République Algérienne Démocratique et Populaire
Ministère de l'Enseignement Supérieure et de la Recherche Scientifique
Université Abderrahmane Mira
Faculté de la Technologie

جامعة بجاية
Tasdawit n Bgayet
Université de Béjaïa

ATE Department

Département d'Automatique, Télécommunications et d'Electronique

Projet de Fin d'Etudes

Pour l'obtention du diplôme de Master

Filière : Télécommunications

Spécialité : Réseaux et télécommunications

Thème

Détection et classification des attaques réseau à base des algorithmes d'apprentissage automatique

Préparé par :

- Cherfi Aicha
- Bouriden Amina

Dirigé par :

M. Diboune

Examiné par :

M. Boualem

M. Tounsi

Année universitaire : 2023/2024

Remerciement

Nous tenons tout d’abord à exprimer notre profonde gratitude envers **Dieu** le tout puissant, qui nous a donné la force et la patience nécessaires pour mener à bien ce modeste travail.

Nous tenons à exprimer notre respectueuse gratitude et nos sincères remerciements à notre Promoteur **Mr. DIBOUNE** pour sa disponibilité, ses précieux conseils et son soutien tout au long de la réalisation de ce mémoire. Ses orientations complémentaires et son expertise ont été d'une aide inestimable.

Nous remercions également les membres du jury, le président **Mr. BOUALEM** et l’examinateur **Mr. TOUNSI**, d'avoir accepté d'examiner et de corriger ce travail.

Nous exprimons également notre profonde gratitude envers nos familles, pour leurs soutiens constants et leurs encouragements précieux qui nous ont permis de surmonter les moments difficiles et de mener à bien ce projet.

Nos remerciements vont aussi à tous nos amis, pour leur présence et leurs conseils avisés, en particulier notre amie **Sonia KHARONI**.

Enfin, nous souhaitons exprimer notre reconnaissance envers toutes les personnes qui ont apporté leur contribution, de près ou de loin, à la réalisation de ce travail. Leur soutien moral et leur encouragement ont été d’une grande valeur pour nous.

Dédicaces

À mes chers parents, pour leur soutien indéfectible et leurs encouragements tout au long de mon parcours. Vous avez toujours cru en moi et m'avez donné la force de surmonter les obstacles pour réaliser ce travail. Ce mémoire vous est dédié, en témoignage de ma profonde gratitude et de mon immense respect.

Amina

À mes chers parents, mes sœurs Kamilia et Naziha, mon frère Boualem et mon amie Siham, ce mémoire vous est dédié, à vous qui m'avez soutenu et encouragé tout au long de ce parcours.

Aicha

Table des Matières

Liste des Abréviations

A

ACK *Acknowledgement.*
AmpV4.8 *Advanced Modular Packet Analyzer version 4.8.*

C

CART *Classification And Regresion Trees.*

D

DBSCAN *Density-Based Spatial Clustering of Applications with Noise.*
DDOS *Distributed Denial-of-Service.*
DMZ *Demilitarized Zone.*
DOS *Denial-of-Service.*

F

FCM *Fuzzy Clustering means.*
FPR *False Positive Rate*
FTP *File Transfer Protocol.*

H

HIDS *Host based IDS.*
Http *Hypertext Transfer Protocol.*

I

ICMP *Internet Control Message Protocol.*
IDS *Intrusion Detection Systems.*
IEEE *Institute of Electrical and Electronics Engineers.*
IPS *Intrusion Prevention System*
ISO *International Organization for Standarization.*

K

KNN *K- Nearest Neighbors.*

L

LAN *Local Area Network.*
LightGBM *Gradient-boosting framework for machine learning.*
LMDRT-SVM *Logarithm Marginal Density Ratios Transformation-SVM.*

M

MAC *Media Access Control.*
MAP *Maximum A Posteriori.*
MIM *Men In The Middle.*
ML *Machine Learning.*

N

NAS *Network Access Server.*
NIDS *Network IDS.*
NMAP *Network Mapper.*
NSL-KDD *Network Security Laboratory- Knowledge Discovery in Databes*

O

OSI *Open system Interconnection.*

P

POD *Ping Of Death.*
POF *Passive OS Fingerprinting.*

R

R2L *Remote To Local.*
RBF *Radial Basis Function.*
RFC *Request For Comments*
RMSE *Root Mean Square Error.*
RST *Reset.*

S

SMTP *Simple Mail Transfer Protocol.*
STG2P *A Two-Stage Pipeline Model for Intrusion Detection Based on Improved LightGBM and Reinforced K-means*
SVM *Support Vector Machine.*
SYN *Synchronize.*

T

TCM-KNN *Transductive Confidence Machines for K-Nearest Neighbors*
TCP/IP *Transmission Control Protocol/ Internet Protocol.*

U

U2R *User To Root.*
UDP *User Datagram Protocol.*
UNIDS *Unsupervised Network Intrusion Detection System.*

URL *Uniform Resource Locator*.

W

WAN *Wide Area Network.*

X

XSS *Cross-Site Scripting*.

Introduction générale

Introduction générale

La circulation des données au sein des réseaux informatiques modernes engendre une vulnérabilité croissante face à diverses menaces, telles que les cyberattaques et les intrusions malveillantes. Ce constat souligne la nécessité de mettre en place des mécanismes de sécurité robustes pour protéger ces infrastructures essentielles.

Les solutions traditionnelles comme les pare-feux et les antivirus demeurent des outils courants dans les réseaux d'entreprise. Cependant, leur efficacité s'avère limitée face à l'évolution rapide des techniques de piratage. Ce défi crée un besoin crucial de déployer des systèmes de détection d'intrusion plus performants, capables d'offrir une protection renforcée des réseaux modernes contre les cybermenaces émergentes.

Les récentes avancées dans le domaine de l'apprentissage automatique ouvrent de nouvelles perspectives prometteuses pour améliorer les performances et la fiabilité de ces systèmes de détection d'intrusion. En analysant en détail le trafic réseau, les IDS basés sur l'apprentissage automatique collectent une multitude d'événements, permettant ainsi d'identifier les tentatives d'intrusion et de générer des alertes. Contrairement aux approches traditionnelles, ces systèmes utilisent des algorithmes avancés tels que les algorithmes de classification pour créer des modèles prédictifs des nouvelles attaques émergentes.

L'objectif de ce mémoire est d'étudier en profondeur les performances des algorithmes d'apprentissage automatique non supervisé et supervisé appliqués aux systèmes de détection d'intrusion. Il s'articulera autour de trois chapitres principaux :

Le premier chapitre présentera un aperçu général de la sécurité des réseaux, en soulignant les enjeux et les défis liés à la protection des infrastructures réseau face aux menaces.

Le deuxième chapitre se concentrera sur les systèmes de détection d'intrusion basés sur l'apprentissage automatique et leur fonctionnement. Il détaillera les différentes approches de détection d'intrusion utilisées, ainsi que le fonctionnement de chaque algorithme.

Enfin, le troisième chapitre se penchera sur l'implémentation et l'évaluation des algorithmes d'apprentissage automatique non supervisé (K-moyennes, DBSCAN) et supervisé (KNN, SVM, DT) dans le cadre de la détection d'intrusion. Il analysera les résultats obtenus et discutera des implications de ces techniques innovantes pour l'amélioration de la sécurité des réseaux.

Ce mémoire s'inscrit dans un contexte de recherche visant à renforcer la sécurité des réseaux informatiques en exploitant les avancées de l'apprentissage automatique. Il contribuera à une meilleure compréhension et évaluation des IDS basés sur ces techniques, dans le but d'aider les professionnels de la sécurité à mieux protéger leurs infrastructures réseau face aux menaces émergentes.

Chapitre I

Généralités sur la sécurité des réseaux

I Généralités sur la sécurité des réseaux

I.1 Introduction

Aujourd'hui, les réseaux informatiques évoluent de plus en plus, ce qui ouvre les portes à l'émergence de nouveau type de cyber attaques, leur sécurisation est donc une priorité absolue, garantissant la confidentialité, l'intégrité et la disponibilité des données sensibles.

Ce chapitre introduit les concepts fondamentaux de la sécurité informatique notamment : les services de sécurité, les étapes d'une attaque réseau, taxonomie des attaques et les parades réseau

I.2 Fondements des protocoles réseau et modèles de communication

I.2.1 Définition de protocole réseau

Un protocole réseau est une norme ou un ensemble de règles qui permet la communication entre deux ou plusieurs entités sur un réseau. Les protocoles réseau définissent les formats de données, les types de messages, les méthodes d'authentification et les méthodes de contrôle d'accès, qui sont utilisés pour assurer la sécurité et la fiabilité de la communication.

I.2.2 Les modèles de communication en couches

Les modèles en couches permettent de décrire l'interaction entre différents protocoles, ainsi que leurs fonctionnements au niveau de chaque couche.

Parmi les modèles en couches existants, on peut citer le modèle OSI et TCP/IP.

1- Modèle OSI

OSI (*Open system Interconnections*) est un modèle en couche conçu par L'organisation ISO (*International Organization for Standarization*) en 1984. Il décrit les fonctionnalités nécessaires à la communication des systèmes informatiques sur un réseau.

Le modèle OSI comporte une architecture de communication divisée en sept couches, chacune ayant une fonction spécifique [1], comme le montre Tableau I-1.

Couche		Fonction
1	Physique	S'occupe de la connexion physique sur le réseau : Elle est responsable de la transmission brute des bits sur un support physique.
2	Liaison de donnée	Elle s'assure que les données sont transmises sans erreur et qu'elles arrivent au bon destinataire.
3	Réseau	Assure la commutation et le routage des paquets entre les nœuds du réseau.
4	Transport	Permet l'établissement, le maintien et la rupture des connexions.
5	Session	Permet d'établir une connexion logique entre deux applications et d'assurer l'organisation et la synchronisation d'une session de communication.
6	Présentation	S'occupe du formatage des données et du cryptage/décryptage.
7	Application	Fournit les services et interfaces de communication aux utilisateurs.

Tableau I-1 : Les couches de modèle OSI et leurs fonctions

2- Modèle TCP/IP

TCP/IP Signifie (*Transmission Control Protocol/ Internet Protocol*), également appelé modèle internet. L'architecture des protocoles TCP/IP est une norme ouverte dont toute la documentation concernant les descriptions et les spécifications connexes est décrite dans une série de rapports techniques accessibles au public appelés RFC (*Request For Comments*), qui sont publiés sur le réseau, puis analysés et examinés par la communauté internet [2]. Il définit une suite de plusieurs protocoles, notamment le protocole IP et TCP qui sont les principaux protocoles de ce modèle.
Comme décrit dans Figure I-1**,** le modèle TCP/IP contient quatre couches principales :

- **La couche Accès réseau :** intègre les services de la couche physique et liaison du modèle OSI, elle permet le contrôle des périphériques et supports qui constituent le réseau.
- **La couche Internet :** correspond à la couche réseau du modèle OSI, elle consiste à acheminer le flux dans le réseau (Routage).
- **La couche Transport :** cette couche est similaire à la quatrième couche de modèle OSI, elle assure la connexion de bout en bout, contrôle le flux et définit la notion de ports.
- **La couche application :** regroupe les couches 5, 6 et 7 (Session, Présentation, Application) du modèle OSI, elle assure le contrôle des sessions de communications, codage des données et leurs représentations aux utilisateurs.

Pour permettre une bonne communication entre deux couches de même niveau, plusieurs protocoles sont implémentés dans chaque couche comme la montre la figure I-1.

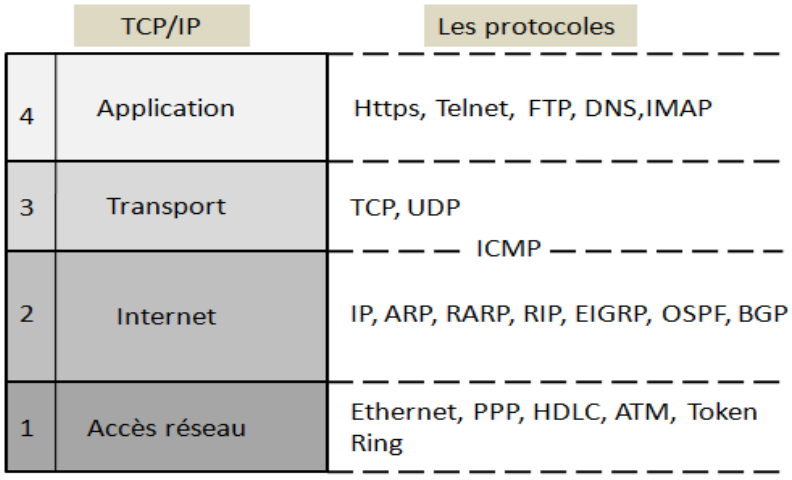

Figure I-1 **:** Les Protocoles associés aux couches de modèle TCP/IP

I.3 Services de sécurité

La sécurité des réseaux est nécessaire pour prévenir les attaques et les failles du système afin d'éviter toute types de risque tel que le vol et l'altération d'information. Elle vise généralement ces cinq objectifs :

- **L'intégrité :** assurer que l'information n'a pas été altérée ou modifiée par des entités non autorisées.
- **La confidentialité :** vise à garantir que l'information ne soit accessible que par des entités (site, personnes, organisation, etc) autorisées et d'une manière légitime. Cette propriété est assurée par la cryptographie des données.
- **L'authenticité :** permet de garantir que l'information reçue provient bien de l'entité émettrice.
- **La non-répudiation :** permet d'empêcher les entités de démentir une action qu'elles ont effectuée précédemment. Par exemple prouver qu'une entité a bien envoyé un message ou effectué une transaction.
- **La disponibilité :** offre une assurance d'accès des entités autorisées aux ressources du réseau avec une qualité de service adéquate et en toute circonstance.

I.4 Terminologie de la sécurité

- **Risque :** est une association d'une menace à une vulnérabilité, autrement dit c'est la possibilité qu'une menace exploite une vulnérabilité.
- **Vulnérabilité :** est une faille de sécurité (faiblesse) dans le système qui peut être exploité par des attaquants. Une vulnérabilité peut provenir de mauvaises configurations, de failles de sécurité dans les logiciels, de faiblesses physiques des équipements informatiques, etc.
- **Menace :** est un événement accidentel ou délibéré visant le réseau, capable s'il se réalise de causer un dommage à un système donné. Il peut être interne ou externe.
- **Attaque :** représente le moyen d'exploiter une vulnérabilité d'un système dont il peut y avoir plusieurs attaques pour une seule vulnérabilité.
- **Intrusion :** fait référence à la réalisation d'une attaque ou d'une menace visant à compromettre la sécurité d'un système**.**

I.5 Les étapes d'une attaque réseau

La mise en place d'une attaque réseau simple ou complexe exige une planification soignée et une exécution précise, pour cela voici les étapes les plus courantes à suivre :

I.5.1 Reconnaissance

Vise à collecter un maximum d'informations sur le réseau en suivant ces deux étapes :

- **Identification de la cible :** cette étape est indispensable à toutes les attaques organisées où l'attaquant commence par collecter des renseignements sur la cible.
- **Le scanning :** cette étape complète l'étape précédente, il s'agit d'un pré attaque qui consiste à récolter des informations plus détaillées sur le réseau cible comme ses vulnérabilités, les adresses IP utilisées et les services accessibles, ce qui assure une meilleure connaissance du système.

I.5.2 Exploitation

Une fois qu'une vulnérabilité a été identifiée, l'attaquant passe à l'action et tente de l'exploiter pour obtenir un accès au réseau cible.

I.5.3 Progression

Après l'obtention de l'accès complet au système, l'attaquant peut exécuter ses actions malveillantes et dissimuler ses traces (suppression des fichiers journaux par exemple) pour rendre difficile l'identification de l'attaque.

I.6 Taxonomie des attaques réseau

Pour une meilleure compréhension et une défense efficace contre les attaques réseau, une classification organisée est essentielle. C'est là qu'intervient la classification basée sur le modèle TCP/IP, qui regroupe les différentes attaques réseau en fonction des couches visées de ce modèle, permettant ainsi une analyse structurée des types d'attaques ciblant chaque couche protocolaire spécifique.

I.6.1 Couche application

1- Cross-site Scripting (XSS)

Une attaque XSS consiste à insérer un code malveillant, généralement écrit avec le langage JavaScript dans un site web fiable. Cette attaque se produit lorsqu'un attaquant injecte son script malveillant dans le contenu du site web légitime, qui est ensuite inclus dans le contenu dynamique reçu par le navigateur de la victime, car le navigateur ne peut pas différencier entre les balises valides et celles du pirate, ce qui permit au scripte nocif d'accéder aux cookies, aux jetons de session et au d'autres informations sensibles stockées par le navigateur et utilisées sur ce site [3].
Les attaques XSS peuvent être classées en deux catégories principales, XSS reflété et XSS stocké.

- **L'attaque XSS reflétée**

Également appelée XSS non persistant, elle se produit lorsque le script malveillant est renvoyé dans la réponse du serveur du site web et transmis à l'utilisateur légitime via une URL (voir FigureI-2).

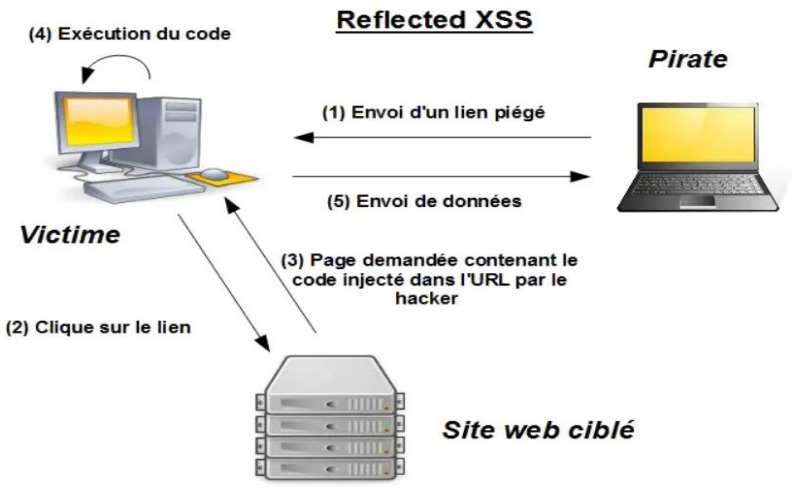

Figure I-2 : L'attaque XSS reflétée [4]

- **L'attaque XSS stockée**

Également appelée XSS persistant, elle se produit lorsque le script malveillant est stocké d'une manière permanente dans le serveur du site web, ce qui permet l'exécution de ce code à chaque fois qu'un utilisateur visite le site web (voir Figure I-3).

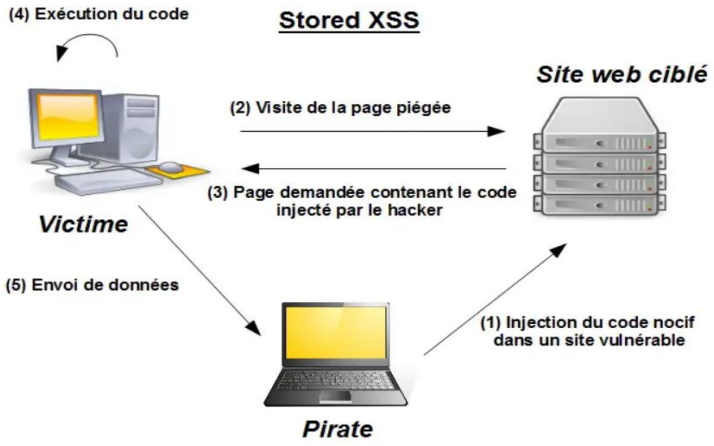

Figure I-3 : L'attaque XSS stockée [4]

2- R2L (Remote To Local)

Est une forme d'attaque qui vise à usurper les paramètres d'authentification d'une cible distante pour exécuter des commandes et des actions malveillantes. Cette attaque se produit lorsqu' un pirate ayant la capacité d'envoyer des paquets à une machine sur un réseau, mais sans avoir de compte sur cette machine, en exploitant une certaine vulnérabilité pour obtenir un accès local en tant qu'utilisateur de cette machine [5]**.**

3- U2R (User To Root)

Est une forme d'attaque utilisée par les pirates pour accéder à des niveaux des privilèges élevés sur un système. L'attaquant commence par l'obtention d'un accès à un compte utilisateur normal sur le système en sniffant les mots de passe, puis il exploite une vulnérabilité spécifique sur ce système, pour obtenir un accès administrateur [5].

4- Injection SQL

SQL est l'abréviation de *Structured Query Language* (Langage de requête structuré). C'est un langage de programmation permettant d'interroger des bases de données relationnelles afin d'obtenir les informations souhaitées.

Etant donné que la majorité des sites et des applications web reposent sur des bases de données SQL, une attaque par injection SQL peut entrainer des conséquences graves pour les organisations.

Une injection SQL, parfois abrégée en SQLi, est un type d'attaque utilisée pour manipuler une base de données en injectant des instructions SQL malveillantes dans des champs d'entrée et de saisie, permettant ainsi aux attaquants d'accéder à des informations potentiellement importantes, de détruire des données sensibles ou d'adopter d'autres comportements de manipulation [6]**.**

Par exemple, lorsqu'un utilisateur saisit des informations de connexion sur un formulaire web, ces données sont vitrifiées par rapport à une base de données pour autoriser l'accès. Mais le problème est que la plupart des formulaires web ne disposent aucun moyen pour arrêter la saisie d'informations supplémentaires sur ces formulaires. Et cela permet aux attaquants d'exploiter cette faille et utiliser les boites de saisie de formulaire pour envoyer leurs propres demandes vers la base de données [6]**.** Figure I-4 schématise les étapes d'exécution d'une attaque par injection SQL.

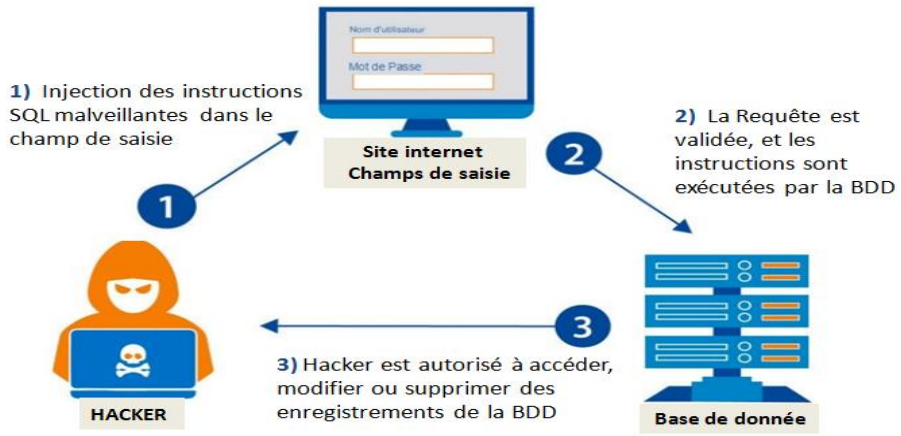

Figure I-4 : Exemple d'une attaque par injection

5- Erreur tampon

Les tampons sont des zones de mémoire utilisées pour stocker temporairement des données d'une certaine taille pendant leur transfert d'un emplacement à un autre.

Si le volume de données entrantes dépasse la capacité de stockage de cette mémoire, il y aura ce que l'on appelle un débordement, c'est-à-dire les données excédentaires débordent dans les zones mémoires adjacentes, ce qui peut détruire ou écraser les données valides stockées dans ces zones.

L'attaque par erreur de tampon, également appelée Buffer overflow en anglais, se produit lorsqu'un attaquant exploite une vulnérabilité dans un programme pour causer un débordement de tampon. Cette technique va lui permettre d'exécuter son code malveillant et compromettre la sécurité du système [7].

6- Le phishing

Le phishing, aussi appelé hameçonnage, est une forme d'escroquerie qui vise à voler des informations sensibles d'un utilisateur, telles que : numéro de sa carte de crédit, informations sur son compte bancaire, etc.

Cette attaque se déroule à plusieurs étapes :

- L'attaquant envoie un message frauduleux, que ce soit par e-mail, SMS ou par un message sur les réseaux sociaux à la victime, qui semble provenir d'une source légitime (Banque, opérateur mobile, etc.).
- Le message incite l'utilisateur à cliquer sur un lien ou à ouvrir une pièce jointe (un exemple est illustré dans Figure I-5**).**
- Le lien ou la pièce jointe redirige l'utilisateur vers un site web frauduleux qui ressemble au site web légitime.
- L'utilisateur est invité à saisir ses informations personnelles sur le site frauduleux.
- L'attaquant récupère les informations saisies par l'utilisateur et les utilise à des fins frauduleuses.

Figure I-5 : Exemple de phishing [8]

7- Attaque par sondage

Une attaque par sondage, également appelée probe ou scan, est une forme d'attaque au cours de laquelle des pirates informatiques cherchent à récupérer des informations essentielles sur une cible, e.g. les adresses IP, les noms de domaine, les systèmes d'exploitation, les applications en usage, etc. afin d'identifier des failles ou des vulnérabilités qu'ils pourront ensuite exploiter.

