République Algérienne Démocratique et Populaire
Ministère de l'Enseignement Supérieur et de la Recherche Scientifique
Université d'Alger 1

Faculté des sciences - Département d'Informatique

Mémoire de Licence

Fillière : Informatique - Spécialité : SI

La Détection D'intrusion Intelligent Dans Le Réseau SDN En Se Basant Sur Le Deep Learning (GAN Autoencoder)

Réalisé par :
Nouasri Lina Manel
Larkem Kahina Bahia

Encadré par :
Mme. BOUCHAIR Maria (Ericsson)
M. ABBAS Amine

Soutenu devant le jury composé de :

DÉDICACE

À celle qui m'a donné la vie, maman ta présence, ton soutien et ton amour inconditionnel m'ont permis d'avancer et de devenir qui je suis aujourd'hui. Je t'aime infiniment.

À mon frère, mon allié de toujours. Depuis notre plus tendre enfance, nous avons tout partagé : nos jeux, nos secrets, nos joies et nos peines. Tu as été mon protecteur, mon confident et l'un de mes plus grands soutiens. Merci d'avoir été là à chaque étape de ma vie, dans les meilleurs comme dans les pires moments. Notre complicité fraternelle restera à jamais l'un de mes plus précieux trésors.

À mes grands parents, votre amour et vos encouragements m'ont porté tout au long de mon parcours. Je vous dois tant et je vous aime profondément.

À mon père, qui m'a permis de forger la personalité que j'ai aujourd'hui.

À mon groupe d'amis qui ont égayé mes journées à la fac, je vous souhaite beaucoup de succès à l'avenir.

À Lina, ma précieuse binôme. Nous avons réussi à traverser cette aventure ensemble, partageant les défis et les victoires. Ton soutien, ta persévérance et ton humour ont rendu ce parcours tellement plus doux.

À tous ceux qui me sont chers, à vous tous.

- Kahina

À ma maman, mon papa et mes sœurs, pour leur amour inconditionnel, leur soutien indéfectible et leurs encouragements constants. Vous êtes mes piliers, ma force et ma joie..

À mes amis, pour leur soutien sans faille, leur croyance en mes capacités et leurs conseils précieux. Votre amitié est un trésor que je chérirai toujours.

Au chef cuisinier d'Ericsson, pour avoir toujours motivé Kahina à se déplacer.

À mon perfectionnisme, pour m'avoir poussé à donner le meilleur de moi-même et à viser l'excellence dans ce projet. Et mes rêves, pour m'avoir inspiré, motivé et guidé tout au long de ce parcours académique. Je continuerai à poursuivre mes rêves avec détermination et passion.

- Lina

REMERCIEMENTS

Le travail présenté, dans le cadre de ce Projet de Fin d'Etudes, a été développé et mis en œuvre à Ericsson-Algérie.
Je tiens à exprimer ma profonde gratitude à mes chers parents, votre présence et vos sacrifices m'ont permis d'atteindre mes objectifs.
J'adresse des remerciements particuliers à mon frère Rayane, qui a toujours cru en moi et m'a soutenu dans les moments les plus difficiles. Ta confiance et tes précieux conseils ont été d'un grand réconfort.
A ma mère, la femme la plus forte que je connaisse, merci pour ton dévouement sans faille et tes prières qui m'ont guidé sur le chemin de la réussite. Ta bienveillance et ta sagesse ont été une source intarissable d'inspiration.
Je tiens également à exprimer ma sincère reconnaissance à Madame BOUCHAIR Maria, mon encadrante, pour ses judicieux conseils, sa disponibilité et son suivi rigoureux tout au long de ce projet. Ses connaissances approfondies et son expertise ont grandement contribué à la réalisation de ce travail.
Mes remerciements vont également à Monsieur ABBAS, mon encadrant, pour avoir accepté de diriger ce projet et pour ses orientations avisées qui ont permis d'en assurer le bon déroulement.
Enfin, je remercie toutes les personnes qui, de près ou de loin, ont contribué à la réalisation de ce projet.

- Kahina

Je tiens tout d'abord à exprimer ma profonde gratitude et amour envers mes parents, Zoubir et Hanifa, pour leur soutien indéfectible, leur amour et leurs encouragements constants tout au long de mes études. Leur soutien inconditionnel a été ma source de motivation et de force. J'espère ne jamais vous décevoir et pouvoir toujours vous rendre fiers de moi tout au long de mon parcours de vie. Et de toujours avoir votre soutien.
Je souhaite également remercier mes sœurs, Marwa et Maissa, pour leur soutien moral et leurs précieux conseils. Leur présence et soutien ont été un réconfort dans les moments difficiles et une source de joie dans les moments de réussite. J'espère toujours être à la hauteur comme votre grande sœur et ne jamais vous décevoir.
Sans oublier mes amis, spécialement Safa et Nabila, pour leur soutien indéfectible et leur croyance en moi, en mes capacités et en mes rêves. Leur présence dans ma vie a été un véritable réconfort et une source de joie inestimable. Leur amitié précieuse est un cadeau que je chérirai toujours. Merci Safa et Nabila pour votre soutien inconditionnel et votre amitié sincère.
Je tiens à exprimer ma reconnaissance envers mes encadrants, Mme Maria Bouchair de l'entreprise Ericsson et M. Amine Abbas, mon professeur. Leurs conseils éclairés, leur expertise et leur disponibilité ont été d'une aide précieuse pour la réalisation de ce mémoire. Leurs encouragements et leur soutien ont été essentiels dans la réussite de ce travail. Merci pour votre aide et d'avoir cru en nous.
Mes remerciement envers ma binôme et mon amie, Kahina, pour sa collaboration et son travail d'équipe tout au long de ce projet. Sa contribution a été précieuse et a grandement contribué à la réussite de ce mémoire. Sa disponibilité, ses idées et son soutien ont été des atouts majeurs dans notre travail en commun. Merci d'avoir été une partenaire aussi engagée et fiable.
Enfin, je tiens à remercier moi-même pour ma persévérance, ma détermination et mon engagement tout au long de ce projet. Ce mémoire représente pour moi une étape importante dans mon parcours académique, et je suis fière d'avoir relevé ce défi avec succès. J'espère être à la hauteur de tout ce qui m'attend dans le futur.

- Lina

RÉSUMÉ

Avec le développement constant des technologies, de nouvelles cyberattaques complexes sont déployées pour exploiter les vulnérabilités des systèmes et mener d'autres activités malveillantes. Les infrastructures réseau sont parmi les systèmes les plus visés, subissant une variété d'attaques telles que le déni de service (DoS), les attaques par déni de service distribué (DDoS), les attaques CP SYN Flood, les attaques Ping of Death, les attaques Teardrop, les attaques par balayage, et bien d'autres. Pour contrer ces attaques et assurer la sécurité du réseau tout en maintenant une haute disponibilité pour les utilisateurs légitimes, de nombreux efforts sont déployés pour développer et mettre en œuvre différentes méthodes et techniques.
Une méthode largement reconnue pour sécuriser les réseaux est la mise en place de systèmes de détection d'intrusion (IDS). L'objectif principal d'un IDS est de surveiller le trafic réseau à la recherche d'activités suspectes ou malveillantes. Plus spécifiquement, un système de détection d'intrusion réseau (NIDS) inspecte le trafic réseau sortant de tous les hôtes en temps réel. Basé sur des critères spécifiques, il peut détecter et identifier les attaques, puis prendre les mesures de sécurité appropriées pour les arrêter ou les bloquer, réduisant ainsi le risque de dommages au réseau.
Cependant, avec la complexité croissante des attaques de cybersécurité, les méthodes actuellement utilisées dans les NIDS ne sont souvent pas suffisantes pour contrer ces attaques. C'est dans ce contexte qu'Ericsson envisage d'adopter une nouvelle solution intelligente basée sur le deep learning (GAN Autoencoder) pour détecter les intrusions avancées. Cette solution reposera sur l'analyse des données, notamment des fichiers de capture de trafic réseau, afin de renforcer la sécurité du réseau.
Les méthodes et outils utilisés pour ce projet incluront les étapes classiques d'un projet d'ingénierie, allant de la spécification des besoins au prototype, aux tests et à la validation. En conclusion, ce projet vise à fournir à Ericsson une solution avancée pour détecter et contrer les attaques avancées sur son réseau, en utilisant des techniques de deep learning pour renforcer la sécurité et la fiabilité de son infrastructure.

PRÉSENTATION DE L'ENTREPRISE

Ericsson, une société suédoise de télécommunications et de réseaux fondée en 1876, est un acteur mondial de premier plan dans le domaine des technologies de communication. La maîtrise de l'intelligence artificielle (IA) est l'un de ses atouts majeurs, lui permettant de résoudre des défis complexes pour les fournisseurs de services. De plus, Ericsson met un accent particulier sur la sécurité des réseaux, garantissant des solutions robustes pour protéger les données et les communications de ses clients. Avec un portefeuille de plus de 54 000 brevets, Ericsson se distingue par son leadership en propriété intellectuelle. Présente en Algérie depuis 1973, l'entreprise y offre son expertise et ses solutions technologiques, contribuant au développement des infrastructures de communication et à la transformation numérique du pays.

TABLE DES MATIÈRES

LISTE DES ACRONYMES

AE Autoencoder

AI / IA Artificial Intelligence

ANN Artificial Neural Network

API Application Programming Interface

CSV Comma-Separated Values

DoS Denial of Service

DDoS Distributed Denial of Service

DL Deep Learning

FN False Negative

FTP File Transfer Protocol

FP False Positive

HIDS Host-based Intrusion Detection System

HTTP Hypertext Transfer Protocol

ICMP Internet Control Message Protocol

IDS Intrusion Detection System

IP Internet Protocol

IPERF Internet Performance Working Group

LR Logistic Regression

ML Machine Learning

MLP Multi-Layer Perceptron

MSE Mean Squared Error

NB Naive Bayes

NIDS Network Intrusion Detection System

ReLU Rectified Linear Unit

RF Random Forest

RMSE Root Mean Squared Error

SDN Software Defined Networking

SMOTE Synthetic Minority Oversampling Technique

TDC Decision Tree Classifier

TCP Transmission Control Protocol

TN True Negative

TP True Positive

UDP User Datagram Protocol

VMware Virtual Machine software platform

TABLE DES FIGURES

LISTE DES TABLEAUX

INTRODUCTION GÉNÉRALE

Contexte général :

Les réseaux informatiques occupent aujourd'hui une place prépondérante dans les infrastructures critiques de toute organisation. Qu'il s'agisse d'entreprises, d'institutions publiques ou d'organismes gouvernementaux, la sécurisation de ces réseaux est devenue un enjeu majeur face à la multiplication des cybermenaces.

Problématique :

Les systèmes de détection d'intrusions (IDS) conventionnels, bien qu'utiles, présentent des limites notables. Leur capacité à détecter de nouvelles formes d'attaques sophistiquées reste limitée, ce qui expose les réseaux à des risques importants. De plus, ces systèmes génèrent souvent un nombre élevé de faux positifs, entravant leur efficacité opérationnelle.

Contribution :

C'est dans ce contexte que les avancées récentes dans le domaine de l'Intelligence Artificielle (IA), et plus particulièrement du Deep Learning, offrent des perspectives prometteuses pour relever ces défis. En exploitant les capacités d'apprentissage automatique des réseaux de neurones profonds, il est possible de développer des modèles de détection d'intrusions plus performants, capables de s'adapter aux nouvelles formes d'attaques et de réduire considérablement les faux positifs. Le présent mémoire s'inscrit dans cette dynamique, en proposant une approche novatrice pour la détection d'intrusions dans les réseaux SDN (Software-Defined Networking) basée sur le Deep Learning. Il explore les fondements théoriques de cette technologie, décrit la méthodologie employée et présente les résultats obtenus à l'issue des expérimentations menées. Dans le cadre de notre projet, nous avons acquis en deux mois les connaissances nécessaires sur les concepts d'intelligence artificielle et de réseaux SDN, atteignant un niveau de maîtrise adéquat, tout en nous efforçant d'aborder ces sujets avec rigueur et professionnalisme.

