République Algérienne Démocratique et Populaire
Ministère de l'Enseignement Supérieur et de la Recherche Scientifique
Université Abderrahmane Mira de Béjaïa

Faculté des Sciences Exactes
Département d'Informatique

Mémoire de fin d'études

Option : Réseau et Sécurité

Thème

Détection des attaques en utilisant l'apprentissage automatique

Réalisé par :

Meniche Kousseila Chikh Islam

Soutenu le 01/07/2024 devant le jury composé de :

Présidente	SABRI Salima	M.C.B	U. A/Mira Béjaïa
Examinatrice	BOUADEM Nassima	M.C.B	U. A/Mira Béjaïa
Encadrant	AMROUN Kamal	Professeur	U. A/Mira Béjaïa
Co-Encadrant	EL-SAKAAN Nadim	Doctorant	U. A/Mira Béjaïa

Promotion 2023 – 2024

Dédicaces

" *Je dédie ce mémoire de master aux personnes qui ont été une source constante d'inspiration, de soutien et d'encouragement tout au long de mon parcours académique.*

À mes amis et collègues, pour leur encouragement, leur soutien et leur camaraderie constante. Votre présence a rendu ce parcours plus agréable et mémorable. Je suis reconnaissant pour les souvenirs que nous avons partagés et les liens que nous avons créés.

Enfin, je me dédie ce mémoire, pour ma persévérance, ma détermination et ma résilience. Cette réussite est un témoignage du travail acharné, des sacrifices et de la dévotion que j'ai consacrés à ma croissance académique et personnelle. Je suis fier de moi et reconnaissant pour l'opportunité de poursuivre ma passion. "

KOUSSEILA

" *Ce mémoire de master est dédié aux personnes qui ont influencé mon parcours académique de manière incommensurable.*

À mes parents, qui m'ont inculqué l'amour de l'apprentissage et m'ont enseigné la valeur du travail acharné. Votre soutien indéfectible et votre confiance en mes capacités ont été la force motrice derrière mon succès. Je suis éternellement reconnaissant pour tout ce que vous avez fait pour moi et je m'efforcerai toujours de vous rendre fiers.

Et enfin, à moi-même, pour les longues heures, le dévouement et le travail acharné investis dans la réalisation de ce parcours. Cette réussite est un rappel de mes capacités, et je suis fier de ce que j'ai accompli. Avec amour et gratitude. "

ISLAM

Remerciements

Après avoir rendu grâce à Dieu Tout-Puissant et Bienveillant, nous tenons à exprimer notre sincère gratitude à tous ceux qui ont participé à la réalisation de cette mémoire.

Nous souhaitons également remercier ***Dr. Nadim El-Sakaan****, pour son soutien indéfectible et ses conseils tout au long de cette recherche. Son expertise et ses perspectives ont été précieuses pour la réalisation de ce mémoire.*

Enfin, nous tenons à remercier les membres du jury pour leur intérêt porté à notre recherche.

Résumé

Les réseaux informatiques sont l'épine dorsale des communications et de l'échange d'informations, s'étendant pour inclure une large gamme d'appareils bénéfiques à de nombreux secteurs d'activité. Cependant, cette expansion crée de nouvelles vulnérabilités exploitées par des acteurs malveillants. Ces derniers disposent d'un éventail d'attaques qui peuvent causer des dommages financiers et réputationnels pour les individus, les entreprises et les gouvernements.

Les attaques botnet et DDoS sont parmi les plus sophistiquées et les dangereuses, elles peuvent paralyser les serveurs et systèmes, les rendant inopérants. En raison de leur complexité et de leur évolutivité, ces attaques sont difficiles à détecter avec les mesures de sécurité traditionnelles. Cependant, l'utilisation de systèmes de détection d'intrusion basés sur l'apprentissage automatique (Machine Learning - ML) et l'apprentissage profond (Deep Learning - DL) offre une solution potentielle à ce problème.

Les algorithmes de ML et DL peuvent s'avérer très efficaces pour la détection d'intrusion dans les réseaux informatiques. Leur entraînement requiert une large quantité de données, et leur efficacité dépend de la qualité de ces datasets. Pour notre projet, nous avons utilisé le dataset CSE-CIC-IDS2018, qui inclut une bonne variété d'attaques et de données de trafic réseau.

Le but de ce projet est la conception d'un système de détection d'intrusion (IDS) efficace basé sur le ML. Au terme de notre travail, nous avons utilisé un dataset de six différentes attaques réseaux et avons conçu plusieurs modèles de ML pour détecter ces types d'attaques. Ces derniers ont été évalués selon certaines critères pour ensuite sélectionner le modèle avec les résultats les plus élevés.

Mots clés : Systèmes de détection d'intrusion, Apprentissage automatique, DDoS, Botnet, CSE-CIC-IDS2018, Deep learning, Machine Learning.

Abstract

Computer networks are the backbone of communication and information exchange, extending to include a wide range of devices beneficial to many sectors. However, this expansion creates new vulnerabilities exploited by malicious actors. These actors have a variety of attacks at their disposal that can cause financial and reputational damage to individuals, businesses, and governments.

Botnet and DDoS attacks are among the most sophisticated and dangerous, capable of paralyzing servers and systems, rendering them inoperative. Due to their complexity and scalability, these attacks are difficult to detect with traditional security measures. However, the use of intrusion detection systems based on machine learning (ML) and deep learning (DL) offers a potential solution to this problem.

ML and DL algorithms can be highly effective for intrusion detection in computer networks. Their training requires a large amount of data, and their effectiveness depends on the quality of these datasets. For our project, we used the CSE-CIC-IDS2018 dataset, which includes a good variety of attacks and network traffic data.

The aim of this project is to design an effective intrusion detection system (IDS) based on ML. By the end of our work, we used a dataset of six different network attacks and designed several ML models to detect these types of attacks. These models were evaluated based on specific criteria to then select the model with the highest results.

Keywords : Intrusion detection systems, Machine learning, DDoS, Botnet, CSE-CIC-IDS2018, Deep learning, Machine Learning.

Table des matières

Table des figures

Liste des tableaux

Liste des acronymes

DDos Distributed denial-of-service
Dos Denial-of-service
DL Deep Learning
DT Decision Tree
FN False Negatif
FP False Positif
HIDS Host Intrusion Detection Systems
IA Intelligence Artificielle
IDS Intrusion detection System
KNN k-Nearest Neighbours
ML Machine Learning
MLP Multi layer perceptron
NaN Not a Number
NIDS Network Intrusion Detection Systems
OSI Open Systems Interconnection
RFC Random Forest Classifier
GNN Graphical Neural Networks
RNN Recurrent Neural Networks
SVM Support Vector Machine
TN True Negative
TP True Positive
VM Virtual Machine

Introduction générale

La cybersécurité est devenue un enjeu crucial dans le monde contemporain, où les systèmes informatiques sont constamment menacés par des cyberattaques sophistiquées. Avec l'expansion continue de l'Internet et l'interconnexion des dispositifs, les réseaux informatiques sont plus vulnérables que jamais aux attaques malveillantes. Les cyberattaques peuvent prendre diverses formes, y compris les logiciels malveillants (malware), les ransomwares, le phishing, les attaques par déni de service (DDoS) et les exploits de vulnérabilités zero-day.

Les conséquences de ces attaques peuvent être dévastatrices. Les entreprises peuvent subir des pertes financières considérables, des interruptions de service prolongées, et des dommages à leur réputation. Par exemple, les ransomwares peuvent paralyser les opérations d'une entreprise en chiffrant ses données critiques, exigeant une rançon pour les déverrouiller. En 2017, l'attaque WannaCry a infecté des centaines de milliers de systèmes dans plus de 150 pays, entraînant des perturbations majeures dans des secteurs aussi divers que la santé, les transports et les services financiers.

De plus, les attaques de phishing et les violations de données peuvent entraîner la divulgation d'informations sensibles, telles que des données personnelles ou financières, exposant les individus et les entreprises à des risques de fraude et de vol d'identité. La violation des données d'Equifax en 2017, par exemple, a exposé les informations personnelles de 147 millions d'Américains, entraînant des conséquences juridiques et financières importantes pour l'entreprise.

Face à ces menaces, les organisations investissent de plus en plus dans des mesures de sécurité sophistiquées pour protéger leurs systèmes. Cependant, les méthodes traditionnelles de défense, telles que les pare-feu et les antivirus, ne sont souvent pas suffisantes pour contrer les attaques les plus avancées. C'est ici qu'interviennent les systèmes de détection d'intrusions (IDS) comme Snort. Bien que Snort se distingue par sa flexibilité et ses divers modes opératoires, son efficacité repose largement sur des règles prédéfinies lors de sa configuration. En l'absence de règles adéquates face à une intrusion, une attaque peut passer inaperçue, exposant ainsi le système à de potentiels dommages.

Cette limitation des systèmes IDS basés sur des règles statiques souligne la nécessité de méthodes plus avancées et adaptatives. Les systèmes de détection d'intrusions basés sur l'apprentissage automatique émergent comme une alternative prometteuse, capable de détecter des anomalies et de pallier les failles des méthodes traditionnelles. Ces systèmes utilisent des algorithmes pour analyser les comportements et les tendances dans les réseaux, permettant de repérer des activités suspectes qui ne correspondent pas aux modèles connus de comportement normal. Cette capacité d'apprentissage et d'adaptation continue permet de mieux anticiper et réagir aux nouvelles menaces, renforçant ainsi la sécurité globale des systèmes informatiques.

Ainsi, dans ce mémoire, nous explorerons cette transition vers des techniques innovantes à travers une structure en quatre chapitres détaillés :

- **Chapitre 1 :** Présentation de la sécurité informatique, des cyberattaques et des contre-mesures telles que les pare-feu, les VPN et les IDS, ainsi qu'une introduction aux algorithmes d'apprentissage automatique.
- **Chapitre 2 :** Présentation de l'organisme d'accueil et exposé de la problématique concernant les limites des IDS existants.
- **Chapitre 3 :** Description de la mise en place des méthodes classiques de détection comme Snort, et conception d'un IDS utilisant une approche basée sur le machine learning.
- **Chapitre 4 :** Entraînement, évaluation des modèles ML et présentation des résultats des testes.

L'objectif de ce mémoire est de démontrer que les IDS basés sur le machine learning offrent une solution prometteuse pour substituer les IDS à base des signatures pour améliorer la détection des cyberattaques, renforçant ainsi la sécurité des systèmes informatiques face aux menaces.

Chapitre 1

Sécurité Informatique et Apprentissage Automatique

1.1 Introduction

La croissance rapide et l'adoption généralisée des appareils intelligents ont considérablement élargi la taille et la diversité de nos réseaux informatiques. Cette expansion a apporté des avantages significatifs à divers secteurs, mais elle a également introduit de nouvelles vulnérabilités dans nos systèmes, parmi lesquelles les attaques de réseaux botnet se distinguent comme étant particulièrement courantes et dangereuses.
Dans ce chapitre, nous examinerons plusieurs aspects fondamentaux de la sécurité des réseaux informatiques pertinents pour les objectifs de notre projet. Nous commencerons par définir les réseaux informatiques, puis nous étudierons les cyberattaques ainsi que les différents mécanismes disponibles pour les contrer. Enfin, nous aborderons également le rôle de l'apprentissage automatique dans la modélisation des systèmes de défense pour renforcer la sécurité des réseaux informatiques.

1.2 Sécurité informatique

La triade CIA, qui signifie Confidentialité, Intégrité et Disponibilité, est un concept fondamental en matière de sécurité [21] (Figure 1.1). Elle constitue un ensemble de principes directeurs et d'objectifs pour les organisations et les individus afin de protéger les informations contre l'accès, la modification ou la perte non autorisés. Pour garantir une véritable sécurité, les trois éléments de la triade CIA doivent être présents simultanément.

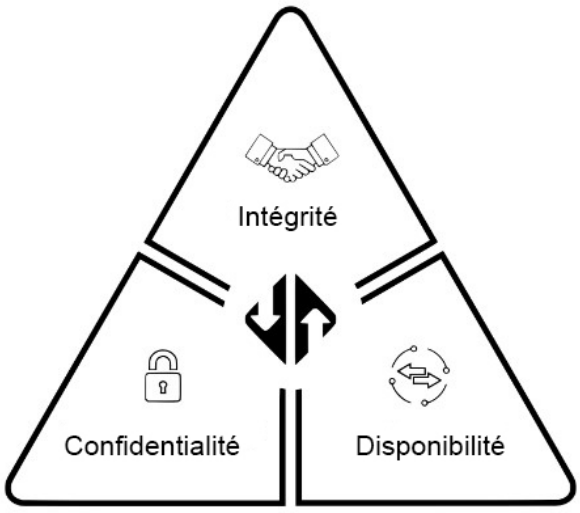

FIGURE 1.1 – La triade CIA.

1.2.1 Confidentialité

La confidentialité garantit que l'accès aux données est restreint uniquement aux individus ou entités autorisés. Plus les informations sont sensibles, plus les mesures de sécurité doivent être strictes. Certaines mesures pour assurer la confidentialité incluent le chiffrement, les mots de passe, l'authentification à deux facteurs, l'identification biométrique et les jetons de sécurité. Ces mesures aident à protéger les informations sensibles contre l'accès non autorisé et à prévenir les violations de données. Il est important de mettre en œuvre les mesures de confidentialité appropriées en fonction du niveau de sensibilité des informations protégées [14].

1.2.2 Intégrité

L'intégrité implique de maintenir l'exactitude et l'intégralité des données. Cela se fait en protégeant les données contre toute modification non autorisée, les modifications accidentelles ou les événements non causés par l'homme tels que les crashs de serveur. Pour garantir l'intégrité des données, plusieurs mesures peuvent être mises en œuvre, telles que le chiffrement, le hachage, les contrôles d'accès utilisateur, les sommes de contrôle, le contrôle des versions et les sauvegardes. Ces mesures aident à prévenir les modifications non autorisées et à garantir l'exactitude des données. Il est important de mettre en œuvre les mesures appropriées en fonction de la sensibilité des informations protégées [38].

1.2.3 Disponibilité

La disponibilité garantit que les utilisateurs autorisés ont accès aux données quand ils en ont besoin. Cela implique la mise en œuvre de mesures telles que la redondance, l'équilibrage de charge, la planification de la reprise après sinistre, la maintenance régulière et la surveillance pour prévenir toute interruption ou temps d'arrêt. En assurant la disponibilité des services réseau, les organisations et les individus peuvent éviter la perte de productivité, la perte de revenus et d'autres conséquences négatives résultant d'une interruption du réseau [38].

En dehors de la triade CIA, il existe un autre ensemble de mesures qui devraient être mises en place pour assurer la sécurité des informations. Ces mesures sont connues sous le nom d'authentification, d'autorisation et de comptabilité.

1.3 Cyberattaques

Une cyberattaque désigne un effort intentionnel visant à voler, exposer, modifier, désactiver ou détruire des données, des applications ou d'autres actifs par le biais d'un accès non autorisé à un réseau, un système informatique ou un appareil numérique.
Les cybercriminels utilisent diverses techniques telles que le phishing, les logiciels malveillants, les attaques par déni de service (DDoS) et l'ingénierie sociale pour accéder illégalement à des données sensibles, perturber les opérations commerciales ou causer des dommages financiers. Comprendre les types de cyberattaques, les motivations des attaquants et les mesures de prévention appropriées est essentiel pour renforcer la sécurité des systèmes informatiques et se prémunir contre les menaces en constante évolution dans le cyberespace [28].
Dans ce qui suit, nous allons explorer quelques attaques qui sont couramment utilisées par les cybercriminels et leurs objectifs.

1.3.1 Botnet

Les botnets sont des réseaux d'appareils informatiques piratés, utilisés pour mener diverses escroqueries et cyberattaques. « Botnet » est une contraction des termes « robot » et « network » (réseau). La création d'un botnet est généralement l'étape d'infiltration d'un système à plusieurs niveaux. Les bots servent d'outil pour automatiser les attaques de masse, comme le vol de données, les plantages de serveur et la distribution de programmes malveillants [34].

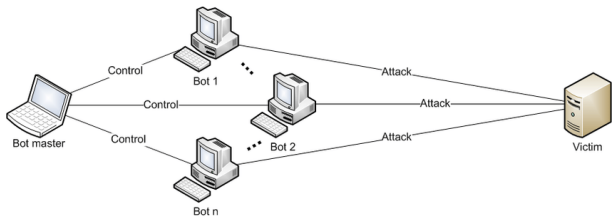

FIGURE 1.2 – Botnet

De ce qui précède et en se basant sur d'autres sources, on peut décrire les botnets et leur fonctionnement comme un réseau d'appareils infectés par des logiciels malveillants qui peut être entièrement contrôlé à partir d'un centre de commande et de contrôle appelé serveur C&C qui est géré par un acteur malveillant (Botmaster). Le réseau lui-même, qui peut être constitué de milliers, voire de centaines de milliers d'appareils, qui se composent d'ordinateurs personnels, serveurs, appareils IoT, etc, est utilisé pour propager les logiciels malveillants et étendre le réseau. Lorsqu'un appareil vulnérable est détecté par les bots, ils l'infectent et le signalent à leur centre de commande et de contrôle (serveur C&C). Ils

sont ensuite chargés de trouver d'autres appareils similaires à infecter, et le processus se poursuit ainsi.

1.3.2 Deni de Service (DOS)

Un DoS (Déni de Service) est un type de cyberattaque qui vise à perturber le fonctionnement normal d'un ordinateur ou d'un autre dispositif en le submergeant de trafic, le rendant ainsi indisponible pour ses utilisateurs prévus. De telles attaques peuvent être effectuées à l'aide d'un seul ordinateur et impliquent généralement de noyer la machine ciblée de requêtes jusqu'à ce que le trafic normal ne puisse plus être traité. Le principal objectif d'une attaque DoS est d'épuiser les ressources de l'appareil ciblé, entraînant ainsi un déni de service pour les demandes légitimes. Ces attaques peuvent causer des perturbations significatives pour les entreprises et les organisations, et diverses mesures peuvent être prises pour prévenir et atténuer leur impact [10]. La figure 1.3 illustre l'attaque DoS.

FIGURE 1.3 – Déni de service.

1.3.2.1 Hulk

Hulk est un outil d'attaque DoS qui envoie un grand nombre de requêtes HTTP à un serveur web afin de le submerger et de le rendre non réactif. Cette attaque tire son nom du personnage de Marvel Comics, car l'outil est conçu pour briser les défenses des serveurs web [5].

1.3.2.2 GoldenEye

GoldenEye est un autre outil d'attaque DoS qui envoie un grand nombre de requêtes HTTP à un serveur web, mais utilise également des techniques de cryptage et d'obfuscation pour échapper à la détection par les systèmes de sécurité. Cette attaque tire son nom du méchant fictif de James Bond, car elle est conçue pour être sophistiquée et difficile à arrêter [25].
Slowloris est un outil d'attaque DoS qui fonctionne en ouvrant plusieurs connexions à un serveur web et en envoyant des requêtes HTTP partielles sans jamais les compléter. Cela mobilise les ressources du serveur et le rend non réactif aux requêtes légitimes.

Cette attaque est nommée d'après le loris paresseux, un type de primate qui se déplace très lentement, car l'attaque est conçue pour épuiser lentement les ressources du serveur ciblé [24].

1.3.3 Distributed Denial of Service (DDOS)

Une attaque DDoS est une tentative malveillante de perturber le trafic normal d'un serveur, d'un service ou d'un réseau ciblé en le submergeant d'un flot de trafic Internet [3]. Les attaques DDoS sont menées avec des réseaux d'ordinateurs et d'autres dispositifs infectés, connus sous le nom de botnets, qui sont contrôlés à distance par l'attaquant. L'attaquant envoie des instructions à distance à chaque bot pour diriger l'attaque, faisant en sorte que chaque bot envoie des requêtes à l'adresse IP de la cible (voir Figure 1.3). Cela peut potentiellement submerger le serveur ou le réseau, entraînant un déni de service pour le trafic normal. Nous énumérons deux des types d'attaques DDoS les plus couramment utilisés.

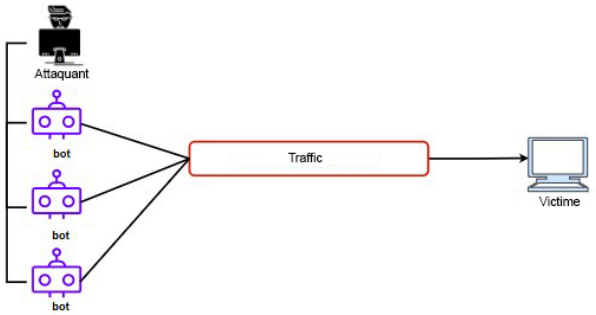

FIGURE 1.4 – Déni de service distribué.

1.3.3.1 HOIC

High Orbit Ion Cannon, il s'agit d'un outil d'attaque DDoS qui utilise un grand nombre de requêtes HTTP GET ou POST pour submerger un serveur ou un réseau cible. HOIC est connu pour sa capacité à mener des attaques hautement coordonnées et concentrées, ce qui peut générer de grandes quantités de trafic et rendre difficile la détection de la source de l'attaque [8].