8- Attaques par programmes malveillants

Un programme malveillant, appelé aussi le malware en anglais, est un code ou un logiciel intrusif installé dans un système informatique à l'insu de son propriétaire. Il peut se lier à un code légitime et se propager, se dissimuler au sein d'applications utiles ou se reproduire sur Internet. Il peut prendre différentes formes [9] telles que :

- **Cheval de Troie**

Le cheval de Troie est un programme ou un logiciel malveillant qui se dissimule sous l'apparence d'un logiciel légitime afin d'inciter les utilisateurs à l'exécuter ou à l'installer. Une fois activé, il peut installer d'autres programmes malveillants sur l'appareil infecté, et aussi le contrôler et l'utiliser à des fins malveillantes, telles que l'espionnage sur la victime, perturbation des performances de l'ordinateur et le vol des données sensibles.

- **Vers**

Un ver ou Worm en anglais, est programme autonome capable de se reproduire et de se propager automatiquement à travers les connexions réseau à l'insu des utilisateurs. Il est généralement conçu dans un objectif malicieux, tels que l'espionnage de l'ordinateur infecté, la création d'une porte dérobée pour des pirates, ou encore de l'envoi de nombreuses requêtes à un serveur internet pour le saturer.

- **Virus**

Un virus est un code malveillant qui est capable de s'auto-reproduire en s'attachant à un autre programme, qu'il s'agisse d'un programme système ou d'une application. Les virus sont conçus pour se propager d'un ordinateur à un autre via tous les moyens d'échange de données numériques (les réseaux, disques durs, clefs USB, etc.) à l'aide du programme légitime sur lequel il est greffé. Le virus peut causer des dommages en perturbant le fonctionnement normal de l'appareil infecté. Par exemple Psybot, découvert en 2009, est considéré comme le seul virus informatique capable d'infecter les routeurs et les modems haut-débit.

I.6.2 Couche transport

1- Attaque TCP syn-flood (DDOS: Distributed Denial-of-Service)

Est un type d'attaque par déni de service (DoS : *Denial of service*) où l'attaquant envoie de nombreuses demandes de connexion (SYN : *synchronize*) à un serveur cible et ce dernier répond à chacune de ces demandes en envoyant des paquets SYN /ACK (*Acknowledgement*). Cependant, l'attaquant ne répond jamais avec les paquets ACK nécessaires pour finaliser la connexion, à cette phase le serveur se retrouve avec un grand nombre de connexions TCP en attente, consommant ces ressources et l'empêchant de répondre au connexions légitimes [10].

La Figure I-6 dépeint la différence entre un déroulement normale d'une session TCP et une session TCP avec attaque TCP SYN flood.

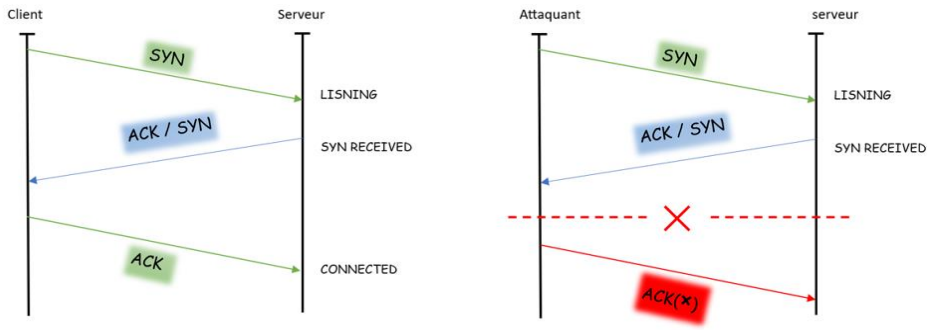

Figure I-6 : Comparaison de déroulement d'une session normale TCP avec celui d'une session avec attaque TCP SYN flood

2- Attaque par balayage de ports

Également appelé port scanning, est une technique qui consiste à identifier les ports ouverts en envoyant des paquets réseau aux différents ports d'une machine, la réponse à ces paquets permet de déterminer si le port est ouvert, fermé ou filtré. Il existe deux types de balayage :

- **Balayage TCP**

Est une technique spécifique de balayage qui nécessite l'intervention du protocole TCP pour identifier les ports ouverts d'une machine. L'attaquant envoie un paquet TCP-SYN à un port spécifique de cette machine, si la machine cible renvoie un paquet SYN/ACK, cela indique que le port correspondant est ouvert et prêt à établir une connexion. En revanche, si la réponse est un paquet RST (reset), cela signifie que le port est fermé (voir Figure I-7)**.** Au cas d'absence de

réponse, le scanner ne peut pas savoir si le port est ouvert ou fermé, car il est probable qu'un filtre de paquet, comme un pare-feu, soit positionné en amont [11].

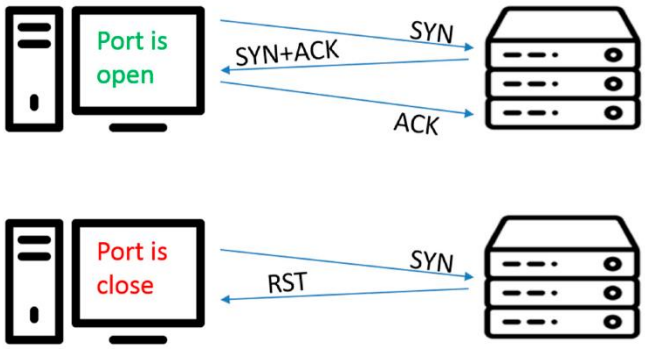

Figure I-7 : Le fonctionnement de balayage TCP [12]

- **Balayage UDP (user datagram protocol)**

Est une technique de balayage qui consiste à envoyer un paquet UDP vide (sans données) vers le port de la machine cible. En fonction de la réponse, l'attaquent peut déterminer l'état du port (ouvert, filtré ou fermé) : si le port est ouvert le paquet UDP sera rejeté et aucun accusé de réception (ACK) n'est envoyé (caractéristique du protocole UDP). Cependant, si le port est fermé un message ICMP (internet control message protocol) « port unreachable » est envoyé [13].

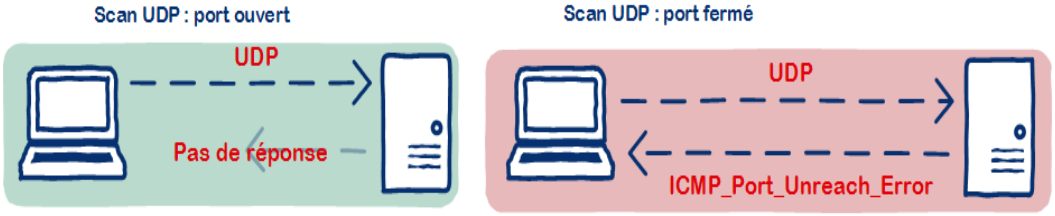

Figure I-8 : Le fonctionnement de balayage UDP [13]

3- Attaque par empreinte digitale

Est une technique utilisée par les attaquants pour identifier un système cible à distance en analysant ses caractéristiques uniques comme les logiciels et les systèmes d'exploitation en cours d'exécution, à l'aide des outils open source tels que :

- **NMAP (Network Mapper) :** un outil de découverte de système et d'audit de sécurité.
- **Ampv4.8 (Advanced Modular Packet Analyzer version 4.8) :** un outil d'analyse d'empreintes digitales de système.

- **POF (Passive OS Fingerprinting) :** un outil de détection d'empreintes digitales de système d'exploitation.

L'attaquant envoie des requêtes spécifiques (paquets ICMP, TCP, UDP, etc.) à une machine cible, puis en analysant les réponses à ces requêtes il extrait les caractéristiques du système cible, et les compare à une base de données d'empreintes digitales connues pour identifier ce système.

I.6.3 Couche internet

1- Attaque par usurpation d'adresse IP

Également appelé IP spoofing est une technique de piratage informatique utilisée pour dissimuler l'identité de l'attaquant, en envoyant un paquet avec une adresse source usurpée, autrement dit l'attaquant modifie l'entête de ce paquet en remplaçant l'adresse IP source (son adresse IP originale) par l'adresse usurpée et l'envoyé vers la destination**.** Un exemple de déroulement de l'attaque par usurpation d'adresse IP est illustré dans Figure I-9**.**

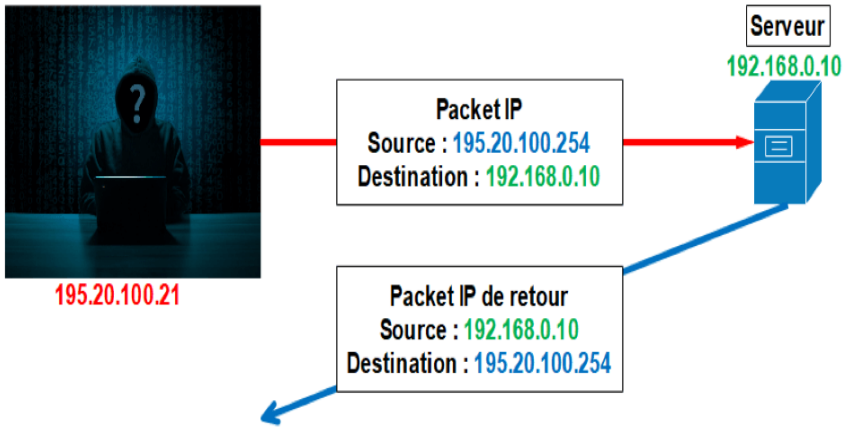

Figure I-9 : Attaque par usurpation d'adresse IP [14]

2- Attaque par déni de service (DOS) et déni de service distribué (DDOS)

Visent toutes les deux à empêcher un serveur de répondre aux requêtes légitimes, en l'inondant de paquets malveillants (déni de service par injection de paquets), saturant ainsi ses ressources (bande passante, mémoire, etc.) [15]. Comme le montre la figure I-10, la différence entre les deux réside dans leurs modes d'exécution.

- **Attaque par déni de service DOS :** provient d'une seule source (un ordinateur contrôlé par l'attaquant).

- **Attaque par déni de service distribué DDOS :** Exploite un réseau d'appareils piratés (botnet) pour lancer l'attaque simultanément à partir de multiples sources.

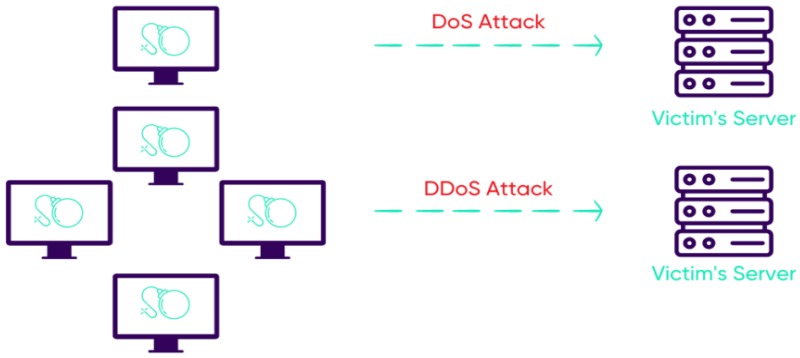

Figure I-10 : La différence entre le DOS et DDOS [16]

3- Attaque par tunnel ICMP (Internet Control Message Protocol)

Egalement connu sous le nom ICMP tunneling en anglais, est une méthode d'attaque qui exploite la nature permissive du protocole ICMP pour contourner les défenses du réseau telles que les pare-feux**.** Cette technique permet d'encapsuler des données malveillantes dans les paquets ICMP (*Echo Request ou Echo Replay généralement*) et les transporter à travers le réseau. A la réception, ces paquets vont être désencapsuler et traiter en exécutant les instructions qu'ils contiennent, ce qui permet à l'attaquant de prendre le contrôle à distance de la machine cible. La figure I-11 illustre l'attaque par ICMP**.**

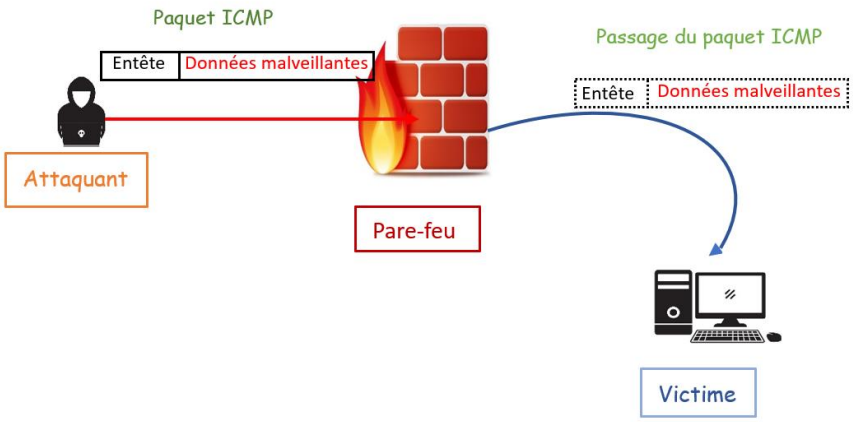

Figure I-11 : Attaque par ICMP Tunneling

4- Attaque par POD (Ping Of Death)

Est une technique d'attaque plus ancienne, qui exploite une vulnérabilité dans certains systèmes d'exploitation pour les submerger de paquets ICMP volumineux. La commande « ping -l 66500 » tente de créer un paquet ping (de type Echo Request) dont la taille dépasse la limite autorisée par le système cible, si le système cible ne dispose pas de protections contre les paquets trop volumineux, il peut planter ou redémarrer de manière inattendue [17].

I.6.4 Couche accès réseau

1- Attaque par MAC (Media Access Control) poisoning

Également connu sous le nom de ARP spoofing, est une forme d'attaque qui exploite les failles du protocole de résolution d'adresses (ARP) utilisé pour associer les adresses IP des appareils à leurs adresses MAC sur un réseau. Lors de cette attaque, la victime reçoit des paquets ARP falsifiés qui lient une adresse IP d'un appareil du réseau à une adresse MAC d'un attaquant ou d'autre appareil. Cette manipulation permet à l'attaquant d'intercepter, de modifier ou de rediriger les données transmises ultérieurement par l'appareil de la victime. La Figure I-12 illustre le déroulement de l'attaque par MAC poisoning.

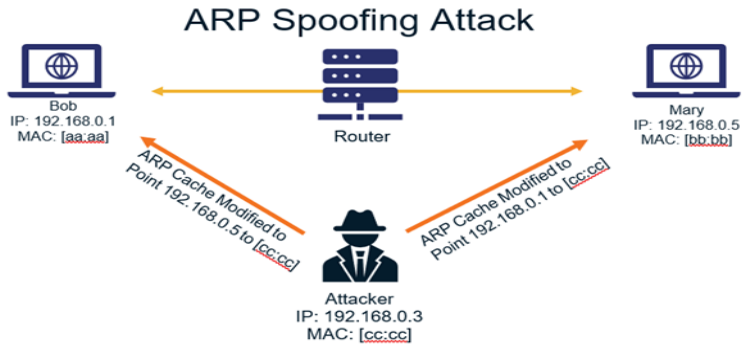

Figure I-12 : Attaque par MAC Poisoning [18]

2- Attaque par MAC Flooding

C'est une technique d'attaque qui exploite la vulnérabilité des commutateurs qui réside dans la taille limitée de leurs tables d'apprentissage MAC. L'attaquant par MAC Flooding vise à surcharger cette table en envoyant un grand nombre de trames Ethernet avec des adresse MAC source falsifiées. A la réception, le commutateur tente d'enregistrer toutes ces adresses dans la table MAC. Une fois que cette dernière est saturée, le commutateur ne peut plus stocker de nouvelles entrées, il se met donc à diffuser toutes les trames reçues sur tous ses ports (agit comme

un HUB), ce qui facilite à l'attaquant d'usurper une adresse MAC d'une machine légitime et se connecter au réseau à sa place [29]. Le déroulement de cette attaque est illustré dans Figure I-13.

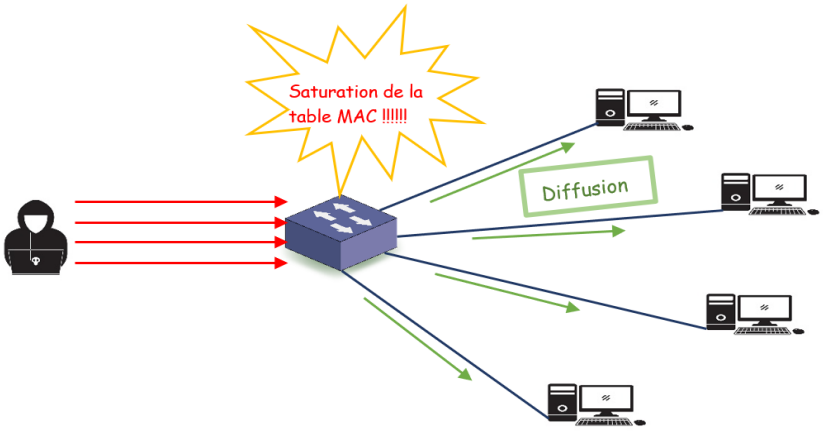

Figure I-13 Attaque par MAC Flooding

3- Attaque par homme du milieu

Également appelé *Men In The Middle* (MIM) en anglais, est une technique qui repose sur l'interception et la manipulation des communications entre deux parties communicantes. L'attaquant se positionne comme un intermédiaire invisible, capable de lire (attaque passive), modifier (attaque active) tous les messages échangés entre les deux parties, en établissant une connexion avec chacune d'elles, toute en les faisant croire que chaque partie communique directement avec son destinataire.

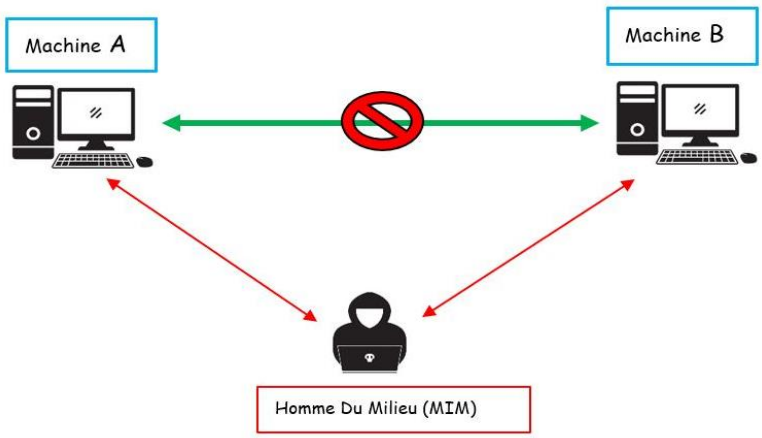

Figure I-14 : Attaque par homme du milieu

I.7 Sécurisation et parades des réseaux

Les réseaux peuvent être confronter à divers types d'attaques qui peuvent perturber leur fonctionnement normal. Par conséquent, des mesures de sécurité sont misent en place pour les protéger. Ces mesures peuvent être classées en deux catégories : mesure préventive et mesure de détection réparation.

I.7.1 Mesures préventives

I.7.1.1 Couche application

1- Proxies

Le proxy est un logiciel informatique qui joue le rôle d'intermédiaire entre deux machines pour contrôler leurs échanges. Nous utilisons également le terme « proxy » pour désigner des dispositifs, tels que les serveurs.

Un serveur proxy est configuré pour un ou plusieurs protocoles de niveau applicatif (HTTP, FTP, SMTP, etc.), mais généralement il est utilisé pour le Web, alors il s'agit d'un proxy http.
Le fonctionnement de ce dernier est assez simple : lorsqu'un utilisateur se connecte à internet via une application cliente configurée pour utiliser un serveur proxy, la requête est d'abord dirigée vers le proxy avant d'atteindre le serveur web. Une fois que le serveur web répond, sa réponse est renvoyée au proxy, qui la transmet ensuite à l'application client [19]. La Figure I-15 illustre le fonctionnement du proxy.

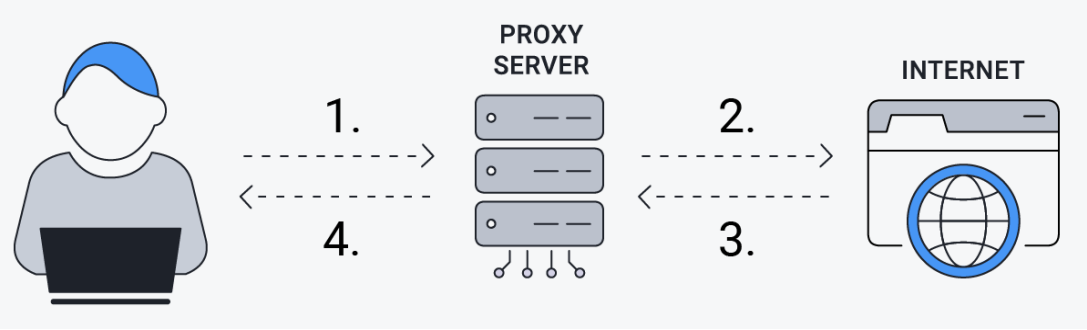

Figure I-15 : Le fonctionnement de PROXY [20]

L'utilisation de proxy, présente plusieurs avantages tels que :

- Le proxy peut cacher l'adresse IP de l'utilisateur dans sa base de données, offrant ainsi une navigation en anonyme.

- Il peut filtrer le trafic entrant et sortant, ce qui peut aider à protéger l'utilisateur contre les logiciels malveillants.
- Il peut mettre en cache les contenus fréquemment demandés, ce qui permet de réduire les temps de chargement et d'améliorer les performances globales.

I.7.1.2 Couche réseau et couche transport

1- Pare-feu

Pare-feu ou firewall en anglais, est un système de sécurité réseau (logiciel/matériel) qui contrôle et filtre le trafic entrant et sortant du réseau, en fonction d'un ensemble de règles prédéfinies (Rule based) appelées politiques de sécurité qui visent à enregistrer des événements du pare-feu, analyser des journaux pour détecter les activités suspectes (journalisation et analyse) et à bloquer les attaques malveillantes et les accès non autorisés en fonction de divers critères tels que : l'adresse IP source et destination, le contenu du paquet, le type de protocole (UDP, TCP, ICMP etc.) et le port source et destination.

Ce système joue un rôle crucial dans la segmentation du réseau en zones de sécurité distinctes : LAN (*Local Area Network*), WAN (*Wide Area Network*) et DMZ (*Demilitarized Zone*) [21].

- **C'est quoi une DMZ ?**

La DMZ est un sous réseau hébergeant les machines qui peuvent être exposées à des accès depuis le réseau extérieur WAN, généralement les serveurs (FTP, messagerie, etc.)**.** Il agit comme une zone tampon entre la zone de confiance (LAN) et la zone à risque (WAN) [21]**.** L'emplacement des différentes zones de sécurité est indiqué dans Figure I-16.

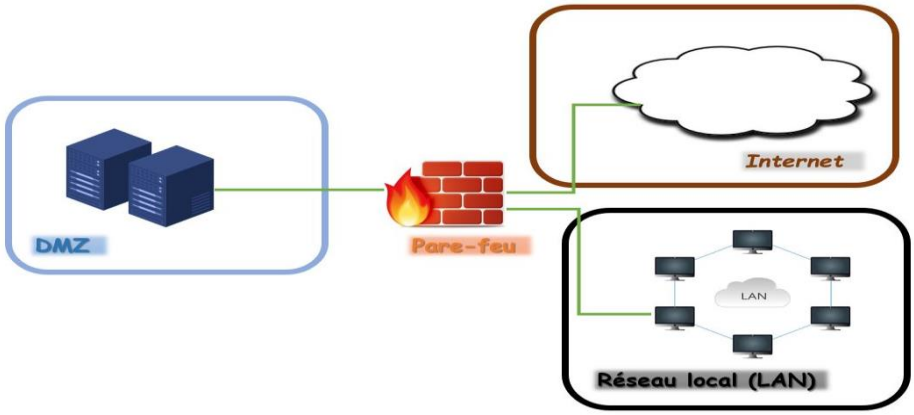

Figure I-16 : L'emplacement des zones de sécurité des pare-feux

Le pare-feu agit comme une barrière entre les trois zones de sécurité, de sorte que le réseau interne (LAN) ne reçoit aucune connexion du réseaux extérieur (WAN) et vice versa, mais il est possible d'établir des communications contrôlées avec le réseau publique (DMZ) [22].

