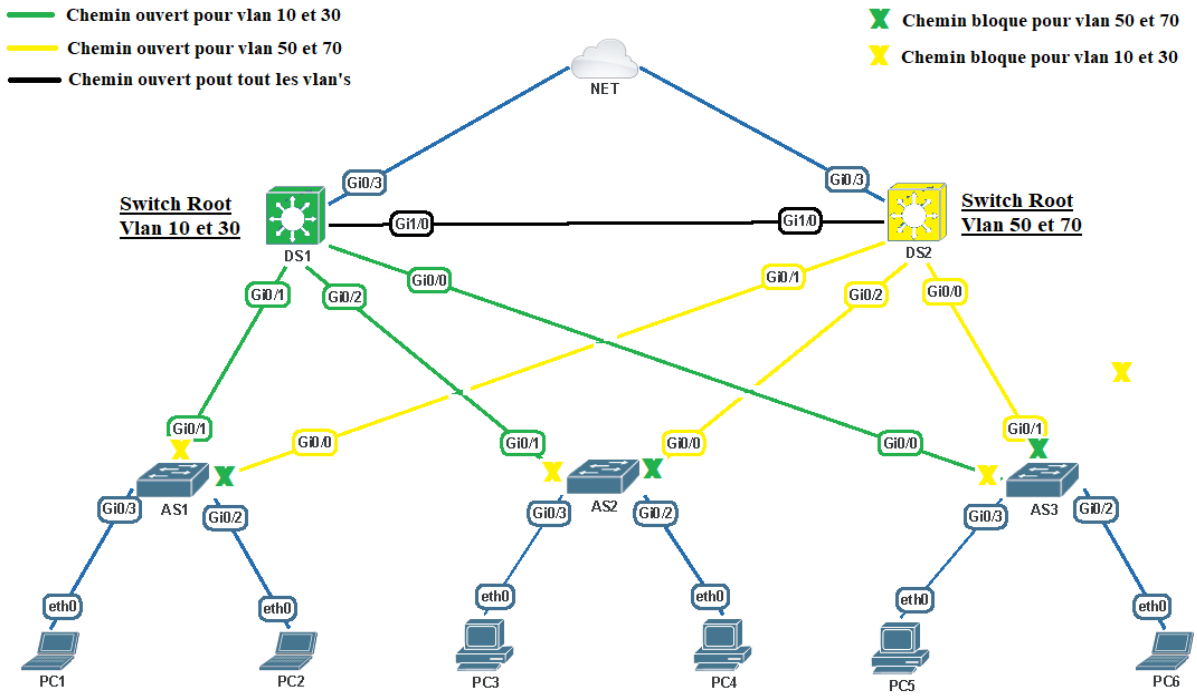


Figure III. 4 – Equilibrage de charge de VLAN en utilisant STP.

Une fonctionnalité essentielle pour accélérer la convergence du protocole Spanning-Tree est le Port-Fast, nous allons l'activer sur les ports des commutateurs de la couche d'accès qui sont connectés aux périphériques finaux. Cela permettra aux utilisateurs de se connecter instantanément au réseau, évitant ainsi les problèmes liés à une convergence lente du réseau.

III.3.5 Mise en place d'une solution de redondance au niveau 3

La conception d'une solution de redondance au niveau 3 est essentielle pour assurer la disponibilité, la fiabilité et la tolérance aux pannes du réseau. En partant du principe qu'en cas de défaillance d'un périphérique, un autre peut automatiquement le remplacer. Dans cette optique, Il existe de nombreux protocoles disponibles pour assurer la redondance, tels que HSRP, VRRP et GLBP. Dans notre conception, nous avons décidé d'adopter le protocole HSRP (Hot Standby Router Protocol) comme solution de redondance. Ce protocole de redondance, propriétaire de Cisco, est réputé pour sa simplicité d'implémentation.

Cependant, le HSRP présente un inconvénient majeur : il n'utilise pas de manière efficace tous les périphériques disponibles. Il se contente d'utiliser un seul commutateur multicouche tandis que l'autre reste en veille. Bien que nous ne puissions pas modifier le concept de base du HSRP, nous avons la possibilité de configurer plusieurs instances de HSRP. En utilisant ces instances, nous pouvons exploiter les deux commutateurs simultanément, ce qui permet d'obtenir un équilibrage de charge HSRP.

En divisant notre réseau en quatre sous-réseaux grâce à la création de quatre VLANs, nous avons la possibilité de configurer une instance distincte de HSRP pour chaque VLAN. Par exemple, nous aurons une instance 10 pour le VLAN 10, une instance 30 pour le VLAN 30, etc. Afin de répartir la charge entre les deux commutateurs, nous ajustons la séquence de priorité de manière à ce que chaque commutateur soit actif pour la moitié des VLANs et en veille pour les autres. Pour ce faire, nous augmentons la priorité des instances 10 et 30 sur DS1 et la réduisons sur DS2, et vice versa pour les instances 50 et 70. Ainsi, DS1 acheminera le trafic des VLANs 10 et 30, tandis que DS2 prendra en charge le trafic des VLANs 50 et 70.

La figure illustre la configuration de notre solution de redondance au niveau 3.

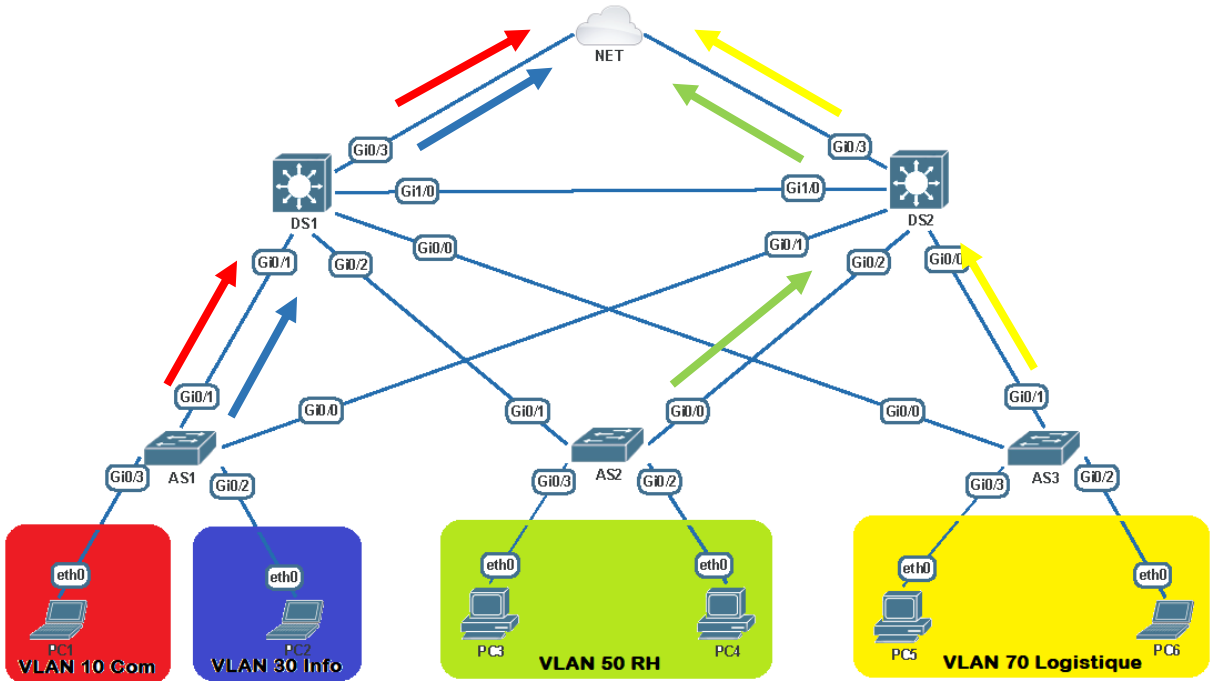


Figure III. 5 – Répartition de charge HSRP.

Dans le cas où le lien reliant le commutateur actif à l'extérieur du LAN se coupe, il existe un risque de perte du trafic qui était censé être acheminé par cette connexion. Afin de prévenir ce risque, nous allons mettre en place un mécanisme de suivi de l'état des interfaces au-delà du LAN qui s'appelle Le Track. En influençant l'élection en réduisant la priorité du commutateur actif, nous assurerons que lorsqu'un changement d'état est détecté (par exemple, passage de "Up" à "Down") sur l'interface surveillée, les commutateurs basculeront automatiquement vers un nouvel état HSRP jusqu'à ce que l'interface retrouve son état normal.

Le fonctionnement du suivi HSRP (Track) peut être décrit selon le schéma présenté dans la figure, illustrant le principe de son fonctionnement.

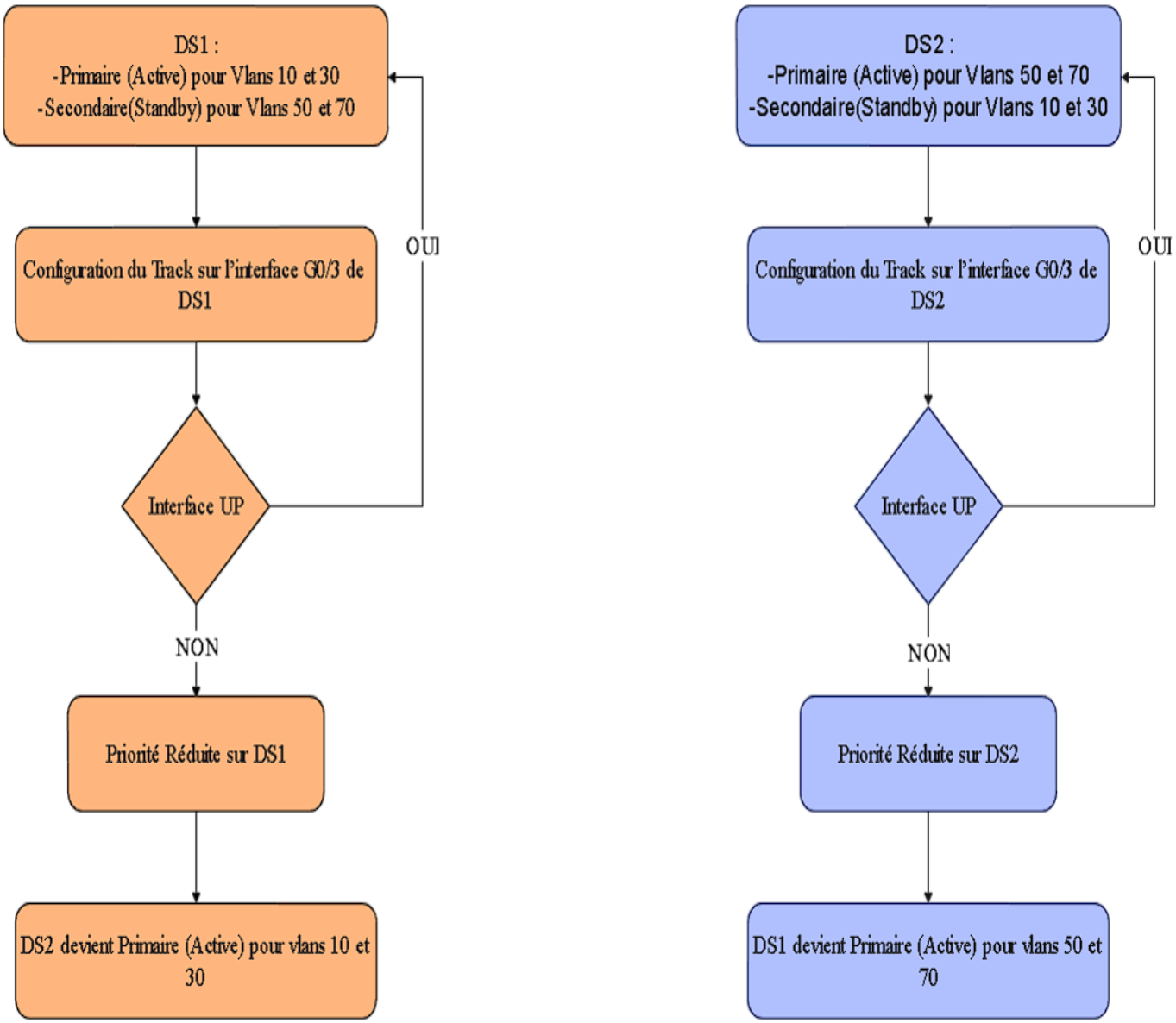


Figure III. 6 – Fonctionnement du suivi HSRP (Track).

Problématique :

Dans notre conception de topologie de base, nous avons montré que les protocoles HSRP et Spanning-Tree (pvst) peuvent être utilisés en plusieurs instances, en fonction des VLANs. Les deux protocoles permettent l'équilibrage de charge, bien que de manière légèrement différente, HSRP divise le trafic sortant vers l'extérieur du LAN sur plusieurs sorties (DS1 et DS2 dans notre cas), tandis que Spanning-Tree le réalise en autorisant certaines instances à traverser un chemin spécifique à l'intérieur du LAN, tout en en interdisant d'autres.