CHAPITRE 1

RÉSEAUX SDN, SÉCURITÉ INFORMATIQUE ET SYSTÈME DE DÉTECTION D'INTRUSIONS

1.1 Les Réseaux SDN

Le SDN (Software-Defined Networking) révolutionne la gestion des réseaux en séparant les plans de contrôle et de données, offrant ainsi une flexibilité et une gestion dynamique des ressources réseau. Ses composants clés incluent les applications SDN, le contrôleur SDN tel que Ryu, les APIs orientées vers le bas et vers le haut, les commutateurs SDN, la séparation du plan de contrôle et du plan de données, ainsi que le protocole OpenFlow.

1.1.1 Architecture Des Réseaux SDN

L'architecture du Réseau Défini par Logiciel (SDN) comprend plusieurs éléments clés comment le montre la figure 1.1 :
1. Couche Application : Les applications SDN sont des programmes logiciels exploitant les interfaces programmables du contrôleur SDN pour mettre en œuvre des services réseau tels que l'ingénierie du trafic, l'équilibrage de charge et la sécurité.
2. Contrôleur SDN : Le contrôleur, élément central de l'architecture SDN, prend des décisions globales sur le comportement du réseau en fonction des informations des commutateurs. Le contrôleur **Ryu**, incarne parfaitement le rôle pivot dans la gestion et le contrôle du réseau. En s'appuyant sur le protocole OpenFlow, Ryu établit une communication bidirectionnelle entre le contrôleur et les dispositifs réseau, ce qui lui permet de prendre des décisions globales sur le comportement du réseau en fonction des informations collectées des commutateurs. Cette interaction fluide offre une gestion centralisée du flux de données à travers le réseau. Grâce à cette approche, le contrôleur peut programmer et adapter dynamiquement le cheminement des paquets en fonction des besoins spécifiques du réseau et des applications en cours d'exécution. Cette flexibilité

accrue permet une optimisation en temps réel des performances du réseau tout en répondant aux exigences changeantes de l'environnement.

3. APIs Orientées Vers Le Bas (Southbound APIs) : Ces APIs permettent la communication entre le contrôleur SDN et les dispositifs réseau (commutateurs, routeurs). OpenFlow est une API bien connue utilisée à cet effet.

4. APIs Orientées Vers Le Haut (Northbound APIs) : Ces APIs facilitent la communication entre le contrôleur SDN et les applications SDN, permettant la personnalisation du comportement du réseau en fonction des besoins spécifiques de l'entreprise ou de l'application.

5. Dispositifs Réseau (Commutateurs) : Les commutateurs SDN transfèrent les paquets de données en suivant les décisions prises par le contrôleur, simplifiant leur rôle par rapport aux dispositifs réseau traditionnels.

6. Séparation Du Plan De Contrôle Et Du Plan De Données : Un principe fondamental du SDN au le contrôleur gère le plan de contrôle (décisions sur le transfert des données), tandis que le plan de données (transfert réel des paquets) est mis en œuvre dans les commutateurs.

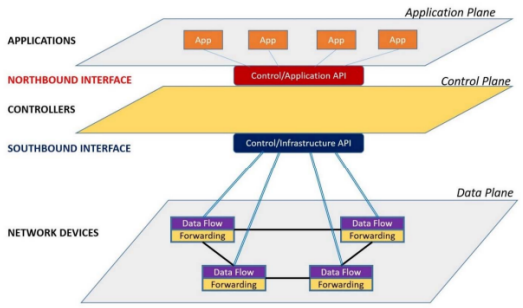


FIGURE 1.1 – Architecture Des Réseaux SDN

7. Hôtes (Hosts) : Les hôtes (clients, serveurs, appareils) dans un réseau SDN communiquent avec le contrôleur SDN pour recevoir des instructions sur leur comportement réseau. Des services comme FTP (transfert de fichiers), Iperf (mesure de performances) et des serveurs web peuvent être hébergés dans le réseau SDN. Le contrôleur SDN peut gérer le trafic vers/depuis ces services en définissant des règles de routage, de qualité de service, etc. Cela permet d'intégrer et d'optimiser ces services grâce aux capacités de programmation et de contrôle du SDN sur le réseau.

8. Protocole OpenFlow : OpenFlow est un protocole largement utilisé normalisant la communication entre le contrôleur SDN et les dispositifs réseau, favorisant ainsi l'interopérabilité et des solutions indépendantes du fournisseur. [Saleh Asadollahi, 2024]

1.1.2 SDN Et La Mise En Réseau Traditionnelle

Comparé aux réseaux traditionnels, le SDN apporte une évolutivité et une flexibilité accrues en déplaçant le contrôle du réseau du matériel vers le logicieCela permet une gestion centralisée et simplifiée des ressources réseau, favorisant ainsi la segmentation du réseau et une configuration plus rapide via une interface utilisateur centralisée (figure 1.2).

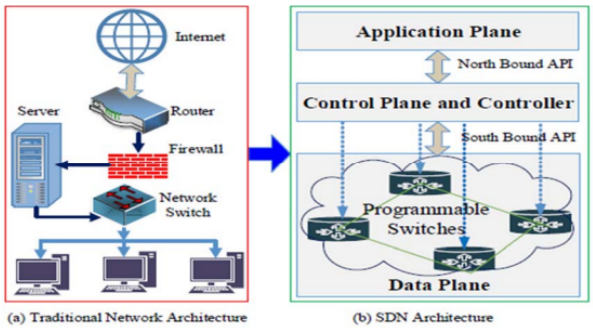


FIGURE 1.2 – SDN vs Réseau Traditionnelle

[chinacablesbuy, 2018]

1.1.3 Vulnérabilités Et Menaces Dans Les Réseaux SDN

Le réseau SDN présente plusieurs vulnérabilités potentielles. Voici quelques-unes des vulnérabilités courantes :

- **Attaques Par Déni De Service (DDoS) :** Les réseaux SDN peuvent être vulnérables aux attaques DDoS, où un grand volume de trafic est envoyé vers le contrôleur SDN ou les commutateurs, saturant les ressources disponibles et rendant le réseau indisponible pour les utilisateurs légitimes.
- **Attaques Sur Le Contrôleur SDN :** Comme le contrôleur SDN est le point central de contrôle du réseau, il est une cible potentielle pour les attaquants. Des vulnérabilités dans le logiciel du contrôleur pourraient être exploitées pour compromettre le contrôle du réseau ou provoquer des dysfonctionnements.
- **Attaques D'interception De Communication :** Comme la communication entre le contrôleur SDN et les commutateurs est généralement réalisée via des canaux ouverts, elle peut être sujette à des attaques d'interception. Les attaquants pourraient surveiller ou modifier le trafic entre le contrôleur et les commutateurs, compromettant ainsi la sécurité du réseau.
- **Falsification De Messages OpenFlow :** Les messages OpenFlow utilisés pour la communication entre le contrôleur SDN et les commutateurs pourraient être falsifiés par des attaquants, entraînant des modifications non autorisées dans la configuration des commutateurs ou des interruptions de service.
- **Attaques Sur Les Commutateurs SDN :** Les commutateurs SDN pourraient être vulnérables à des attaques directes visant à compromettre leur sécurité ou à contourner les politiques de sécurité définies par le contrôleur.
- **Vulnérabilités Liées Aux Applications SDN :** Les applications SDN développées pour personnaliser et étendre les fonctionnalités du réseau pourraient contenir des vulnérabilités, telles que des injections de code ou des défauts de sécurité, susceptibles d'être exploitées par des attaquants pour compromettre le

réseau.

1.2 Protocole Open Flow

Le protocole OpenFlow définit l'interface entre un contrôleur OpenFlow et un commutateur OpenFlow (figure 1.3), permettant au contrôleur de contrôler le traitement des paquets de données entrants.

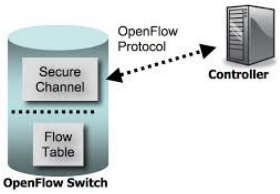


FIGURE 1.3 – Protocole Open Flow

1.2.1 Architecture Open Flow

L'architecture Open Flow se compose du commutateur OpenFlow qui reçoit et exécute les instructions du contrôleur OpenFlow pour traiter les paquets de données. Le contrôleur OpenFlow qui gère les commutateurs dans le réseau et peut être implémenté via différents logiciels, comme Ryu. Les applications de contrôle qui s'exécutent sur le contrôleur et mettent en œuvre la logique de gestion du réseau, telles que le routage et la sécurité. L'interface Sud qui permet la communication entre le contrôleur et les commutateurs via le protocole OpenFlow, et l'interface Nord qui facilite la communication entre le contrôleur et les applications de contrôle pour définir les politiques réseau et récupérer des informations sur l'état du réseau.

1.3 Sécurité Informatique

La sécurité recouvre l'ensemble de technique informatiques, organisationnels, juridiques et humains permettant de réduire au maximum les chances de fuites d'information, de modification de données ou de détérioration des services, pour atteindre un certain niveau de protection et de réduire la vulnérabilité d'un système contre les menaces accidentelles ou intentionnelles. Elle engendre des solutions dont le but est de garantir les objectifs essentiels suivants [Abbas, 2023] :

1- La confidentialité : est le fait de s'assurer que l'information n'est accessible qu'à ceux dont l'accès est autorisé. Cela implique la mise en œuvre des algorithmes de chiffrement.

2- L'intégrité : consiste à garantir que les données sont bien celles que l'on croit être, qui n'ont subi aucune altération ou destruction volontaire ou accidentelle, et qu'ils conservent un format permettant leur utilisation au cours de la communication. On utilise le hachage comme fonction pour garantir cet objectif.

3- L'identification : est un moyen de « connaître » l'identité d'une entité, souvent à l'aide d'un identifiant.

4- La non-répudiation : que la transaction ne peut être niée par aucun des correspondants.
5- La disponibilité : est d'assurer le bon fonctionnement d'un système et d'être accessible, déchiffrable aux utilisateurs autorisés à l'endroit et au moment prévu.
6- L'authentification : permet de valider la légitimité de l'accès de l'entité (limiter l'accès uniquement aux personnes autorisées). En d'autres termes, elle consiste à assurer l'identité d'un utilisateur avant l'échange des données.

1.3.1 Attaque De Déni De Service Distribué

L'attaque par déni de service ou Denial of Service (DoS) est une cyber-attaque dans le monde informatique, dans lequel l'attaquant tente temporairement d'interrompre ou suspendre les services d'un appareil connecté au serveur/Internet pour créer une ressource informatique ou réseau inaccessible à ses utilisateurs légitimes.
Cependant Un déni de service distribué ou Distributed Denial Of Servise (DDoS) est l'une des attaques les plus courantes, la plus grave, la plus dangereuse et la plus difficile à identifier. Il s'agit d'une attaque à grande échelle qui empêche un utilisateur légitime d'accorder des services. Il s'agit d'une tentative malveillante de surcharger la victime du flot de trafic en ligne pour interrompre l'habituel trafic d'un réseau ou d'un serveur ciblé. Une DDoS nécessite un attaquant pour accéder au contrôle d'un ordinateur en ligne sur le réseau, chaque ordinateur est infecté par un malware et se transforme en bot (comme les zombies). Il dispose d'un contrôle d'accès à distance au groupe de bots (réseau de zombies), créent ainsi ce qu'on appel un Botnet (un ensemble de mots « robot » et« réseau »). Après la mise en place d'un botnet, l'attaquant peut envoyer des instructions au botnet directement via une télécommande système. Pour compliquer davantage l'identification et la défaite de l'attaque, un attaquant peut impliquer une usurpation d'adresse IP. Comme un bot est un utilisateur légitime, le trafic normal peut provoquer une surcharge et ralentir le serveur en cas de panne. [Afzaal, 2022]

1.3.2 Les Types D'attaques DDoS

Les types d'attaques DDoS sont souvent nommés ou décrits selon la terminologie du modèle de référence OSI (Open Systems Interconnection)[Afzaal, 2022, IBM, 2023d, Kime, 2022] :

Attaques Au Niveau De La Couche Application :

Les attaques les plus courantes de la couche application (7ème du modèle OSI) sont :

1. L'attaque Par Inondation HTTP (ou HTTP Flood) : elle vise à saturer un serveur web en lui envoyant un grand nombre de requêtes HTTP

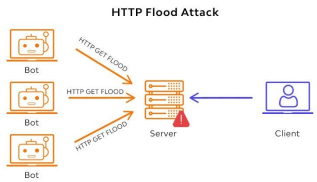


FIGURE 1.4 – Attaque HTTP Flood

(figure 1.4). Ces requêtes peuvent être légitimes ou non, et l'objectif est de consommer toutes les ressources du serveur, le rendant inaccessible aux utilisateurs légitimes. Les attaques HTTP Flood peuvent être lancées à partir de multiples sources, ce qui les rend difficiles à bloquer.