I.7.1.3 Couche accès

1- Le protocole 802.1x

802.1x est une norme mise en point par l'IEEE (*Institute of Electrical and Electronics Engineers*), permet de contrôler l'accès aux périphériques d'infrastructures réseau en fournissant une couche de sécurité pour les réseaux filaires et sans fil. Cela se nécessite la présence d'un NAS (*Network Access Server*) qui agit en tant qu'authentificateur ou serveur d'accès réseaux tel qu'un commutateur de niveau 2 ou un point d'accès sans fil, et un serveur d'authentification qui peut être un serveur RADIUS. Le NAS envoie la demande d'authentification au serveur RADIUS, qui vérifie ensuit les informations d'identification de l'utilisateur avec l'annuaire LDAP (*lightweight directory access protocol*) [23].

Voici le fonctionnement de 802.1x comme montré dans la figure suivante **:**

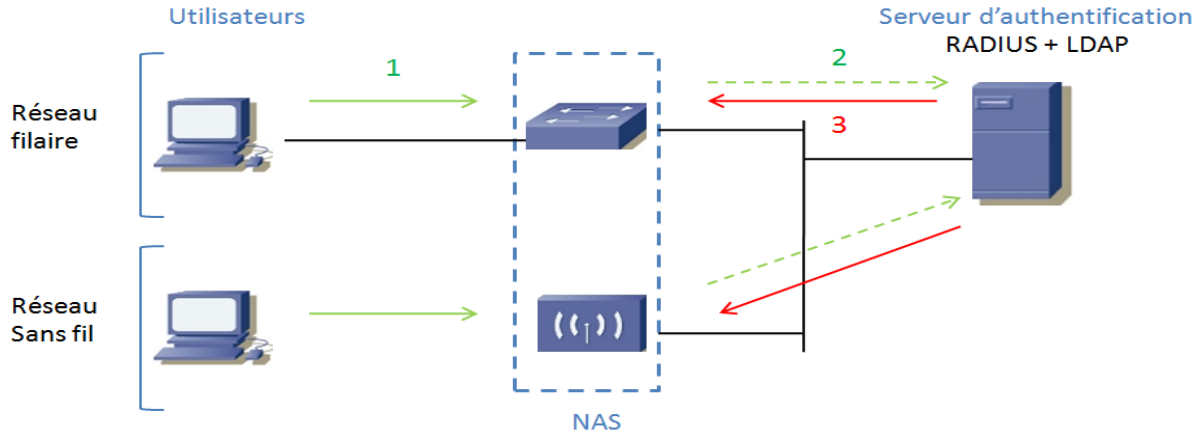

Figure I-17 : Fonctionnement de la Norme 802.1x

- **L'étape 1 :** par défaut, le port de commutateur est fermé, alors pour qu'un appareil se connecte, le commutateur active le port en ne laissant que passer la trame 802.1x.
- **L'étape 2** : Le commutateur établit la communication entre le serveur RADIUS et l'appareil, puis ce dernier envoie ses informations d'authentification au RADIUS, Celui-ci s'adresse à l'annuaire LDAP pour vérification.
- **L'étape 3** : Le serveur RADIUS répond au commutateur soit par **Accept** ou **Rejecte**.

- ✓ **Accept** : RADIUS accepte la demande de client, et il va demander au commutateur d'activer le port et de le mettre dans un VLAN particulier.
- ✓ **Reject** : RADIUS refuse la demande de client, demandant au commutateur de ne pas autoriser l'appareil à se connecter au réseau.

I.7.2 Détection réparation

I.7.2.1 Antivirus

Les antivirus sont des logiciels conçus pour identifier, neutraliser et éliminer des fichiers exécutables ou des logiciels malveillants, notamment les virus, les vers, les chevaux de Troie, et parfois des logiciels espions qui peuvent infecter un appareil. Mais il ne protège pas contre les intrus qui utilisent un logiciel légitime, ou contre un utilisateur légitime ayant un accès non autorisé.

Les antivirus sont composés généralement d'un ou plusieurs scanners qui analysent les fichiers et les programmes en temps réel ou à la demande de l'utilisateur pour détecter les virus et les autres programmes malveillant. Lorsqu'un fichier est scanné, le scanner compare son contenu avec les signatures des programmes malveillants stockées dans la base de signatures, une méthode connue sous le nom de l'analyse des signatures. Les antivirus utilisent également d'autres approches pour détecter les programmes malveillants comme : la méthode heuristique qui tend à découvrir un code malveillant par son comportement [24].

I.7.2.2 Leurre informatique (Honeypots)

Est un système informatique appât conçu pour attirer l'attaquant et le dissuader d'attaquer le système réel. Il fonctionne en simulant un système vulnérable pour inciter les attaquants à s'y engager, tout en collectant des informations sur les techniques d'attaques et leurs outils et en dissuadant les attaques en faisant croire que le système est plus facile à attaquer qu'il ne l'est en réalité. Les honeypots peuvent être utilisés pour détecter les attaques, collecter des preuves d'attaque (supervision et journalisation) pour une enquête ultérieure, et aider à identifier les attaquants [25].

I.7.2.3 La journalisation centralisée (supervision réseau)

Est une pratique fondamentale pour la supervision des réseaux. Elle implique la collecte et le stockage des journaux provenant de tous les équipements du réseau dans un emplacement

centralisé, ce qui offre une vision globale des activité réseau, simplifiant ainsi la détection et la résolution des problèmes.

I.7.2.4 Système de prévention d'intrusions IPS (Intrusion Prevention System)

1- Définition

Est un dispositif de sécurité réseau qui surveille le trafic réseau, offrant ainsi une couche de protection supplémentaire et complémentaire aux pare-feux et aux autres mesures de sécurité. Il a pour but de bloquer les activités suspectes en temps réel, en utilisant des approches de prévention suivantes [26] :

- **Approche basée sur les signatures :** consiste à analyser les paquet réseau, en les comparant à une base de données de signatures connues d'attaques.
- **Approche basée sur les anomalies :** repose sur l'analyse de trafic réseau pour détecter les comportements inhabituelles ou anormaux qui pourraient indiquer une attaque, en utilisant des algorithmes d'apprentissage automatique qui permet d'identifier les modèles des comportements normaux et détecter les écarts par rapport à ces modèles.
- **Approche basée sur les politiques :** consiste à appliquer un ensemble de règles prédéfinies pour autoriser ou bloquer un trafic réseau en fonction des paramètres spécifiques.

2- Mode de déploiement

Les IPS peuvent être déployer selon deux modes principaux :

- **Déploiement In-line :** en mode en ligne, l'IPS est directement intégré dans le flux du trafic réseau, analysant en temps réel tous les paquets entrants et sortant du réseau à la recherche des activités suspectes. Si une activité malveillante est détectée, l'IPS prend des mesures pour la bloquer ou alerter l'administrateur réseau.
- **Déploiement Off-line :** en mode hors ligne, l'IPS n'est pas directement intégré dans le flux de de trafic, mais plutôt utilisé pour analyser des enregistrements de trafic réseau copiés à des fins d'analyse. Cette méthode permet à l'IPS d'analyser le trafic à son propre rythme, sans affecter les performances du réseau principal.

I.7.2.5 Système de détection d'intrusion

1- Définition

Un système de détection d'intrusion, également appelé *Intrusion Detection System* (IDS) en anglais [25], est un ensemble de composants logiciels et/ou matériels qui a pour rôle de surveiller et analyser le réseau, dans le but de rechercher des activités malveillantes ou des signes de comportement anormal.

Les administrateurs peuvent définir des règles pour repérer des attaques. L'IDS compare l'activité du réseau avec ces règles afin d'identifier toute activité suspecte indiquant une attaque ou une intrusion. En cas de détection d'une activité malveillante, l'IDS envoie une alerte à l'administrateur pour qu'il puisse analyser la situation et prendre des mesures pour prévenir les dommages ainsi que d'autres intrusions.

2- Les Types d'IDS

Il existe deux types de l'IDS : Système de détection d'intrusion sur l'hôte et Système de détection d'intrusion réseau [27] :

- **Système de détection d'intrusion sur l'hôte (HIDS : Host Based IDS) :**

C'est un logiciel installé sur une machine du réseau spécifique pour analyser et contrôler en temps réel les flux de trafic et les fichiers journaux de cette machine, contrairement au système de détection d'intrusion réseau (NIDS : *Network IDS*) qui contrôle le trafic de tout le réseau ou une partie précise. Cette approche permet aux HIDS d'être plus précis dans la détection des types d'attaques subies par la machine. Les HIDS utilisent deux types de sources pour fournir des informations sur l'activité de la machine : les journaux d'activité (Logs) et les traces d'audit du système d'exploitation.

- **Système de détection d'intrusion réseau (NIDS) :**

C'est un système de détection d'intrusions réseau qui surveille et analyse en temps réel tout le trafic réseau. Il utilise généralement une sonde, qui est souvent une machine, pour écouter et surveiller le trafic réseau, en générant des alertes si des intrusions ou des paquets suspects sont détectés. Les NIDS, tout comme les HIDS, utilisent des sources similaires pour fournir des informations sur l'activité du réseau, telles que les journaux d'activité et les traces d'audit du système d'exploitation.

3- Architecture d'un IDS

L'architecteur de système de détection d'intrusion IDS [28] intègre plusieurs composants comme illustré dans la figure I -18.

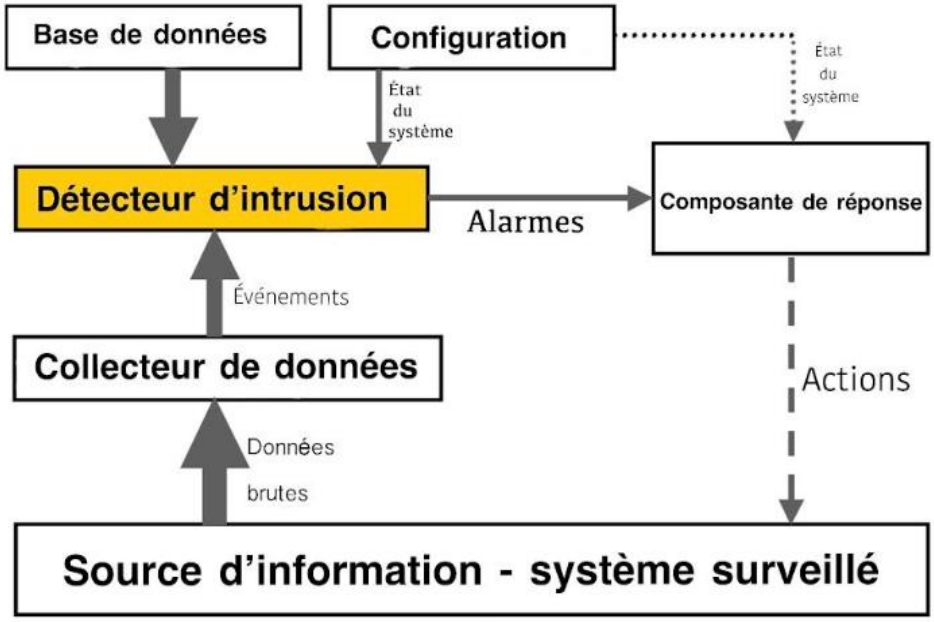

Figure I-18 : Architecture de base du système de détection d'intrusion (IDS)

- **Collecteur de données (capteur) :** est responsable de la collecte des données du système surveillé.
- **Détecteur d'intrusion :** traite les données collectées par les capteurs pour identifier les activités intrusives.
- **Base de données (base de connaissances) :** contient les informations collectées par les capteurs, mais sous un format prétraité. Ces informations sont généralement fournies par des experts en réseau et en sécurité, comme la base de données des attaques connues et leurs signatures**.**
- **Dispositif de configuration :** fournit des informations sur l'état actuel du système de détection d'intrusion.
- **Composant de réponse :** est chargé de déclencher des actions lorsqu'une intrusion est détectée. Ces réponses peuvent prendre deux formes : Réponses automatisées (actives) ou réponses avec interaction humaine (inactives).

4- Comparaison entre l'IDS et l'IPS :

La figure I-19 montre la différence entre un IPS et IDS, et les fonctionnalités qu'ils ont en commun.

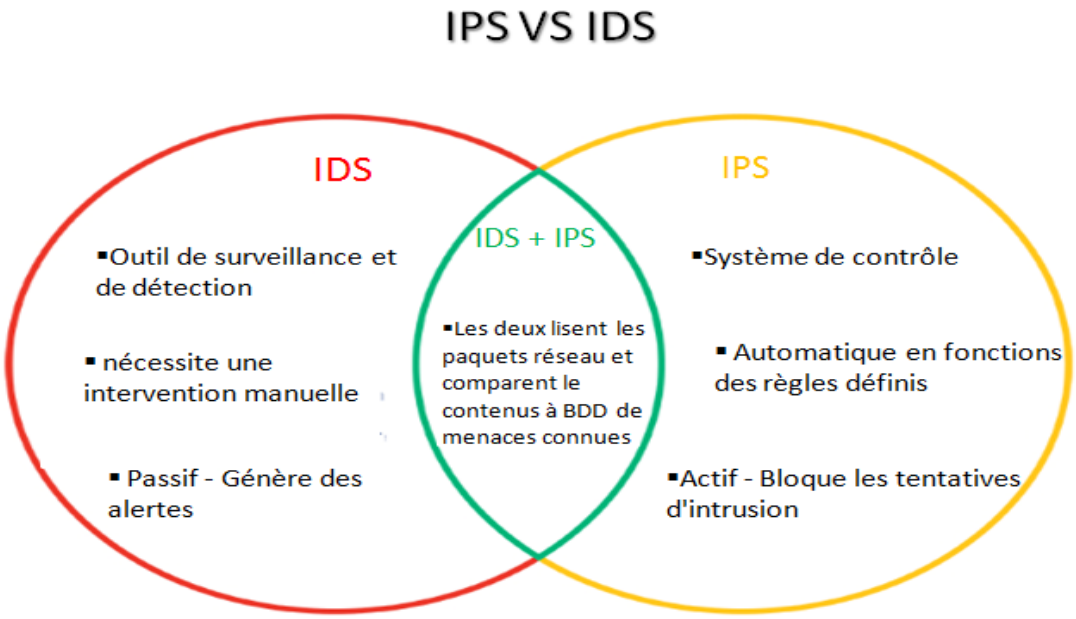

Figure I-19 : Comparaison entre L'IDS et L'IPS

I.8 Conclusion

La sécurité des réseaux est un domaine crucial qui nécessite une compréhension approfondie des vulnérabilités, des attaques potentielles et des mesures de prévention et détection de ces dernières, afin de garantir l'atteinte des objectifs de sécurité.

Ce chapitre nous amené à découvrir quelques types d'attaques réseau existantes et les vulnérabilités courantes soulignant l'importance de la sécurité informatique et les différentes mesures de protection mises en place pour prévenir et détecter ces attaques. Les mesures de sécurité ont été classées en deux classes : mesures préventives et mesures de détection réparation. La deuxième classe, en particulier les systèmes de détection d'intrusion (IDS), sera le sujet du chapitre suivant.

Chapitre II

Les systèmes de détection d'intrusion basés sur l'apprentissage automatique

II Les systèmes de détection d'intrusion basés sur l'apprentissage automatique

II.1 Introduction

Les systèmes de détection d'intrusion (IDS) sont des outils essentiels pour assurer la sécurité des réseaux, en identifiant les activités malveillantes. Cependant, ces systèmes sont confrontés à des défis majeurs tels que la détection des nouvelles attaques Zero-Day, les faux positifs (alertes des activités inoffensives) et les faux négatifs (non-détection d'un trafic malveillant). Pour surmonter ces limitations, l'intégration des techniques d'apprentissage automatique devient cruciale. Alors, comment ces techniques peuvent être utilisées pour améliorer la précision et l'efficacité des IDS ?

Dans ce chapitre, nous explorons le principe des IDS basés sur les méthodes classiques basées sur les signatures et celles basées sur l'apprentissage automatique en examinant les deux principales approches de cette dernière, à savoir l'apprentissage supervisées et non-supervisées, décrivant ainsi quelques algorithmes représentatifs pour chacune, suivis d'une comparaison entre eux.

II.2 Classification des systèmes de détection d'intrusion selon les méthodes de détection d'intrusion utilisées

Les systèmes de détection d'intrusion peuvent être classés en fonction de la méthodologie de détection d'intrusion utilisée. Les trois approches les plus courantes sont :

II.2.1 Détection par signature

Également connue sous le nom de détection d'intrusion par scénario, est une méthode de détection d'intrusion couramment utilisée pour identifier les activités malveillantes sur un réseau, en interrogeant une base de données de signatures (modèles) contenant un ensemble de caractéristiques des attaques connues telles que le numéro du port, la taille du paquet et le protocole employé, afin de les comparer à celles du trafic réseau. Une fois qu'une correspondance entre les deux signatures est détectée, l'analyste (l'expert en sécurité) tente de déterminer s'il s'agit d'un véritable incident (vrai positif), une erreur (faux positif) ou d'une attaque non identifiée (faux négatif) en examinant les données supplémentaires. Ensuite, il met à jour la base de signatures, puis déclenche une alerte pour réagir de manière appropriée aux menaces détectées [24].

En effet, cette technique est incapable de détecter les nouvelles attaques inconnues (e.g. les attaques zero day), car elle nécessite des mises à jour régulières de la base de données de signatures [30].

II.2.2 Détection par comportement

Également appelée détection d'intrusion par anomalies [31], cette approche, basée sur le Machine Learning (ML), repose sur l'analyse statistique de comportement de l'utilisateur ou d'une application pour détecter les intrusions dans un réseau, telles que les heures de connexion inhabituelles, le volume de données échangées, nombre excessif de tentatives de connexion avec des mots de passe incorrects.

Dans cette méthode, le système apprend le comportement normal des machines en les observant pendant une certaine période appelée phase d'apprentissage. Ce comportement de référence est enregistré dans la base de données et comparé avec le comportement présent des machines, et si une déviation entre ces comportements est observée, une alerte sera générée. Pour rester pertinent face à l'évolution naturelle des comportements des machines, le modèle de référence est mis à jour périodiquement par un processus de réapprentissage. Ce processus consiste à revoir et ajuster les critères de normalité en analysant les nouvelles données collectées, afin d'intégrer les changements de comportement légitimes tout en restant sensible aux anomalies potentiellement malveillantes.

La détection d'intrusion par comportement possède des avantages, tels que la capacité de repérer des attaques nouvelles et inconnues (zero day attacks), ainsi que son adaptation aux changements du comportement du système. Cependant, elle présente des limites en termes de précision et de fiabilité. Les alertes générées manquent souvent de détails sur les attaques et nécessitent une analyse approfondie et complémentaire, tandis que les changements de comportement normaux peuvent déclencher des alertes inutiles (faux positifs). De plus, la qualité des données utilisées pour l'apprentissage est cruciale pour une détection efficace des intrusions mais cela n'est pas toujours garanti.

II.2.3 Détection hybride

La méthode hybride de détection d'intrusion est souvent utilisée en pratique [32], elle permet de surmonter les limites des approches précédentes en combinant les avantages de chaque méthode afin de créer un système de détection d'intrusion plus robuste et plus efficace.

Cette technique permet de détecter les attaques connues en utilisant les méthodes basées sur les signatures tout en identifiant les comportements anormaux à travers des approches basées sur les anomalies, réduisant ainsi les faux positifs et améliorant la sécurité globale.

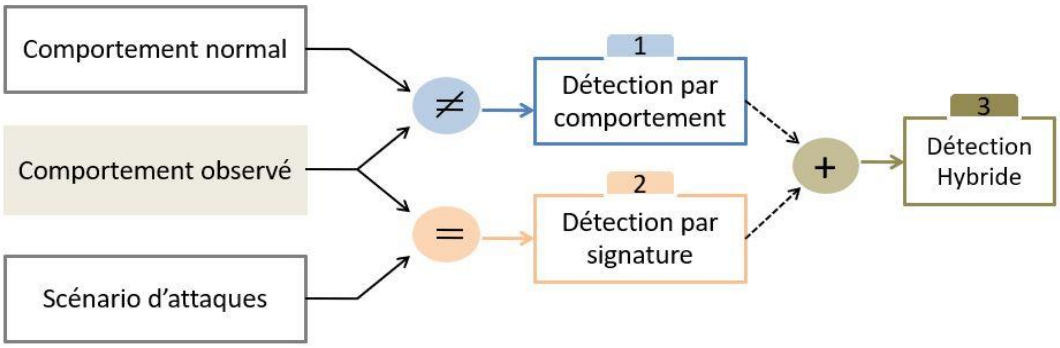

Figure II-1 : Schéma explicatif des approches de détection d'intrusion

II.3 IDS basé sur l'apprentissage automatique

Un IDS basé sur l'apprentissage automatique est un système de détection d'intrusion qui emploie des techniques d'apprentissage automatique pour modéliser le comportement normal d'un système ou d'un réseau (détection par comportement). Il fonctionne en deux phases clés : la phase d'apprentissage où le système collecte des données sur le trafic réseau et les activités normales du système. Des algorithmes d'apprentissage automatique sont appliqués pour construire un modèle statistique du comportement normal. La phase de détection consiste à comparer le trafic en temps réel au modèle appris. Toute déviation statistiquement significative du comportement normal est considérée comme une activité suspecte potentiellement malveillante et déclenchera une alerte.

Grâce à cette approche basée sur le comportement, les IDS peuvent détecter des attaques inconnues et des menaces évolutives, sans nécessiter de définitions de signatures d'attaques préalables, l'apprentissage automatique donc offre la capacité d'adaptation et de généralisation nécessaire pour faire face à la complexité et à l'évolution constante des menaces.

II.3.1 Apprentissage automatique

L'apprentissage automatique, également appelé Machine Learning (ML) en anglais, est une méthode d'analyse de données qui consiste à construire un modèle à partir de données grâce à l'utilisation d'un algorithme d'apprentissage automatique. Ce modèle doit être doté d'une bonne

propriété de généralisation des données d'apprentissages, i.e. fonctionner au mieux sur de nouvelles données [33].

II.3.2 Apprentissage automatique pour la détection d'intrusion

Considérons les éléments suivants [34] :

- E représente une expérience d'utilisation d'un réseau contenant des données de trafic normal et des cyberattaques, enregistrées et étiquetées ;
- T est une tâche de classification qui consiste à prédire le type de trafic ;
- P une métrique telle que la précision ;

Un système de détection d'intrusions réseau utilisant une technique d'apprentissage automatique est un programme informatique qui apprend, à partir de l'ensemble E des données enregistrées sur un réseau, à réaliser la tâche T de classification en trafic normal ou en attaques, en optimisant la métrique P choisie, dans le but d'améliorer la précision de détection des intrusions dans le réseau.

II.3.3 Fonctionnement d'un IDS basé sur l'apprentissage automatique

Les systèmes de détection d'intrusion (IDS) qui utilisent des techniques d'apprentissage automatique fonctionnent généralement selon un schéma en plusieurs étapes clés :

1- **Collecte de Données :** l'IDS commence par collecter et stocker les données sur le trafic réseau.

2- **Détection par Comportement**

   a) **Division des Données en Test et Apprentissage :** les données collectées sont divisées en deux ensembles : un ensemble d'apprentissage et un ensemble de test. L'ensemble d'apprentissage sert à entraîner les algorithmes ML, tandis que l'ensemble de test permet d'évaluer les performances du modèle.

   b) **Application des Algorithmes ML sur les Données d'Apprentissage :** des algorithmes d'apprentissage automatique, sont appliqués à l'ensemble d'apprentissage. Ces algorithmes apprennent à détecter les motifs de comportement normal et anormal.

   c) **Création du Modèle :** à partir de l'apprentissage, un modèle prédictif est créé pour identifier les activités anormales dans le trafic réseau.

   d) **Test du Modèle :** le modèle créé est testé sur l'ensemble de données de test pour évaluer ses performances de détection.

e) **Déploiement du Modèle :** une fois les performances jugées satisfaisantes, le modèle est déployé en production pour la détection d'intrusions.

3- **Système de Détection par Signature :** en parallèle, l'IDS utilise également un module de détection par signatures d'attaques connues, afin de réduire l.

4- **Génération d'Alertes :** lorsqu'une correspondance avec une signature d'attaque est détectée, l'IDS génère une alerte.

La figure II-2 illustre un schéma de fonctionnement d'un IDS basé sur l'apprentissage automatique.

