RÉPUBLIQUE DU SÉNÉGAL

Un Peuple – Un But – Une Foi



MINISTÈRE DE L'ENSEIGNEMENT SUPÉRIEUR, DE LA RECHERCHE ET DE L'INNOVATION



UNIVERSITE CHEIKH ANTA DIOP DE DAKAR

****

FACULTE DES SCIENCES ET TECHNIQUES

****

DEPARTEMENT DE MATHEMATIQUES ET INFORMATIQUE

SECTION INFORMATIQUE

MÉMOIRE DE MASTER INFORMATIQUE

Spécialité : Réseaux et Télécommunications

Option : Réseaux, Systèmes et Services

Sujet : Solution intelligente de détection précoce d'attaques DDoS basée sur le Machine Learning

Présenté et soutenu publiquement par :

Melle Maty MBENGUE

Encadré par Professeur Cherif DIALLO

Le : 13/05/2025

Jury

Pr. Ibrahima NIANG	Professeur Titulaire, UCAD	Président
Dr. Mohamed Ould DEYE	Maître Assistant Titulaire, UCAD	Membre
Dr. Ibrahima DIANE	Maître Assistant Titulaire, UCAD	Membre
Pr. Ousmane SADIO	Assistant Titulaire, UCAD	Membre
Pr. Chérif Diallo	Professeur Titulaire, UGB	Encadreur

Année universitaire 2023 – 2024

Dédicace

*À ma très chère mère : **Aissata Oury Mbaye Thiongane***

La plus merveilleuse des mamans, qui m'a tout donné sans compter, je te dédie ce mémoire, fruit de tes conseils et encouragements. Ton soutien et ton amour inconditionnels ont fait de moi ce que je suis aujourd'hui. J'espère que ce travail témoigne de ma profonde gratitude et affection. Puisse Dieu te protéger et te combler de bonheur, pour que je puisse te rendre un peu de ce que je te dois maman.

***À** mon **père Birane** qu'Allah le pardonne et l'accueil dans son paradis céleste.*

*Particulièrement à **ma famille** en commençant par : Maty Khady Sy, Faty Kara, Elhadji, Mame Néné, Moussa, Kairou, Amath, Modou, Faty Nguirane, Ousmane Ndiaye, Mama Khady, Mame Ndiaye, Bocar Sylla, Mama Diakhou, Baye Mousse mais aussi à mes dames Awa Gueye, Rocky Cissé, Haby Camara et à leur fils pour tous leurs soutiens et conseils.*

*À ma **famille tuteur à Dakar** : Mère Ouly Sarr, Awa, Ndioba Gueye, Aissatou Gueye, Seynabou Gueye, Khady Gueye, ModouMouNdaw, Kara, pour tous ceux qu'ils ont fait pour moi.*

*Je tiens également à dédier ce travail à mes chères **tantes, oncles** pour leur soutien indéfectible et leur amour constant*

*À mes **promotionnaires et enseignants du lycée Sédo Sébé** : Dr. Oumar Diop, Dr Mariéme Ndom, Aly Sylla, Mamadou Diop, Coumba Thiongane, Demba Samb, Ndeye Bodé Samb, Faty Sambou, Faty Sow, Mame Goor Ndiaye, Mr Diao, Mr Ngom, …*

*À mes **promotionnaires de l'université** : Aminata Dia, Fatima Ba, Nazila, Modou, Serigne Mbacké Sow, Samba Traoré, Ousmane Sankharé, Ibrahima Thioune, Amadou Diallo, Arouna…*

*À mes **amis** : Ouly Bigué Gueye, Khady Thiongane, Alpha Ndiaye, Ndiaga Sine, Pape Diallo, AEECSS*

*À nos **ainés de la section informatique** : Aissatou Yatte, Jeanne D'arc Sagna, Fallou Sarr je vous dédie ce mémoire.*

*À mes **colocataires de l'appartement de Mogo** : filles comme garçons je vous dédie ce mémoire.*

Remerciements

Je remercie

ALLAH

Le très Haut, le très Grand, le Clément, L'Omniscient, l'Omnipotent.

Le Tout Puissant, le Très Miséricordieux d'avoir permis à ce travail d'aboutir à son terme.

*Au **PROPHETE MOHAMED** paix et salut sur lui.*

*Je tiens à exprimer ma sincère reconnaissance envers mon encadrant de mémoire, **Pr Cherif Diallo, Professeur Titulaire en Informatique** à **l'université Gaston Berger de Saint-Louis**, Son soutien indéfectible et son engagement tout au long de ce travail ont été d'une valeur inestimable. Sa disponibilité constante, sa bienveillance et ses conseils éclairés ont grandement contribué à la réussite de ce projet.*

Je suis reconnaissant pour ses orientations précieuses, sa modestie exemplaire et sa compréhension profonde de mes besoins académiques. Son expertise et son dévouement m'ont inspiré et motivé à donner le meilleur de moi-même.

Au corps professoral de la section informatique

Je tiens à exprimer ma sincère gratitude à l'ensemble du corps professoral pour leur contribution précieuse tout au long de mon parcours académique. Leur dévouement à l'enseignement et à l'encadrement des étudiants comme moi a été une source constante d'inspiration et de soutien.

À chacun des professeurs qui ont croisé ma route, je souhaite exprimer ma reconnaissance pour leur expertise, leur passion pour leur domaine, et leur engagement à nous aider à atteindre nos objectifs éducatifs. Leurs enseignements ont enrichi mon savoir et façonné ma vision du monde.

Je suis reconnaissant pour leur patience, leur encouragement et leurs conseils avisés qui m'ont guidé à travers les défis académiques. Leur soutien inconditionnel m'a permis de grandir sur le plan intellectuel et personnel.

Je tiens également à saluer leur volonté de partager leurs connaissances et leur expérience, ainsi que leur capacité à inspirer les générations futures. Leur impact sur la communauté éducative est immense et mérite d'être célébré.

Enfin, je souhaite exprimer ma profonde gratitude pour le temps et l'énergie qu'ils consacrent à leur métier, ainsi que pour leur engagement envers l'excellence académique. Leur dévouement ne passe pas inaperçu et je suis reconnaissant d'avoir eu l'opportunité de bénéficier de leur enseignement et de leur mentorat.

Merci à tous les professeurs pour leur contribution essentielle à mon développement personnel et professionnel. Leur influence durable continuera à guider mes pas dans les années à venir.

Résumé :

La détection précoce des attaques visant les services internet est aujourd'hui un enjeu stratégique majeur pour la sécurité des réseaux. Ce mémoire s'inscrit dans cette problématique en proposant une étude complète allant de la compréhension des différentes menaces jusqu'à la mise en œuvre de modèles de détection basés sur l'analyse du trafic réseau.

Le travail commence par une exploration des concepts fondamentaux liés aux attaques DDoS (Distributed Denial of Service), en insistant sur leur mécanisme, leur impact sur les infrastructures et l'évolution des méthodes d'attaque. L'objectif est de mettre en lumière la nécessité d'une détection rapide et efficace pour limiter les dommages causés par ces menaces.

L'étude se poursuit par une analyse des approches classiques de détection, mettant en évidence leurs limites face à l'ampleur et à la sophistication des attaques modernes. Cela conduit naturellement à l'introduction des techniques d'apprentissage automatique comme alternative performante pour détecter les anomalies dans le trafic réseau.

Afin de tester et valider ces approches, un dataset de référence, le **CIC-DDoS2019**, est utilisé. Ce jeu de données riche et varié permet de simuler des scénarios d'attaques réalistes. Un important travail de préparation des données est réalisé : nettoyage, sélection des caractéristiques pertinentes et normalisation, afin de garantir des conditions optimales pour l'entraînement des modèles.

Plusieurs techniques d'apprentissage supervisé sont ensuite mises en œuvre. Des modèles tels que les forêts aléatoires, les arbres de décisions et les régressions logistiques sont entraînés pour différencier le trafic normal des flux malveillants. Une évaluation rigoureuse est effectuée en utilisant des métriques comme l'exactitude, la précision, et le rappel afin de comparer les performances des différents algorithmes.

Les résultats expérimentaux mettent en évidence l'efficacité de certains modèles dans la détection rapide et fiable des attaques, avec des taux de détection élevés et de faibles taux de faux positifs. Le mémoire démontre ainsi qu'il est possible d'implémenter des systèmes intelligents capables de repérer précocement des comportements anormaux et d'alerter les administrateurs réseau avant que les attaques ne produisent des effets critiques.

Mots-clés : Détection précoce, cybersécurité, attaques DDoS, apprentissage automatique, analyse du trafic réseau, classification supervisée, détection d'anomalies, CIC-DDoS2019.

Abstract:

Early detection of attacks targeting Internet services has become a major strategic challenge for network security. This thesis addresses this issue by providing a comprehensive study ranging from understanding various threats to the implementation of detection models based on network traffic analysis. The work begins with an exploration of the fundamental concepts related to Distributed Denial of Service (DDoS) attacks, emphasizing their mechanisms, impact on infrastructures, and the evolution of attack techniques. The need for fast and effective detection to limit damage caused by such threats is highlighted.

The study continues with an analysis of traditional detection approaches, revealing their limitations in dealing with the scale and sophistication of modern attacks. This naturally leads to the introduction of machine learning techniques as a powerful alternative for detecting anomalies in network traffic. To test and validate these approaches, the CIC-DDoS2019 dataset is used, offering realistic attack scenarios. Significant data pre-processing is performed — including cleaning, feature selection, and normalization — to ensure optimal conditions for model training.

Several supervised learning techniques are implemented. Models such as random forests, decision trees, and logistic regression are trained to distinguish between normal traffic and malicious flows. A rigorous evaluation is conducted using metrics such as accuracy, precision, and recall to compare the performance of the various algorithms. The experimental results highlight the effectiveness of certain models in reliably and quickly detecting attacks, with high detection rates and low false positive rates. The thesis thus demonstrates the feasibility of implementing intelligent systems capable of early detection of abnormal behaviors and alerting network administrators before attacks cause critical damage.

Keywords: Early detection, cybersecurity, DDoS attacks, machine learning, network traffic analysis, supervised classification, anomaly detection, CIC-DDoS2019.

Table des matières

Dédicace I
Remerciements II
Résumé : III
Abstract: IV
Introduction Générale XII
Chapitre 1 : Cadre théorique et méthodologique 1
1.2 Contexte 1
1.2 Problématique 2
1.3 Objectifs 2
1.4 Pertinence du sujet 3
1.5 Délimitation du sujet 3
1.6 Difficultés rencontrées 3
Chapitre 2 : Les attaques DDOS 4
2.1 Introduction 4
2.2 Qu'est-ce qu'une attaque DDoS ? 4
2.3 Modes des attaques DDoS 5
2.3.1 Consommation des ressources limitées 5
2.3.2 Destruction ou modification des informations de configuration 6
2.3.3 Destruction physique et altération des composants du réseau 6
2.4 Les types d'attaques DDOS 7
2.4.1 Attaques basées sur le volume 7
2.4.1.1 L'attaque Amplification DNS 7
2.4.1.2 UDP Flood 9
2.4.2 Attaques de protocole 9
2.4.2.1 L'attaque SYN flood 9
2.4.3 Attaques d'application 10
2.5 Quelques vecteurs d'attaque 11
2.5.1 Les botnets 11
2.5.2 Les attaques basées sur la réflexion 11
2.5.3 Les attaques basées sur l'amplification 12
2.5.4 Les attaques ciblant des applications 13
2.6 Qui peut être visé 13

2.7 Les trois phases de gestion d'une crise DDoS 13
2.8 Les Mécanismes de détection 14
2.8.1 Mécanismes de détection basés sur les signatures 14
2.8.2 Mécanismes de détection basés sur les anomalies 15
2.8.3 Détection hybride 15
2.8.4 Détection de trace de source et d'usurpation 15
2.8.5 Filtrage basé sur le nombre 15
2.9 Les classes de mécanismes de détection d'anomalies 16
2.10 Conclusion 17
Chapitre 3 : Machine Learning 18
3.1 Introduction 18
3.2 Définition de L'intelligence artificielle 18
3.3 Définition de Machine Learning 19
3.3.1 Cycle de développement du Machine Learning 19
3.4 Types de systèmes d'apprentissage 21
3.4.1 Apprentissage supervisé 21
3.5 Algorithmes de Classification 23
3.5.1 Classification naïve de Bayes 23
3.5.2 Les arbres de décision 25
3.5.3 Régression logistique 27
3.5.4 Random Forest Classifier 28
3.6 Métriques utilisés 30
3.6.1 Accuracy 30
3.6.2 Classification reporte 30
3.7 Conclusion 31
Chapitre 4 Conception et études comparatives 32
4.1 Introduction 32
4.2 Conception de notre solution 32
4.3 Les Datasets 33
4.4 Entraînement du modèle 35
4.4.1 Google colab 35
4.4.2 AWS EC2 (Amazon Web Services) 36
4.4.3 Microsoft Azure 37
4.4.4 Kaggle 39
4.5 Les bibliothèques à télécharger 39

4.5.1 Sklearn 39
4.5.2 Pandas 40
4.5.3 Matplotlib 40
4.5.4 Seaborn 41
4.6 Conclusions 41
Chapitre 5 Mise en œuvre, validation et tests 42
5.1 Architecture proposé 42
5.2 Choix du data set 44
5.2.1 La réduction des données 46
5.3 Etape de l'implémentation 46
5.3.1 L'analyse exploratoire des données 46
5.3.2 Prétraitement 49
5.3.3 Modélisation 52
5.4 Résultat des Algorithmes de machine Learning 53
5.4.1 Application de l'algorithme « Random Forest » 53
5.4.2 Application de l'algorithme « Decision Tree » 55
5.4.3 Application de l'algorithme « Gaussienne NB » 58
5.4.4 Application de l'algorithme « Regression Logistic » 60
5.5 Comparaison et conclusion sur les algorithmes 64
5.6 Utilisation du modèle Random Forest 65
5.7 Conclusion 70
Conclusion générale et perspectives 71

Liste des figures

Figure 2.1 – Exemple d'attaque DDoS 5
Figure 2.2 – Exemple d'amplification 8
Figure 2.3 – Exemple d'attaques udp flood 9
Figure 2.4 – Exemple d'attaque de la couche d'application 10
Figure 2.5 – Principe d'une attaque par réflexion 11
Figure 2.6 – Les trois phases de gestion d'une crise ddos 14
Figure 3.1 – Intelligence artificial, Machine learning, Deep Learning 19
Figure 3.2 – Cycle de développement du Machine Learning 20
Figure 3.3 – Un jeu d'entraînement étiqueté pour un apprentissage supervisé 22
Figure 3.4 – arbres de décision 26
Figure 4.1 – Les étapes de machine Learning 32
Figure 5.1 – Architecture proposée 42
Figure 5.2 – les types d'attaques ddos dans dataset 2019 45
Figure 5.3 – Téléchargement des bibliothèques 46
Figure 5.4 – Importation des fichiers .csv 47
Figure 5.5 – Le nombre de ligne et de colonne 47
Figure 5.6 – visualisation de la target 48
Figure 5.7 – Sélection des colonnes 49
Figure 5.8 – Datafram 49
Figure 5.9 – Encodage 50
Figure 5.10 – Affichage apes encodage ..... Erreur ! Signet non défini.55
Figure 5.11 – Nettoyage 51
Figure 5.12 – Le nombres d'attributs 52
Figure 5.13 – Répartition des données 51
Figure 5.14 – importation des algorithmes 52
Figure 5.15 – Déclaration des algorithmes 52
Figure 5.16 – Résultat test « Random Forest » 53
Figure 5.17 – Matrice de Confusion « Random Forest » 55
Figure 5.18 – Résultat test « Arbre Décision » 56
Figure 5.19 – Matrice de Confusion « Decision Tree » 57
Figure 5.20 – Résultat test « Gaussienne NB » 58
Figure 5.21 – Matrice de Confusion « Naïves Bayes » 60
Figure 5.22 – Résultat test « Régression Logistic » 60

Figure 5.23 – Matrice de Confusion « Logistic Regression » 62
Figure 5.24 – résultats 63
Figure 5.25 – Sauvegarde du model « Random Forest »..... 66
Figure 5.26 – Chargement du model « Random Forest »..... 67
Figure 5.27 – Prédiction 68
Figure 5.28 – Envoie de mail d'alerte 69
Figure 5.29 – Envoie de mail d'alerte 69
Figure 5.30 – mail reçu par l'utilisateur 70

LISTE DES TABLEAUX

Table 2.1 – Comparaison des classes de mécanismes de détection d'anomalies **16**
Tableau 4.2 – Google Colab vs AWS EC2 vs Microsoft Azure **38**
Tableau 5.13 – Résultat test « Arbre Décision » **56**
Table 5.1 – Comparaison entre les algorithmes **64**

LISTE DES ABREVIATIONS

- **DDoS** : Déni de Service Distribué
- **DoS** : Déni de Service
- **TCP** : Protocole de Contrôle de Transmission
- **UDP** : Protocole de Datagramme Utilisateur
- **ICMP** : Protocole de Message de Contrôle Internet
- **OSI** : Modèle d'Interconnexion des Systèmes Ouverts
- **DNS** : Système de Noms de Domaine
- **CPU** : Unité Centrale de Traitement
- **RAM** : Mémoire Vive
- **IA** : Intelligence Artificielle
- **ML** : Apprentissage Automatique

Introduction Générale

Avec l'évolution rapide des technologies de l'information et de la communication, internet est devenu un pilier fondamental pour les entreprises, les institutions publiques et les utilisateurs individuels. Cependant, cette dépendance accrue à l'égard des services en ligne s'accompagne d'une augmentation significative des cybermenaces, notamment les attaques par déni de service distribué (DDoS). Ces attaques visent à rendre un service ou un réseau indisponible en submergeant ses ressources avec un flux massif de requêtes malveillantes.