- **Attaques GET :** les attaquants utilisent un botnet pour envoyer un grand nombre de requêtes GET simultanées pour des fichiers volumineux tels que des fichiers PDF ou des vidéos volumineux.

2. L'attaque Slowloris : elle cible les serveurs web en les obligeant à maintenir de nombreuses connexions ouvertes simultanément, ce qui épuise les ressources du serveur et le rend inaccessible aux utilisateurs légitimes. L'attaque est réalisée en envoyant lentement des requêtes HTTP incomplètes au serveur, ce qui maintient les connexions ouvertes pendant une longue période sans en libérer de nouvelles.

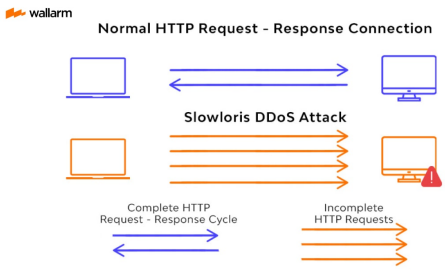


FIGURE 1.5 – Attaque Slowloris

Attaques Par Protocole :

Ciblent la couche réseau (3ème) et la couche transport (4ème) du modèle OSI. L'attaque par protocole la plus courante est[IBM, 2023d, Kime, 2022] :

1. Attaques Par Inondation SYN (ou SYN Flood) : une attaque par inondation SYN tire parti du protocole d'établissement de liaison TCP, le processus par lequel deux appareils établissent une connexion l'un avec l'autre. Lors d'un établissement de liaison TCP standard, un dispositif envoie un paquet **SYN** [le périphérique demandeur envoie un numéro de séquence synchronisé dans un paquet à un serveur ou à un autre périphérique de destination] pour initier la connexion, l'autre répond avec un paquet **SYN/ACK** [le numéro de séquence synchronisé plus un numéro d'accusé de réception (ACK)] pour confirmer la demande, puis le dispositif d'origine renvoie un paquet **ACK** [l'appareil demandeur renvoie un numéro d'accusé de réception de réponse (numéro ACK d'origine + 1) au serveur] pour finaliser la connexion.

Dans une attaque par inondation SYN, l'attaquant envoie au serveur cible un grand nombre de paquets SYN associés à des adresses IP source usurpées (figure 1.6). Le serveur envoie sa réponse à l'adresse IP usurpée et attend le paquet ACK

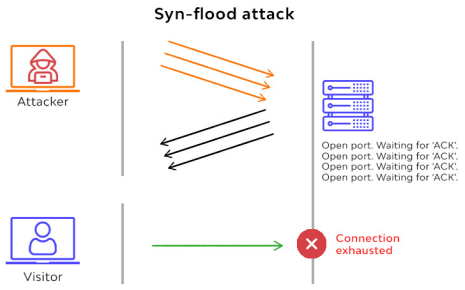


FIGURE 1.6 – Attaque SYN Flood

final, qui n'arrivent jamais. Le serveur est bloqué par un grand nombre de connexions inachevées, ce qui le rend indisponible pour les établissements de liaison TCP légitimes.

Attaques Volumétriques :

Les attaques DDoS volumétriques consomment toute la bande passante disponible au sein d'un réseau cible ou entre un service cible et le reste d'Internet, empêchant ainsi les utilisateurs légitimes de se connecter aux ressources du réseau. Elles inondent souvent les réseaux et les ressources avec des volumes de trafic très élevés, même par rapport aux autres types d'attaques DDoS. Les types courants d'attaques volumétriques comprennent [IBM, 2023d, Kime, 2022] :

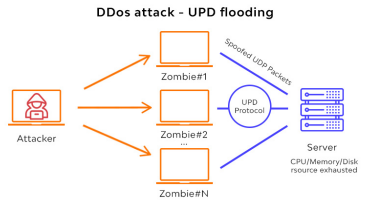


FIGURE 1.7 – Attaque UDP Flood

1. Inondations UDP (ou UDP Flood) : ces attaques consistent à envoyer de faux paquets UDP (User Datagram Protocol) aux ports d'un hôte cible, invitant ce dernier à rechercher une application qui recevra les paquets vu que UDP n'établit pas de session bidirectionnelle avec un serveur, ce qui rend l'attaque plus difficile à filtrer (figure 1.7).

2. Inondations ICMP (ou ICMP (Ping) Flood) : également appelées « attaques par inondation ping », ces attaques bombardent les cibles de demandes d'écho ICMP provenant de plusieurs adresses IP usurpées. Contraint de répondre à toutes ces requêtes, le serveur cible est surchargé et ne peut plus traiter les demandes d'écho ICMP valides (figure 1.8).

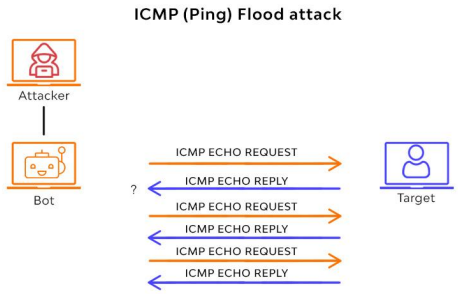


FIGURE 1.8 – Attaque ICMP Flood

Attaques Multi-vectorielles :

Comme leur nom l'indique, les attaques multi-vectorielles exploitent plusieurs vecteurs d'attaque afin de maximiser les dommages et de contrecarrer les efforts d'atténuation des attaques DDoS. Les attaquants peuvent utiliser plusieurs vecteurs simultanément ou passer d'un vecteur à l'autre en cours d'attaque, lorsque l'un d'eux est contrecarré. Par exemple, les pirates peuvent commencer par une attaque ICMP, puis, une fois que le trafic provenant des appareils du réseau est coupé, lancer une inondation UDP à partir de leur botnet.

1.4 Système De Détection D'intrusion (IDS)

La détection d'intrusion est le processus de surveillance d'un trafic réseau et d'analyse de celui-ci pour détecter des signes d'éventuelles intrusions, telles que des tentatives d'exploitation et des incidents pouvant constituer des menaces imminentes pour un réseau. Trois méthodologies de détection IDS sont généralement utilisées pour détecter les incidents [Loubna Dali, 2015, IBM, 2023c] :

— **La détection basée sur les signatures :** compare les signatures avec les événements observés pour identifier des incidents potentiels. C'est la méthode la plus simple, car elle compare uniquement l'unité d'activité en cours (telle qu'un paquet ou une entrée de journal) à une liste de signatures en utilisant des opérations de comparaison de chaîne.

— **La détection basée sur les anomalies :** compare les définitions d'une activité considérée comme normale avec les événements observés afin d'identifier les écarts significatifs. Cette méthode de détection peut se révéler très efficace pour repérer les menaces inconnues.

— **L'analyse dynamique des protocoles :** compare les profils prédéterminés des définitions généralement acceptées de l'activité bénigne du protocole, pour chaque état du protocole, avec les événements observés, afin d'identifier les écarts.

1.4.1 Architecture D'un Système De Détection D'intrusion

Elle se compose de 3 éléments essentiels comme le montre la figure 1.9 [Zamboni, 1998, BOUROUH Mouloud, 2017] :

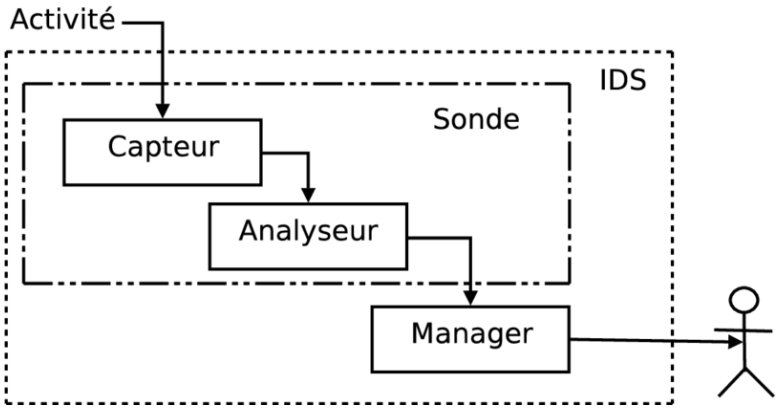


FIGURE 1.9 – Architecture IDS

- **Capteurs Des Données :** Ils sont chargés de surveiller le trafic réseau ou les journaux système pour détecter les activités suspectes ou les comportements malveillants.
- **Analyseur :** Il traite les données collectées par les capteurs pour détecter les signes

d'intrusion ou d'activité anormale. Il utilise des algorithmes et des règles préétablies pour identifier les menaces potentielles.

- **Manageur :** En plus de la notification des alertes, il offre à l'administrateur la possibilité de configurer une sonde et de gérer les alertes envoyées par l'analyseur.

1.4.2 Types De Systèmes De Prévention Contre Les Intrusions

Les IDS sont classés en fonction de leur emplacement dans un système et du type d'activité qu'ils surveillent [IBM, 2023c] :

Les Systèmes De Détection D'intrusion Réseau (NIDS) :

Un NIDS (Network Intrusion Detection System) est un système de détection d'intrusion qui surveille le trafic réseau pour détecter les activités suspectes ou malveillantes. Il analyse les copies de paquets de données qui traversent le réseau à la recherche de modèles correspondant à des signatures d'attaques connues ou à des comportements anormaux. Le NIDS peut être basé sur des règles prédéfinies ou utiliser des algorithmes d'apprentissage automatique pour détecter les menaces. Il peut fonctionner de manière passive, en alertant les administrateurs des violations détectées, ou de manière active, en prenant des mesures pour bloquer ou atténuer les attaques.

Les Systèmes De Détection D'intrusion Hôte (HIDS) :

Un HIDS (Host-based Intrusion Detection System) est un système de détection d'intrusion qui surveille et analyse les activités sur un seul système informatique (hôte). Contrairement au NIDS qui surveille le trafic réseau, le HIDS examine les journaux système, les fichiers de configuration et les modifications de fichiers pour détecter les signes d'intrusion ou d'activité anormale. Il utilise des signatures d'attaques connues, des règles prédéfinies et des algorithmes d'apprentissage automatique pour identifier les menaces potentielles. Le HIDS peut générer des alertes en cas de détection d'une intrusion et peut également prendre des mesures pour bloquer ou atténuer les attaques sur le système hôte.

Les Systèmes De Détection D'intrusion Hybride :

C'est la combinaison entre NIDS et HIDS afin de développer une vue complète pour surveiller le réseau et l'hôte simultanément avec un seul outil. Dans ce type d'IDS, les informations proviennent à la fois du réseau et des machines, ce qui les rend complexes, mais les avantages des deux IDS sont combinés. Pour cela, ce système est plus efficace par rapport aux autres systèmes de détection d'intrusions.