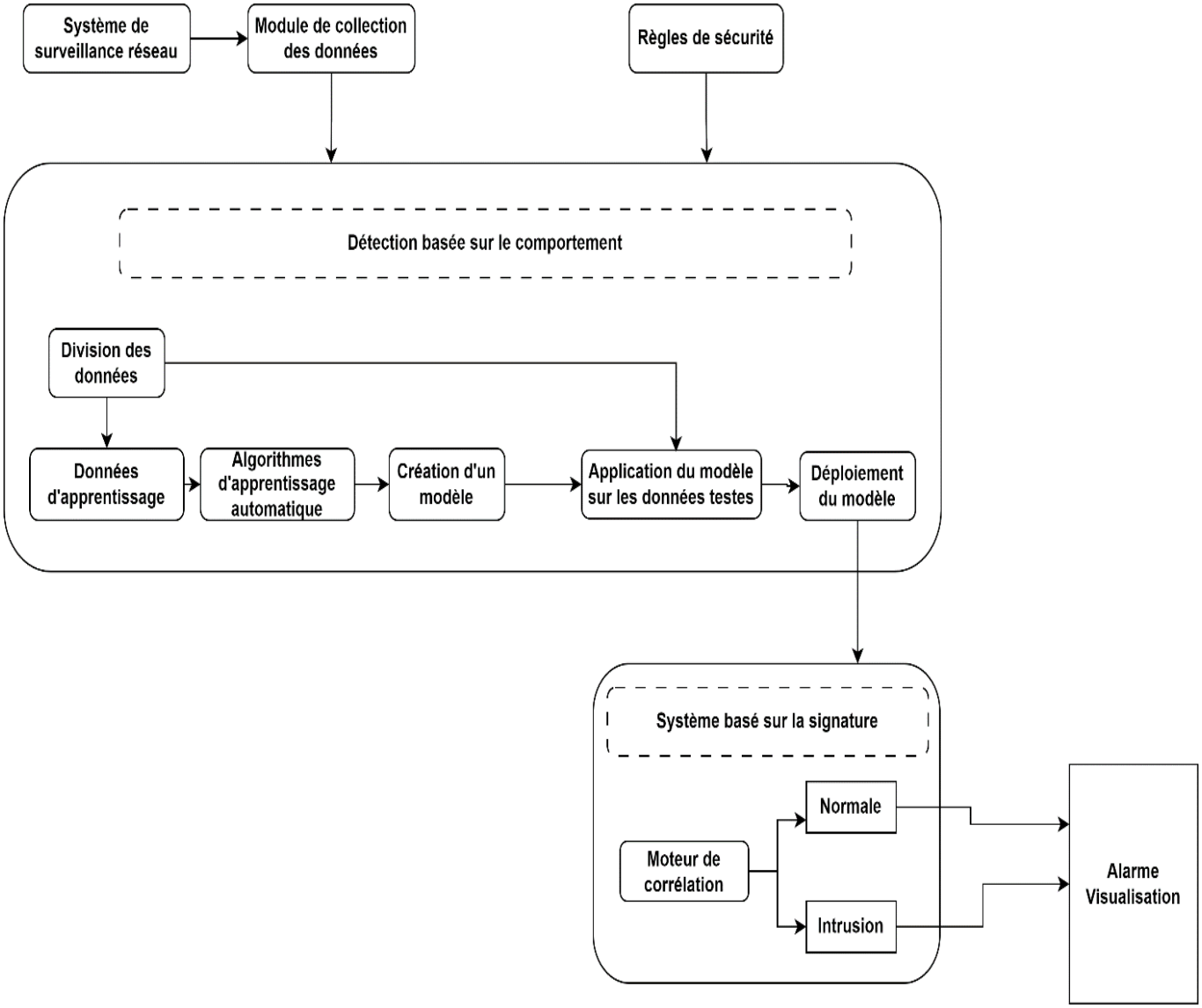

Figure II-3 : Schéma de fonctionnement d'un IDS basé sur l'apprentissage automatique

II.4 Les algorithmes d'apprentissage automatique appliqués à la détection d'intrusion

La catégorisation des méthodes de ML implique l'utilisation des algorithmes qui décrivent le processus de chaque approche, principalement pour les big data. Parmi les algorithmes les plus couramment utilisées on peut mentionner : les algorithmes non supervisés et supervisés.

II.4.1 Les algorithmes non supervisés

Les algorithmes non supervisés visent à analyser un ensemble de données pour en extraire un nombre de caractéristiques qui les décrivent suffisamment, ainsi que leurs structures cachées. Un des exemples les plus populaire des algorithmes non-supervisés est le regroupement (clustering), qui consiste à regrouper dans un même cluster (groupe) des données considérées similaires, sans avoir de connaissances préalables sur les classes existantes (données non étiquetées). L'objectif est de minimiser la distance entre les objets d'un même cluster et de maximiser la distance entre les clusters [35].

Les algorithmes non supervisés les plus utilisés sont : K-moyenne, Fuzzy c-means, Partitionnement hiérarchique et DBSCAN.

1- L'algorithme de K-moyennes

Également connue sous le nom de K-means en anglais, est une méthode de partitionnement de données qui vise à regrouper un ensemble d'instances en K clusters (K étant généralement un hyperparamètre) en minimisant les variances intra-cluster (entre les instances de même cluster), tout en maximisant la distance inter-cluster.

L'algorithme K-moyennes se déroule en 3 étapes principales [25] :

Etape 1 : initialisation de nombre de clusters K et choix aléatoire des K centroïdes (points centraux) de ces clusters.

Etape 2 : calcul de la distance entre chaque instance et les centroïdes, attribuant à chaque instance l'étiquète de centroïde le plus proche.

Etape 3 : une fois la première classification est réalisée, l'algorithme K-means recalcule de centroïde (le centre de gravité du cluster) et réexécute les étapes 2 et 3 jusqu'à est ce qu'un critère d'arrêt soit atteint, (le nombre maximal d'itérations est atteint ou les classes ne montrent plus aucun changement après un certain nombre d'itérations).

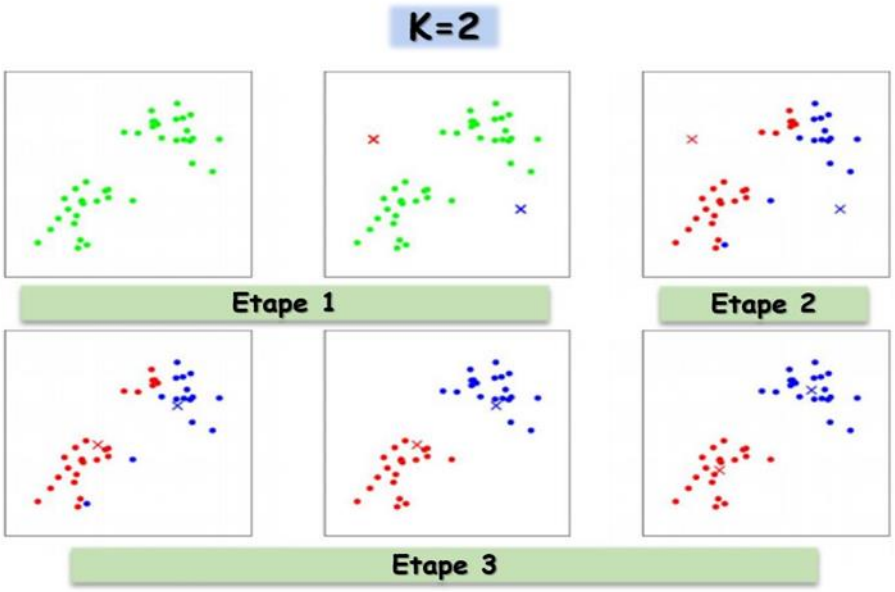

Figure II-4 : Exemple illustratif d'une classification par l'algorithme K-moyenne

Avantages

L'approche K-moyennes offre plusieurs avantages, parmi lesquelles on peut citer [36] :

- La facilité d'implémentation et de compréhension.
- L'assurance d'une meilleure performance du classifieur en évitant l'étape d'apprentissage, qui est généralement très couteuse en termes du temps, ce qui le rend particulièrement adapté pour l'analyse de données volumineuses (big data).
- Bonnes performances pour le traitement de données négatives clairsemées (sparce).

Inconvénients

Bien que cet algorithme présente des avantages, il comporte également des inconvénients tels que [36] :

- La difficulté de sélectionner les bons paramètres du modèle tel que la valeur de K, pour garantir une détection d'intrusion.
- La convergence vers des optimaux locaux en cas de mauvais choix des centroïdes initiaux.
- La qualité du partitionnement dépend des valeurs initiales des centroïdes, qui sont sélectionnés d'une manière aléatoire.

Dans ce sillage, plusieurs travaux connexes ont été proposés dans la littérature :

Zhang et al. [37] ont introduit un modèle de pipeline basé sur l'algorithme des K-moyennes pour la détection d'intrusions dans les réseaux (STG2P : *A Two-Stage Pipeline Model for Intrusion Detection Based on Improved LightGBM and Reinforced K-means*). En effet, ce modèle combine l'algorithme des K-moyennes et LightGBM (*Gradient-boosting framework for machine learning*), une plateforme open-source pour l'apprentissage automatique basée des algorithmes d'apprentissages automatiques, notamment, le Gradiant boosting, les arbres de décision, et les forêts d'arbres décisionnels. Le principe de base consiste à améliorer la classification initiale et imprécise issue du LightGBM par l'application du K-moyennes**.** Ainsi, l'utilisation de l'algorithme des K- moyennes permet de réduire les faux positifs identifiés lors de la classification initiale.

En outre, Chen et al. [38] ont présenté la méthode QALO-K, une amélioration de l'algorithme K-moyennes. Cette approche combine l'algorithme K-moyennes avec un algorithme d'optimisation basé sur l'heuristique Quantum-Ant-Lion, assurant ainsi une convergence vers un optimum global.

2- Fuzzy c-means (FCM)

Le Fuzzy C-means (FCM), appelé également fuzzy clustering [39] [40], est une version améliorée de l'algorithme K-moyennes qui permet à une instance d'appartenir à plusieurs clusters avec degré d'appartenance spécifique. Comme les autres algorithmes de classification non supervisé, FCM utilise une fonction d'objectif pour minimiser les distances intra-cluster et maximiser les distances inter-cluster, tout en attribuant un certain degré d'appartenance à chaque cluster pour chaque instance. Cet algorithme nécessite la connaissance préalable du nombre de clusters et génère les classes par un processus itératif en minimisant la fonction d'objective.

Les étapes de l'algorithme FCM sont les suivantes :

Etape 1 : Initialisation d'une matrice d'appartenance à n lignes et c colonnes, où n est le nombre d'instances et c est le nombre de clusters. Chaque élément de la matrice, $w_k(x)$, représente le degré d'appartenance du l'instance x au cluster k.

Etape 2 : Le calcul des centroïdes des clusters :

$$C_k = \frac{\sum_{x \epsilon D} (w_k(x))^m \, x}{\sum_{x \epsilon D} (w_k(x))^m} \qquad \text{(II-1)}$$

Tel que m est un paramètre de fuzziness ;

Etape 3 : Mise à jour de la matrice d'appartenance en calculant le degré d'appartenance $w_k(x)$ en fonction de la distance entre le point de donnée et le centroïde du cluster.

$$w_k(x) = \frac{1}{\sum_{i=1}^{c} \left(\frac{\|x - c_k\|}{\|x - c_i\|} \right)^{\frac{2}{m-1}}} \quad \text{(II-2)}$$

Etape 4 : Répétition des étapes 2 et 3 jusqu'à convergence de la fonction objectif, qui est minimisé à la convergence de l'algorithme, i.e. lorsque les centroïde des clusters ne changent plus.

Avantage

- Permet de réaliser un clustering à fonction objective non-convexe.

Inconvénients

L'algorithme Fuzzy C-means présente certains inconvénients [41], notamment :

- La difficulté à gérer les points aberrants, ce qui peut entrainer des résultats imprécis dans le clustering.
- Dans un cluster, l'appartenance d'un point de données dépend directement des valeurs d'appartenance des autres centroïdes de cluster, ce qui peut conduire à des résultats indésirables au sein d'un cluster.
- Difficulté à traiter des ensembles de données de grande dimension, en plus d'être sensible à l'initialisation. Mauvais choix peut conduire à des solutions locales non optimales.

Dans ce contexte, Ganapathy et al. [42] ont développé un modèle de Fuzzy C-Means pour la détection des partitions dans les bases de données dédiées à la détection d'intrusions réseau, en intégrant des algorithmes génétiques pour améliorer les performances de l'algorithme C-Means dans les ensembles de données à très grandes dimensions.

3- Partitionnement hiérarchique

Le principe de cette approche est de construire une arborescence appelée dendrogramme, qui représente les différents regroupements des données. Il existe principalement deux types d'approche de partitionnement hiérarchique [43] **:** approche agglomérative et divisive.

a- Approche agglomérative (ascendante) : consiste à initialiser chaque instance comme un seul cluster (singletons), puis à fusionner les deux clusters les plus proches en un seul cluster. Ce processus est répété jusqu'à ce qu'un seul cluster soit obtenu à la fin.

b- Approche divisive (descendante) : cette approche consiste en revanche à regrouper l'ensemble des données en un seul cluster au départ qui sera divisé jusqu'à l'obtention des clusters contenant chacun une seule instance.

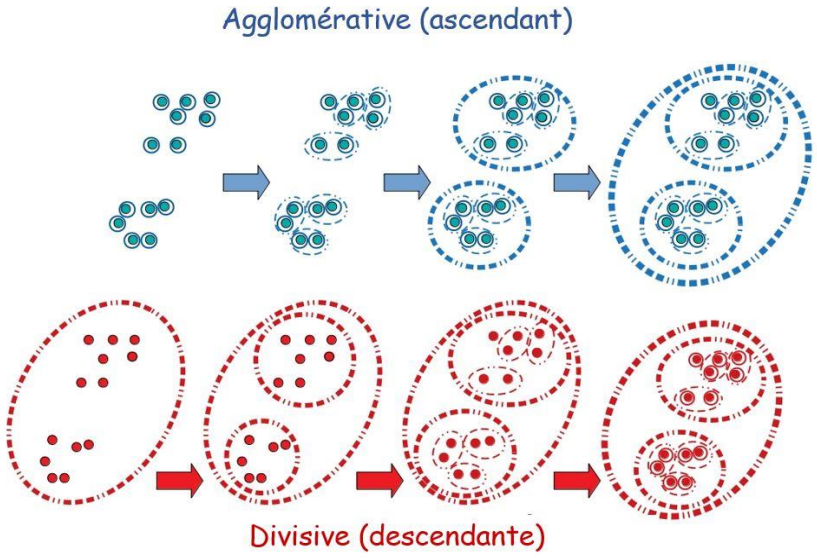

Figure II-5 : Classification par partitionnement hiérarchique

4- Density-Based Spatial Clustering of Applications with Noise (DBSCAN)

Le DBSCAN un algorithme de clustering de données proposées en 1996 par Ester et al. [44] dont l'objectif est de capturer les zones de fortes densités pour former des clusters, tout en identifiant les points (instances) n'appartenant pas à ces zones comme des valeurs aberrantes. DBSCAN nécessite deux paramètres pour être applicable : Eps et MinPts.

- **MinPts** : le nombre minimum de points requis pour former une région dense (un cluster).
- **Eps** : également noté ε, est une mesure de distance utilisée pour localiser les points situés dans le voisinage d'un point donné.

Selon l'algorithme DBSCAN et en fonction de ces deux paramètres, trois catégories de points sont identifiées : points centraux, points frontières et points de bruits, comme c'est montré dans Figure II-5.

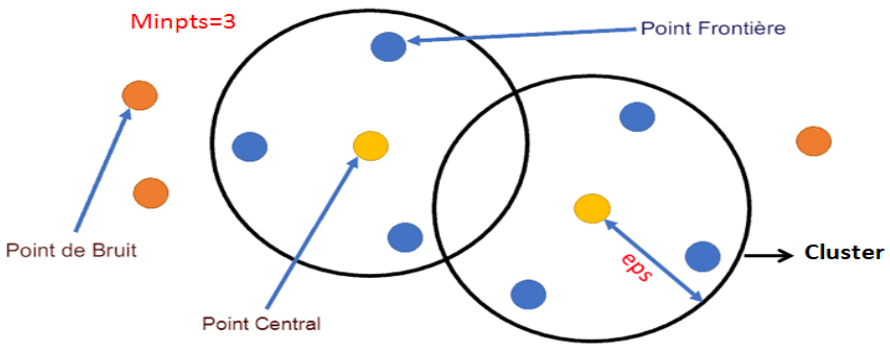

Figure II-6 : Les types des points de données dans l'algorithme DBSCAN

- **Points centraux (les centroïdes) :** sont des points qui possèdent au moins des points MinPts dans sa distance Eps.
- **Points frontières :** sont les points voisins de point centrale, situés à la limite d'un cluster.
- **Points de bruits :** appelés aussi points aberrants, sont des points qui ne font partie d'aucun cluster, car elles ne possèdent pas suffisamment de points voisins dans leurs voisinage eps pour être considérés comme des points centraux ou frontières.

L'algorithme DBSCAN se déroule en quatre étapes :

Étape 1 : DBSCAN choisit un point arbitrairement qui n'a pas encore été visité dans l'ensemble de données.

Étape 2 : détermination de voisinage Eps en extrayant tous les points situés à une distance Eps du point sélectionné. Si ce voisinage contient au moins un nombre minimum de points (MinPts), alors le point initial est considéré comme un point central et un cluster est créé.

Étape 3 : Si un point P fait partie d'un cluster C, alors tous les points qui sont densité atteignable à partir de P font également partie de C. Plus précisément, si un point Q est densité atteignable à partir de P, cela signifie qu'il existe une chaine de point P_1 à P_n, avec $P_1 = Q$ et $P_n = P$, telle que la distance entre chaque paire de points consécutifs Pi et Pi_{+1} soit inférieur ou égale à Eps.

Étape 4 : ces étapes sont répétées jusqu'à ce que tous les points soient visités. Les points qui ne font pas partie d'un cluster sont identifiés comme des points de bruit.

Le choix des deux paramètres Eps et MinPts est essentiel pour garantir une détection précise des anomalies par l'algorithme DBSCAN. La Figure II-6 illustre l'influence de ces deux paramètres sur la capacité de DBSCAN pour identifier les anomalies.

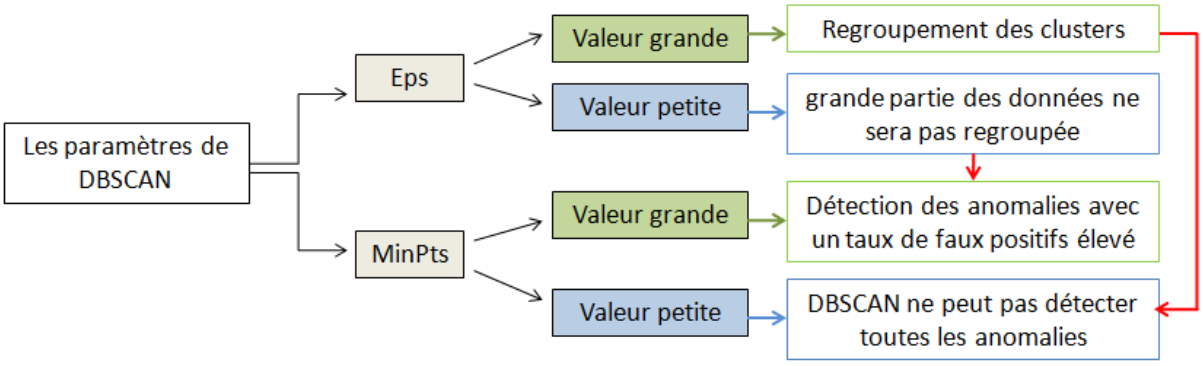

Figure II-7 : Schéma illustratif de l'effet des paramètres Eps et MinPts sur la détection des anomalies par DBSCAN

Par conséquent, il est important de choisir de bonnes valeurs pour les paramètres Eps et MinPts afin de minimiser les faux positifs et de maximiser la détection des anomalies.

Avantages

- Capacité à trouver des clusters de formes non sphérique (arbitraires) sans spécifier le nombre de clusters à l'avance.
- Flexibilité dans la détection de structures de densités différentes.

Inconvénients

- Les données des réseaux sont souvent dynamiques et changeantes dans le temps, ce qui peut poser des défis pour l'algorithme DBSCAN. De plus, le faible taux des données d'attaques par rapport aux comportements normaux peut entrainer un biais de DBSCAN vers la partition des données des comportements normaux, limitant ainsi sa capacité à détecter efficacement les attaques.
- L'évolution rapide des données réseau et l'émergence de nouvelles attaques (Zero-day) peuvent également compliquer la détection des attaques réseau via DBSCAN. De plus, dans le cas de partitionnement dynamique, les DBSCAN produit généralement plusieurs clusters qui ne sont pas bien séparés.

Dans ce contexte, Alfoudi et al. [45] ont proposé un modèle d'hyper-clustering basé sur DBSCAN. La principale contribution consiste à améliorer le DBSCAN en ajoutant un nouveau processus évolutif basé sur la mesure de distance inter-cluster afin de contourner le problème de données à classes déséquilibrées.

-La première étape consiste à utiliser le DBSCAN pour obtenir un partitionnement primaire des données. Ensuite, les silhouettes des clusters obtenus sont calculées selon l'équation suivante :

$$SIL = \frac{1}{n}\sum_{i=1}^{k}\sum_{j=1}^{m}\frac{x_j - u_k}{max\{x_j, u_k\}} \quad \text{(II-3)}$$

Tels que, k : nombre de clusters ; m : taille du cluster i ; x_j: une donnée appartenant au cluster i ; u_k: le plus proche centroïde du point x_j.

-Les clusters sont classés en selon les valeurs de leurs silhouettes (deux valeurs seuils sont nécessaires) en trois classes :

- Pool primitif ayant une bonne caractéristique de partitionnement (les données ont des propriétés très similaires).
- Les partitions à qualités moyennes (Pool positif).
- Les partitions à qualités inférieurs (Pool négatif).

- A l'étape de redistribution, tous les points du Pool négatif sont redistribués sur les clusters positifs selon des valeurs d'appartenance (floues) décrites par la fonction cosinus suivante :

$$Sim\,(X, Ci) = \frac{X.C_i}{\|X\|.\|C_i\|} \quad \text{(II-4)}$$

Tel que X : est un vecteur de caractéristiques d'un point de donnée, C_i: le vecteur de caractéristiques du centroïde du cluster i.

Ce modèle d'hyper-clustering offre plusieurs avantages, notamment :

- Amélioration de la qualité de partitionnement ainsi que le taux de détection des attaques de la classe minoritaire dans l'ensemble des données.
- Réduction de la complexité temporelle.
- Le classifieur permet l'apprentissage d'un modèle sur des clusters (apprentissage sur des sous-ensembles de données) afin de prédire efficacement les attaques, plus particulièrement, les classes d'attaques minoritaires.

En outre, Casas et al. [46] ont présenté un système de détection d'intrusion (UNIDS, *Unsupervised Network Intrusion Detection System*), qui vise à repérer les attaques réseau inconnus sans l'utilisation des signatures. Ce système repose sur la détection d'activités aberrantes (anormales) en utilisant la méthode de clustering des sous-espaces et l'accumulation de preuves multiples (fusion de données) pour détecter les différents types d'attaques réseau. En pratique, en considérant une matrice de caractéristiques $X \in \mathbb{R}^{n \times m}$, où n représente le nombre d'instances et m le nombre de caractéristiques. Le processus se déroule en deux étapes principales :

1. Extraire les N sous-espaces $X_i \subseteq X$, $n \in \{1, ..., N\}$ par la projection de X sur k caractéristiques.
2. Partitionnement de chaque sous-espace en utilisant l'algorithme DBSCAN, générant ainsi un ensemble (i) clusters $\{C_1^i, C_2^i, \ldots, C_{p(i)}^i\}$ et $q(i)$ outliers $\{o_1^i, o_2^i, \ldots, o_{q(i)}^i\}$ pour chaque sous ensemble.

Ce système présente des avantages tels que :

- L'utilisation de valeurs $k \ll m$ et le partitionnement sur des sous-espaces à dimensions inférieures est plus efficace et permet une convergence plus rapide.
- Il est prouvé que DBSCAN permet une meilleure qualité de clustering dans des espaces de petites dimensions.
- Les espaces de grandes dimensions présentent généralement une distribution clairsemée (faible densité), ce qui rend difficile la séparation entre les régions dense et clairsemée, ainsi UNIDS permet une l'amélioration des performances de DBSCAN.
- L'utilisation des sous-espaces multiples permet la détection des attaques ayant des caractéristiques très différentes.

II.4.2 Les algorithmes supervisés

L'apprentissage supervisé [34] est un domaine fondamental de l'apprentissage automatique, caractérisé par l'utilisation de données étiquetées pour entraîner des algorithmes à réaliser des tâches de classification ou de régression précises. En classification, des étiquettes sont attribuées à de nouvelles instances parmi un ensemble fini de valeurs, tandis qu'en régression, des étiquettes sont représentées par des valeurs continues.

