N°d'ordre: .....

République Algerienne Démocratique et Populaire
Ministère de l'enseignement supérieur et de la Recherche scientifique

Université Djillali Liabès de Sidi Bel Abbès
Faculté des Sciences Exactes
Département d'informatique

Mémoire de Master

Domaine : Mathématiques-informatique
Filière : Informatique
Spécialité : Ingénierie des Systèmes d'Information et du Logiciel

Par

Mr DIB Abdelkrim Yassine Taki-Eddine

Conception et implémentation du protocole DHCP : Approche utilisant le langage de programmation Rust

Soutenu le ..-..-2023 devant le jury :

Pr.	NOM JURY	Université jury	Président du jury
Pr.	NOM JURY	Université jury	Rapporteur
Pr.	NOM JURY	Université jury	Examinateur
Pr.	NOM JURY	Université jury	Examinateur
Pr.	BOUKLI-HACÉNE Sofiane	.....	Directeur de thèse
Pr.	.....	UDL-SBA	Co-Directeur de thèse

Année Universitaire : 2022 - 2023

Je dédie ce modeste travail à : Mes très chers parents pour tous ses sacrifices et grâce à vous je n'ai manqué de rien « MERCI »

Table des matières

Liste des figures

Liste des tableaux

Préface, Introduction...

Dans les milieux industriels comme ...

L'objectif de cette thèse a été de ...

Nos contributions portent sur : ...

Le *premier chapitre* expose la problématique de la thèse.
Le *deuxième chapitre* présente en détail le modèle utilisé.

etc.
Cette thèse a fait l'objet de divers travaux écrits : ...

Introduction 1

Sommaire

Notions de Base et Généralités 2

Sommaire

2.1 Architecture Client-Serveur et Modèles de Communication

2.1.1 Définition des Systèmes Distribués

Diverses définitions des systèmes distribués ont été proposées dans la littérature, sans qu'aucune ne soit pleinement satisfaisante ni unanimement acceptée. Pour notre propos, nous pouvons les caractériser de manière simple : "Un système distribué est un ensemble d'ordinateurs indépendants qui apparaissent à leurs utilisateurs comme un système unique et cohérent." [01]

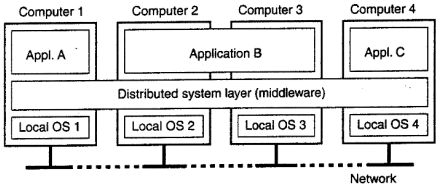

Figure 2.1 – *Description de l'image*

2.1.2 Buts des Systèmes Distribués

Un système distribué vise à répondre à plusieurs objectifs fondamentaux :

1. **Accessibilité des ressources :**
   Un système distribué permet d'accéder à des ressources distantes de manière efficace et transparente (imprimantes, stockage, fichiers, réseaux, etc.). Cela favorise la collaboration et réduit les coûts, mais soulève des problèmes de sécurité tels que les intrusions, l'interception de données et la confidentialité.
2. **Transparence de distribution :**
   Le système doit masquer la distribution physique des ressources et processus pour offrir une expérience fluide à l'utilisateur. Différents types de transparence existent :
   — **Accès :** Cache les différences dans les méthodes d'accès aux données.
   — **Localisation :** Masque l'emplacement physique des ressources.
   — **Migration :** Dissimulation du déplacement des ressources.
   — **Relocalisation :** Cache le changement d'emplacement d'une ressource en cours d'utilisation.
   — **Réplication :** Masque la duplication des ressources pour une meilleure disponibilité.
   — **Concurrence :** Gère le partage des ressources entre plusieurs utilisateurs.
   — **Défaillance :** Masque les pannes et facilite la récupération du système.
3. **Ouverture :** Un système distribué doit suivre des protocoles standards pour garantir l'interopérabilité et permettre l'intégration de nouveaux composants sans modification majeure.

4. **Scalabilité :** La capacité du système à s'adapter à une augmentation du nombre d'utilisateurs et de la charge de travail sans dégradation des performances.

2.1.3 Définition de l'Architecture Client-Serveur

L'architecture client-serveur est un modèle informatique où plusieurs clients (ordinateurs distants) envoient des requêtes et reçoivent des services d'un serveur centralisé (ordinateur hôte). Les ordinateurs clients offrent une interface permettant à l'utilisateur de demander des services, tandis que le serveur attend les requêtes et y répond. Idéalement, un serveur offre une interface standardisée et transparente aux clients, rendant inutile la connaissance des détails techniques du système sous-jacent. [03] Les clients peuvent exécuter des applications locales tout en demandant des services au serveur, qui peut être une base de données, un serveur de fichiers ou un serveur web. Ce modèle se distingue du modèle mainframe, où un ordinateur central traitait toutes les opérations pour des terminaux passifs.

2.1.4 Principe de l'Architecture Client-Serveur

Malgré l'absence de consensus sur plusieurs aspects des systèmes distribués, la plupart des chercheurs et professionnels s'accordent à dire que l'approche client-serveur facilite la gestion de leur complexité. Dans le modèle client-serveur de base :

— Un serveur implémente un service particulier (ex. base de données, système de fichiers).
— Un client envoie une requête et attend la réponse du serveur.
— Cette interaction est appelée comportement requête-réponse.