1.4.3 Limite Des Systèmes De Détection D'intrusion

Les limites des systèmes de détection d'intrusion sont de plus en plus résolues grâce à l'apprentissage machine, la puissance de calcul et de stockage par les technologies cloud permettre d'améliorer la précision des résultats.
Cependant des problèmes restent fondamentaux selon les méthodes de détection. En effet Le modèle de comportement normal des systèmes est basé sur des données collectées sur une période de fonctionnement normal ; les activités intrusives manquées pendant cette période sont susceptibles d'être considérées comme des comportements normaux. Ainsi certains adversaires peuvent biaiser le processus en compromettant les données des IDS. Aussi les techniques de détection peuvent difficilement détecter les attaques furtives parce que ces types d'attaques sont généralement cachés dans un grand nombre d'instances de comportements normaux. De plus, les types de paramètres utilisés comme entrées des modèles normaux sont généralement décidés par des experts en sécurité. Toute erreur survenant au cours du processus de définition de ces paramètres augmentera le taux de fausses alarmes et diminuera l'efficacité du système de détection d'intrusion.
Par conséquent, la conception des méthodes de détection et la sélection des caractéristiques du système ou du réseau à surveiller sont deux des principaux problèmes ouverts dans la détection d'intrusion. [Illy, 2018]

CHAPITRE 2

L'INTELLIGENCE ARTIFICIELLE

2.1 L'intelligence Artificielle

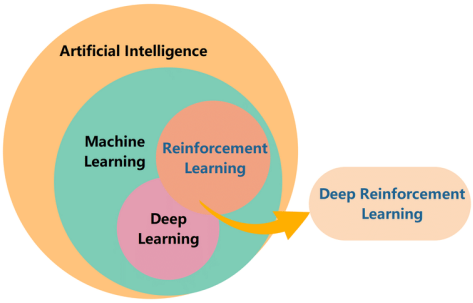


FIGURE 2.1 – Relation entre AI, ML, RL et DL

On utilise le terme "d'intelligence artificielle" ou d'IA pour désigner les ordinateurs et programmes informatiques avec des puissances de calcul capables de performances habituellement associées à l'intelligence humaine. Par exemple, la capacité à interagir avec l'homme, à traiter de grandes quantités de données ou encore à apprendre progressivement et donc à s'améliorer de manière continue. Elle comprend également les sous-domaines du machine learning et du deep learning, qui sont souvent utilisés en conjonction avec l'intelligence artificielle (figure 2.1). [Microsoft, 2023, IBM, 2023b]

2.1.1 Les Types De L'intelligence Artificielle

AI Faible (Narrow AI)

L'IA faible, ou IA étroite (ANI, Artificial Narrow Intelligence), est une IA conçue pour accomplir une tâche spécifique. Elle est « faible » non pas parce qu'elle est inefficace, mais parce qu'elle est limitée à une fonction précise. [Microsoft, 2023, IBM, 2023b]

AI Forte (General AI & Super AI)

L'intelligence artificielle générale (AGI), ou IA générale est une forme théorique d'intelligence artificielle qui serait dotée d'une conscience autonome capable de résoudre des problèmes généraux sans être spécifiquement programmée pour chacun.
La super intelligence artificielle (ASI), également appelée super intelligence, surpasserait l'intelligence et les capacités du cerveau humain. [Microsoft, 2023, IBM, 2023b]

L'IA Symbolique Et IA Connexionniste

L'IA symbolique, aussi appelée IA classique, fonctionne sur la base de règles explicites inscrites dans le code par les programmeurs. Elle est très efficace pour résoudre des problèmes définis avec des règles claires, comme les jeux d'échecs, mais elle a du mal à apprendre de nouvelles tâches sans programmation explicite.
L'IA connexionniste, en revanche, se base sur des réseaux de neurones artificiels pour « apprendre » à accomplir des tâches. C'est cette approche qui est utilisée dans le machine learning et le deep learning, où les systèmes apprennent à partir de grandes quantités de données sans programmation explicite. [Microsoft, 2023, IBM, 2023b]

2.2 Apprentissage Automatique (Machine Learning)

L'apprentissage automatique (ML) est une sous-catégorie de l'IA qui utilise des algorithmes pour apprendre automatiquement des informations et reconnaître des modèles à partir des données, appliquant ainsi cet apprentissage pour prendre des décisions de plus en plus judicieuses. Le machine learning peut être supervisé ou non supervisé. [Columbia, 2023]

2.2.1 L'Apprentissage Supervisé

L'apprentissage supervisé, est l'utilisation d'ensembles de données étiquetées (Features/Label). Cette méthode permet de réaliser deux types de tâches [IBM, 2021c] :
La classification : utilise un algorithme pour classer avec précision les données de test dans des catégories spécifiques.
La régression : ici, on n'attribue pas une classe mais une valeur mathématique (un pourcentage ou une valeur absolue).

2.2.2 L'apprentissage Non Supervisé

Contrairement à l'apprentissage supervisé, les données sont non étiquetées, et on doit soit les regrouper par distance ou par similarité (Clustering) ou trouver une relation importante entre les données (Association). [Google, 2020, datascientest, 2024, IBM, 2021a]

2.2.3 Apprentissage par renforcement (Reinforcement Learning)

Est la tâche d'apprendre à partir d'expériences successives. Ce qu'il convient de faire de façon à trouver la meilleure solution. Dans un tel problème, on dit qu'un agent (l'algorithme) interagit avec «l'environnement » pour trouver la solution optimale.

2.2.4 Algorithmes ML De Classification

Arbre De Décision (Decision Tree)

Est un algorithme utilisé pour la classification et la régression. Il divise les données en sous-groupes homogènes en fonction des caractéristiques des données. Chaque nœud de l'arbre représente une caractéristique, chaque branche une décision basée sur cette caractéristique, et chaque feuille une classe ou une valeur de prédiction. L'arbre est construit de manière récursive en choisissant à chaque étape la caractéristique qui divise le mieux les données en sous-groupes homogènes (figure 2.2). L'arbre est ensuite utilisé pour prédire la classe ou la valeur cible d'une nouvelle instance en la faisant passer à travers l'arbre selon les règles de division apprises.[IBM, 2023a]

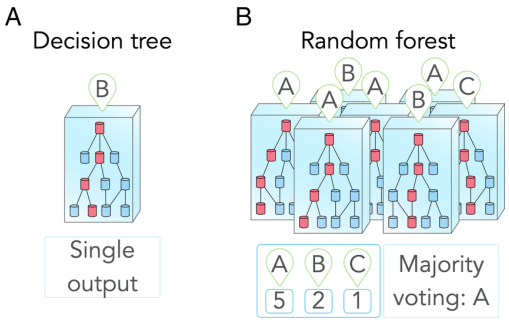


FIGURE 2.2 – Decision Tree vs Random Forest

Forêt D'arbre De Décision (Random Forest)

Est un algorithme qui permet d'assembler les sorties de plusieurs arbres de décision pour atteindre un résultat unique. Chaque arbre de décision est construit de manière aléatoire à partir d'un sous-ensemble aléatoire des données d'entraînement et des caractéristiques. L'algorithme agrège les prédictions de chaque arbre pour obtenir une prédiction finale (figure 2.2). Random Forest est efficace pour éviter le sur-apprentissage et pour traiter des ensembles de données complexes avec de nombreuses caractéristiques.[IBM, 2021b]

Naive Bayes

Est un algorithme basé sur le théorème de Bayes, qui suppose l'indépendance conditionnelle entre chaque paire de caractéristiques (équation 2.1). Il est souvent utilisé pour la classification de texte et de documents. L'algorithme calcule la probabilité qu'une

instance appartienne à chaque classe en se basant sur les fréquences des caractéristiques dans chaque classe. Il peut donner de bons résultats, pour les données de grande taille et de haute dimension. Cependant, son hypothèse d'indépendance peut être trop simpliste pour certains problèmes.[IBM, 2021e]

$$P(C_k|x_1, x_2, \ldots, x_n) = \frac{P(C_k) \cdot P(x_1, x_2, \ldots, x_n|C_k)}{P(x_1, x_2, \ldots, x_n)} \tag{2.1}$$

P est souvent décomposée en utilisant l'hypothèse d'indépendance conditionnelle :

$$P(x_1, x_2, \ldots, x_n|C_k) = P(x_1|C_k) \cdot P(x_2|C_k) \cdot \ldots \cdot P(x_n|C_k) \tag{2.2}$$

Régression Logistique (Logistic Regression)

Est un algorithme utilisé pour la classification binaire, qui prédit la probabilité qu'une observation appartienne à une classe particulière. Elle utilise une fonction logistique (équation Sigmoid 2.3) pour modéliser la relation entre les variables indépendantes et la variable dépendante. L'algorithme cherche à ajuster les paramètres du modèle de manière à maximiser la probabilité d'observer les données d'entraînement. Elle est largement utilisée en raison de sa simplicité, de son interprétabilité et de sa capacité à gérer des problèmes de classification simples et rapides. Cependant, elle peut être limitée pour des tâches de classification complexes ou non linéaires.[AWS, 2023]

$$f(x) = \frac{1}{1 + e^{-x}} \tag{2.3}$$

2.3 Apprentisage Profond (Deep Learing)

Le deep learning ou apprentissage profond est un type d'intelligence artificielle dérivé du machine learning qui se concentre spécifiquement sur les réseaux neuronaux avec plusieurs couches où la machine est capable d'apprendre par elle-même. Ces réseau de neurones artificiels s'inspire du cerveau humain. Ce réseau est composé de dizaines voire de centaines de «couches» de neurones, chacune recevant et interprétant les informations de la couche précédente. [Deluzarache, 2023]

2.3.1 Réseaux De Neurones Artificiel (Artificial Neural Network ANN) :

Les réseaux de neurones, inspirés par le fonctionnement du cerveau humain, imitent la manière dont les neurones biologiques transmettent des signaux.
Un réseau de neurones artificiels est composé de couches de nœuds, comprenant généralement une couche d'entrée, une ou plusieurs couches cachées et une couche de

sortie. Chaque nœud, ou neurone artificiel, est connecté à d'autres nœuds et possède des poids et des seuils.

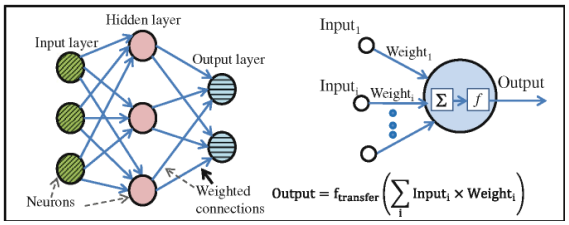


FIGURE 2.3 – Réseaux De Neurones Artificiel

Lorsque la sortie d'un nœud dépasse son seuil, il est activé et envoie des données à la couche suivante du réseau comme le montre la figure 2.3. Ils apprennent à partir de données d'entraînement, ajustant leurs poids et leurs seuils pour améliorer leur précision. Une fois entraînés, ces réseaux peuvent être utilisés pour classer et regrouper rapidement les données. [IBM, 2021d]

Perceptrons multicouches (MLP)

MLP est un réseau de neurones artificiels Feedforward, composé d'une couche d'entrée, d'une ou plusieurs couches cachées et d'une couche de sortie. Lorsqu'un MLP a plus d'une couche cachée, il est considéré comme un MLP profond.Les éléments d'un MLP sont :
Poids : Permettent de déterminer l'importance d'une variable donnée.
Biais : C'est un paramètre supplémentaire qui est utilisé pour ajuster la courbe la frontière de décision afin de s'adapter au mieux aux données pour obtenir une sortie bien précise.
Fonction d'activation : Elles permettent l'introduction de caractéristiques non linéaires dans le réseau comme :

Fonction d'activation	Formule mathématique	Propriétés
Sigmoid	$\sigma(x) = \frac{1}{1+e^{-x}}$	— Transforme l'entrée en une valeur comprise entre 0 et 1. — Utilisée dans les couches de sortie pour la classification binaire.
Tanh	$\tanh(x) = \frac{e^{x}-e^{-x}}{e^{x}+e^{-x}}$	— Transforme l'entrée en une valeur comprise entre -1 et 1. — Centrée sur zéro, ce qui peut aider à la convergence.
ReLU (Rectified Linear Unit)	$\mathrm{ReLU}(x) = \max(0, x)$	— Renvoie x si x > 0, sinon renvoie 0. — Convergence rapide et largement utilisée dans les réseaux de neurones profonds.