Pour maximiser l'efficacité de notre configuration, nous avons choisi DS1 comme commutateur "Active" pour HSRP et comme commutateur racine pour les instances 10 et 30 de Spanning-Tree. DS2, quant à lui, assume les mêmes rôles pour les instances 50 et 70. Dans un scénario idéal, la charge sera équilibrée et le trafic suivra un chemin optimal pour sortir du LAN, comme illustré précédemment dans la figure 7.

Toutefois, il existe des circonstances où l'état du HSRP peut changer. Par exemple, si nous avons configuré un traque (Track) sur le lien reliant le commutateur DS1 à l'extérieur du réseau local, et que ce lien se coupe, DS2 deviendra le commutateur "Active" pour toutes les instances HSRP, tandis que DS1 passera en mode "Standby" pour ces dernières. Cependant, l'état du Spanning-Tree restera inchangé, ce qui pose un problème majeur. En effet, le trafic des VLAN 10 et 30 empruntera désormais un chemin plus long en raison du lien bloqué par le protocole Spanning-Tree de manière non optimale. Cela aura un impact négatif sur les performances du réseau. La figure ci-dessus illustre la situation décrite précédemment.

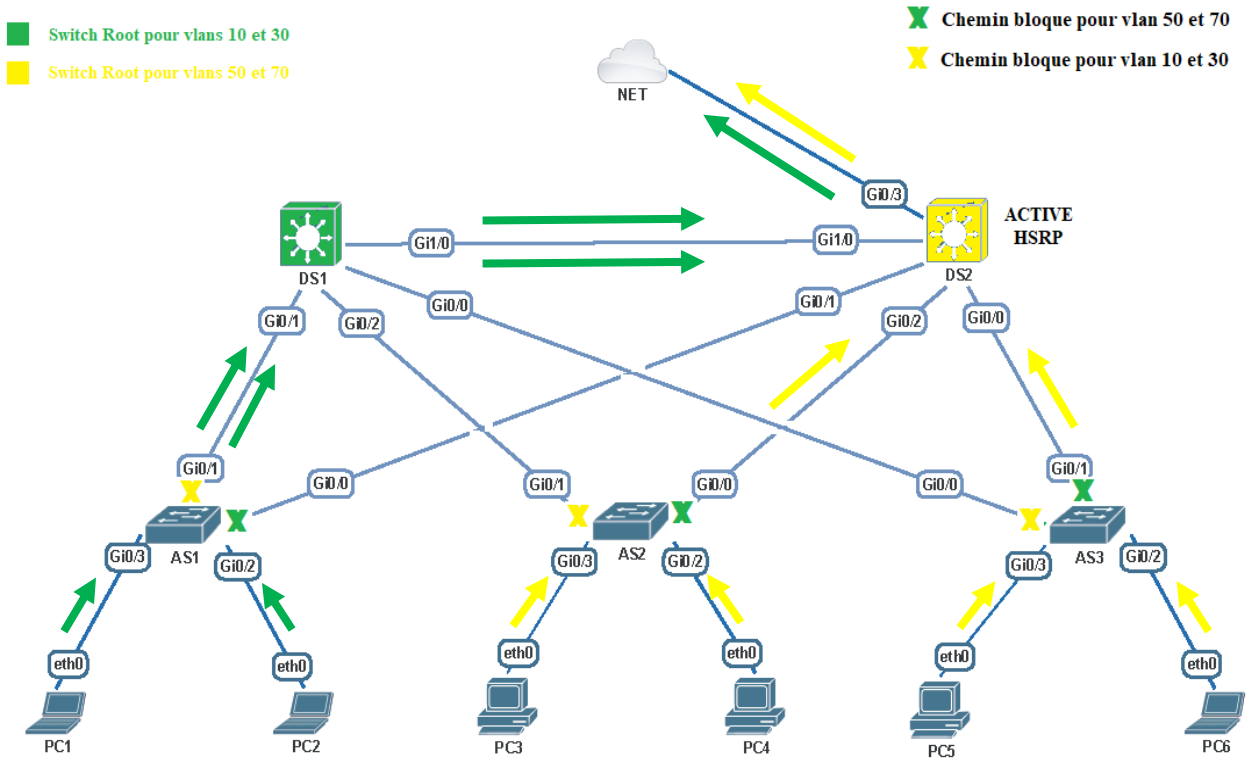


Figure III. 7 – Gestion du trafic en l'absence de communication STP-HSRP.

Un scénario plus critique peut se produire si, d'une manière ou d'une autre, le commutateur DS1 devient la racine de tous les instances de Spanning-Tree, tandis que DS2 est le commutateur primaire "Active" en HSRP pour toutes les instances. Dans ce cas, tout le trafic devrait passer par le lien reliant les deux commutateurs de la couche du noyau effondré, qui pourrait devenir surchargé. Cela entraînerait une dégradation complète du réseau. Vous pouvez observer ce cas dans la figure III.8.

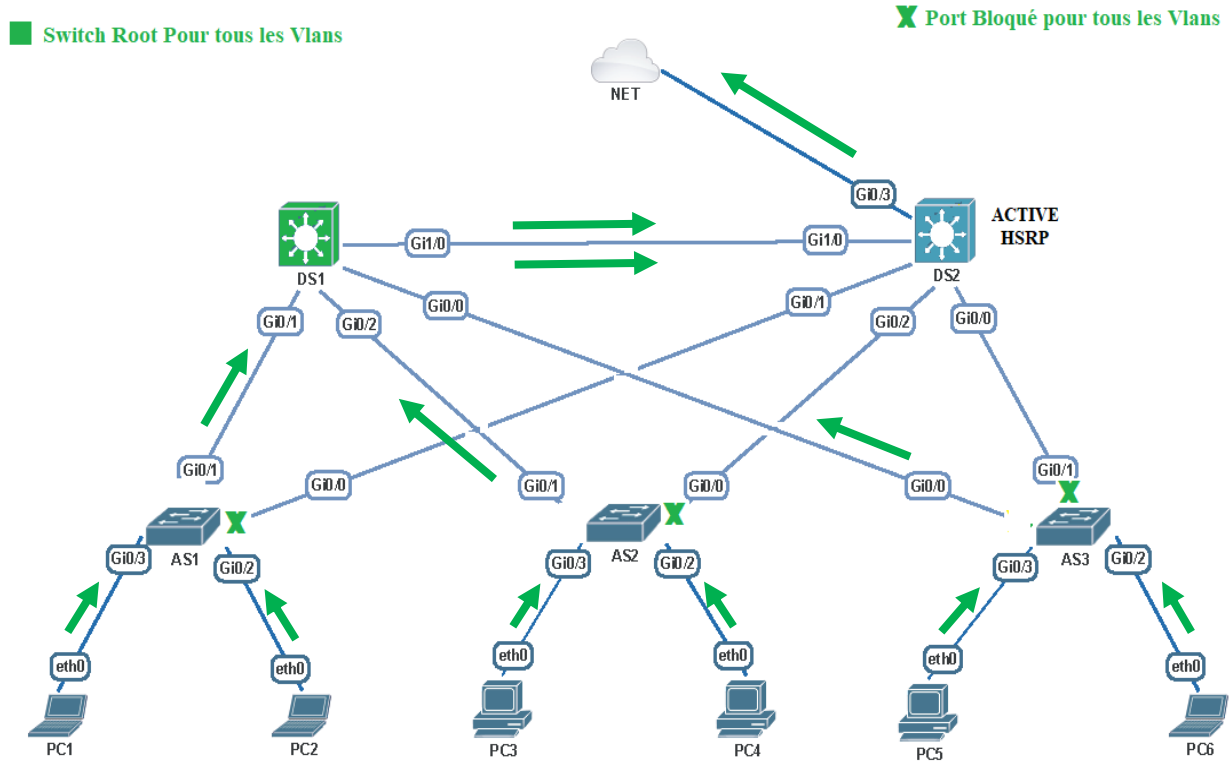


Figure III. 8 – Acheminement du trafic en cas d'absence de communication STP-HSRP.

L'absence de communication entre les protocoles HSRP et Spanning-Tree est probablement la principale cause de ces problèmes. En effet, ces deux protocoles n'ont pas été conçus pour échanger des informations entre eux. Pour résoudre cette problématique, l'automatisation du réseau représente la solution idéale. Elle permet d'établir une liaison entre deux protocoles totalement différents, afin qu'ils puissent s'adapter mutuellement. En outre, elle offre de nombreux autres avantages, que nous aborderons dans la suite de ce chapitre.

III.4 Optimisation de l'architecture

Dans cette section, nous explorons les différentes stratégies pour optimiser l'architecture de notre réseau. Nous avons identifié trois approches distinctes qui peuvent contribuer à cette optimisation.

III.4.1 L'intégration d'un serveur dédié d'automatisation

Comme mentionné précédemment, la configuration manuelle du réseau est une tâche lente et laborieuse. Sa complexité croît et elle devient plus sujette aux erreurs à mesure que le nombre de périphériques à configurer et les technologies à déployer augmentent. En revanche, l'automatisation du réseau peut nous libérer de cette tâche, diminuant ainsi les risques d'erreurs humaines et permettant d'économiser un temps précieux en évitant les configurations répétitives quotidiennes. De plus, grâce à ses nombreuses fonctionnalités supplémentaires par rapport à la configuration manuelle, l'automatisation du réseau améliore de manière significative les performances de celui-ci.

Afin d'introduire le concept d'automatisation dans notre topologie et de tirer parti de ses avantages, nous mettrons en place un serveur d'automatisation basé sur Ansible, directement connecté au commutateur multicouche DS1. Ce serveur sera responsable de l'orchestration, de la surveillance et de la configuration automatique de l'ensemble de la topologie. Par la suite, nous configurerons SSH sur chaque périphérique de notre architecture, permettant ainsi au serveur d'automatisation d'injecter les configurations nécessaires. Dans la Figure III.9, on peut observer comment le serveur est intégré à notre architecture et comment le trafic est acheminé à travers SSH.

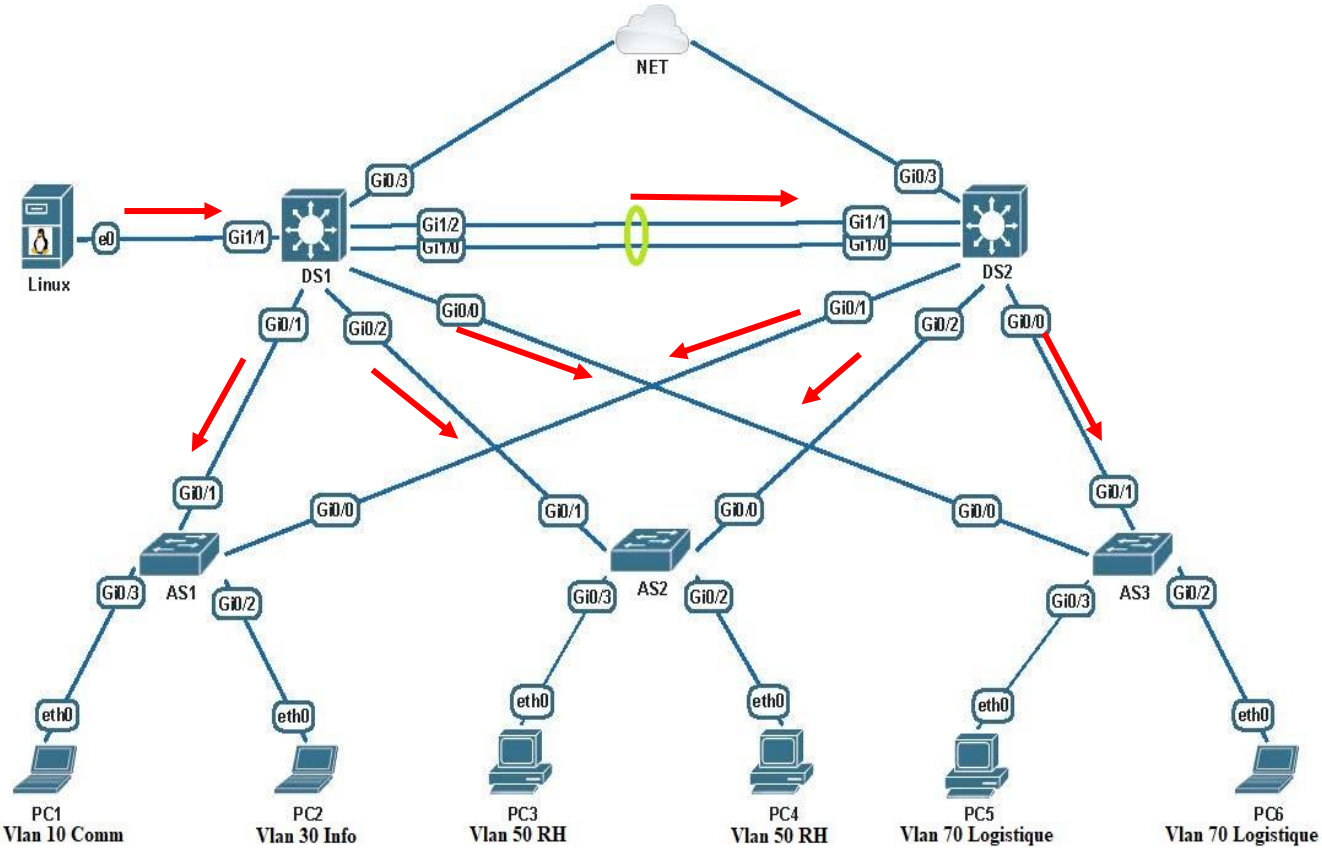


Figure III. 9 – Intégration d'un serveur d'automatisation.