1.3.3.2 LOIC :

Low Orbit Ion Cannon, c'est également un outil d'attaque DDoS qui envoie un flot de trafic à un serveur ou un réseau cible. Contrairement à HOIC, LOIC peut utiliser à la fois des attaques par inondation HTTP et UDP. Les attaques par inondation HTTP sont

similaires à celles utilisées par HOIC, tandis que les attaques par inondation UDP envoient un grand nombre de paquets UDP à la cible, consommant ainsi les ressources du réseau et rendant la cible non réactive [9].

La différence entre ces deux attaques réside dans leur niveau de sophistication. HOIC est connu pour sa capacité à mener des attaques plus avancées et coordonnées, tandis que LOIC est considéré comme un outil plus basique et direct.

1.3.4 Brute force

Une attaque par force brute est une méthode d'essai-erreur utilisée pour obtenir un accès non autorisé à un système ou une application en devinant à plusieurs reprises des noms d'utilisateur et des mots de passe potentiels jusqu'à ce que la combinaison correcte soit trouvée [47]. L'attaquant peut utiliser un logiciel automatisé pour générer un grand nombre de tentatives successives en peu de temps (voir figure 1.5).

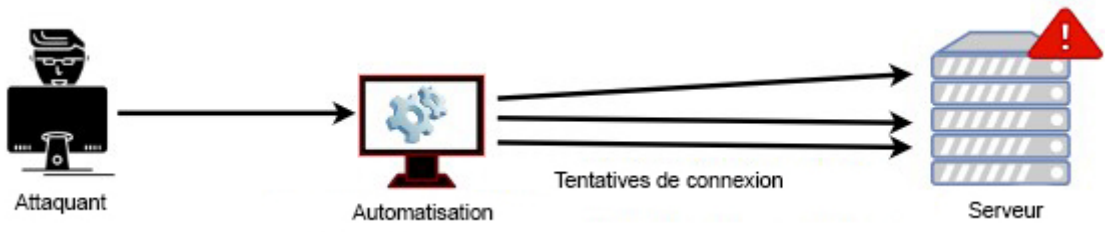

FIGURE 1.5 – Attaque force brute.

1.3.4.1 FTP Bruteforce :

Le protocole de transfert de fichiers (FTP) est un protocole réseau utilisé pour transférer des fichiers. Il utilise un modèle client-serveur où les utilisateurs peuvent se connecter à un serveur à l'aide d'un client FTP [51]. L'authentification se fait avec un nom d'utilisateur et un mot de passe, typiquement transmis en texte clair, mais peut également prendre en charge des connexions anonymes si elles sont disponibles.

1.3.4.2 SSH Bruteforce :

Ce type d'attaque cible le protocole Secure Shell (SSH), utilisé pour accéder et gérer à distance des serveurs [1]. L'attaquant essaie différentes combinaisons de noms d'utilisateur et de mots de passe jusqu'à ce qu'il réussisse à se connecter au serveur SSH. Une fois qu'il a accès, il peut effectuer diverses activités malveillantes telles que le vol de données ou l'installation de logiciels malveillants.

1.3.4.3 Web Bruteforce :

Ce type d'attaque cible les systèmes de connexion basés sur le web, tels que ceux utilisés pour la banque en ligne ou les comptes de messagerie électronique. L'attaquant utilise des outils automatisés pour essayer différentes combinaisons de noms d'utilisateur et de mots de passe jusqu'à ce qu'il réussisse à se connecter au système. Une fois qu'il a accès, il peut voler des informations sensibles ou effectuer d'autres activités malveillantes [41].

1.3.5 Infiltration

L'infiltration est l'acte de gagner un accès non autorisé à un système ou à un réseau, généralement à des fins malveillantes. Les attaquants peuvent utiliser diverses méthodes pour infiltrer un réseau, notamment l'exploitation de vulnérabilités logicielles ou matérielles, l'utilisation de techniques d'ingénierie sociale pour tromper les utilisateurs afin qu'ils divulguent des informations sensibles ou exécutent du code malveillant, ou encore la violation physique des contrôles de sécurité.

Une forme d'infiltration implique l'exploitation de vulnérabilités à l'intérieur du réseau lui-même. Dans ce scénario, les attaquants peuvent utiliser des techniques telles que l'envoi d'un fichier malveillant par e-mail à une victime et exploiter une vulnérabilité de l'application. Une fois l'attaque réussie, une porte dérobée est exécutée sur l'ordinateur de la victime, permettant à l'attaquant de scanner le réseau interne à la recherche d'autres systèmes vulnérables et de les exploiter si possible [35].

1.4 Mécanismes de défense contre les cyberattaques

La défense contre les cyberattaques est cruciale pour protéger l'intégrité et la confidentialité des données. Cette discipline comprend diverses techniques et stratégies visant à détecter, prévenir et contrer les menaces informatiques. Que ce soit pour des entreprises, des gouvernements, des institutions financières ou des utilisateurs individuels, la sécurité informatique est une priorité, car ces attaques peuvent causer d'énormes dommages financiers et nuire à la réputation.
Pour répondre à ces défis, des solutions telles que les systèmes de détection d'intrusion, la gestion des vulnérabilités et l'authentification multifactorielle sont essentielles. Une approche proactive et adaptable, combinée à une sensibilisation continue des utilisateurs, est nécessaire pour faire face à l'évolution rapide des menaces.
Dans la suite, nous explorerons les outils de défense contre les cyberattaques, en mettant en lumière leur fonctionnement et leur importance dans une stratégie de sécurité globale.

1.4.1 Pare-feu

Un pare-feu est un dispositif ou un logiciel conçu pour protéger un réseau informatique en contrôlant et en filtrant le trafic entrant et sortant, il agit comme la première ligne de défense contre les attaques. Son rôle principal est de servir de barrière de sécurité entre un réseau privé et des réseaux externes, tels qu'Internet, afin de prévenir les attaques malveillantes, les intrusions et les fuites de données [20].

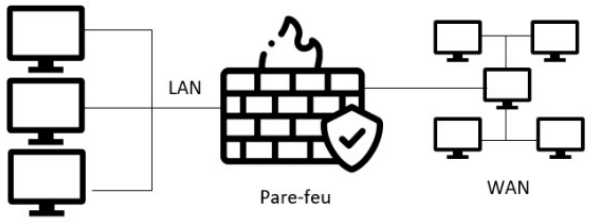

FIGURE 1.6 – Pare-feu.

En plus de filtrer le trafic, certains peuvent offrir d'autres fonctionnalités telles que la translation d'adresses réseau (NAT), la prévention des intrusions (IPS), la détection des attaques par déni de service (DDoS), la gestion des applications et des utilisateurs, ainsi que des fonctionnalités avancées de sécurité telles que les VPN (Virtual Private Network) pour sécuriser les communications à distance.

Le pare-feu peut être déployé sous forme matérielle ou logicielle (applications installées sur des serveurs), offrant ainsi une protection adaptée aux besoins spécifiques de chaque environnement réseau. Pour sécuriser un réseau le pare-feu peut :

— Empêcher des intrus d'accéder au réseau interne.
— Empêcher les utilisateurs de réseau local de sortir n'importe où.
— Filtrer les entrées et les sorties (adresse IP et port).

1.4.1.1 Fonctionnement :

Le pare-feu fonctionne sur les couches suivantes du modèle OSI :

— **Couche réseau :** Pour le filtrage des adresses IPs.
— **Couche transport :** Pour filtrer les ports.
— **Couche application :** Pour filtrer certains protocoles de cette couche.

La réalisation de ces filtrages implique qu'un pare-feu soit configuré en respectant certaines règles prédéfinies, on parle alors des politiques de sécurité.

1.4.1.2 Politique de sécurité :

C'est la première phase avant de créer des règles de filtrage sur le pare-feu. Elle se base sur les besoins des utilisateurs de réseau en termes de connexions à Internet et par la suite transformer ces besoins en règles tout en assurant non seulement la sécurité de réseau mais aussi son bon fonctionnement. Deux politiques sont possibles :

1. Autoriser tout le trafic et bloquer les services dangereux.
2. Bloquer tout le trafic et autoriser que les services nécessaires au bon fonctionnement de l'entreprise.

La politique bloquer tout est souvent la plus utilisée.

1.4.1.3 Les types de pare-feu

Il existe plusieurs types de pare-feu allons des pare-feu sans état (Stateless), avec état (Stateful) jusqu'aux pare-feu applicatifs (WAF).

— **Les pare-feux sans état (Stateless packet inspection firewall) :** C'est le plus ancien pare-feu, il agit au niveau de la couche réseau et la couche transport. De ce fait ses règles se basent sur les adresses IPs source et destination, ainsi que les numéros de ports source et destination. Tout échange avec des ports ou adresses IPs non autorisés sera bloqué [6].

— **Les pare-feux avec état (Stateful packet inspection firewall) :** Ce type de pare-feu vérifie que chaque paquet de connexion est bien la suite du précédent paquet ou la réponse d'un paquet dans l'autre sens (conformité des paquets). Il peut alors prendre des décisions de filtrage en fonction des informations récoltées lors des connexions précédentes (appartenant à la même connexion) en consultant le tableau des états, et donc il ne se base pas uniquement sur les règles prédéfinies par l'administrateur. Cette façon de procéder lui permet de protéger le réseau de certaines attaques DOS [22].

— **Les pare-feux applicatifs (WAF) :** Ce type de pare-feu permet de filtrer les communications application par application. Les requêtes sont vérifiées par un processus dédié. Chaque requête doit être conforme aux spécifications du protocole concerné par la requête. Si par exemple nous avons une requête FTP, cette dernière sera filtrée par un processus proxy FTP et doit être conforme aux spécifications du protocoles FTP. Dans ce cas le pare-feu va pouvoir filtrer le protocole FTP et non pas le port FTP (21). Ce type de pare-feu assure plus de sécurité que les autres pare-feux [40].

1.4.2 Les Réseaux Privés Virtuels (VPN)

Les Réseaux Privés Virtuels (VPN) fournissent un accès distant sécurisé à un réseau en créant un tunnel chiffré sur Internet (voir Figure 1.7). Ils sont couramment utilisés par les travailleurs à distance qui ont besoin d'accéder aux ressources de l'entreprise depuis l'extérieur du bureau [7].

FIGURE 1.7 – VPN.

1.4.2.1 Accés distant

Un VPN d'accès distant connecte de manière sécurisée un appareil en dehors du bureau d'entreprise à un réseau privé via Internet. Ce type de VPN est couramment utilisé par les employés qui ont besoin d'accéder aux ressources de l'entreprise depuis l'extérieur du bureau, par exemple depuis leur domicile ou lors de leurs déplacements. La technologie VPN s'est développée pour permettre la vérification de la sécurité des périphériques afin de s'assurer qu'ils répondent à certaines exigences de sécurité avant d'être autorisés à accéder au réseau privé. Les VPN d'accès distant offrent un moyen sécurisé aux employés d'accéder aux ressources de l'entreprise tout en garantissant la confidentialité, l'intégrité et la disponibilité des données sensibles sur le réseau privé.

1.4.2.2 Site-à-Site

Un VPN site-à-site est un type de VPN qui connecte le bureau principal d'une entreprise à ses succursales via Internet, créant ainsi une connexion sécurisée entre deux ou plusieurs réseaux. Ce type de VPN est couramment utilisé pour la connectivité des succursales, la migration vers le cloud et la reprise après sinistre. Des équipements dédiés sont utilisés pour établir et maintenir la connexion, rendant impraticable les connexions réseau directes entre ces bureaux. Les VPN site-à-site sont utilisés lorsque la distance rend difficile la mise en place de connexions réseau directes entre les bureaux. On les appelle souvent accès réseau à réseau. Les VPN site-à-site offrent une manière sécurisée aux organisations de connecter plusieurs réseaux, y compris deux réseaux d'entreprise ou un réseau d'entreprise et un fournisseur de services cloud, tout en garantissant la confidentialité, l'intégrité et la disponibilité des données sensibles.

1.4.3 Chiffrement

Le chiffrement est une technique efficace pour protéger la confidentialité et l'intégrité des données en convertissant le texte en clair en une forme codée, rendant ainsi difficile l'accès aux informations par des parties non autorisées. Cette méthode de sécurité peut être utilisée dans divers scénarios tels que la communication sécurisée par e-mail et le stockage de données sensibles. Cependant, la mise en œuvre de chiffrement peut avoir un impact sur les performances des systèmes et peut nécessiter des compétences avancées en gestion en raison de sa complexité. Néanmoins, les avantages de chiffrement en font un outil puissant pour sécuriser les informations sensibles. Il existe plusieurs types chiffrement, notamment : Chiffrement symétrique, asymétrique et hachage [11].

1.4.3.1 Chiffrement symétrique et asymétrique

Le chiffrement symétrique, également appelée chiffrement à clé partagée, utilise une seule clé pour à la fois chiffrer et déchiffrer les données. Cela la rend rapide et efficace, ce qui en fait un choix idéal pour traiter de grandes quantités de données. Cependant, la sécurité des données chiffrées dépend de la conservation secrète de la clé [11].

Le chiffrement asymétrique, également connue sous le nom chiffrement à clé publique, utilise deux clés différentes : une pour le chiffrement et une pour le déchiffrement. La clé publique est largement distribuée, tandis que la clé privée est gardée secrète. Bien que le chiffrement asymétrique soit plus lente que le chiffrement symétrique, elle offre une meilleure sécurité car la clé privée est maintenue secrète, rendant difficile pour les attaquants d'accéder aux données chiffrées [11].

1.4.3.2 Hachage

Le hachage est un processus de chiffrement à sens unique qui prend du texte en clair et produit une chaîne de caractères de longueur fixe, appelée hachage. Le même texte en clair produira toujours le même hachage, mais il est pratiquement impossible d'inverser le processus et de récupérer le texte en clair à partir du hachage. Il existe plusieurs méthodes courantes de hachage telles que MD5, SHA, bcrypt et scrypt [16].

1.4.4 Sytème de détection d'intrusions (IDS)

Intrusion Detection System (IDS) est un programme ou un appareil qui surveille le trafic réseau pour détecter les signes d'activité suspects ou les violations des règles de sécurité

réseau. Les IDS peuvent être basés sur le réseau ou sur l'hôte et peuvent détecter les failles de sécurité potentielles. Ils peuvent également alerter les administrateurs du système en cas d'incidents de sécurité potentiels, ce qui leur permet de prendre des mesures pour tenter de limiter leurs effets [50]. Un IDS est basé sur trois aspects fonctionnels, à savoir :

1. Un moteur d'analyse qui trouve des signes d'intrusion.
2. Une source d'information qui fournit un flux d'enregistrements d'événements.
3. Un composant de réponse qui génère des réactions basées sur les résultats du moteur d'analyse.

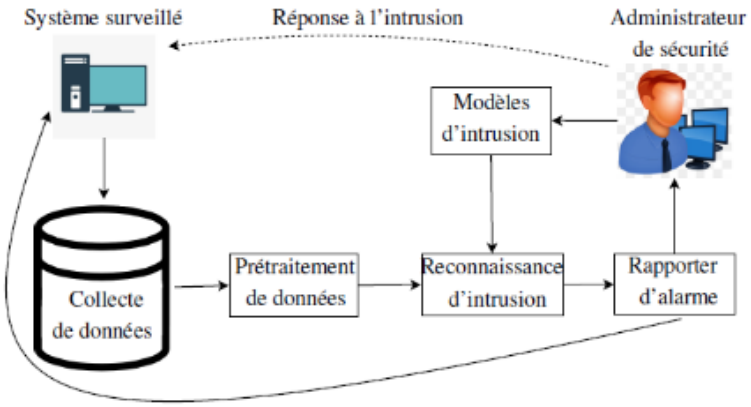

FIGURE 1.8 – Fonctionnement d'un IDS

La Figure 1.8 représente le processus de détection d'intrusion. Pour identifier une intrusion, un IDS effectue les tâches suivantes : (a) collecte des données, (b) prétraitement des données, (c) reconnaissance de l'intrusion, et (d) mise en œuvre de mesures correctives. Les données sont collectées à partir d'une ou plusieurs sources de données, y compris les pistes d'audit, le trafic réseau, la trace des appels système, etc. Un prétraitement est effectué sur les données collectées et seront transférées dans un format compréhensible par le composant de détection. Ce dernier est utilisé pour caractériser le comportement intrusif en utilisant plusieurs techniques et algorithmes. Enfin, le composant de réponse signale l'intrusion et éventuellement les informations temporelles correspondantes.

1.4.4.1 Taxonomie des IDS

Les IDS peuvent être classés en fonction de l'emplacement de déploiement et de la méthode de détection (voir la figure 1.9). En fonction de l'emplacement du module IDS dans le réseau, nous pouvons distinguer les IDS en trois classes : les IDS basés sur l'hôte, les IDS basés sur le réseau, et les IDS hybrides.
En fonction de la méthode de détection, les IDS peuvent être classés selon ces catégories : les IDS basés sur la détection des abus, les IDS basés la détection d'anomalie [2].

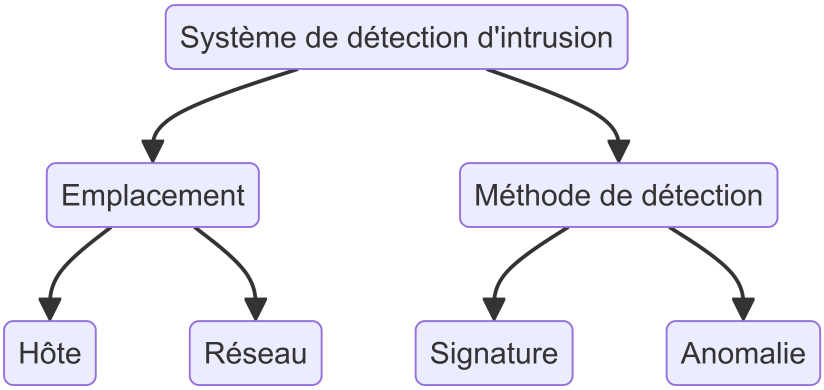

FIGURE 1.9 – Taxonomie des IDS

1.4.4.2 IDS basé sur les méthodes de détection

Dans le domaine de la détection d'intrusions, notamment dans les réseaux, les IDS reposent sur diverses méthodes d'analyse des attaques. Ces méthodes sont regroupées en deux catégories principales : la détection basée sur la signature et la détection basée sur les anomalies.
Dans ce qui suit, nous examinerons en détail ces différentes approches, en mettant en avant leurs avantages et leurs limites respectifs.

1. **Détection par signature :**
   Les intrusions sont repérées en analysant les attaques déjà répertoriées, en construisant ainsi une base de données contenant les signatures de ces attaques préalablement détectées. Cette méthode permet d'anticiper les intrusions et de les identifier facilement, avec un risque d'échec minimal.
   Cette base de données est couplée à un système d'alerte qui se déclenche dès qu'une correspondance avec les signatures enregistrées est repérée dans le trafic réseau. Bien adaptée à la détection des attaques connues, cette approche se révèle toutefois inefficace face à des formes d'attaques inédites ou inconnues. Par conséquent, la mise à jour régulière de la base de données des signatures pourrait être envisagée comme une solution, mais cette tâche est chronophage et coûteuse en ressources.
2. **Détection d'anomalie :**
   Contrairement à la détection par signatue, la détection des anomalies se concentre sur l'établissement de profils d'activité normale pour le système. Son objectif est d'identifier le comportement typique du trafic réseau considéré comme normal, afin de repérer les écarts de comportement, qualifiés d'anomalies. Cette approche permet ainsi de détecter de nouveaux types d'attaques en analysant les caractéristiques du trafic réseau, comblant ainsi une lacune majeure de la détection par signature.

1.4.4.3 IDS basé sur l'emplacement

La collecte de données représente une étape cruciale dans la conception des IDS, influençant l'ensemble du processus de conception et de déploiement ainsi que le résultat final de la détection.
En règle générale, les attaques ne ciblent pas seulement un seul ordinateur, mais souvent un ensemble d'ordinateurs. Par conséquent, certaines intrusions peuvent se manifester par des comportements anormaux au niveau du réseau, tandis que d'autres peuvent présenter des anomalies au niveau des hôtes individuels. Pour garantir une couverture complète des différentes formes d'intrusions, il est nécessaire de surveiller chaque point d'entrée potentiel. Les sources de données peuvent inclure des journaux d'audit, des appels système ou des paquets réseau.
En fonction de ces considérations et de l'emplacement du module IDS, on distingue généralement trois classes d'IDS : les IDS basés sur l'hôte, les IDS basés sur le réseau et les IDS hybrides.

1. **IDS basé sur le réseau (NIDS) :**
   Au fur et à mesure que les environnements informatiques ont évolué des systèmes centralisés vers des réseaux de postes de travail, les recherches sur la détection des intrusions se sont progressivement concentrées sur les attaques ciblant les réseaux. Contrairement aux attaques ciblant les systèmes d'exploitation, les attaques réseau ne peuvent pas être détectées en se basant uniquement sur les traces laissées par le système d'exploitation. C'est ainsi que les IDS basés sur le réseau ont été développés, positionnant le module IDS à l'intérieur du réseau pour une surveillance globale. Les systèmes de détection d'intrusion basés sur le réseau (NIDS) collectent et analysent les données directement capturées depuis le réseau. Leur fonctionnement repose généralement sur la capture et l'analyse des types et du contenu des paquets ou des flux traversant le réseau, afin d'identifier d'éventuels schémas d'attaques.