Les attaques DDoS peuvent causer des perturbations majeures, affectant la disponibilité des services critiques tels que les sites web, les applications financières, les plateformes de commerce en ligne ou les infrastructures des opérateurs télécoms. En plus des pertes financières, ces attaques peuvent entraîner des dommages à la réputation et compromettre la confiance des utilisateurs.

Face à cette menace croissante, il devient essentiel de développer des mécanismes de détection précoce capables d'identifier rapidement les signes avant-coureurs d'une attaque DDoS afin de déclencher des mesures de protection appropriées. Parmi les techniques émergentes, l'intelligence artificielle (IA) et l'apprentissage automatique (Machine Learning) offrent des perspectives prometteuses pour analyser en temps réel les comportements réseau et détecter les anomalies indicatives d'une attaque.

Ce mémoire se propose d'étudier les différentes techniques de détection d'attaques DDoS et de mettre en œuvre une solution innovante basée sur l'apprentissage automatique. L'objectif principal est de concevoir un modèle prédictif performant en utilisant des algorithmes de classification appliqués à des jeux de données de référence tels que **CIC-DDoS2019**.

La méthodologie adoptée comprend le prétraitement des données, la sélection des caractéristiques pertinentes, l'entraînement de modèles prédictifs et l'évaluation de leurs performances à travers des métriques telles que la précision, le rappel et le score F1. Les résultats obtenus permettront de démontrer l'efficacité de la solution proposée en termes de détection rapide et précise des attaques DDoS.

Chapitre 1 : Cadre théorique et méthodologique

1.2 Contexte

Dans un monde de plus en plus connecté, les services internet jouent un rôle essentiel dans le fonctionnement des institutions publiques, des entreprises et de la société en général. Des secteurs critiques tels que l'e-commerce, l'e-gouvernement, les plateformes éducatives et les services financiers dépendent fortement de la disponibilité continue et de la sécurité de leurs services en ligne. La moindre interruption ou compromission de ces services peut avoir des conséquences majeures, allant de pertes financières à des atteintes à la réputation, en passant par des perturbations opérationnelles significatives.

Parmi les cybermenaces les plus redoutées figurent les attaques par déni de service distribué (**DDoS**), qui visent à rendre indisponibles des services en saturant leurs ressources réseau avec un flux massif de requêtes malveillantes. L'évolution constante des techniques utilisées lors des attaques DDoS, couplée à l'utilisation croissante des botnets et des techniques d'amplification, rend ces attaques de plus en plus difficiles à détecter et à atténuer avec les systèmes de sécurité traditionnels.

Les solutions de sécurité actuelles, souvent basées sur des signatures ou des règles statiques, peinent à détecter efficacement les nouvelles variantes d'attaques ou à répondre aux schémas de trafic anormaux. Dans ce contexte, la détection précoce des attaques DDoS devient un enjeu crucial pour anticiper les menaces et réagir avant que celles-ci ne causent des dommages significatifs aux infrastructures numériques.

Pour relever ce défi, l'utilisation de l'intelligence artificielle (**IA**) et de l'apprentissage automatique (**Machine Learning**) apparaît comme une solution prometteuse. Ces technologies permettent de passer d'une approche réactive à une stratégie proactive, capable d'analyser en temps réel les flux réseau, d'identifier les comportements anormaux et de prédire les attaques potentielles. Grâce à leur capacité d'apprentissage adaptatif, ces outils peuvent détecter des schémas d'attaque même inconnus auparavant, réduisant ainsi le risque de fausses alertes et augmentant la précision de la détection.

Ce mémoire s'inscrit dans cette démarche en étudiant les différentes techniques de détection précoce des attaques DDoS et en proposant une solution basée sur l'analyse comportementale et le machine Learning. En utilisant des jeux de données de référence tels que **CIC-DDoS2019**, cette étude vise à développer un modèle prédictif robuste, permettant d'améliorer significativement la résilience des services internet face aux attaques DDoS.

1.2 Problématique

La prolifération des attaques ciblant les services internet met en danger la disponibilité, l'intégrité et la confidentialité des données. Ces menaces, qui évoluent rapidement, nécessitent des systèmes de détection précoce efficaces pour minimiser les impacts. Cependant, plusieurs défis techniques subsistent, notamment en termes de précision, rapidité, et capacité à s'adapter à de nouveaux types d'attaques. Dès lors, la question centrale est : **comment concevoir et mettre en œuvre une solution de détection précoce adaptée aux services internet modernes ?**

D'où les questions **techniques** suivantes qui déterminent la problématique de notre mémoire.

- Quels algorithmes de machine Learning ou d'intelligence artificielle sont les plus efficaces pour détecter les attaques émergentes sur les services internet en temps réel ?
- Comment optimiser les performances du système de détection (en termes de précision, rappel, et temps de réponse) pour minimiser les faux positifs et négatifs ?
- Quels outils ou Framework peuvent être utilisés pour collecter, analyser, et interpréter les données réseau ou applicatives pertinentes pour la détection d'anomalies ?
- Comment concevoir une architecture modulaire et scalable qui puisse s'intégrer facilement dans des infrastructures existantes et s'adapter à des environnements diversifiés ?

1.3 Objectifs

Objectif General

Notre objectif dans ce travail est de concevoir et mettre en œuvre une solution de détection précoce des attaques, en combinant des techniques modernes de machine Learning et d'analyse des données pour identifier, prévenir et atténuer les menaces avant qu'elles ne causent des dommages significatifs.

Objectifs spécifiques

Pour atteindre l'objectif général énoncé ci-dessus, les étapes (sous objectifs) du projet sont les suivantes :

- Évaluer les solutions actuelles de détection des attaques.
- Concevoir un système efficace de détection précoce basé sur des techniques modernes.
- Mettre en œuvre, tester et valider le système

1.4 Pertinence du sujet

Notre étude se porte sur une étude et mise en œuvre de solution de détection précoce d'attaque DDos, la gestion d'un moteur d'apprentissage performant et durable. L'idée est de proposer une approche d'intelligence artificielle, évolutif après avoir mis en place un système de détection.

1.5 Délimitation du sujet

Comme nous l'avons dit plus haut, le but de ce travail consiste à une étude et mise en œuvre de solution de détection précoce d'attaque DDos. Le point de départ de notre méthode de recherche était la revue de la littérature. Comme pour tout autre projet, l'état de l'art est un élément essentiel pour situer la question principale de recherche par rapport aux anciens travaux réalisés sur le même sujet. Cela nous permettra :
De distinguer les problématiques qui ont été débattues sur la détection précoce

- D'avoir une idée claire sur les approches d'intelligence artificielle
- De savoir ce qui doit être remis en cause

Par la suite, nous allons à la Contributions, qui est le but principal de cette mémoire constitue :

- Détecter des attaques qui sont en cours ;
- Conception d'un système de détection ;
- Implémentation de ce système.

1.6 Difficultés rencontrées

Les difficultés que j'ai eu à rencontrer est le problème de traiter les datasets que j'ai déjà téléchargé avec leur nombreux fichier à faire le prétraitement des données.

Chapitre 2 : Les attaques DDOS

2.1 Introduction

Les attaques par déni de service distribué (DDoS) comptent parmi les menaces les plus sérieuses pour la sécurité des systèmes d'information. Leur objectif principal est de rendre un service ou une infrastructure réseau indisponible en le saturant avec un trafic malveillant massif. Au fil des années, ces attaques ont gagné en ampleur, en complexité et en efficacité, rendant leur détection et leur atténuation de plus en plus difficiles.

2.2 Qu'est-ce qu'une attaque DDoS ?

Une attaque par déni de service distribué (DDoS – Distributed Denial of Service) vise à rendre un serveur indisponible en l'inondant de requêtes malveillantes, dépassant ainsi ses capacités de traitement. L'objectif peut être d'affecter les services en ligne ou le réseau d'une entreprise en saturant différentes ressources du système [1] :

- **Bande passante** : en générant un volume de trafic excessif.
- **Espace de stockage** : en saturant les journaux ou bases de données.
- **Puissance de traitement** : en sollicitant excessivement le processeur et la mémoire RAM.
- **Sessions TCP** : en ouvrant un grand nombre de connexions incomplètes.
- **Traitements concurrents** : en surchargeant les files d'attente des bases de données.

Le DDoS est une extension de l'attaque DoS (Denial of Service), mais avec une capacité de nuisance bien plus élevée en raison de l'utilisation de plusieurs machines zombies (botnets). La figure 2.1 illustre un exemple simple d'une attaque DDoS.

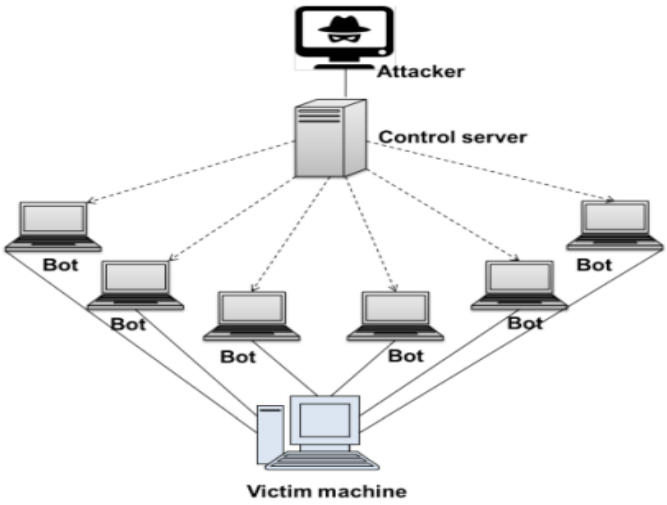


Figure 2.1 – Exemple d'attaque DDoS

2.3 Modes des attaques DDoS

Les attaques DDoS peuvent être classées en plusieurs catégories en fonction des techniques utilisées.

2.3.1 Consommation des ressources limitées

Les infrastructures informatiques nécessitent diverses ressources pour fonctionner correctement (mémoire, bande passante, puissance de traitement, etc.). Une attaque DDoS peut viser à saturer ces ressources de différentes manières :

- **Attaque sur la Connectivité réseau :**

Dans ce type d'attaque, l'attaquant empêche les utilisateurs légitimes d'accéder au service. Un exemple est l'attaque "SYN Flood", où l'attaquant envoie de nombreuses demandes de connexion TCP sans les finaliser, forçant le serveur à allouer des ressources inutiles jusqu'à l'expiration des connexions incomplètes. Quand l'attaquant est perdu un serveur victime avec un grand nombre de tels paquets, le serveur doit conserver la structure des données dans sa mémoire tous ces paquets jusqu'à ce que la minuterie expire. Par conséquent, le serveur est toujours occupé et ne peut pas être servi autres demandes légitimes pendant cette période.

- **Utilisation des ressources des victimes contre elles-mêmes :**

L’attaquant manipule des machines pour qu’elles communiquent en boucle entre elles, comme dans l’attaque UDP, où des paquets falsifiés créent un dialogue incessant entre serveurs [2].

- **Saturation de la bande passante :**

L’attaquant peut rediriger un grand nombre de paquets vers le réseau de la victime, consommant tout sa bande passante entrante. Ces paquets peuvent être quoi que ce soit, généralement utilisés sont les paquets ICMP Echo. Pour rendre cette attaque efficace, un attaquant peut utiliser plusieurs machines pour envoyer un grand nombre de paquets vers la victime.

- **Saturation d’autres ressources**

De nombreux attaquants peuvent tenter de consommer des ressources autres que la bande passante du réseau, comme la puissance de traitement ou la mémoire. Le serveur peut stocker également temporairement des informations dans ses données structures et un attaquant peut exploiter ce mécanisme en demandant au serveur d’allouer informations par diverses demandes à la fois. Serveur le stockage regorge d’informations relatives aux attaquants, rendant le système indisponible pour desservir d’autres demandes. L’attaquant peut également demander en générant divers processus pour le CPU qui doivent attendre pour être exécuté. Les exemples sont les bombardements par e-mail, générant intentionnellement des erreurs qui doivent être consigné.

2.3.2 Destruction ou modification des informations de configuration

Si certaines informations de configuration sont modifiées ou retiré de la machine, alors on ne peut pas utiliser la machine correctement et il se peut que la machine ne puisse pas servir les autres utilisateurs qui s’y sont connectés. L’attaquant tente de s’introduire dans la machine en modifiant sa configuration et rendre le système inaccessible aux utilisateurs généraux. Par exemple, si un attaquant est capable de changer de routage informations de table d’un routeur mal sécurisé, puis d’autres les utilisateurs peuvent ne pas être en mesure d’atteindre la machine connectée [2].

2.3.3 Destruction physique et altération des composants du réseau

Dans certains cas extrêmes, un attaquant peut cibler physiquement l'infrastructure, en détruisant des équipements critiques tels que des routeurs ou des serveurs.

2.4 Les types d'attaques DDOS

Il existe de nombreux types d'attaques DDoS et les hackers en utilisent souvent plusieurs pour infliger des ravages à leurs cibles. Les trois principaux types d'attaques sont les attaques volumétriques, protocolaires et au niveau de la couche applicative. L'objectif de toutes les attaques est de ralentir fortement ou d'empêcher le trafic légitime d'atteindre sa destination. Par exemple, cela peut consister à empêcher un utilisateur d'accéder à un site Web, d'acheter un produit ou un service, de regarder une vidéo ou d'interagir sur les médias sociaux. En outre, en rendant les ressources indisponibles ou en diminuant les performances, les attaques DDoS peuvent entraîner un arrêt de l'activité. Cela peut avoir pour conséquence d'empêcher les employés d'accéder au courrier électronique, aux applications Web ou de mener leurs activités comme d'habitude.

Pour mieux comprendre le fonctionnement des attaques DDoS, examinons les différentes voies que les hackers peuvent emprunter. Le modèle d'interconnexion des systèmes ouverts (OSI) est une structure par couches pour diverses normes de réseau. Le modèle OSI est un cadre conceptuel pour décrire les connexions réseau dans sept couches différentes :

Les attaques DDoS sont classées dans trois grandes catégories en fonction de leur cible :

2.4.1 Attaques basées sur le volume

Il s'agit du type d'attaque DDoS le plus courant, les attaquants essaient de consommer toute la bande passante du réseau en envoyant un nombre volumineux de paquets à un serveur victime afin que cela devienne très difficile pour utilisateurs légitimes d'accéder au serveur en raison d'un blocage à routeurs d'interface. Les attaques volumétriques utilisent des botnets créés à partir d'armées de systèmes et de dispositifs individuels infectés par des logiciels malveillants. Contrôlés par un attaquant, les bots sont utilisés pour provoquer la congestion d'une cible et d'Internet en général avec un trafic malveillant qui sature toute la bande passante disponible.
L'ampleur de ces types d'attaques est mesurée en bits par seconde. Exemple : Amplification DNS, UDP Flood, ICMP Flood, autres inondations de paquets usurpés [3].

2.4.1.1 L'attaque Amplification DNS :

Il s'agit d'une attaque DDos (déni de service distribué) volumétrique, fondée sur la réflexion, lors de laquelle un pirate exploite la fonctionnalité de résolveurs DNS ouverte pour surcharger un serveur ou un réseau cible avec une quantité de trafic amplifié, afin de rendre inaccessibles le serveur et son infrastructure environnante [4].

L'attaque par amplification DNS peut être divisée en quatre étapes :

- Le pirate utilise un point de terminaison compromis pour envoyer des paquets UDP à un récurseur DNS à l'aide d'adresses IP usurpées. L'adresse usurpée contenue dans les paquets renvoie vers l'adresse IP réelle de la victime.
- Chacun des paquets UDP envoie une requête à un résolveur DNS, en transmettant souvent un argument du type « ANY », afin de recevoir la réponse la plus volumineuse possible.
- Après avoir reçu ces requêtes, le résolveur DNS (qui essaie d'effectuer son travail en répondant) envoie une réponse volumineuse à l'adresse IP usurpée.
- L'adresse IP de la cible reçoit la réponse et l'infrastructure de réseau environnante se retrouve submergée sous un flot de trafic, provoquant ainsi un déni de service [4].