III.4.2 Configuration via le serveur d'automatisation

La configuration manuelle des protocoles et technologies mentionnés précédemment dans ce chapitre est complexe, ce qui rend cette méthode peu pratique dans ce type de réseau. En effet, elle est chronophage et nécessite un grand nombre de commandes à répéter sur chaque périphérique.

Pour simplifier cette complexité, nous allons convertir ces commandes CLI en fichiers YAML (Playbooks), Ansible prendra en charge l'exécution de ces fichiers, ce qui lui permettra d'apporter les modifications nécessaires aux périphériques réseau. Ainsi, il n'est plus nécessaire d'avoir un accès physique aux périphériques, et les fichiers de configuration YAML peuvent être sauvegardés et réutilisés à tout moment. De plus, lors de l'ajout d'un nouveau dispositif, il suffit d'effectuer une modification rapide et d'exécuter le fichier correspondant pour adapter ce nouveau dispositif au fonctionnement de notre topologie.

III.4.3 Conception d'une solution de communication combinant STP et HSRP

Comme indiqué dans la problématique, Spanning-Tree et HSRP sont deux protocoles distincts qui fonctionnent de manière autonome, sans prendre en compte les informations de l'autre. Par conséquent. Leur configuration traditionnelle ne permet pas d'optimiser les performances du réseau. Cependant, grâce à l'intégration d'un serveur d'automatisation dans la topologie, nous avons la possibilité de résoudre cette complexité en surveillant les états de ces protocoles et en intervenant en cas de dysfonctionnement, afin d'ajuster leurs paramètres.

Création des Playbook Ansible :

Afin de permettre au serveur d'automatisation de modifier le fonctionnement du protocole Spanning-Tree, il est nécessaire d'implémenter des Playbooks qui ajustent les paramètres de ce protocole en fonction de l'état du protocole HSRP. Ces Playbooks doivent être exécutés par le serveur pour appliquer les modifications sur le réseau.

Ainsi, plusieurs critères peuvent avoir un impact sur le fonctionnement du STP, tels que la priorité des commutateurs, les adresses MAC des commutateurs, les coûts des liens et sur certains cas l'ID du port. Nos Playbooks se baseront principalement sur les critères suivants pour effectuer les ajustements nécessaires :

- **Priorité des commutateurs :** La priorité joue un rôle essentiel dans l'élection du commutateur racine. Par conséquent, le serveur d'automatisation garantira que le commutateur "Actif" en HSRP sera toujours désigné comme le commutateur racine du réseau. Ainsi, le lien reliant un commutateur de la couche d'accès au commutateur actif restera naturellement ouvert.

- **Couts des liens :** Si la modification de la priorité ne parvient pas à influencer le lien bloqué pour une raison quelconque, le serveur d'automatisation utilisera alors le coût des liens. En réduisant le coût du lien entre un commutateur de la couche d'accès et le commutateur "Actif" en HSRP, et en augmentant le coût du lien vers le commutateur

en état "Standby", on garantit que le premier lien reste ouvert tandis que l'autre reste bloqué.

L'organigramme présenté dans la Figure III.10 démontre comment nous avons utilisé ces deux critères dans nos Playbooks pour effectuer des modifications sur le fonctionnement du réseau.

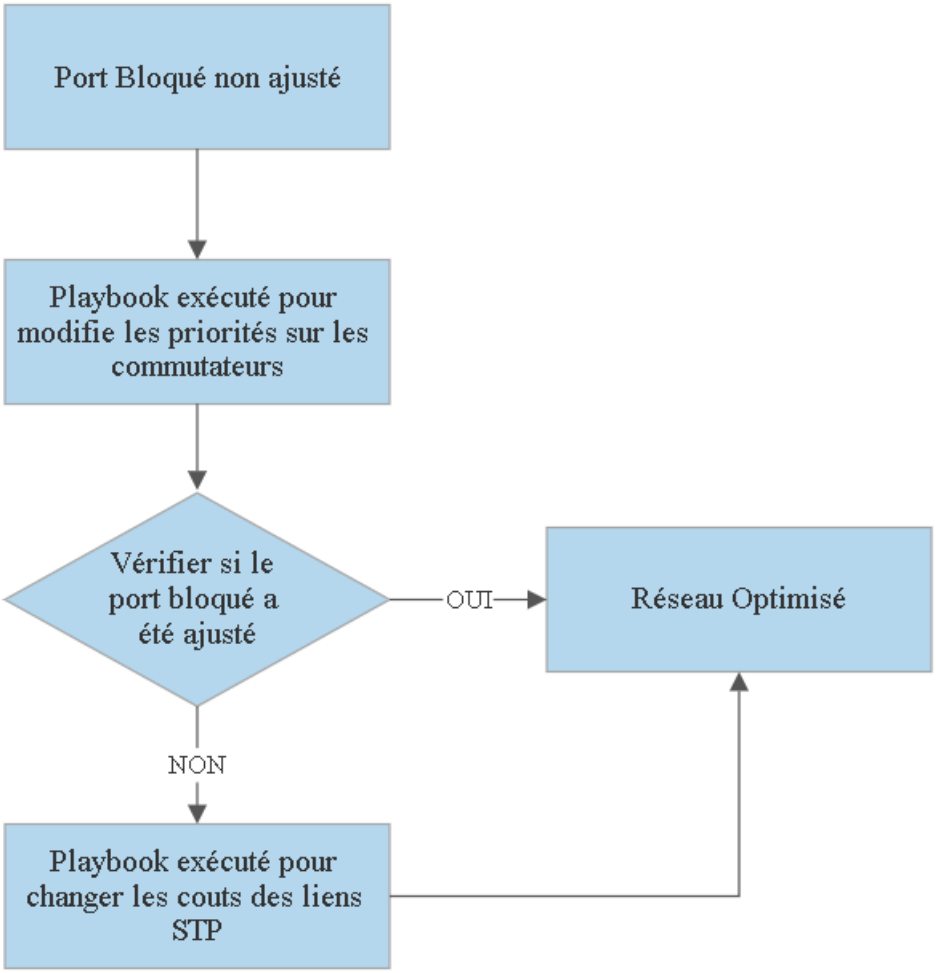


Figure III. 10 – Utilisation de la priorité et le cout pour modifier les paramètres de stp dans une Playbook.

Avant d'exécuter les Playbooks de modification des paramètres STP, il est essentiel de connaître l'état des protocoles STP et HSRP afin de déterminer s'il est nécessaire d'apporter des modifications. Nous mettrons donc en place d'autres Playbooks qui récupéreront l'état de ces protocoles et le stockeront dans des fichiers texte. Nous mettrons également en place un script Python qui analysera ces fichiers et nous fournira des informations sur le commutateur en état "Active" en HSRP ainsi que sur le lien bloqué par le STP.

Surveillance du réseau

Pour éviter que l'administrateur réseau n'ait à exécuter périodiquement les Playbooks pour vérifier l'état des protocoles et ajuster leur fonctionnement, le serveur d'automatisation doit être

doté d'un outil de surveillance en continu du réseau. Cet outil lui permettra de détecter tout dysfonctionnement et d'optimiser automatiquement le réseau en conséquence.

En considérant que notre serveur d'automatisation fonctionne sous le système d'exploitation Linux (Cent OS), le Shell est l'outil le plus approprié pour assurer la surveillance et l'orchestration du réseau, le Shell est à la fois un interpréteur de commandes et un langage de programmation qui permet aux utilisateurs d'interagir avec le noyau d'un système d'exploitation.

Nous allons mettre en place un script Shell qui sera automatiquement lancé au démarrage du serveur et qui restera en surveillance continue du réseau. L'objectif principal de ce script est de commencer par exécuter les Playbooks qui récupèrent l'état des protocoles STP et HSRP, enregistrent ces états dans des fichiers.txt, puis les traitent à l'aide d'un script Python pour extraire les informations pertinentes. Ensuite, le script Shell vérifie si le lien entre un commutateur de la couche d'accès et le commutateur actif en HSRP est bloqué. Si c'est le cas, il exécute les Playbooks qui modifient les paramètres du protocole STP. De cette manière, le réseau reste constamment optimisé.

Le fonctionnement du script Shell peut être visualisé à travers le schéma illustratif présenté dans la Figure III.11.

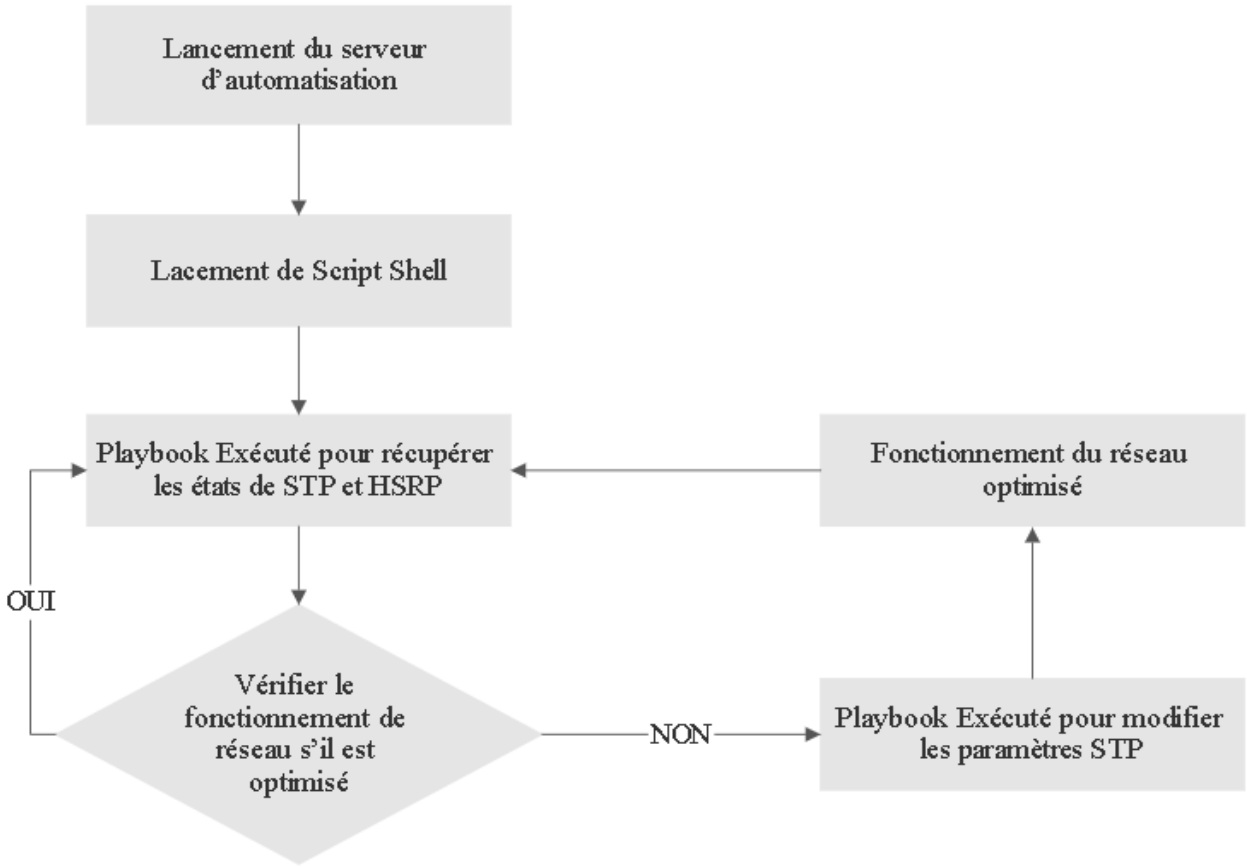


Figure III. 11 – Fonctionnement de Script Shell.

III.5 Conception d'une interface graphique

La ligne de commande (CLI) est la méthode principale utilisée pour configurer et gérer les périphériques réseau. Elle offre une rapidité d'exécution pour ceux qui sont familiers avec les commandes en ligne, fournissant des résultats immédiats après la saisie des commandes. De plus, la CLI requiert peu de mémoire et consomme peu de temps de traitement CPU.

Néanmoins, L'utilisation de la ligne de commande requiert une courbe d'apprentissage abrupte, Étant une interface exclusivement textuelle, elle présente certains défis. Pour les nouveaux utilisateurs, la ligne de commande peut être intimidante, car les instructions à suivre sont souvent limitées. De plus, la ligne de commande comporte un grand nombre de commandes, ce qui peut être un inconvénient même pour les utilisateurs expérimentés. Elle laisse également peu de place à l'erreur. En cas de saisie incorrecte d'une commande, le périphérique ne peut pas la traiter correctement, ce qui peut entraîner des problèmes. De plus, en cas d'erreur, les utilisateurs doivent souvent recommencer le processus depuis le début.
De plus en plus, l'utilisation d'une interface graphique (GUI) est privilégiée pour la configuration réseau, afin de surmonter les inconvénients de la ligne de commande (CLI), Elle offre aux utilisateurs la possibilité d'interagir avec les dispositifs réseau en utilisant des éléments graphiques et de remplacer les commandes textuelles par des actions conviviales. À la différence des systèmes de CLI qui exigent que les utilisateurs retiennent les commandes, les systèmes de GUI adoptent une approche relativement intuitive. Même les utilisateurs sans formation poussée peuvent facilement apprendre à utiliser le système et atteindre leurs objectifs de gestion. En revanche, les systèmes GUI offrent un environnement visuel plus attrayant, ce qui en fait une caractéristique souhaitable pour la plupart des utilisateurs finaux.