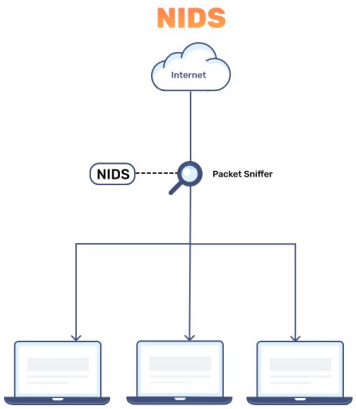

FIGURE 1.10 – NIDS.

2. **IDS basé sur l'hôte (HIDS) :**
   Les systèmes de détection d'intrusion basés sur l'hôte (HIDS) surveillent les activités sur un ordinateur protégé en examinant différentes sources de données présentes sur cet ordinateur, telles que les fichiers journaux, les appels système, les accès aux fichiers ou le contenu de la mémoire. En général, un HIDS se présente sous la forme d'un logiciel qui s'exécute directement sur l'ordinateur protégé, limitant ainsi sa portée de protection à cette seule machine. Par conséquent, pour sécuriser l'ensemble du réseau, un HIDS doit être déployé sur chaque machine du réseau interne. Deux sources de données principales sont couramment utilisées pour la détection basée sur l'hôte : les journaux d'audit et les appels système. Les journaux d'audit représentent un ensemble d'événements générés par le système d'exploitation pour diverses tâches, tandis que les appels système décrivent le comportement de chaque application essentielle à l'utilisateur exécutée sur le système d'exploitation.

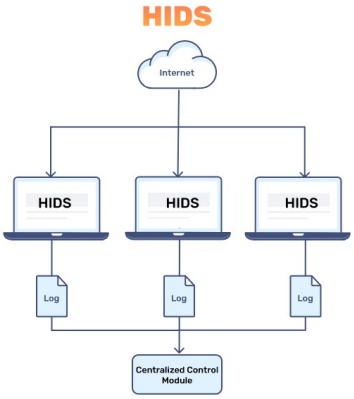

FIGURE 1.11 – HIDS.

3. **IDS hybrides :**
   Ces deux types d'IDS présentent des inconvénients distincts : l'IDS basé sur le réseau risque d'augmenter la charge de travail et de ne pas détecter toutes les activités malveillantes, tandis que l'IDS basé sur l'hôte ne surveille pas l'ensemble du trafic réseau et implique généralement une charge de travail moindre que l'IDS basé sur le réseau. Ainsi, pour pallier ces limitations, l'IDS hybride est déployé à la fois sur le réseau et sur les hôtes. Cette approche permet de surveiller simultanément les activités spécifiques aux clients ainsi que le trafic réseau global.

1.4.4.4 Selon la réponse aux intrusions

Une autre façon de classer les IDS est de les classer en fonction de leur mécanisme de réponse.

- Les IDS passifs opèrent habituellement de manière hors ligne, analysant les journaux système et les traces du trafic réseau. Dans certaines situations, ils peuvent

également fonctionner en ligne pour surveiller passivement les données d'audit des hôtes et le trafic réseau. Une fois qu'ils ont détecté une éventuelle attaque, les IDS passifs envoient des alertes d'intrusion à l'administrateur du système ou du réseau, qui prend alors les mesures appropriées en fonction de ces informations.

- En contraste, les IDS sont considérés comme actifs s'ils réagissent de manière proactive à une attaque en prenant des mesures pour arrêter l'attaque dès sa détection, sans nécessiter d'intervention humaine. Deux techniques principales sont utilisées à cette fin ; la reconfiguration du pare-feu et l'interruption de la connexion. La reconfiguration du pare-feu implique de bloquer le trafic malveillant au niveau du pare-feu en fermant le port utilisé ou en interdisant l'adresse de l'attaquant. D'autre part, un IDS actif peut interrompre une session établie entre un attaquant et sa cible afin de prévenir le transfert de données ou toute modification sur le système attaqué

1.4.4.5 SNORT

Snort est un Système de Détection d'Intrusions (IDS) et de Prévention d'Intrusions (IPS) open-source qui surveille le trafic réseau et identifie les menaces potentielles à la sécurité en se basant sur des règles prédéfinies. Développé en 1998 par Martin Roesch, il est maintenant entretenu par Cisco. Il peut également fonctionner comme un analyseur de paquets, ce qui lui permet de surveiller le trafic du système en temps réel.
Snort fonctionne en comparant le trafic réseau à un ensemble de règles définies par l'utilisateur. Lorsqu'il détecte un trafic correspondant à une règle, il peut déclencher des alertes ou effectuer d'autres actions spécifiées. Les règles de Snort sont relativement faciles à créer et à mettre en œuvre, et il peut être déployé sur divers systèmes d'exploitation, notamment Linux, Unix et Windows. [4]
Ce dernier a prouvé son efficacité en détectant plusieurs types d'attaques (tentative de fingerprinting, la vulnérabilité log4shell, l'attaque buffer overflow, botnets, l'utilisation du P2P, etc.) SNORT a trois modes de fonctionnement principaux :

- **Le mode sniffer (Reniflage de Paquets) :** dans ce mode, SNORT lit les paquets transitant le réseau et les affiche d'une façon continue sur l'écran.
- **Le mode (Packet Logger) :** dans ce mode SNORT journalise (log) le trafic réseau dans des répertoires sur le disque.
- **Le mode détection/prévention d'intrusion réseau (NIDS/NIPS) :** dans ce mode, SNORT analyse le trafic du réseau, et le traite. Ce qui signifie qu'il le compare à des règles prédéfinies par l'administrateur réseau ou l'équipe de sécurité et établit des actions à exécuter (accepter le trafic, alerter, bloquer le trafic, journaliser, etc.).

1. **Architecture de Snort :** L'architecture de Snort se compose de quatre composants principaux :

- **Moteur de détection :** Le moteur de détection est le cœur de Snort. Il applique les règles configurées aux paquets prétraités et génère des alertes lorsqu'il détecte une activité suspecte. Il utilise une combinaison de techniques de correspondance de modèles et de détection d'anomalies pour identifier les menaces potentielles.
- **Préprocesseurs :** se charge d'analyser et de recomposer le trafic capturé. Ils reçoivent les paquets directement capturés, éventuellement les retravaillent puis les fournissent au moteur de recherche de signatures.
- **Analyseur de paquets :** Ce composant capture et décode les paquets entrants. Il est responsable de l'extraction des données brutes des paquets réseau et de leur transmission au préprocesseur.
- **Sortie :** Le composant de sortie est responsable de la journalisation et du rapport des alertes générées par le moteur de détection. Il peut envoyer des alertes vers diverses destinations telles qu'une console, un fichier, un serveur Syslog ou une adresse e-mail.

2. **Les règles de SNORT :** La syntaxe d'une règle SNORT est comme illustré dans la figure 1.12 :

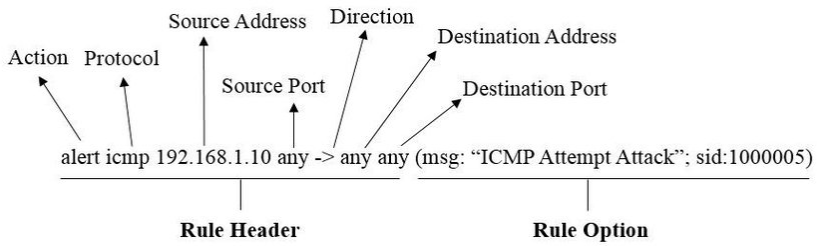

FIGURE 1.12 – La syntaxe d'une règle SNORT

Les règles de SNORT sont composées de deux parties distinctes : le header et les options.

— **Header :** Cela permet de définir le type d'alerte à émettre (alerte, journal ou pass) et de spécifier les champs de base nécessaires pour le filtrage : le protocole, ainsi que les adresses IP et les ports sources et de destination.
Les règles header :
  — **Action :** action de la règle. Exemple action = alert signifie que Snort va générer une alerte quand l'ensemble des conditions est rempli.
  — **Protocole :** protocole de la couche transport (TCP/UDP) utilisé ou de la couche réseau (ICMP).
  — **IP source :** la source du trafic.

- **Port source :** le port source du trafic.
- $\rightarrow$ **:** la direction du trafic.
- **IP destination :** la destination du trafic.
- **Port destination :** le port destination du trafic.

— **Options :**
Les spécifications entre parenthèses permettent de détailler l'analyse en divisant la signature en diverses valeurs à surveiller parmi certains champs de l'en-tête ou les données. Nous disposons de plusieurs options, notamment :

- **flags :** flag du header TCP activé.
- **content :** chercher un contenu dans le paquet.
- **msg :** Le message que Snort va afficher quand il envoie l'alerte.
- **sid :1000005 :** Identifiant de la règle Snort.
- **rev :1 :** Revision number (numéro de révision). Cette option permet une maintenance simplifiée de règle.
- **classtype :** Permet de catégoriser la règle comme par exemple "icmp-event" (l'une des catégories Snort prédéfinies). Permet aussi l'organisation des règles.

1.5 Apprentissage automatique

Le machine learning, également connu sous le nom d'apprentissage automatique, représente un domaine de l'intelligence artificielle axé sur l'enseignement aux machines la capacité d'apprendre à partir de données. Ceci se réalise à travers des modèles mathématiques, permettant ainsi aux machines de faire des prédictions ou de prendre des décisions sans nécessiter une programmation explicite. Ce domaine englobe l'utilisation de techniques statistiques, d'optimisation mathématique, et d'algorithmes informatiques pour analyser de vastes ensembles de données et en extraire des modèles ainsi que des relations significatives.

1.5.1 Types d'apprentissage automatique

Diverses méthodes d'apprentissage automatique existent, chacune présentant ses propres points forts et faiblesses.

— **Apprentissage supervisé :**
Il s'agit d'une forme d'apprentissage automatique où un modèle est formé sur un ensemble de données étiquetées afin de prédire des résultats pour de nouvelles données

non vues auparavant. Cette méthode est largement utilisée dans divers domaines tels que la reconnaissance d'images et vocale, le traitement du langage naturel, ainsi que les systèmes de recommandation.

— **Apprentissage non supervisé :**
Il s'agit d'entraîner un modèle sur un ensemble de données non étiquetées pour identifier des schémas et des liens au sein de ces données. Cette méthode a été utilisée dans divers contextes, tels que la détection d'anomalies, le regroupement et la réduction de la complexité des données

— **Apprentissage semi-supervisé :**
C'est un hybride entre l'apprentissage supervisé et non supervisé où des données étiquetées sont ajoutées aux données non étiquetées en tant que données de supervision pour l'entrainement du modèle.

— **Le renforcement learning :**
Il s'agit du "renforcement Learning" ou apprentissage par renforcement, une méthode d'apprentissage automatique où un agent apprend à prendre des actions dans un environnement afin de maximiser une récompense cumulée. Contrairement aux autres formes d'apprentissage, il n'y a pas de données d'entraînement étiquetées. Au lieu de cela, l'agent explore son environnement et apprend par essais et erreurs quelle action lui permet d'obtenir la meilleure récompense. Cette approche est largement utilisée dans des domaines tels que la robotique, les jeux et l'optimisation, où il est difficile de définir explicitement des règles ou des stratégies.

1.5.2 Algorithmes d'apprentissage automatique

Les algorithmes d'apprentissage automatique permettent aux systèmes informatiques de reconnaître des motifs et de prendre des décisions basées sur des données. Ces techniques sont cruciales dans divers domaines, allant de la reconnaissance d'images à la prévision financière. Les deux principales catégories d'algorithmes d'apprentissage supervisé sont la classification et la régression, chacune ayant des applications spécifiques et des méthodes distinctes pour résoudre des problèmes complexes.

- **Classification :** La classification est une technique d'apprentissage automatique qui consiste à entraîner un modèle à attribuer une étiquette de classe à une entrée donnée. Il s'agit d'une tâche d'apprentissage supervisé, ce qui signifie que le modèle est formé sur un ensemble de données étiquetées qui comprend des exemples de données d'entrée et les étiquettes de classe correspondantes.
  Le modèle vise à apprendre la relation entre les données d'entrée et les étiquettes de classe afin de prédire l'étiquette de classe pour de nouvelles entrées inédites [31].
- **Regréssion :** la régression est un type d'apprentissage supervisé dont l'objectif est de prédire une variable dépendante c sur la base d'une ou plusieurs caractéristiques

d'entrée (également appelées prédicteurs ou variables indépendantes).
Les algorithmes de régression sont utilisés pour modéliser la relation entre les entrées et les sorties et faire des prédictions basées sur cette relation. La régression peut être utilisée pour des variables dépendantes continues ou catégorielles.
En général, l'objectif de la régression est de construire un modèle qui peut prédire avec précision la sortie sur la base des caractéristiques d'entrée et de comprendre la relation sous-jacente entre les caractéristiques d'entrée et la sortie [23].

1.5.2.1 Arbre de décision

Un arbre de décisions est un algorithme d'apprentissage supervisé non paramétrique, qui est utilisé à la fois pour les tâches de classification et régression. Il a une structure hiérarchique, une structure arborescente, qui se compose d'un nœud racine, de branches, de nœuds interne et de nœuds feuille.
L'algorithme de l'Arbre de Décision identifie automatiquement les meilleures caractéristiques pour bâtir l'arbre initial, puis il procède à un élagage pour éliminer les branches non pertinentes et ainsi prévenir le surajustement du modèle [27].

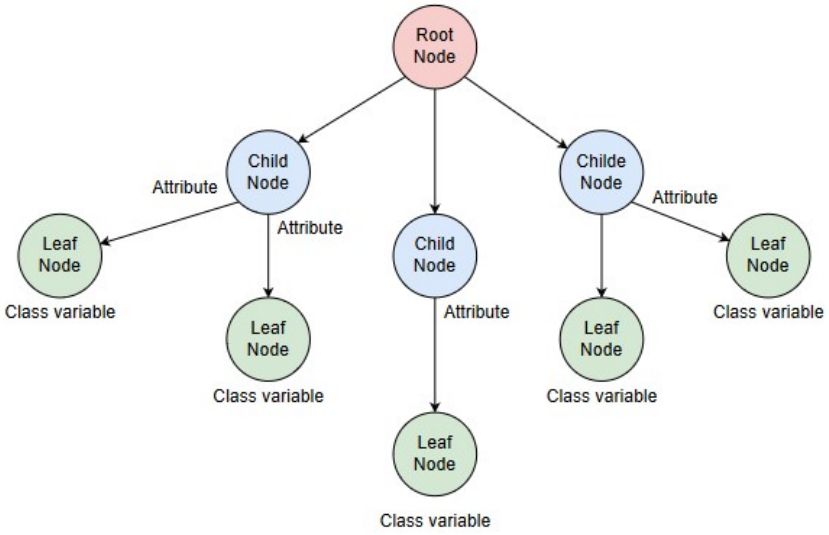

FIGURE 1.13 – Arbre de décision.

1.5.2.2 Les forêts aléatoires

Les forêts aléatoires sont un algorithme de machine learning conçu pour obtenir une prédiction fiable grâce à un système de sous-espaces aléatoires. Elles sont composées de plusieurs arbres de décision, entrainés de manière indépendante sur des sous-ensembles du dataset d'apprentissage (méthode de bagging). Chacun produit une estimation, la moyenne (ou le vote, dans le cas d'un problème de classification) de tous les arbres est la

prédiction finale. Elles sont utilisées pour réduire la variance des prévisions d'un arbre de décision seul, améliorant ainsi leurs performances. Les forêts aléatoires ont été proposées par Leo Breiman et Adele Cutler en 2001 [30].

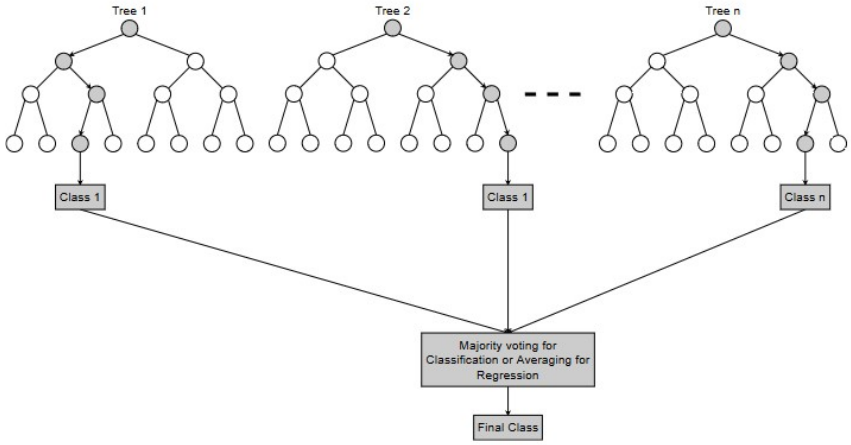

FIGURE 1.14 – Foret aléatoire.

1.5.2.3 Machine à vecteur de support (SVM)

Les Machines à Vecteurs de Support (SVM) sont une méthode de classification supervisée qui opère en identifiant un hyperplan de séparation, aussi appelé frontière de décision, dans un espace de caractéristiques. Elles sont utilisées pour résoudre des problèmes à la fois linéaires et non linéaires. L'objectif est de trouver cet hyperplan de manière à maximiser la marge entre les deux classes, assurant ainsi une séparation optimale Mathématiquement, l'hyperplan est décrit par une équation linéaire de la forme $ax + b = 0$, où la variable a représente le vecteur de poids qui définit la direction de l'hyperplan, tandis que b, le biais, détermine sa position dans l'espace [37].

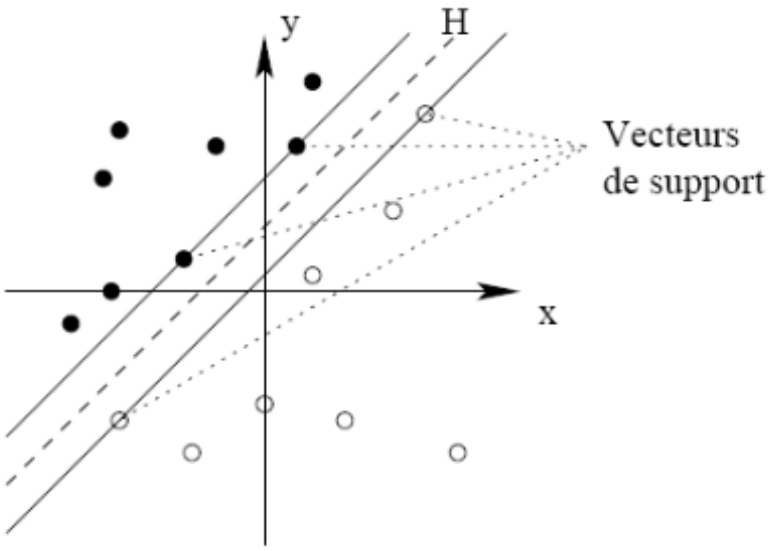

FIGURE 1.15 – Hyperplan séparateur de deux classes.

Dans le schéma 1.15, on détermine un hyperplan qui sépare les deux ensembles de points. Les points les plus proches, qui seuls sont utilisés pour la détermination de l'hyperplan, sont appelés vecteurs de support. Il est évident qu'il existe une multitude d'hyperplan valide mais la propriété remarquable des SVM est que cet hyperplan doit être optimal. Nous allons donc en plus chercher parmi les hyperplans valides, celui qui passe « au milieu » des points des deux classes d'exemples. Intuitivement, cela revient à chercher l'hyperplan le « plus sûr ».

En effet, supposons qu'un exemple n'ait pas été décrit parfaitement, une petite variation ne modifiera pas sa classification si sa distance à l'hyperplan est grande. Formellement, cela revient à chercher un hyperplan dont la distance minimale aux exemples d'apprentissage est maximale. On appelle cette distance « marge » entre l'hyperplan et les exemples. L'hyperplan séparateur optimal est celui qui maximise la marge. Comme on cherche à maximiser cette marge, on parlera de séparateurs à vaste marge.

1.5.2.4 Les k plus proches voisins (KNN)

Le K-Nearest Neighbor (KNN) est l'un des algorithmes de classification les plus basiques mais essentiels en apprentissage automatique. Il utilise la proximité pour effectuer des classifications ou des prédictions sur le regroupement d'un point de données individuel. Pour déterminer si un point de données appartient à une classe spécifique, il commence par sélectionner le nombre de voisins (k) et calcule la distance pour trouver les k voisins les plus proches du point de données non classé. Parmi ces k voisins, le nombre de points de données est calculé dans chaque classe, ce qui permet de classer l'échantillon inconnu

dans la classe avec le plus grand nombre de voisins [13].