Les algorithmes d'apprentissage supervisé apprennent à partir d'exemples d'entrées (variables X) et de sorties souhaitées (variables Y) pour maîtriser une fonction de mappage entre les entrées et les sorties, ce qui permet au modèle de prédire correctement les sorties pour de nouvelles données non étiquetées ayant la même distribution sous-jacente que les données d'apprentissage, assurant ainsi une meilleure généralisation. Les algorithmes supervisés les plus utilisés sont : K plus proche voisins, Machine à vecteur de support, Classifieur Bayésien Naïf**,** L'arbre de décision**.**

1- K plus proches voisins

L'algorithme k plus proches voisins ou K- Nearest Neighbors (KNN) en anglais, est l'un des algorithmes les plus simple***s*** en apprentissage automatique. Son principe de base consiste à identifier, dans un ensemble de données d'apprentissage étiquetées, les instances qui sont plus similaires à une nouvelle instance à classer (instance de test), en se basant sur la distance d qui les sépare, plus cette distance est faible, plus les instances sont considérées comme similaires. Enfin, K instances plus proches sont sélectionnées afin d'évaluer la classe de la nouvelle instance, et l'étiquète de la plus fréquente parmi ces K instances est attribuée à cette dernière [47].

La figure-II-7 illustre un exemple explicatif de fonctionnement de cet algorithme.

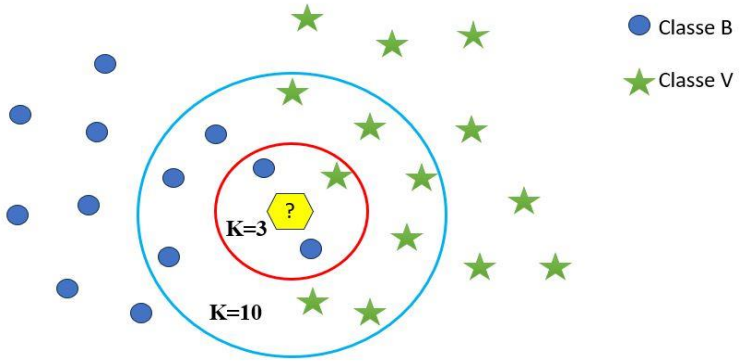

Figure II-8 : Exemple explicatif d'une classification avec l'algorithme KNN

La nouvelle instance à classer est représentée par l'hexagon en jaune. Avec un nombre de voisins K=3 (cercle rouge), deux sont de la classe B et un de la classe V, donc la nouvelle instance sera classée dans la classe B (la plus dominante). En revanche, dans le cas où le nombre de voisins K=10 (représenté par le cercle bleu), la majorité des 10 plus proches voisins appartiennent à la classe V, ce qui conduit à la classification de la nouvelle instance dans la classe V.

La performance de l'algorithme KNN dépend de la distance d minimale qui peut être calculée à l'aide d'une fonction de distance telle que [34] :

- **La fonction de Manhatan :** $d1(x,y) = \sum_{i=1}^{n}|x_i - y_i|$
- **La fonction de Minkowski :** $d2(x,y) = (\sum_{i=1}^{n}|x_i - y_i|^p)^{1/p}$
- **La fonction euclidienne :** $d3(x,y) = \sqrt{\sum_{i=1}^{n}(x_i - y_i)^2}$ (cas particulier de la distance de Minkowski pour p=2).
- **La distance de Tchebychev :** $d_4(x,y) = \max_{i\in[\![1,n]\!]}|x_i - y_i|$

Avec : les coordonnées i ème des deux vecteurs d'instances x et y de dimensions n.

Avantages

- Simplicité d'implémentation et de compréhension [47].
- Le choix de la distance et de la valeur de K permet l'adaptation de l'algorithme aux différents types de données.

Inconvénients

- La valeur de k a un impact significatif sur les résultats, un mauvais choix de celui-ci peut conduire à une mauvaise classification [34].
- La performance du KNN peut se dégrader dans des espaces à haute dimension, où la distance entre les points devient moins significative.

Dans ce contexte, Li et Guo [48] ont proposé un modèle de détection d'intrusions basé sur l'algorithme K-NN et une méthode de sélection des données d'apprentissage basée sur l'apprentissage actif : TCM-KNN (*Transductive Confidence Machines for K-Nearest Neighbors*). Ce modèle utilise l'apprentissage actif pour sélectionner les instances de données pertinentes, ce qui permet une meilleure optimisation du modèle.

Plus formellement : Soit $\{(x_1, y_1), (x_2, y_2), \ldots, (x_n, y_n)\}$ ensemble d'entrainement de n éléments tels que $x_i = \{x_i^1, \ldots, x_i^k\}$ est l'ensemble de caractéristiques d'une donnée x_i, et y_i est la classe du point x_i. Soit D_i^y la séquence triée par ordre croissant des distances d'un point x_i des autres points appartenant à une classe y, D_i^{-y}la séquence triée par ordre croissant des distances d'un point x_i

des autres points appartenant à une classe autre que y, D_{ij}^{y} représente la j ème distance la plus courte dans cette séquence. D_{ij}^{-y} représente la j ème distance dans la séquence triée des distances des points appartenant à des classes différentes de y.

L'idée de base consiste à chercher, pour chaque point x_i et pour chaque classe y l'ensemble d'entrainement D_i^y et D_{ij}^{-y}, ensuite un facteur de mesure d'étrangeté est calculé sous la forme $\alpha_{ij} = \frac{\sum_{j=1}^{k} D_{ij}^{y}}{\sum_{j=1}^{k} D_{ij}^{-y}}$.

Pour chaque point test x_r, le vecteur de distance $distr \in \mathbb{R}^n$de toutes les instances de données d'entrainement est calculé, (sachant que $distr[j]$ est la distance entre le point test x_r et l'instance d'entrainement x_j).

Pour chaque classe y, pour chaque point d'entrainement x_r et pour chaque point d'entrainement x_t:

Si $D_{tk}^{y} > dist_{\mathrm{r}}[t]$ ou si $D_{tk}^{y} > dist_{\mathrm{r}}[t]$ alors recalculer α_{ty}, tout en considérant x_r comme étant de classe y (notons la nouvelle valeur α_{ty}^{new}).

La valeur $\rho_y = \frac{\#\{t:\alpha_{ty} > \alpha_{ty}^{new}\}}{n+1}$ est calculée (la classe de x_r est la classe maximisant ρ_y).

Ce modèle offre un avantage en permettant la détection d'anomalies avec une précision élevée et un faible nombre de faux positifs, tout en utilisant un nombre réduit de données et de caractéristiques d'entrainement.

2- Machine à vecteur de support

La machine à vecteur de support, appelée également *Support Vector Machine* (SVM) en anglais, est une méthode de classification binaire introduite par Vapnik et ses collaborateurs en 1992 [49]. Son objectif est de trouver un classificateur linéaire appelé hyperplan, qui sépare de manière optimale un ensemble de données en deux classes distinctes.

Formellement, Soit un ensemble d'apprentissage $S = \{(x1, y1), \dots, (xn, yn)\}$ de données $xi \in \mathbb{R}n$ avec leurs étiquettes correspondantes $yi \in Y$, $Y \in \{+1, -1\}$, générées selon une distribution inconnue.

L'objectif principal de SVM est d'apprendre une fonction $g : X \rightarrow Y$ permettant une classification correcte de nouvelles instances (x, y) générées selon la même distribution sous-jacente que celle des données d'apprentissage, i.e. assurer une meilleure généralisation.

Soit $\varphi : I \subseteq \mathbb{R}n \rightarrow F \subseteq \mathbb{R}n$ un mappage de l'espace $I \subseteq \mathbb{R}n$ à l'espace de caractéristiques $F \subseteq \mathbb{R}n$. SVM considère une fonction correspondant à un hyperplan $< w, \phi(x_i) \geq = b$ pour séparer les données d'entrainements X selon leurs étiquettes tout en maximisant la marge, i.e. la distance entre l'hyperplan et les points les plus proches de chaque classe (les vecteurs de support).
Cette marge est définie par :

$$\gamma = min_{1 \leq i \leq l}\, y_i(< w, \phi(x_i) > -b) = min_{1 \leq i \leq l} \quad y_i\, g(x_i) \qquad \text{(II-5)}$$

Avec <,> est le produit intérieur ; w: vecteur de dimension ; l, b : seuil.

La distance entre x_i et l'hyperplan est donnée par la formule suivante : $\frac{(<w,\phi(xi)>-b)}{\|w\|}$.

Un exemple de classification binaire en deux dimensions est illustré par la figure II-8.

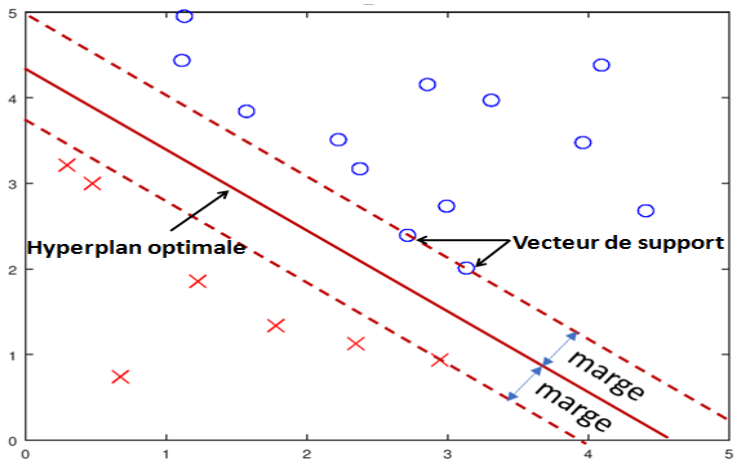

Figure II-9 : Exemple de classification binaire en deux dimensions avec l'algorithme SVM

Lorsque les données sont linéairement séparables, Les SVM peuvent facilement trouver un classificateur linéaire de manière directe. Cependant, dans de nombreux cas réels, il n'est pas possible de séparer linéairement les données, rendant l'utilisation du classificateur de marge maximale inefficace, car il fonctionne seulement si les classes de données d'apprentissage sont linéairement séparables.

Malgré ces limitations, SVM reste une option valable même dans le cas des données non linéairement séparable. L'idée est de projeter les données dans un espace de plus grande dimension, dans l'espoir que ces données soient linéairement séparables. Cela permet de trouver un hyperplan optimal dans cet espace transformé. Comme illustré dans la figure II-9 :

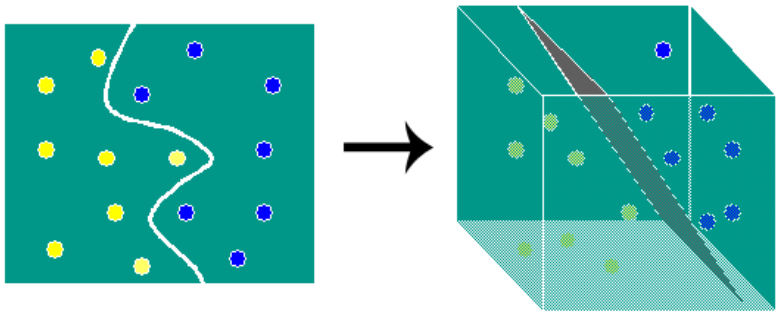

Figure II-10 : classification des données non linéairement séparables avec

Dans ces cas, les transformations non linéaires se font par l'utilisation de noyaux (Kernel) :

- Noyaux basés sur une fonction polynomiale : $K(xi, xj) = (< xi, xj > +1)^d$
- Fonction à base radiale gaussienne : $K(xi, xj) = exp\,(\frac{-\|xi - xj\|^2}{\delta^2})$
- Fonction RBF (*Radial Basis Function*) : $K(xi, xj) = exp\,(-Y\,\|xi - xj\|^2)$

Avantages

- Les SVM offrent de très bonnes performances en termes de précision.
- Capacité de gérer des données non linéaire grâce à l'utilisation de noyaux.
- Robustesse aux données aberrantes grâces à la marge maximale.
- En plus de la classification, les SVM peuvent également être utilisés pour des tâches de régression.

Inconvénients

- SVM a une complexité temporelle élevée.
- Les techniques de SVM sont difficilement utilisables pour des ensembles de données de grandes dimensions [34].

Pour améliorer la classification (précision) et réduire le temps d'apprentissage des SVM dans la détection des intrusions réseau, Wang et al. [50] ont proposé un Framework LMDRT-SVM (*Logarithm Marginal Density Ratios Transformation*) basé sur deux étapes principales :

Étape 01 : La première étape consiste à appliquer la transformation des Rapports de Densité Marginale Logarithmique sur les données d'apprentissage originales afin d'obtenir des données transformées de meilleure qualité en termes de séparabilité et capacité de classification par des modèles basés sur SVM. Le processus est le suivant :

a) D'abord, l'ensemble de données d'apprentissage est partitionné aléatoirement en deux sous-ensembles : $S_1 = (X^{(1)}, Y^{(1)})$, $S_2 = (X^{(2)}, Y^{(2)})$, $S^{(1)} \cap S^{(2)} = \emptyset$, $S^{(1)} \cup S^{(2)} = S$.

b) Soit les sous-ensembles, $X^{(1)+}$ et $X^{(1)-}$définis par : $X^{(1)+} = \{X^{(1)} : Yi^{(1)} = +1\}$ et $X^{(1)-} = \{X^{(1)} : Yi^{(1)} = -1\}$

c) Une estimation de densité par noyau est appliquée à ces deux sous-ensembles. On note par $\hat{f} = (\hat{f}_1, \hat{f}_2, \ldots, \hat{f}_p)$, $\hat{g} = (\hat{g}_1, \hat{g}_2, \ldots, \hat{g}_p)$ les estimations de S_1.

$$\hat{f}_J = \frac{1}{N1^+ h} \sum_{i=1}^{Ni^+} K\left(\frac{Xij^{(1)+}}{h}\right) \quad ; \quad \hat{g}_J = \frac{1}{N1^- h} \sum_{i=1}^{Ni^-} K\left(\frac{Xij^{(1)-}}{h}\right)$$

Tels que : $N1^+$= card ($X^{(1)-}$) ; h : paramètre de lissage ; K(.) : la fonction noyau.

d) Les transformations de LMDRT sont appliquées sur $X^{(2)}$ de $S^{(2)}$, où

$X_i^{(2)\prime} = log((\hat{f}(X_i^{(2)\prime})) - log(\hat{g}(X_i^{(2)\prime}))$.

Étape 02 : un modèle SVM est entrainé sur l'ensemble de données transformée $Z = (X_i^{(2)\prime}, Y^{(2)})$, et le modèle obtenu est utilisé pour classifier les instances de données de l'ensemble test.

3- Classifieur Bayésien Naïf

Le classifieur Bayésien naïf [51, 52] est un modèle probabiliste basé sur l'application du théorème de bayes avec l'hypothèse naïve, i.e. les caractéristiques qui décrivent les instances sont supposées indépendantes les unes des autres lors de la classification. Formellement, en considérant X= ($x_1, x_2, \ldots, x_n$) comme l'ensemble des caractéristiques, et C_i la classe à prédire, le théorème de bayes s'exprime par l'équation suivante :

$$P(C_i|x_1, x_2, \ldots, x_n) = \frac{P(x_1, x_2, .., x_n|C_i) * P(C_i)}{P(x_1, x_2, \ldots, x_n)} \qquad \text{(II-6)}$$

Avec $P(C_i|x_1, x_2, \ldots, x_n)$ c'est la probabilité à posteriori de la classe C_i sachant les valeurs des caractéristiques, $<x_1, x_2, \ldots, x_n>$.

Le dénominateur $P(x_1, x_2, \dots, x_n)$ est souvent négligé car il est une constante pour toutes les classes, ainsi l'équation de bayes peut-être simplifiée comme suit :

$$P(C_i|x_1, x_2, \dots, x_n) = P(x_1, x_2, .., x_n|C_i) * P(C_i) \quad \text{(II-7)}$$

En appliquant la loi des probabilités conditionnelles :

$$P(A, B|C) = P(A|C) * P(B|C, A) \quad \text{(II-8)}$$

On aura : $P(x_1, x_2, .., x_n|C_i) = P(x_1|C_i) * P(x_2, x_3, .., x_n|C_i, x_1)$

$$= P(x_1|C_i) * P(x_2|C_i, x_1) * P(x_3, x_4, .., x_n|C_i, x_1, x_2)$$

$$= P(x_1|C_i) * P(x_2|C_i, x_1) * \dots * P(x_n|C_i, x_1, x_2 \dots x_{n-1})$$

Pour simplifier les calculs, en appliquant l'hypothèse naïve :

$$P(x_j|C_i, x_1 \dots x_{j-1}) = P(x_j|C_i) \quad \text{(II-9)}$$

Donc : $P(x_1, x_2, .., x_n|C_i) = P(x_1|C_i) * P(x_2|C_i) * P(x_3|C_i) \dots P(x_n|C_i) = \prod_{j=1}^{n} P(x_j|C_i)$

Ainsi, la probabilité à posteriori devient :

$$P(C_i|x_1, x_2, \dots, x_n) = P(C_i) * \prod_{j=1}^{n} P(x_j|C_i) \quad \text{(II-10)}$$

Le Classifieur Bayésien Naïf utilise cette probabilité postérieure pour classer les nouvelles instances en attribuant la classe qui présente la probabilité maximale. Cette méthode, connu sous le nom de Maximum a posteriori (MAP), est exprimée par l'équation suivante :

$$C_{MAP} = argmax\, P(C_i|x_1, x_2, \dots, x_n) = argmax\, P(C_i) * \prod_{j=1}^{n} P(x_j|C_i) \quad \text{(II-11)}$$

Où C_{MAP}représente la classe à laquelle l'instance appartient.

Avantages

- La facilité et la simplicité de leur implémentation.
- Leur rapidité.
- Le Classifieur Bayésien naïf peut donner des résultats de classification acceptables.

Inconvénients

- Son hypothèse naïve peut limiter sa performance lorsqu'il s'agit d'une grande quantité de données à traiter.

Dans ce contexte, Mukherjee et Sharma [53] ont proposé une approche de détection d'intrusion basée sur un classifieur Bayesien naïf avec réduction de dimensionnalités. Cette méthode se fait en deux étapes :

Etape1 : Réduction de dimensionnalité

Cette étape consiste à réduire la dimensionnalité en utilisant les trois méthodes suivantes :

1- Gain d'information

Permettant d'évaluer les attributs en mesurant le gain en information par rapport à la classe. Plus formellement, soit C l'ensemble d'entrainement de c données, C_i est un ensemble d'entrainement contenant un échantillon de c_i instances de la classe I. Alors on définit la métrique Expected Information EI comme suit :

$$EI\ (C_i, C_1, \ldots, C_m\ )\ =\ -\sum_{i=1}^{m} \frac{C_i}{C}\ log\left(\frac{C_i}{C}\right) \qquad \text{(II-12)}$$

Tel que $\frac{C_i}{C}$ est la probabilité qu'un échantillon quelconque appartient à une classe C_i.

Ainsi, pour une caractéristiques F ayant v valeurs possibles $\{F_1, F_2, \ldots, F_V\ \}$, on divise l'ensemble d'apprentissage en v sous-ensembles $C_1, C_2, \ldots, C_V$ tels que $C_i,\ i \in \{1, \ldots, v\}$ est les sous-ensembles des instances de l'ensemble d'entrainement vérifiant : $F = f_i$.

Soit c_{ij} le nombre d'instances de la classe i dans le sous-ensemble C_j, alors l'entropie de la caractéristique F est donnée par la formule suivante :

$$E(F) = \sum_{j=1}^{v} \frac{c_{ij} + \cdots c\ +\ c_{mj}}{c} \times\ EI\left(c_{ij}, c1, \ldots\ , c_{mj}\right) \qquad \text{(II-13)}$$

Enfin, le gain d'information est défini par $Gain(F) = IE\ (C_i, C_1, \ldots, C_m) - E(F)$.

2- Taux de gain

Le taux de gain d'une caractéristique F est définie comme suit :

$$GainRatio(F) = \frac{Gain(F)}{SplitInfo\ (C)} \quad \text{(II-14)}$$

Tel que : $SplitInfo_F(C) = \sum_{i=1}^{v} \frac{|Ci|}{|C|} log_2\left(\frac{|Ci|}{C}\right)$, C est l'ensemble d'entrainement : $C = U_{i=1}^{m}\ Ci$, et $|Ci| = c_i$.

3- Sélection de caractéristiques basée sur la corrélation (CFS)

Corrélation du sous-ensemble sommé d'un sous-ensemble de caractéristiques S est donné par la formule :

$$M_s = \frac{K\ \overline{r_{ef}}}{\sqrt{K + K\ (K-1)\overline{r_{ff}}}} \quad \text{(II-15)}$$

Tel que $|S| = K$, $\overline{r_{ef}}$ est la moyenne de la corrélation entre les caractéristiques et la variable de classe ; $\overline{r_{ff}}$ est la Corrélation moyenne intra-sous-ensemble de S.

Etape 2 : Classification

Application de l'algorithme naïf bayésien sur les données à dimensionnalité réduite pour classer les attaques réseau.

Cette approche offre des avantages tels que : meilleure classification (précision), réduction de temps d'apprentissage par réduction de la dimension des données (complexité temporelle de naïf bayésienne dépend de nombre de caractéristiques).

4- L'arbre de décision

L'arbre de décision ou méthode de partitionnement récursif, est un ensemble de règles de classification qui prennent des décisions en se basant sur des tests associés aux attributs organisés sous forme d'une arborescence. Dans cette structure, les nœuds représentent les caractéristiques, les branches désignent les différentes valeurs possibles pour ces caractéristiques et les feuilles correspondent aux classes existantes.

Pour classer une nouvelle instance, on examine les caractéristiques testées à chaque nœud de l'arbre et on suit la branche correspondante aux valeurs observées, jusqu'à atteindre la feuille qui indique la classe de cette instance **[**54].

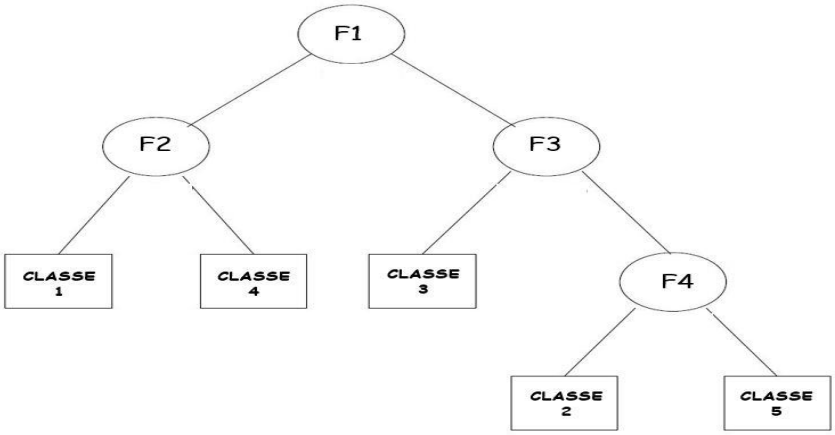

Figure II-11 : Exemple illustratif d'un arbre de décision

L'algorithme CART (*Classification and Regression Trees*) introduit par Breiman et al. (1984) est largement utilisé pour générer ces arbres de décision. Il se base sur l'utilisation de l'indice de Gini comme critère de division.

CART procède à une division récursive des nœuds en sélectionnant la variable qui maximise le gain d'information. À chaque division, un seuil est défini pour séparer de manière optimale les individus en deux groupes homogènes. La construction de l'arbre se termine lorsque le nœud est suffisamment homogène ou lorsque le nombre minimal d'observations par nœud est atteint. Les feuilles de l'arbre final sont assignées à la classe majoritaire ou à la valeur moyenne des observations qu'elles contiennent [55].

L'équation de l'impureté de Gini est représentée par la formule suivante [55] :

$$Gini = 1 - \sum_{i=1}^{n} p_i^2 \qquad \text{(II-16)}$$

Avec : p_i est la proportion de la classe *i* parmi toutes les classes de l'ensemble de données et *n* est le nombre total de classes.

Avantages

- Les arbres de décision sont faciles à utiliser et offrent des performances robustes [56].

- Leur structure arborescente permet de prendre des décisions de manière efficace et interprétable.
- Ils sont largement utilisés dans de nombreux domaines [56].
- Robustesse aux valeurs aberrantes dans les données (leur algorithme de segmentation isole ces observations atypiques dans des nœuds séparés), évitant ainsi qu'elles n'affectent négativement les résultats du modèle.

Inconvénients

- La complexité du calcul des explications abductives ou raisons suffisantes pour une instance peut être exponentielle, rendant leur obtention difficile en pratique [57].
- Bien que les arbres de décision soient généralement robustes aux valeurs aberrantes, leur stabilité peut être limitée, les rendant sensibles aux variations dans les données d'apprentissage.

II.5 Evaluation des algorithmes d'apprentissage automatique

L'évaluation des performances d'un algorithme d'apprentissage automatique nécessite le calcul de diverses métriques, telles que la matrice de confusion, la précision, la sensibilité, la spécificité, le taux de rappel (Recall) et FScore.

II.5.1 La matrice de confusion

Est généralement représentée sous forme de tableau, chaque ligne correspond à une classe prédite et chaque colonne à une classe réelle et inversement. Les éléments de la diagonale principale représentent les instances correctement classées, tandis que les autres éléments représentent les erreurs de classification.