Dans cette optique, nous allons élaborer une interface graphique qui simplifiera les tâches de surveillance et de gestion pour les administrateurs réseau. Cette interface graphique sera intégrée au sein du serveur d'automatisation et s'appuiera sur la solution Ansible pour récupérer les informations sur les dispositifs réseau et exécuter des commandes correspondantes. L'interface graphique permettra les actions suivantes :

- Affichage des états des protocoles Spanning-Tree et HSRP pour chaque commutateur.
- Configuration des VLANs.
- Configuration des serveurs DHCP.

III.6 Conclusion

Dans ce chapitre, nous avons développé notre topologie en deux étapes distinctes. Dans la première étape, nous avons établi une architecture de base sur laquelle nous avons présenté diverses solutions technologiques pour assurer le bon fonctionnement de notre infrastructure. Dans la deuxième étape, nous avons proposé l'ajout d'un serveur d'automatisation qui jouera les rôles de surveillance et d'orchestration du réseau. Dans le prochain chapitre, nous allons détailler les phases d'implémentation des conceptions d'infrastructure proposées et tester le bon fonctionnement de notre environnement de travail.

Chapitre IV : Mise en œuvre technique

IV.1 Introduction

Une fois que nous avons identifié toutes les technologies nécessaires pour assurer le bon fonctionnement de notre projet dans la phase de conception, nous allons maintenant décrire en détail les étapes de déploiement qui nous mèneront à la mise en œuvre de la solution finale.

Tout d'abord, nous allons introduire l'environnement de simulation, puis nous mettrons en évidence les étapes de configuration.

IV.2 Installation de l'environnement de travail :

Avant de passer à la phase de configuration, il est nécessaire de préparer l'environnement de simulation. Pour ce projet, nous avons choisi d'utiliser PNETLab (Packet Network EmulatorToolLab), un outil puissant et gratuit. PNETLab (Packet Network EmulatorToolLab) est une plateforme qui répond aux exigences des réseaux modernes en proposant un environnement de simulation pour différentes solutions technologiques, le tout à un coût minimal et dans un délai court. Cet outil présente de nombreux avantages, notamment :

- **Coût minimal :** PNETLab est une solution gratuite, ce qui permet de réduire les coûts liés à l'acquisition d'autres plateformes de simulation.
- **Simplicité d'utilisation :** PNETLab est conçu pour être convivial, ce qui facilite la prise en main et l'utilisation de l'outil, même pour les utilisateurs novices.
- **Large éventail de fonctionnalités :** PNETLab offre un ensemble complet de fonctionnalités pour la simulation de diverses solutions technologiques, ce qui permet de modéliser et de tester des scénarios complexes.
- **Flexibilité :** PNETLab prend en charge différentes topologies réseau, ce qui permet de créer des configurations personnalisées en fonction des besoins spécifiques du projet.
- **Temps de déploiement réduit :** Grâce à PNETLab, il est possible de déployer rapidement un environnement de simulation, ce qui permet de gagner du temps lors du processus de configuration et de test.

Pour utiliser PNETLab, il est nécessaire de respecter certaines exigences matérielles. Voici les prés requis :

Processeur : i5/i7

Mémoire RAM : 8 Go

Espace de stockage : 40 Go

Interface réseau : NAT

De plus, il est recommandé d'activer la virtualisation Intel dans le BIOS de votre système. Vous devrez également télécharger le fichier ISO de PNETLab et installer une machine virtuelle, telle que VMware Workstation Pro, sur laquelle vous exécuterez PNETLab.

L'interface graphique de PNETLab est accessible via une interface web à partir d'une adresse IP fournie dans l'environnement VMware, comme illustré dans la figure IV.1.

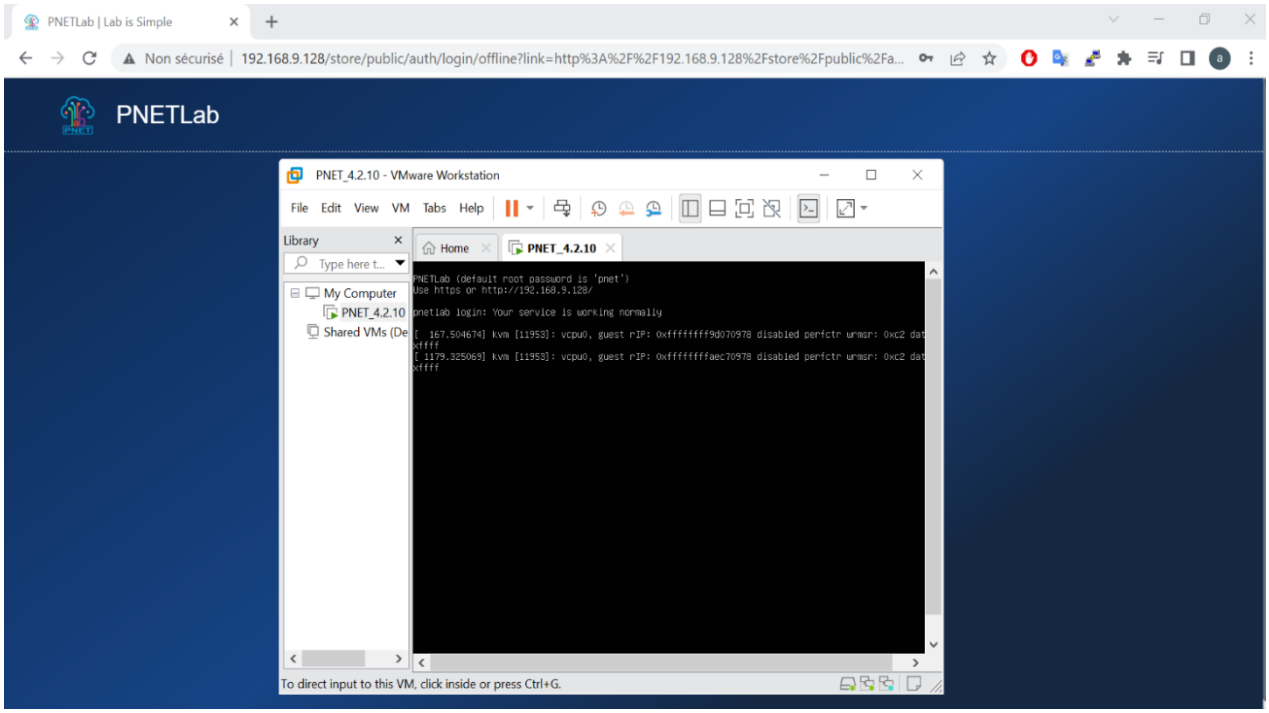


Figure IV. 1 – Interface de PNETLab.

La topologie qui a été présentée dans le chapitre précédent est composée de plusieurs commutateurs et terminaux. Dans le cadre de ce projet, nous utiliserons cette topologie pour mettre en place notre environnement de travail et effectuer nos configurations. Nous avons opté pour l'image vios_l2-adventerprisek9-m développée par Cisco IOS Software pour les commutateurs de notre topologie. En ce qui concerne les terminaux, nous utiliserons les PC virtuels (VPC) fournis par PNETLab. Par la suite nous allons améliorer la topologie en intégrant un serveur Ansible. Pour ce serveur, nous avons opté pour le système d'exploitation CentOS 7 qui est une distribution GNU/Linux populaire et fiable, largement utilisée dans les environnements de serveurs. Elle est prise en charge par l'entreprise RedHat, ce qui crée un lien étroit entre CentOS et Ansible. Cette compatibilité entre les deux offre un avantage considérable en termes de fonctionnalités et de facilité d'intégration.

Nous utiliserons le logiciel Putty pour accéder au mode CLI des périphériques de PNETLab.

Cela nous permettra d'établir une connexion via le protocole Telnet, comme le montre la figure IV.2.

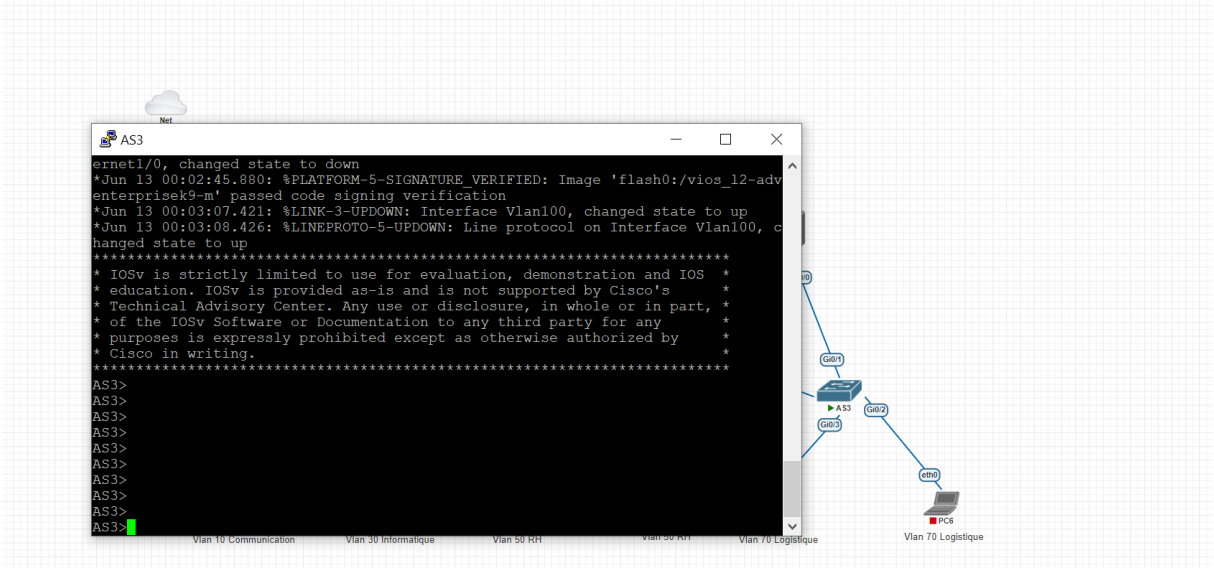


Figure IV. 2 – Accès au mode CLI du commutateur via Telnet.

IV.3 Mise en œuvre de l'architecture de base

Avant de commencer la mise en œuvre de la solution principale de notre projet, nous allons d'abord présenter la mise en place de l'architecture de base telle qu'elle a été conçue dans le chapitre précédent, en incluant les différentes technologies et protocoles utilisés. La façon la plus évidente de créer cette architecture est d'utiliser l'interface en ligne de commande (CLI), qui est l'approche traditionnelle. Néanmoins cette méthode est laborieuse et sujette aux erreurs. Par conséquent, nous avons préféré utiliser le serveur d'automatisation Ansible. En effet, la configuration à l'aide d'Ansible ne nécessite que la création et l'exécution de playbooks, ce qui offre plus de sécurité et de simplicité, car il permet de cibler plusieurs périphériques simultanément. De plus, les Playbooks peuvent être sauvegardés et consultés à tout moment.

Dans cette section, nous fournirons un aperçu des commandes CLI nécessaires pour configurer chaque protocole, puis nous mettrons en évidence les Playbooks que nous avons exécutés pour activer ces protocoles.

IV.3.1 Implémentation du plan d'adressage et des VLANs

Pour permettre la communication entre les périphériques du réseau, nous devrons leur attribuer des adresses IP.

Dans notre réseau local, nous utiliserons une plage d'adresses IP privées appartenant à la classe A : 10.0.0.0/8.Puisque nous avons créé quatre VLAN dans notre topologie (10 : comm, 30 : info, 30 : RH 70 : logistique et 100 : administration),

Nous allons partitionner cette adresse réseau en plusieurs sous-réseaux ayant la forme 10.0.X.0/24, où "X" représente l'ID du VLAN. Ainsi, chaque VLAN disposera de son propre sous-réseau. De plus, afin de séparer le trafic du serveur Ansible que nous ajouterons ultérieurement à notre topologie, nous allons l'affecté au VLAN 100 (administration) spécifiquement dédié à la gestion à travers ce serveur.

En outre, pour permettre l'activation du routage inter-VLAN et la gestion à distance des commutateurs, il est essentiel de créer des interfaces VLAN sur chaque commutateur. Le tableau IV.1 ci-dessous présente un aperçu de l'adressage des interfaces VLAN utilisées sur chaque commutateur :

Vlan	Switch	Adresse Interface Vlan
10	**DS1**	10.0.10.1
	DS2	10.0.10.2
30	**DS1**	10.0.30.1
	DS2	10.0.30.2
50	**DS1**	10.0.50.1
	DS2	10.0.50.2
70	**DS1**	10.0.70.1
	DS2	10.0.70.2
100	**DS1**	10.0.100.1
	DS2	10.0.100.2
	AS1	10.0.100.10
	AS2	10.0.100.20
	AS3	10.0.100.30

Table IV. 1– Adressage des interfaces VLAN.