Modèles de Communication Client-Serveur

Deux principaux protocoles de communication sont utilisés entre clients et serveurs :

1. **Protocole sans connexion**
   — Utilisé lorsque le réseau est fiable (ex. LAN).
   — Le client envoie un message de requête sans établir de connexion préalable.
   — Le serveur traite la requête et envoie une réponse.
   — Avantage : rapide et efficace.
   — Inconvénient : difficile à gérer en cas de perte de message (problème de retransmission et duplication).
   — Certaines opérations peuvent être répétées sans effet indésirable (opérations idempotentes).

2. **Protocole orienté connexion**
   — Privilégié pour les communications peu fiables (ex. WAN, Internet).
   — Basé sur des connexions fiables comme TCP/IP.
   — Avant d'envoyer une requête, le client établit une connexion avec le serveur.
   — Le serveur utilise cette connexion pour envoyer la réponse.
   — Avantage : fiabilité accrue.
   — Inconvénient : mise en place et destruction de la connexion coûteuse en ressources.

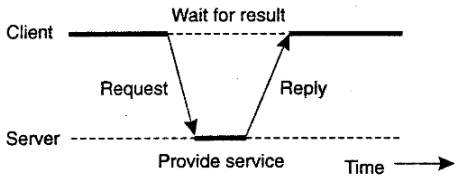

FIGURE 2.2 – *Description de l'image*

2.1.5 Comparaison avec d'autres modèles

Définitions

— **Systm̀es Peer-to-Peer :** Un système Peer-to-Peer (P2P) est un système auto-organisé constitué d'entités égales et autonomes (pairs). Il vise à permettre le partage des ressources distribuées dans un environnement réseau en évitant les services centralisés [05].
— **Edge Computing :** Edge computing est un cadre de calcul distribué dont les infrastructures résident à la périphérie des réseaux. La périphérie ("edge") est un type d'appareil réseau qui connecte différents réseaux et peut souvent contrôler l'équipement sous-jacent de l'infrastructure réseau [06, 07].
— **Cloud Computing :**Un modèle permettant un accès réseau pratique et à la demande à un ensemble partagé de ressources informatiques configurables (réseaux, serveurs, stockage, applications et services) qui peuvent être provisionnées et libérées rapidement avec un effort de gestion minimal ou une interaction avec le fournisseur de services [08, 09].

Comparaison

La comparaison entre les différents modèles de calcul est présentée dans le tableau suivant :

black

cyan!30 **Critère**	**Client-Serveur**	**Peer-to-Peer (P2P)**	**Edge Computing**	**Cloud Computin**
Architecture	Centralisée	Décentralisée	Semi-décentralisée	Centralisée
Gestion des ressources	Serveur principal	Distribuée entre pairs	Localement sur les appareils	Serveurs distants
Latence	Moyenne	Variable (peut être élevée)	Faible	Peut être élevée s lon la distance
Scalabilité	Limité par le serveur	Très élevée	Moyenne	Très élevée
Sécurité	Contrôlée par le serveur	Difficile à gérer	Variable selon les appareils	Dépend du fou nisseur cloud
Exemples	Applications web, bases de données	Partage de fichiers (BitTorrent), blockchain	IoT, voitures autonomes	Google Driv AWS

TABLE 2.1 – *Comparaison des architectures Client-Serveur, P2P, Edge Computing et Cloud Computing*

2.1.6 Avantages et Inconvénients de l'Architecture Client-Serveur

black

cyan!30 **Critères**	**Avantages**	**Inconvénients**
Simplicité	Modèle structuré et bien défini	Dépendance à un serveur central
Sécurité	Contrôle centralisé des accès et des données	Risque de panne du serveur (point unique de défaillance)
Performance	Optimisé pour gérer un grand nombre de clients simultanés	Peut devenir un goulot d'étranglement en cas de surcharge
Maintenance	Administration centralisée facilitant les mises à jour et corrections	Les mises à jour peuvent affecter l'ensemble des clients
Scalabilité	Peut être amélioré en ajoutant des serveurs plus puissants	Expansion coûteuse (besoin d'infrastructure supplémentaire)
Gestion des ressources	Allocation efficace des ressources sur le serveur	Les clients dépendent des capacités du serveur
Sécurité des données	Sauvegarde et protection des données sur un serveur central	Vulnérabilité aux cyberattaques sur le serveur
Exemples d'utilisation	Web, bases de données, applications métiers	Peu adapté aux applications nécessitant une grande décentralisation (ex : blockchain)

TABLE 2.2 – *Avantages et Inconvénients de l'Architecture Client-Serveur*

2.1.7 Applications et Cas d'Usage de l'Architecture Client-Serveur dans la Gestion des Adresses IP

L'architecture client-serveur est largement utilisée pour gérer les adresses IP dans les réseaux d'entreprise, les fournisseurs d'accès Internet et les infrastructures cloud. Voici quelques applications et cas d'usage concrets :

Serveurs DHCP (Dynamic Host Configuration Protocol)