		Les classes réelles	
		Positif (attaque)	**Négatif (normale)**
Les classes prédites	**Positif (attaque)**	Vrai positif (TP)	Faux positif (FP)
	Négatif (normale)	Faux négatif (FN)	Vrai négatif (TN)

Tableau II-1 : Matrice de confusion

- **TP (VP) :** Vrais Positifs (True Positives), représente les cas où un trafic normal est correctement classé comme normal.
- **FP :** Faux Positifs (False Positives), représente les cas où un trafic normal est incorrectement classé comme une attaque.
- **FN :** Faux Négatifs (False Negatives), représente les cas où une attaque est incorrectement considérée comme un trafic normal.
- **TN (VN) :** Vrais Négatifs (True Negatives), désigne les cas où les attaques sont correctement détectées comme telles.

II.5.2 Accuracy

Est une métrique qui mesure la proportion de prévisions correctes par rapport au total de prévisions faites par l'algorithme. Comme est exprimé dans l'équation ci-dessous :

$$Accuracy = \frac{TP + TN}{TP + TN + FP + FN} \quad \text{(II-17)}$$

II.5.3 La précision

Est une mesure de la capacité d'un algorithme d'apprentissage automatique à classifier correctement les instances. Son équation de calcul est la suivante :

$$Précision = \frac{TP}{TP + FP} \quad \text{(II-18)}$$

II.5.4 Le taux de rappel (Recall)

Le taux de rappel mesure la proportion d'instances positives réellement identifiées comme telles par le classifieur par rapport à l'ensemble des instances positives réelles.

$$Taux\ de\ rappel = \frac{TP}{TP + FN} \quad \text{(II-19)}$$

II.5.5 Le F-score

Également appelé F1-score, est une mesure de la précision d'un modèle de classification qui tient compte à la fois de la précision et du rappel. Il est calculé en prenant en considération l'harmonique moyenne de la précision et du rappel.

$$F_1 Score = \frac{2 \times Precision \times Taux\ de\ rappel}{Precision + Taux\ de\ rappel} \quad \text{(II-20)}$$

II.5.6 La spécificité

Est la capacité d'identifier correctement les vrais négatifs, c'est-à-dire les instances négatives qui sont correctement classées comme telles par le classifieur.

$$Spécificité = \frac{TN}{TN + FP} \quad \text{(II-21)}$$

II.5.7 False positif rate

Le false positive rate (FPR) est la proportion de cas négatifs qui sont incorrectement identifiés comme positifs par le modèle de classification. Autrement dit, c'est la probabilité qu'un résultat positif soit en réalité un faux positif. Le FPR est calculé comme suit :

$$FPR = \frac{FP}{FP + TN} \quad \text{(II-22)}$$

II.6 Comparaison entre les algorithmes de l'apprentissage automatique

Les tableaux II-2 et II-3 montrent la comparaison entre les différents algorithmes d'apprentissage supervisés et non supervisés.

Algorithmes	Travaux	Attaques	Métriques	Avantages
SVM	Framework LMDRT-SVM [50]	Classification binaire (attaques/comportement normal)	Taux de rappel, FPR Accuracy	Meilleure classification. Réduction de temps d'apprentissage.
KNN	TCM-KNN [48]	Classification Binaire	TP, FP	Meilleure classification.
Classifieur bayésien naïf	[53] (2012)	Probe, U2R, R2L, DoS, normal	RMSE, Taux de rappel, Accuracy	Meilleure classification. Réduction de temps d'apprentissage par réduction de nombre de caractéristique.

Tableau II-2 : Les algorithmes supervisés

Avec : $RMSE$ (*Root Mean Square Error*) $= \frac{1}{n}\sqrt{\sum_{i=1}^{n}(Y_i - \widehat{y_l})^2}$

Algorithme	Travail	Attaques	Métriques	Avantages
K-moyennes	Modèle pipeline « STG2P » [37]	Détection des comportements anormaux en se basant sur : les activités d'authentification, création et suppression des processus par à partir des PC Systèmes Windows, DNS Lookup, flux des paquets collectés au niveaux des routeurs, comportements du Red Team.	Taux de rappel, FPR	Assurance d'une meilleure performance du classifieur tout en évitant l'étape d'apprentissage. Bonnes performances, dans le cas des données négatives clairsemées (sparce).
	Méthode « QALO-K » [38]	DOS, Prob, R2L, U2L.	**A**ccuracy Taux de rappel, FPR F_1 Score.	Assurance de la convergence vers un optimal global.
Fuzzy C-means	[42] (2012)	DoS, R2L, U2L,Prob.	FPR**,** Accuracy. Précision. Taux de rappel	Détection de la solution optimale globale.
DBSCAN	[45] (2022)	Prob, DoS, R2L, U2R. Exploit, Fuzzers. Shell code.	FPR. Accuracy. Taux de rappel Précision. F1Score.	Bonne qualité de partitionnement. Détection des classes minoritaires. Réduction de la complexité temporelle.
	Système de détection d'intrusion UNIDS. [46]	DoS (ICMP/SYN) Port scan Network scan Propagation des vers informatiques	Taux de rappel, Accuracy	Meilleure qualité de clustering. Amélioration des performances de DBSCAN. Détection des attaques de différentes caractéristiques. Convergence rapide.

Tableau II-3 : Les algorithmes non supervisés

II.7 Conclusion

En conclusion, les systèmes de détection d'intrusion basés sur l'apprentissage automatique représentent une approche prometteuse pour la détection efficace des activités malveillantes sur les réseaux.

Ce chapitre a présenté divers algorithmes d'apprentissage automatique couramment utilisés, en mettant en lumière leurs avantages et inconvénients, décrivant ainsi certains travaux de recherches existants, pour identifier les limitations et les caractéristiques de chacun de ces algorithmes, qui seront examinées et vérifiées lors de leur implémentation dans le chapitre suivant.

Chapitre III

Implémentation et interprétation des résultats

III Implémentation et Interprétation des Résultats

III.1 Introduction

L'évaluation des performances des algorithmes d'apprentissage automatique revêt une importance capitale pour assurer leur fiabilité et leur efficacité dans des applications concrètes.

Dans le cadre de ce chapitre, nous concrétisons les concepts théoriques des algorithmes en les implémentant à l'aide du logiciel MATLAB. Cette mise en pratique est suivie d'une analyse approfondie de leurs performances, où nous combinons différentes métriques et nous appuyons sur des fondements théoriques solides. Cette approche nous permettra d'approfondir notre compréhension des points forts et des limites de chaque algorithme, ainsi que de leur aptitude à détecter et classifier efficacement divers types d'attaques. Cette démarche renforcera notre connaissance de leur fonctionnement et de leur pertinence dans des contextes réels.

III.2 L'environnement de développement

III.2.1 Matériel

Les spécifications techniques du micro-ordinateur utilisé pour cette étude sont répertoriées dans le tableau suivant :

Composants du matériel	**Spécifications techniques**
Processeur	Intel(R) Core (TM) i5-8350U
Vitesse de processeur	1.70GHz
Mémoire	8.00 Go
Système d'exploitation	Windows 10 / 64 bits

Tableau III-1 : Les spécifications techniques du micro-ordinateur

III.2.2 Logiciel

Pour l'implémentation des algorithmes d'apprentissage supervisé et non supervisé, nous avons choisi d'utiliser le logiciel MATLAB, un environnement de développement intégré conçu pour le calcul numérique et la visualisation des données. Il propose une plateforme complète avec de nombreuses toolboxes dédiées pour la mise en œuvre des différents algorithmes d'apprentissage automatiques, offrant ainsi une grande flexibilité et une puissance de calcul élevées.

Grâce à ces capacités, MATLAB constitue un choix idéal pour l'implémentation et l'évaluation de ces algorithmes de manière efficace et précise.

III.2.3 Ensemble de données

Dans cette étude, l'ensemble de données utilisé est le NSL-KDD (Network Security Laboratory-Knowledge Discovery in Databes), qui est une version améliorée du KDD CUP'99, ensemble de données issues de l'initiative de la DARPA en 1998 afin de permettre l'évaluation de systèmes de détection d'intrusion. Les données sont générées à partir d'une simulation d'un réseau, s'abstrayant ainsi du caractère sensible que peuvent représenter les données réseau d'une entreprise [58]. Cette nouvelle version du jeu de données KDD est utilisé comme un ensemble de données de référence efficace pour aider les chercheurs à comparer différentes méthodes de détection d'intrusions. De plus, le nombre d'enregistrements dans les ensembles d'apprentissage et de test NSL-KDD est raisonnable. Cet avantage permet d'exécuter les expériences sur l'ensemble complet des données sans avoir besoin de sélectionner aléatoirement une petite portion [59].

1- Ensemble de données d'apprentissage

Les données d'apprentissage ou données d'entrainement, sont un ensemble d'exemples utilisés pour entrainer un modèle. Ces exemples sont constitués de paires d'entrées et de sorties, où l'entrée représente les données brutes et la sortie correspond à la valeur souhaitée que le modèle doit prédire.

2- Ensemble de données test

Un ensemble de données test est un jeu de données composé d'exemples d'entrée et de leurs sorties correspondantes, qui n'ont pas été utilisés pendant l'entraînement du modèle. Il est utilisé pour évaluer les performances d'un modèle d'apprentissage automatique supervisé entraîné sur un ensemble de données d'apprentissage distinct.

Le jeu de données NSL-KDD, comprenant les ensembles d'apprentissage et de test, est directement accessible et téléchargeable à partir du lien suivant :
https://www.kaggle.com/datasets/hassan06/nslkdd

Le tableau III-2 fournit une description détaillée de la base de données NSL-KDD utilisée, affichant les 23 catégories d'attaques qu'elle contient, ainsi que le type d'attaque et le nombre d'instances d'apprentissage et de test pour chaque catégorie.

Numéro de la catégorie	Catégories	Type d'attaque	Nombre d'instances	
			Apprentissage	Test
1	back	DoS	956	359
2	buffer_overflow	U2R	30	20
3	ftp_write	R2L	8	3
4	guess_passwd	R2L	**53**	**1231**
5	imap	R2L	11	1
6	ipsweep	Probe	3599	141
7	land	DoS	18	7
8	loadmodule	U2R	9	2
9	multihop	R2L	**7**	**18**
10	neptune	DoS	41214	1579
11	nmap	Probe	1493	73
12	normal	/	67343	2152
13	perl	U2R	3	2
14	phf	R2L	4	2
15	pod	DoS	201	41
16	portsweep	Probe	2931	156
17	rootkit	U2R	10	13
18	satan	Probe	3633	727
19	smurf	DoS	2646	627
20	spy	R2L	2	**0**
21	teardrop	DoS	892	12
22	warezclient	R2L	890	**0**
23	warezmaster	R2L	**20**	**944**

Tableau III-2 : Description de la base de données NSL-KDD (23 classes)

Les 23 catégories d'attaques peuvent être regroupées en 5 classes principales (DOS, U2R, R2L, Probe, Normal) comme le montre le tableau III-3, pour révéler certaines catégories critiques.

Numéro de la classe	Classe	Nombre d'instances	
		Apprentissage	Test
1	DOS	45927	2625
2	U2R	52	37
3	R2L	995	2199
4	Probe	11656	1097
5	Normal	67343	2152

Tableau III-3 : Description de NSL-KDD (5 classes)

La figure III-1 représente le pourcentage de chaque classe par rapport au nombre totale d'instances dans les données apprentissage et test.

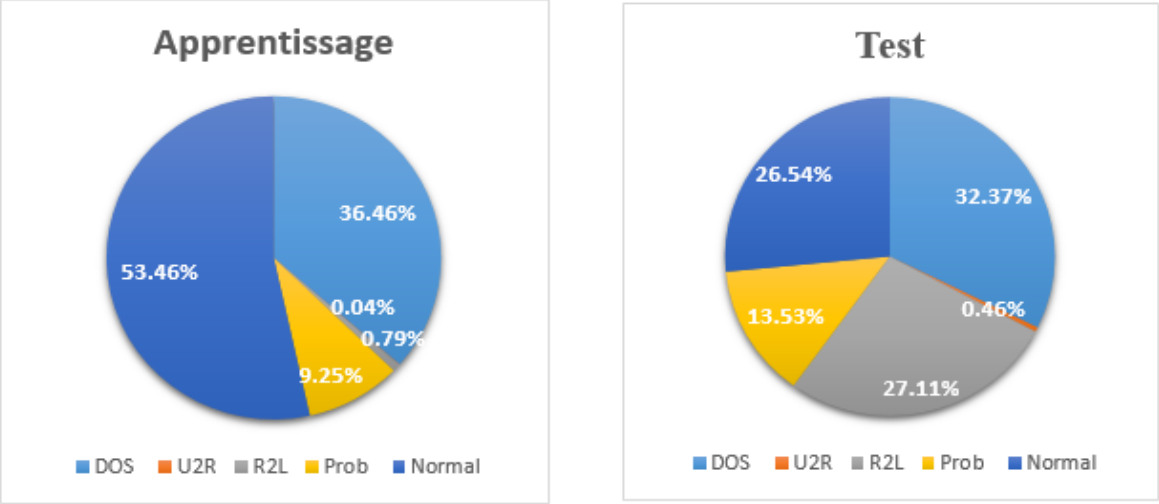

Figure III-1 : Répartition des classes dans la base de données NSL-KDD

D'après cette étude statistique, les classes DOS et Normal sont des classes majoritaires, suivies de la classe Probe. Cependant, les classes U2R et R2L sont considérées comme des classes minoritaires, avec des pourcentages plus faibles par rapport aux classes majoritaires.

III.3 Les algorithmes implémentés

Dans le cadre de cette étude, nous avons choisi de tester certains algorithmes généralement considérés comme les plus utilisés dans la littérature pour évaluer leurs performances.

Les algorithmes d'apprentissages non supervisés implémentés sont :

- K- moyennes.
- DBSCAN.

Les algorithmes d'apprentissage supervisés implémentés sont :

- KNN.
- SVM.
- DT.

III.4 Méthodes d'évaluation des algorithmes

III.4.1 Algorithmes non supervisés

Pour évaluer les performances des algorithmes non supervisés, nous calculons la métrique « précision » qui représente le pourcentage des instances appartenant à la classe représentative de chaque cluster, où la classe représentative est la classe ayant le plus grand nombre d'instances dans un cluster (classe majoritaire).

III.4.2 Algorithmes supervisés

L'évaluation des algorithmes supervisés repose sur le calcul des différentes métriques, et une méthode de validation appelée k-fold.

Les métriques mesurées pour ce type d'algorithmes sont :

$\boldsymbol{Accuracy} = \frac{TP+TN}{TP+TN+FP+FN}$, $\boldsymbol{Précision} = \frac{TP}{TP+FP}$, $\boldsymbol{Taux\ de\ rappel} = \frac{TP}{TP+FN}$,

$\boldsymbol{FScore} = \frac{2\times Precision\times Taux\ de\ rappel}{Precision+Taux\ de\ rappel}$ et $\boldsymbol{spécificité} = \frac{TN}{TN+FP}$.

(Leurs explications est détaillées dans le chapitre précédent (section II-5)).

La méthode de validation k-fold consiste à diviser l'ensemble de données en k sous-ensembles égaux. Chaque itération de cette méthode consiste à utiliser un des k sous-ensembles comme ensemble de validation, tandis que les K-1 autres sous-ensembles servent d'ensemble d'apprentissage. L'algorithme est ensuite entraîné sur l'ensemble d'apprentissage et évalué sur l'ensemble de validation pour mesurer sa performance à l'aide des métriques telles que la précision. En moyennant les performances obtenues sur chaque itération, on obtient une estimation plus fiable de la performance globale de l'algorithme.

III.5 Résultats et interprétations

Dans cette section, nous allons montrer et interpréter les résultats que nous avons obtenus lors de l'implémentation des différents algorithmes supervisé et non supervisé.

III.5.1 K-moyennes

L'implémentation de l'algorithme K- moyennes se fait en deux volets de simulation :

1- Classification binaire qui permet de diviser la base de données en 2 classes :
   - Classe « 1 » : regroupe les attaques (DoS, Probe, U2R, R2L).
   - Classe « 2 » : représente les comportements normaux.
2- Classification en cinq classes :
   - Classe 1 : DOS.
   - Classe 2 : U2R.
   - Classe 3 : R2L.
   - Classe 4 : Probe.
   - Classe 5 : Normal.

Et cela, en variant le nombre de cluster K (K= 2, 3, 5, 8).

- **Résultats de K-moyennes binaire**

K	Partition (indice de cluster)	Classe représentative	Pourcentage (%)	Moyenne (%)
2	**2**	**1 (Attaques)**	**79.409**	**79.409**
	1	2 (Normale)	57.988	57.988
3	**2**	**1 (Attaques)**	**82.717**	**82.717**
	1	2 (Normale)	57.925	71.318
	3	2(Normale)	84.710	
5	**2**	**1 (Attaques)**	**85.365**	**85.365**
	1	2 (Normale)	57.915	77.758
	3	2 (Normale)	98.060	
	4	2 (Normale)	83.598	
	5	2 (Normale)	71.459	
8	**2**	**1 (Attaques)**	**86.874**	**86.874**
	1	2 (Normale)	57.834	80.202
	3	2 (Normale)	78.049	
	4	2 (Normale)	90.591	
	5	2 (Normale)	74.005	
	6	2 (Normale)	64.543	
	7	2 (Normale)	98.071	
	8	2 (Normale)	98.319	

Tableau III-4 : Résultats de K-moyennes binaire

- **Résultats de K-moyennes pour la détection des 5 classes**

K	Partition (indice de cluster)	Classe représentative	Pourcentage (%)	La moyenne (%)
2	**2**	**1 (DOS)**	**45.213**	**45.213**
	1	5 (Normale)	57.988	57.988
3	**2**	**1 (DOS)**	**47.407**	**47.407**
	1	5 (Normale)	57.925	71.318
	3	5 (Normale)	84.710	
5	**2**	**1 (DOS)**	**48.515**	**48.515**
	1	5 (Normale)	57.834	74.080
	3	5 (Normale)	98.146	
	4	5 (Normale)	64.906	
	5	5 (Normale)	75.433	
8	**3**	**1 (DOS)**	**49.719**	**49.719**
	1	5 (Normale)	57.769	78.025
	2	5 (Normale)	93.831	
	4	5 (Normale)	79.578	
	5	5 (Normale)	76.510	
	6	5 (Normale)	98.316	
	7	5 (Normale)	73.944	
	8	5 (Normale)	66.224	

Tableau III-5 : Résultats de K-moyennes 5 classes

- **Interprétation des résultats**

K-moyennes permet un faible coût computationnel : (temps moyen < 4s). En effet, la complexité temporelle de l'algorithme K-moyenne est de $O(nKdl)$, n: nombre d'instances $n = 125\,973$, K: nombre de clusters, d: dimension (nombre de caractéristiques, $d = 42$), l: nombre d'itérations avant convergence, ce qui fait que le temps d'exécution augmente de manière linéaire en fonction de nombre de clusters, et le nombre d'itérations de l'algorithme.

D'après les tableaux III-4 et III-5, le paramètre K a un impact significatif sur les performances de l'algorithme**,** plus K augmente, plus on obtient de meilleures précision (regroupement plus représentatif des différentes instances de données) :

1. K-moyennes binaire a accompli une classification optimale des données en deux classes distinctes, à savoir la classe 1 (Attaques) et la classe 2 (comportement normal) avec un taux de classification optimal respectifs de **85.858%** et **80.202%** pour un nombre de cluster **k =8.**
2. Dans le cas de 5 classes, K-moyennes permet une meilleure détection de la classe 5 (comportement normal) et la classe 1 (DOS) avec un pourcentage optimal respectifs de **78.025%** et **49.719%** pour un nombre de cluster **k =8**.

Ceci peut être expliqué par le fait que l'augmentation de K permet une plus grande granularité et une meilleure détection de la structure mesoscopique des données dont les structures des clusters ne sont pas forcément convexes, i.e. l'augmentation de K permet la détection d'un nombre plus nombreux de clusters, plus menus, de taille relativement similaire, mais avec moins de faux positifs.

L'algorithme des k-moyennes multi-classes n'a pas montré de performances satisfaisantes pour la classification des attaques appartenant aux classes 2, 3 et 4 (U2R, R2L et Probe). En effet, ces classes représentent respectivement **0.04%**, **0.79%** et **9.25%** de la distribution des données, et ne sont pas apparues dans les résultats présentés dans le tableau III-5. Cela signifie que les performances de l'algorithme k-moyennes en termes de précision ont tendance à être affectées par les distributions biaisées de données, i.e. des données (classes/clusters) déséquilibrées. En effet, K-moyennes présente une caractéristique connue sous le nom d'effet d'uniformité qui signifie que l'algorithme a tendance à détecter des partitions de tailles relativement similaires, même si les données sous-jacentes appartiennent à des partitions de tailles significativement différentes (vérité terrain).

III.5.2 DBSCAN

L'évaluation des performances de l'algorithme DBSCAN nécessite la variation des deux hyper-paramètres Eps et MinPts.

- **Eps :** également noté ε, représente la distance minimale qui sépare le point cœur (centroïde) du cluster et les points qui le composent.

MinPts : représente le nombre minimum de points requis pour former une région dense (un cluster).

- **Résultats**

Eps	MinPts	Partition (L'indice de cluster)	Classe représentative	Pourcentage (%)	La moyenne (%)
0.5	700	1	5 (Normale)	53.458	53.458
0.05	2000	**2**	**1 (DOS)**	**48.963**	**48.963**
		1	5 (Normale)	57.902	57.902
0.005	500	**3**	**1 (DOS)**	**52.090**	**52.090**
		1	5 (Normale)	57.303	72.677
		2	5 (Normale)	86.763	
		4	5 (Normale)	73.966	
0.0005	1000	**3**	**1 (DOS)**	**52.090**	**52.090**
		1	5 (Normale)	57.303	72.033
		2	5(Normale)	86.763	
0.00005	200	**3**	**1 (DOS)**	**52.090**	**50.974**
		7	**1 (DOS)**	**49.858**	
		1	5 (Normale)	57.303	71.965
		2	5 (Normale)	86.763	
		4	5 (Normale)	73.966	
		5	5 (Normale)	63.954	
		6	5 (Normale)	77.840	

Tableau III-6 : Résultats de DBSCAN

- **Interprétation des résultats**

L'algorithme DBSCAN est performant en termes de temps d'exécution. En effet, le temps moyen d'exécution est de l'ordre de 1min 30. La complexité temporelle du DBSCAN est $O(n \log(n))$, tel que n est le nombre d'instances.

Les résultats montrent que les paramètres Eps et MinPts ont un impact significatif sur le nombre de clusters et les performances de cet algorithme.

- **Le nombre de cluster :** plus Eps est grand et MinPts petit, plus l'algorithme aura tendance à former moins de clusters mais de densités plus éparses et moins précises.
- **Les performances de DBSCAN :** les meilleures performances sont obtenues pour des valeurs petites de Eps et grandes de MinPts (Eps =0.0005 et MinPts = 1000).

Généralement, l'algorithme DBSCAN est plus efficace pour détecter les clusters de données avec des distributions plus denses, comme les classes majoritaires (la classe DOS et la classe Normale), que pour les classes minoritaires (U2R et R2L) qui ont des distributions plus éparses, autrement

dit, il ne permet pas une détection performante des clusters lorsque les données sont fortement déséquilibrées.

Enfin, il est utile de préciser que l'algorithme DBSCAN s'exécute sous l'hypothèse que les partitions ont des densités similaires. Si les partitions ont des densités très différentes, DBSCAN peut échouer à identifier efficacement toutes les partitions. Ainsi, les partitions plus denses peuvent masquer (absorber) les partitions les plus dispersées durant le processus d'exécution.

III.5.3 KNN

Les hyper-paramètres à varier lors de l'implémentation de l'algorithme KNN sont :

- K (le nombre de voisins) : [3, 5, 8, 13].
- Type de distance : [Euclidienne, Minkowski, Tchebychev].