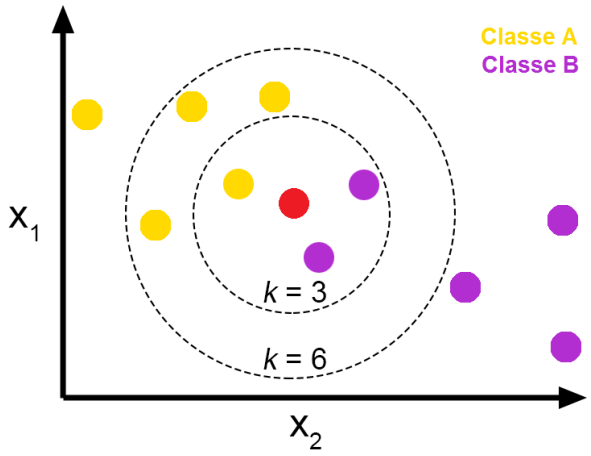

FIGURE 1.16 – KNN.

1.5.3 Algorithmes d'apprentissage profond

L'apprentissage profond (deep learning) est une sous-catégorie de l'apprentissage automatique qui utilise des réseaux neuronaux multicouches, appelés réseaux neuronaux profonds, pour simuler la capacité complexe de prise de décision du cerveau humain. Le deep learning a véritablement transformé divers secteurs tels que la vision artificielle, le traitement du langage naturel et la reconnaissance vocale, en délivrant des performances remarquables dans une multitude de domaines [29].

1.5.3.1 Perceptrons multicouches (MLP)

Le perceptron multicouche (MLP) est un type de réseau de neurones artificiels conçu pour traiter des données complexes et résoudre des problèmes de classification et de régression. Il se compose de plusieurs couches de neurones : une couche d'entrée, une ou plusieurs couches cachées, et une couche de sortie. Chaque neurone de chaque couche est connecté à chaque neurone de la couche suivante via des poids ajustables. Les MLP utilisent des fonctions d'activation non linéaires, permettant de capturer des relations complexes dans les données. Ils sont entraînés par rétropropagation, ajustant les poids pour minimiser l'erreur de prédiction.

Par exemple, nous connectons trois perceptrons : les deux premiers reçoivent les entrées x_1 et x_2, effectuent des calculs basés sur leurs paramètres, et renvoient deux sorties y_1 et y_2. Ces sorties sont ensuite envoyées au troisième perceptron, qui effectue des calculs pour produire la sortie finale y_3.

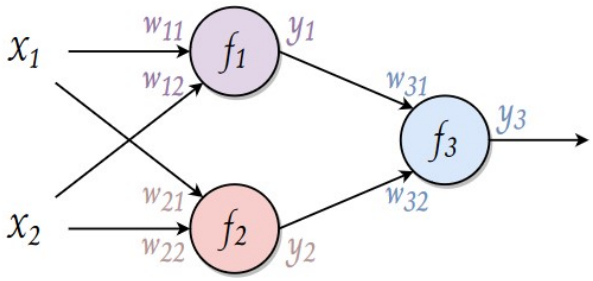

FIGURE 1.17 – Modèle MLP.

1.5.4 Classification des datasets conçus pour les IDS

Il existe divers ensembles de données pour l'entraînement des modèles de machine learning, créés à l'aide d'outils variés, de générateurs de trafic et de simulateurs de réseau. Ils sont classifiés selon leur environnement de création et leurs objectifs spécifiques [18].

1.5.4.1 Datasets basés sur les flux

Les datasets basés sur les flux regroupent des données de trafic réseau mettant l'accent sur les caractéristiques des flux de communication entre hôtes. Ces ensembles incluent KDD cup 1999, CSE-CIC-IDS2018 et IoT_Botnet_Dataset, capturant des informations agrégées telles que la durée, le volume de données, le nombre de paquets, le taux de transfert, et les adresses et ports utilisés.

1.5.4.2 Datasets basés sur les paquets

Les datasets basés sur les paquets enregistrent chaque paquet individuel transmis sur un réseau, fournissant des détails granulaires tels que les en-têtes des paquets (adresses IP, ports, etc.), les charges utiles, les horodatages et la taille des paquets. Parmi eux, on trouve CICDataset-ISCX-Bot-2014, UNSW-NB15 et NSL-KDD.

1.5.4.3 Datasets générés avec les données des applications

Les datasets générés à partir des données des applications capturent directement les interactions des utilisateurs, les transactions et les événements internes des applications. Ils incluent AndroidBotnets_Dataset et CICMalDroid_2020, fournissant des données transactionnelles, des logs d'activité des utilisateurs, des événements d'application, des données de performance et des métadonnées détaillées.

Ces classifications facilitent le choix d'ensembles de données adaptés aux besoins spécifiques en détection d'intrusions, analyse de trafic et optimisation des systèmes de sécurité réseau.

1.5.5 Méthodes pour la sélection des attributs

La sélection des attributs (features) est l'une des approches utilisées pour la réduction des dimensions de dataset. Il s'agit d'une étape cruciale qui influence directement la performance de l'IDS. Les features pertinentes doivent être choisies pour optimiser la capacité de l'algorithme à détecter les intrusions. Il y a de nombreux facteurs à prendre en compte lors de la sélection des features pour un modèle de machine learning :

— **Pertinence :** Les features doivent être pertinentes par rapport au problème que nous sommes en train de traiter.
— **Corrélation :** Les features ne doivent pas être fortement corrélées entre elles. Cela peut conduire à la multicolinéarité, ce qui peut rendre difficile pour le modèle de faire des prédictions précises.
— **Qualité :** Les features doivent être de haute qualité, avec un minimum de valeurs manquantes ou aberrantes.
— **Représentation :** Les features doivent fournir une représentation complète des données.

Il existe plusieurs méthodes pour déterminer les caractéristiques qui auront le plus d'influence sur nos modèles, ces méthodes on peut les partager en trois catégories : Méthodes de filtrage, wrapper et intégrées [26]. Il existe aussi des méthodes statistiques pour obtenir un aperçu sur la nature des relations entre les caractéristiques comme l'analyse de corrélation.

1.5.5.1 Analyse de corrélation

L'analyse de corrélation est une technique statistique fondamentale employée pour évaluer la relation entre deux variables. Elle permet de quantifier la force et la direction de cette association, offrant des informations précieuses dans divers domaines scientifiques et d'analyse de données [32].

• **Fonctionnment :** La corrélation mesure le degré auquel deux variables varient ensemble. Une forte corrélation positive indique que les deux variables augmentent ou diminuent simultanément, tandis qu'une forte corrélation négative suggère que l'une augmente lorsque l'autre diminue. Une corrélation proche de zéro implique une relation faible ou

inexistante.
Le coefficient de corrélation, noté "r", est une mesure numérique de la corrélation, variant généralement entre -1 et 1. Un coefficient de corrélation positif indique une relation positive, tandis qu'un coefficient négatif indique une relation négative. Un coefficient de corrélation de 0 indique l'absence de relation linéaire entre les variables.

• **Types d'analyse de corrélation :** Divers types d'analyse de corrélation existent, chacun adapté à des situations spécifiques :

1. **Corrélation de Pearson :** La méthode la plus courante, adaptée aux données continues linéaires.
2. **Corrélation de Spearman :** Plus robuste aux valeurs aberrantes et adaptée aux données non paramétriques.
3. **Corrélation de Kendall :** Mesure la concordance entre les rangs des observations, utile pour les données ordinales.

• **Interprétation du coefficient de corrélation :** L'interprétation du coefficient de corrélation va au-delà de sa valeur numérique. Il est crucial de considérer le contexte et la nature des données analysées. Un coefficient de corrélation élevé ne garantit pas nécessairement une relation causale entre les variables. D'autres facteurs, tels que des variables confondantes, peuvent influencer l'association observée.

1.6 Conclusion

En conclusion, ce chapitre souligne l'importance croissante de l'apprentissage automatique dans la sécurité informatique. Nous avons exploré divers types d'attaques cybernétiques et les mécanismes de défense correspondants, ainsi que les applications de l'apprentissage automatique dans la détection d'anomalies et la prévention des attaques. Cette convergence entre sécurité informatique et apprentissage automatique ouvre de nouvelles voies pour renforcer la résilience des systèmes face aux menaces en ligne, mais nécessite une approche réfléchie et collaborative pour relever les défis futurs.

Chapitre 2

Présentation de l'entreprise d'accueil

2.1 Introduction

Ce chapitre présente Tchin-Lait, une entreprise algérienne fondée en 1999 et spécialisée dans la production de lait UHT. Nous examinerons son historique, sa transition stratégique vers le lait UHT, son partenariat avec Candia, ainsi que sa structure juridique et organisationnelle. En outre, nous aborderons le réseau informatique de Tchin-Lait, mettant en lumière son infrastructure technologique et son rôle dans le fonctionnement global de l'entreprise.

2.2 Présentation de l'entreprise :

Dans cette section, on va suivre l'évolution remarquable de Tchin-Lait, entreprise fondée en 1999 après la transition stratégique de Tchin-Tchin vers la production de lait UHT, marquant ainsi son impact dans le secteur agro-alimentaire algérien.

2.2.1 Historique et Fondation

Tchin-Lait est une entreprise algérienne fondée le 17 août 1999, succédant à la société Tchin-Tchin, un leader dans la fabrication de boissons gazeuses pendant plus de 50 ans. Le nom Tchin-Lait symbolise la continuité de l'héritage de la famille Berkati, qui a dirigé Tchin-Tchin et a lancé des marques célèbres comme SLIM. Face à la concurrence croissante des multinationales comme Coca-Cola et Pepsi, Tchin-Tchin a dû réviser sa stratégie et a finalement choisi de se reconvertir dans la production de lait UHT, un segment alors inexistant en Algérie [46].

2.2.2 Transition vers la Production de Lait UHT

La décision de produire du lait UHT a été influencée par plusieurs facteurs :

- La demande croissante de produits laitiers en Algérie.
- L'inexistence de lait UHT sur le marché national, qui était dominé par le lait pasteurisé.
- Les conditions climatiques et les infrastructures de distribution en Algérie, rendant le lait UHT plus adapté à la distribution sans chaîne de froid.

2.2.3 Partenariat avec Candia

Pour assurer une transition réussie, Tchin-Lait a opté pour un partenariat en franchise avec Candia, une marque de renom en Europe. Ce partenariat, signé le 21 avril 1999, a permis à Tchin-Lait de bénéficier du savoir-faire, de l'assistance technique, et de l'expertise commerciale de Candia, facilitant ainsi l'acquisition de compétences cruciales et l'accès à des outils de gestion et de promotion efficaces [46].

2.2.4 Situation Juridique et Organisationnelle

Initialement créée en tant que SARL, Tchin-Lait a évolué pour devenir une Société par Actions (SPA) en 2017. Cette transformation s'est accompagnée de la filialisation de la Générale Laiterie Jugurtha et de la création de nouvelles filiales comme Tchin Agro SPA et Tchin Logistique SPA, permettant une meilleure gestion des ressources et des responsabilités [46].

2.2.5 Capacités de Production et Gamme de Produits

Tchin-Lait dispose d'une capacité de production totale de 415 millions de litres par an, répartie sur trois sites de production à Béjaïa, Alger, et Sétif. L'entreprise propose une large gamme de produits laitiers, incluant des laits stérilisés UHT (entier, partiellement écrémé, écrémé, et enrichi en vitamines), des boissons au lait aromatisées, et des laits additionnés de jus de fruits. Le procédé UHT utilisé par Tchin-Lait garantit la préservation des qualités nutritionnelles et organoleptiques du lait [46].

2.2.6 Réseau de Distribution et Ressources Humaines

En termes de distribution, Tchin-Lait a considérablement étendu son réseau, passant de 15 wilayas couvertes en 2008 à 46 wilayas en 2018, avec plus de 52 000 points de vente livrés chaque semaine. L'entreprise emploie 1 196 agents, incluant des cadres, des agents de maîtrise, et des agents d'exécution, tous bénéficiant de formations spécialisées pour assurer la qualité et l'efficacité de la production.

Tchin-Lait, en tant qu'acteur majeur du secteur agro-alimentaire algérien, continue de se développer et d'innover pour répondre aux besoins des consommateurs tout en maintenant ses valeurs de qualité et de durabilité [46].

2.2.7 Réseau de l'entreprise

Un réseau d'entreprise est un système informatique interconnecté utilisé par une organisation pour permettre la communication et le partage de ressources entre ses employés et ses équipements [12].
Dans ce qui suit, on va présenter l'architecture réseau de l'entreprise (Figure 2.1).

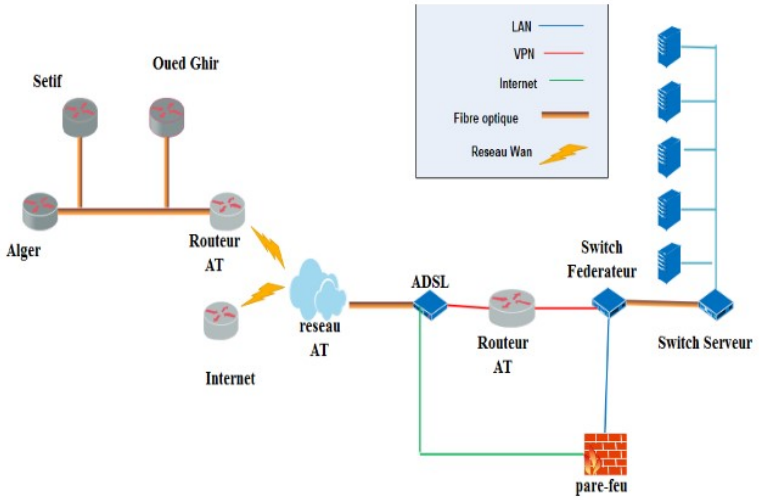

FIGURE 2.1 – L'architecture réseau du Tchin-Lait

2.3 Problématique

Les IDS sont d'une importance cruciale pour la sécurité des réseaux informatiques moderne. En surveillant en temps réel les activités réseau et système, ils permettent de repérer et de réagir rapidement aux comportements malveillants avant qu'ils ne causent des dommages.

2.3.1 Description du problème

Selon [46], Tchin-Lait utilise un dispositif agissant comme un pare-feu pour filtrer le trafic réseau entre le LAN et le WAN. Ce pare-feu, le SOPHOS XG, est équipé d'un système de prévention des intrusions (IPS) intégré, capable de déclencher des alertes et d'appliquer des mesures de prévention en fonction du type d'attaque détectée. Cet IPS repose sur SNORT, un outil largement reconnu dans le domaine de la sécurité pour son efficacité et sa fiabilité.

Cependant, SNORT fonctionne sur la base de signatures. En d'autres termes, il peut uniquement détecter les attaques dont les signatures sont déjà enregistrées dans sa base

de données. Cette approche présente une limitation significative : elle rend l'IPS inefficace contre les attaques de type zero-day, qui sont des vulnérabilités inconnues des concepteurs de logiciels, ou contre les variantes d'attaques existantes, car une légère modification d'une attaque peut générer une signature différente de celle initialement répertoriée.
Les faux positifs peuvent également poser problème, car parfois, du trafic légitime est identifié à tort comme malveillant. De plus, la détection basée sur les signatures peut être gourmande en ressources, car elle doit analyser chaque paquet de données par rapport à un grand nombre de signatures, ce qui peut ralentir les performances du réseau.

2.3.2 Techniques d'évasion des IDS basés sur la signature

Cette section aborde les techniques qu'un cybercriminel peut utiliser pour éviter la détection par les IDS, telles que la fragmentation, l'inondation, l'obfuscation et le chiffrement. Ces techniques posent un défi aux IDS actuels car elles contournent les méthodes de détection existantes.

2.3.2.1 Fragmentation

Un paquet est divisé en paquets plus petits. Les paquets fragmentés sont ensuite réassemblés par le nœud récepteur au niveau IP avant de les transmettre au niveau Application. Pour examiner correctement le trafic fragmenté, l'IDS doit assembler ces fragments de la même manière qu'au point de fragmentation. La restructuration des paquets nécessite que l'IDS conserve les données en mémoire et compare le trafic à une base de données de signatures.
Les méthodes utilisées par les attaquants pour échapper à la détection en cachant les attaques sous forme de trafic légitime incluent le chevauchement (Overlap), l'écrasement (Overwrite) et les délais de fragmentation (Timeouts) [19]. L'attaque par fragmentation remplace les informations dans les paquets fragmentés constituants par de nouvelles informations pour générer un paquet malveillant.

FIGURE 2.2 – Chevauchement des paquets lors de la fragmentation.

La durée pendant laquelle l'IDS peut maintenir un état de trafic peut être inférieure à celle

pendant laquelle l'hôte de destination peut maintenir un état de trafic. Les auteurs de logiciels malveillants tentent de tirer parti de toute lacune dans la méthode de détection en livrant des fragments d'attaque sur une longue période [44].

2.3.2.2 Inondation (Flooding)

L'attaquant commence l'attaque pour submerger l'IDS, ce qui entraîne une défaillance du mécanisme de contrôle. Lorsque l'IDS échoue, tout le trafic est autorisé. Une méthode populaire pour créer une situation d'inondation consiste à usurper le protocole de datagramme utilisateur (UDP) légitime et le protocole de message de contrôle Internet (ICMP). L'inondation de trafic est utilisée pour dissimuler les activités anormales du cybercriminel. Par conséquent, les IDS auraient énormément de difficulté à trouver des paquets malveillants dans une grande quantité de trafic [48].

2.3.2.3 Obfuscation

Les techniques d'obfuscation peuvent être utilisées pour échapper à la détection, ce sont les techniques de dissimulation d'une attaque en rendant le message difficile à comprendre. La terminologie de l'obfuscation signifie modifier le code du programme de manière à le garder fonctionnellement identique tout en réduisant sa détectabilité à tout type d'analyse statique ou de processus d'ingénierie inverse et en le rendant obscur et moins lisible. Cette obfuscation des logiciels malveillants leur permet d'échapper aux IDS actuels.
L'obfuscation tente d'exploiter toutes les limitations de la base de données de signatures et de sa capacité à dupliquer la manière dont l'hôte informatique examine les données de l'ordinateur. Un IDS efficace devrait prendre en charge le format de codage hexadécimal ou avoir ces chaînes hexadécimales dans son ensemble de signatures d'attaque [19]. La norme Unicode/UTF-8 permet de symboliser un caractère de plusieurs formats différents. Les cybercriminels peuvent également utiliser des données doublement encodées, augmentant exponentiellement le nombre de signatures nécessaires pour détecter l'attaque.
Les IDS basés sur les signatures (SIDS) reposent sur la correspondance de signatures pour identifier les logiciels malveillants, où les signatures sont créées par des experts humains en traduisant un logiciel malveillant du code machine en un langage symbolique tel que Unicode. Cependant, l'utilisation de l'obfuscation de code est très précieuse pour les cybercriminels afin d'éviter les IDS.

2.3.2.4 Chiffrement

Généralement, le chiffrement offre un certain nombre de services de sécurité, tels que la confidentialité des données, l'intégrité et la vie privée. Les auteurs de logiciels malveillants utilisent ces attributs de sécurité pour échapper à la détection et dissimuler les attaques pouvant cibler un système informatique. Par exemple, les attaques sur les protocoles chiffrés tels que le protocole HTTPS (HyperText Transfer Protocol Secure) ne peuvent pas être lues par un IDS [48].
L'IDS ne peut pas faire correspondre le trafic chiffré aux signatures de la base de données existante s'il n'interprète pas le trafic chiffré. Par conséquent, l'examen du trafic chiffré rend difficile la détection des attaques par les IDS. Par exemple, les caractéristiques basées sur le contenu des paquets ont été largement appliquées pour identifier les logiciels malveillants du trafic normal, ce qui ne peut pas être facilement appliqué si le paquet est chiffré.

La problématique majeure ici est la capacité limitée de cet IPS à identifier et à prévenir les nouvelles menaces. Les attaques zero-day représentent un risque particulièrement élevé car elles exploitent des failles de sécurité avant même que des signatures puissent être développées et ajoutées à la base de données de l'IPS. De plus, les cybercriminels peuvent facilement modifier des attaques connues pour créer de nouvelles signatures où utiliser les téchniques d'évasions citées ci-dessus pour échapper à la détection basée sur les signatures.

Ces défis motivent les chercheurs à utiliser certaines caractéristiques statistiques du flux réseau, qui ne reposent pas sur le contenu des paquets. En conséquence, les logiciels malveillants peuvent potentiellement être identifiés à partir du trafic normal.

2.3.3 Les risques courus en l'absence d'un IDS performant

En l'absence d'un IDS performant, les systèmes d'information d'une industrie sont particulièrement vulnérables à une série de risques graves. Tout d'abord, les tentatives d'intrusion peuvent passer inaperçues, permettant aux attaquants de s'infiltrer dans le réseau et d'accéder à des informations sensibles, telles que des données financières, des secrets commerciaux et des informations personnelles des clients. Cette exposition accrue aux intrusions non détectées augmente considérablement le risque de perte de données, ce qui peut avoir des conséquences financières et opérationnelles désastreuses pour l'entreprise.