Le serveur DHCP permet d'attribuer dynamiquement des adresses IP aux clients lorsqu'ils se connectent au réseau [10]. Il est utilisé dans les réseaux locaux d'entreprise et par les fournisseurs d'accès Internet (FAI).

p4cm | X |
cyan !30 **Catégorie Détails**
Virtualisation des réseaux Des technologies comme SDN (Software-Defined Networking) et NFV (Network Function Virtualization) permettent une gestion plus flexible et évolutive des infrastructures réseau [88].
Gestion des IP dans les environnements cloud Des plateformes comme AWS, Azure et OpenStack intègrent des solutions de gestion dynamique des adresses IP, facilitant le provisionnement des services [89].
Sécurité et segmentation des réseaux virtuels Les réseaux virtuels requièrent des mesures avancées de segmentation et d'isolation pour prévenir les attaques et garantir la sécurité des données [90].
Impact sur la scalabilité et la gestion des ressources L'automatisation et l'orchestration des adresses IP améliorent la scalabilité des infrastructures cloud et optimisent l'allocation des ressources réseau [91].

TABLE 2.3 – *Tableau comparatif sur l'impact du Cloud et de la virtualisation*

black 2whitegray !10

cyan !30**Avantages**	cyan !30**Inconvénients**
cyan !30Gestion centralisée et automatique des IP	cyan !30Dépendance au serveur DHCP
cyan !30Évite les conflits d'adresses	cyan !30Risque de saturation en cas de panne
cyan !30Adapté aux réseaux évolutifs	cyan !30

TABLE 2.4 – *Serveurs DHCP (Dynamic Host Configuration Protocol)*

Serveurs DNS (Domain Name System)

Le serveur DNS convertit les noms de domaine en adresses IP pour permettre aux utilisateurs d'accéder aux services Internet [11]. Il est utilisé par les entreprises, les fournisseurs cloud et les services web.

Réseaux d'Entreprise et Gestion des Équipements

Cette application consiste à attribuer et gérer les adresses IP aux postes de travail, imprimantes, caméras IP et serveurs internes. Elle repose sur l'utilisation de serveurs centralisés (Active Directory, DHCP, DNS) [12].

black 2whitegray !10

cyan !30**Avantages**	cyan !30**Inconvénients**
cyan !30Accélère l'accès aux services en ligne	cyan !30Vulnérable aux attaques (ex : DNS Spoofing)
cyan !30Gère efficacement les adresses IP publiques et privées	cyan !30Charge élevée sur les serveurs en cas de forte demande

Table 2.5 – *Serveurs DNS (Domain Name System)*

black 2whitegray !10

cyan !30**Avantages**	cyan !30**Inconvénients**
cyan !30Sécurisation et surveillance des connexions	cyan !30Configuration complexe pour les grandes entreprises
cyan !30Administration simplifiée des équipements	cyan !30Risque de saturation des serveurs

Table 2.6 – *Réseaux d'Entreprise et Gestion des Équipements*

L'architecture client-serveur est indispensable pour la gestion des adresses IP, assurant une attribution efficace, une sécurisation du réseau et une gestion centralisée des ressources.

2.2 Protocoles Réseaux et Modèles de Communication

2.2.1 Modèle OSI vs. Modèle TCP/IP

Présentation du modèle OSI

Le modèle OSI (Open Systems Interconnection) est une norme conceptuelle élaborée par l'ISO pour structurer la communication en réseaux en sept couches distinctes. Chaque couche a une fonction spécifique facilitant l'interopérabilité entre les systèmes hétérogènes [13].

Présentation du modèle TCP/IP

Le modèle TCP/IP repose sur une approche pragmatique avec quatre couches fonctionnelles. Il est utilisé principalement sur Internet et repose sur des protocoles standards comme TCP et IP [14].

Comparaison des deux modèles

2whitegray!15

cyan!20**Critère**	cyan!20**Modèle OSI**	cyan!20**Modèle TC**
cyan!20Nombre de couches	cyan!207	cyan!204
cyan!20Développement	cyan!20Théorique, conçu par l'ISO	cyan!20Pratique, conçu par
cyan!20Utilisation	cyan!20Référence académique	cyan!20Utilisé sur In
cyan!20Flexibilité	cyan!20Rigide avec séparation stricte	cyan!20Plus flexible et a

Table 2.7 – *Comparaison des modèles OSI et TCP/IP*

2.2.2 Rôle des couches réseau dans la communication

Voila la table qui presente le role des couches réseau dans la communication

2whitegray!15

cyan!20**Couche**	cyan!20**Fonction principale**
cyan!20Physique et liaison de données	cyan!20Assure la transmission des bits et la détection d'erreurs
cyan!20Réseau	cyan!20Gère le routage et l'adressage IP
cyan!20Transport	cyan!20Fournit des services de communication fiables (TCP) ou rapides (UDP)
cyan!20Application et services	cyan!20Interface avec les applications utilisateur

TABLE 2.8 – *Rôle des couches réseau dans la communication*

2.2.3 Protocoles majeurs dans les réseaux IP (ICMP, ARP, TCP, UDP)

Voila la table qui presente les protocoles majeurs dans les réseaux IP (ICMP, ARP, TCP, UDP)

2whitegray!15

cyan!20**Protocole**	cyan!20**Fonction principale**
cyan!20ICMP	cyan!20Gère les messages d'erreur et les diagnostics réseau
cyan!20ARP	cyan!20Convertit les adresses IP en adresses MAC
cyan!20TCP	cyan!20Assure la fiabilité et le contrôle de flux des transmissions
cyan!20UDP	cyan!20Permet une communication rapide sans connexion

TABLE 2.9 – *Protocoles majeurs dans les réseaux IP*

2.2.4 Protocole UDP : Fonctionnement et usages

Différences entre UDP et TCP

UDP est un protocole non connecté et rapide tandis que TCP garantit une transmission fiable avec contrôle de flux [23].