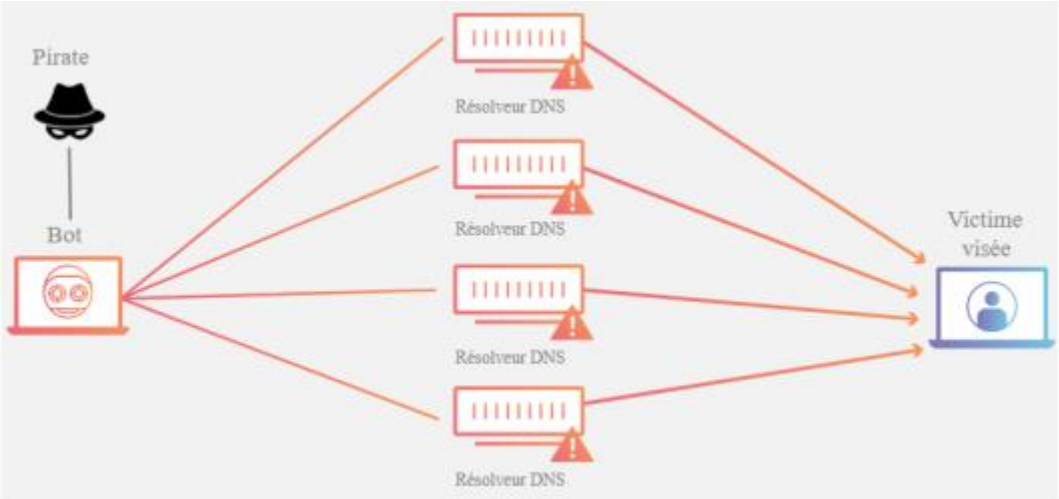


Figure 2.2 – Exemple d'amplification

2.4.1.2 UDP Flood

UDP Flood est un type d'attaque par déni de service dans laquelle un grand nombre de paquets UDP (User Datagram Protocol) sont envoyés à un serveur ciblé dans le but d'écraser la capacité de cet appareil à traiter et à répondre. Le pare-feu protégeant le serveur ciblé peut également s'épuiser à la suite d'une inondation UDP, entraînant un déni de service du trafic légitime [4].

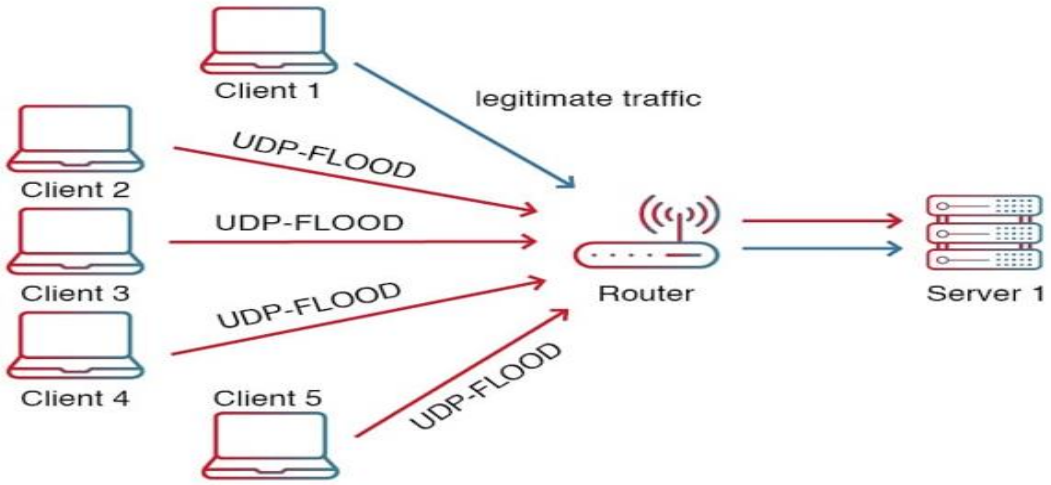


Figure 2.3 – Exemple d'attaques udp flood

2.4.2 Attaques de protocole

Les attaques de protocole consomment les ressources de la machine victime comme la mémoire, la capacité de traitement, etc... Peut créer de longues files d'attente ou d'autres structures de données sur les périphériques réseau intermédiaires tels que les pares-feux, les routeurs ou équilibreurs de charge, etc. Ampleur du protocole les attaques sont mesurées en paquets par seconde. Exemple : SYN Flood, attaque de paquets fragmentés.

2.4.2.1 L'attaque SYN flood

Une attaque SYN flood est un type d'attaque par déni de service (DDoS) qui vise à rendre un serveur indisponible pour le trafic légitime en consommant toutes les ressources serveur disponibles. En envoyant à plusieurs reprises des paquets de demande de connexion initiale (SYN), le pirate est en mesure de submerger tous les ports disponibles sur une machine serveur ciblée, ce qui oblige l'appareil ciblé à répondre lentement au trafic légitime, ou l'empêche

totalement de répondre. L'attaques SYN flood fonctionne en utilisant le processus d'établissement de liaison des connexions TCP.

Généralement, une connexion TCP a trois processus indépendants pour établir une connexion.

- Tout d'abord, le client envoie un paquet SYN au serveur pour établir une connexion.
- Le serveur répond alors à ce paquet initial avec un paquet SYN/ACK pour confirmer la réception de la communication.
- Enfin, le client renvoie un paquet ACK pour confirmer la réception du paquet depuis le serveur. Après avoir terminé cette séquence d'envoi et de réception de paquets de données, la connexion TCP est ouverte et les données peuvent être envoyées et reçues [4].

2.4.3 Attaques d'application

Considérées comme étant les types d'attaques DDoS les plus sérieuses et sophistiquées, ces attaques ciblent les applications web en exploitant les vulnérabilités qu'elles comportent. Aussi appelées « Attaques couche 7 », les attaques d'application fonctionnent toujours de la même manière mais demandent beaucoup moins de force car elles ciblent les faiblesses au sein des serveurs ciblés. Beaucoup moins de trafic web est nécessaire pour monopoliser les processus et protocoles sur ces points faibles, ce qui rend aussi l'attaque bien plus difficile à détecter en raison du faible volume de trafic généré qui semble légitime [3].

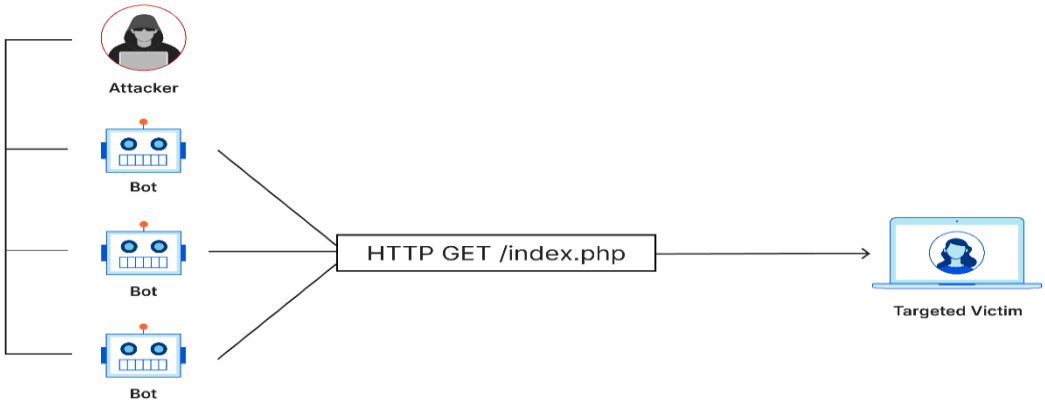


Figure 2.4 – Exemple d'attaque de la couche d'application

2.5 Quelques vecteurs d'attaque

2.5.1 Les botnets

Les attaques DDoS peuvent être lancées à partir du réseau de la machine infectée Cela s'appelle un botnet. De nombreux outils disponibles en ligne peuvent tirer parti des botnets [5]. Dans ces dernières années, des services DDoS en ligne (communément appelés Bootstrap ou stress) ont vu le jour. Le prix de ces services permet utilisé par des individus et permet aux utilisateurs de lancer des attaques opposez-vous au but qu'il a choisi. En outre, certains programmes de démarrage fournissent également des services de test gratuit pendant quelques minutes. La diversité des outils et services permet lancer une attaque par déni de service distribué peut aider à augmenter beaucoup de ces attaques.

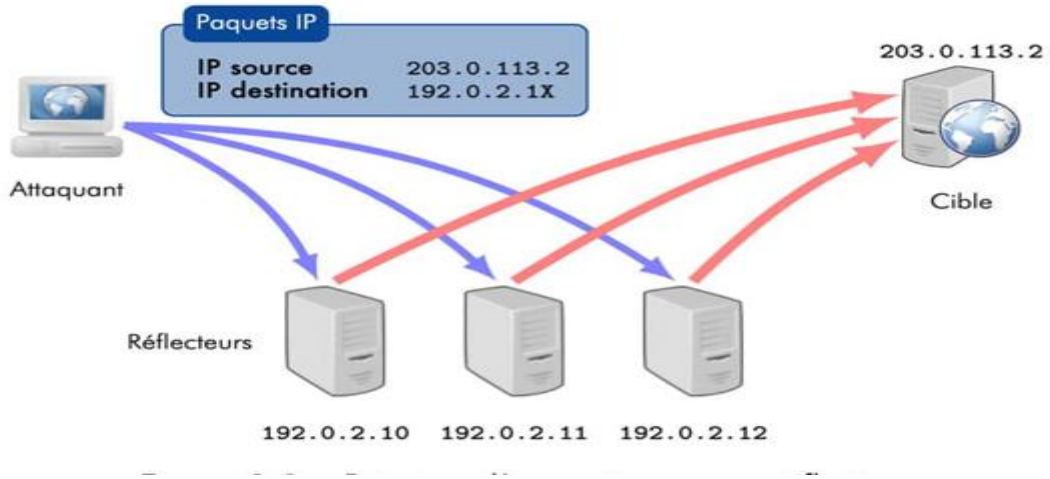


Figure 2.5 – Principe d'une attaque par réflexion

2.5.2 Les attaques basées sur la réflexion

Certaines attaques utilisent des ordinateurs accessibles sur internet et répondent demandes de n'importe quelle source : ce sont des réflecteurs. L'attaque de La réflexion consiste à envoyer des paquets à ces réflecteurs en utilisant les adresses IP suivantes : La victime est l'adresse IP source : c'est ce qu'on appelle l'usurpation d'adresse IP. Cette réaction de ces réflecteurs à la victime peut induire un trafic non sollicité de la destination de ce dernier.

Ce débit peut être suffisamment important pour saturer lien réseau de la victime, qui conduit à un déni de service. L'attaquant interroge le serveur en usurpant l'adresse IP ses victimes (203.0.113.2). Par conséquent, le serveur envoie la réponse à la demande faite par l'agresseur à la victime. Si le déni de service est effectif, Le trafic causé par ces réponses dépasse la bande passante du réseau disposition de la victime. Les attaques par réflexion impliquent généralement des protocoles basés sur des protocoles transmission UDP. En fait, le protocole UDP laisse la tâche à la couche application identifié la source afin que l'usurpation d'adresse IP puisse être effectuée. Au fait, UDP Il n'est pas nécessaire d'établir une session avant d'envoyer des données (contrairement à TCP). Cette fonctionnalité permet à l'attaquant d'utiliser UDP pour demander un service via un seul paquet de données et générer un la réponse du réflecteur. Les attaques par réflexion ne se limitent pas aux protocoles de transmission UDP. Par exemple, vous pouvez envoyer des paquets TCP SYN en usurpant l'adresse IP de la victime génère un paquet SYN-ACK en réponse à la cible. Enfin, il convient de noter que les attaques par réflexion peuvent être lancées à partir de botnets.

2.5.3 Les attaques basées sur l'amplification

Le but d'une attaque par lots est d'épuiser la bande passante réseau disponible pour rendre un ou plusieurs services inaccessibles.

Ce type d'attaque est généralement réalisé en utilisant les propriétés de certains protocoles pour maximiser le trafic généré. De plus, le but des attaques de volume est de produire très grand nombre de paquets de données par seconde pour saturer les ressources de traitement le but. La réponse générée par certains protocoles est beaucoup plus grande que réclamer. Le nombre de paquets provoqués par la réponse peut également être plus grand dépassement du nombre de paquets requis pour envoyer la demande. L'amplification produit grâce à ces protocoles, il peut être utilisé pour mener des attaques de volume. Dans la plupart des cas, les attaques volumétriques utilisent la réflexion et l'amplification. De nombreux protocoles peuvent être utilisés pour implémenter ce genre d'attaque. Parmi eux, on peut notamment citer DNS (Domain Name System), NTP (Network Time Protocol), SNMP (Simple Network Management) Protocole), SSDP (Simple Service Discovery Protocol) ou CHARGEN (Protocole Générateur de Caractères). Il est important de noter qu'une entité peut être victime d'une attaque de volume utilisez le protocole, même s'il n'a aucun service actif exposé sur internet Basé sur le même accord.

2.5.4 Les attaques ciblant des applications

Certaines attaques visent à épuiser les capacités de traitement d'une cible. Par exemple, un attaquant peut chercher à atteindre la limite du nombre de connexions concurrentes qu'un serveur web peut traiter. Dans ce cas, l'attaquant envoie en permanence un grand nombre de requêtes HTTP GET ou POST au serveur ciblé. Il est également possible d'envoyer des requêtes partielles, puis de transmettre la suite de ces requêtes à intervalles réguliers, dans le but de maintenir les connexions ouvertes le plus longtemps possible et d'éviter la fermeture des connexions au-delà d'un délai fixé. D'autres types d'attaques applicatives cherchent à épuiser les ressources de calcul d'un serveur en initiant un grand nombre de sessions TLS, ou encore à tirer parti de faiblesses dans la conception d'une application web.

2.6 Qui peut être visé

Toute entité dont l'activité dépend de l'infrastructure réseau connectée à Internet peut être la cible d'attaques DDoS. Les motivations et les objectifs des assaillants sont variés, allant des revendications idéologiques à la vengeance en passant par l'extorsion. En outre, il semble que certaines attaques aient été menées pour détourner l'attention et dissimuler d'autres activités illégales, telles que des transactions bancaires frauduleuses. Bien que de nombreuses entités soient affectées par cette menace, certains types d'activités sont plus vulnérables aux cibles DDoS. Parmi eux, on peut citer notamment le e-commerce, les institutions financières, le gouvernement et même les structures d'hébergement informatique. Dans ce cas, il est particulièrement important de prévoir des solutions de protection adaptées dès le début d'un projet de mise en œuvre d'un système d'information et d'une infrastructure réseau. Les attaques DDoS sont très courantes aujourd'hui. Par exemple, les opérateurs français ont observé plus d'un millier d'attaques chaque jour en 2014.

En outre, certains rapports publics indiquent que le nombre de ces attaques est en augmentation. En plus de l'augmentation du nombre, l'ampleur des attaques a également augmenté de manière significative ces dernières années.

2.7 Les trois phases de gestion d'une crise DDoS

La plupart des articles comme quand ils résolvent les solutions des attaques ddos ils classent les phases de gestion d'une crise DDoS en trois phases : avant l'attaque, pendant l'attaque et après l'attaques, le schéma ci-dessous montre les détails de chaque phase :

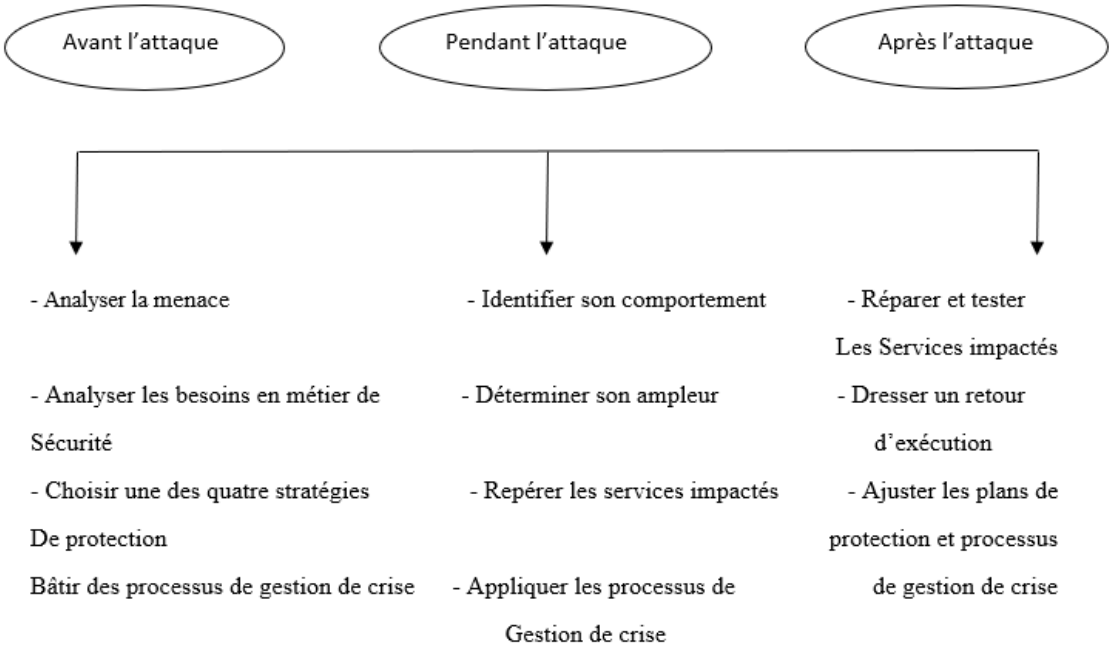


Figure 2.6 – Les trois phases de gestion d'une crise ddos

2.8 Les Mécanismes de détection

2.8.1 Mécanismes de détection basés sur les signatures

La détection basée sur la signature implique un ensemble de signatures d'attaque de différents modèles d'attaques connues. Les signatures des modèles d'attaque seront stockées dans une base de données.