De plus, les cyberattaques peuvent entraîner l'interruption des opérations quotidiennes. Les attaques par déni de service (DoS) et autres formes d'intrusions peuvent perturber les activités industrielles, provoquant des temps d'arrêt coûteux et des pertes de productivité. Ces interruptions peuvent être particulièrement dévastatrices dans des secteurs où

la continuité des opérations est cruciale. L'absence de mesures adéquates de détection des intrusions peut également conduire à des violations de la conformité réglementaire. De nombreuses industries sont soumises à des réglementations strictes en matière de protection des données, et ne pas respecter ces normes peut exposer l'entreprise à des amendes, des sanctions et des actions en justice.

Une intrusion réussie peut également nuire gravement à la réputation de l'entreprise. La perte de confiance des clients, des partenaires et des investisseurs peut avoir des effets durables sur la position de l'entreprise sur le marché. Faute d'un IDS performant, les faiblesses et vulnérabilités du système d'information peuvent être exploitées sans être détectées, permettant aux attaquants de lancer des attaques plus sophistiquées et destructrices. En outre, les menaces internes, telles que les employés malveillants ou négligents, peuvent abuser de leurs accès pour compromettre les systèmes d'information, et sans surveillance adéquate, ces activités peuvent causer des dommages importants.

Enfin, les malwares peuvent se propager rapidement dans un réseau non surveillé, infectant plusieurs systèmes et entraînant des pertes de données et des interruptions de service. L'absence d'un IDS efficace laisse les systèmes d'information d'une industrie extrêmement vulnérables à une gamme de menaces cybernétiques, avec des conséquences potentiellement dévastatrices pour la sécurité, l'intégrité et la disponibilité des données et des services.

2.4 Conclusion

En conclusion, ce chapitre a permis de présenter l'entreprise d'accueil, en mettant en lumière son rôle crucial dans son secteur d'activité. Nous avons exploré l'importance des systèmes de détection d'intrusion (IDS) pour protéger les infrastructures informatiques contre les menaces cybernétiques. Bien que des solutions comme Snort et Suricata soient largement utilisées et efficaces pour détecter des attaques connues, elles présentent des limites notables face aux nouvelles menaces et aux attaques sophistiquées. Ces observations soulignent la nécessité pour l'entreprise de compléter ses IDS traditionnels avec des approches de sécurité plus avancées et adaptatives afin de renforcer sa posture de défense et de mieux protéger ses ressources critiques.

Chapitre 3

IDS avec les méthodes classiques et modernes

3.1 Introduction

Dans les chapitres précédents, nous avons exploré les réseaux informatiques actuels et les diverses menaces croissantes qui les mettent en danger. Nous avons également étudié les réseaux de botnets et le danger qu'ils représentent en termes de cyberattaques potentielles contre nos infrastructures réseau.

L'un des outils essentiels pour prévenir ces actes malveillants est le système de détection d'intrusions (IDS), qui permet de réagir à temps pour stopper ces attaques. De nos jours, les IDS les plus performants utilisent des modèles de machine learning entraînés sur des données réseau pour identifier diverses menaces.

Dans ce chapitre, nous allons examiner les deux approches présentes pour la mise en fonction d'un IDS surveillant un système. La première qui est une méthode classique en utilisant Snort et pour la deuxième, nous allons faire usage des techniques d'apprentissage automatique pour entraîner un modèle sur un dataset basé sur les flux le rendant ainsi, capable de détecter des anomalies au sein d'un réseau.

3.2 Mise en place de SNORT

Dans cette section, nous allons explorer les étapes nécessaires pour la mise en place de Snort dans un système linux, et les différentes configurations pour surveiller et contrôler le trafic.

3.2.1 Environnement

Snort est un système de détection d'intrusions open-source, rendu disponible sur plusieurs systèmes d'exploitation tels que Linux et Windows grâce aux efforts de sa vaste communauté. Cet outil est couramment utilisé dans des environnements nécessitant une surveillance constante du trafic réseau afin de protéger les données sensibles des serveurs et des bases de données.

Pour la mise en place de cet IDS dans notre démonstration, nous avons choisi d'utiliser un système Linux pour faciliter la configuration de l'outil et assurer une compatibilité maximale. Ainsi, nous avons opté pour **Kali Linux**. En utilisant **VMware Workstation**, nous avons créé une machine virtuelle avec l'image de Kali, spécialement conçue pour la virtualisation. Les caractéristiques de cette VM sont indiquées dans le tableau suivant :

Image	Kali Rolling (2024.2) x64
CPU	i7 4770 | 2 coeurs, 2 threads
RAM	4 GB
Stockage	80 Go SSD

TABLE 3.1 – Ressources allouées à la VM.

3.2.2 Installation

Pour installer Snort sur un système Linux, deux méthodes principales s'offrent à nous. La première, et la plus simple, consiste à utiliser le gestionnaire de paquets APT sur une distribution Debian. La seconde, plus complexe, nécessite de compiler Snort à partir de son code source disponible sur GitHub. Dans ce guide, nous allons utiliser la méthode la plus simple.

Une fois la machine virtuelle démarrée, on ouvre un terminal bash ou zsh et on exécute les commandes suivantes :

```
$ sudo apt-get update
```

Lors de la première utilisation du système, cette commande actualise les liens vers les répertoires listés dans le fichier des sources `/etc/sources.list`, permettant ainsi de télécharger Snort via le gestionnaire de paquets APT.

Ensuite, pour installer Snort, on exécute :

```
$ sudo apt-get install snort
```

Cette commande télécharge et installe Snort ainsi que toutes les dépendances nécessaires.

Une fois l'execution des commandes est terminée, on vérifie si l'installation est réussie avec la commande suivante :

```
$ snort -V
```

On devrait avoir une sortie indiquant que snort se lance sans erreurs comme dans la figure suivante :

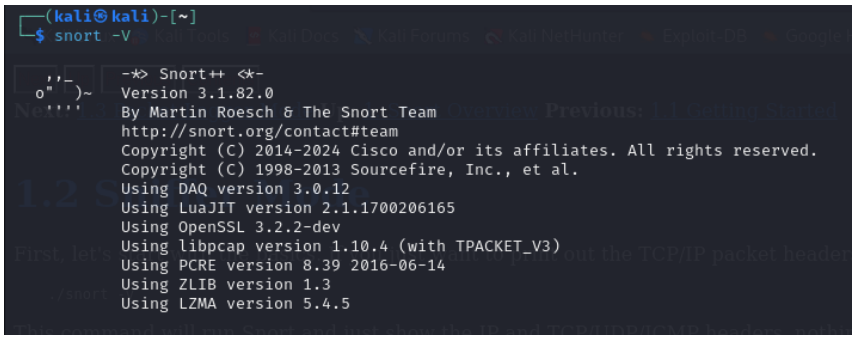

FIGURE 3.1 – Versions des dépendances Snort.

3.2.3 Utilisation de snort

Snort est un moteur polyvalent extrêmement puissant. Dans cette section, nous allons aborder les bases de l'utilisation de Snort en ligne de commande, discuter brièvement de la manière de configurer et d'ajuster ses paramètres, et enfin, expliquer comment utiliser Snort pour détecter et prévenir les attaques.

3.2.3.1 Les lignes de commandes basiques

Exécuter Snort en ligne de commande est facile, mais le nombre d'arguments disponibles peut être intimidant au début. Commençons donc par les bases.
Toutes les commandes Snort commencent par 'snort', et exécuter cette commande seule affichera les instructions d'utilisation de base :

```
$ snort
usage:
    snort -?: list options
    snort -V: output version
    snort --help: help summary
    snort [-options] -c conf [-T]: validate conf
    snort [-options] -c conf -i iface: process live
    snort [-options] -c conf -r pcap: process readback
```

Snort fournit un ensemble de commandes d'aide très robuste qui détaille presque tous les aspects du moteur.

Pour voir le "répertoire" principal de l'aide, on exécute la commande suivante :

```
$ snort --help
```

Snort propose plusieurs options pour obtenir plus d'aide :

```
-? list command line options (same as --help)
--help this overview of help
--help-commands [<module prefix>] output matching commands
--help-config [<module prefix>] output matching config options
--help-counts [<module prefix>] output matching peg counts
--help-limits print the int upper bounds denoted by max*
--help-module <module> output description of given module
--help-modules list all available modules with brief help
```

Comme nous pouvons le voir dans la sortie, Snort contient des pages d'aide séparées pour les différentes parties du moteur Snort, et ces sous-pages peuvent être utilisées pour obtenir des informations d'aide détaillées sur un composant particulier. Voici quelques-unes de ces sous-pages d'aide.

Lister tous les modules disponibles de Snort :

```
$ snort --list-modules
```

Obtenir de l'aide sur un module spécifique de Snort :

```
$ snort --help-module http_inspect
```

Obtenir de l'aide sur un module d'option de règle spécifique :

```
$ snort --help-module http_uri
```

Lister les options de ligne de commande disponibles :

```
$ snort -?
```

Obtenir de l'aide sur l'option de ligne de commande "-A" :

```
$ snort --help-options A
```

3.2.3.2 Lecture de trafic

Snort est à son meilleur lorsqu'il a du trafic réseau à analyser, et il peut effectuer cette inspection de plusieurs manières différentes. Cela inclut la lecture du trafic directement à partir d'une capture de paquets, l'exécution passive sur une interface réseau pour renifler le trafic, et le test des capacités d'injection en ligne de Snort localement. Avant de pouvoir entrer dans le détail, nous devons d'abord voir comment fournir du trafic à Snort pour qu'il puisse l'analyser.

Pour que Snort fonctionne correctement, il est crucial de configurer l'installation pour qu'elle puisse localiser la bibliothèque LibDAQ appropriée, installée précédemment. LibDAQ, ou "Data Acquisition Library", est une couche d'abstraction utilisée par les modules pour interagir avec des sources de données réseau, qu'elles soient matérielles ou logicielles. Par exemple, un module DAQ installé par défaut est le module pcap, qui repose sur la bibliothèque `libpcap` pour écouter les interfaces réseau ou lire des fichiers `.pcap`.

— **Lecture des fichiers .pcap :**
La manière la plus simple de voir Snort en action est de l'exécuter sur un fichier de capture de paquets. Il suffit de passer le nom du fichier pcap à l'option -r en ligne de commande, et Snort le traitera en conséquence :

```
$ snort -r get.pcap
```

Si l'opération réussit, Snort affichera des informations de base sur le fichier pcap lu telles que le nombre de paquets et les protocoles détectés (voir figure 3.2).

FIGURE 3.2 – Lecture d'un fichier pcap avec Snort.

Les utilisateurs peuvent également exécuter Snort sur un répertoire entier de fichiers pcap avec l'option `--pcap-dir`. Si ce répertoire contient d'autres fichiers que des pcap, l'option `--pcap-filter` permet de spécifier à Snort quels fichiers traiter. Par exemple :

```
$ snort --pcap-dir /chemin/vers/repertoire/pcap --pcap-filter '*.pcap'
```

— **Trafic des interfaces :**
Snort peut également écouter sur des interfaces réseau actives, et cette configuration se fait avec l'option -i suivie des noms des interfaces à utiliser. Par exemple, la commande suivante exécute Snort sur l'interface réseau eth0 :

```
$ sudo snort -i eth0
```

— **Modes opératoires :**
Avec certains modules DAQ, Snort peut utiliser deux modes de fonctionnement différents : passif et actif. Le mode passif permet à Snort d'observer et de détecter le trafic sur une interface réseau, mais il empêche le blocage direct du trafic. Le mode actif, en revanche, permet à Snort de bloquer le trafic si un paquet particulier le justifie.

Snort déduit le mode de fonctionnement particulier en fonction des options utilisées en ligne de commande. Par exemple, la lecture à partir d'un fichier pcap avec l'option -r ou l'écoute sur une interface avec l'option -i fera que Snort fonctionnera en mode passif par défaut. Si le DAQ prend en charge le mode actif, les utilisateurs peuvent spécifier l'option -Q pour l'exécuter avec ce mode.

3.2.4 Configuration

Une fois Snort configuré pour traiter le trafic, il est temps de lui indiquer comment le faire, et cela se fait par le biais de la configuration. La configuration de Snort gère des éléments tels que la définition des variables globales, les différents modules à activer ou désactiver, les paramètres de performance, les politiques de journalisation des événements et les chemins vers les fichiers de règles spécifiques à activer.

La configuration de Snort est désormais entièrement réalisée en Lua, et ces options de configuration peuvent être fournies à Snort de trois manières différentes : via la ligne de commande, avec un seul fichier de configuration Lua, ou avec plusieurs fichiers de configuration Lua.

3.2.4.1 Fichiers de configuration

Il existe de nombreuses options de configuration différentes pouvant être ajustées dans Snort, mais heureusement, la version open source de Snort 3 est livrée avec un ensemble de fichiers de configuration standard qui aident les utilisateurs à démarrer rapidement. Ces fichiers par défaut sont situés dans le répertoire `lua/`, et les fichiers `snort.lua` et `snort_defaults.lua` qui s'y trouvent constituent ce que l'on considère comme la configuration de base. Cette configuration par défaut est un excellent modèle sur lequel s'appuyer, et elle peut être directement utilisée dans Snort.

Par défaut, Snort ne recherche pas un fichier de configuration spécifique, mais on peut facilement lui en fournir un en utilisant l'argument `-c` :

```
$ snort -c $my_path/snort.lua
```

Cette commande valide simplement le fichier de configuration fourni, et si tout est en ordre, le message de sortie inclura "Snort a validé la configuration avec succès" (figure 3.3).

```
Snort successfully validated the configuration (with 0 warnings).
o")~   Snort exiting
```

FIGURE 3.3 – Importation des fichiers de config Snort.

3.2.5 Les modes opératoires

Au cœur de Snort se trouve un système de détection d'intrusions (IDS) et un système de prévention d'intrusions (IPS), ce qui signifie qu'il a la capacité de détecter les intrusions sur un réseau et également de les prévenir. Une configuration indique à Snort comment traiter le trafic réseau. Ce sont les règles qui déterminent si Snort doit agir sur un paquet particulier.

Les règles de Snort peuvent être directement placées dans le(s) fichier(s) de configuration Lua via le module ips, mais la plupart du temps, elles résideront dans des fichiers distincts `.rules` qui sont "inclus". Par exemple, supposons que nous ayons un fichier `malware.rules` dans le même répertoire que notre fichier de configuration Lua. Nous pourrions "inclure" ce fichier de règles de la manière suivante :

```
ips = { include = 'malware.rules' }
```

Si on veut inclure plusieurs fichiers `.rules`, on peut le faire comme ceci :

```
    ips =
{
    rules = [[
        include /chemin/vers/fichier/rulesfile1.rules
        include /chemin/vers/fichier/rulesfile2.rules
        ...
    ]]
}
```

Alternativement, un seul fichier de règles ou un chemin vers un répertoire de règles peut être passé directement à Snort en ligne de commande. Cela se fait soit avec l'option `-R` pour un seul fichier de règles, soit avec l'option `--rule-path` pour spécifier un répertoire entier de fichiers de règles. C'est pratique lorsqu'on veut vérifier ou dépanner une règle ou des règles par rapport à un pcap.
Par exemple, la commande ci-dessous exécutera toutes les règles présentes dans `local.rules` sur le trafic dans `get.pcap` :

```
$ snort -c $my_path/snort.lua -R local.rules -r get.pcap
```

3.2.5.1 Génération des fichiers log (Mode sniffer) :

Snort peut être utilisé comme analyseur de paquets, permettant la capture du trafic réseau transitant par une interface spécifique et l'enregistrement de ce trafic dans des fichiers journaux. La configuration de Snort pour ce mode s'effectue via la ligne de commande en utilisant le paramètre -A, spécifiant le format du fichier journal, ou en modifiant le fichier de configuration pour activer une fonction de sortie appropriée. Quelques modes de journalisation disponibles : `none`, `dump`, `pcap`, et `log\_*`.

```
$ snort -c /mypath/snort.lua -i eth0 -L pcap
```

La commande ci-dessus va exécuter snort avec le fichier de configuration `snort.lua`, écouter le trafic qui transite par l'interface `eth0` et exporter le trafic capturé dans un fichier log sous format `pcap`.

3.2.5.2 Génération des alertes (Mode passif) :

L'inspection d'un fichier log par défaut va afficher diverses statistiques sur l'exécution particulière. Cela inclut des détails sur les applications identifiées, les événements de détection, les types de services détectés, et bien plus encore. Les événements de détection montreront combien d'alertes ont été déclenchées sur le trafic fourni, mais parfois nous voulons en savoir plus que cela.

Snort propose quelques options différentes de "mode d'alerte" qui peuvent être définies en ligne de commande pour ajuster la façon dont les alertes sont affichées. Ces modes incluent `cmg` qui affiche les alertes aux côtés d'un hexdump des paquets alertants, ainsi que quelques modes alert_* différents indiqués ci-dessous :

```
$ snort --help-modules | grep alert
alert_csv (logger): output event in csv format
alert_fast (logger): output event with brief text format
alert_full (logger): output event with full packet dump
alert_json (logger): output event in json format
alert_syslog (logger): output event to syslog
alert_talos (logger): output event in Talos alert format
alert_unixsock (logger): output event over unix socket
alerts (basic): configure alerts
```

Ces modes sont définis avec l'option -A suivie du mode d'alerte désiré, et nous pouvons

nous concentrer uniquement sur les alertes en incluant également le drapeau -q (silencieux). Le mode d'alerte cmg, par exemple, ressemblera à quelque chose comme ceci (figure 3.4) :

```
$ snort -c $my_path/snort.lua -q -r get.pcap -R local.rules -A cmg
```

`local.rules` contient une seule règle qui détecte les paquets avec le protocole ICMP (ping).

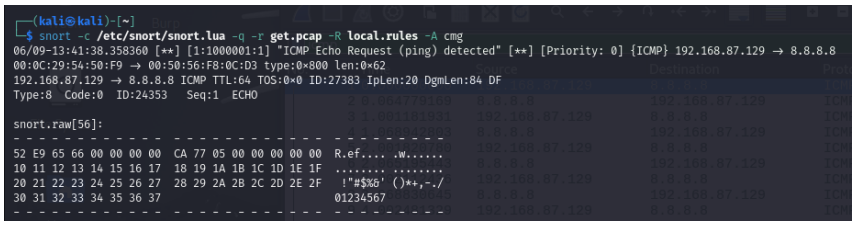

FIGURE 3.4 – Les alertes de snort avec le mode cmg.

Le mode alert_talos est un autre mode utile qui affiche les alertes dans un format simple et facile à comprendre.

```
$ snort -c $my_path/lua/snort.lua -q -r get.pcap -R local.rules -A alert_talos
```

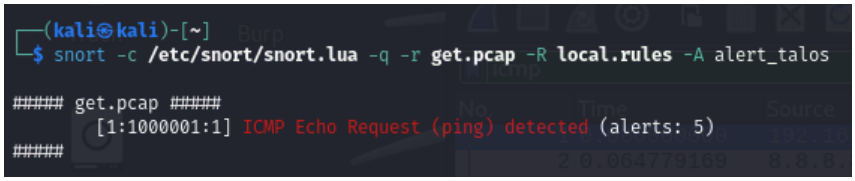

FIGURE 3.5 – Les alertes de snort avec le mode talos.

Enfin, certains des modes alert_* sont personnalisables. Par exemple, alert_csv permet la personnalisation des différents "champs" pouvant être affichés. L'exemple suivant montre une configuration d'alerte CSV personnalisée en utilisant le drapeau de ligne de commande –lua :

```
$ snort -c $my_path/snort.lua -r get.pcap -R local.rules
--lua 'alert_csv = { fields = "action pkt_num gid sid rev msg service
src_addr src_port dst_addr dst_port", separator = "," }'
```

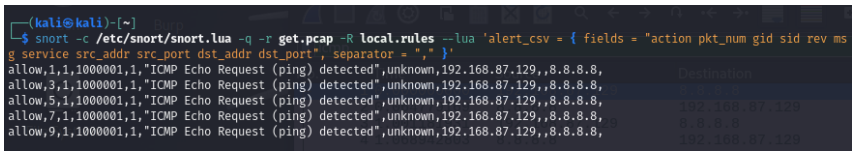

FIGURE 3.6 – Les alertes de snort avec le mode alert_csv.

3.2.5.3 Mode prévention

Pour protéger les réseaux, il est également important de s'assurer que nos règles bloquent les attaques de manière appropriée, et le dump DAQ nous permet de le faire.

En spécifiant l'option -Q pour activer le mode actif, puis en définissant `--daq` sur dump, nous allons "dumper" le trafic qui aurait été passé à travers, émulant ainsi un fonctionnement actif en temps réel. Le trafic résultant sera, par défaut, sauvegardé dans un fichier nommé inline-out.pcap :

```
$ snort -Q --daq dump -q -r get.pcap -R local.rules
```

Dans l'exemple ci-dessus, si le fichier `local.rules` contient une règle de blocage qui se déclenche sur certains trafics dans le fichier `get.pcap`, alors le fichier `inline-out.pcap` résultant contiendra uniquement le trafic qui n'a pas été bloqué. Nous pouvons utiliser cette fonctionnalité pour tester si nos règles empêchent effectivement les paquets d'attaque réels de passer.