Utilisation du protocole UDP pour l'allocation IP (DHCP)

Le protocole DHCP utilise UDP pour l'attribution dynamique d'adresses IP aux hôtes d'un réseau [24].

Applications courantes du protocole UDP

UDP est utilisé dans les jeux en ligne, la VoIP et le streaming multimédia où la rapidité prime sur la fiabilité [25].

2.3 Le Protocole DHCP : Concepts et Fonctionnalités

2.3.1 Présentation du protocole DHCP et son importance

Définition et rôle du DHCP dans les réseaux

Le Dynamic Host Configuration Protocol (DHCP) est un protocole réseau permettant l'attribution automatique des adresses IP aux hôtes d'un réseau. Il simplifie la gestion des adresses IP et réduit les conflits liés aux configurations manuelles [26].

Avantages de l'attribution dynamique des adresses IP

L'attribution dynamique des adresses IP via DHCP permet une gestion efficace des ressources réseau, notamment en évitant les erreurs de configuration manuelle et en facilitant l'administration des réseaux de grande envergure [27].

Cas d'utilisation du DHCP dans les réseaux domestiques et d'entreprise

Le DHCP est utilisé aussi bien dans les réseaux domestiques que dans les environnements professionnels où de nombreux appareils nécessitent une adresse IP sans intervention manuelle [28].

2.3.2 Fonctionnement détaillé du protocole DHCP

Le protocole DHCP fonctionne sur un modèle client-serveur et repose sur une séquence d'échanges standardisée entre les appareils d'un réseau. Son rôle est d'attribuer automatiquement des adresses IP aux clients qui en font la demande, tout en fournissant des informations complémentaires comme la passerelle par défaut, les serveurs DNS et d'autres options de configuration réseau.

Processus DORA :

L'attribution d'une adresse IP via DHCP suit le processus DORA (Discovery, Offer, Request, Acknowledge), qui implique une interaction entre le client et le serveur DHCP.

>m3cm | X |
cyan!30 **Étape** **Description**
Discovery Un client sans adresse IP envoie un message DHCPDISCOVER en broadcast pour rechercher un serveur DHCP disponible sur le réseau **?**.
Offer Un ou plusieurs serveurs DHCP répondent avec un message DHCPOFFER, proposant une adresse IP ainsi que des paramètres réseau **?**.
Request Le client sélectionne une offre et envoie un message DHCPREQUEST au serveur qui a fait la proposition, demandant à utiliser l'adresse fournie **?**.
Acknowledge Le serveur confirme l'attribution de l'adresse avec un message DHCPACK contenant les informations de configuration finale **?**.

TABLE 2.10 – *Processus DORA (DHCP)*

Si le client ne reçoit pas de réponse du serveur DHCP, il peut répéter le processus Discovery après un délai défini. Une fois l'adresse attribuée, celle-ci est utilisée pour la durée du bail définie.

Durée du bail et renouvellement des adresses

Chaque adresse IP attribuée par un serveur DHCP a une durée de bail (lease time), au terme de laquelle le client doit demander son renouvellement. Ce processus se déroule en plusieurs étapes :

p4cm | X |
cyan!30 **Étape du renouvellement** **Description**
T1 (50 % du bail écoulé) Le client envoie une requête de renouvellement (DHCPREQUEST) au serveur initial.
T2 (87,5 % du bail écoulé) Si aucune réponse n'est reçue, le client envoie une nouvelle demande en broadcast.
Expiration du bail Si aucune réponse n'est reçue après expiration du bail, le client doit recommencer le processus DORA pour obtenir une nouvelle adresse IP.

TABLE 2.11 – *Durée du bail et renouvellement des adresses*

Options DHCP et configuration avancée (PXE, BOOTP, DNS)

Le protocole DHCP prend en charge différentes options avancées permettant d'étendre ses fonctionnalités :

p4cm	X
cyan !30 **Option**	**Description**
BOOTP)	Un protocole précurseur de DHCP, utilisé pour le démarrage sans disque des machines.
PXE (Preboot Execution Environment)	Permet à une machine de télécharger un système d'exploitation depuis un serveur réseau lors du démarrage.
-Options DNS et passerelle	Le DHCP peut configurer automatiquement les adresses des serveurs DNS et la passerelle par défaut pour chaque client.
VLAN et segmentation réseau	Certaines implémentations DHCP permettent d'attribuer dynamiquement des VLANs en fonction des clients connectés.