Pour mettre en œuvre la technologie VLAN que nous avons conçue, il sera nécessaire d'exécuter plusieurs commandes sur chaque commutateur. La figure IV.3 illustre un exemple de création du VLAN 10 et d'une interface VLAN 10 sur le commutateur AS1 en utilisant la ligne de commande.

```
DS1(config)# VLAN 10
DS1(config-VLAN)# name comm
DS1(config-VLAN)# exit
DS1(config)# interface VLAN 10
DS1(config-if)# ip address 10.0.10.1
DS1(config-if)# exit
```

Figure IV. 3 – Configuration des VLAN 10 sur AS1 via CLI.

Comme nous utilisons Ansible, nous devons traduire les commandes présentées dans la figure IV.3 sous forme de playbook. La figure IV.4 illustre une partie du playbook qui permet la création des VLANs et des interfaces VLAN sur tous les commutateurs.

```
---
- name: Vlans
  hosts: Switches

  tasks:
    - name: Vlan Creation
      cisco.ios.ios_vlans:
        config:
        - name: comm
          vlan_id: 10
        - name: info
          vlan_id: 30
        - name: RH
          vlan_id: 50
        - name: logistique
          vlan_id: 70
        state: merged

- name: Vlan Interfaces on DS1
  hosts: DS1

  tasks:
    - name: Create interface vlan 10
      cisco.ios.ios_config:
        parents: interface vlan 10
        lines:
        - ip address 10.0.10.1 255.255.255.0
        - no shutdown
```

Figure IV. 4 – Configuration VLAN 10 sur DS1 via Ansible.

IV.3.2 Implémentation du protocole DHCP :

La configuration des adresses IP des serveurs et commutateurs de notre topologie doit être configurées manuellement. En revanche, nous allons utiliser le protocole DHCP pour attribuer automatiquement les adresses aux terminaux. Plus précisément, DS1 sera responsable d'assigner les adresses IP aux hôtes des VLANs 10 et 30, tandis que DS2 se chargera des hôtes des VLANs 50 et 70.

La figure IV.5 présente un exemple de configuration du protocole DHCP sur DS1 à l'aide de l'interface en ligne de commande (CLI).

```
DS1(config)# ip dhcp excluded-address 10.0.10.250
DS1(config)# ip dhcp pool VLAN10
DS1(dhcp-config)# default-router 10.0.10.1
DS1(dhcp-config)# network 10.0.10.0 255.255.255.0
DS1(dhcp-config)# exit
DS1(config)# ip dhcp excluded-address 10.0.30.250
DS1(config)# ip dhcp pool VLAN30
DS1(dhcp-config)# default-router 10.0.30.1
DS1(dhcp-config)# network 10.0.30.0 255.255.255.0
DS1(dhcp-config)# exit
```

Figure IV. 5 – Configuration du DHCP sur DS1 via CLI.

La Figure IV.6 illustre une section de la Playbook exécutée, qui configure le commutateur DS1 pour fonctionner en tant que serveur DHCP pour les VLAN 10 et 20.

```
---
- name: DHCP sur DS1
  hosts: DS1

  tasks:
    - name: Configure DHCP to Vlan 10
      cisco.ios.ios_config:
        parents: ip dhcp pool Vlan10
        lines:
        - network 10.0.10.0 255.255.255.0
        - default-router 10.0.10.1
        - dns-server 8.8.8.8

    - name: Configure DHCP to Vlan 30
      cisco.ios.ios_config:
        parents: ip dhcp pool Vlan20
        lines:
        - network 10.0.30.0 255.255.255.0
        - default-router 10.0.30.1
        - dns-server 8.8.8.8

    - name: Exclude addresses
      cisco.ios.ios_config:
        lines:
        - ip dhcp excluded-address 10.0.10.1
        - ip dhcp excluded-address 10.0.10.250
        - ip dhcp excluded-address 10.0.30.1
        - ip dhcp excluded-address 10.0.30.250
```

Figure IV. 6 – Configuration d'un serveur DHCP via Ansible.

IV.3.3 Mise en œuvre du protocole Spanning-Tree

Nous avons opté pour la version PVST+ du protocole Spanning-Tree, car elle offre une prise en compte des VLANs dans ses fonctionnalités. De plus, nous avons pris la décision de modifier les priorités sur les commutateurs au niveau de la couche du noyau effondré du réseau afin de les désigner comme commutateurs racine, Les instances VLANs 10 et 30 Pour DS1 et le DS2 pour les instances restantes. Cette approche permet d'équilibrer la répartition des charges dans le réseau local. Parallèlement, nous avons également pris des mesures pour réduire le temps de convergence du STP. Nous avons mis en œuvre la fonctionnalité Port-Fast sur les commutateurs au niveau de la couche d'accès, spécifiquement sur les ports connectés aux terminaux.

La figure IV.7 présente un exemple des commandes CLI qui doivent être exécutées sur DS1 et AS1 afin de permettre cette approche.

```
DS1(config)# spanning-tree mode rapid-pvst
DS1(config)# spanning-tree VLAN 10,30 root primary
DS1(config)# spanning-tree VLAN 50,70 root secondary

AS1(config)# interface range g0/2-3
AS1(config-if-range)# spanning-tree portfast
AS1(config-if-range)# exit
```

Figure IV. 7 – Configuration CLI du protocole STP sur DS1 et AS1.

Néanmoins, la Figure IV.8 présente une section de la Playbook que nous avons utilisée pour activer le comportement mentionné précédemment.

```
---
- name: PVST on DS1
  hosts: DS1

  tasks:
    - name: Configure rapid-pvst
      cisco.ios.ios_config:
        lines:
        - spanning-tree mode rapid-pvst
        - spanning-tree vlan 10,30 root primary
        - spanning-tree vlan 50,70 root secondary
```

Figure IV. 8 – Configuration du STP via Playbook Ansible.

Afin de vérifier que notre configuration opère conformément à nos attentes, nous utilisons la commande "**show spanning-tree summary**". La Figure IV.9 présente le résultat obtenu suite à l'exécution de cette commande sur DS1.

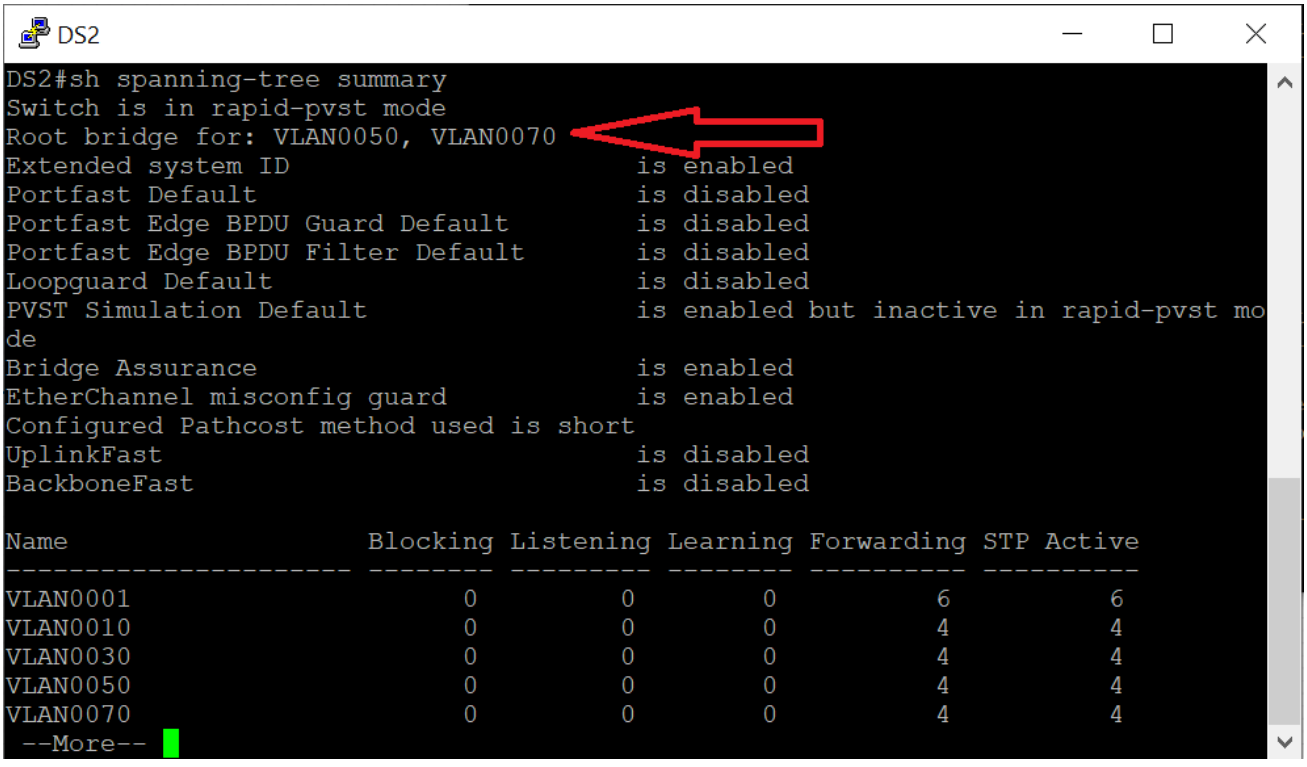


Figure IV. 9 – État STP sur DS2.

IV.3.4 Mise en œuvre du protocole PAgP

La configuration du protocole d'agrégation des liens PAgP est relativement simple par rapport à d'autres protocoles. Il suffit d'exécuter les commandes CLI répertoriées dans la figure IV.10 sur les deux extrémités de la liaison.

```
DS1(config)# interface range g1/0, g1/2
DS1(config-if)# channel-group 1 mode desirable
```

Figure IV. 10 – Configuration du protocole PAGP sur DS1.

La Figure IV.11 présente la Playbook qui permet de configurer l'agrégation des liens avec le protocole PAgP.

```
---
- name: interface-Pagp
  hosts: DS1

  tasks:
    - name: Configure Etherchannel
      cisco.ios.ios_lag_interfaces:
        config:
        - name: 1
          members
          - member: GigabitEthernet1/0
            mode: desirable
          - member: GigabitEthernet1/2
            mode: desirable
        state: merged
```

Figure IV. 11 – Configuration du protocole PAgP via Playbook Ansible.

En exécutant la commande "**show etherchannel summary**", l'état de ce protocole est affiché. Le résultat de cette commande sur DS1 démontre la réussite de notre configuration, comme illustré dans la Figure IV.12.

```
DS1#show etherchannel summary
Flags:  D - down        P - bundled in port-channel
        I - stand-alone s - suspended
        H - Hot-standby (LACP only)
        R - Layer3      S - Layer2
        U - in use      N - not in use, no aggregation
        f - failed to allocate aggregator

        M - not in use, minimum links not met
        m - not in use, port not aggregated due to minimum links not met
        u - unsuitable for bundling
        w - waiting to be aggregated
        d - default port

        A - formed by Auto LAG


Number of channel-groups in use: 1
Number of aggregators:           1

Group  Port-channel  Protocol    Ports
------+-------------+-----------+-----------------------------------------------
1      Po1(SU)         PAgP      Gi1/0(P)    Gi1/2(P)
```

Figure IV. 12 – État du protocole PAgP sur DS1.

IV.3.5 Mise en œuvre du protocole HSRP

Nous allons mettre en place quatre instances HSRP en fonction des VLANs utilisés dans la topologie. L'objectif principal était de manipuler les priorités de ces instances afin que DS1 achemine le trafic pour la moitié des VLANs tandis que DS2 se charge de l'autre moitié. Afin de mettre en œuvre le protocole HSRP, il sera nécessaire d'attribuer une adresse virtuelle (VIP) à chaque instance. Ces adresses seront utilisées comme passerelles par défaut pour les terminaux. Cette adresse virtuelle prendra de même la forme 10.0. X.3. De plus, nous mettrons en place un suivi (Track) sur les interfaces reliant les commutateurs de la couche du noyau effondré à l'extérieur du réseau local. Ceci nous permettra de maintenir la connectivité avec les réseaux distants.

Nous allons créer les instances HSRP pour les interfaces VLANs, Puis nous allons augmenter la priorité à 110 sur DS1 pour les instances 10 et 30, Nous suivrons la même approche sur DS2, mais cette fois-ci pour les instances 50 et 70. Ensuite, nous procéderons à l'activation du suivi (Track) sur les interfaces qui connectent au-delà du LAN, et en cas de défaillance de l'une de ces interfaces, nous réduirons la priorité de 20.