Comme il est possible de le constater, bien que Snort se distingue par sa flexibilité et ses divers modes opératoires, il repose largement sur des règles prédéfinies lors de sa configuration pour assurer la détection et la prévention. En l'absence de règles adéquates face à une intrusion, il peut en résulter un scénario où l'attaque passe inaperçue, exposant ainsi le système à d'éventuels dommages. Pour cette raison, cette méthode est progressivement supplantée par des approches plus innovantes qui pallient ses limites connues, notamment les systèmes de détection d'intrusions (IDS) basés sur l'apprentissage automatique pour la détection des anomalies.

3.3 Conception d'un IDS basé sur le ML

Cette section décrit la conception d'un système de détection d'intrusion utilisant le machine learning et le deep learning sur le dataset CSE-CIC-IDS 2018, comprenant diverses attaques. Elle couvre le prétraitement des données, l'utilisation de la méthode SelectKBest

pour sélectionner les features importantes, et la préparation des données pour différents algorithmes (Random Forest, SVM, KNN, MLP). La partition du dataset en ensembles d'entraînement, de validation et de test est également abordée pour garantir une évaluation rigoureuse des modèles.

3.3.1 CSE-CIC-IDS 2018 dataset

Le CSE-CIC-IDS 2018 dataset a été élaboré par le Canadian Institute for Cybersecurity (CIC) en collaboration avec l'Université du Nouveau-Brunswick. Ce dataset comprend sept scénarios d'attaque différents : la force brute, Heartbleed, le botnet, le déni de service (DoS), le déni de service distribué (DDoS), les attaques Web et l'infiltration du réseau de l'intérieur. L'infrastructure attaquante comprend 50 machines, tandis que l'organisation victime compte 5 départements avec un total de 420 machines et 30 serveurs. Le dataset inclut les captures du trafic réseau et les journaux système de chaque machine, avec 80 features extraites du trafic capturé à l'aide de CICFlowMeter-V3. [49]

• **Types d'attaque :**

— **Force brute :** Tentatives répétées de deviner des mots de passe ou des identifiants de connexion.
— **Botnet :** Réseau de machines compromises utilisées pour effectuer des attaques coordonnées.
— **DoS :** Attaques visant à rendre un service indisponible en surchargeant un système.
— **DDoS :** Version distribuée du DoS, utilisant plusieurs machines pour lancer une attaque simultanée.
— **Attaques Web :** Comprennent des techniques comme les injections SQL et les scripts intersites (XSS).
— **Infiltration :** Compromission de machines internes pour espionner ou exfiltrer des données.

Le dataset est composé de plusieurs fichiers comme indiqué dans la table 3.2 :

Nom de fichier	**Taille**	**Attaques**
Bruteforce-Wednesday-14-02-2018.csv	341 Mo	Bruteforce
Botnet-Friday-02-03-2018.csv	336 Mo	Botnet
DDoS1-Tuesday-20-02-2018.csv	366 Mo	DDoS
DDoS2-Wednesday-21-02-2018.csv	316 Mo	DDoS
DoS1-Thursday-15-02-2018.csv	358 Mo	DoS
DoS2-Friday-16-02-2018.csv	318 Mo	DoS
Infil1-Wednesday-28-02-2018.csv	199 Mo	Infiltarion
Infil2-Thursday-01-03-2018.csv	102 Mo	Infiltartion
Web1-Thursday-22-02-2018.csv	364 Mo	Attaques Web
Web2-Friday-23-02-2018.csv	365 Mo	Attaques Web

TABLE 3.2 – Les scenarios d'attaques dans CSE-CIC-IDS 2018 dataset.

Le dataset contient 9332770 instances et 80 features avec 15 classes uniques (1 normale + 14 attaques). Les caractéristiques du dataset combiné et la répartition détaillée par classe sont présentées dans la table 3.3 et la table 3.4.

Nom de dataset	CIC-IDS2018
Type de dataset	Multi-classe
Date de réalisation	2018
Nombre totale d'instances	9332770
Nombre de features	80
Nombre de classes uniques	15

TABLE 3.3 – Caractéristiques globales du dataset CIC-IDS2018.

Les classes	**Instances**
Benign	6584535
DDoS attack-HOIC	686012
DDoS attacks-LOIC-HTTP	576191
DoS attacks-Hulk	461912
Bot	286191
FTP-BruteForce	193360
SSH-Bruteforce	187589
Infilteration	161934
DoS attacks-SlowHTTPTest	139890
DoS attacks-GoldenEye	41508
DoS attacks-Slowloris	10990
DDOS attack-LOIC-UDP	1730
Brute Force -Web	611
Brute Force -XSS	230
SQL Injection	87

TABLE 3.4 – les instances des classes uniques du dataset CIC-IDS2018.

Features	**Dtype**	**Features**	**Dtype**
Flow ID	object	Bwd Pkts/s	float64
Src IP	object	Pkt Len Min	float64
Src Port	int64	Pkt Len Max	float64
Dst IP	object	Pkt Len Mean	float64
Dst Port	int64	Pkt Len Std	float64
Protocol	int64	Pkt Len Var	float64
Timestamp	object	FIN Flag Cnt	int64
Flow Duration	int64	SYN Flag Cnt	int64
Tot Fwd Pkts	int64	RST Flag Cnt	int64
Tot Bwd Pkts	int64	PSH Flag Cnt	int64
TotLen Fwd Pkts	int64	ACK Flag Cnt	int64
TotLen Bwd Pkts	int64	URG Flag Cnt	int64
Fwd Pkt Len Max	int64	CWE Flag Cnt	int64
Fwd Pkt Len Min	int64	ECE Flag Cnt	int64
Fwd Pkt Len Mean	float64	Down/Up Ratio	float64
Fwd Pkt Len Std	float64	Pkt Size Avg	float64
Bwd Pkt Len Max	int64	Fwd Seg Size Avg	float64
Bwd Pkt Len Min	int64	Bwd Seg Size Avg	float64
Bwd Pkt Len Mean	float64	Fwd Byts/b Avg	float64
Bwd Pkt Len Std	float64	Fwd Pkts/b Avg	float64
Flow Byts/s	float64	Bwd Byts/b Avg	float64
Flow Pkts/s	float64	Bwd Pkts/b Avg	float64
Flow IAT Mean	float64	Fwd Blk Rate Avg	float64
Flow IAT Std	float64	Bwd Blk Rate Avg	float64
Flow IAT Max	float64	Subflow Fwd Pkts	int64
Flow IAT Min	float64	Subflow Fwd Byts	int64
Fwd IAT Tot	float64	Subflow Bwd Pkts	int64
Fwd IAT Mean	float64	Subflow Bwd Byts	int64
Fwd IAT Std	float64	Init Fwd Win Byts	int64
Fwd IAT Max	float64	Init Bwd Win Byts	int64
Fwd IAT Min	float64	Fwd Act Data Pkts	int64
Bwd IAT Tot	float64	Fwd Seg Size Min	float64
Bwd IAT Mean	float64	Active Mean	float64
Bwd IAT Std	float64	Active Std	float64
Bwd IAT Max	float64	Active Max	float64
Bwd IAT Min	float64	Active Min	float64
Fwd PSH Flags	int64	Idle Min	float64
Bwd PSH Flags	int64	Idle Max	float64
Fwd URG Flags	int64	Idle Mean	float64
Bwd URG Flags	int64	Idle Std	float64
Fwd Header Len	int64	Fwd Pkts/s	float64
Bwd Header Len	int64	Label	object

TABLE 3.5 – Les 80 features et leurs types.

3.3.2 Flux de travail

Le résumé de notre travail est représenté dans l'organigramme suivant :

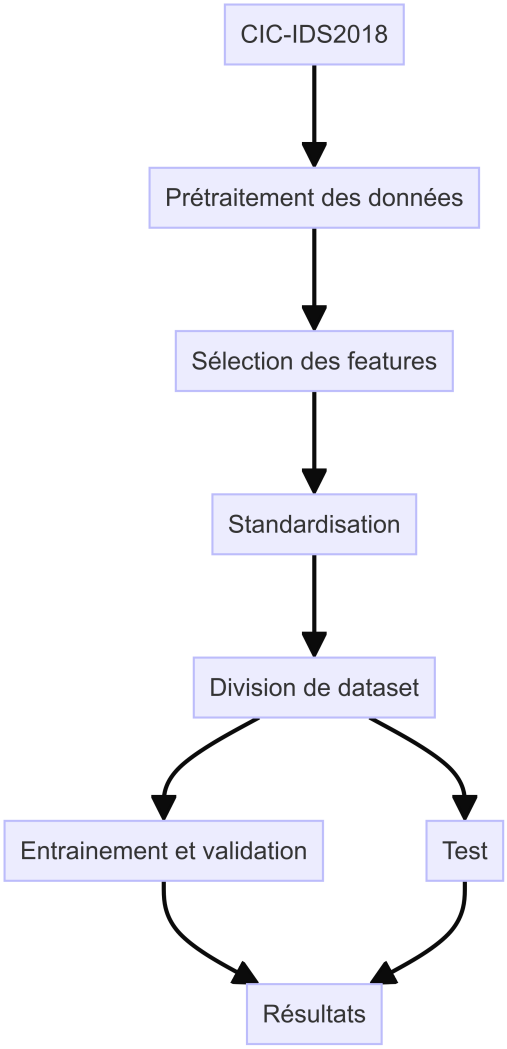

FIGURE 3.7 – Organigramme représentant les étapes de conception.

3.3.3 Prétraitement des données

Les algorithmes de machine learning apprennent à partir des données qui leur sont fournies. Par conséquent, si ces données sont de mauvaise qualité, c'est-à-dire incorrectes, corrompues, mal formatées, dupliquées ou incomplètes, l'algorithme résultant sera lui-même de mauvaise qualité, puisqu'il est censé refléter ce qu'il observe dans les données. Pour cette raison, il est impératif de pré-traiter nos données avant de les fournir au modèle d'apprentissage, cette phase étant connue sous le nom de prétraitement des données.

Nous avons constaté que les données du dataset CIC-IDS2018 sont divisées en dix fichiers

différents. Étant donné que traiter chaque fichier individuellement est une tâche fastidieuse, nous avons décidé de faciliter le traitement de nos données en fusionnant tous les fichiers en un seul fichier CSV appelé "AIO.csv", qui contient un total de 9332770 instances. La distribution des classes dans le dataset CIC-IDS2018 est illustrée dans les figures 3.8 et 3.9.

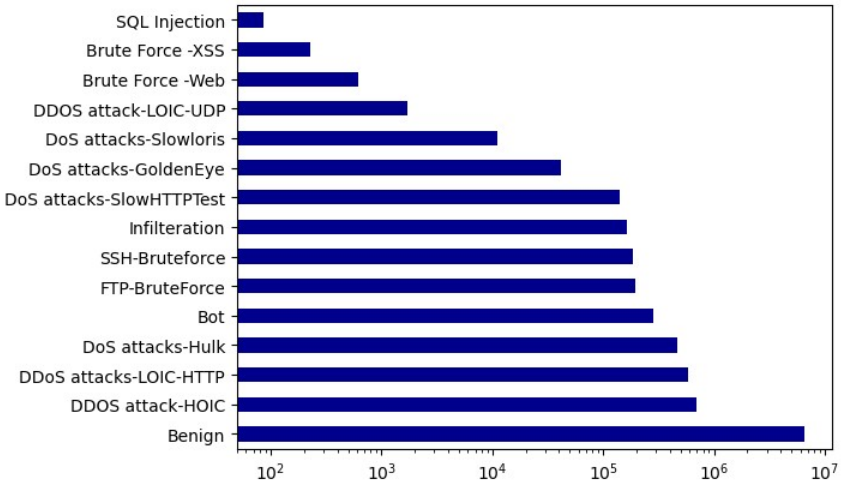

FIGURE 3.8 – Distribution des classes dans le dataset.

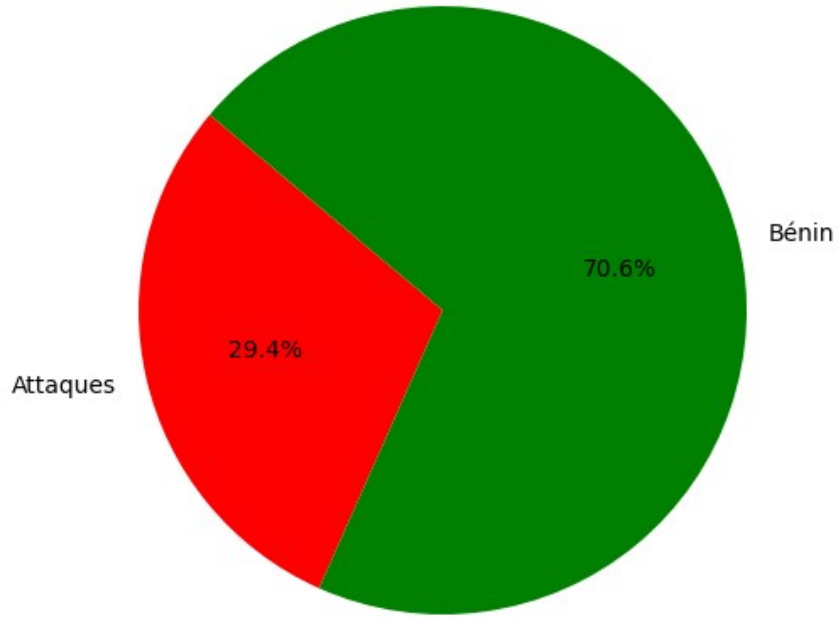

FIGURE 3.9 – Diagramme circulaire de distribution des classes.

3.3.3.1 Nettoyage des données

Nous commencerons par supprimer les valeurs infinies (inf) et NaN en les remplaçant par des 0. Ensuite, nous éliminerons les enregistrements redondants. Une fois cette étape

terminée, nous sélectionnerons les colonnes contenant uniquement des 0 et les supprimerons, car elles n'apportent aucune variance et seraient considérées comme du bruit pour l'apprentissage du modèle.

Étant donné que nos modèles ne peuvent apprendre qu'à partir de valeurs numériques, nous éliminerons les colonnes qui ne contiennent pas de valeurs numériques, comme la colonne "Timestamp". Cette colonne, qui enregistre le moment où les données ont été capturées, est de type objet et ne sera pas utile pour la classification. Voici un résumé des étapes :

- Remplacement des valeurs infinies et NaN par des 0.
- Suppression des enregistrements redondants.
- Suppression des colonnes non numériques, telles que "Timestamp".

Ces étapes garantiront que notre ensemble de données est propre et prêt pour l'entraînement du modèle de classification binaire.

3.3.3.2 Encodage des données

Comme mentionné précédemment, notre jeu de données contient plusieurs classes dans la colonne "Label". Étant donné que les différences d'instances entre les types d'attaques sont importantes, ça peut nuire aux performances de nos modèles. En effet, lorsque les modèles de machine learning sont confrontés à une grande variabilité entre les instances de chaque classe, ils peuvent avoir des difficultés à généraliser correctement. Cette hétérogénéité peut entraîner une mauvaise classification des instances, surtout si certaines classes sont sous-représentées dans les données d'entraînement.

Nous allons regrouper ces classes (1 classe normale et 14 classes d'attaques) en deux catégories, 0 pour le trafic normal et 1 pour le trafic malveillant.
Pour ce faire, nous appliquerons une fonction sur l'ensemble de données, qui remplacera toutes les classes d'attaques par 1 et la classe bénigne par 0. Cette transformation permettra de simplifier le problème de classification et de faciliter l'entraînement du modèle.

3.3.3.3 Échantillonage et partitionnement

Pour l'entraînement des modèles, nous utilisons un échantillon représentant 10% du dataset, qui se compose de 9 millions d'instances. Cette approche est justifiée par plusieurs raisons pratiques. Premièrement, traiter l'intégralité du dataset nécessiterait des ressources matérielles considérables, telles que la mémoire vive et la puissance de calcul, qui peuvent ne pas être disponibles.

De plus, l'utilisation d'un échantillon plus petit permet de réduire significativement le temps de traitement et d'entraînement des modèles, tout en restant représentatif de l'ensemble des données. Nous avons également veillé à préserver la proportion des attaques et du trafic bénin, soit 30% d'attaques et 70% de trafic bénin, afin de garantir que l'échantillon reste équilibré et représentatif des conditions réelles. Cela nous permet d'itérer plus rapidement sur les modèles et d'effectuer des ajustements plus efficients, tout en préservant la qualité et la pertinence des résultats.

3.3.3.4 Sélection des features

Avant de passer à d'autres manipulations telles que la normalisation et la standardisation de toutes les données, y compris toutes les colonnes, nous avons remarqué qu'il est judicieux de sélectionner les caractéristiques les plus importantes pour l'entraînement et le test des modèles à ce stade. Cela permet de réduire le temps de traitement des données et d'éviter l'utilisation de caractéristiques qui pourraient ne pas être pertinentes pour les étapes ultérieures. La sélection des caractéristiques est une étape cruciale qui influe directement sur les performances de nos modèles. Pour cela, nous avons choisi d'utiliser une classe du module "feature_selection" de la bibliothèque "scikit-learn" appelée **SelectKBest**.

SelectKBest est un type de méthode de sélection de caractéristiques basée sur des filtres. Dans les méthodes de sélection de caractéristiques basées sur des filtres, le processus de sélection des caractéristiques est effectué indépendamment de tout algorithme d'apprentissage automatique spécifique. Au lieu de cela, il repose sur des mesures statistiques pour évaluer et classer les caractéristiques.

Cette classe utilise des tests statistiques comme le test du chi-carré, le test (ANOVA) ou la mesure d'information mutuelle pour évaluer et classer les caractéristiques en fonction de leur relation avec la variable de sortie. Ensuite, il sélectionne les k caractéristiques ayant les scores les plus élevés pour être incluses dans le sous-ensemble final de caractéristiques. **SelectKBest** prend deux paramètres en entrée : fonction de score et k. [15]

- k est un nombre qui représente les meilleures caractéristiques à sélectionner.
- Fonction de score est utilisée pour évaluer l'importance des caractéristiques. Il existe plusieurs fonctions de score, certaines destinées aux problèmes de classification et d'autres aux problèmes de régression.

Pour notre part, nous avons utilisé k=10 pour sélectionner les 10 meilleures caractéristiques et la fonction de score **mutual_info_classif**. Cette méthode est destinée aux problèmes de classification. Elle mesure la dépendance entre chaque caractéristique et la variable cible en utilisant l'information mutuelle, capturant les relations non linéaires grâce à des techniques basées sur les k-plus-proches voisins. Une valeur élevée indique

une forte dépendance, aidant à identifier les caractéristiques les plus informatives pour la classification.
Voici la distribution des poids des caractéristiques :

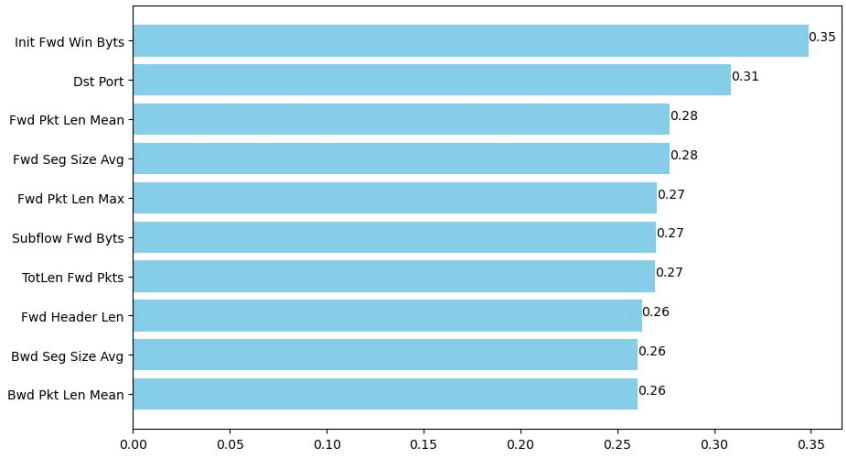

FIGURE 3.10 – Distribution des poids sur les caractéristiques.

Une fois les 10 caractéristiques obtenues, on effectue une analyse de corrélation avec la méthode de Pearson pour éliminer les features qui ont des corrélations dépassant un seuil définit sur 0.9. Les caratéristiques finales obtenues après toute la procédure sont les suivantes :

- **Dst Port (Destination Port) :** Identifie un service réseau spécifique sur la machine de destination.
- **TotLen Fwd Pkts (Total Length of Forward Packets) :** Longueur totale des paquets envoyés vers le destinataire.
- **Fwd Pkt Len Max (Forward Packet Length Maximum) :** Longueur maximale d'un paquet envoyé vers le destinataire.
- **Bwd Pkt Len Mean (Backward Packet Length Mean) :** Longueur moyenne des paquets renvoyés par le destinataire.
- **Init Fwd Win Byts (Initial Forward Window Bytes) :** Taille initiale de la fenêtre d'envoi en octets de l'expéditeur.

Caractéristique	Poids
Init Fwd Win Byts	0.35
Dst Port	0.31
TotLen Fwd Pkts	0.27
Fwd Pkt Len Max	0.27
Bwd Pkt Len Mean	0.26

TABLE 3.6 – Features sélectionnées et leurs poids.