TABLE 2.12 – *Options DHCP et configuration avancée (PXE, BOOTP, DNS)*

Le relai DHCP et son utilité

Dans les réseaux de grande envergure, il n'est pas toujours possible d'avoir un serveur DHCP sur chaque sous-réseau. Le relai DHCP (ou DHCP Relay Agent) est une solution permettant aux requêtes DHCP de traverser des sous-réseaux distincts :

— Lorsqu'un client envoie une requête DHCPDISCOVER en broadcast, le relai DHCP capture le message et le transmet en unicast au serveur DHCP.
— Le serveur DHCP répond avec une adresse IP et les options réseau, que le relai DHCP retransmet ensuite au client.
— Cette approche optimise la gestion des adresses IP et réduit le besoin de multiplier les serveurs DHCP dans une infrastructure complexe [35].

2.3.3 DHCPv4 vs. DHCPv6 : Évolutions et différences

Comparaison des mécanismes d'attribution d'adresses

Le passage de DHCPv4 à DHCPv6 a introduit plusieurs évolutions significatives, notamment en matière de gestion des adresses, de sécurité et de scalabilité. Le tableau suivant compare ces deux versions du protocole :

X	X	X
cyan !30 **Critère**	**DHCPv4**	**DHCPv6**
Format d'adresse	IPv4 (32 bits)	IPv6 (128 bits)
Mode d'attribution	Centralisé	Centralisé et Stateless
Sécurité	Moins sécurisé	Amélioré avec IPsec
Support des options	Limité	Étendu (supporte plus d'options de configuration)
Gestion des adresses	Uniquement par le serveur DHCP	Possibilité de combiner SLAAC et DHCPv6
Interopérabilité	Ne prend en charge que IPv4	Compatible avec les réseaux IPv4 et IPv6
Scalabilité	Moins efficace pour les grands réseaux	Conçu pour supporter des infrastructures étendues

TABLE 2.13 – *Comparaison entre DHCPv4 et DHCPv6*

SLAAC et DHCPv6 : Approches complémentaires

Le Stateless Address Autoconfiguration (SLAAC) est une alternative à DHCPv6 qui permet aux hôtes de générer leurs propres adresses sans interaction avec un serveur DHCP. SLAAC utilise les messages ICMPv6 Router Advertisement (RA) envoyés par les routeurs pour informer les clients du préfixe réseau.

Contrairement à DHCPv6, SLAAC ne nécessite pas de serveur centralisé, ce qui améliore la résilience du réseau [36].

Support de la configuration sans état (Stateless) en IPv6

Avec DHCPv6 Stateless, un client peut obtenir des informations de configuration réseau comme les adresses DNS sans recevoir d'adresse IPv6 complète. Cette approche réduit la charge sur les serveurs DHCP tout en permettant une gestion centralisée de certains paramètres réseau [37].

Améliorations en matière de sécurité et de scalabilité

L'évolution vers DHCPv6 a permis d'améliorer la sécurité et la scalabilité du protocole. Plusieurs mécanismes ont été intégrés pour renforcer la protection contre les attaques et garantir un fonctionnement optimal des réseaux de grande envergure.

p4cm | X |
cyan!30 **Amélioration** **Description**
Authentification DHCP Permet d'éviter l'utilisation de serveurs DHCP malveillants.
Compatibilité avec IPsec Sécurise les échanges entre clients et serveurs DHCP.
Meilleure gestion IPv6 Facilite la scalabilité des infrastructures réseau [38].

TABLE 2.14 – *Améliorations en matière de sécurité et de scalabilité*

2.3.4 Sécurité et vulnérabilités du DHCP

Attaques DHCP courantes

Les attaques DHCP exploitent les vulnérabilités du protocole pour compromettre la sécurité du réseau. Le tableau suivant présente quelques attaques courantes :

p4cm | X |
cyan!30 **Type d'attaque** **Description**
Rogue DHCP Un attaquant configure un faux serveur DHCP pour piéger les utilisateurs [39].
Starvation Attack L'attaquant épuise les adresses disponibles en envoyant de nombreuses requêtes DHCP [40].
Attaque MITM via DHCP Spoofing Un attaquant intercepte et manipule les requêtes DHCP pour rediriger le trafic [41].

TABLE 2.15 – *Améliorations en matière de sécurité et de scalabilité*

Solutions de sécurisation du DHCP

Des solutions comme le DHCP Snooping et l'authentification 802.1X permettent de protéger les réseaux contre les attaques DHCP [42].

p4cm	X
cyan!30 **Solution**	**Description**
DHCP Snooping	Filtre les messages DHCP non autorisés pour empêcher les attaques de type rogue DHCP.
802.1X	Assure l'authentification des clients avant de leur attribuer une adresse IP, réduisant les risques d'accès non autorisé.

TABLE 2.16 – *Solutions de sécurisation du DHCP Snooping et 802.1X)*

2.3.5 Limitations du DHCP et perspectives d'amélioration

Dépendance à un serveur centralisé et risques de panne

Le DHCP repose sur un serveur centralisé, ce qui peut poser problème en cas de panne du serveur. Une défaillance du serveur DHCP entraîne une incapacité des clients à obtenir une adresse IP, rendant ainsi le réseau inutilisable. Pour pallier ce problème, il est recommandé d'implémenter une architecture redondante avec plusieurs serveurs DHCP fonctionnant en haute disponibilité et utilisant des mécanismes de basculement automatique [43].