TABLE 2.1 – Fonctions D'activation Courantes

Pour développer un modèle linéaire, il faut effectuer tout d'abord la somme pondérée des entrées multipliées par leurs poids W_n respectives plus une coefficient complémentaire b :

$$\mathrm{Z} = \sum w_i x_i + \mathrm{bias} = w_1 x_1 + w_2 x_2 + w_3 x_3 + \mathrm{bias} \tag{2.4}$$

Ensuite, on applique la somme à la fonction d'activation. Enfin, le résultat obtenu est transmis aux neurones suivants :

$$\mathrm{Y} = \mathrm{A(Z)} \tag{2.5}$$

Ce processus définit ce réseau de neurones comme un réseau à propagation avant (Forward propagation). Au fur et à mesure que nous entraînons le modèle, nous voulons évaluer sa précision à l'aide d'une fonction de coût. Il existe plusieurs fonctions de coût comme :

$$\text{Mean Square Error} = \mathrm{MSE} = \frac{1}{n}\sum_{i=1}^{n}(y_i - \hat{y}_i)^2 \tag{2.6}$$

$$\text{Mean Absolute Error} = \mathrm{MAE} = \frac{1}{n}\sum_{i=1}^{n}|y_i - \hat{y}_i| \tag{2.7}$$

$$\text{Root Mean Squared Error} = \mathrm{RMSE} = \sqrt{\frac{1}{n}\sum_{i=1}^{n}(y_i - \hat{y}_i)^2} \tag{2.8}$$

En fin de compte, l'objectif est de minimiser notre fonction de coût à mesure que le modèle ajuste ses poids et ses biais (Backpropagation). La méthode utilisée par l'algorithme pour ajuster ses poids est celle de la descente de gradient. Elle permet au modèle de déterminer la direction à prendre pour réduire les erreurs. Elle représente la dérivée de la fonction de coût :

1. Si la dérivée est négative, il faudra augmenter W si l'on veut réduire nos erreurs.
2. Si la dérivée est positive, il faudra diminuer W si l'on veut réduire nos erreurs.

2.4 Autoencoder (AE)

Un autoencoder est un type de réseau de Deep Learning entraîné pour répliquer ses données d'entrée. Il est constitué de trois parties [Dave Bergmann, 2023, Kamalov et al., 2021] :

1. **Encodeur :** Est la couche qui consiste à coder ou compresser les données d'entrée (Features) les plus importante à apprendre.
2. **Bottleneck (code) :** Contient la représentation la plus compressée de l'entrée : c'est à la fois la couche de sortie du réseau d'encodage et la couche d'entrée du réseau de décodage.
3. **Décodeur :** Est une couche qui reconstruit les données de sortie à partir de leur forme codée.

2.4.1 Autoencoder Pour La Détection D'intrusions

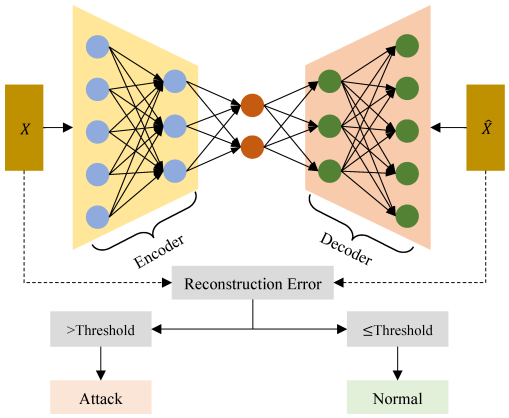


FIGURE 2.4 – Fonctionnement D'un Autoencodeur Lors De La Détection D'intrusions

L'approche proposée pour détecter d'intrusions IDS implique initialement de former l'auto-encodeur sur des données normales. Puis classer les nouveaux flux de trafic en fonction de la reconstruction de perte. Si la perte de reconstruction est importante, alors l'IDS signale la fluidité du trafic. Dans le cas contraire, elle est considérée comme bénigne, comme l'illustre la figure 2.4.[Dave Bergmann, 2023, Kamalov et al., 2021]

2.5 Processus De Développement D'un Modèle De Détection Via L'AI

Dans cette section, nous nous intéressons au processus de l'utilisation de l'algorithme DL *Autoencoder* dans la détection des intrusions. Les deux premières étapes, **Processus De Préparation Des Données** et **Pré-traitement Des Données**, seront expliquées en détail dans le chapitre suivant.

2.5.1 Fractionnement Des Données

Les données sont fractionnées différemment pour le machine learning et le deep learning :

ML		DL		
Training Data	Test Data	Training Data	Validation Data	Test Data
70% - 80%	20% - 30%	55% - 60%	15% - 20%	20% - 30%

TABLE 2.2 – Fractionnement Des Données En ML/DL

2.5.2 Indicateurs De Performance

D'abord voici une figure(2.5) qui explique c'est quoi Un True Positive (TP), un True Negative (TN), un False Positive (FP) et un False Negative (FN) :

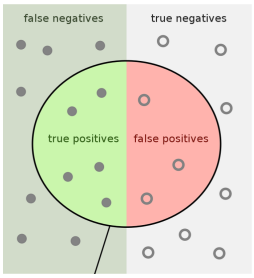


FIGURE 2.5 – TP vs TN vs FP vs FN

1. Vrais positifs (True Positives, TP) : Ces points se trouvent dans la partie verte à l'intérieur du cercle. Ils représentent les instances que le modèle a correctement prédit comme positives (par exemple, prédire une attaque DDoS correctement).
2. Faux positifs (False Positives, FP) : Ces points se trouvent dans la partie rouge à l'intérieur du cercle. Ils représentent les instances que le modèle a incorrectement prédit comme positives (par exemple, prédire une attaque DDoS alors qu'il n'y en a pas).
3. Faux négatifs (False Negatives, FN) : Ces points se trouvent dans la partie verte à l'extérieur du cercle. Ils représentent les instances que le modèle a incorrectement prédit comme négatives (par exemple, ne pas prédire une attaque DDoS alors qu'il y en a une).
4. Vrais négatifs (True Negatives, TN) : Ces points se trouvent dans la partie grise à l'extérieur du cercle. Ils représentent les instances que le modèle a correctement prédit comme négatives (par exemple, ne pas prédire une attaque DDoS correctement).
En utilisant ces derniers, on calcule les indicateurs de performance (Accuracy & Precision & Recall & F- score). Le tableau suivant 2.3 l'illustre :

Accuracy	Precision	Recall	F- score
$\frac{TP + TN}{TP + FP + TN + FN}$ (2.9)	$\frac{TP}{TP + FP}$ (2.10)	$\frac{TP}{TP + FN}$ (2.11)	$2 \times \frac{Precision \times Recall}{Precision + Recall}$ (2.12)

TABLE 2.3 – Indicateurs De Performance

1. Accuracy (Précision) : C'est le ratio des exemples correctement prédits par le modèle par rapport à l'ensemble des exemples.
2. Precision (Précision) : C'est le ratio des exemples positifs correctement prédits par le modèle par rapport à l'ensemble des exemples prédits comme positifs.
3. Recall (Rappel) : C'est le ratio des exemples positifs correctement prédits par le modèle par rapport à l'ensemble des exemples réellement positifs.
3. F-score (Score F) : C'est une mesure qui combine à la fois la précision et le rappel en une seule valeur.

CHAPITRE 3

IMPLÉMENTATION, MISE EN ŒUVRE DE L'APPLICATION ET TESTS

3.1 Introduction

Pour entraîner notre modèle d'intelligence artificielle, nous avons utilisé Mininet pour générer du trafic et simuler des scénarios réseau variés. En parallèle, nous avons exploité Ryu pour contrôler et surveiller le trafic dans notre réseau SDN. Grâce à cette combinaison d'outils, nous avons pu collecter un ensemble de données comprenant à la fois des exemples d'attaques et de trafic normal, essentiels pour entraîner notre modèle.
Ce chapitre sera dédié à la présentation détaillée de notre application, en commençant par les configuration de Ryu et Mininet. Nous aborderons ensuite les tests et traitements réalisés pour évaluer la performance et la fiabilité de notre système, ainsi que les résultats obtenus.

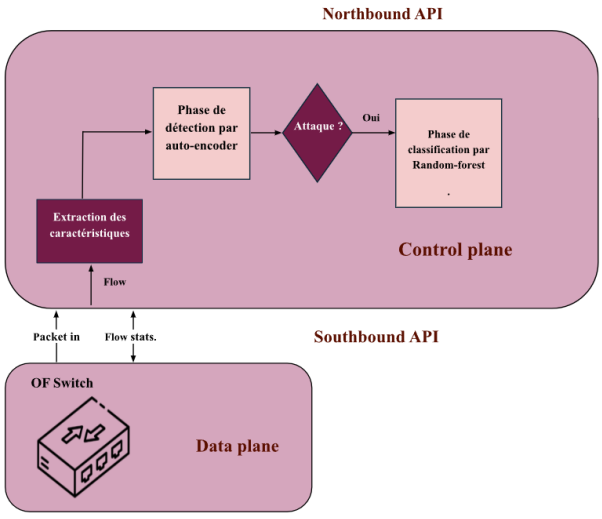


FIGURE 3.1 – Diagramme Explicatif De La Partie SDN

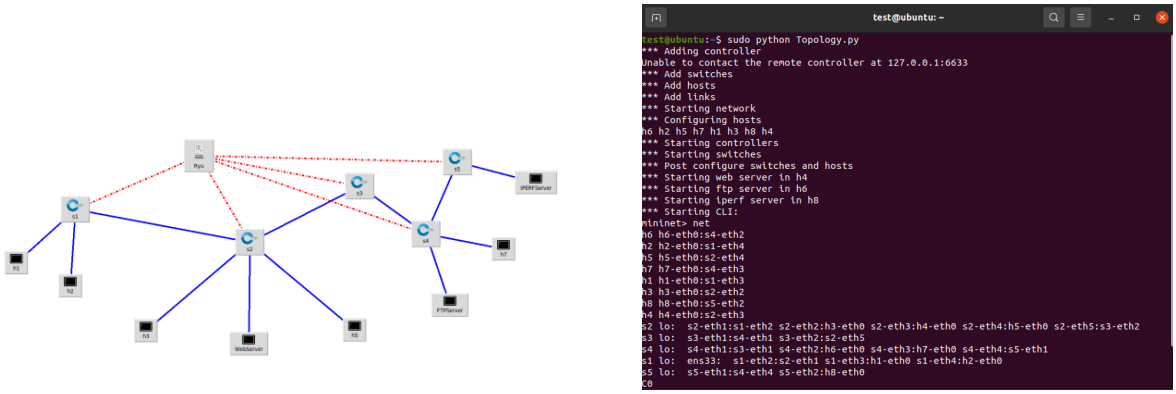
3.2 Environnement De Tests Des Modèles

Les tests et la réalisation de notre solution ont été effectués dans un environnement expérimental sous Visual Studio Code ayant les caractéristiques suivantes :

- **Processeur :** AMD Ryzen 5 5600H with Radeon Graphics, 3301 MHz, 6 cœur(s), 12 processeur(s) logique(s)
- **RAM :** 16.0 Go
- **Disque Dur :** 1 To
- **Navigateur :** Opera GX Browser

3.3 Configuration De L'environnement De Test Avec Mininet Sur VMware

3.3.1 Topologie



(a) Notre Toplogie Dans Mininet (b) Exécution Du Fichier De La Topologie

FIGURE 3.2 – Notre Topologie

En raison de la politique de sécurité de l'entreprise Ericsson, qui ne fournit pas son propre ensemble de données, nous avons été contraints de créer notre propre dataset de trafic. Nous avons configuré une topologie comprenant plusieurs switches et hôtes en utilisant les fonctionnalités de Mininet. Les switches ont été connectés entre eux et aux hôtes pour établir un réseau cohérent. Un contrôleur SDN a été ajouté au réseau pour gérer le trafic. Chaque switch a été associé au contrôleur lors du démarrage du réseau (figure 3.2).
Pour tester la connectivité et démarrer des services, des adresses IP ont été attribuées aux hôtes, et des services tels que des serveurs web, FTP et Iperf ont été lancés.