3.3.3.5 Transformation des données

Nous allons entraîner plusieurs algorithmes qui ont chacun une manière de fonctionnement différente de l'autre, cette différence impose des changements sur les données. Des algorithmes comme **Random Forest (RF)** et **Decision Tree (DT)** n'ont généralement pas besoin de standardisation ni de normalisation. Ces algorithmes basés sur les arbres sont invariants par rapport aux changements d'échelle des caractéristiques car ils segmentent les données en fonction des valeurs de seuils, indépendamment de la distribution des caractéristiques.

Support Vector Machine (SVM) nécessite une standardisation. SVM est sensible à l'échelle des caractéristiques car il cherche à maximiser la marge entre les classes, ce qui implique des calculs basés sur les distances. Si les caractéristiques ont des échelles différentes, cela peut entraîner des marges incorrectes. La normalisation peut aussi être utile, mais la standardisation est plus courante.

K-Nearest Neighbors (KNN) nécessitent une standardisation ou une normalisation. KNN est basé sur la distance (souvent la distance euclidienne) entre les points de données. Si les caractéristiques ont des échelles différentes, celles avec des valeurs plus grandes domineront le calcul des distances. La standardisation ou la normalisation permet de mettre toutes les caractéristiques sur un pied d'égalité.

Multi-Layer Perceptron (MLP) nécessite une standardisation ou une normalisation des données pour améliorer l'apprentissage. Cela évite les gradients explosifs ou vanissants causés par des fonctions d'activation sensibles à l'échelle des données. De plus, cela stabilise et accélère la mise à jour des poids pendant l'entraînement, et empêche certaines caractéristiques de dominer les autres, assurant une meilleure généralisation du modèle.

3.3.4 Implémentation des modèles ML

En apprentissage automatique, il n'est jamais approprié d'évaluer la performance d'un modèle sur les mêmes données qui ont été utilisées pour son entraînement. Le dataset CIC-IDS2018 contient un seul ensemble de données non séparé en jeux d'entraînement et de test dédiés. Par conséquent, les données doivent être divisées en deux parties : un ensemble d'entraînement, dont les données sont utilisées pour entraîner le modèle, et un ensemble de test, réservé uniquement à l'évaluation de la performance du modèle. La partition généralement préférée est de 20% de données de test et 80% de données d'entraînement. Pour ce faire, nous utilisons la fonction 'train_test_split()' du module 'model_selection' de Sklearn. Cette fonction mélange aléatoirement notre dataset avant de le diviser en deux parties. (Figure 3.11)

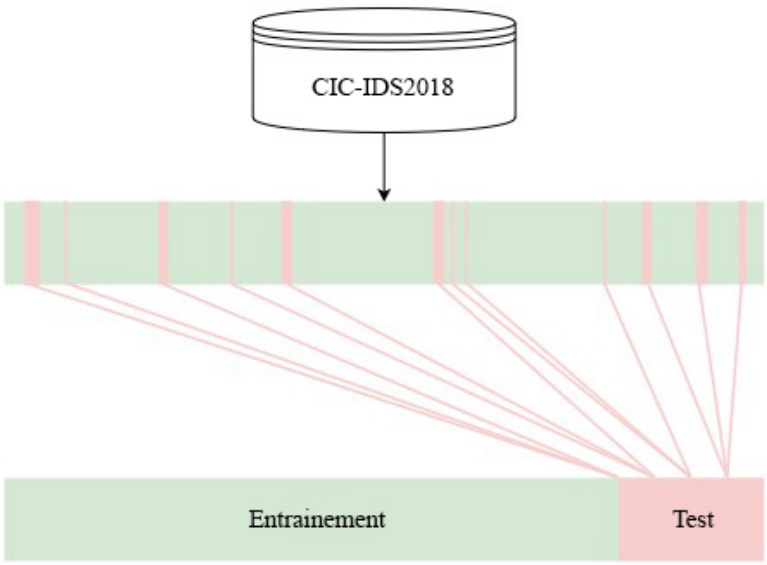

FIGURE 3.11 – Division de dataset en deux parties.

3.3.4.1 Hyper-paramètres des modèles

Pour garantir la robustesse de nos modèles, nous avons adopté une approche prudente dans la sélection des hyperparamètres. Dans un premier temps, pour les modèles Decision Tree, Random Forest et KNN, nous avons opté pour l'utilisation des paramètres par défaut. Cette décision découle de notre volonté de maintenir une base solide avant d'explorer des ajustements plus spécifiques.

En revanche, pour les modèles SVM, nous avons suivi une démarche guidée par des recommandations de sources spécialisées pour le choix des hyperparamètres. Nous avons entraîné plusieurs modèles en variant les fonctions de noyau, notamment polynomiale, gaussienne, fonction de base radiale (RBF) et noyau sigmoïde. La sélection des valeurs de C, paramètre de régularisation, et de gamma, un paramètre crucial pour les noyaux non linéaires, a été informée par ces recommandations. Ainsi, gamma a été fixé à l'inverse du nombre de caractéristiques, tandis que C a été fait varier entre 0,1 et 2 selon les suggestions de la source. Le modèle qui a émergé comme le plus performant était celui avec un noyau 'rbf', un paramètre de régularisation C de 2 et une valeur gamma automatique ('auto') [18]. Cette approche méthodique, basée sur des sources fiables, a permis d'obtenir des résultats optimaux en tirant parti de la flexibilité du modèle SVM.

Quant au modèle MLP, une série de classificateurs a été entraînée avec diverses configurations. Ces configurations ont inclus différentes fonctions d'activation telles que logistique, tangente hyperbolique et unité linéaire rectifiée, ainsi que des solveurs comme lbfgs, sgd et adam. De plus, nous avons exploré différentes architectures de couches cachées avec un nombre maximal de nœuds cachés de 15. Suite à cette exploration, la configuration

optimale s'est avérée être celle du MLPClassifier avec un solveur 'adam', un alpha de 1e-5, des couches cachées de dimensions (5, 2), une fonction d'activation tangente hyperbolique ('tanh') et un état aléatoire de 1. Cette approche exhaustive a permis de sélectionner les paramètres les plus adaptés à notre modèle MLP, garantissant ainsi des performances optimales dans nos expériences de classification.

Classificateur	Hyper-paramètres
DT	RandomForestClassifier(random_state=0)
RF	DecisionTreeClassifier(random_state=0)
KNN	KNeighborsClassifier()
SVM	SVC(kernel = 'rbf', C = 2, gamma = 'auto')
MLP	MLPClassifier(solver='adam', alpha=1e-5, hidden_layer_sizes=(5, 2), activation='tanh',random_state=1)

TABLE 3.7 – Hyper-paramètres des classificateurs.

3.4 Conclusion

En conclusion, Snort illustre une méthode classique et éprouvée pour surveiller et protéger les réseaux contre les intrusions. Snort, avec sa large adoption et sa flexibilité, reste un outil de référence dans la détection des intrusions réseau.

La seconde approche explore les avancées modernes en apprentissage automatique pour entraîner un modèle sur un dataset basé sur les flux réseau. Cette méthode, bien que plus complexe, promet une détection plus sophistiquée des anomalies et une capacité améliorée à identifier des menaces émergentes. Ces deux approches, bien que différentes, convergent vers un même objectif : sécuriser nos infrastructures réseau contre des attaques de plus en plus sophistiquées. En combinant des techniques traditionnelles et modernes, nous sommes mieux équipés pour répondre aux défis de la cybersécurité actuelle et future.

Dans le chapitre qui suit, nous allons tester les modèles sélectionnes sur une partie de dataset et discuter des résultats d'évaluation obtenus.

Chapitre 4

Tests et résultats

4.1 Introduction

Les performances des systèmes de détection d'intrusions basés sur l'apprentissage automatique dépendent fortement du dataset utilisé pour le développement du modèle. Dans ce travail, nous avons utilisé CIC-IDS2018, un dataset fourni par l'Institut Canadien de Cybersécurité (CIC), pour détecter et classifier un large éventail d'attaques telles que les attaques par déni de service, le PortScan, les attaques Web et de nombreuses autres attaques répandues. Nous avons implémenté 4 algorithmes d'apprentissage automatique (Random Forest (RF), Decision Tree (DT), Support Vector Machine (SVM) et K-Nearest Neighbors (KNN)) et un algorithme d'apprentissage profond (Multi-Layer Perceptron (MLP)) et comparé leurs performances en utilisant diverses mesures d'évaluation (Accuracy, Precision, Recall, F1 score).

4.2 Environnement d'exécution

L'apprentissage automatique est un domaine qui nécessite la disponibilité de ressources matérielles (en particulier les GPU) capables d'effectuer des calculs intensifs. Pour réaliser notre travail, nous avons choisi de travailler dans un environnement expérimental avec les caractéristiques suivantes :

- **OS :** Windows 10 64 bits.
- **CPU :** Intel Core i7-4770 @3.40GHz.
- **RAM :** 8GB.
- **SSD :** 512GB.

Du côté logiciel, nous avons commencé par configurer un environnement de développement Python local en utilisant la plateforme **Anaconda** [17], qui est une distribution des langages de programmation Python et R pour le calcul scientifique, visant à simplifier la gestion et le déploiement des packages.

Pour le langage de programmation, nous avons choisi **Python (3.12)** [43], un langage de programmation interprété, open source et de haut niveau, qui offre une excellente approche de la programmation orientée objet. C'est le langage le plus courant et le plus populaire pour l'apprentissage automatique, utilisé par les data scientists pour divers projets et applications de science des données, grâce à sa flexibilité et au grand nombre de bibliothèques logicielles open source qu'il prend en charge, telles que **Numpy**, **Pandas**, **Matplotlib**, **Scikit-Learn**, etc.

Nous avons également utilisé l'environnement de développement **Jupyter Notebook**

(6.6.3) [33] qui est un environnement de développement interactif open source qui permet de créer et de partager des documents combinant du code, des visualisations et du texte pour écrire et exécuter des codes Python.

Librairie	Description	Version
Numpy [39]	Une bibliothèque Python qui fournit des fonctions mathématiques pour manipuler des tableaux de grandes dimensions. Elle offre diverses méthodes/fonctions pour les tableaux, les métriques et l'algèbre linéaire.	1.26.4
Pandas [42]	Un outil d'analyse et de manipulation de données rapide, puissant, flexible et facile à utiliser, open source et basé sur le langage de programmation Python.	2.1.4
Scikit-learn [45]	C'est une bibliothèque de machine learning gratuite pour le langage de programmation Python. Elle propose divers algorithmes de classification, régression et regroupement.	1.22
Matplotlib [36]	Matplotlib est une bibliothèque complète pour créer des visualisations statiques, animées et interactives en Python.	3.8.0

TABLE 4.1 – Les librairies python utilisées.

4.3 Critères d'évaluation

Cette section expose les critères primordiaux qui ont été employés pour réaliser une évaluation plus authentique des systèmes de détection d'intrusion.

— **Erreurs de classification :**
Lorsqu'un IDS procède à la classification des données, sa décision peut être exacte ou incorrecte. Il existe différents types d'erreurs émanant du détecteur, lesquelles affectent sa capacité de manière variable.

- **Vrais Positifs (TP) :** se produisent lorsqu'un IDS classifie correctement une intrusion.
- **Vrai Négatifs (TN) :** se produisent lorsqu'un événement normal est correctement classé comme une action légitime.
- **Faux positifs (FP) :** se produisent lorsque le système reconnaît à tort que des actions légitimes sont une intrusion.

- **Faux négatifs (FN) :** se produisent lorsqu'un IDS classifie à tort des intrusions comme des actions légitimes.

Par conséquent, l'objectif d'un IDS est de produire autant de TP et TN que possible, tout en essayant de réduire le nombre de FP et FN.

— **Exactitude :**
L'exactitude (Accuracy) représente la mesure de la fiabilité d'un IDS, évaluant le pourcentage de détections réussies et échouées ainsi que le nombre de fausses alertes générées par le système. Un système affichant une précision de 80% signifie qu'il classe correctement 80 instances sur 100 dans leur classe réelle. La précision est calculée comme suit :

$$\text{Accuracy} = \frac{TP + TN}{TP + FP + TN + FN} \times 100$$

— **Précision :**
Précision (Precision) s'agit d'une métrique indiquant combien d'événements, qui sont prédits par un IDS comme étant intrusifs, sont des intrusions réelles. L'objectif d'un IDS est d'obtenir une précision élevée, ce qui signifie que le nombre de fausses alarmes est minimisé. La précision est calculée comme suit :

$$\text{Precision} = \frac{TP}{TP + FP} \times 100$$

— **Rappel :**
Le rappel (Recall) évalue ce que la précision ne capture pas : combien d'événements normaux sont correctement identifiés par le système. Ainsi, un rappel élevé est souhaitable pour un système. Le calcul du rappel se fait de la manière suivante :

$$\text{Recall} = \frac{TP}{TP + FN} \times 100$$

— **F1 score :**
Le F1 score est une mesure qui combine à la fois la précision et le rappel d'un modèle de classification en une seule valeur. Il est utile lorsque les classes sont déséquilibrées, c'est-à-dire lorsque certaines classes ont beaucoup plus d'exemples que d'autres. Le F1 score prend en compte à la fois les faux positifs et les faux négatifs et est calculé comme la moyenne harmonique de la précision et du rappel. En bref, plus le F1 score est élevé (proche de 1), meilleur est le modèle de classification. Il est calculé comme suit :

$$\text{F1 score} = \frac{2 \times \text{Precision} \times \text{Recall}}{\text{Precision} + \text{Recall}}$$

— **Courbe ROC :**
La courbe ROC est tracée en utilisant le taux de vrais positifs contre le taux de faux positifs (rappel). C'est une courbe de probabilité, et l'AUC exprime le degré de séparabilité. En d'autres termes, elle démontre la capacité du modèle à distinguer entre les classes prédites. Lorsque l'AUC est proche de la valeur 1, cela signifie que le modèle est plus efficace.

— **Matrice de confusion :**
La matrice de confusion permet de résumer les vrais positifs, les faux négatifs, les faux positifs et les vrais négatifs.

4.4 Résultats et discussion

Dans cette section, nous avons minutieusement évalué les performances des modèles Decision Tree (DT), Random Forest (RF), K-nearest Neighbors (KNN), Support Vector Machine (SVM) et Multi-Layer Perceptron (MLP) sur nos données de test. Chaque modèle a été soumis à des tests rigoureux pour évaluer leur capacité à généraliser et à classifier efficacement les données non vues. Les résultats ont révélé des différences significatives entre les performances des différents modèles.

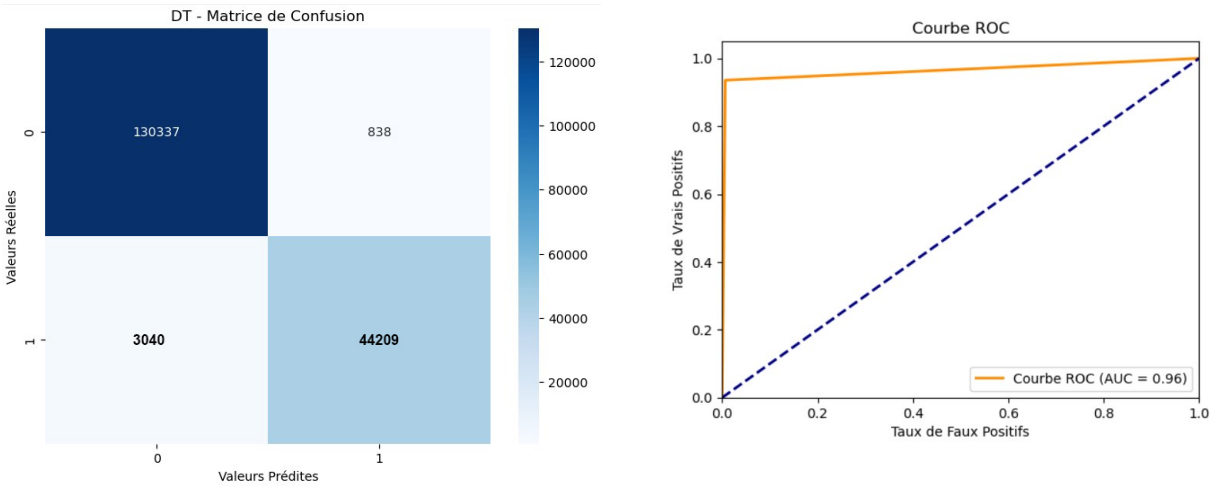

FIGURE 4.1 – Performances de l'arbre de décision.

Le modèle d'arbre de décision (DT) montre une performance globale très satisfaisante. La matrice de confusion indique 130337 vrais négatifs et 44209 vrais positifs, démontrant une haute précision pour la classe 0 et une bonne capacité de prédiction pour la classe 1, malgré 3040 faux négatifs et 838 faux positifs. La courbe ROC, avec une AUC de 0.96, confirme l'excellente capacité du modèle à distinguer entre les classes positives et négatives. Bien que les résultats soient globalement très bons, le nombre de faux négatifs indique qu'il y a encore place à amélioration pour la reconnaissance de la classe 1.

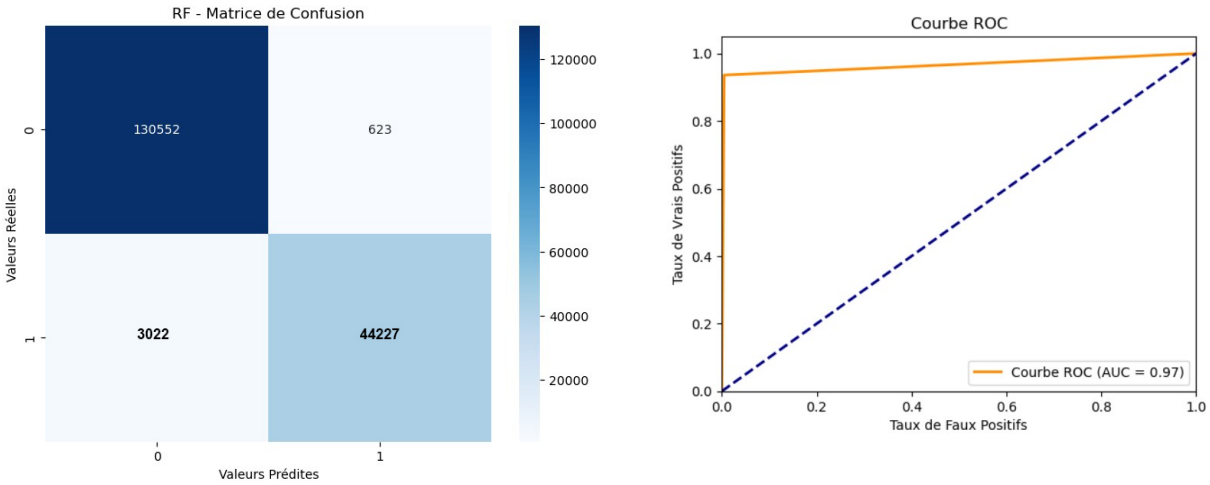

FIGURE 4.2 – Performances de Forêt aléatoire.

Le modèle de forêt aléatoire (Random Forest) a démontré une excellente performance, avec 130573 vrais négatifs, 44229 vrais positifs, seulement 602 faux positifs et 3020 faux négatifs, et une AUC de 0.97. Ces résultats, marqués par une haute précision, peuvent être attribués à la robustesse du modèle RF, qui utilise plusieurs arbres de décision pour améliorer la généralisation et réduire le risque de surapprentissage.

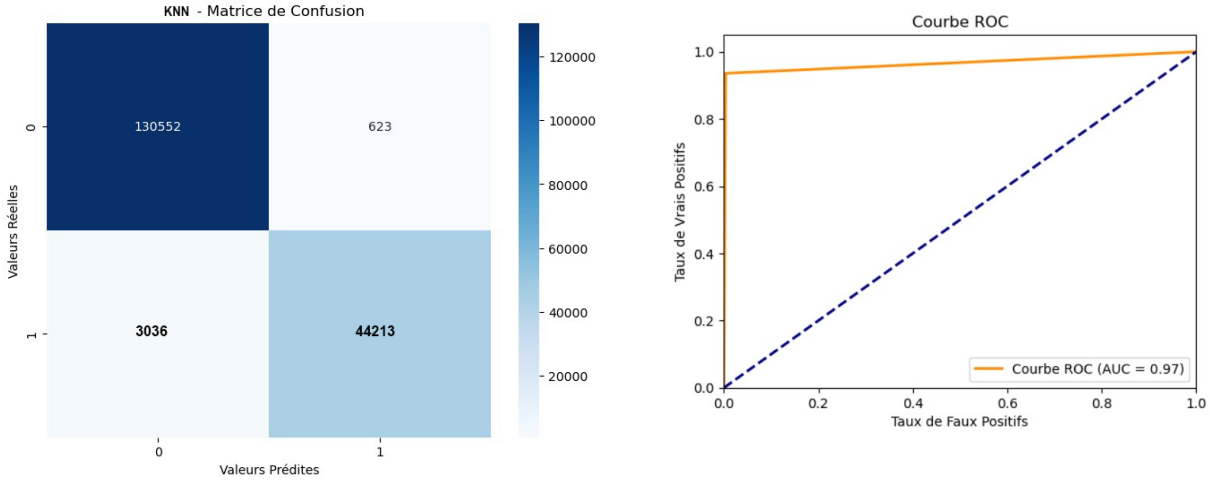

FIGURE 4.3 – Performances de KNN.