Problèmes de latence et de gestion dans les grands réseaux

Dans les réseaux de grande envergure, le DHCP peut introduire une latence notable en raison du volume élevé de requêtes. Une mauvaise gestion des baux et des ressources peut entraîner une congestion du réseau. Pour améliorer la performance, des solutions telles que le relai DHCP, l'optimisation des temps de bail et l'utilisation d'algorithmes d'équilibrage de charge peuvent être mises en place. De plus, la segmentation du réseau avec des VLANs permet de mieux distribuer les requêtes DHCP et d'assurer une gestion efficace des adresses IP [44].

Alternatives et évolutions (IPv6, SDN, IA pour la gestion des adresses)

L'émergence de nouvelles technologies améliore l'efficacité de l'attribution des adresses IP :

p4cm	X
cyan!30 **Alternative**	**Avantages**
IPv6	Permet une auto-configuration sans besoin d'un serveur centralisé.
SDN (Software-Defined Networking)	Apporte une gestion dynamique et centralisée des ressources réseau.
IA	Optimise la répartition des adresses IP en fonction de l'utilisation réelle du réseau [45].

TABLE 2.17 – *Solutions de sécurisation du DHCP Snooping et 802.1X)*

2.4 Le Langage Rust et son Utilisation dans le Développement Réseau

2.4.1 Introduction à Rust : Principes et philosophie

Origines et objectifs du langage Rust

Rust est un langage de programmation développé par Mozilla Research en 2010. Son objectif principal est d'offrir une alternative moderne aux langages traditionnels comme C et C++, en mettant l'accent sur la sécurité, la performance et la gestion efficace de la mémoire [46].

Principes fondamentaux : sécurité, performance, concurrence

p4cm | X |
cyan !30 **Principe** **Description**
Sécurité Grâce à son système de gestion de la mémoire sans garbage collector, Rust élimine les erreurs courantes comme les dépassements de tampon et les accès mémoire après libération [47].
Performance Comparable à celle du C et du C++, Rust est optimisé pour offrir une exécution rapide et efficace [48].
Concurrence Il facilite l'écriture de code concurrent sûr en évitant les conditions de course grâce à son système de propriété et d'emprunt [49].

Table 2.18 – *Principes fondamentaux*

Adoption et communauté Rust

Depuis sa création, Rust a connu une adoption croissante dans divers domaines, notamment le développement réseau, la cybersécurité et les systèmes embarqués. Il bénéficie d'une communauté active qui contribue à son écosystème via des bibliothèques et des outils open-source [50].

2.4.2 Sécurité mémoire et gestion des ressources

Système de gestion de la mémoire sans garbage collector

Rust adopte un modèle unique de gestion de la mémoire basé sur un système de propriété (ownership). Contrairement aux langages utilisant un garbage collector, Rust libère la mémoire de manière déterministe lorsqu'une variable sort de sa portée, réduisant ainsi les risques de fuites mémoire et améliorant les performances [51].

Concept de propriété, emprunt et durée de vie

Le système de propriété repose sur trois règles fondamentales :

p4cm | X |
cyan !30 **Principe** **Description**
Propriété Chaque valeur possède un unique propriétaire.
Emprunt Une valeur peut être empruntée temporairement sans en modifier la propriété.
Durée de vie La durée de vie d'une valeur est contrôlée par des annotations explicites évitant les erreurs d'accès mémoire [52].

TABLE 2.19 – *Concept de Propriété, Emprunt et Durée de vie*

Prévention des vulnérabilités mémoire (buffer overflow, use-after-free)

Grâce à son strict contrôle de la mémoire, Rust prévient efficacement les erreurs courantes :

p4cm | X |
cyan !30 **Vulnérabilité** **Prévention en Rust**
Buffer overflow Le compilateur Rust empêche tout accès hors des limites d'un tableau [53].
Use-after-free Le système de propriété garantit qu'aucun pointeur invalide ne subsiste après la libération de la mémoire [54].

TABLE 2.20 – *Concept de Propriété, Emprunt et Durée de vie*

Rust est ainsi un choix idéal pour la mise en œuvre sécurisée de protocoles réseau comme DHCP grâce à sa gestion stricte de la mémoire et sa compatibilité avec la programmation asynchrone [55].

2.4.3 Rust et la programmation réseau

Modèle de programmation asynchrone en Rust

Rust propose un modèle de programmation asynchrone performant basé sur le mot-clé **async** et **await**, permettant l'exécution efficace de tâches concurrentes sans blocage du thread principal [56].

Bibliothèques et outils pour le réseau

Rust dispose de plusieurs bibliothèques permettant d'implémenter des applications réseau robustes et performantes :

p4cm | X |
cyan !30 **Bibliothèque** **Description**
Tokio Une bibliothèque asynchrone de haute performance offrant des primitives pour la gestion des connexions réseau, des sockets et des threads [57].
async-std Une alternative à Tokio qui simplifie la programmation asynchrone en offrant une API similaire à la bibliothèque standard de Rust [58].
smoltcp Une pile réseau minimaliste conçue pour les systèmes embarqués et les environnements contraints en ressources [59].