Tout le trafic entrant est comparé aux signatures existantes qui sont stockées dans la base de données pour détecter les modèles de trafic d'attaque. Ces mécanismes consistent à détecter avec précision attaques connues si la base de données est toujours mise à jour. En raison de la facilité de reconfiguration, les règles sont nécessaires pour mettre à jour les signatures dans la base de données pour les attaques inconnues. Cependant, les attaques inconnues ne peuvent pas être détectées si les signatures ne sont pas mises à jour. Comme un résultat, toute variation de la signature d'attaque stockée connue ou des attaques obscures entraînera un fort taux de faux négatifs. De plus, la mise à jour des signatures pour chaque attaque n'est pas faisable à mesure que les attaques changent.

2.8.2 Mécanismes de détection basés sur les anomalies

La détection d'anomalies implique l'identification de modèles dans les données qui sont incompatibles avec comportement attendu. Ces modèles qui ne sont pas compatibles sont appelés anomalies modèles. Ces mécanismes visent à identifier des événements qui semblent anormaux dans le comportement normal du système. Normalement, le comportement du trafic est collecté sur une certaine période de temps pour comparer avec le trafic réseau entrant.

2.8.3 Détection hybride

La combinaison de deux ou plusieurs mécanismes est utilisé par la détection hybride. Tels les mécanismes de détection hybride comprennent l'utilisation à la fois des mécanismes - signature et détection basée sur les anomalies pour augmenter le taux de détection. Les forces et les limites de ces mécanismes de détection dépendent de l'algorithme utilisé par une telle détection mécanismes. Ces mécanismes de détection sont capables de classer les règles avec précision, en raison de l'avantage combiné de plusieurs techniques. Cependant, les performances et le coût de calcul de ces mécanismes de détection dépendent du nombre de techniques de détection combinées.

2.8.4 Détection de trace de source et d'usurpation

La technique de traçage est utile pour trouver la source à partir de laquelle les attaques DDoS sont généré. Ces attaques ont tendance à utiliser des adresses frauduleuses pour lancer des attaques DDoS. Une attaque par réflecteur est un exemple de cette attaque. Le traçage de la source et l'usurpation sont extrêmement important pour les mécanismes de détection. Cependant, ce processus nécessite un service les fournisseurs prennent en charge et plusieurs composants réseau, tels que les serveurs et les routeurs de périphérie.

2.8.5 Filtrage basé sur le nombre

Les paramètres de ressource réseau sont utilisés pour indiquer le début de l'attaque dans le filtrage basé sur le nombre et sont ensuite utilisés pour détecter une occurrence d'attaque telle que le nombre de sauts, nombre de connexions et nombre de requêtes pour une seule source dans une unité de temps. La principale force de ces mécanismes de détection peut être facilement déployée, ces mécanismes permettre aux administrateurs de surveiller rapidement la

situation. Cependant, il faut une base de données pour être continuellement mis à jour et avoir également une variété d'applications hétérogènes dans différents systèmes qui causent des problèmes lors de l'utilisation de ces méthodes, et cela nécessite également un étalonnage du taux de fausses alarmes. En outre, il souffre également d'usurpation d'adresse IP qui conduit à des problèmes d'intégrité et de précision.

2.9 Les classes de mécanismes de détection d'anomalies

Les mécanismes de détection d'anomalies sont classés en cinq classes selon la classification d'en fonction des algorithmes utilisés dans le processus de détection, à savoir l'apprentissage automatique, exploration de données, intelligence artificielle, classificateurs, et statistique. La figure suivante présente une comparaison de la détection d'anomalies classes basées sur les algorithmes utilisés dans le processus de détection ainsi que leurs forces et limites.

Table 2.1 – Comparaison des classes de mécanismes de détection d'anomalies

Classes	Avantages.	Limites
Basé sur l'apprentissage automatique	• Détection des modèles d'attaque DDoS à haute efficacité. •cela peut changer leur stratégie d'exécution tout au long de la détection	•Des frais généraux élevés entraînent baisse des performances.
Basé sur Exploration de données	• Augmenter la vitesse de détection processus. • Faible coût de calcul. • Il peut gérer une énorme base de données.	• Inefficace en cas de forte volume du réseau entrant circulation. • Valeurs manquantes de l'ensemble de données influencera la détection processus.
Basé sur l'Intelligence Artificiel	• Classer les comportements comme normaux ou Intrusif. • Détecter efficacement les attaques DDoS.	• La précision de détection dépend du profil de formation. • Problèmes d'évolutivité
Statistique Basé	• Apprentissage du comportement attendu de observations sans connaissance préalable des activités normales. • Détection précise des programmes malveillants Activité	•Difficulté de fixer une valeur optimale seuil. •Besoin de justifier les hypothèses qui sont nécessaires pour déterminer le taux de classement.

2.10 Conclusion

Les attaques par déni de service distribué (DDoS) constituent une menace majeure pour la disponibilité des services en ligne et des infrastructures réseau. Leur évolution en termes de complexité et d'efficacité rend leur détection et leur atténuation de plus en plus difficiles. Ces attaques exploitent différentes stratégies, allant de la saturation des ressources réseau et système à la manipulation des protocoles de communication pour perturber le bon fonctionnement des serveurs. Face à cette menace croissante, les entreprises et les organisations doivent adopter des mesures de protection efficaces, telles que la mise en place de pare-feu avancés, l'utilisation de solutions de détection d'intrusion et l'application de bonnes pratiques de sécurité. L'analyse du trafic réseau, la surveillance en temps réel et le recours à des services de protection DDoS permettent de limiter les impacts de ces attaques et de garantir la disponibilité des services pour les utilisateurs légitimes. En conclusion, la lutte contre les attaques DDoS nécessite une approche proactive combinant technologies de défense, bonnes pratiques de cybersécurité et une vigilance constante face aux nouvelles méthodes employées par les attaquants.

Chapitre 3 : Machine Learning

3.1 Introduction

Le Machine Learning, ou plus largement l'intelligence artificielle (dont il est un sous-domaine), évoque souvent la science-fiction pour beaucoup de personnes. Pourtant, l'apprentissage automatique n'est pas un concept futuriste : il est déjà bien ancré dans notre quotidien. En réalité, il existe depuis plusieurs décennies, comme en témoignent les filtres anti-spam des années 80 ou encore l'intégration du Machine Learning dans les jeux vidéo dès le début des années 2000.

Aujourd'hui, le Machine Learning est omniprésent et a profondément transformé notre monde, notamment avec des innovations telles que la voiture autonome de Google, la reconnaissance vocale de Siri, l'algorithme de recommandation de Facebook ou encore le concept de magasin autonome d'Amazon.

3.2 Définition de L'intelligence artificielle

Selon le Larousse, l'intelligence artificielle se définit comme « l'ensemble des théories et techniques mises en œuvre en vue de réaliser des machines capables de simuler l'intelligence ». Il s'agit donc d'ordinateurs ou de machines équipés de programmes leur permettant d'effectuer des tâches comparables à celles de l'intelligence humaine, voire de les surpasser grâce à la technologie.

Ces machines sont capables de :

- Raisonner.
- Traiter de grandes quantités de données.
- Détecter des modèles invisibles à l'œil humain.
- Comprendre et analyser ces modèles.
- Interagir avec l'Homme.
- Apprendre progressivement.

- Améliorer continuellement leurs performances [6].

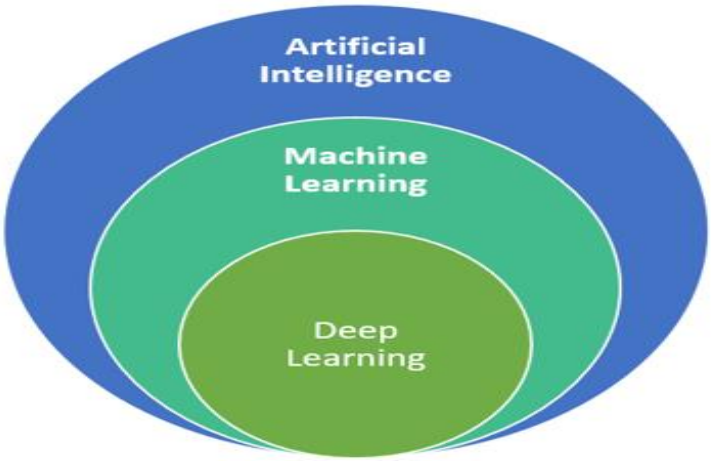


Figure 3.1 – Intelligence artificial, Machine learning, Deep Learning

3.3 Définition de Machine Learning

Le Machine Learning, ou apprentissage automatique en français, est une branche de l'intelligence artificielle qui permet aux ordinateurs d'apprendre sans être explicitement programmés. Cette technologie repose sur le développement de modèles capables d'évoluer lorsqu'ils sont exposés à de nouvelles données.

Le Machine Learning est une méthode d'analyse de données permettant d'automatiser la construction de modèles analytiques. Grâce à des algorithmes capables d'apprendre de manière itérative, il permet aux ordinateurs d'identifier des tendances cachées sans qu'ils aient été programmés pour les rechercher [7].

3.3.1 Cycle de développement du Machine Learning

Les domaines d'application du Machine Learning sont variés : prédiction des valeurs financières, cybersécurité, moteurs de recherche influencés par le profil utilisateur, détection de vols, implémentation d'antivirus ou encore cryptanalyse.

Le cycle de développement d'un modèle de Machine Learning suit les étapes suivantes :

1. Obtention et nettoyage des données.

2. Conception du modèle.

3. phase d'apprentissage.

4. phase de validation.

5. phase d'exécution.

Le développement d'un modèle de Machine Learning ne se fait pas en une seule tentative. On commence souvent par un modèle simple et rapide à concevoir, puis on l'évalue afin de détecter des problèmes de variance ou de biais. En fonction des résultats, des ajustements sont réalisés pour améliorer les performances du modèle. La figure 3.2 illustre ce cycle de développement.

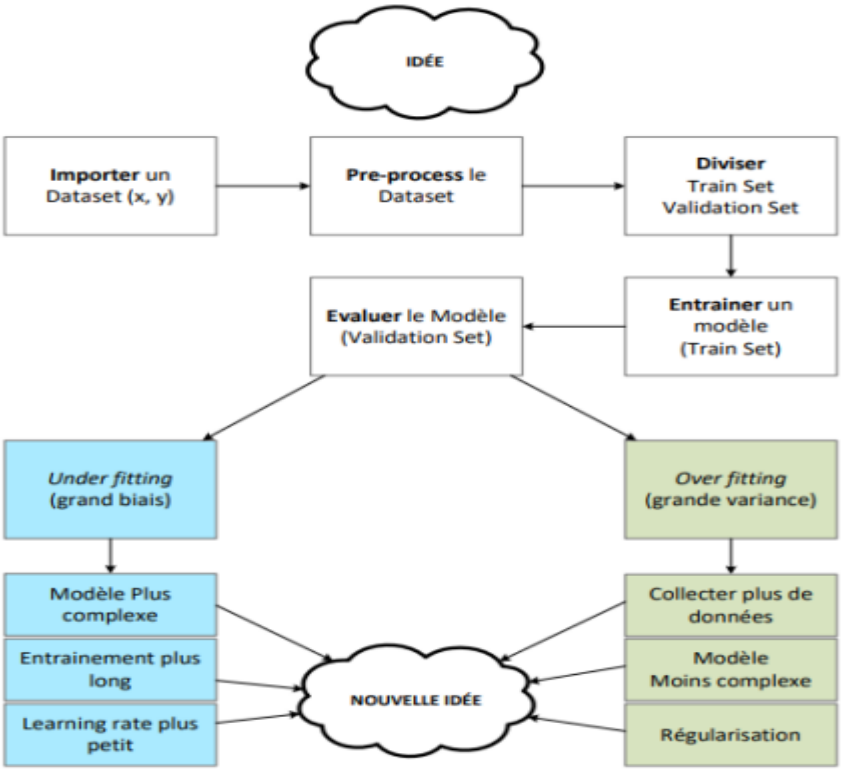


Figure 3.2 – Cycle de développement du Machine Learning

3.4 Types de systèmes d’apprentissage

Il existe plusieurs types de systèmes d’apprentissage, qui varient en fonction du type de problème à résoudre. Ils peuvent être classés selon le degré et la nature de la supervision nécessaire durant la phase d'entraînement.

3.4.1 Apprentissage supervisé

L’apprentissage supervisé est un type d’apprentissage automatique où l’algorithme apprend à partir de données d’entraînement étiquetées (associées à une réponse attendue). L’objectif est de prédire une variable de sortie (Y) en fonction d’une variable d’entrée (X).

Parmi les algorithmes les plus courants en apprentissage supervisé, on retrouve :

- Les machines à vecteurs de support (SVM).
- La régression linéaire.
- La régression logistique.
- L’algorithme naïf bayésien.
- L’algorithme des k plus proches voisins (KNN).

• Un exemple d’utilisation de l’apprentissage supervisé est le filtre anti-spam, l’apprentissage s’effectue à l’aide de nombreux exemples d’e-mails qu’on a étiqueté spam ou normal, à partir de cela, le filtre doit alors être capable de classer de nouveaux e-mails.

• Un autre exemple consiste à prédire le prix d’une voiture à partir des valeurs d’un certain nombre d’attributs ou variables qu’on appelle caractéristiques d’une observation ou features en anglais. Ces variables peuvent être le kilométrage, l’âge, la marque, etc. Ils sont également appelés variables explicatives ou prédictives. L’entraînement se fait alors à partir de ces variables et des étiquettes. La catégorie de la variable prédite Y fait décliner [8] [9] [10].

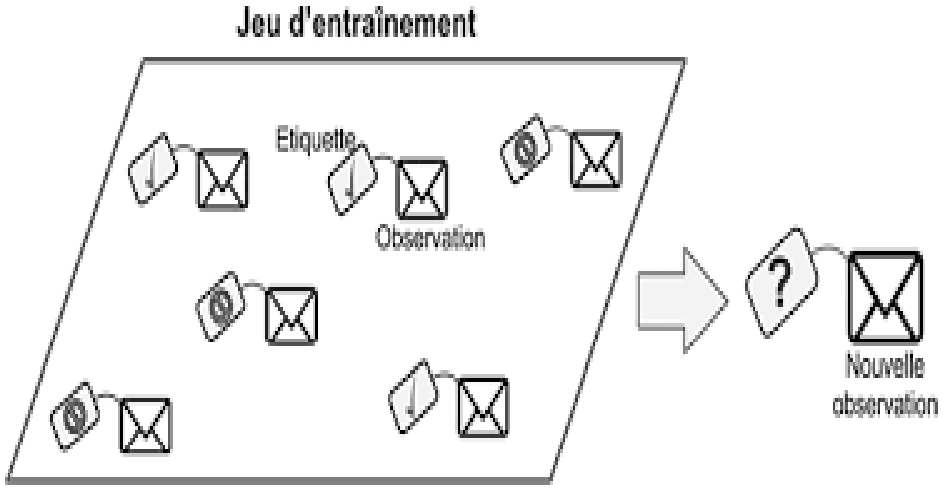


Figure 3.3 – Un jeu d'entraînement étiqueté pour un apprentissage supervisé

Il existe deux types d'apprentissage supervisé :

❖ Classification :

Un problème de classification survient lorsque la variable de sortie est une catégorie, telle que « rouge », « bleu » ou « maladie » et « pas de maladie ». Exemples :

• En finance et dans le secteur bancaire pour la détection de la fraude par carte de crédit (fraude, pas fraude).

• Détection de courrier électronique indésirable (spam, pas spam).

• Dans le domaine du marketing utilisé pour l'analyse du sentiment de texte (heureux, pas heureux).

• En médecine, pour prédire si un patient a une maladie particulière ou non.