- **Résultats**
  - ✓ **Résultats d'Accuracy**

Distance	Classe	K			
		3	5	8	13
Euclidienne	DOS	98.076 %	98.089 %	98.298 %	98.187 %
	U2R	99.593 %	99.568 %	99.544 %	99.544 %
	R2L	72.898 %	72.885 %	72.885 %	72.873 %
	Probe	94.032 %	94.106 %	94.229 %	94.279 %
	Normale	65.684 %	65.709 %	65.746 %	65.697 %
Minkowski	DOS	97.941 %	98.064 %	98.15 %	98.2 %
	U2R	99.581 %	99.556 %	99.544 %	99.544 %
	R2L	73.021 %	72.885 %	72.873 %	72.873 %
	Probe	93.872 %	94.020 %	93.983 %	84.18 %
	Normale	65.995 %	65.586 %	65.61 %	65.808 %
Tchebychev	DOS	96.782 %	96.991 %	96.893 %	96.72 %
	U2R	99.556 %	89.544 %	99.544 %	99.544 %
	R2L	73.315 %	72.935 %	72.762 %	72.774 %
	Probe	94.328 %	94.118 %	94.71 %	94.821 %
	Normal	66.363 %	66.375 %	65.892 %	65.906

Tableau III-7 : Résultats d'Accuracy de KNN

✓ **Résultats de Précision**

Distance	Classe	K			
		3	5	8	13
Euclidienne	DOS	97.463 %	97.463 %	97.335 %	97.290 %
	U2R	**100 %**	**100 %**	/	/
	R2L	66.667 %	50 %	50 %	0 %
	Probe	76.445 %	76.983 %	77.709 %	78.234 %
	Normal	32.735 %	42.782 %	42.798 %	42.788 %
Minkowski	DOS	97.415 %	97.389 %	97.468 %	97.219 %
	U2R	80 %	**100 %**	/	/
	R2L	86.667 %	50 %	0 %	0 %
	Probe	75.641 %	76.379 %	76.273 %	77.533 %
	Normal	42.814 %	42.643 %	42.666 %	42.86 %
Tchebychev	DOS	95.956 %	96.126 %	95.758 %	95.061 %
	U2R	60 %	50 %	50 %	50 %
	R2L	79.31 %	62.5 %	0 %	0 %
	Probe	74.747 %	73.957 %	76.341 %	77.189 %
	Normal	43.123 %	43.182 %	42.743 %	42.766 %

Tableau III-8 : Résultats de Précision de KNN

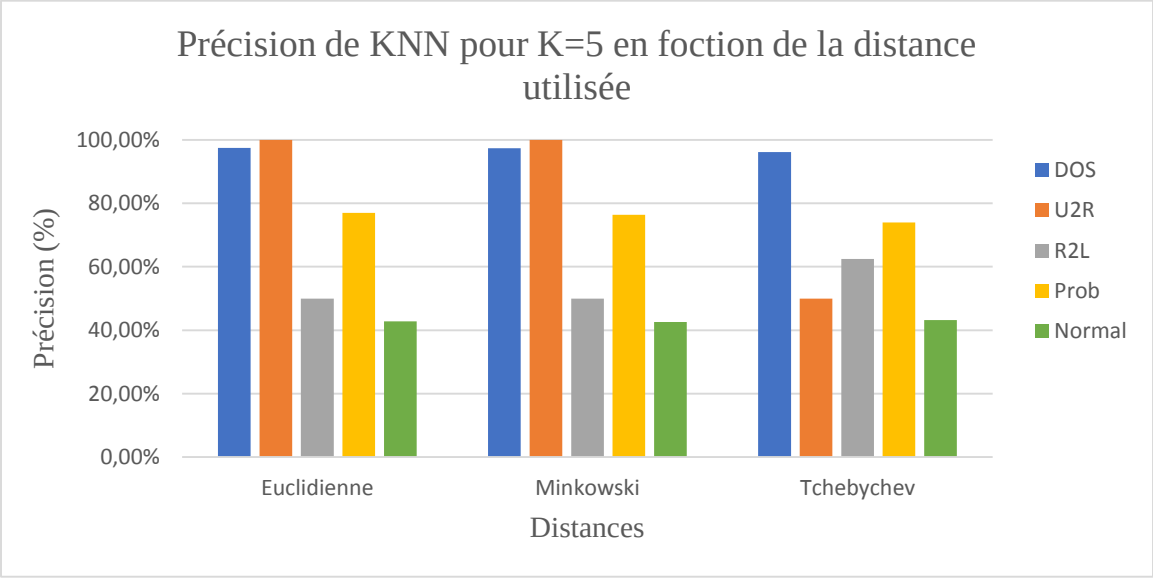

Figure III-2 : Précision de KNN (K=5)

✓ **Résultats de Taux de rappel**

Distance	Classe	K			
		3	5	8	13
Euclidienne	DOS	96.571 %	96.610 %	97.410 %	97.105 %
	U2R	10.811 %	5.405 %	0 %	0 %
	R2L	0.091 %	0.045 %	0.045 %	0 %
	Probe	80.766 %	80.492 %	80.401 %	79.945 %
	Normal	86.245 %	86.617 %	86.431 %	86.849 %
Minkowski	DOS	96.19 %	96.61 %	96.8 %	97.219 %
	U2R	10.811 %	2.703 %	0 %	0 %
	R2L	0.591 %	0.045 %	0 %	0 %
	Probe	80.675 %	80.766 %	80.583 %	80.219 %
	Normal	86.106 %	86.059 %	86.106 %	86.617 %
Tchebychev	DOS	94.019 %	94.514 %	94.59 %	94.781 %
	U2R	8.108 %	2.703 %	2.703 %	2.703 %
	R2L	2.092 %	0.455 %	0 %	0 %
	Probe	87.694 %	87.238 %	88.241 %	87.603 %
	Normal	83.922 %	84.619 %	84.154 %	84.201 %

Tableau III-9 : Résultats de taux de rappel de KNN

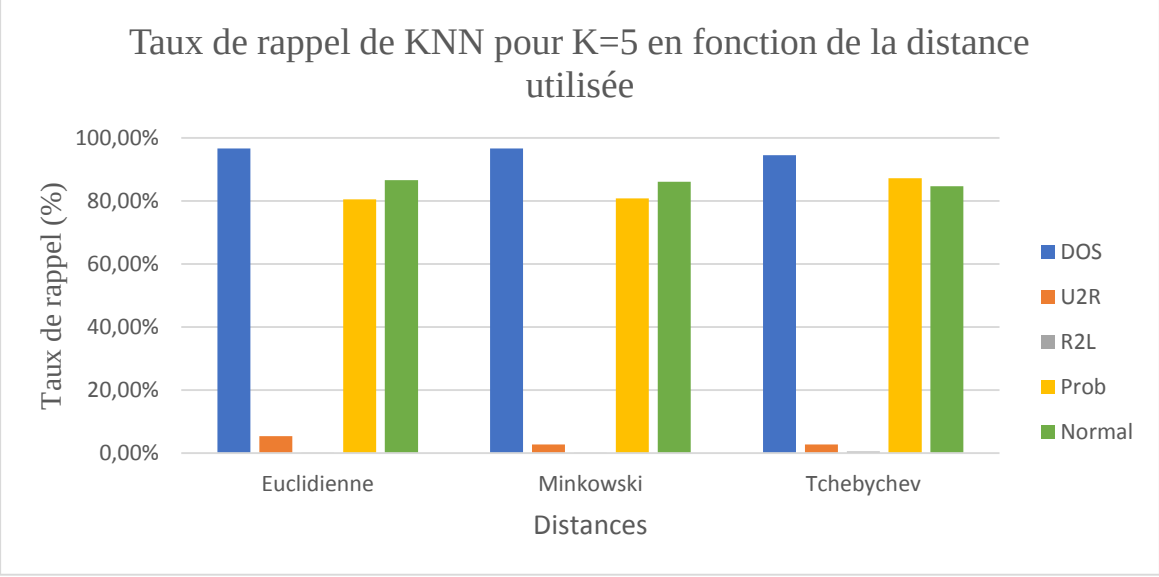

Figure III-3 : Taux de rappel de KNN (K=5)

✓ **Résultats de F-Score**

Distance	Classe	K			
		3	5	8	13
Euclidienne	DOS	97.015 %	97.035 %	97.372 %	97.197 %
	U2R	19.512 %	10.256 %	0 %	0 %
	R2L	0.182 %	0.091 %	0.091 %	0 %
	Probe	78.546 %	78.699 %	79.032 %	79.08 %
	Normal	57.152 %	57.275 %	57.248 %	57.331 %
Minkowski	DOS	96.799 %	96.998 %	97.133 %	97.219 %
	U2R	19.048 %	5.263 %	0 %	0 %
	R2L	1.174 %	0.091 %	0 %	0 %
	Probe	78.077 %	78.511 %	78.369 %	78.853 %
	Normal	57.191 %	57.028 %	57.059 %	57.345 %
Tchebychev	DOS	94.978 %	95.313 %	95.171 %	94.926 %
	U2R	14.286 %	5.128 %	5.128 %	5.128 %
	R2L	4.096 %	0.903 %	0 %	0 %
	Probe	80.705 %	80.05 %	81.86 %	82.067 %
	Normal	56.972 %	57.183 %	56.691 %	56.722 %

Tableau III-10 : Résultats de F- Score de KNN

✓ **Résultats de Spécificité**

Distance	Classe	K			
		3	5	8	13
Euclidienne	DOS	98.797 %	98.797 %	98.724 %	98.706 %
	U2R	100 %	100 %	100 %	100 %
	R2L	99.983 %	99.983 %	99.983 %	99.983 %
	Probe	96.107 %	96.236 %	96.392 %	96.521 %
	Normal	58.258 %	58.157 %	58.275 %	58.056 %
Minkowski	DOS	98.778 %	98.76 %	98.797 %	98.669 %
	U2R	99.988 %	100 %	100 %	100 %
	R2L	99.966 %	99.983 %	99.983 %	99.983 %
	Probe	95.936 %	96.093 %	96.079 %	96.364 %
	Normal	58.459 %	58.191 %	56.207 %	58.291 %
Tchebychev	DOS	98.104 %	98.177 %	97.995 %	97.648 %
	U2R	99.975 %	99.988 %	99.988 %	99.988 %
	R2L	99.797 %	99.898 %	99.831 %	99.848 %
	Probe	95.366 %	95.195 %	95.722 %	95.95 %
	Normal	60.02 %	59.785 %	59.282 %	59.298 %

Tableau III-11 : Résultats de spécificité de KNN

- **Interprétation des Résultats**

L'algorithme KNN est performant en termes de temps d'exécution (**21s**). En effet, la complexité temporelle de KNN est évaluée à $O(nd)$, tels que n est le nombre d'instances, tandis que d est la dimension (nombre de caractéristiques) des instances.

Les résultats montrent que le choix de K et la distances ont un impact significatif sur les performances de l'algorithme KNN :

- Dans les catégories attaques, l'accuracy est très élevée pour les trois classes DOS, U2R, Probe (>90 %) et assez bonnes pour la classe R2L (>72%). Néanmoins, le taux de rappel est assez élevé pour la classe majoritaire DOS (>94%) et la classe Probe (>80%), mais très réduit pour les classes minoritaires, notamment U2R (<11%), et R2L (<2.1%). Ceci peut être expliqué par l'augmentation du nombre de Faux négatifs des classes minoritaires. En effet, le modèle K-NN peut privilégier la prédiction des classes majoritaires (à plus grand nombre d'instances) qui par leurs distributions dans la base de données ont tendance à former la majorité du voisinage des différentes instances de données.

- Les meilleurs résultats sont obtenus pour une distance euclidienne et Minkowski pour la plupart classes (classes majoritaires) et différentes valeurs de K. Ce qui signifie que les deux distances sont bien plus adaptées à l'ensemble de données utilisé (NSL-KDD) i.e. elle prend en considération toutes les caractéristiques des données.

- L'algorithme est très peu performant à la détection des classes minoritaires (outliers) pour des valeurs élevées de K, ce qui nécessite un nettoyage de données.

- **Nettoyage de données**

Le nettoyage des données consiste à supprimer les classes minoritaires, U2R et R2L, afin d'éviter le déséquilibre des classes. Cette approche est motivée par l'espoir d'obtenir de meilleures performances de modèle KNN.

✓ **Résultats d'Accuracy**

Distance	Classe	K			
		3	5	8	13
Euclidienne	DOS	95.625 %	95.182 %	96.255 %	96.459 %
	Probe	90.177 %	90.637 %	91.028 %	91.199 %
	Normal	89.07 %	89.173 %	89.292 %	89.292 %
Minkowski	DOS	95.761 %	96.238 %	96.306 %	96.476 %
	Probe	90.432 %	91.079 %	90.926 %	91.181 %
	Normal	89.019 %	89.156 %	89.309 %	89.326 %
Tchebychev	DOS	93.718 %	93.531 %	93.82 %	93.548 %
	Probe	88.951 %	88.764 %	88.849 %	88.934 %
	Normal	87.347 %	87.572 %	87.504 %	87.487 %

Tableau III-12 : Résultats d'Accuracy de KNN après le nettoyage de données

✓ **Résultats de Précision**

Distance	Classe	K			
		3	5	8	13
Euclidienne	DOS	97.474 %	97.495 %	97.361 %	97.411 %
	Probe	71.667 %	73.316 %	74.611 %	75.394 %
	Normal	84.633 %	84.675 %	84.979 %	84.883 %
Minkowski	DOS	97.52 %	97.397 %	97.364 %	97.263 %
	Probe	72.536 %	75.11 %	74.31 %	75.462 %
	Normal	84.454 %	84.479 %	85.083 %	84.993 %
Tchebychev	DOS	96.844 %	96.868 %	96.968 %	96.29 %
	Probe	67.284 %	66.692 %	67.079 %	67.393 %
	Normal	83.272 %	82.765 %	82.643 %	82.755 %

Tableau III-13 : Résultats de Précision de KNN après le nettoyage de données

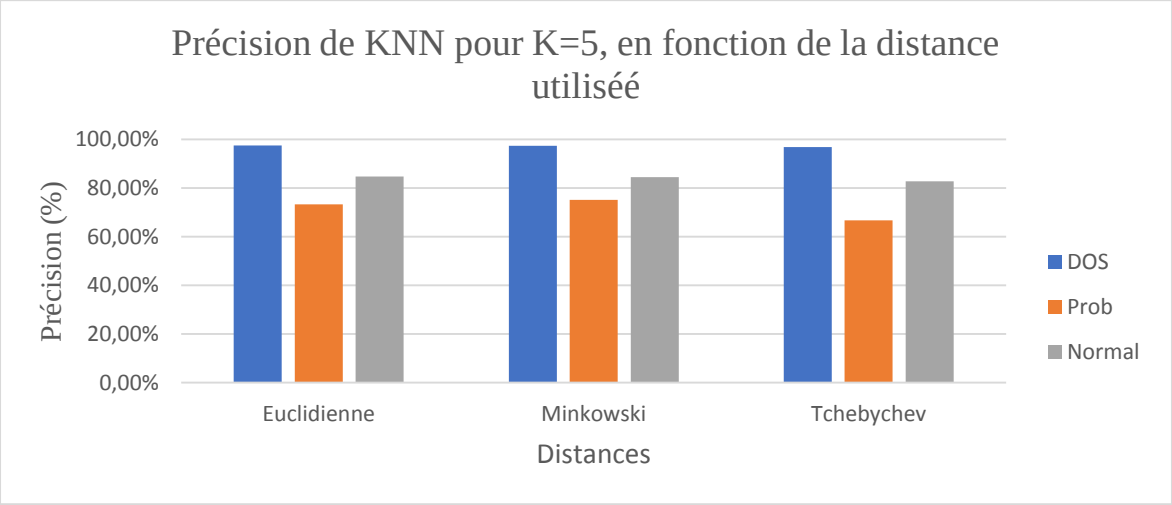

Figure III-4 : Précision de KNN après le nettoyage de données (K=5)

✓ **Résultats de taux de Rappel**

Distance	Classe	K			
		3	5	8	13
Euclidienne	DOS	92.61 %	93.41 %	94.171 %	94.59 %
	Probe	78.396 %	78.396 %	78.76 %	78.487 %
	Normal	85.734 %	86.013 %	85.967 %	86.106 %
Minkowski	DOS	82.876 %	94.095 %	94.286 %	94.781 %
	Probe	78.487 %	78.122 %	78.578 %	78.213 %
	Normal	85.827 %	86.245 %	85.874 %	86.059 %
Tchebychev	DOS	88.838 %	88.381 %	88.952 %	88.99 %
	Probe	79.49 %	79.581 %	79.125 %	78.943 %
	Normal	83.968 %	83.457 %	83.411 %	83.178 %

Tableau III-14 : Résultats de taux de rappel de KNN après le nettoyage des données

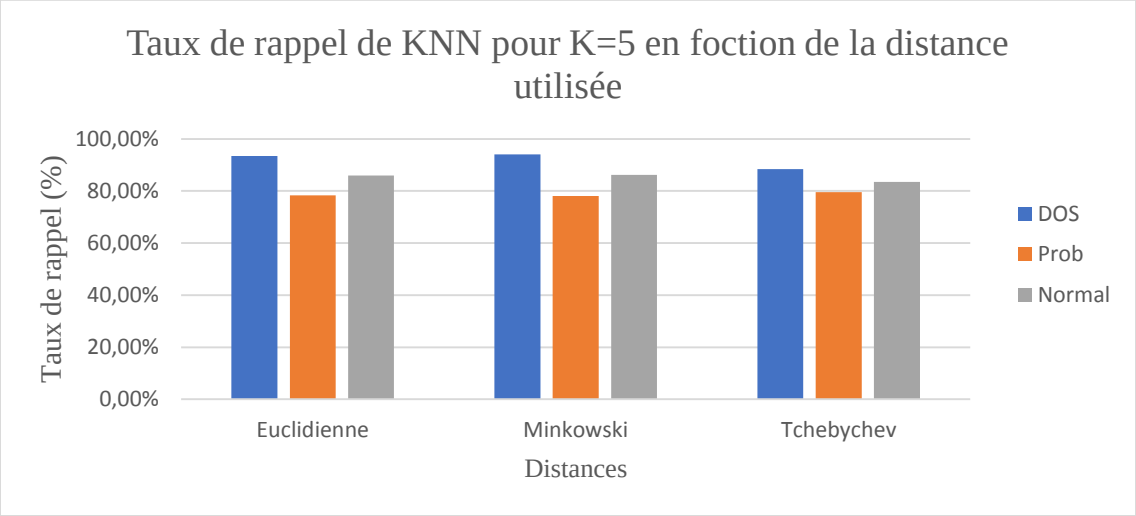

Figure III-5 : Taux de rappel de KNN après le nettoyage de données (K=5)

✓ **Résultats de F-Score**

Distance	Classe	K			
		3	5	8	13
Euclidienne	DOS	94.979 %	95.409 %	95.74 %	95.98 %
	Probe	74.88 %	75.771 %	76.63 %	76.909 %
	Normal	85.18 %	85.339 %	85.47 %	85.49 %
Minkowski	DOS	95.141 %	95.718 %	95.8 %	96.006 %
	Probe	75.394 %	96.586 %	76.385 %	76.813 %
	Normal	85.135 %	85.353 %	85.476 %	85.623 %
Tchebychev	DOS	92.668 %	92.43 %	92.788 %	92.497 %
	Probe	72.879 %	72.569 %	72.606 %	72.712 %
	Normal	83.619 %	83.11 %	83.025 %	82.966 %

Tableau III-15 : Résultats F-Scor de KNN après le nettoyage de données

✓ **Résultats de Spécificité**

Distance	Classe	K			
		3	5	8	13
Euclidienne	DOS	98.061 %	98.061 %	97.938 %	97.969 %
	Probe	92.883 %	93.448 %	93.846 %	94.118 %
	Normal	90.999 %	90.999 %	91.214 %	91.134 %
Minkowski	DOS	98.092 %	97.969 %	97.938 %	97.845 %
	Probe	93.176 %	94.055 %	93.762 %	94.16 %
	Normal	90.835 %	90.838 %	91.295 %	91.214 %
Tchebychev	DOS	97.661 %	97.692 %	97.753 %	97.23 %
	Probe	91.124 %	90.873 %	91.082 %	91.229 %
	Normal	90.247 %	89.952 %	89.871 %	89.979 %

Tableau III-16 : Résultats de spécificité de KNN après le nettoyage de données

- **Interprétation des résultats**

Après ce prétraitement, on distingue que les meilleurs résultats sont obtenus pour les classes à distribution moins déséquilibrée (Probe, DOS, Normale). En effet, après élimination des classes minoritaires, les k plus proches voisins d'une instance de données à classifier sont plus susceptibles d'appartenir aux différentes classes dépendamment des valeurs de leurs caractéristiques. Ainsi, KNN peut détecter de façon plus précise les similarités entre les points de données et identifier les frontières de décision pour une meilleure classification.

Un autre résultat intéressant est que, après nettoyage de donnée, le KNN présente de bien meilleurs résultats globaux avec le F-Score, une métrique combinant la précision et le taux de rappel et permettant de dépeindre une vision plus globale des performances de l'algorithme. Ainsi, après élimination des classes minoritaire, la valeur de F-score de la classe Normale s'est accrue drastiquement passant de valeurs > 57% à des valeurs $> 82\%$. Ce qui permet dans le cas applicatif de réduire le taux de fausses alarmes au niveau de l'IDS. En outre, bien que le F-Score dans le cas des deux autres classes (DOS et Probe) s'est reculé de 3%, la valeur reste très appréciable (>74% pour Probe et > 94% pour DOS).

III.5.4 SVM

Dans cette étude, L'implémentation de l'algorithme SVM se base sur une classification binaire qui permet de diviser la base de données en 2 classes :

- Classe « 0 » : représente le comportement normal.
- Classe « 1 » : regroupe les attaques (DOS, Probe, U2R, R2L).

Et cela, en variant l'hyper-paramètres noyau (Kernel) : Linéaire, Polynomiale, Gaussienne et RBF.

- **Résultats**

Kernel	Classe	L'accuracy	Précision	Taux de Rappel	Fscore	Spécificité
Linéaire	0 (Normale)	78.059 %	5.02 %	1.162 %	1.887 %	**95.123 %**
	1 (Attaques)	78.059 %	81.263 %	**95.123 %**	**87.648 %**	1.162 %
Polynomiale ordre 2	0 (Normale)	**81.949 %**	**93.333 %**	0.651 %	1.292 %	**99.99 %**
	1 (Attaques)	**81.949 %**	**81.935 %**	**99.99 %**	**90.066 %**	0.651 %
Polynomiale ordre 3	0 (Normale)	18.093 %	18.1 %	**99.582 %**	30.632 %	0.01 %
	1 (Attaques)	18.093 %	10 %	0.01 %	0.021 %	**99.582 %**
Polynomiale ordre 4	0 (Normale)	18.118 %	18.2 %	**99.721 %**	30.668 %	0.01 %
	1 (Attaques)	18.118 %	14.286 %	0.01 %	0.021 %	**99.721 %**
Gaussienne	0 (Normale)	49.376 %	25.299 %	**91.543 %**	39.642 %	40.019 %
	1 (Attaques)	49.376 %	**95.521 %**	40.019 %	56.406 %	91.543
RBF	0 (Normale)	49.603%	25.380%	**91.496 %**	39.738%	40.307%
	1 (Attaques)	49.603%	**95.528%**	40.307%	56.693%	**91.496%**

Tableau III-17 : Résultats SVM binaire

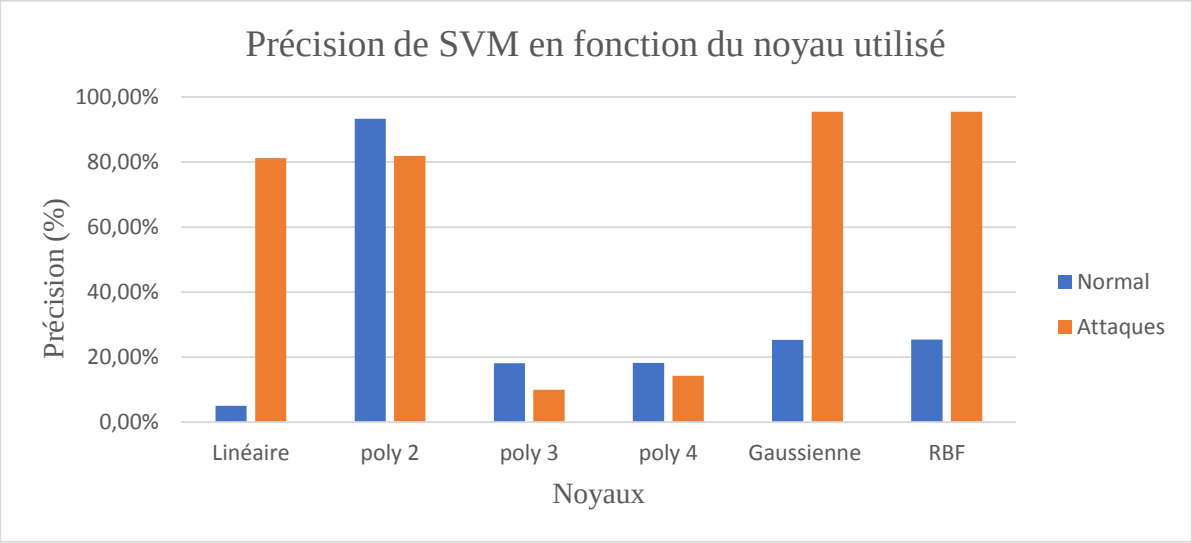

Figure III-6 : Précision de SVM binaire

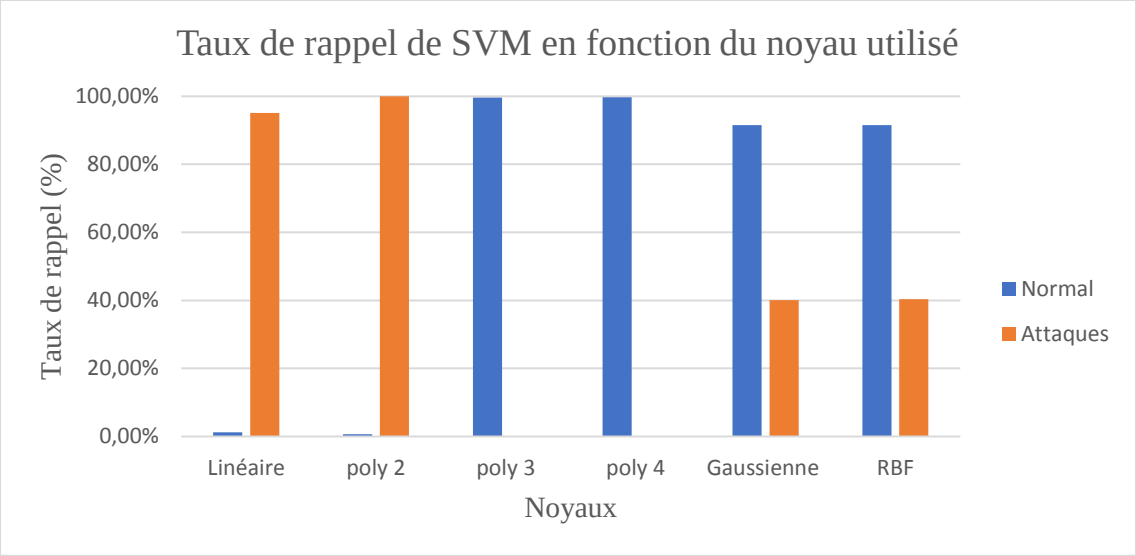

Figure III-7 : Taux de rappel de SVM binaire

- **Interprétation des résultats**

Le temps d'exécution de l'algorithme SVM est influencé par le calcul du noyau. Les noyaux polynomiaux et linéaire sont généralement les plus coûteux en termes de temps d'exécution avec une moyenne tend vers **2 heures**, tandis que les noyaux RBF et Gaussien sont plus rapides (**14min**). En effet, bien que la complexité temporelle de l'algorithme SVM est généralement notée $O(nd)$, tels que n: le nombre d'instances, d: dimension (nombre de caractéristiques), la complexité de l'évaluation du kernel dans le cas polynomial est : $O(n^2 \times d \times deg)$, tel que n est le nombre d'instances, deg:degré du polynôme, d: dimension (nombre de caractéristiques), linéaire et Gaussienne : $O(n^2 d)$, RBF : $O(nd)$.