3.3.2 Processus De Préparation Des Données Avec Contrôleur SDN

Nous avons développé un contrôleur SDN en utilisant le framework Ryu, afin de collecter des statistiques sur le trafic réseau dans notre environnement SDN simulé (figure 3.1). Son objectif principal est de recueillir des données telles que les adresses IP source et de destination, les types de protocole, les ports utilisés, etc. Le fonctionnement du contrôleur implique la réception d'événements réseau et l'analyse des statistiques de flux envoyées par les commutateurs. Les données collectées sont ensuite formatées et enregistrées dans un fichier CSV. Le contrôleur prend en charge plusieurs protocoles réseau, notamment IPv4, ICMP, TCP, UDP et ARP, et traite chaque type de protocole pour collecter les statistiques requises. Ce script fonctionne en arrière-plan pendant que la simulation est en cours, permettant une collecte continue des statistiques de trafic pour notre évaluation.

Simulation de trafic et collecte de données avec SDN dans Mininet

Dans notre configuration, les hôtes de la topologie Mininet sont lancés automatiquement lors de son démarrage. Pour générer du trafic, certains hôtes sont configurés avec des services spécifiques tels que des serveurs web, des serveurs FTP ou IPERF (Figure 3.3).

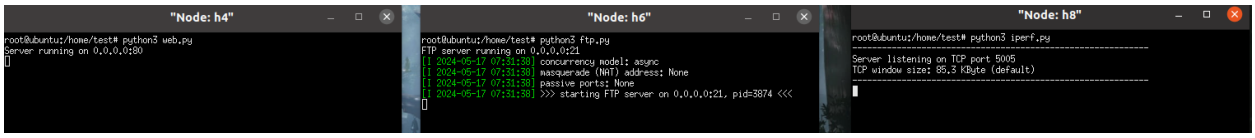
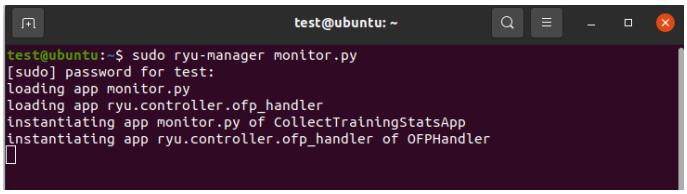
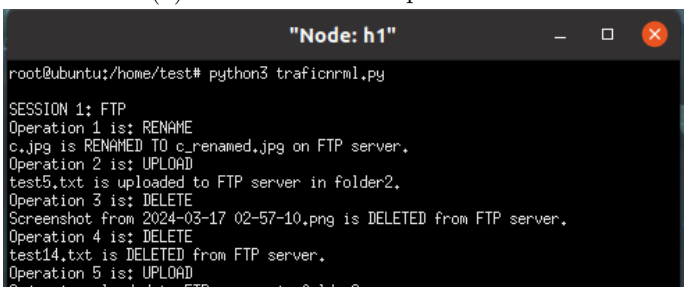


FIGURE 3.3 – Configuration Des Hosts

Ces services génèrent du trafic correspondant à leur fonctionnalité dès le démarrage. Parallèlement, notre script contrôleur SDN (Monitor) collecte en temps réel des statistiques sur le trafic en envoyant des requêtes aux commutateurs. Ces données, incluant les adresses IP et les types de protocole, sont enregistrées dans un fichier CSV pour une analyse ultérieure. Cette approche nous permet de simuler divers types de trafic tout en recueillant des données pour l'évaluation et l'analyse de notre réseau Mininet. Voir la figure 3.4.
Pour simuler des attaques de type UDP flood, SYN flood, Slowloris, ICMP flood et HTTP flood, nous avons adopté une approche similaire à celle utilisée pour générer du trafic normal. Au lieu de lancer le script de surveillance, nous avons exécuté des scripts d'attaque spécifiques sur notre contrôleur SDN.
Chaque script d'attaque était conçu pour cibler un type spécifique d'attaque. Par exemple, pour l'UDP flood, nous avons utilisé un script qui envoyait un grand nombre de paquets UDP vers la cible, surchargeant ainsi sa capacité à traiter les demandes. De même, pour le SYN flood, nous avons lancé un script qui initiait un grand nombre de connexions SYN, épuisant les ressources du système cible.



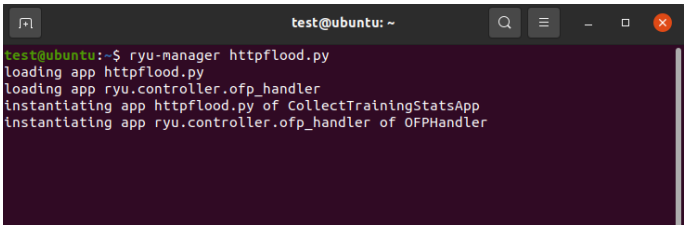
(a) Exécution Du Script Moniteur



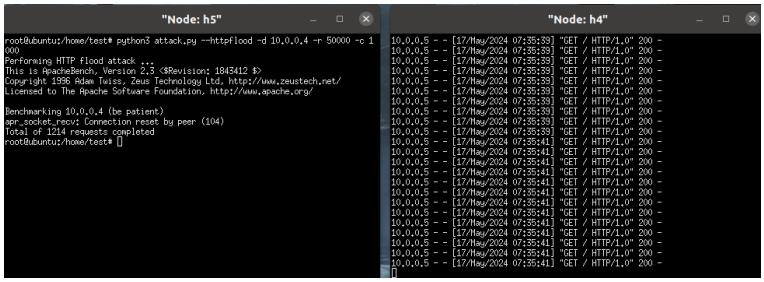
(b) Lancement D'un Trafic Normal

FIGURE 3.4 – Génération D'un Trafique Normal

Une fois les scripts d'attaque lancés, notre topologie Mininet était prête à recevoir et à traiter le trafic généré par ces attaques. Les données sur le trafic d'attaque étaient collectées en temps réel par notre contrôleur SDN, qui envoyait des requêtes aux commutateurs pour obtenir des informations sur les paquets entrants. Ces données étaient ensuite enregistrées dans le même fichier CSV ou on a stocké notre trafic normal. Voir la figure 3.5



(a) Exécution Du Script HTTP Flood



(b) Lancement D'un Attaque Contre Le Serveur Web

FIGURE 3.5 – Génération D'une Attack HTTP Flood

3.4 Pré-traitement des données

L'application des algorithmes de notre solution nécessite un Dataset soumis à certaines conditions pour améliorer la qualité globale des résultats dans l'entraînement et le test des

modèles.

Nettoyage des données

Dans notre processus de nettoyage des données, nous avons utilisé la méthode fillna() pour traiter les valeurs manquantes dans notre ensemble de données. Nous avons remplacé les valeurs manquantes dans des colonnes spécifiques telles que 'Arpcode', 'Icmpcode' et 'Icmptype' par '-1' pour indiquer leur absence, et dans un ensemble plus large de colonnes, nous les avons remplacées par '0'. Cette approche a été choisie pour maintenir l'intégrité des données tout en signalant clairement les valeurs manquantes, assurant ainsi la fiabilité de nos analyses ultérieures.

Correlation Analysis

Nous avons identifié plusieurs paires de variables fortement corrélées au sein de notre ensemble de données. Pour éviter les problèmes de multicollinéarité et garantir la fiabilité de nos analyses, nous avons choisi de retirer les variables présentant des corrélations élevées. Cette démarche nous permet de simplifier notre ensemble de données en éliminant les redondances et en nous concentrant sur les variables les plus pertinentes pour notre étude.
La figure 3.6 illustre le résultat qu'on a eu.

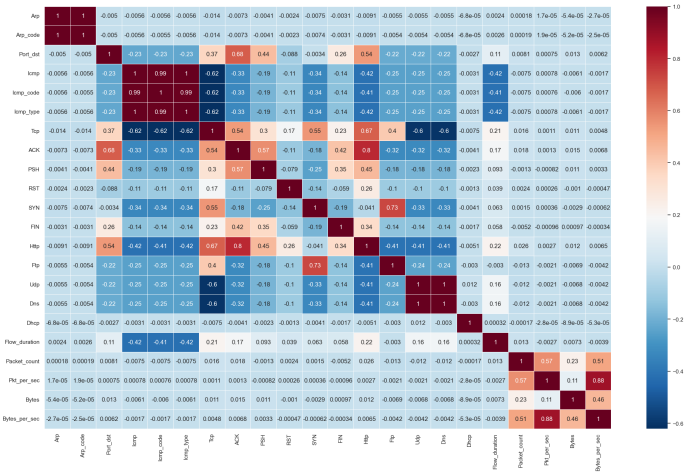


FIGURE 3.6 – Matrice De Correlation

Encodage des données

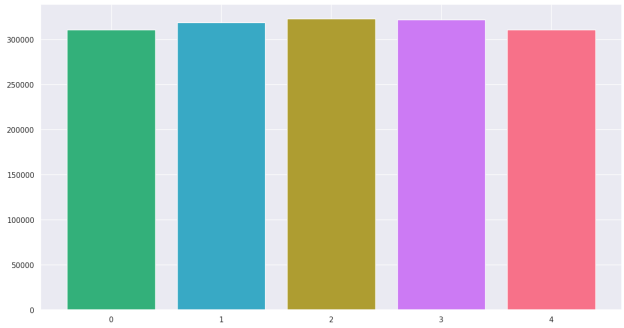


FIGURE 3.7 – Encodage De AttackType

Les deux modèles de notre solution ne traitent que des données numériques. Nous avons supprimé la colonne « AttackType » dans le dataframe correspondant au trafic normal, donc nous n'avons pas eu besoin de l'encoder. En revanche, il était nécessaire de le faire pour le deuxième dataframe. Pour cela, nous avons utilisé LabelEncoder() afin de convertir la colonne « AttackType », qui contient les types d'attaque. Comme il est illustrer dans la figure 3.7.

Resampling

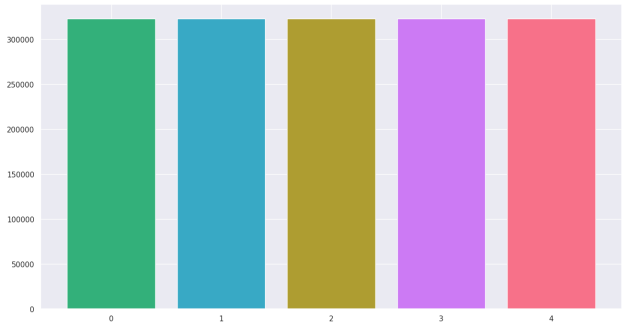


FIGURE 3.8 – Dataset Après Le Resampling

Lorsque nous avons visualisé les valeurs de type « Typesattaques » dans notre Dataset, nous avons remarqué qu'il existe le problème nommé Imbalanced Data (déséquilibrés). Nous avons utilisé l'algorithme SMOTE (Synthetic Minority Oversampling Technique) pour résoudre ce problème en ajoutant d'autres exemples de la classe minoritaire (Oversampling). La figure 3.7 illustre les types avant et la figure 3.8 après l'application de l'algorithme SMOTE.