La figure IV.13 présente les commandes CLI essentielles à exécuter sur DS1.

```
DS1(config)# interface VLAN 10
DS1(config-if)# standby 10 ip 10.0.10.3
DS1(config-if)# standby 10 priority 110
DS1(config-if)# standby 10 preempt
DS1(config-if)# standby 10 track g0/3 20
DS1(config-if)# exit
DS1(config)# interface VLAN 30
DS1(config-if)# standby 30 ip 10.0.30.3
DS1(config-if)# standby 30 priority 120
DS1(config-if)# standby 30 preempt
DS1(config-if)# standby 30 track g0/3 30
DS1(config-if)# exit
DS1(config)# interface VLAN 50
DS1(config-if)# standby 50 ip 10.0.50.3
DS1(config-if)# standby 50 preempt
DS1(config-if)# exit
DS1(config)# interface VLAN 70
DS1(config-if)# standby 70 ip 10.0.70.3
DS1(config-if)# standby 70 preempt
DS1(config-if)# exit
```

Figure IV. 13 – Configuration du protocole HSRP sur DS1.

Cependant, la figure IV.14 illustre une partie de la Playbook que nous avons utilisée pour mettre en place le protocole HSRP.

```
---
- name: HSRP on DS1
  hosts: DS1

  tasks:
    - name: Create Track on G0/3
      cisco.ios.ios_config:
        lines:
        - track 1 interface gigabitethernet0/3 line-protocol

    - name: Configure HSRP on vlan 10
      cisco.ios.ios_config:
        parents: interface vlan 10
        lines:
        - standby 10 ip 10.0.10.3
        - standby 10 priority 110
        - standby 10 preempt
        - standby 10 track 1 decrement 20

    - name: Configure HSRP on vlan 30
      cisco.ios.ios_config:
        parents: interface vlan 30
        lines:
        - standby 30 ip 10.0.30.3
        - standby 30 priority 110
        - standby 30 preempt
        - standby 30 track 1 decrement 20
```

Figure IV. 14 – Configuration du HSRP via Ansible sur DS1.

En exécutant la commande "**show standby brief**", l'état des instances du protocole HSRP est affiché. Le résultat de cette commande sur DS1 démontre la réussite de notre configuration, comme illustré dans la Figure IV.15.

```
DS1#show standby brief
                     P indicates configured to preempt.
                     |
Interface   Grp  Pri P State   Active          Standby         Virtual IP
Vl10        10   110 P Active  local           10.0.10.2       10.0.10.3
Vl30        30   110 P Active  local           10.0.30.2       10.0.30.3
Vl50        50   100 P Standby 10.0.50.2       local           10.0.50.3
Vl70        70   100 P Standby 10.0.70.2       local           10.0.70.3
DS1#
```

Figure IV. 15 – État HSRP sur DS1.

IV.4 Mise en œuvre d'une solution de communication STP - HSRP.

Une fois que les protocoles et techniques nécessaires ont été mis en place pour garantir le bon fonctionnement de notre réseau, et suite à une observation du fonctionnement du réseau, nous avons remarqué une augmentation considérable de la latence lorsque le lien entre les commutateurs de la couche d'accès et le commutateur actif en HSRP est bloqué. Dans l'objectif d'améliorer l'efficacité de notre réseau, notre solution consiste à établir une communication entre les protocoles Spanning-Tree et HSRP en utilisant le serveur d'automatisation. Nous allons commencer par créer les Playbooks qui récupèrent les états des protocoles Spanning-Tree et HSRP. Ensuite, nous élaborerons les Playbooks qui apporteront des modifications aux priorités des commutateurs et aux coûts des liens. Enfin, nous écrirons un script Shell qui orchestrera ces Playbooks.

IV.4.1 Création des Playbook de récupération des états des protocoles STP/ HSRP

La vérification des états des périphériques à l'aide des commandes "show" est une étape essentielle. Ces commandes fournissent des informations cruciales sur la configuration et le fonctionnement des périphériques. Dans notre cas, nous nous concentrons sur les protocoles Spanning-Tree et HSRP. Pour détecter les liens bloqués par STP, nous utiliserons la commande "show spanning-tree blockedports" sur les commutateurs de la couche d'accès. Cette commande nous permettra d'identifier les ports bloqués par le protocole STP. Quant à HSRP, nous utiliserons la commande "show standby brief" sur les commutateurs de la couche du noyau effondré. Cette commande nous permettra de repérer le commutateur actif en HSRP pour chaque instance de VLAN.

Pour effectuer la vérification via le serveur d'automatisation, nous allons mettre en œuvre des Playbooks qui exécutent les tâches mentionnées précédemment et enregistrent les résultats dans des fichiers au format texte ".txt". L'illustration de la Figure IV.16 donne un aperçu du Playbook dédié à la récupération de l'état du protocole Spanning-Tree.

```
---
 - name: Show STP state sur AS1
   hosts: AS1

   tasks:
    - name: Show STP
      ios_command:
        commands:
        - show spanning-tree blockedports
      register: config

    - name: save output
      copy:
        content: "{{ config.stdout | replace('\\n', '\n') }}"
        dest: "STP_AS1.txt"
```

Figure IV. 16 – Récupération d'état du protocole STP via Une Playbook.

Les fichiers résultant au format ".txt" seront analysés par un script Python qui fournira les informations requises pour le bon fonctionnement du script Shell.

Nous allons maintenant procéder à la création des Playbooks qui permettent de modifier les paramètres STP.

Les ajustements nécessaires aux paramètres du protocole Spanning-Tree dépendent du comportement spécifique du réseau. Parfois, il suffit de modifier les priorités des commutateurs pour adapter ce comportement, tandis que dans d'autres cas, il est également nécessaire de modifier les coûts des liens. Afin d'éviter de créer une Playbook spécifique pour chaque situation, nous avons développé un script Python qui génère dynamiquement la Playbook nécessaire en fonction du comportement actuel du réseau. Ce script requiert plusieurs paramètres en entrée, tels que le type de modification, le nom d'hôte (Hostname) et le numéro de VLAN. La partie de script Python illustrée dans la figure IV.17 donne un aperçu de son fonctionnement.

```
f2 = open("modify_cost.yml","a")
t =1

for l in line1:
  elt = str(l).split()
  if  len(elt) > 3 :
    if elt[0] !="[u'Port":
      if elt[1] != "Vlans":
        if elt[0] == blk_int:
          f2.write(

     "    - name: Decrease cost on "+elt[0]+"\n"
    +"      cisco.ios.ios_config:"+"\n"
    +"        parents: interface "+elt[0]+"\n"
    +"        lines:"+"\n"
    +"        - spanning-tree vlan "+ vlan +" cost 4"+"\n \n"
    )
        else:
          f2.write(
     "    - name: Increase cost on "+elt[0]+"\n"
    +"      cisco.ios.ios_config:"+"\n"
    +"        parents: interface "+elt[0]+"\n"
    +"        lines:"+"\n"
    +"        - spanning-tree vlan "+ vlan +" cost 1000"+"\n \n"
```

Figure IV. 17 – Partie du script de création dynamique des Playbooks.

IV.4.2 Implémentation du script Shell

Dans le but d'assurer la coordination entre les Playbooks de récupération d'état et de modification du comportement du réseau, nous avons développé un script Shell qui effectue une surveillance continue et intervient pour optimiser le réseau en cas de comportement inapproprié. Le script commence par exécuter les Playbooks de vérification de l'état des protocoles STP et HSRP, ce qui génère deux fichiers ".txt". Ces fichiers seront ensuite traités par un script Python qui détectera et renverra les informations sur les ports bloqués par le STP, le commutateur associé à ces ports (en utilisant le protocole CDP pour obtenir ces informations) et le commutateur actif en HSRP. Les informations obtenues seront ensuite analysées pour déterminer si le comportement du réseau est optimisé ou s'il nécessite des modifications. Si aucune modification n'est requise, le processus de vérification sera répété. Cependant, si des modifications sont nécessaires, un script Python sera exécuté avec les paramètres appropriés pour générer une Playbook de modification des paramètres STP. La Playbook de modification sera ensuite exécutée, et on observera que le comportement du réseau sera optimisé par la suite. Cette étape de vérification sera répétée indéfiniment afin d'ajuster le réseau en cas de comportement non optimal. Une partie de script Shell illustrée dans la figure IV.18 donne un aperçu de son fonctionnement.

```
#!/bin/bash

cd /etc/ansible/playbooks

ansible-playbook show_hsrp.yaml
ansible-playbook show_stp.yaml

echo "------------------------Phase 1--------------------------"

for vlan in 10 30 50 70
do
    echo "Vlan $vlan"

    hsrp=$(python hsrp_state.py $vlan 2>&1 >/dev/null)

    IFS=' '
    read -r -a tab <<< "$hsrp"

    state1=${tab[0]}
    name1=${tab[1]}
    state2=${tab[2]}
    name2=${tab[3]}
```

Figure IV. 18 – Une partie du script Shell.

Pour évaluer l'efficacité de la solution, nous allons prendre en considération un scénario dans lequel le commutateur DS2 sera actif en HSRP pour toutes les instances, tel qu'illustré dans la figure IV.19.

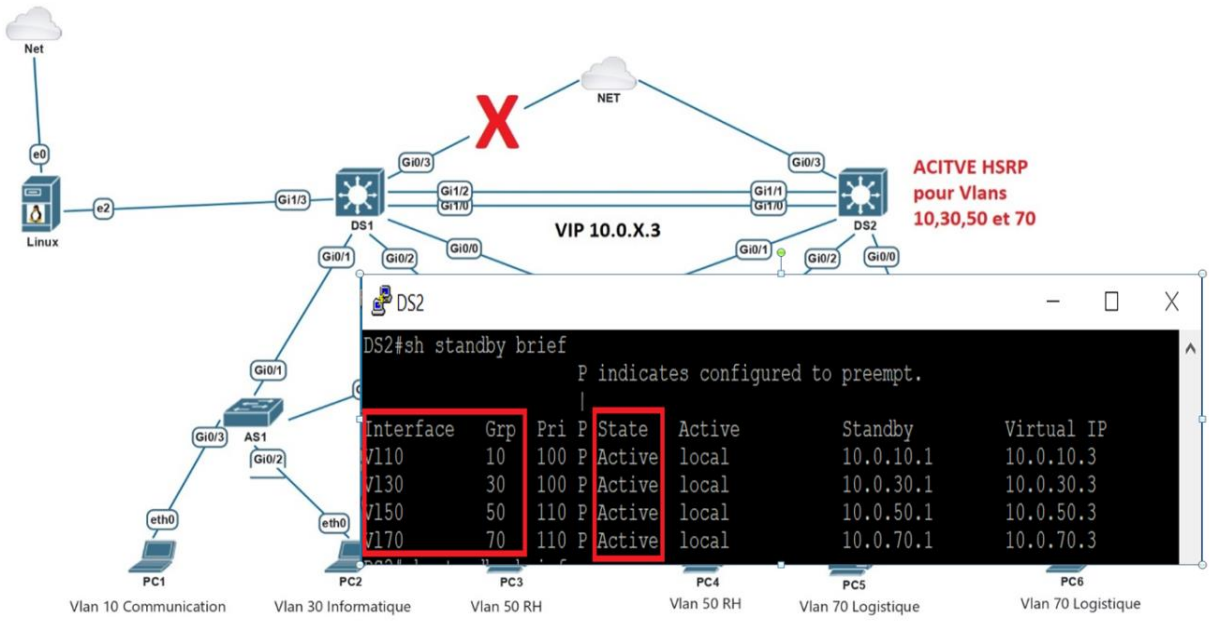


Figure IV. 19 – Etat du protocole HSRP.

Suite à l'exécution du script Shell, nous avons constaté une optimisation du réseau, où le port Gi0/1 du commutateur AS1 est maintenant bloqué pour toutes les instances de VLAN. Cette observation est étayée par la figure IV.20.

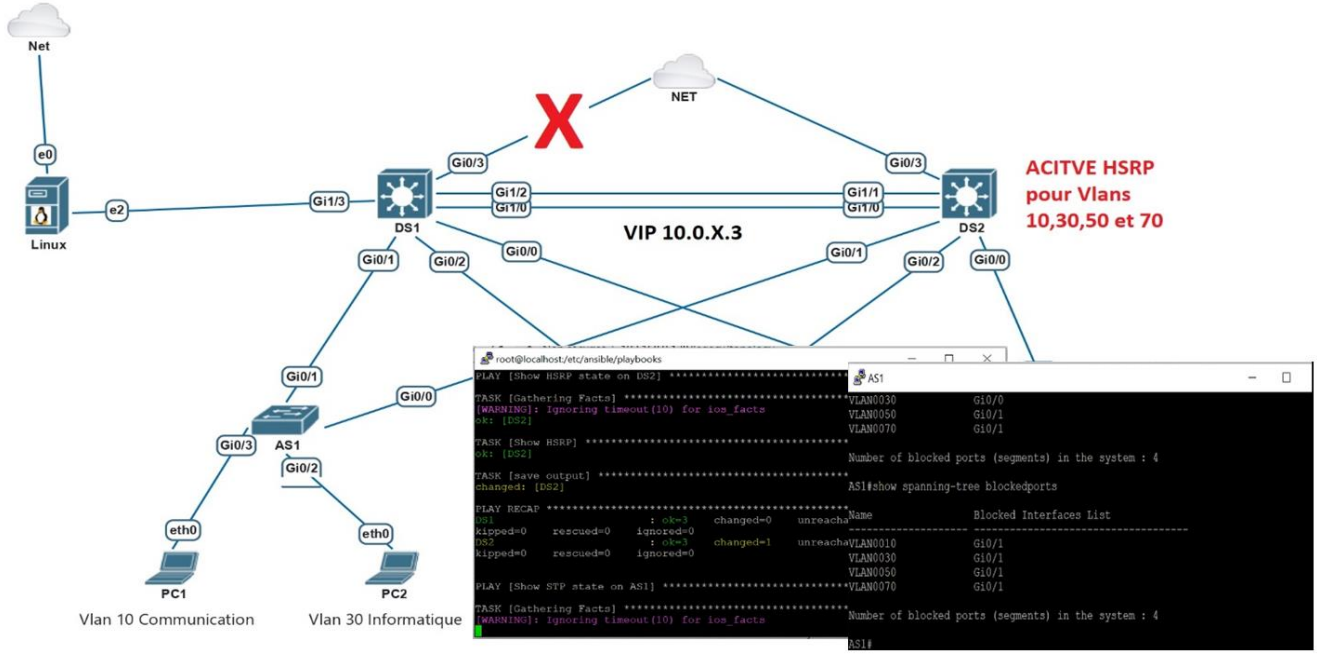


Figure IV. 20 – Ports bloqués sur AS1 après l'exécution du script Shell.