Le choix de noyau a un impact majeur sur les performances de cet algorithme :

- Dans le cas du noyau linéaire, la précision et le taux de rappel sont faibles pour la classe normale (0). Tandis que les deux métriques sont assez élevées pour la classe attaques (1), cela indique que le modèle fonctionne globalement mal pour l'identification des vrais positifs de la classe normale (0).
- Le noyau polynomial d'ordre 2 donne des meilleurs résultats globaux en termes de précision et d'accuracy pour les deux classes (normale et attaques). Cependant, le taux de rappel et F-score sont faibles pour la classe normale ce qui implique que le modèle a mal à reconnaitre les vrais positifs pour cette classe.

- Les noyaux polynomiaux d'ordre 3 et 4 et le noyau gaussien et RBF ont des faibles performances par rapport aux noyaux linéaire et polynomiale d'ordre 2.

En conclusion, le noyau basé sur la fonction polynomiale d'ordre 2 permet d'avoir une bonne séparation linéaire des données, ce qui signifie que SVM est plus performant pour cette fonction par rapport aux autres. L'obtention des faibles résultats pour les fonctions polynomiales d'ordre 3, 4 est probablement dû au problème de surajustement (surapprentissage) du modèle. i.e. Ce modèle s'ajuste bien aux données d'apprentissage, mais ne peut pas généraliser efficacement à de nouvelles données.

III.5.5 DT

L'hyper-paramètre à varier lors de l'implémentation de l'algorithme DT est le nombre de division ND.

- **Résultats**

Nombre de division (ND)	Classe	L'accuracy	Précision	Taux de Rappel	F-Score	Spécificité
5	DOS	80.284 %	65.922 %	80.914 %	72.653 %	79.982 %
	U2R	**99.544 %**	/	0 %	0 %	**100 %**
	R2L	72.885 %	/	0 %	0 %	**100 %**
	Probe	**92.935 %**	66.838 %	**94.804 %**	78.402 %	**92.642 %**
	Normale	53.169 %	25.3 %	39.173 %	30.744 %	58.228 %
15	DOS	**89.544 %**	**94.806 %**	71.619 %	81.597 %	**98.122 %**
	U2R	**99.544 %**	/	0 %	0 %	**100 %**
	R2L	72.885 %	/	0 %	0 %	**100 %**
	Probe	**92.972 %**	67.816 %	**91.431 %**	77.873 %	**93.213 %**
	Normale	61.677 %	39.716 %	**85.781 %**	45.294 %	52.971 %
Non limité	DOS	**88.742 %**	**96.421 %**	67.733 %	79.57 %	**98.797 %**
	U2R	**99.544 %**	/	0 %	0 %	**100 %**
	R2L	73.058 %	59.722 %	1.955 %	3.787 %	**99.509 %**
	Prob	81.467 %	41.775 %	**93.984 %**	57.84 %	79.509 %
	Normal	53.736 %	28.529 %	49.396 %	36.169 %	55.304 %

Tableau III-18 : Résultats de DT

- **Interprétation des résultats**

L'algorithme DT est performant en termes de temps d'exécution (**10s – 13s**). En effet, la complexité temporelle de DT est $O(n\log(n)\, d)$ tel que n est le nombre d'instances et d la dimension des instances.

D'après les résultats obtenus, on distingue que :

- L'augmentation de paramètre « nombre de divisions » induit aux meilleures performances dont les meilleurs résultats sont obtenus pour un nombre de divisions non limité, ce qui signifie que DT s'adapte bien aux données et peut produire des résultats satisfaisants sans avoir à optimiser ce paramètre. Cette propriété simplifie la mise en œuvre de cet algorithme.
- L'algorithme DT est peu performant pour la détection des classes minoritaires (U2R, R2L), ce qui nécessite un prétraitement qui sert à les éliminer.

- **Nettoyage de données**

Le nettoyage des données consiste à supprimer les classes minoritaires, U2R et R2L, afin d'éviter le déséquilibre des classes. Cette approche est motivée par l'espoir d'obtenir de meilleure performances de modèle DT.

- **Résultats après le prétraitement des données**

Nombre de divisions	Classe	L'accuracy	Précision	Taux de rappel	Fscore	Spécificité
5	DOS	71.978 %	66.29 %	75.886 %	70.764 %	68.821 %
	Probe	**93.292 %**	75.789 %	**94.166 %**	**83.984 %**	**93.092 %**
	Normal	69.979 %	62.55 %	43.773 %	51.504 %	**84.847 %**
15	DOS	84.866 %	**96.818 %**	68.381 %	80.152 %	**98.184 %**
	Probe	**93.207 %**	77.012 %	**90.702 %**	**83.288 %**	**93.783 %**
	Normal	81.546 %	69.575 %	**88.197 %**	77.787 %	77.7 %
Non limité	DOS	**83.623 %**	**95.813 %**	66.248 %	78.333 %	**97.661 %**
	Probe	78.328 %	46.085 %	**94.439 %**	61.943 %	74.628 %
	Normal	69.646 %	60.188 %	50.651 %	55.009 %	80.629 %

Tableau III-19 : Résultats de DT après le prétraitement des données

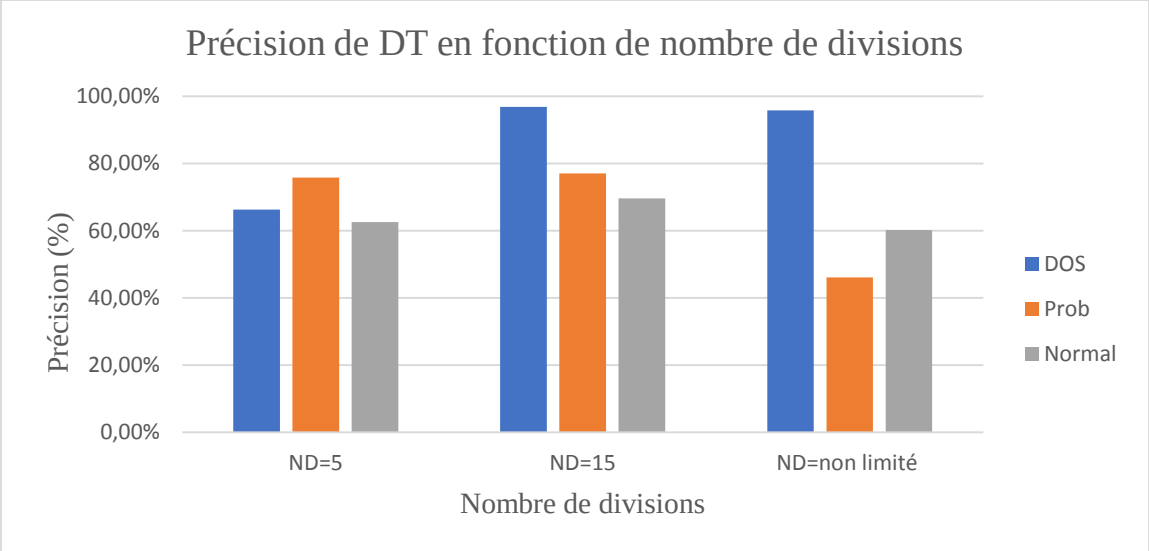

Figure III-8 : Précision de DT après le nettoyage de données

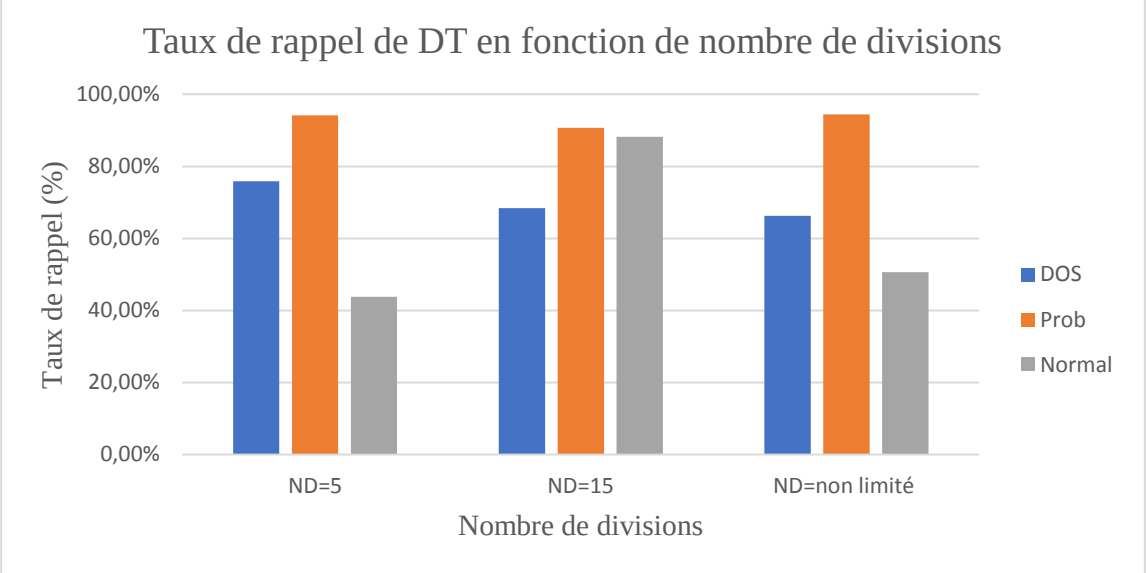

Figure III-9 : Taux de rappel de DT après le nettoyage de données

- **Interprétation des résultats**

Après ce prétraitement, on distingue que les meilleurs résultats sont obtenus pour les classes équilibrées (Probe, DOS, Normal).

III.6 Conclusion

En conclusion, l'implémentation efficace des algorithmes d'apprentissage automatique est essentielle pour évaluer les performances des modèles. Dans ce chapitre, nous avons mis en œuvre une variété d'algorithmes d'apprentissage automatique et les avons évalués en utilisant diverses méthodes, puis interprétés les résultats obtenus pour améliorer la compréhension des performances des modèles.

Nous avons observé que certains algorithmes, tels que K-moyenne, DBSCAN, KNN et DT, offrent des performances élevées en termes de temps d'exécution, mais peuvent être moins efficaces pour la détection des classes minoritaires. D'autres algorithmes, comme SVM, peuvent être coûteux en termes de temps d'exécution mais très efficaces pour la classification.

En résumé, chaque algorithme présente des forces et des faiblesses distinctes, ce qui rend essentiel de prendre en compte les conditions de performances spécifiques de chaque algorithme, telles que la base de données utilisée, la tolérance au déséquilibre des classes et le temps d'exécution, pour choisir le meilleur algorithme pour un problème donné.

Conclusion générale

Conclusion générale

Les systèmes de détection d'intrusions (IDS) basés sur les techniques d'apprentissage automatique constituent un outil essentiel pour lutter contre les cybermenaces sophistiquées d'aujourd'hui. Leur capacité à apprendre et à s'adapter, associée à une analyse comportementale avancée, offre une protection proactive et efficace contre les attaques en constante évolution.

Cette étude est portée sur l'implémentation et l'évaluation de certains algorithmes d'apprentissage automatique, supervisés et non supervisés, tels que K-moyenne, DBSCAN, KNN, SVM et DT, en utilisant une base de données nommée NSL-KDD, afin d'identifier ceux qui offrent la meilleure précision et le temps d'exécution le plus court pour une application pratique.

D'après les résultats obtenus, on a distingué que :

- K-moyenne est rapide et efficace pour la détection et la classification des attaques, mais ces performances peuvent être affectées par les distributions biaisées des données.
- DBSCAN est performant en termes de temps d'exécution et de complexité temporelle, mais il peut échouer à détecter les clusters dans les données sont fortement déséquilibrées.
- KNN et DT sont rapides et efficaces pour l'identification des classes majoritaires.
- SVM est robuste et performant pour les noyaux polynomiaux d'ordre 2, mais peut être affecté par le sur apprentissage.

Par conséquent, le choix des algorithmes de clustering et de classification utilisée dans les IDS doit se faire en fonction des spécificités des données et de l'application visée. Il n'existe pas de solution universelle, et la question clé est de déterminer dans quelles conditions une méthode particulière peut significativement surpasser les autres sur un problème donné. Pour offrir une protection proactive et efficace contre les attaques en constante évolution, ces algorithmes doivent être combinés de manière judicieuse afin de tirer parti des avantages de chacun.

Bibliographie

[1] Arnould, G. (2006). Etude et conception d'architectures haut-débit pour la modulation et la démodulation numériques. Manuscript of thesis, LICM, University of Metz.

[2] RFC Index. Juin 2024. URL : https://www.rfc-editor.org/rfc-index-100d.html.

[3] Site web Kaspersky. Qu'est-ce qu'une attaque XSS (Cross-Site Scripting) ? Définition et explication.URL :https://www.kaspersky.fr/resource-center/definitions/what-is-a-cross-site-scripting-attack.

[4] Mickael franc.OWASP/ CROSS-SITE SCRIPTING (XSS). 2024. URL: https://www.clever-age.com/owasp-cross-site-scripting-xss/.

[5] Merouane, M. (2010). Reseaux bayesiens pour la detection d'intrusions dans un reseau informatique (Doctoral dissertation, Blida).

[6] Kaspersky. L'injection SQL et comment l'éviter. URL: https://www.kaspersky.fr/resource-center/definitions/sql-injection.

[7] Kuperman, B. A., Brodley, C. E., Ozdoganoglu, H., Vijaykumar, T. N., & Jalote, A. (2005). Detection and prevention of stack buffer overflow attacks. Communications of the ACM, 48(11), 50-56.

[8] Alex Legeay. Top 9 des e-mails de phishing les plus courants. URL: https://blog.usecure.io/fr/les-exemples-les-plus-commun-demails-de-phishing.

[9] Poinsot, L. Introduction à la sécurité informatique (Support de cours, Université Paris 13 Institut Galilée).

[10] Terkouia, D. Hakima, D. (2014). Mise en place d'une solution de sécurité d'un réseau informatique cas d'une banque (Doctoral dissertation, Université Mouloud Mammeri).

[11] Ionos. Qu'est-ce que le port scanning. URL: https://www.ionos.fr/digitalguide/serveur/know-how/balayage-de-ports-principes-et-situation-juridique/.

[12] Sensors. The Design of Large-Scale IP Address and Port Scanning Tool. URL : https://www.mdpi.com/1424-8220/20/16/4423.

[13] It-connecte. Techniques de scan de port UDP. URL : https://www.it-connect.fr/technique-de-scan-de-port-udp/.

[14] Formip. Les attaques d'usurpation d'identité. URL: https://www.formip.com/pages/blog/les-attaques-dusurpation-didentite.

[15] Dridi, L. (2017). Mitigation des attaques de déni de service dans les réseaux définis par logiciel (Doctoral dissertation, École de technologie supérieure).

[16] UBIKA. Attaque DDoS : comment protéger votre organisation? URL: https://www.ubikasec.com/articles/attaque-ddos-le-pire-cauchemar-dune-organisation/.

[17] Erickson, J. (2008). Hacking: the art of exploitation.

[18] Hashedout. Everything You Need to Know About ARP Spoofing. URL: https://www.thesslstore.com/blog/everything-you-need-to-know-about-arp-spoofing/.

[19] Berkani, D. Bouzeria, M. (2022) Etude et mise en place d'une infrastructure réseau sécurisée (Mémoire de master, Université Abderrahmane Mira-Bejaia).

[20] AVG Antivirus. Qu'est-ce qu'un serveur proxy ?
URL : https://www.avg.com/fr/signal/proxy-server-definition.

[21] ANSSI. (2008). Recommandations relatives à l'interconnexion d'un système d'information à internet.

[22] Bloch, L., Wolfhugel, C., Queinnec, C., Schauer, H., & Makarévitch, N. (2013). Sécurité informatique: Principes et méthodes à l'usage des DSI, RSSI et administrateurs. Editions Eyrolles.

[23] Bouaichi, K. Bennai, A. (2022). Détection d'intrusions et classification des attaques réseaux par les réseaux de neurones (Mémoire de master, Université Abderrahmane Mira-Bejaia).

[24] Yende, R. G. (2018). Sécurité informatique et crypto. (Support de cours).

[25] Nguyen, T. Q. (2023). Apprentissage automatique non supervisé pour la détection de trafics illégitimes (Doctoral dissertation, Université Paul Sabatier-Toulouse III).

[26] Dagorn, N. (2006). Détection et prévention d'intrusion : présentation et limites.

[27] Baudoin, N., & Karle, M. (2004). NT Réseaux : IPS et IDS. Université de Marne la Vallée, France.

[28] Aleksandar Lazarevic, Vipin Kumar, Jaideep SrivastavaComputer Science Department, University of Minnesota. INTRUSION DETECTION: A SURVEY. Janvier 2005.ResearchGate. file:///C:/Users/Administrateur/Desktop/Downloads/Intrusion_Detection_A_Survey.pdf.

[29] Cisco Systems. (2010). Cisco ASA 8.2 Configuration Guide.

[30] Riquet, D. (2015). Discus: Une architecture de détection d'intrusions réseau distribuée basée sur un langage dédié (Doctoral dissertation, Université Lille 1-Sciences et Technologies).

[31] Jabou, Ch. Schillings, M. Hantach, A. (2009). TER Détection D'anomalies sur le réseau (Université Paris Descartes).

[32] Majorczyk, F. (2008). Détection d'intrusions comportementale par diversification de COTS : application au cas des serveurs web (Doctoral dissertation, Université Rennes 1).

[33] Lerman, L., Markowitch, O., & Bontempi, G. (2011). Les systemes de detection d'intrusion bases sur du machine learning. M. Coulibaly et al. M. Coulibaly et al.

[34] Rosay, A. (2022). Détection d'intrusions dans les objets connectés par des techniques d'apprentissage automatique: étude dans les domaines de l'éducation et des voitures connectées (Doctoral dissertation, Le Mans Université).

[35] Fataicha, Y. (2005). Recherche d'information dans les images de documents (Doctoral dissertation, École de technologie supérieure).

[36] Ngon, J. M., & Segers, J. Exploration des méthodes Sparse K-Means et leurs utilisations dans le contexte des données textuelles.

[37] Zhang, Z., Wang, L., Chen, G., Gu, Z., Tian, Z., Du, X., & Guizani, M. (2022). STG2P: A two-stage pipeline model for intrusion detection based on improved LightGBM and K-means. Simulation Modelling Practice and Theory, 120, 102614.

[38] Chen, J., Qi, X., Chen, L., Chen, F., & Cheng, G. (2020). Quantum-inspired ant lion optimized hybrid k-means for cluster analysis and intrusion detection. Knowledge-Based Systems, 203, 106167.

[39] Fard, M. M. (2020). Learning data representations in unsupervised learning (Doctoral dissertation, Université Grenoble Alpes 2020).

[40] Settouti, N. (2011). Renforcement de l'Apprentissage Structurel pour la reconnaissance du Diabète.

[41] Grover, N. (2014). A study of various fuzzy clustering algorithms. International Journal of Engineering Research, 3(3), 177-181.

[42] Ganapathy, S., Kulothungan, K., Yogesh, P., & Kannan, A. (2012). A novel weighted fuzzy C–means clustering based on immune genetic algorithm for intrusion detection. Procedia Engineering, 38, 1750-1757.

[43] Labroche, N. (2012). Méthodes d'apprentissage automatique pour l'analyse des interactions utilisateurs (Doctoral dissertation, Université Pierre et Marie Curie, Paris 6).

[44] Ester, M., Kriegel, H. P., Sander, J., & Xu, X. (1996, August). A density-based algorithm for discovering clusters in large spatial databases with noise. In kdd (Vol. 96, No. 34, pp. 226-231).

[45] Alfoudi, A. S., Aziz, M. R., Alyasseri, Z. A. A., Alsaeedi, A. H., Nuiaa, R. R., Mohammed, M. A. & Jaber, M. M. (2022). Hyper clustering model for dynamic network intrusion detection. IET Communications.

[46] Casas, P., Mazel, J., & Owezarski, P. (2012). Unsupervised network intrusion detection systems: Detecting the unknown without knowledge. Computer Communications, 35(7), 772-783.

[47] Sovilj, M., Popov, D., Subotić, M., Brekman, G., Babić, P. B., Garyev, G. & Tivadar, H. (2017). SPEECH AND LANGUAGE 2017.

[48] Li, Y., & Guo, L. (2007). An active learning based TCM-KNN algorithm for supervised network intrusion detection. Computers & security, 26(7-8), 459-467.

[49] Boser, B. E., Guyon, I. M., & Vapnik, V. N. (1992, July). A training algorithm for optimal margin classifiers. In Proceedings of the fifth annual workshop on Computational learning theory (pp. 144-152).

[50] Wang, H., Gu, J. Wang, S. (2017). An effective intrusion detection framework based on SVM with feature augmentation. Knowledge-Based Systems, 136, 130-139.

[51] Fareh, M. (2016). Classifieur Naïf de Bayes. (Support de cours,Université de Blida).

[52] Merouane, M. (2010). Reseaux bayesiens pour la detection d'intrusions dans un reseau informatique (Doctoral dissertation, Blida).

[53] Mukherjee, S., & Sharma, N. (2012). Intrusion detection using naive Bayes classifier with feature reduction. Procedia Technology, 4, 119-128.

[54] Chikhi, S. Benhammada, S. Etude comparative de méthodes de séléction de caractéristiques en apprentissage automatique. Oroposition d'une variante.

[55] Abedinia, A. Seydi, V. (2024). Building semi-supervised decision trees with semi-cart algorithm. International Journal of Machine Learning and Cybernetics, 1-18.

[56] Poterie, A. (2018). Arbres de décision et forêts aléatoires pour variables groupées (Doctoral dissertation, INSA de Rennes).

[57] Audemard, G., Bellart, S., Bounia, L., Koriche, F., Lagniez, J. M., & Marquis, P. (2022). Sur le pouvoir explicatif des arbres de décision. Extraction et Gestion des Connaissances: EGC'2022, 38.

[58] Tavallaee, M., Bagheri, E., Lu, W., & Ghorbani, A. A. (2009, July). A detailed analysis of the KDD CUP 99 data set. In 2009 IEEE symposium on computational intelligence for security and defense applications (pp. 1-6). Ieee.

[59] Dhanabal, L., & Shantharajah, S. P. (2015). A study on NSL-KDD dataset for intrusion detection system based on classification algorithms. International journal of advanced research in computer and communication engineering, 4(6), 446-452.