TABLE 2.21 – *Bibliothèques et outils pour le réseau*

Ces bibliothèques permettent de développer des applications réseau performantes tout en garantissant la sécurit

2.4.4 Comparaison avec d'autres langages dans le développement réseau

Rust s'impose progressivement comme une alternative fiable aux langages traditionnels utilisés dans le développement réseau. Grâce à son système de gestion de la mémoire sécurisé et son modèle de concurrence efficace, il permet d'éviter les erreurs courantes rencontrées en C, C++ et Go. Cependant, chaque langage possède ses propres avantages et inconvénients, influençant leur adoption selon les exigences spécifiques des projets. La comparaison suivante met en évidence les principales différences entre Rust et ces langages dans un contexte de développement réseau.

cyan!30 **Critère**	**Rust vs. C**	**Rust vs. C++**	**Rust vs. Go**
Sécurité mémoire	Rust prévient les erreurs de segmentation grâce à son système de propriété [60].	C++ propose des pointeurs intelligents, mais ne garantit pas une sécurité absolue [61].	Go inclut un ramasse-miettes qui empêche certaines erreurs mémoire, mais avec un coût en performance [62].
Gestion des erreurs	Gestion stricte via **Result** et **Option** [63].	Exceptions et code non sécurisé peuvent poser des problèmes [64].	Gestion simplifiée via **error**, mais moins contrôlée que Rust [65].
Performance	Comparable à C, mais sans compromis sur la sécurité [66].	Performant, mais avec des optimisations complexes [67].	Similaire, mais le garbage collector peut ralentir l'exécution [68].
Concurrence	Rust utilise **async/await** et **Tokio** pour une exécution efficace [69].	Concurrence basée sur les threads et **std : :thread** [70].	Modèle de goroutines simple et efficace [71].
Utilisation en réseau	Très adapté avec **Tokio**, **async-std**, **smoltcp** [72].	Moins adapté aux applications modernes sans bibliothèques additionnelles [73].	Excellent support grâce à sa bibliothèque standard et ses services cloud [74].

TABLE 2.22 – *Comparaison de Rust avec d'autres langages dans le développement réseau*

Rust se distingue par sa sécurité mémoire, sa gestion rigoureuse des erreurs et son absence de garbage collector, ce qui le rend idéal pour le développement réseau exigeant en performance et en fiabilité.

2.5 Autres Protocoles Complémentaires et Technologies Associées

2.5.1 DNS et son rôle dans la gestion des noms de domaine

Le Domain Name System (DNS) est un composant fondamental d'Internet, permettant de convertir des noms de domaine lisibles par les humains en adresses IP exploitables par les machines. Il repose sur une structure hiérarchique et divers mécanismes de sécurité pour garantir son bon fonctionnement et empêcher les attaques telles que l'usurpation d'identité (spoofing).

Le DNS joue un rôle clé dans la gestion des noms de domaine et l'accès aux services en ligne. Il garantit la fiabilité et la rapidité de la navigation sur Internet tout en nécessitant des mesures de sécurité strictes pour éviter les attaques potentielles.

p4cm	X
cyan!30 **Aspect**	**Description**
Résolution de noms	ULe DNS traduit les noms de domaine en adresses IP à l'aide de requêtes envoyées aux serveurs DNS [75].
Hiérarchie du DNS	Il est structuré en plusieurs niveaux : root, TLD (Top-Level Domain), et serveurs récursifs qui assurent la résolution [76].
Sécurité DNS	Les mécanismes comme DNSSEC protègent contre la falsification des réponses DNS et le cache poisoning [77].

TABLE 2.23 – *DNS et son rôle dans la gestion des noms de domaine*

2.5.2 Protocole IPv6 : Défis et impact sur l'allocation dynamique d'adresses

IPv6 est la nouvelle génération du protocole Internet, conçue pour remplacer IPv4 en raison de l'épuisement des adresses. Son adoption présente des défis techniques et stratégiques [78].

p4cm	X
cyan!30 **Catégorie**	**Détails**
Besoin et adoption de l'IPv6	IPv6 est nécessaire pour répondre à la pénurie d'adresses IPv4 et permettre une meilleure connectivité des objets connectés (IoT). Son adoption reste lente en raison des coûts de transition et du manque de compatibilité avec certains équipements anciens [79].
Différences fondamentales entre IPv4 et IPv6	IPv6 utilise des adresses sur 128 bits contre 32 bits pour IPv4, élimine la nécessité du NAT, et offre une meilleure gestion de la qualité de service (QoS) et de la sécurité [80].
Allocation dynamique des adresses IPv6	IPv6 prend en charge deux méthodes principales : SLAAC (Stateless Address Autoconfiguration) qui permet aux hôtes de s'auto-configurer, et DHCPv6 qui offre un contrôle plus centralisé par un serveur [81].
Défis de migration et compatibilité avec IPv4	La cohabitation entre IPv4 et IPv6 nécessite des mécanismes comme le tunneling, la double pile (dual-stack) et la traduction d'adresses (NAT64). Ces solutions ajoutent de la complexité et augmentent la latence réseau [82].