❖ Régression :

Un problème de régression se pose lorsque la variable de sortie est une valeur réelle, telle que « dollars » ou « poids ». Exemples :

• Prédire le prix de l'immobilier.

• Prédire le cours de bourse.

3.5 Algorithmes de Classification

La classification est une technique d'exploration de données permettant d'attribuer des observations à différentes catégories prédéfinies. De nombreux algorithmes de classification existent, chacun utilisant des méthodes mathématiques spécifiques comme les arbres de décision, la programmation linéaire ou les réseaux de neurones. Ces techniques analysent les données disponibles de diverses manières pour faire des prédictions. Cependant, certains sont plus couramment utilisées que d'autres :

3.5.1 Classification naïve de Bayes

Le modèle de classification naïve de Bayes est basé sur le théorème de Bayes, qui est une règle mathématique fondamentale utilisée pour calculer des probabilités conditionnelles.

Dans le contexte de la classification, ce modèle suppose que les caractéristiques utilisées pour la classification sont indépendantes les unes des autres, ce qui peut être une simplification souvent irréaliste, mais qui permet néanmoins d'obtenir des résultats efficaces dans de nombreux cas.

Le théorème de Bayes calcule la probabilité qu'un événement se produise étant donné la probabilité qu'un autre événement s'est déjà produit. Le théorème de Bayes s'exprime mathématiquement sous la forme de l'équation suivante [11] :

$$\text{Formule de probabilité : } P(A|B) = \frac{P(A, B)}{P(B)} = \frac{P(B|A)\,P(A)}{P(B)}$$

La règle de Bayes stipule que, si l'évènement B peut être influencé par l'évènement A, alors le fait que B a eu lieu permet également de prévoir la survenue de A.
Avec, P(A) et P(B) les probabilités pour que les évènements A et B se produisent respectivement. P (A, B) représente la probabilité pour que A et B surviennent tous les deux. P(A|B) (respectivement P(B|A)) est la probabilité de survenue de l'évènement A, sachant que

l’évènement B a eu lieu (respectivement la probabilité de survenue de l’évènement B, sachant que l’évènement A eu lieu).

Fonctionnement

- **Théorème de Bayes** :

Le modèle de Bayes calcule la probabilité qu'un exemple appartienne à une classe donnée en utilisant le théorème de Bayes. Il est basé sur deux éléments
Principaux :

La probabilité a priori des classes et la probabilité conditionnelle des caractéristiques données la classe.

- **Indépendance Conditionnelle** :

L'hypothèse d'indépendance conditionnelle, bien que simplificatrice, permet au modèle de calculer efficacement les probabilités des classes en se basant sur les probabilités des caractéristiques individuelles [12].

- **Prédiction** :

Pour prédire la classe d'un nouvel exemple, le modèle calcule la probabilité a posteriori de chaque classe en utilisant les caractéristiques de cet exemple. La classe avec la probabilité a posteriori la plus élevée est alors sélectionnée comme prédiction.
Cette méthode a différents avantages et inconvénients :

❖Avantages

- Facile à mettre en œuvre et efficace en termes de calcul.
- Efficace dans les cas comportant un grand nombre de fonctionnalités.
- Fonctionne bien même avec des données d'entraînement limitées [13].
- L’algorithme offre de bonne performance.
- Un réseau bayésien est polyvalent, nous pouvons nous servir du même modèle pour évaluer, prévoir, diagnostiquer, ou optimiser des décisions.

- La représentation graphique d'un réseau bayésien est explicite, intuitive et compréhensible par un non spécialiste, ce qui facilite à la fois la validation du modèle, ses évolutions éventuelles et surtout son utilisation.
- Les réseaux bayésiens donnent la possibilité de rassembler et de fusionner des connaissances de diverses natures dans un même modèle.

❖Inconvénients

- Suppose que les fonctionnalités sont indépendantes, ce qui n'est pas toujours le cas dans les données réelles.
- Peut être influencé par des attributs non pertinents.
- Peut attribuer une probabilité nulle à des événements invisibles, conduisant à une mauvaise généralisation.
- La généralité du formalisme des réseaux bayésiens aussi bien en termes de représentation que d'utilisation les rend difficiles à manipuler à partir d'une certaine taille.
- La prédiction devient erronée si l'hypothèse d'indépendance conditionnelle est invalide.

3.5.2 Les arbres de décision

Cette technique divise le problème de classification en sous-problèmes. Il crée un arbre de décision qui est utilisé pour développer un modèle utilisé aux fins de la classification. Effectuer une classification est relativement simple.
Il suffit de répondre aux questions en descendant dans l'arbre de décision.
Les arbres de décision sont des modèles non-paramétriques et trouvent des règles en général assez puissantes. Ils peuvent traiter des très grands dataset et ils peuvent aussi utiliser des prédicateurs mixtes (catégoriques et nombres). Les variables redondantes sont éliminées, parce que l'arbre ne les prend même pas en compte A chaque nœud, on doit donc trouver deux choses : quelle variable 'feature' utiliser et quel est le point de séparation des deux zones, de sorte à minimiser l'erreur quadratique pour la régression et l'impureté pour la classification. On fait ainsi grandir l'arbre de la manière le plus grand possible [14].

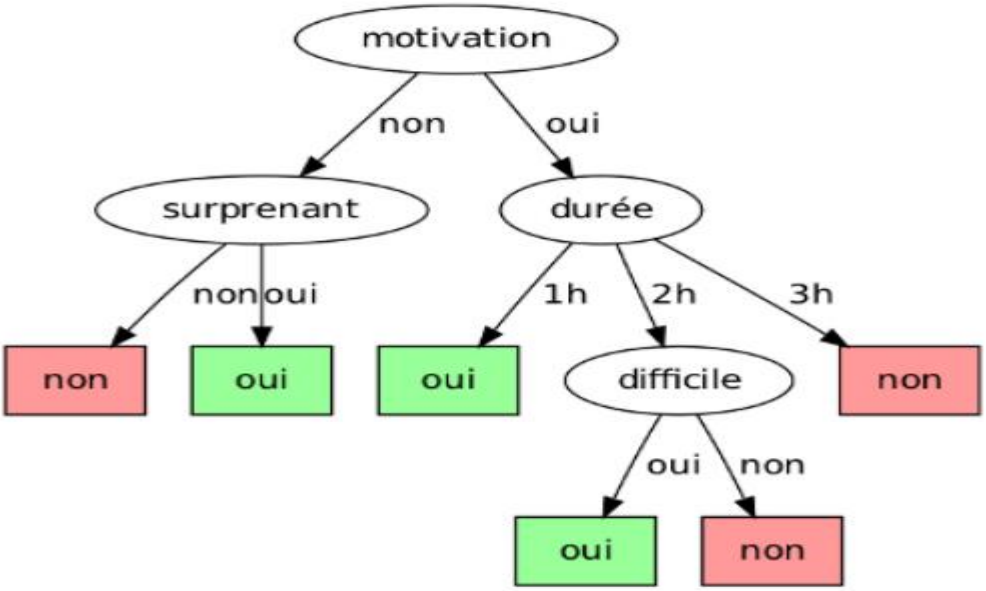


Figure 3.4 – arbres de décision

Fonctionnement de l'algorithme de l'arbre de décision

L'arbre de décision fonctionne en analysant l'ensemble de données pour prédire sa classification. Cela commence au nœud racine de l'arborescence, où l'algorithme visualise la valeur de l'attribut racine par rapport à l'attribut de l'enregistrement dans l'ensemble de données réel. Sur la base de la comparaison, il suit la branche et passe au nœud suivant.

L'algorithme répète cette action pour chaque nœud suivant en comparant ses valeurs d'attribut avec celles des sous-nœuds et en poursuivant le processus. Il se répète jusqu'à ce qu'il atteigne le nœud feuille de l'arbre.

Le mécanisme complet peut être mieux expliqué grâce à l'algorithme donné ci-dessous.

Étape 1 : Commencez l'arborescence avec le nœud racine, dit S, qui contient l'ensemble de données complet.

Étape 2 : Trouvez le meilleur attribut de l'ensemble de données à l'aide de la mesure de sélection d'attribut (ASM).

Étape 3 : Divisez le S en sous-ensembles contenant les valeurs possibles pour les meilleurs attributs.

Étape 4 : générez le nœud de l'arbre de décision, qui contient le meilleur attribut.

Étape 5 : Créez de manière récursive de nouveaux arbres de décision en utilisant les sous-ensembles de l'ensemble de données créé à l'étape -3. Continuez ce processus jusqu'à ce qu'une

étape soit atteinte où vous ne pouvez pas classer davantage les nœuds et appelez le nœud final en tant qu'algorithme d'arbre de classification et de régression de nœud feuille.

❖Avantages

- Simple à comprendre et à interpréter.
- Capable de traiter à la fois des données quantitatives et qualitatives.
- Performant sur de grands jeux de données.
- Cela peut être très utile pour résoudre des problèmes liés à la décision.
- Il y a moins d'exigences de nettoyage des données par rapport aux autres algorithmes.

❖Inconvénients

- L'apprentissage par arbre de décision peut amener des arbres de décision très complexes, qui généralisent mal l'ensemble d'apprentissage.
- L'arbre de décision contient de nombreuses couches, ce qui le rend complexe.
- Il peut y avoir un problème de su rajustement, qui peut être résolu à l'aide de l'algorithme Random Forest.
- Pour plus d'étiquettes de classe, la complexité informatique de l'arbre de décision peut augmenter.

3.5.3 Régression logistique

La régression logistique est une technique de classification couramment utilisée en apprentissage automatique pour modéliser la probabilité qu'une donnée appartienne à une ou plusieurs classes. Elle est particulièrement efficace pour les problèmes de classification binaire, mais peut également être étendue à la classification multi label.

- Régression logistique binaire :

Dans cette approche, la réponse ou la variable dépendante est dichotomique par nature, c'est-à-dire qu'elle n'a que deux résultats possibles (par exemple, 0 ou 1). Parmi les exemples populaires de son utilisation, citons la prédiction du caractère spam ou non d'un e-mail, ou de la nature maligne ou non d'une tumeur. Dans le champ de la régression logistique, il s'agit de l'approche la plus couramment utilisée et, plus généralement, de l'une des méthodes de classification binaire les plus courantes.

❖Avantages

- La régression logistique est plus facile à mettre en œuvre, à interpréter et très efficace à entraîner.
- Il ne fait aucune hypothèse sur la distribution des classes dans l'espace des fonctionnalités.
- Bonne précision pour de nombreux ensembles de données simples et fonctionne bien lorsque l'ensemble de données est linéairement séparable.

❖Inconvénients

- Si le nombre d'observations est inférieur au nombre d'entités, la régression logistique ne doit pas être utilisée, sinon cela pourrait conduire à un surajustement.
- Il construit des limites linéaires.
- Il ne peut être utilisé que pour prédire des fonctions discrètes. Par conséquent, la variable dépendante de la régression logistique est liée à l'ensemble de nombres discrets.

3.5.4 Random Forest Classifier

Random Forest est un algorithme d'apprentissage automatique d'ensemble qui combine plusieurs arbres de décision pour créer un modèle prédictif plus robuste et plus précis.
Il a été introduit comme une amélioration par rapport aux arbres de décision uniques afin de réduire le su rajustement et d'améliorer les performances prédictives [15].
En termes simples, l'approche de forêt aléatoire augmente les performances des arbres de décision. C'est l'un des meilleurs algorithmes car il peut utiliser à la fois des techniques de classification et de régression. En tant qu'algorithme d'apprentissage supervisé, la forêt aléatoire utilise la méthode du bagging dans les arbres de décision et, par conséquent, augmente la précision du modèle d'apprentissage [15] [16] [17].

Fonctionnement

Le fonctionnement d'une forêt aléatoire (Random Forest) repose sur plusieurs concepts clés en apprentissage automatique et statistique. Voici une explication détaillée de son fonctionnement

- **Échantillonnage Bootstrap (Bagging)** :

L'échantillonnage bootstrap consiste à générer plusieurs échantillons de données à partir du jeu de données d'entraînement en tirant aléatoirement avec remplacement. Cela signifie qu'un même exemple peut être inclus plusieurs fois dans un échantillon bootstrap, tandis que d'autres exemples peuvent être omis. Cela crée des échantillons uniques et diversifiés, chacun étant utilisé pour entraîner un arbre de décision différent.

- **Construction d'arbres de décision** :

Chaque arbre de la forêt aléatoire est construit en utilisant un échantillon bootstrap. L'arbre est développé de manière récursive en divisant les données en sous-groupes homogènes à chaque nœud, selon la caractéristique qui maximise la pureté de la division (par exemple, l'indice de Gini pour la classification ou la réduction de variance pour la régression).

- **Sélection aléatoire des caractéristiques** :

Pour chaque nœud de chaque arbre, au lieu de considérer toutes les caractéristiques possibles pour décider de la meilleure division, on sélectionne un sous-ensemble aléatoire de caractéristiques. Ce sous-ensemble est généralement choisi avant la construction de chaque arbre et reste fixe pendant la croissance de l'arbre. La taille de ce sous-ensemble est contrôlée par le paramètre max_features [12].

- **Vote majoritaire (Classification) / Moyenne (Régression)** :

Pour faire des prédictions avec une forêt aléatoire, les prédictions de chaque arbre individuel sont agrégées. En classification, la classe prédite est déterminée par un vote majoritaire parmi tous les arbres de la forêt. En régression, la prédiction finale est la moyenne des prédictions de tous les arbres.

❖Avantages

- Robustesse : En combinant plusieurs arbres, le Random Forest réduit le risque de surapprentissage par rapport à un seul arbre de décision.
- Flexibilité : Il peut gérer des données avec des caractéristiques continues et catégorielles.
- Réduction de la variance : La moyenne des prédictions de plusieurs arbres de décision indépendants tend à avoir une variance plus faible que celle d'un seul arbre.

❖Inconvénients

- Complexité : En raison du grand nombre d'arbres, le modèle peut devenir complexe et coûteux en termes de calcul et de mémoire.
- Interprétabilité : Contrairement à un arbre de décision unique, le Random Forest est moins interprétable, car il est composé de nombreux arbres

3.6 Métriques utilisés

3.6.1 Accuracy

Dans la classification multi-étiquette, cette fonction calcule la précision des sous-ensembles : l'ensemble d'étiquettes prédites pour un échantillon doit correspondre exactement à l'ensemble d'étiquettes correspondant dans y-true.

3.6.2 Classification reporte

Sklearn. métriques. Classification-report : résumé textuel de la précision, rappel, score f1, pour chaque classe. Dictionnaire retourné si output-dict est true. Le dictionnaire a la structure suivante :

- Précision :

La précision est le rapport entre les vrais positifs et le total des vrais positifs et des faux positifs. La précision regarde pour voir combien de junk positifs ont été jetés dans le mélange. S'il n'y a pas de mauvais positifs (ces fps), alors le modèle a eu la précision 100%. Plus il y a de fps qui entrent dans le mélange, plus la précision sera laide. Mathématiquement [18] :

$$\text{Précision} = \frac{\text{TP}}{\text{TP} + \text{FP}}$$

- **TP (True Positives)** : Nombre de vrais positifs, c'est-à-dire les cas où le modèle a correctement prédit la classe positive.
- **FP (False Positives)** : Nombre de faux positifs, c'est-à-dire les cas où le modèle a prédit une classe positive alors que la classe réelle était négative.

- Recall :

Le rappel est la mesure de notre modèle identifiant correctement les vrais positifs. Ainsi, pour tous les patients qui souffrent réellement d'une maladie cardiaque, le rappel nous indique combien nous avons correctement identifiés comme ayant une maladie cardiaque. Mathématiquement

$$\text{Rappel} = \frac{\text{TP}}{\text{TP} + \text{FN}}$$

- **FN (False Négatives)** : Nombre de faux négatifs, c'est-à-dire les cas où le modèle a prédit une classe négative alors que la classe réelle était positive.

- F1 Score :

Le **F1-Score** combine la précision et le rappel en une seule métrique. Il s'agit de la moyenne harmonique entre ces deux mesures, donnant un équilibre entre les fausses alertes et les faux rejets.

$$F1 = 2 \times \frac{\text{Précision} \times \text{Rappel}}{\text{Précision} + \text{Rappel}}$$

- Lorsque la précision et le rappel sont équilibrés, le F1-Score est élevé.
- Si l'une des deux valeurs (précision ou rappel) est faible, le F1-Score diminue considérablement.

3.7 Conclusion

Ce chapitre nous a permis d'avoir une idée globale sur l'intelligence artificielle, l'importance de machine Learning, le types de systèmes d'apprentissage et les algorithmes de classification avec leurs avantages et inconvénients.

Chapitre 4 Conception et études comparatives

4.1 Introduction

Comme mentionné précédemment dans le chapitre trois, le machine learning est omniprésent. Dans ce chapitre, nous suivons les étapes d'un projet d'apprentissage automatique dans le but de détecter les attaques DDoS et d'obtenir des résultats satisfaisants.