Le modèle des k-plus proches voisins (KNN) a affiché une performance remarquable, avec 130706 vrais négatifs et 44213 vrais positifs, contre seulement 469 faux positifs et 3036 faux négatifs, et une AUC de 0.97. Ces résultats soulignent la précision élevée et la capacité du KNN à bien distinguer entre les différentes classes. Sans spécifier les hyperparamètres, le KNN s'est avéré efficace en s'appuyant sur les similarités locales pour faire des prédictions fiables, ce qui démontre sa robustesse même dans des conditions par défaut. Le faible taux de faux positifs et de faux négatifs met en évidence la capacité du modèle à généraliser efficacement à partir des données d'entraînement.

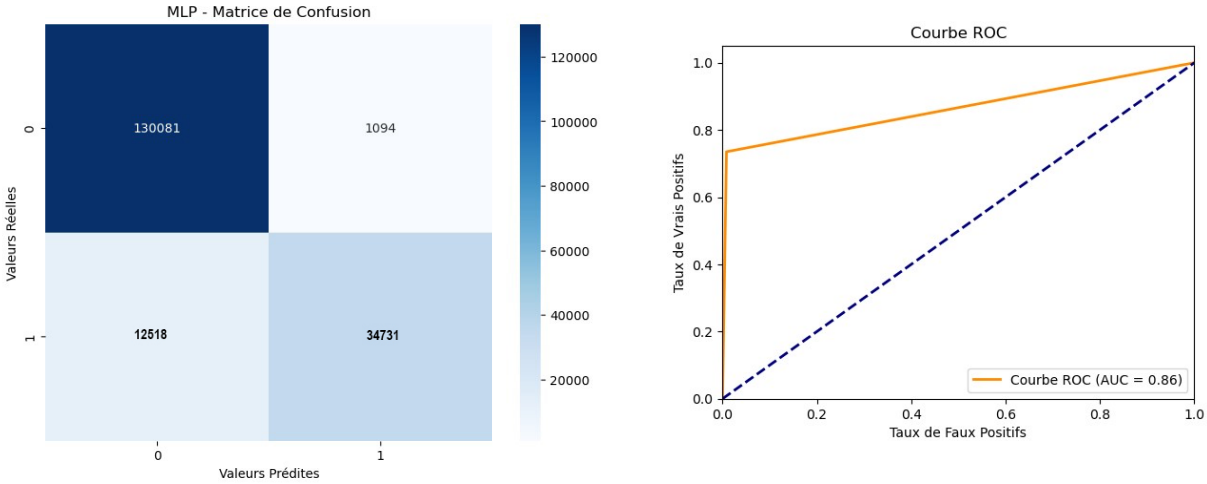

FIGURE 4.4 – Performances de MLP.

Le modèle de perceptron multicouche (MLP) a montré une performance solide, avec 130081 vrais négatifs, 34731 vrais positifs, 1094 faux positifs et 12518 faux négatifs, et une AUC de 0.86. Ces résultats révèlent une bonne capacité de discrimination entre les classes, démontrant l'efficacité du MLP à capturer des relations complexes dans les données. Cependant, le nombre plus élevé de faux négatifs indique une marge d'amélioration pour mieux identifier les instances positives.

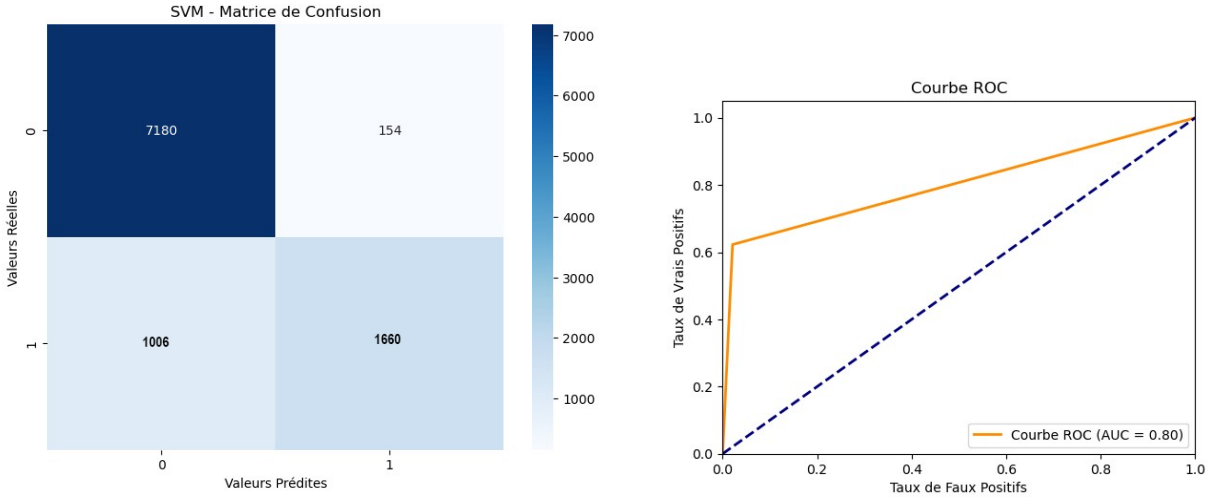

FIGURE 4.5 – Performances de SVM.

Le modèle de machine à vecteurs de support (SVM) a affiché une performance relativement inférieure parmi les cinq algorithmes étudiés, avec 7180 vrais négatifs, 1660 vrais positifs, 154 faux positifs et 1006 faux négatifs, et une AUC de 0.8. Bien que le SVM ait montré une capacité raisonnable de discrimination entre les classes et une efficacité à définir une frontière de décision optimale avec un noyau RBF dans des données complexes, le

nombre élevé de faux négatifs indique des difficultés potentielles à détecter correctement les instances positives.

Remarque : Vu la complexité du noyau de SVM choisi « RBF » l'entraînement de ce modèle prend un temps énorme et pour y remédier, nous avons décidé de prendre seulement une portion de nos données pour l'entraîner et l'évaluer.

	DT	RF	KNN	SVM	MLP
Exactitude	97.82%	97.95%	**98.03%**	88.4%	92.37%
Précision	97.92%	98.20%	**98.34%**	89.61%	94.08%
Rappel	96.46%	96.57%	**96.61%**	80.08%	86.34%
F1Score	97.16%	97.35%	**97.43%**	83.32%	89.32%

TABLE 4.2 – La performance des algorithmes selon les catégories Bénin/Attaque.

Les résultats présentés dans le tableau 4.2 montrent une comparaison des performances de plusieurs algorithmes de classification, à savoir l'arbre de décision (DT), la forêt aléatoire (RF), les k-plus proches voisins (KNN), la machine à vecteurs de support (SVM) et le perceptron multicouche (MLP). Il est évident que le KNN, le RF et le DT affichent les meilleures performances en termes d'exactitude, de précision, de rappel et de F1-score. Plus précisément, le KNN atteint la meilleure exactitude avec 98,03 %, suivi de près par le RF avec 97,95 % et le DT avec 97,81 %. En termes de précision, le RF et le KNN sont en tête avec 98,34 % et 98.2 % respectivement, et en ce qui concerne le rappel, le DT, RF et le KNN montrent également des résultats supérieurs, chacun atteignant 97 %.

D'un autre côté, le SVM montre des performances significativement inférieures par rapport aux autres modèles, avec une exactitude de 88,4 %, une précision de 89.61 %, un rappel de 80.08 % et un F1-score de 83,32 %. Cela pourrait être attribué au fait que le SVM a été entraîné sur une portion plus petite des données en raison du temps d'apprentissage prolongé qu'il nécessite. En comparaison, le MLP affiche des performances intermédiaires, surpassant le SVM mais restant en dessous des résultats obtenus par le KNN, le RF et le DT, avec une exactitude de 92,37 %, une précision de 94.08 %, un rappel de 86.34 % et un F1-score de 89.32 %.

En conclusion, les algorithmes KNN et RF se démarquent comme les plus performants pour la classification dans ce contexte, avec des performances globalement supérieures à celles des autres modèles. Le DT, bien qu'un peu en retrait par rapport au KNN et au RF, reste très performant. Le MLP offre des performances acceptables mais n'atteint pas le niveau des trois meilleurs modèles dû à la taille des données. Enfin, le SVM, en raison de son besoin en temps d'entraînement plus élevé et de l'utilisation d'une plus petite portion de données pour l'entraînement, montre des performances moins impressionnantes. Les modèles KNN et RF semblent être les choix les plus appropriés.

4.5 Conclusion

L'objectif du dernier chapitre était de présenter une solution qui répond efficacement aux diverses limitations des systèmes de détection d'intrusion (IDS). La solution proposée améliore les performances globales des IDS en augmentant la précision de la détection et de la catégorisation d'une large gamme d'attaques, tout en minimisant les fausses alertes. Pour que les IDS puissent reconnaître des schémas et identifier de nouvelles attaques, y compris celles précédemment inconnues, il est crucial de disposer d'un large éventail d'échantillons pour chaque type d'attaque. Les expériences menées valident l'efficacité de l'approche proposée, produisant des résultats hautement satisfaisants.

Conclusion générale

Avec le développement actuel des technologies et la prolifération des services proposés sur Internet, les données massives stockées en ligne par les entreprises et les utilisateurs sont devenues des cibles privilégiées pour diverses parties malveillantes. Bien que des mécanismes de défense tels que les pare-feu et les systèmes de détection d'intrusion (IDS) aient été mis en place, les cybercriminels parviennent souvent à les contourner grâce à des techniques de plus en plus sophistiquées.

Au cours de notre projet, nous avons examiné les méthodes de détection classiques, y compris les pare-feu et Snort. Bien que ces méthodes aient prouvé leur efficacité dans la détection d'attaques connues, elles présentent des limites significatives, en particulier contre les nouvelles attaques zero-day et les variantes d'attaques existantes. Par conséquent, il est nécessaire de trouver des solutions capables de combler ces lacunes.

L'essor des techniques de machine learning a considérablement bénéficié à la sécurité informatique, introduisant des approches novatrices comme la détection d'anomalies. Les IDS basés sur le machine learning, qui utilisent les statistiques des flux réseau pour distinguer le trafic malveillant du trafic normal, sont devenus très courants ces dernières années.

Dans le cadre de notre contribution, nous avons conçu un IDS capable de détecter les anomalies en utilisant le dataset CIC-IDS2018. Confrontés à un déséquilibre du nombre d'instances des différentes classes d'attaques, nous avons choisi de regrouper toutes les attaques sous un même label, optant pour une classification binaire plutôt que multi-classes. Nous avons aussi fais usage des méthodes de filtrage et l'analyse de corrélation pour sélectionner les caractéristiques les plus importantes à l'apprentissage de nos modèles.

Nous avons sélectionné cinq algorithmes de machine learning : DT (Decision Tree), RF (Random Forest), KNN (K-Nearest Neighbors), SVM (Support Vector Machine) et MLP (Multilayer Perceptron) et avons choisis les hyper-paramètres de certains de ces modèles en utilisant la technique de cross-validation. Après les avoir entraînés et testés, nos résultats montrent que RF et KNN sont les plus appropriés pour ce type de classification.

En conclusion, bien que les méthodes traditionnelles de détection d'intrusion continuent d'apporter une contribution significative, l'intégration des techniques de machine learning offre une solution prometteuse pour améliorer la détection des anomalies et renforcer la sécurité des systèmes informatiques contre les menaces modernes.

Perspectives

Une approche encore plus avancée consisterait à exploiter des algorithmes de deep learning, tels que les réseaux de neurones récurrents (RNN) et les réseaux de neurones graphiques (GNN). Ces algorithmes surpassent les méthodes traditionnelles de machine learning en termes de complexité et de capacité à capturer des dépendances temporelles et structurelles dans les données. Les RNN sont particulièrement efficaces pour analyser des séquences de données, ce qui est crucial pour la détection d'intrusions où le contexte temporel est souvent déterminant. Les GNN, quant à eux, sont aptes à modéliser des relations complexes entre les différents nœuds d'un réseau, permettant une analyse plus approfondie et nuancée des flux de données réseau.

Cependant, ces algorithmes sophistiqués nécessitent des quantités de données beaucoup plus importantes et des ressources informatiques considérablement accrues pour l'apprentissage. Un dataset plus grand et bien équilibré entre les différentes classes d'attaques est crucial pour entraîner ces modèles de manière efficace. En disposant d'un volume suffisant de données représentatives et équilibrées, un IDS basé sur ces technologies pourrait réaliser une classification multi-classes avec une précision et une robustesse accrues, sans compromettre les performances.

En utilisant des RNN, on pourrait capturer des séquences complexes d'événements dans le réseau, améliorant ainsi la détection des attaques qui se déroulent sur plusieurs étapes ou qui présentent des signatures comportementales spécifiques sur le temps. Par exemple, des attaques de type Advanced Persistent Threat (APT) pourraient être plus facilement identifiées grâce à la capacité des RNN à comprendre des patterns sur des séquences de longue durée.

Les GNN, de leur côté, pourraient permettre de mieux comprendre les relations et les interdépendances entre différents appareils et utilisateurs au sein du réseau, facilitant ainsi la détection d'attaques qui exploitent des vulnérabilités structurelles du réseau. Par exemple, des attaques de type botnet, où plusieurs machines infectées communiquent et coopèrent pour exécuter des tâches malveillantes, pourraient être détectées plus efficacement.

En conclusion, bien que l'implémentation de RNN et GNN pour la détection d'intrusion soit exigeante en termes de données et de puissance de calcul, les bénéfices potentiels en termes de performance et de précision justifient pleinement cet investissement. Avec un dataset suffisamment large et équilibré, ces algorithmes pourraient révolutionner la détection des intrusions, offrant une solution de classification multi-classes capable de faire face à la diversité et à la complexité croissantes des cyberattaques modernes.

Bibliographie

[1] R. Ashraf. How to brute-force ssh in linux. https://rootinstall.com/tutorial/bruteforce-ssh-in-linux/, November 9 2021. Consulté le 28 avril 2024.

[2] Stefan Axelsson. Intrusion detection systems : A survey and taxonomy. Technical report, Chalmers University of Technology, 2000.

[3] S. Behal and K. Kumar. Characterization and comparison of ddos attack tools and traffic generators : A review. *International Journal of Network Security*, 19(3) :383–393, 2017.

[4] Andrew Baker Brian Caswell, Jay Beale. *Snort Intrusion Detection and Prevention Toolkit*. Syngress, 2007.

[5] S. Chatterjee. Hulk. https://github.com/R3DHULK/HULK, May 25 2022. Consulté le 28 avril 2024.

[6] Check Point. What is a stateless firewall ? https://www.checkpoint.com/fr/cyber-hub/network-security/what-is-firewall/what-is-a-stateless-firewall/. Accessed : 2024-04-24.

[7] Cisco. What is a vpn ? - virtual private network. https://www.cisco.com/c/en/us/products/security/vpn-endpoint-security-clients/what-is-vpn.html?dtid=osscdc000283#~types-of-vpns. Consulté le 29 avril 2024.

[8] Cloudflare. Http flood ddos attack. https://www.cloudflare.com/learning/ddos/http-flood-ddos-attack/. Consulté le 28 avril 2024.

[9] Cloudflare. Udp flood ddos attack. https://www.cloudflare.com/learning/ddos/udp-flood-ddos-attack/. Consulté le 28 avril 2024.

[10] Cloudflare. What is a denial-of-service (dos) attack ? https://www.cloudflare.com/learning/ddos/glossary/denial-of-service/. Consulté le 28 avril 2024.

[11] Cloudflare. What is asymmetric encryption ? | asymmetric vs. symmetric encryption. https://www.cloudflare.com/learning/ssl/what-is-asymmetric-encryption/. Consulté le 29 avril 2024.

[12] CloudFlare. Qu'est-ce qu'un réseau d'entreprise ?, 2024. (Visité le 05/2024).

[13] T. Cover and P. Hart. Nearest neighbor pattern classification. *IEEE Transactions on Information Theory*, 13(1) :21–27, 1967.

[14] CyberSecura. Intégrité, confidentialité, disponibilité : Définitions. https://www.cybersecura.com/post/integrite-confidentialite-disponibilite-definitions-1. Consulté le 22 juin 2024.

[15] Kavya D. Optimizing performance : Selectkbest for efficient feature selection in machine learning, 2023.

[16] Q. H. Dang. Secure hash standard. https://nvlpubs.nist.gov/nistpubs/FIPS/NIST.FIPS.180-4.pdf, July 2015. Consulté le 29 avril 2024.

[17] Anaconda Software Distribution. Anaconda. Disponible en ligne.

[18] AMROUN Kamal ELSAKAAN Nadim. A comparative study of machine learning binary classification methods for botnet detection, 2023. Consulté le 24 mai 2024.

[19] Khraisat et al. Survey of intrusion detection systems : techniques, datasets and challenges. *SpringerOpen*, 2019.

[20] Jonathan Farhi. Le pare-feu : définition et configuration, 2024.

[21] Fortinet. Cia triad. https://www.fortinet.com/fr/resources/cyberglossary/cia-triad. Consulté le 22 juin 2024.

[22] Fortinet. Stateful firewall. https://www.fortinet.com/resources/cyberglossary/stateful-firewall. Accessed : 2024-04-24.

[23] geeksforgeeks. Ml classification vs regression. https://www.geeksforgeeks.org/ml-classification-vs-regression/. Consulté le 24 mai 2024.

[24] GeeksforGeeks. Slowloris ddos attack tool in kali linux. https://www.geeksforgeeks.org/slowloris-ddos-attack-tool-in-kali-linux/. Consulté le 28 avril 2024.

[25] GeeksforGeeks. Goldeneye ddos tool in kali linux. https://www.geeksforgeeks.org/goldeneye-ddos-tool-in-kali-linux/, June 21 2021. Consulté le 28 avril 2024.

[26] Huawei. Qu'est-ce qu'une image de machine virtuelle ? https://forum.huawei.com/enterprise/fr/qu-est-ce-qu-une-image-de-machine-virtuelle/thread/667498022805848064-667481002609618944, Consulté en 2024. Consulté le 24 juin 2024.

[27] IBM. Qu'est-ce qu'un arbre de décisions ?, 2024. (Visité le 05/2024).

[28] IBM. Qu'est-ce qu'une cyberattaque ?, 2024. (Visité le 05/2024).

[29] IBM. What is deep learning ?, 2024. Visitée : 2024-06-11.

[30] Javapoint. Random forest algorithm. (Visité le 05/2024)'.

[31] jedha. Algorithmes de machine learning. https://www.jedha.co/formation-ia/algorithmes-machine-learning, 2024. Consulté le 24 mai 2024.

[32] JMP. What is correlation ? https://www.jmp.com/fr_ca/statistics-knowledge-portal/what-is-correlation.html, Consulté en 2024. Consulté le 24 juin 2024.

[33] Jupyter. (En ligne).

[34] Kaspersky. Qu'est-ce qu'un botnet ?, 2024. (Visité le 05/2024).

[35] Les Inrocks. Infiltrations, attaques informatiques : la cyberguerre a commencé. https://www.lesinrocks.com/actu/infiltrations-attaques-informatiques-la-cyberguerre-a-commence-97453-10-04-2011/, April 10 2011. Consulté le 22 juin 2024.

[36] Matplotlib. (Documentation).

[37] Fomani Boris Mohamadally Hasan. Svm : Machines à vecteurs de support ou séparateurs à vastes marges. *BD Web, ISTY3 Versailles St Quentin, France*, 2006.

[38] Netwrix. Confidentialité, intégrité et disponibilité : Application dans le monde réel. https://blog.netwrix.fr/2020/06/23/confidentialite-integrite-et-disponibilite-application-dans-le-monde-reel/. Consulté le 22 juin 2024.

[39] Numpy. (Documentation).

[40] Oracle. Web application firewall (waf) definition. https://www.oracle.com/fr/security/waf-definition-pare-feu/. Accessed : 2024-04-24.

[41] OWASP Foundation. Brute force attack. https://owasp.org/www-community/attacks/Brute_force_attack. Consulté le 28 avril 2024.

[42] Pandas. (Documentation).

[43] Python. (En ligne).

[44] Saylor. Cs406 : Information security.

[45] Scikit-learn. (En ligne).

[46] Tchin-Lait. Source interne.

[47] Techopedia. Brute force attack. https://www.techopedia.com/definition/18091/brute-force-attack, July 1 2020. Consulté le 28 avril 2024.

[48] Timothy N Newsham Thomas H. Ptacek. Insertion evasion and denial of service, eluding network intrusion detection. *Secure Networks, Inc*, 1998.

[49] UNB. Ids 2018, 2018. https ://registry.opendata.aws/cse-cic-ids2018.

[50] V.Kanimozhi and T.Prem Jacob. Machine learning approaches for classification. https://www.sciencedirect.com/science/article/pii/S2405959520304926, 2020. Accessed : 2024-04-24.

[51] WonderHowTo. How to brute-force ftp credentials & get server access. https://null-byte.wonderhowto.com/how-to/brute-force-ftp-credentials-get-server-access-0208763/, March 11 2020. Consulté le 28 avril 2024.