TABLE 2.24 – *Tableau comparatif sur IPv6 et ses défis*

L'adoption d'IPv6 constitue une évolution incontournable pour assurer la croissance continue d'Internet et répondre aux besoins émergents en matière de connectivité. Toutefois, les défis liés à sa mise en œuvre, notamment la transition depuis IPv4, la nécessité de nouveaux équipements compatibles et la complexité des mécanismes de cohabitation, freinent encore son déploiement massif. Une planification rigoureuse et l'adoption progressive de solutions hybrides comme le dual-stack sont essentielles pour une transition fluide et efficace vers IPv6.

2.5.3 Sécurité des infrastructures réseau (Cryptographie et authentification)

La sécurité des infrastructures réseau repose sur l'utilisation de techniques avancées de cryptographie et de protocoles d'authentification robustes. Ces solutions garantissent la confidentialité, l'intégrité et l'authenticité des communications sur Internet [83].

L'augmentation des cyberattaques et des violations de données impose aux entreprises d'adopter des mesures de sécurité renforcées. Une combinaison efficace de cryptographie avancée et de protocoles d'authentification robustes est

p4cm	X
cyan!30 **Catégorie**	**Détails**
Principes de cryptographie appliqués aux réseaux	L'usage de la cryptographie permet de sécuriser les échanges en garantissant la confidentialité (chiffrement), l'intégrité (hachage) et l'authenticité (signatures numériques) [84].
Protocoles de sécurité réseau	Des protocoles comme SSL/TLS assurent la protection des communications sur Internet, tandis qu'IPSec permet le chiffrement des paquets IP pour des connexions sécurisées [85].
Authentification et contrôle d'accès	Les protocoles comme RADIUS et 802.1X permettent une gestion centralisée de l'authentification et du contrôle d'accès aux réseaux d'entreprise et aux infrastructures critiques [86].

TABLE 2.25 – *Tableau comparatif sur la sécurité des infrastructures réseau*

essentielle pour assurer la résilience des infrastructures réseau face aux menaces émergentes.

2.5.4 Cloud et virtualisation : Impact sur la gestion des adresses IP

Avec l'essor du cloud computing et des technologies de virtualisation, la gestion des adresses IP évolue pour s'adapter aux environnements dynamiques et évolutifs des datacenters modernes [87].

p4cm	X
cyan!30 **Catégorie**	**Détails**
Virtualisation des réseaux	Des technologies comme SDN (Software-Defined Networking) et NFV (Network Function Virtualization) permettent une gestion plus flexible et évolutive des infrastructures réseau [88].
Gestion des IP dans les environnements cloud	Des plateformes comme AWS, Azure et OpenStack intègrent des solutions de gestion dynamique des adresses IP, facilitant le provisionnement des services [89].
Sécurité et segmentation des réseaux virtuels	Les réseaux virtuels requièrent des mesures avancées de segmentation et d'isolation pour prévenir les attaques et garantir la sécurité des données [90].
Impact sur la scalabilité et la gestion des ressources	L'automatisation et l'orchestration des adresses IP améliorent la scalabilité des infrastructures cloud et optimisent l'allocation des ressources réseau [91].

TABLE 2.26 – *Tableau comparatif sur l'impact du Cloud et de la virtualisation*

L'intégration du cloud et de la virtualisation transforme profondément la gestion des réseaux et des ressources IP. Ces technologies offrent des avantages considérables en termes de flexibilité et d'évolutivité, mais elles nécessitent également des solutions de sécurité adaptées pour prévenir les vulnérabilités spécifiques aux environnements virtualisés.

2.6 Conclusion du Chapitre

Ce chapitre a abordé les enjeux de l'allocation des adresses IP face aux évolutions technologiques. La transition vers IPv6, bien que nécessaire, reste un défi technique et économique, nécessitant des mécanismes hybrides pour une adoption progressive. Par ailleurs, la cybersécurité demeure une priorité avec l'intégration de solutions cryptographiques et de protocoles d'authentification

robustes. Enfin, l'essor du cloud et de la virtualisation redéfinit la gestion des adresses IP, apportant flexibilité et scalabilité, tout en exigeant des stratégies de sécurité adaptées. Le chapitre suivant explorera les outils modernes de gestion réseau pour améliorer performance et sécurité.

Bibliographie

Notations

PMDM	Processus de Markov déterministe par morceaux
$p.s.$	presque sûrement
$\mathbb{N}, \mathbb{N}^*$	ensemble des entiers naturels, des entiers strictement positifs
$\mathbb{R}, \mathbb{R}_+$	ensembles des réels et des réels positifs
$\mathbb{R}^d$	ensemble des vecteurs réels à d dimensions
$\mathbb{P}, \mathbb{E}$	probabilité et espérance
$(\Omega, \mathcal{F}, ¶)$	espace probabilisé
B_t	mouvement brownien
$\|E\|$	cardinal de l'ensemble E

الملخص

إلملخص الملخص

Résumé

Résumé Résumé

Abstract

Abstract Abstract