4.2 Conception de notre solution

Notre solution repose sur la détection des attaques DDoS à l'aide du machine learning. Pour ce faire, nous devons suivre les étapes classiques du processus d'apprentissage automatique, à savoir : le nettoyage, le prétraitement des données, la normalisation, l'apprentissage, la validation et l'exécution. Le fonctionnement du système de détection proposé est illustré dans l'organigramme présenté dans la figure suivante :

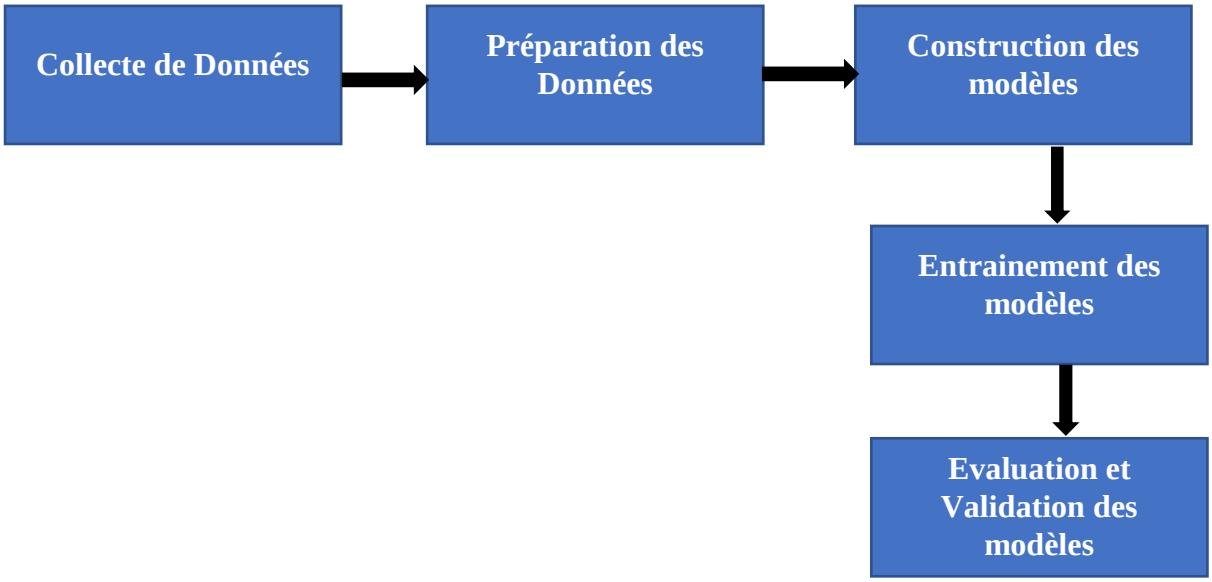


Figure 4.1 – Les étapes de machine Learning

4.3 Les Datasets

L'évaluation de tous les algorithmes d'apprentissage automatique proposés dépend des ensembles de données utilisés dans les études publiées sur l'application du machine learning en cybersécurité. En raison de problématiques liées à la confidentialité, certains de ces ensembles de données ne sont pas librement accessibles. Voici quelques exemples de datasets dédiés à la détection des attaques :

- KDD Cup'99

KDD Cup'99 Les ensembles de données de découverte d'interruption KDD Cup'99 qui dépendent absolument de l'ensemble de données DARPA '98 donnent un ensemble de données nommé à l'analyste s'exécutant à l'intérieur de la zone d'identification d'interruption et parlent à l'ensemble de données nommé librement accessible. Le jeu de données KDD'99 est fait de l'utilisation d'une reproduction d'un système militaire. Enfin, il existe un renifleur qui enregistre toutes les informations d'activité du système transmises en utilisant l'arrangement Tcp dump. L'ensemble de données de préparation KDD contient environ 4 900 000 vecteurs d'association uniques, qui intègrent tous 41 qualités et sont classés comme agression ou typique, avec correctement un type d'agression indiqué. Les agressions imitées sont réparties dans les quatre classes suivantes : agressions par déni de service (Dos), sonde, distant vers local (r2l) et utilisateur vers racine (u2r).

- NSL-KDD

NSL-KDD Pour résoudre les problèmes de l'ensemble de données de la KDD Cup, ils ont proposé un nouvel ensemble de données, à savoir NSL-KDD, qui consiste en des enregistrements sélectionnés de l'ensemble de données complet de la KDD Cup'99. Voici les avantages de l'ensemble de données NSL-KDD par rapport à l'ensemble de données KDD Cup'99 :

✓ Il n'inclut pas les enregistrements non pertinents dans la rame, de sorte que les classificateurs ne seront pas partisans d'enregistrements plus répétés à partir de chaque enregistrement de difficulté, le nombre d'enregistrements choisis est inversement proportionnel au pourcentage d'enregistrements dans l'ensemble de données KDD. Ainsi, il en résulte que le pourcentage de classification des différentes techniques ML (Machine Learning) diffère dans une large gamme. Cela rend l'évaluation complète et structurée des approches de ML.

✓ Dans l'ensemble de données d'apprentissage et de test, le nombre d'enregistrements est logique, ce qui facilite la réalisation des expériences sur l'ensemble de données sans avoir à choisir de petits segments aléatoires. Par conséquent, les résultats d'évaluation des différents travaux seront stables.

➢ MAWI

MAWI L'ensemble de données MAWI contient des traces de trafic quotidiennes de 15 minutes avec des en-têtes de transport couvrant la dernière décennie. Bien que l'ensemble de données soit disponible pour la communauté au sens large depuis un certain temps, la courte durée de chaque trace l'a prêté à une étude plus approfondie dans les zones. La présence de traces de flux complètes est moins importante, comme les anomalies Internet ou lorsque la caractérisation du trafic est basée sur les paquets, reposant uniquement sur l'inspection de l'en-tête IP et des numéros de port.

➢ CICIDS2017

CICIDS2017 Il s'agit d'un ensemble de données public accessible gratuitement à (http://www.unb.ca/cic/datasets/IDS2017.html). Il se compose de données réelles qui ont été collectées sur la base du comportement d'un réseau de 25 utilisateurs basé sur les protocoles FTP, HTTPS, HTTP, e-mail et SSH. CICIDS2017 inclut le trafic d'attaques malveillantes bénignes et différentes telles que l'infiltration, l'attaque Web, le botnet et le saignement cardiaque..., etc. Nous avons sélectionné le fichier qui contient une attaque de botnet (FridayWorkingHours-Morning.pcap_ISCX) et l'avons testé dans notre proposition maquette. Ce fichier comprend 191033 trafic bénin et 1966 trafics Botnet.

➢ CIC-DDOS-2019

CIC-DDoS-2019 Il s'agit de véritables données de flux réseau avec plusieurs formes d'attaques DDOS les plus récentes et les plus courantes) regroupe tous les paquets dans une fenêtre temporelle qui partagent des attributs spécifiques mais ne transportent pas de charge utile. Il existe deux types de données dans l'ensemble de données : les données PCAP brutes et les données CSV. Les auteurs ont extrait plus de 80 caractéristiques des fichiers PCAP à l'aide de l'analyseur de trafic CICFlowMeter-V3, et les résultats ont été enregistrés dans des fichiers CSV formatés et étiquetés par l'Université du Nouveau-Brunswick [21].

Voici un tableau comparatif des principales options pour les datasets :

Tableau 4.1 – Différents Dataset

Critère	CIC-DDoS-2019	CICIDS2017	MAWI	NSL-KDD	KDD Cup'99
Accessibilité	Taille importante (~20 Go), nécessite un espace de stockage et une puissance de traitement	Taille importante (~50 Go), complexité de traitement	Trafic réel, non labellisé, difficile à utiliser pour l'apprentissage supervisé	Léger (~300 Mo), facile à télécharger et à utiliser	Léger (~700 Mo), largement utilisé comme référence
Efficacité	Spécifique aux attaques DDoS, utile pour la recherche sur ce type de menace	Diversifié, inclut plusieurs types d'attaques modernes	Trafic réel, utile pour des analyses en temps réel	Version améliorée de KDD Cup'99, moins de redondance	Large adoption, base de référence historique
Temps de traitement	Temps de traitement élevé en raison de la taille et de la complexité des données	Temps de traitement très élevé en raison de la taille et de la diversité des données	Temps de traitement variable, dépend de la collecte en temps réel	Temps de traitement faible en raison de la petite taille et de la simplicité	Temps de traitement faible en raison de la petite taille et de la simplicité

4.4 Entraînement du modèle

L'entraînement du modèle est une étape cruciale dans le développement d'un modèle de **machine learning**. Il consiste à apprendre à partir des données pour réaliser une tâche spécifique comme la classification, la régression ou la reconnaissance de motifs.

L’apprentissage d’un modèle implique des calculs intensifs, en particulier sur des matrices de grande dimension. Pour optimiser cette phase, plusieurs plateformes ont été envisagées en fonction des ressources et des coûts.

4.4.1 Google colab

Google colab est une plateforme cloud gratuite qui permet aux utilisateurs d'accéder à des ressources puissantes, telles que des GPU et TPU, pour exécuter des tâches de calcul intensif. Cela élimine la nécessité de disposer d'une infrastructure locale coûteuse. Colab offre également un environnement de travail basé sur Jupiter Notebook, ce qui facilite l'exécution et le partage de code Python. Les utilisateurs peuvent collaborer en temps réel, ce qui en fait un outil pratique pour les projets en équipe. Grâce à ses ressources cloud, Google Colab simplifie l'accès à des technologies avancées de calcul.

Avantages :

- ✓ Coût nul pour les projets académiques.
- ✓ Interface conviviale et aucune configuration complexe requise.
- ✓ Partage facile des notebooks entre collaborateurs.

Limites :

- ✓ Restrictions sur les ressources, notamment une durée limitée des sessions.
- ✓ Performances potentiellement limitées pour des projets de très grande envergure.

4.4.2 AWS EC2 (Amazon Web Services)

AWS EC2 (Elastic Compute Cloud) offre une infrastructure cloud flexible qui permet de créer et de configurer des serveurs virtuels selon les besoins spécifiques des utilisateurs. Il propose des instances avec des GPU puissants, idéales pour l’entraînement de modèles de machine Learning à grande échelle. Cette solution permet de déployer facilement des applications et d’ajuster les ressources en fonction des exigences de performance. AWS EC2 offre également une large gamme de configurations et de tarifs pour s’adapter à différents types de projets. Grâce à sa scalabilité, AWS EC2 est une option de choix pour des besoins computationnels variés.

Avantages :

- ✓ Capacité à traiter des datasets volumineux.
- ✓ Options de personnalisation étendues pour s'adapter aux besoins du projet.
- ✓ Large écosystème de services connexes (S3 pour le stockage, Sage Maker pour la machine Learning).

Limites :

- ✓ Coûts élevés pour des ressources puissantes.
- ✓ Courbe d'apprentissage plus élevée pour les développeurs peu familiers avec AWS.

4.4.3 Microsoft Azure

Microsoft azure est une plateforme cloud qui fournit une infrastructure robuste et performante pour le développement et l'exécution de projets d'intelligence artificielle. Elle propose une gamme de services, notamment des machines virtuelles, des GPU et des outils d'apprentissage automatique pour faciliter l'entraînement des modèles. Azure permet une gestion flexible des ressources et offre des options de scalabilité pour s'adapter aux besoins des utilisateurs. Il dispose également d'une intégration étroite avec d'autres services Microsoft, facilitant la mise en œuvre des solutions. Grâce à sa fiabilité et ses performances, Azure est une solution prisée pour les projets IA à grande échelle.

Avantages :

- ✓ Grande variété d'options de configuration.
- ✓ Services spécialisés pour la machine Learning (Azure ML).
- ✓ Bonne compatibilité avec des projets à grande échelle.

Limites :

- ✓ Modèle de tarification complexe et souvent onéreux.
- ✓ Besoin de connaissances préalables pour une utilisation optimale.

Voici un tableau de Comparaison de ces Infrastructures d'entraînement du modèle

Tableau 4.2 – Google Colab vs AWS EC2 vs Microsoft Azure

Critère	Google colab	AWS EC2	Microsoft Azure
Accessibilité	Accès rapide et sans configuration locale	Nécessite une configuration initiale et une gestion manuelle des instances	Nécessite une configuration initiale et une gestion manuelle des instances
Coût	Gratuit pour l'utilisation de base, y compris l'accès à des GPU/TPU	Payant base sur l'utilisateur	Payant avec des crédits gratuits pour les nouveaux utilisateurs
Performance	Bonnes performances pour des projets académiques et de taille modérée	Très bonne performance, adaptée aux projets de grande envergure.	Très bonne performance, adaptée aux grande projets
Adaptabilité	Idéal pour des projets de petite à moyenne taille, facile à utiliser	Idéal pour des projets à grande échelle	Idéal pour des projets à grande échelle avec des besoins personnalisées

Choix Final : Google colab

Pour ce projet, **Google colab** a été sélectionné comme plateforme principale pour l'entraînement. Pour l'accès dans ce service il nous suffit simplement d'avoir un compte Google. Pour le langage de développement on a choisi Python, un langage de programmation interprété, multi-paradigme et multi-plateformes, il est aussi un langage plus commun et plus populaire pour l'apprentissage automatique et l'intelligence artificielle grâce à sa flexibilité et aussi parce qu'il a un nombre important de bibliothèques logicielles open source disponible. Permet ses bibliothèques utilisées dans notre projet : Pandas, Numpy, Scikit-Learn etc.

Ce choix repose sur les critères suivants :

- **Accessibilité** : Google Colab est gratuit et facilement accessible, ce qui est idéal pour les projets académiques.

- **Infrastructure GPU/TPU gratuite** : Permet un entraînement rapide et efficace sans coûts supplémentaires.

4.4.4 Kaggle

Kaggle est une plateforme web organisant des compétitions en science des données appartenant à Google. Sur cette plateforme, les entreprises proposent des problèmes en science des données et offrent un prix aux data logistes obtenant les meilleures performances.

4.5 Les bibliothèques à télécharger

4.5.1 Sklearn

Scikit-learn, souvent abrégé en sklearn, est une bibliothèque open source très populaire en Python dédiée à l'apprentissage automatique (machine Learning). Elle offre des outils simples et efficaces pour la modélisation prédictive, l'analyse de données et d'autres tâches liées à l'apprentissage automatique. Scikit-learn est construit sur d'autres bibliothèques populaires de l'écosystème Python telles que NumPy et Matplotlib.
Quelques caractéristiques et fonctionnalités importantes de Scikit-learn incluent :

- ✓ **Algorithmes d'Apprentissage** :

Scikit-learn propose une variété d'algorithmes d'apprentissage supervisé et non supervisé, notamment pour la classification, la régression, le clustering, la réduction de dimensionnalité, et plus encore. Parmi les algorithmes de classification, on trouve arbres de décision, k plus proches voisins, régression logistique, etc.

- ✓ **Prétraitement des Données** :

La bibliothèque offre des outils pour le prétraitement des données, y compris la normalisation, la standardisation, la gestion des valeurs manquantes, l'extraction de caractéristiques, etc.

- ✓ **Évaluation des Modèles** :

Scikit-learn propose des outils d'évaluation de modèles tels que les métriques de performance (précision, rappel, F1-score, etc.) et des fonctions pour la validation croisée.

- ✓ **Sélection de Modèles** :

Il inclut des utilitaires pour la sélection de modèles, la recherche d'hyperparamètres et la validation croisée, contribuant ainsi à une meilleure performance des modèles.

- ✓ **Intégration avec NumPy et SciPy** :

Scikit-learn s'intègre bien avec les bibliothèques NumPy et SciPy, permettant ainsi la manipulation efficace de données numériques et scientifiques.

- ✓ **Facilité d'Utilisation** :

La bibliothèque est conçue de manière à être facile à utiliser pour les débutants tout en offrant suffisamment de flexibilité pour les utilisateurs avancés.

- ✓ **Documentation Complète** :

Scikit-learn dispose d'une documentation complète et bien organisée, avec des exemples détaillés qui aident les utilisateurs à comprendre et à mettre en œuvre les fonctionnalités de la bibliothèque.

4.5.2 Pandas

Pandas est une bibliothèque open source de manipulation et d'analyse de données en Python. Elle offre des structures de données performantes et faciles à utiliser, ainsi que des outils pour lire et écrire des données dans divers formats. Pandas est largement utilisé dans le domaine de l'analyse de données, du traitement de données et de la préparation de données en raison de sa flexibilité et de ses fonctionnalités avancées. Pandas prend en charge la lecture et l'écriture de données à partir de nombreux formats, tels que CSV, Excel, SQL, JSON, HTML, et bien d'autres. Pandas propose des méthodes puissantes pour indexer, sélectionner et filtrer des données dans un DataFrame. Pandas offre des outils pour nettoyer et transformer des données, notamment la gestion des valeurs manquantes, le remplacement de valeurs, le regroupement, le tri, et la fusion de DataFrames.