IV.4.3 Implémentation de l'interface graphique

L'interface graphique est extrêmement utile pour les administrateurs réseau, car sa présentation graphique simple et attrayante rend le travail des ingénieurs plus facile et stimulant, Dans cette optique, nous développons une interface graphique sous forme d'interface web, permettant de visualiser les états des différents protocoles et de les configurer facilement en quelques clics. Cette interface sera implémentée sur le serveur d'automatisation, offrant ainsi un contrôle complet sur notre infrastructure réseau.

Pour mettre en place notre interface graphique, nous avons opté pour l'utilisation des langages PHP, HTML et CSS, ainsi que du serveur web Apache. Cette interface graphique sera accessible via un navigateur web, que ce soit en interne ou en externe, en utilisant l'adresse IP du serveur ou en se connectant localement (localhost). Lorsqu'un utilisateur accède à cette adresse, il sera automatiquement redirigé vers la page d'accueil. À partir de là, il aura le choix entre accéder à la page de visualisation des états des protocoles pour surveiller le réseau, ou accéder à la page de configuration. La figure IV.21 offre un aperçu de la page d'accueil.

Figure IV. 21 – Interface principale.

Le menu dédié à la visualisation des états offre à l'administrateur réseau un accès aux informations essentielles. Il lui permet de consulter l'état du protocole STP, l'état du protocole HSRP et l'état des interfaces pour chaque commutateur. Cette fonctionnalité offre un aperçu détaillé de l'état actuel du réseau, fournissant ainsi des informations clés pour la gestion et le dépannage. Les figures ci-dessous illustrent cette section de l'interface graphique, offrant une représentation visuelle de ces informations importantes.

Figure IV. 22 – Menu d'affichage des états des protocoles.

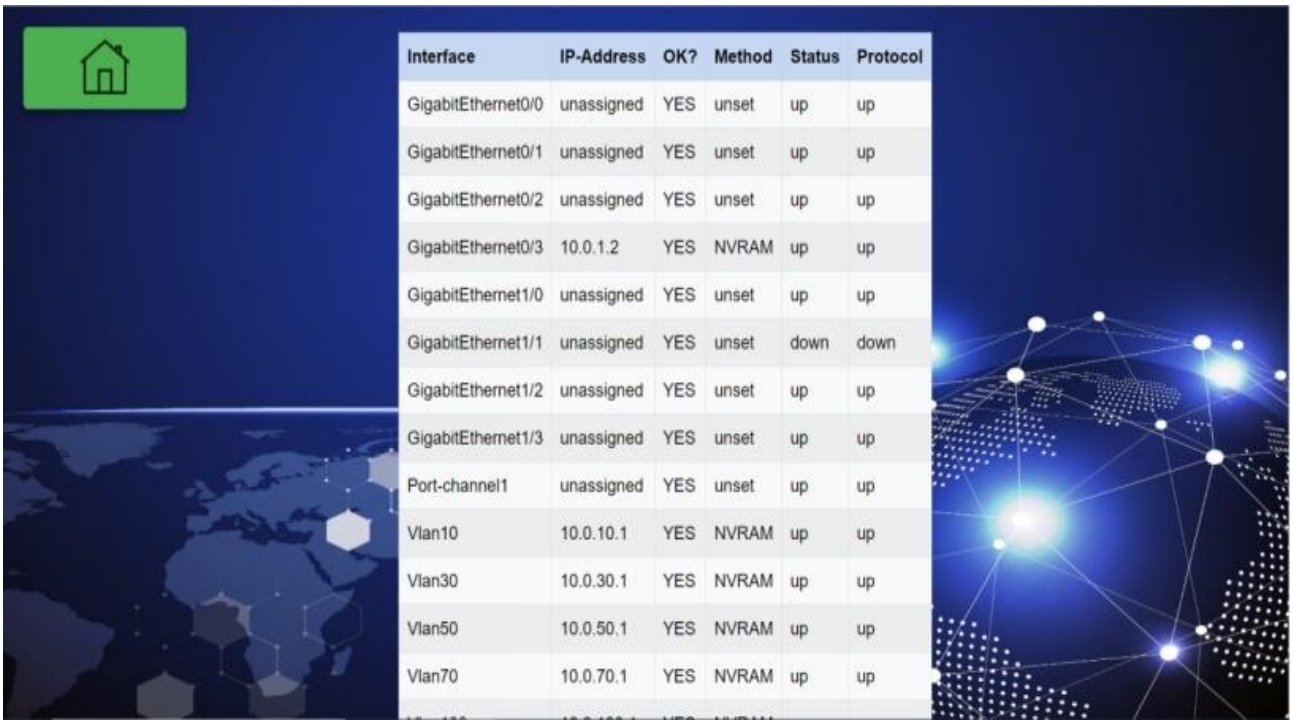
Figure IV. 23 – Port bloque du protocole STP sur AS1.

La figure IV.23 illustre une situation où le protocole Spanning Tree (STP) bloque un port spécifique sur le commutateur AS1.

La figure ci-dessous présente l'état du protocole HSRP sur DS2. La table affiche les informations relatives aux groupes HSRP présents sur l'interface DS2. Chaque groupe est identifié par un numéro de groupe (Grp) et est configuré avec une priorité (Pri). La colonne "State" indique le statut du protocole HSRP sur l'état active ou standby.

Interface	Grp	Pri	State	Active	Standby	IP Virtual
Vl10	10	100	Standby	10.0.10.1	local	10.0.10.3
Vl30	30	100	Standby	10.0.30.1	local	10.0.30.3
Vl50	50	110	Active	local	10.0.50.1	10.0.50.3
Vl70	70	110	Active	local	10.0.70.1	10.0.70.3

Figure IV. 24 – Etat du protocole HSRP sur DS2.



Interface	IP-Address	OK?	Method	Status	Protocol
GigabitEthernet0/0	unassigned	YES	unset	up	up
GigabitEthernet0/1	unassigned	YES	unset	up	up
GigabitEthernet0/2	unassigned	YES	unset	up	up
GigabitEthernet0/3	10.0.1.2	YES	NVRAM	up	up
GigabitEthernet1/0	unassigned	YES	unset	up	up
GigabitEthernet1/1	unassigned	YES	unset	down	down
GigabitEthernet1/2	unassigned	YES	unset	up	up
GigabitEthernet1/3	unassigned	YES	unset	up	up
Port-channel1	unassigned	YES	unset	up	up
Vlan10	10.0.10.1	YES	NVRAM	up	up
Vlan30	10.0.30.1	YES	NVRAM	up	up
Vlan50	10.0.50.1	YES	NVRAM	up	up
Vlan70	10.0.70.1	YES	NVRAM	up	up

Figure IV. 25 – Etat des interfaces sur DS1.

La figure IV.25 présente l'état des interfaces sur DS1. La table affiche les informations concernant l'état actuel des interfaces sur le commutateur DS1. Chaque interface est présentée avec son nom et son état correspondant.

En outre, l'utilisateur dispose également d'options de configuration multiples à travers la section dédiée à la configuration (création des VLANs et configuration des serveur DHCP), Les figures ci-dessous illustrent cette partie de l'interface graphique.

Figure IV. 26 – Menu Configuration.

La création des VLANs est illustrée dans la figure IV.27. Cette section offre la possibilité de créer un VLAN sur l'ensemble des commutateurs en saisissant simplement le nom du VLAN et son identifiant.

Figure IV. 27 – Création VLAN.

La configuration d'un serveur DHCP est présente dans cette section, où l'utilisateur a la possibilité de sélectionner un commutateur et le configurer pour agir en tant que serveur DHCP. Cette fonctionnalité est illustrée dans la figure IV.28.

Figure IV. 28 – Configuration serveur DHCP.

IV.5 Conclusion

Dans ce chapitre, nous avons débuté en préparant l'environnement de travail, puis nous avons implémenté tous les protocoles et technologies discutés précédemment. Par la suite, nous avons expliqué en détail le processus d'implémentation de la solution, qui nous a permis de maintenir notre réseau constamment optimisé. Enfin, nous avons présenté l'interface graphique qui a été mise en place dans le serveur d'automatisation, et qui simplifiera grandement les tâches des administrateurs réseau.

Conclusion Générale

L'automatisation des réseaux représente une avancée majeure dans le domaine des technologies de l'information. Elle offre de nombreux avantages, tant du point de vue de l'efficacité opérationnelle que de la performance des réseaux. L'automatisation permet de simplifier et d'accélérer les tâches de configuration, de déploiement, de surveillance et de gestion des réseaux. Elle réduit la dépendance aux processus manuels sujets aux erreurs humaines, améliorant ainsi la fiabilité et la stabilité des réseaux.

Dans le cadre de notre projet, nous avons pu expérimenter et appliquer la technologie d'automatisation des réseaux en utilisant l'outil puissant qu'est Ansible. Notre démarche a commencé par la conception et la mise en place d'une architecture réseau à deux niveaux.

Ensuite, nous avons utilisé Ansible pour automatiser différentes tâches liées à la configuration, au déploiement et à la gestion du réseau. Cela nous a permis de simplifier et d'accélérer ces processus, en évitant les erreurs humaines et en améliorant la cohérence des configurations. Nous avons également utilisé Ansible pour surveiller l'état du réseau en temps réel, en collectant des informations sur les protocoles STP et HSRP, ainsi que sur les interfaces des commutateurs. Cela nous a donné une visibilité complète sur l'état du réseau et nous a permis de prendre des mesures correctives rapidement en cas de problème. Enfin nous avons développé une interface graphique conviviale pour visualiser et configurer les paramètres du réseau, Grace à cette interface, ils ont pu surveiller l'état du réseau et effectuer des modifications avec facilité et rapidité

En utilisant Ansible, nous avons réussi à optimiser le fonctionnement de notre réseau, en ajustant les priorités des commutateurs et les coûts des liens en fonction des besoins. Cela a permis d'améliorer les performances du réseau, d'optimiser l'utilisation des ressources et de garantir la disponibilité des services.

En conclusion Nous avons pu mettre en pratique les principes de l'automatisation pour simplifier les tâches, améliorer la fiabilité et optimiser les performances du réseau. L'utilisation d'Ansible comme outil d'automatisation s'est avérée efficace et nous a permis de réaliser des gains significatifs en termes d'efficacité opérationnelle.

Au cours de la mise en œuvre de ce projet, nous avons été confrontés à divers défis et obstacles. En raison du manque de documentation et de tutoriels de l'outil Ansible nous avons rencontré pas mal de problème notamment la connexion entre Ansible et les périphériques réseau de notre topologie. Lors de l'étape de déploiement, nous avons également rencontré des problèmes et des difficultés, ou l'image IOS des périphériques que nous avons utilisées n'étaient pas des images réelles, c'est juste pour émuler le fonctionnement réel des périphériques. Cependant, ces images présentaient plusieurs bugs qui ont entravé notre progression dans le projet.

Il existe des perspectives prometteuses pour améliorer notre travail à l'avenir, notamment :

- Enrichir l'interface graphique afin de pouvoir configurer les protocoles de routage et la technologie Etherchannel

- Une possibilité d'amélioration future consisterait à étendre les fonctionnalités de l'interface graphique pour permettre une configuration personnalisée des protocoles Spanning-Tree et HSRP.
- Implémentation de la solution de détection des nouveaux périphériques
- Enrichir l'interface graphique pour inclure la gestion des comptes utilisateurs et de leurs privilèges.