3.4.1 Dataset Separation

Nous avons divisé notre ensemble de données en deux dataframes distincts : l'un pour la classe "attaque" et l'autre pour le trafic normal. Cette division nous permet d'analyser séparément les caractéristiques et les comportements associés à chaque classe. Tel que le DataFrame avec le trafic normal va être utilisé dans l'autoencoder et celui avec les attaques va être utilisé dans la partie classification.

3.4.2 Fractionnement des données

Pour le premier modèle nous avons utilisé le DataFrame contenant les données du trafic normal. Ensuite, nous l'avons divisé en trois ensembles distincts. Nous avons utilisé les fractions de 60% et 80% de la longueur totale du DataFrame pour déterminer les points de division. Ainsi, 60% des données ont été assignées à l'ensemble d'entraînement, 20% à l'ensemble de validation et les 20% restants à l'ensemble de test. Pour le deuxième modèle, nous avons utilisé le deuxième DataFrame, qui contient les données d'attaque. Nous avons choisi de diviser les données avec une proportion de 80% pour l'ensemble d'entraînement et 20% pour l'ensemble de test, en fixant une graine aléatoire (randomstate) à 42 pour assurer la reproductibilité des résultats.

3.4.3 Normalisation des données

Permet de rendre le graphe de distribution des données semblable à une distribution normale. Nous avons choisi d'utiliser la classe CountFrequencyEncoder pour la variable 'Portdst' dans nos ensembles de données d'entraînement, de validation et de test. En utilisant cette méthode, nous avons converti les catégories de 'Portdst' en valeurs numériques basées sur leur fréquence d'apparition dans les données d'entraînement. Et

nous avons utilisé la méthode StandardScaler pour 'Packet_count', 'Pkt_per_sec', 'Bytes' et 'Bytes_per_sec'. Cette méthode ajuste les données en soustrayant la moyenne et en divisant par l'écart type de chaque caractéristique, garantissant ainsi que chaque caractéristique a une moyenne nulle et une variance unitaire.

3.5 Création du modèle Autoencoder

Layer (type)	Output Shape	Param #
input_layer (InputLayer)	(None, 18)	0
dense (Dense)	(None, 20)	380
dense_1 (Dense)	(None, 15)	315
dense_2 (Dense)	(None, 5)	80
dense_3 (Dense)	(None, 15)	90
dense_4 (Dense)	(None, 20)	320
dense_5 (Dense)	(None, 18)	378

FIGURE 3.9 – Choix Du Nombre De Couches Et De Neurones

Nous avons conçu un autoencodeur en deux parties : l'encodeur et le décodeur (7 couches dense figure 3.9). L'encodeur utilise des couches denses avec des activations sigmoïde et tanh pour capturer les caractéristiques des données. Le décodeur, basé sur les sorties de l'encodeur, utilise des activations tanh et ReLU pour reconstruire les données originales.

Après la création du modèle, nous l'avons compilé avec l'optimiseur Adam, une fonction de perte basée sur l'erreur quadratique moyenne et une métrique de précision pour évaluer les performances du modèle lors de l'entraînement (Figure 3.10).

Ensuite, nous évaluons notre modèle autoencodeur sur l'ensemble de test en calculant la perte moyenne quadratique (MSE), accuracy, voir figure 3.11. Puis, on a défini un seuil de perte pour détecter les valeurs aberrantes. En utilisant ce seuil, nous prédisons les échantillons de données d'attaque et les classifions comme aberrants ou non aberrants en fonction de leur perte de prédiction. Enfin, nous comptons le nombre d'échantillons non aberrants.

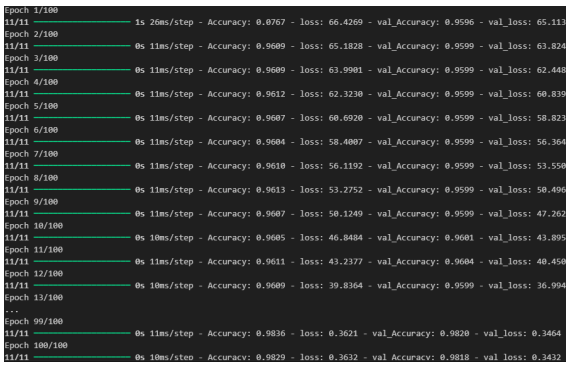


FIGURE 3.10 – Entrainement De L'autoencoder

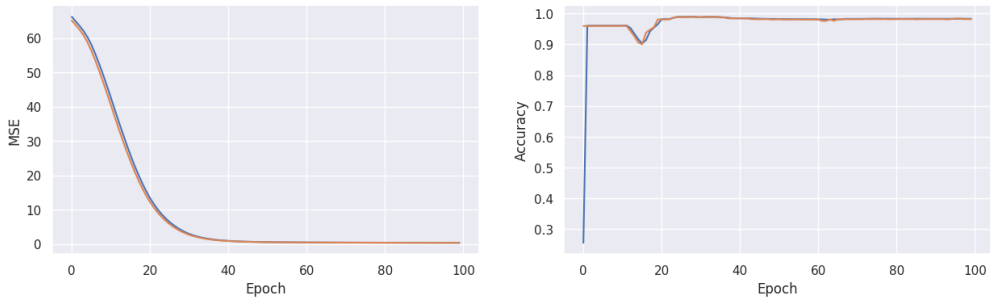
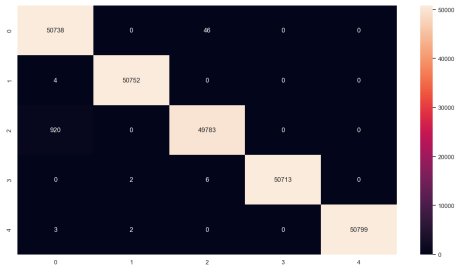


FIGURE 3.11 – Évolution De L'erreur Quadratique Moyenne (MSE) Et De La Précision (Accuracy) Au Fil Des Époques Pendant L'entraînement Du Modèle.

3.6 Modèle de classification de type d'attaques

Voici les données collectées qui montrent les performances de chaque algorithme :

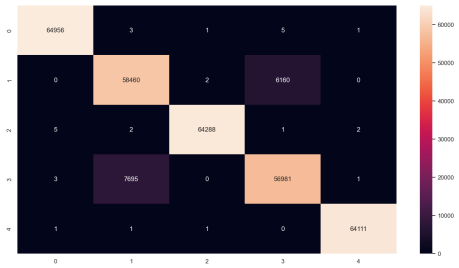
Algorithme Decision Tree Classifier avec les données Train & Test



(a) Matrice de confusion du modèle Decision Tree Classifier (Train)

```
___________Train Decision Tree Classifier___________
accuracy : 0.9600525288211231
mean absolute error : 0.0798988161646213
mean squared error : 0.15985728895500187
root mean squared error : 0.39982157139779473

Classification report Training
              precision    recall  f1-score   support

   SYN flood       1.00      1.00      1.00    257714
  http flood       0.89      0.91      0.90    258058
  icmp flood       1.00      1.00      1.00    258382
   slowloris       0.91      0.89      0.90    258000
   udp flood       1.00      1.00      1.00    258566

    accuracy                           0.96   1290720
   macro avg       0.96      0.96      0.96   1290720
weighted avg       0.96      0.96      0.96   1290720
```

(b) Raport De Classification Decision Tree Classifier (Train)



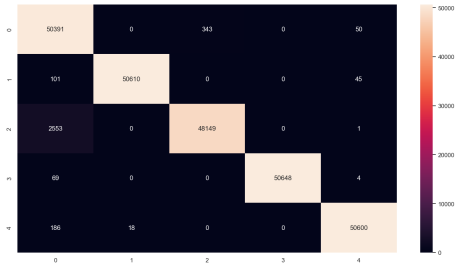
(c) Matrice de confusion du modèle Decision Tree Classifier (Test)

```
___________Test Decision Tree Classifier___________
accuracy : 0.9569728523614727
mean absolute error : 0.08606669145903062
mean squared error : 0.17223875046485682
root mean squared error : 0.41501656649446755

Classification report Test
              precision    recall  f1-score   support

   SYN flood       1.00      1.00      1.00     64966
  http flood       0.88      0.90      0.89     64622
  icmp flood       1.00      1.00      1.00     64298
   slowloris       0.90      0.88      0.89     64680
   udp flood       1.00      1.00      1.00     64114

    accuracy                           0.96    322680
   macro avg       0.96      0.96      0.96    322680
weighted avg       0.96      0.96      0.96    322680
```

(d) Raport De Classification Decision Tree Classifier (Test)

FIGURE 3.12 – Résultat Du Training & Testing Avec Decision Tree Classifier

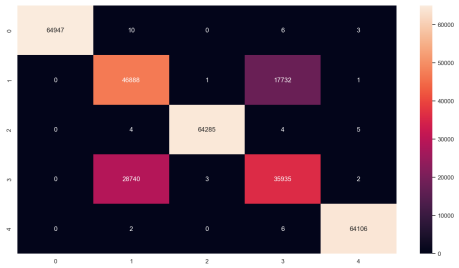
Algorithme Logistic Regression avec les données Train & Test



(a) Matrice de confusion du modèle Logistic Regression (Train)

```
_____Train Logistic Regression_____
accuracy : 0.85574485558448
mean absolute error : 0.2884723255237387
mean squared error : 0.5770585409693815
root mean squared error : 0.7596436934309279

Classification report Training
              precision    recall  f1-score   support

   SYN flood       1.00      1.00      1.00    257714
  http flood       0.62      0.72      0.67    258058
  icmp flood       1.00      1.00      1.00    258382
   slowloris       0.67      0.55      0.61    258000
    udp flood      1.00      1.00      1.00    258566

    accuracy                           0.86   1290720
   macro avg       0.86      0.86      0.85   1290720
weighted avg       0.86      0.86      0.85   1290720
```

(b) Raport De Classification Logistic Regression (Train)



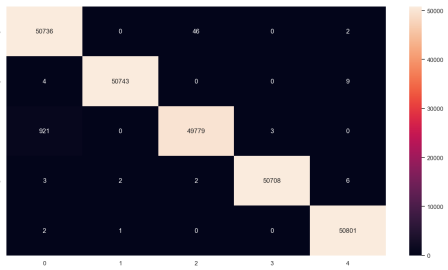
(c) Matrice de confusion du modèle Logistic Regression (Test)

```
_____Test Logistic Regression_____
accuracy : 0.8558355026651792
mean absolute error : 0.28828250898723196
mean squared error : 0.576630979298377
root mean squared error : 0.759361638437074

Classification report Test
              precision    recall  f1-score   support

   SYN flood       1.00      1.00      1.00     64966
  http flood       0.62      0.73      0.67     64622
  icmp flood       1.00      1.00      1.00     64298
   slowloris       0.67      0.56      0.61     64680
    udp flood      1.00      1.00      1.00     64114

    accuracy                           0.86    322680
   macro avg       0.86      0.86      0.86    322680
weighted avg       0.86      0.86      0.85    322680
```

(d) Raport De Classification Logistic Regression (Test)

FIGURE 3.13 – Résultat Du Training & Testing Avec Logistic Regression

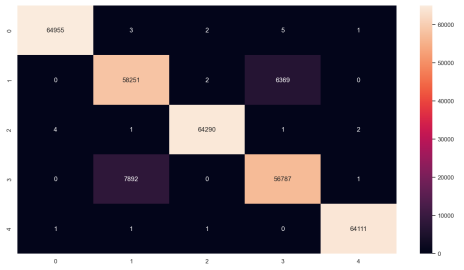
Algorithme Random Forest avec les données Train & Test



(a) Matrice de confusion du modèle Random Forest (Train)

```
____________Train Random Forest_____________
accuracy : 0.9600494297756291
mean absolute error : 0.0798980414032478
mean squared error : 0.15983482087517045
root mean squared error : 0.3997934727770958

Classification report Training
              precision    recall  f1-score   support

   SYN flood       1.00      1.00      1.00    257714
  http flood       0.89      0.91      0.90    258058
  icmp flood       1.00      1.00      1.00    258382
   slowloris       0.91      0.89      0.90    258000
   udp flood       1.00      1.00      1.00    258566

    accuracy                           0.96   1290720
   macro avg       0.96      0.96      0.96   1290720
weighted avg       0.96      0.96      0.96   1290720
```