4.5.3 Matplotlib

Matplotlib est la bibliothèque qui permet de visualiser nos Datasets, nos fonctions, nos résultats sous forme de graphes, courbes et nuages de points [19].

4.5.4 Seaborn

Seaborn est une bibliothèque permettant de créer des graphiques statistiques en Python. Elle est basée sur Matplotlib, et s'intègre avec les structures Pandas [20].

4.6 Conclusions

Dans ce chapitre, nous avons établi les bases de notre système de détection des attaques DDoS par apprentissage automatique. Nous avons exploré les étapes clés du machine Learning, analysé plusieurs jeux de données et sélectionné celui le plus adapté à notre projet. Après comparaison des infrastructures cloud, nous avons choisi Google Colab pour son accessibilité et sa compatibilité avec Python. Ces choix nous permettront d'expérimenter et d'optimiser notre modèle dans les prochaines étapes.

Chapitre 5 Mise en œuvre, validation et tests

5.1 Architecture proposé

L'architecture proposée pour la phase d'apprentissage et d'évaluation et d'intégration se compose de plusieurs éléments clés comme le décline la figure

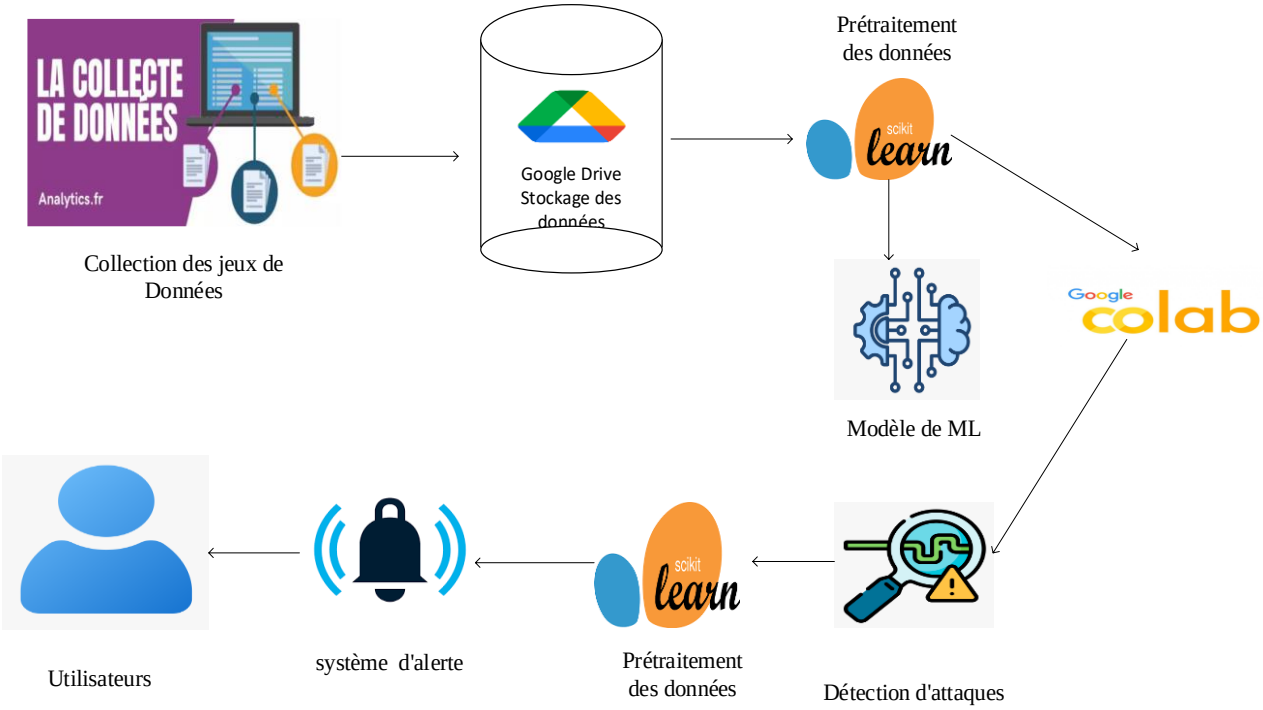


Figure 5.1 – Architecture proposée

➢ Phase d'apprentissage

Cette architecture illustre un **processus complet** de détection des attaques en utilisant le Machine Learning (ML).

Collection des Jeux de Données

- ❖ Les données sont collectées à partir de différentes sources (trafic réseau, logs de serveurs, fichiers de données).
- ❖ Ces jeux de données peuvent contenir des exemples d'attaques et de trafic normal.

❖ L’objectif est d’alimenter un modèle d’apprentissage automatique avec des données représentatives.

Stockage des Données dans Google Drive

❖ Une fois collectées, les données sont stockées sur **Google Drive**, permettant ainsi un accès facile via Google Colab.

❖ Ce stockage permet aussi de conserver et de partager les données pour les traitements futurs.

Prétraitement des Données avec Scikit-learn

❖ Avant d’entraîner un modèle de ML, les données doivent être nettoyées et transformées.

❖ Le prétraitement inclut :

- La **suppression des valeurs manquantes**
- La **normalisation ou standardisation** des valeurs
- L’**encodage des données catégorielles**
- La **réduction de la dimensionnalité** si nécessaire

❖ **Scikit-learn** est utilisé pour effectuer ces transformations.

Entraînement du Modèle de Machine Learning sur Google Colab

❖ Une fois les données prétraitées, elles sont envoyées vers Google Colab, qui permet d’entraîner et tester des modèles de Machine Learning.

❖ Les algorithmes ML couramment utilisés pour la détection d’attaques incluent :

- **Random Forest**
- **Regression Logistic**
- **Decision Tree**
- **Naive Bayes**

❖ L’objectif est de créer un modèle capable de **différencier le trafic normal des attaques**.

Détection des Attaques

- ❖ Une fois entraîné, le modèle est utilisé pour analyser **en temps réel** les nouveaux flux de trafic réseau.
- ❖ Il classe chaque événement en **normal** ou **malveillant**.
- ❖ S'il détecte une anomalie, il génère une alerte.

Système d'Alerte

- ❖ Lorsqu'une attaque est détectée, un **système d'alerte** est déclenché.
- ❖ Cela peut être sous forme :

- D'un **email ou notification**

Utilisateurs

- ❖ Les utilisateurs (administrateurs réseau, analystes en cybersécurité) sont informés via le système d'alerte.

Ils peuvent ainsi analyser la menace et prendre des mesures pour la contrer (ex : bloquer l'adresse

- ❖ IP, renforcer la sécurité du réseau).

5.2 Choix du data set

Dans cette étude, l'ensemble de données utilisé a été obtenu à partir du CICDDoS2019. CIC-DDoS 2019 est la dernière data sets synthétiques de haute qualité. Cet ensemble de données est un projet conjoint de l'établissement de la communication et sécurité canadien (CSE) et l'institut canadien de cyber sécurité (CIC). Cette data set contient seulement des attaques ddos et bénin, elle a 84 variables. Elle a été collectée dans deux jours pour le training et l'évaluation des tests. L'ensemble d'entraînement a été capturé le 12 janvier 2019, et contient 12 types d'attaques DDoS différents, chaque type d'attaque séparé dans un fichier PCAP. Les types d'attaque dans la journée de training comprend UDP, SNMP, NetBIOS, LDAP, TFTP, NTP, SYN, WebDDoS, MSSQL, UDP-Lag, DNS et SSDP DDoS based attack. Les données de test ont été créées le 11 mars 2019 et contiennent 7 types d'attaque DDoS SYN, MSSQL, UDP-Lag, LDAP, UDP et NetBIOS. Ces captures est représentent dans la figure suivante [21].

Jours	Attaques	Heure d'attaque
Premier jour	Plan du port	9:43 - 9:51
	NetBIOS	10h00 - 10h09
	LDAP	10h21 - 10h30
	MSSQL	10:33 - 10:42
	Protocole UDP	10:53 - 11:03
	Décalage UDP	11:14 - 11:24
	SYN	11:28 - 17:35
Deuxième jour	NTP	10h35 - 10h45
	DNS	10:52 - 11:05
	LDAP	11:22 - 11:32
	MSSQL	11h36 - 11h45
	NetBIOS	11h50 - 12h00
	SNMP	12:12 - 12:23
	Programme SSDP	12:27 - 12:37
	Protocole UDP	12h45 - 13h09
	Décalage UDP	13:11 - 13:15
	WebDDoS	13:18 - 13:29
	SYN	13:29 - 13:34
	TFTP	13h35 - 17h15

Figure 5.2 – les types d'attaques ddos dans dataset 2019

5.2.1 La réduction des données

Le volume de l'ensemble de données nous a contraints à réduire significativement le nombre d'échantillons issus des ensembles d'apprentissage et de test. L'ensemble de données initial est composé de 11 fichiers CSV, totalisant plus de 22 Go pour l'apprentissage, ainsi que 7 autres fichiers CSV de plus de 8 Go pour le test. La réduction des données d'apprentissage et de test a été effectuée séparément, en suivant la même procédure. La lecture et la préparation de ces fichiers volumineux n'étaient pas des tâches aisées, même en utilisant Google Colab, qui offre une capacité de 12 Go de RAM.

- Analyser et traiter chaque fichier CSVs séparément, afin de charger les données pour certains fichiers de grande taille (plus 8 Go).

5.3 Etape de l'implémentation

Voici les étapes que nous avons suivie pour faire l'implémentions de notre projet

5.3.1 L'analyse exploratoire des données

C'est la première étape de la définition de la stratégie de modélisation par la compréhension des variables.

- **Téléchargement des bibliothèques**

```
[ ] # Importer les bibliothèques
    %matplotlib inline
    import pandas as pd
    import numpy as np
    import matplotlib.pyplot as plt
    import seaborn as sns
    import zipfile
    import os
```

Figure 5.3 – Téléchargement des bibliothèques

- **On importe les fichiers dans google colab**



Figure 5.4 – Importation des fichiers .csv

➢ Analyse de la forme

Cette capture d'écran montre les dimensions du jeu de données, avec un total de plus de 5 millions d'enregistrements et 88 variables. Chaque enregistrement représente une observation ou un événement dans l'ensemble de données, tandis que chaque variable correspond à un attribut ou une caractéristique spécifique de ces événements. Cela signifie qu'il y a un grand nombre de données à analyser, avec de nombreuses informations disponibles pour chaque observation. Ces dimensions sont importantes pour évaluer la taille du jeu de données et déterminer les ressources nécessaires pour le traitement ou l'analyse.

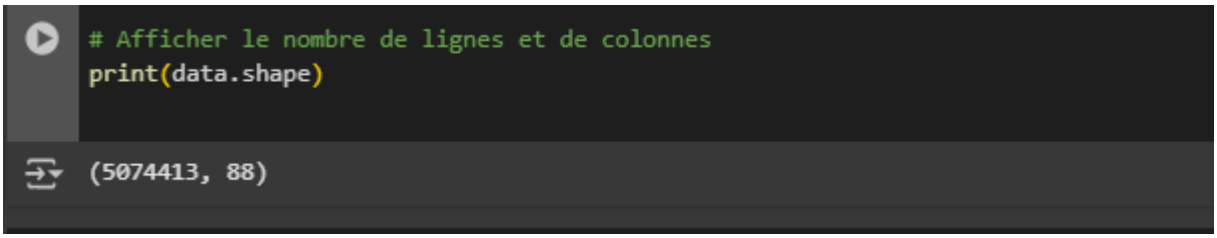


Figure 5.5 – Le nombre de ligne et de colonne

➢ Analyse du Fond

Cette capture montre la répartition des échantillons dans le jeu de données, où chaque échantillon est étiqueté selon qu'il représente une attaque ou du trafic normal.

- **DDoS** : Il s'agit d'une catégorie représentant des attaques par déni de service distribué. Dans ce cas, la grande majorité des échantillons (5 071 011) appartiennent à cette catégorie, ce qui indique que l'attaque est le cas le plus courant dans le jeu de données.

- **Bénin** : Cette catégorie représente le trafic normal ou non malveillant. Ici, seulement 3 402 échantillons sont classés comme bénins, ce qui montre que ces données sont très peu nombreuses par rapport aux données d'attaque.

```
Label
DDoS     5071011
Bénin       3402
Name: count, dtype: int64
```

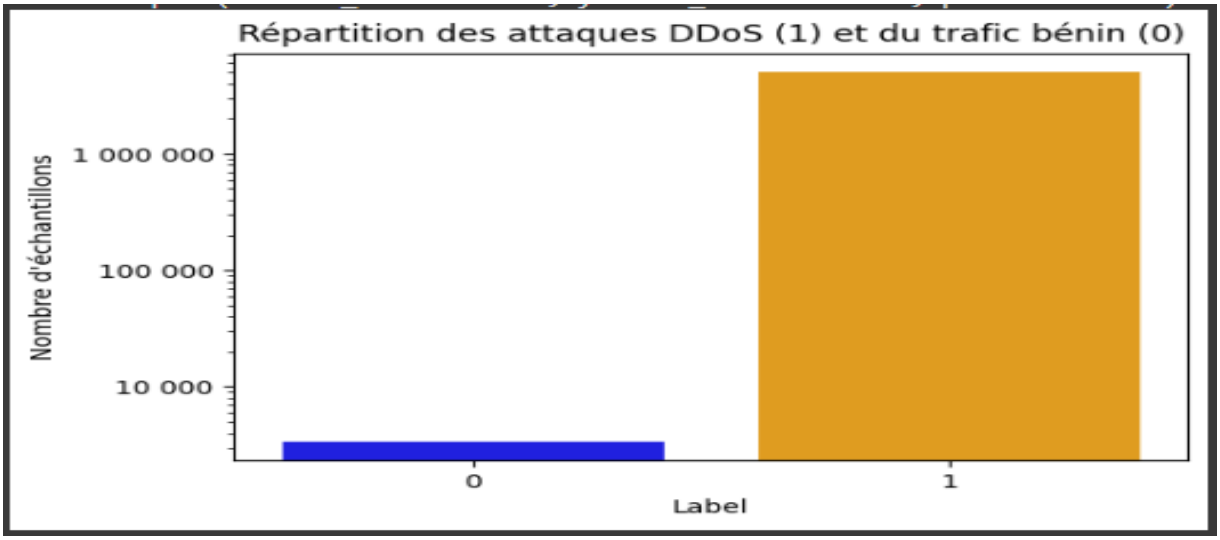


Figure 5.6 – visualisation de la target

➢ Sélection des colonnes

Cette capture d'écran extrait un sous-ensemble de données à partir d'un DataFrame data en sélectionnant uniquement les colonnes pertinentes pour l'analyse du trafic réseau et la détection d'attaques. Il filtre notamment des métriques comme le débit, le nombre de paquets envoyés/reçus, la durée des flux, les ports source et destination, ainsi que des indicateurs liés aux attaques (SYN Flag Count, ACK Flag Count). L'objectif est de simplifier l'analyse et de préparer les données pour une éventuelle modélisation en machine learning.

```
# Sélectionner les colonnes d'intérêt
columns_of_interest = columns_of_interest = [
    'Flow Bytes/s', 'Flow Packets/s', 'Total Fwd Packets',
    'Total Backward Packets', 'Protocol', 'Source Port',
    'Destination Port', 'Flow Duration', 'Fwd Packet Length Mean',
    'Fwd IAT Mean', 'Bwd IAT Mean', 'SYN Flag Count',
    'ACK Flag Count', 'Label'
]
```

Figure 5.7 – Sélection des colonnes

- **L'affichage du Datafram**

Cette capture d'écran présente un DataFrame Pandas contenant des données de trafic réseau. Chaque ligne correspond à une connexion réseau, décrite par plusieurs caractéristiques telles que le débit des flux (Flow Bytes/s, Flow Packets/s), le nombre total de paquets envoyés et reçus, le protocole utilisé, les ports source et destination, ainsi que des indicateurs comme les flags SYN et ACK. La dernière colonne, "Label", identifie la nature du trafic, indiquant si la connexion est bénigne ou associée à une attaque.