Liste des abréviations

- **AAP** Advanced Access Protocol
- **API** Application Programming Interface
- **ARP** Address Resolution Protocol
- **ATM** Asynchronous Transfer Mode
- **AVF** Active Virtual Forwarder
- **AVG** Active Virtual Gateway
- **AWX** Ansible WorX
- **AS** Access Switch
- **BGP** Border Gateway Protocol
- **BID** Bridged Identity
- **BIOS** Basic Input/Output System
- **BOOTP** Bootstrap Protocol
- **CDP** Cisco Discovery Protocol
- **CLI** Command-Line Interface
- **CLNP** Connectionless Network Protocol
- **CPU** Central Processing Unit
- **CSS** Cascading Style Sheets
- **DHCP** Dynamic Host Configuration Protocol
- **DSL** Domain Specific Language
- **DS** Distribution Switch
- **EIGRP** Enhanced Interior Gateway Routing Protocol
- **ERB** Embedded Ruby
- **FDDI** Fiber Distributed Data Interface
- **FHRP** First Hop Redundancy Protocol
- **FIX** Financial Information Exchange
- **FLSM** Fixed-Length Subnet Masking
- **FTP** File Transfer Protocol
- **GLBP** Gateway Load Balancing Protocol
- **GUI** Graphical User Interface
- **HDLC** High-level Data Link Control
- **HSRP** Hot Standby Router Protocol
- **HTML** HyperText Markup Language
- **HTTP** Hypertext Transfer Protocol
- **ICMP** Internet Control Message Protocol
- **ID** Identification
- **IETF** Internet Engineering Task Force
- **IGP** Interior Gateway Protocol

- **IGRP** Interior Gateway Routing Protocol
- **IMAP** Internet Message Access Protocol
- **IP** Internet Protocol
- **ISDN** Integrated Services Digital Network
- **IS-IS** Intermediate System to Intermediate System
- **ISO** International Organization for Standardization
- **LACP** Link Aggregation Control Protocol
- **LAN** Local Area Network
- **LLC** Logical Link Control
- **LSDB** Link State Database
- **MAC** Media Access Control
- **MAN** Metropolitan Area Network
- **MLF** Multi-Layer Forwarding
- **MPLS** Multiprotocol Label Switching
- **MSTP** Multiple Spanning Tree Protocol
- **MTU** Maximum Transmission Unit
- **NCP** Network Control Program
- **NFS** Network File System
- **OSI** Open Systems Interconnection
- **OSPF** Open Shortest Path First
- **PAgP** Port Aggregation Control Protocol
- **PAN** Personal Area Network
- **PDH** Plesiochronous Digital Hierarchy
- **POP3** Post Office Protocol version 3
- **PPP** Point-to-Point Protocol
- **PVST** Per-VLAN Spanning Tree
- **RARP** Reverse Address Resolution Protocol
- **RIP** Routing Information Protocol
- **RS-232** Recommended Standard 232
- **RSTP** Rapid Spanning Tree Protocol
- **RTP** Real-time Transport Protocol
- **SDH** Synchronous Digital Hierarchy
- **SFTP** SSH File Transfer Protocol
- **SMTP** Simple Mail Transfer Protocol
- **SONET** Synchronous Optical Networking
- **SPX** Sequenced Packet Exchange
- **SSH** Secure Shell
- **TCAP** Transaction Capabilities Application Part
- **TCP** Transmission Control Protocol
- **UDP** User Datagram Protocol
- **VIP** Virtual IP

- **VLAN** Virtual Local Area Network
- **VLSM** Variable-Length Subnet Masking
- **VPC** Virtual Private Cloud
- **VRRP** Router Redundancy Protocol
- **WAN** Wide Area Network
- **XNS** Xerox Network Systems
- **YAML** YAML Ain't Markup Language

Bibliographie

[2] A. L. Marwan Al-shawi, Designing for Cisco Network Service Architectures (ARCH) Foundation Learning Guide: CCDP ARCH 300-320, Cisco Press, Dec 30, 2016.

[3] A. Bruno and S. Jordan, CCDA 200-310 Official Cert Guide. Cisco Press, 2016. [Online]. Available: https://ptgmedia.pearsoncmg.com/images/9781587144547/samplepages/9781587144547.pdf

[6] J. Postel, "Internet Protocol," Sep. 1981. doi: 10.17487/rfc0791.

[9] W. M.Eddy, «RFC 9293: Transmission Control Protocol (TCP),» vol. 1, p. August, 2022

[14] "Cisco Networking Academy Program," Google Books. https://books.google.dz/books/about/Cisco_Networking_Academy_Program.html?id=wYtnQgAACAAJ&redir_esc=y (Accessed May 24, 2023).

[15] T. Lammle, CCENT ICND1 Study Guide: Exam 100-105. John Wiley & Sons, 2016.

[16] N. Kocharians, P. Paluch, and W. Odom, CCIE Routing and Switching V5.0 Official Cert Guide, Volume 1, Fifth Edition. Cisco Press, 2014.

[17] D. Savage, J. Ng, S. Moore, D. Slice, P. Paluch, and R. White, "Cisco's Enhanced Interior Gateway Routing Protocol (EIGRP)," www.rfc-editor.org, May 2016, doi: https://doi.org/10.17487/RFC7868.

[18] J. West, T. Dean, and J. Andrews, Network+ Guide to Networks. Cengage Learning, 2015

[19] "EXOLAB Découverte du Protocole Spanning Tree (STP -Spanning Tree Protocol)." [Online]. Available: https://www.reseaucerta.org/sites/default/files/Exolab-SpanningTree-1-enonce.pdf (Accessed May 24, 2023).

[20] R. Perlman, «An Algorithm for Distributed Computation of a Spanning Tree in an Extended LAN,» ACM SIGCOMM Computer Communication Review, vol. 15, 1985

[29] B. Edgeworth, Ramiro Garza Rios, J. Gooley, and D. Hucaby, CCNP and CCIE enterprise core ENCOR 350-401. San Jose, California: Cisco Press, 2020.

[33] W. Stallings et al., "CRYPTOGRAPHY AND NETWORK SECURITY PRINCIPLES AND PRACTICE SEVENTH EDITION GLOBAL EDITION." Available: https://www.cs.vsb.cz/ochodkova/courses/kpb/cryptography-and-network-security_-principles-and-practice-7th-global-edition.pdf

[34] J. Atwell, Network Automation: The Definitive Guide, O'Reilly Media, Inc., 2016.

[35] D. López, Network Automation and Orchestration: Automating and Programming Networks and Services, Springer, 2017.

[36] Gartner Says Network Automation Can Reduce Time to Deploy New Services by 40% to 90%, Gartner, 2018.

[37] The Cost of Data Center Outages, Ponemon Institute, 2016.

[39] IDC, "IDC Survey Reveals Network Automation and Agility as Key to Modernizing IT Infrastructure and Driving Digital Transformation," IDC, 2020

[49] “Ansible concepts — Ansible Documentation,” docs.ansible.com. https://docs.ansible.com/ansible/latest/getting_started/basic_concepts.html (Accessed May 25, 2023).

[50] “How to build your inventory — Ansible Documentation,” docs.ansible.com. https://docs.ansible.com/ansible/latest/inventory_guide/intro_inventory.html (Accessed May 25, 2023).

Webographie

[1] “Network Types,” Computer Talk. http://cmscomputertalk.weebly.com/network-types.html (Accessed Jun. 24, 2023).

[4] “Intro to Networking - Introduction to Virtual LANs (VLANs) and Tagging,” Ubiquiti Networks Support and Help Center. https://help.ui.com/hc/en-us/articles/222183968-Intro-to-Networking-Introduction-to-Virtual-LANs-VLANs-and-Tagging (Accessed May 24, 2023).

[5] “Types of VLAN | Function, Benefits and Different Types of VLAN,” EDUCBA, Jan. 08, 2020. https://www.educba.com/types-of-vlan/

[7] “Internet Protocol,” Cisco. https://www.cisco.com/c/en/us/td/docs/net_mgmt/prime/network/3-8/reference/guide/ip.html#wp1043468 (accessed Jun. 24, 2023).

[8] “Définition ICMP (Internet Control Message Protocol),” Actualité Informatique, Oct. 20, 2020. https://actualiteinformatique.fr/definition/definition-icmp-internet-control-message-protocol (Accessed Jun. 24, 2023).

[10] “Home,” IETF. https://www.ietf.org (Accessed Jun. 24, 2023).

[11] “What is TCP/IP in Networking? | Fortinet,” Fortinet. https://www.fortinet.com/resources/cyberglossary/tcp-ip#:~:text=Transmission%20Control%20Protocol%20(TCP)%20is,data%20and%20messages%20over%20networks.

[12] “Protocole DHCP,” FRAMEIP.COM, Mar. 13, 2023. https://www.frameip.com/dhcp/ (Accessed Jun. 24, 2023).

[13] “Le protocole DHCP,” apcpedagogie. https://apcpedagogie.com/le-protocole-dhcp/le-protocole-dhcp-2/ (Accessed May 24, 2023).

[21] «Spanning Tree Protocol (STP),» Cisco Goffinet, [En ligne]. Available: https://cisco.goffinet.org/ccna/redondance-de-liens/spanning-tree-rapid-stp-pvst-cisco/.(Accessed le 19 May 2023).

[22] P. Sahu, “Spanning Tree Protocol Operation,” networkgalaxy, Jan. 23, 2022. https://www.networkgalaxy.org/2015/06/spanning-tree-protocol-operation.html (Accessed May 24, 2023)

[23] «IEEE Standards Association,» IEEE, [En ligne]. Available: https://standards.ieee.org/ieee/802.1D/3387/.(Accès le 19 May 2023).

[24] «Spanning Tree et Rapid Spanning-tree Cisco,» Cisco Goffinet, [En ligne]. Available: https://cisco.goffinet.org/ccna/redondance-de-liens/spanning-tree-rapid-stp-pvst-cisco/. (Accessed le 19 May 2023)

[25] “Commscope Technical Content Portal.” https://docs.commscope.com/fr-FR/bundle/fastiron-08090-l2guide/page/GUID-F9905D20-F2BB-4286-B606-49BC36596CF1.html

[26] “Understand the Multiple Spanning Tree Protocol (802.1s),” Cisco, Dec. 07, 2022. https://www.cisco.com/c/en/us/support/docs/lan-switching/spanning-tree-protocol/24248-147.html

[27] F. Goffinet, “Spanning-Tree et Rapid Spanning-tree Cisco,” cisco.goffinet.org, Jan. 01, 2018. https://cisco.goffinet.org/ccna/redondance-de-liens/spanning-tree-rapid-stp-pvst-cisco/ (accessed Jul. 14, 2023).

[28] “Configuring Spanning Tree PortFast, BPDU Guard, BPDU Filter, UplinkFast, BackboneFast, and Loop Guard,” Cisco, Mar. 21, 2015. https://www.cisco.com/c/en/us/td/docs/switches/lan/catalyst4000/8-2glx/configuration/guide/stp_enha.html?utm_source=CAM#wp1046787(Accessed May 22, 2023)

[30] «Hot Standby Router Protocol (HSRP) and Virtual Router Redundancy Protocol (VRRP),» GeeksforGeeks, [En ligne]. Available: https://www.geeksforgeeks.org/hot standby-router-protocol-hsrp-virtual-router-redundancy-protocol-vrrp/. (Accès le 19 May 2023)

[32] «Cisco Etherchannel : configuration, vérification, dépannage,» Goffinet, [En ligne]. Available: https://cisco.goffinet.org/ccna/redondance-de-liens/cisco-etherchannel configuration-verification-depannage/. (Accessed le 19 May 2023).

[38] “Close the gap in your protection and automate security,” Fortinet. https://www.fortinet.com/solutions/enterprise-midsize-business/automate-security#:~:text=Every%20day%20we%20detect%20and%20block%20attacks%2C%20but (Accessed May 25, 2023).

[40] “Agent Vs Agentless Configuration Management,” ServerTribe, Jul. 29, 2021. https://www.servertribe.com/agent-vs-agentless-configuration-management/ (Accessed May 24, 2023).

[41] P. Leach and clairecadman, “What is Puppet?,” puppet.com. https://www.puppet.com/docs/puppet/8/what_is_puppet.html (Accessed May 24, 2023).

[42] “What Is Puppet Software and How Do You Use It?,” Liquid Web, Jun. 09, 2022. https://www.liquidweb.com/kb/what-is-puppet/ (Accessed May 25, 2023).

[43] Jessie, “Configuration Management Tools - Ansible, Chef, Puppet,” Study CCNA, Sep. 17, 2021. https://study-ccna.com/configuration-management-tools-ansible-chef-puppet/ (Accessed May 25, 2023).

[45] J. Walter, “Ansible contre. Chef : l’outil d’automatisation informatique qui vous convient,” Geekflare, Apr. 19, 2023. https://geekflare.com/fr/ansible-vs-chef/ (Accessed May 25, 2023).

[46] M. C. T. V. Age, “Ansible Architecture | Ansible Automation,” Technology Focused Hub, Jul. 21, 2022. https://network-insight.net/2022/07/21/ansible-architecture-ansible-automation/#Ansible%20Architecture:%20The%20Drive%20For%20Automation (Accessed May 25, 2023).

[47] “Chef Vs Puppet Vs Ansible - Comparison of DevOps Management Tools,” Veritis. https://www.veritis.com/blog/chef-vs-puppet-vs-ansible-comparison-of-devops-management-tools/ (Accessed May 25, 2023).

[48] X. Steampunk, “Why choose Ansible for your IT automation | XLAB Steampunk blog,” steampunk.si. https://steampunk.si/blog/why-choose-ansible-for-it-automation/# (Accessed May 25, 2023).

[51] S. Doran, “Security is Hard. Why Not Automate It?” www.ansible.com. https://www.ansible.com/blog/security-automation (Accessed May 25, 2023).