(b) Raport De Classification Random Forest (Train)



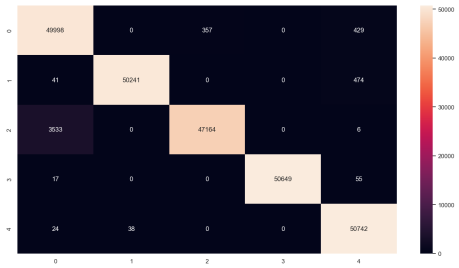
(c) Matrice de confusion du modèle Random Forest (Test)

```
____________Test Random Forest_____________
accuracy : 0.9557270360728896
mean absolute error : 0.08855212594520888
mean squared error : 0.17718482707326144
root mean squared error : 0.4209332810235625

Classification report Test
              precision    recall  f1-score   support

   SYN flood       1.00      1.00      1.00     64966
  http flood       0.88      0.90      0.89     64622
  icmp flood       1.00      1.00      1.00     64298
   slowloris       0.90      0.88      0.89     64680
   udp flood       1.00      1.00      1.00     64114

    accuracy                           0.96    322680
   macro avg       0.96      0.96      0.96    322680
weighted avg       0.96      0.96      0.96    322680
```

(d) Raport De Classification Random Forest (Test)

FIGURE 3.14 – Résultat Du Training & Testing Avec Random Forest

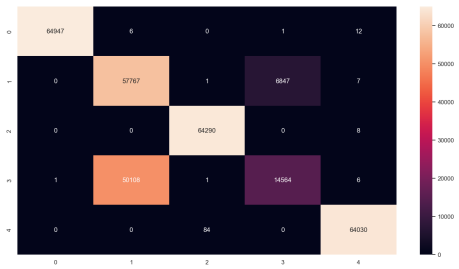
Naive Bayes avec les données Train & Test



(a) Matrice de confusion du modèle Naive Bayes (Train)

```
____________naive bayes____________
____________Train naive bayes____________
accuracy : 0.8235357010040908
mean absolute error : 0.35296966034461386
mean squared error : 0.706209724085781
root mean squared error : 0.8403628456854683

Classification report Training
              precision    recall  f1-score   support

   SYN flood       1.00      1.00      1.00    257714
  http flood       0.54      0.89      0.67    258058
  icmp flood       1.00      1.00      1.00    258382
   slowloris       0.68      0.22      0.34    258000
    udp flood      1.00      1.00      1.00    258566

    accuracy                           0.82   1290720
   macro avg       0.84      0.82      0.80   1290720
weighted avg       0.84      0.82      0.80   1290720
```

(b) Raport De Classification Naive Bayes (Train)

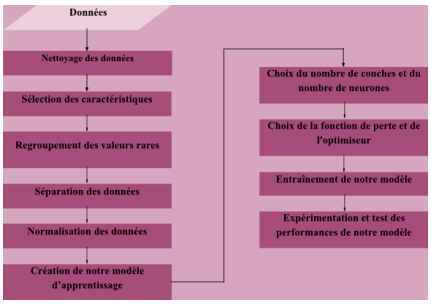


(c) Matrice de confusion du modèle Naive Bayes (Test)

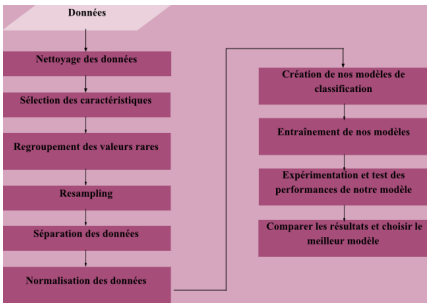
```
____________Test naive bayes____________
accuracy : 0.8231002851121855
mean absolute error : 0.35385831164001486
mean squared error : 0.7080544192388745
root mean squared error : 0.8414596955522435

Classification report Test
              precision    recall  f1-score   support

   SYN flood       1.00      1.00      1.00     64966
  http flood       0.54      0.89      0.67     64622
  icmp flood       1.00      1.00      1.00     64298
   slowloris       0.68      0.23      0.34     64680
    udp flood      1.00      1.00      1.00     64114

    accuracy                           0.82    322680
   macro avg       0.84      0.82      0.80    322680
weighted avg       0.84      0.82      0.80    322680
```

(d) Raport De Classification Naive Bayes (Test)

FIGURE 3.15 – Résultat Du Training & Testing Avec Naive Bayes



(a) Diagramme De La Création Du Modèle De L'autoencodeur



(b) Diagramme De La Création Du Modèle De Classification

FIGURE 3.16 – Diagrammes De La Création Du Modèle De L'autoencodeur Et Classification

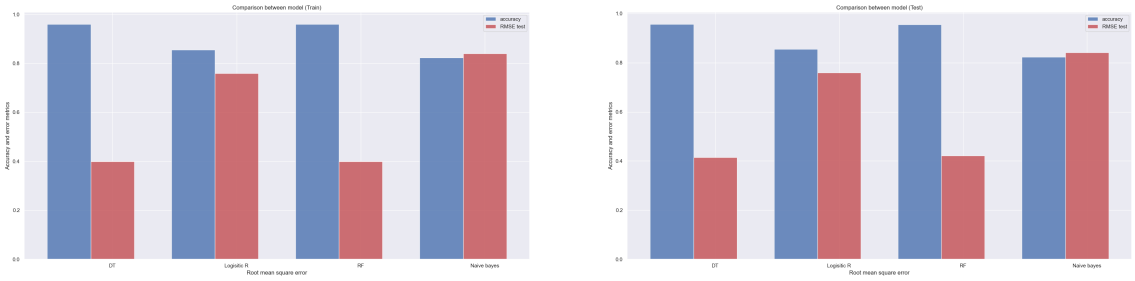
3.7 Discussion des résultas

Dans le tableau suivant, nous allons discuter les résultats d'Accuracy et le coût obtenus des différents algorithmes choisis (Tableau 3.1).

Indicateurs de performance	LR	NB	DTC	RF
Accuracy	0.98	0.98	0.99	0.99
Loss	0.063	0.044	0.030	0.030

TABLE 3.1 – Indicateurs De Performance Des Algorithmes De Classification

Nous avons présenté dans le graphe, illustré par la figure 3.17, le résultat des algorithmes de classification que nous avons utilisé dans notre solution. Nous avons pris les valeurs de l'Accuracy et le RMSE de chaque algorithme pour comparer et choisir le meilleur modèle pour les données train et test.



(a) Comparaison Entre Les Modèles De Classification (Train)

(b) Comparaison Entre Les Modèles De Classification (Test)

FIGURE 3.17 – Comparaison Entre Les Modèles De Classification (Train & Test)

La majorité des algorithmes ont donné des résultats différents. Les modèles Random Forest et Decision Tree Classifier ont donné le moins d'erreurs avec la meilleure valeur d'Accuracy pour les données d'entraînement. Cependant, avec les données de test, on remarque que même si l'accuracy est la même, le RMSE du Decision Tree Classifier est légèrement meilleur que celui du Random Forest. Pour cela, nous constatons que c'est le modèle Decision Tree Classifier qui présent le meilleur résultat.

CHAPITRE 4

CONCLUSION

Les réseaux actuels sont confrontés à une variété croissante de menaces, mettant en évidence le besoin pressant d'améliorer les systèmes de détection d'intrusions. Malgré les avancées technologiques, les IDS actuels présentent des lacunes, telles que les faux positifs et la détection des nouvelles attaques. L'application de l'Intelligence Artificielle offre des perspectives prometteuses pour relever ces défis.

Dans notre étude, centrée sur l'application de l'Apprentissage Profond aux IDS, nous avons exploité les réseaux SDN et entraîné deux modèles distincts. L'un était dédié à la classification, tandis que l'autre utilisait un autoencodeur. Cette approche a renforcé l'importance des politiques de sécurité robustes et a approfondi notre compréhension de l'intégration de l'IA dans ce domaine. Ces travaux constituent une première étape cruciale vers de futures avancées, ouvrant la voie à des innovations significatives dans la sécurisation des réseaux.

Avec les concepts ainsi appris et maîtrisés pendant ces deux mois de stage chez Ericsson, nous aspirons à ce que ce travail serve de support et de source de référence pour les étudiants à venir, afin de les guider dans d'éventuelles améliorations.

BIBLIOGRAPHIE

[Abbas, 2023] Abbas, A. (26 Avril 2023). MANAGEMENT DE LA SÉCURITÉ DES SYSTÈMES D'INFORMATION.

[Afzaal, 2022] Afzaal, M. (2022). An overview of defense techniques against dos attacks. In *2022 9th International Conference on Internet of Things : Systems, Management and Security (IOTSMS)*, pages 1–8.

[AWS, 2023] AWS (2023). Qu'est-ce que la régression logistique ?

[BOUROUH Mouloud, 2017] BOUROUH Mouloud, K. Z. (2016-2017). Détection d'intrusions à base des réseaux de neurones et algorithmes génétiques. pages 8–10.

[chinacablesbuy, 2018] chinacablesbuy (2018). Fibre Optic Cable Solutions.

[Columbia, 2023] Columbia (2023). Artificial Intelligence (AI) vs. Machine Learning.

[datascientest, 2024] datascientest (15 janvier 2024). Apprentissage Non Supervisé : principe et utilisation.

[Dave Bergmann, 2023] Dave Bergmann, C. S. (23 novembre 2023). Qu'est-ce qu'un auto-encodeur ?

[Deluzarache, 2023] Deluzarache, C. (14 Octobre 2023). Deep Learning : qu'est-ce que c'est ?

[Google, 2020] Google (2020). Unsupervised Learning.

[IBM, 2021a] IBM (2021a). Approches courantes de l'apprentissage non supervisé.

[IBM, 2021b] IBM (2021b). Qu'est-ce que l'algorithme de forêt aléatoire (random forest) ?

[IBM, 2021c] IBM (2021c). Qu'est-ce que l'apprentissage supervisé ?

[IBM, 2021d] IBM (2021d). Qu'est-ce qu'un réseau de neurones ?

[IBM, 2021e] IBM (2021e). What are Naïve Bayes classifiers ?

[IBM, 2023b] IBM (2023b). Qu'est-ce que l'intelligence artificielle (IA) ?

[IBM, 2023c] IBM (2023c). Qu'est-ce qu'un système de détection d'intrusion (IDS) ?

[IBM, 2023d] IBM (2023d). Qu'est-ce qu'une attaque par déni de service distribué (DDoS) ?

[IBM, 2023a] IBM (23 novembre 2023a). Qu'est-ce qu'un arbre de décisions ?

[Illy, 2018] Illy, P. (25 Avril 2018). Les systèmes de détection d'intrusion (ids). pages 48–50.

[Kamalov et al., 2021] Kamalov, F., Zgheib, R., Leung, H. H., Al-Gindy, A., and Moussa, S. (2021). Autoencoder-based intrusion detection system. In *2021 International Conference on Engineering and Emerging Technologies (ICEET)*, pages 1–5.

[Kime, 2022] Kime, C. (December 19, 2022). Complete Guide to the Types of DDoS Attacks.

[Loubna Dali, 2015] Loubna Dali, karim abouelmehdi, A. B. D. H. E. E. A. E. F. B. A. (2015). A survey of intrusion detection system. pages 1–6.

[Microsoft, 2023] Microsoft (9 avril 2023). Intelligence artificielle : tout ce qu'il faut savoir.

[Saleh Asadollahi, 2024] Saleh Asadollahi, Bhargavi Goswami, M. S. (February 26,2024). Ryu controller's scalability experiment on software defined networks. pages 1–5.

[Zamboni, 1998] Zamboni, D. (1998). An Architecture for Intrusion Detection using Autonomous Agents.