[22]

	Flow Bytes/s	Flow Packets/s	Total Fwd Packets	Total Backward Packets	Protocol	Source Port	Destination Port	Flow Duration	Fwd Packet Length Mean	Fwd IAT Mean	Bwd IAT Mean	SYN Flag Count	ACK Flag Count	Label
0	1.502024e+06	3.413690e+03	97	0	17	634	60495	28415	440.0	295.989583	0.0	0	0	DrDoS_DNS
1	4.400000e+08	1.000000e+06	2	0	17	634	60495	2	440.0	2.000000	0.0	0	0	DrDoS_DNS
2	1.812602e+06	4.119549e+03	200	0	17	634	46391	48549	440.0	243.964824	0.0	0	0	DrDoS_DNS
3	1.820552e+06	4.137617e+03	200	0	17	634	11894	48337	440.0	242.899497	0.0	0	0	DrDoS_DNS
4	2.747767e+06	6.244926e+03	200	0	17	634	27878	32026	440.0	160.934673	0.0	0	0	DrDoS_DNS
...	...	...	...	...	...	...	...	...	...	...	...	...	...	...
5074408	2.944000e+09	2.000000e+06	2	0	17	900	23579	1	1472.0	1.000000	0.0	0	0	DrDoS_DNS
5074409	9.813333e+07	6.666667e+04	2	0	17	900	54596	30	1472.0	30.000000	0.0	0	0	DrDoS_DNS
5074410	2.944000e+09	2.000000e+06	2	0	17	900	14341	1	1472.0	1.000000	0.0	0	0	DrDoS_DNS
5074411	2.944000e+09	2.000000e+06	2	0	17	900	46229	1	1472.0	1.000000	0.0	0	0	DrDoS_DNS
5074412	1.472000e+09	1.000000e+06	2	0	17	900	23895	2	1472.0	2.000000	0.0	0	0	DrDoS_DNS

Figure 5.8 – Datafram

5.3.2 Prétraitement

Le prétraitement c'est l'étape qui consiste a préparé nos données avant de les fournir à la machine pour son apprentissage, notre objectif et de :

Mètre les données dans un format propice au machine Learning

- **Encodage**

Sur la capture d'écran, nous pouvons voir que la colonne "Label" a été modifiée pour contenir des valeurs numériques à la place des noms textuels. Les labels 'DrDoS_DNS' et autres types d'attaques ont été convertis en **1**, tandis que 'BENIGN' a été remplacé par **0**, ce qui est essentiel pour préparer les données à des analyses ou des modèles d'apprentissage automatique, qui fonctionnent généralement mieux avec des données numériques plutôt qu'avec des chaînes de caractères.

```python
# Créer un dictionnaire de codes pour encoder la colonne "Label"
code = {
    'Syn': 1,
    'UDP-Ing': 1,
    'DrDos_UDP': 1,
    'WebDDos': 1,
    'UDP': 1,
    'UDPLag': 1,
    'MSSQL': 1,
    'DrDos_MSSQL': 1,
    'TFTP': 1,
    'DrDos_DNS': 1,
    'DrDos_SNMP': 1,
    'DrDos_SSDP': 1,
    'DrDos_LDAP': 1,
    'DrDos_NTP': 1,
    'DrDos_NetBIOS': 1,
    'BENIGN': 0
}

# Encoder les colonnes de type "object" en utilisant le dictionnaire de codes
for col in df.select_dtypes(include=['object']):
    df[col] = df[col].map(code)

print(df['Label'].value_counts())

df.head()
```

Figure 5.9 – Encodage

[26]

	Flow Bytes/s	Flow Packets/s	Total Fwd Packets	Total Backward Packets	Protocol	Source Port	Destination Port	Flow Duration	Fwd Packet Length Mean	Fwd IAT Mean	Bwd IAT Mean	SYN Flag Count	ACK Flag Count	Label
0	1.502024e+06	3.413690e+03	97	0	17	634	60495	28415	440.0	295.989583	0.0	0	0	1
1	4.400000e+08	1.000000e+06	2	0	17	634	60495	2	440.0	2.000000	0.0	0	0	1
2	1.812602e+06	4.119549e+03	200	0	17	634	46391	48549	440.0	243.964824	0.0	0	0	1
3	1.820552e+06	4.137617e+03	200	0	17	634	11894	48337	440.0	242.899497	0.0	0	0	1
4	2.747767e+06	6.244926e+03	200	0	17	634	27878	32026	440.0	160.934673	0.0	0	0	1
...	...	...	...	...	...	...	...	...	...	...	...	...	...	...

[27]

27	12.484017	0.469053	26	26	6	59099	80	110861755	23.692308	4.427287e+06	4.427289e+06	0	0	0
31	1.735465	0.471055	9	10	6	59102	443	40335006	0.888889	5.041876e+06	4.471610e+06	0	0	0
53	0.000000	0.494505	56	0	0	0	0	113244633	0.000000	2.058993e+06	0.000000e+00	0	0	0
93	0.209142	0.428740	21	20	6	59155	80	95628949	0.952381	4.781447e+06	4.750622e+06	0	0	0
94	0.209176	0.428811	21	20	6	59131	80	95613243	0.952381	4.780662e+06	4.751452e+06	0	0	0
...	...	...	...	...	...	...	...	...	...	...	...	...	...	...
5073059	0.000000	0.630591	12	0	0	0	0	19029778	0.000000	1.729980e+06	0.000000e+00	0	0	0
5073060	0.000000	11.679582	1	1	6	57177	80	171239	0.000000	0.000000e+00	0.000000e+00	0	0	0
5073061	0.000000	13574.660633	1	2	6	80	57177	221	0.000000	0.000000e+00	2.000000e+00	0	0	0
5073063	2955.680851	96.380897	4	2	6	57225	443	62253	23.000000	2.075100e+04	3.000000e+00	0	0	0
5073064	0.000000	0.472213	9	0	0	0	0	19059205	0.000000	2.382401e+06	0.000000e+00	0	0	0

Figure 5.10 – Affichage après encodage

➢ Nettoyage

Élimination des nan, inf : c'est-à-dire éliminé tous les valeurs manquantes et tout valeurs infinies en peut voir ça dans la figure suivante :

```
[ ] from sklearn.model_selection import train_test_split
    import numpy as np
    # Replace infinities with NaN in both X_train and X_test
    X_train.replace([np.inf, -np.inf], np.nan, inplace=True)
    X_test.replace([np.inf, -np.inf], np.nan, inplace=True)
```

Figure 5.11 – Nettoyage

➢ Division ou répartition des données en 80% pour l'entrainement et 20% pour le test

Sur cette capture, on effectue la séparation des données en features **(X)** et labels **(y)** en excluant la colonne "Label" des caractéristiques. Ensuite, les données sont divisées en un ensemble d'entraînement (**80%**) et un ensemble de test (**20%**) à l'aide de la fonction train_test_split de scikit-learn, avec un random_state fixer à 42 pour assurer la reproductibilité des résultats.

```
[ ] from sklearn.model_selection import train_test_split
    # Séparation des features et labels
    X = df_filtered.drop(columns=["Label"])
    y = df_filtered["Label"]

    # Séparation des données en train / test
    X_train, X_test, y_train, y_test = train_test_split(X, y, test_size=0.2, random_state=42)
```

Figure 5.12 – Répartition des données

Cette capture montre les dimensions des ensembles de données après nettoyage. L'ensemble d'entraînement (X_train) contient 4 059 530 exemples avec 13 caractéristiques, tandis que l'ensemble de test (X_test) comprend 1 014 883 exemples également répartis sur 13 caractéristiques. Ces ensembles seront utilisés respectivement pour entraîner et tester un modèle de machine learning afin de prédire les étiquettes (par exemple, détection d'attaques DDoS). Le nettoyage des données a permis de traiter les valeurs manquantes et infinies, rendant ainsi les ensembles prêts à être utilisés pour l'apprentissage et l'évaluation.

```
[25] X_train.shape
     (4059530, 13)

[26] X_test.shape
     (1014883, 13)
```

Figure 5.13 – Le nombres d'attributs

5.3.3 Modélisation

Nous allons enfin pouvoir appliquer nos algorithmes de classification, pour chaque algorithme nous allons montrer les différentes fonctions utilisées :

```
[ ] from sklearn.linear_model import LogisticRegression
    from sklearn.tree import DecisionTreeClassifier
    from sklearn.metrics import accuracy_score
    from sklearn.naive_bayes import GaussianNB
    from sklearn.ensemble import RandomForestClassifier, AdaBoostClassifier
```

Figure 5.14 – importation des algorithmes

Nous avons utilisé quatre modèles de classification :

L'arbre de décision, la régression logistique, le classificateur de forêt aléatoire, gaussienne NB.

La figure 15 représente les estimateurs Les algorithmes que nous avons utilisés.

```
[ ] # Decision Tree Classifier
    DC = DecisionTreeClassifier(random_state=0, splitter="random", max_depth=12, criterion='entropy')

    # Random Forest Classifier
    RF = RandomForestClassifier(n_estimators=100, random_state=0, max_depth=5)

    # Logistic Regression
    lr = LogisticRegression(solver='liblinear', max_iter=5)

    # Gaussian Naive Bayes
    nb = GaussianNB(var_smoothing=0.001)
```

Figure 5.15 – Déclaration des algorithmes

5.4 Résultat des Algorithmes de machine Learning

Nous présenterons les résultats de chaque algorithme que nous avons obtenus. Enfin Nous faisions une comparaison des résultats.

5.4.1 Application de l'algorithme « Random Forest »

- **Résultat du test :**

```
Accuracy: 0.9999911319827015

Classification Report:
              precision    recall  f1-score   support

           0       0.99      1.00      0.99       673
           1       1.00      1.00      1.00   1014210

    accuracy                           1.00   1014883
   macro avg       1.00      1.00      1.00   1014883
weighted avg       1.00      1.00      1.00   1014883
```

Figure 5.16 – Résultat test « Random Forest »

- **Discussions des résultats :**

Précision (Precision)

- **Classe 0 (Bénin)** : Une précision de 0.99 signifie que 99 % des prédictions de trafic bénin étaient correctes, avec seulement 1 % de faux positifs. Cela montre que le modèle ne confond presque jamais un trafic bénin avec une attaque, ce qui limite les interruptions inutiles dans le réseau.
- **Classe 1 (Attaque)** : Une précision parfaite de 1.00 indique qu'à chaque fois que le modèle détecte une attaque, il s'agit bien d'une attaque réelle. Il n'y a donc aucune fausse alerte, ce qui est essentiel en cybersécurité pour éviter les fausses alarmes qui peuvent surcharger les équipes de sécurité.

Rappel (Recall)

- **Classe 0 (Bénin)** : Un rappel de 1.00 pour les échantillons bénins montre que tous les trafics légitimes ont été correctement identifiés. Aucun trafic bénin n'a été classé à tort comme une attaque, ce qui réduit le risque de blocages injustifiés.
- **Classe 1 (Attaque)** : Un rappel parfait de 1.00 signifie que le modèle a détecté 100 % des attaques**.** Cela garantit qu'aucune menace n'est ignorée, ce qui améliore la sécurité du système en permettant une réponse rapide aux incidents.

F1-Score :

- L'équilibre parfait entre précision et rappel pour les deux classes reflète une excellente performance globale. Pour la classe 1 (Attaque)**,** cela montre que le modèle est non seulement précis dans ses détections, mais également exhaustif. Pour la classe 0 (Bénin)**,** cela signifie que le modèle réussit à maintenir un faible taux de faux positifs tout en capturant tous les trafics légitimes.

Macro Average (macro avg)

- **Définition :** Moyenne arithmétique des métriques (précision, rappel, F1-score) calculée indépendamment pour chaque classe, puis moyennée sans tenir compte de la proportion des échantillons dans chaque classe.
    - Précision pour la classe 0 : 0.99
    - Précision pour la classe 1 : 1.00
    - **Macro Précision = (0.99 + 1.00) / 2 = 0.995**

Weighted Average (weighted avg) :

- **Définition :** Moyenne pondérée des métriques pour chaque classe, où chaque score est pondéré par le nombre d'échantillons réels de cette classe.
    - Classe 0 : 673 échantillons
    - Classe 1 : 1,014,210 échantillons
    - **Weighted Précision = (0.99 × 673 + 1.00 × 1,014,210) / (673 + 1,014,210) ≈ 1.00**

➢ Matrice de Confusion

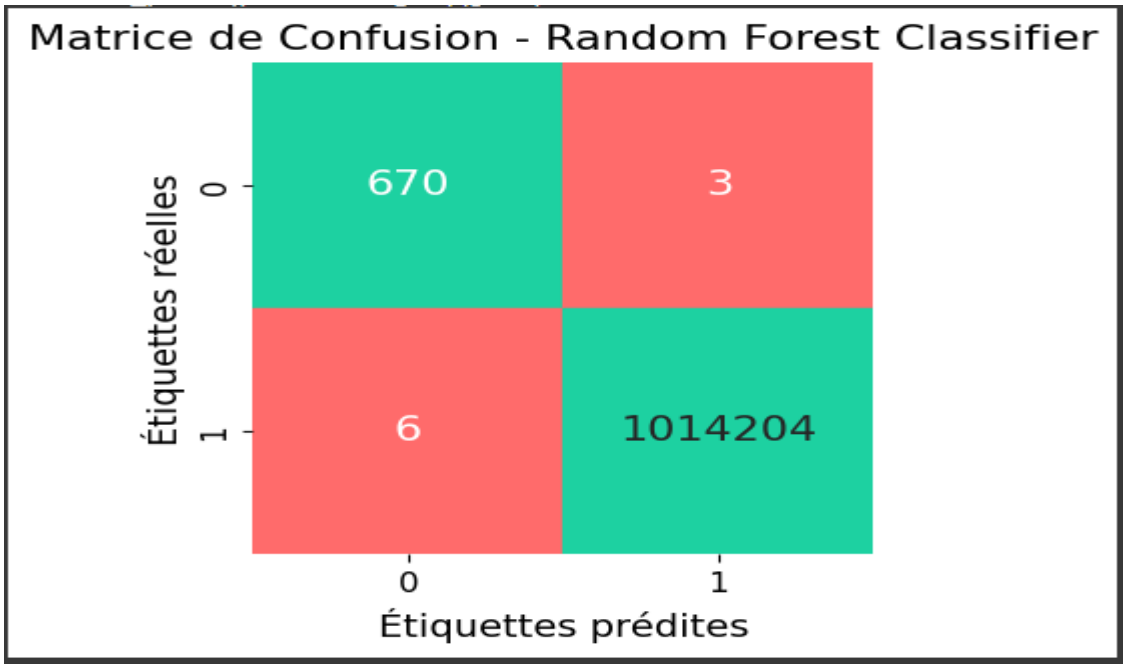


Figure 5.17 – Matrice de Confusion « Random Forest »

- **Vrais négatifs (TN) :** 670 (trafic bénin bien détecté)
- **Faux positifs (FP) :** 3 (trafic bénin mal classé comme attaque)
- **Faux négatifs (FN) :** 6 (attaques non détectées)
- **Vrais positifs (TP) :** 1 014 204 (attaques bien détectées)

Ces résultats montrent que le modèle **Random Forest** offre une détection fiable et efficace pour les attaques DDoS, tout en minimisant les impacts sur le trafic bénin. Cette combinaison est essentielle pour garantir une protection robuste tout en maintenant une expérience utilisateur fluide pour le trafic légitime.

5.4.2 Application de l'algorithme « Decision Tree »

➢ **Résultat du test :**

```
Accuracy: 0.9999822639654029

Classification Report:
              precision    recall  f1-score   support

           0       0.98      0.99      0.99       673
           1       1.00      1.00      1.00   1014210

    accuracy                           1.00   1014883
   macro avg       0.99      1.00      0.99   1014883
weighted avg       1.00      1.00      1.00   1014883
```

Figure 5.18 – Résultat test « Arbre Décision »

- **Discussions des résultats :**

Précision (Precision) :

- **Classe 0 (Bénin) : 0.98**
  Parmi tous les échantillons prédits comme bénins, 98 % étaient effectivement bénins. Cependant, cela signifie qu'il y a eu 2 % de faux positifs, où du trafic normal a été classé par erreur comme une attaque.
- **Classe 1 (Attaque) : 1.00**
  La précision parfaite pour la détection des attaques indique qu'il n'y a aucune fausse alerte concernant les attaques, garantissant une grande fiabilité en cybersécurité.

Rappel (Recall) :

- **Classe 0 (Bénin) : 0.99**
  Le modèle a identifié correctement 99 % des échantillons bénins.
  Seulement 1 % des échantillons bénins ont été classés par erreur comme des attaques.
- **Classe 1 (Attaque) : 1.00**
  Tous les échantillons d'attaque ont été détectés, ce qui assure une couverture totale des menaces.

F1-Score :

- **Classe 0 (Bénin) : 0.99**
  Un bon équilibre entre précision (0.98) et rappel (0.99), ce qui signifie que le modèle gère bien le trafic bénin tout en minimisant les faux positifs.
- **Classe 1 (Attaque) : 1.00**
  F1-Score parfait, ce qui est essentiel pour une détection robuste des attaques.

Moyennes Macro et Pondérée :

- **Macro avg (0.99 précision, 1.00 rappel, 0.99 F1-score)**
  La moyenne non pondérée montre une bonne performance générale sur les deux classes, même si elle ne reflète pas totalement le déséquilibre des données.
- **Weighted avg (1.00 pour toutes les métriques)**
  Comme les échantillons d'attaque sont largement majoritaires, la moyenne pondérée montre que le modèle reste très performant même en prenant en compte la distribution réelle des données.

➢ Matrice de Confusion

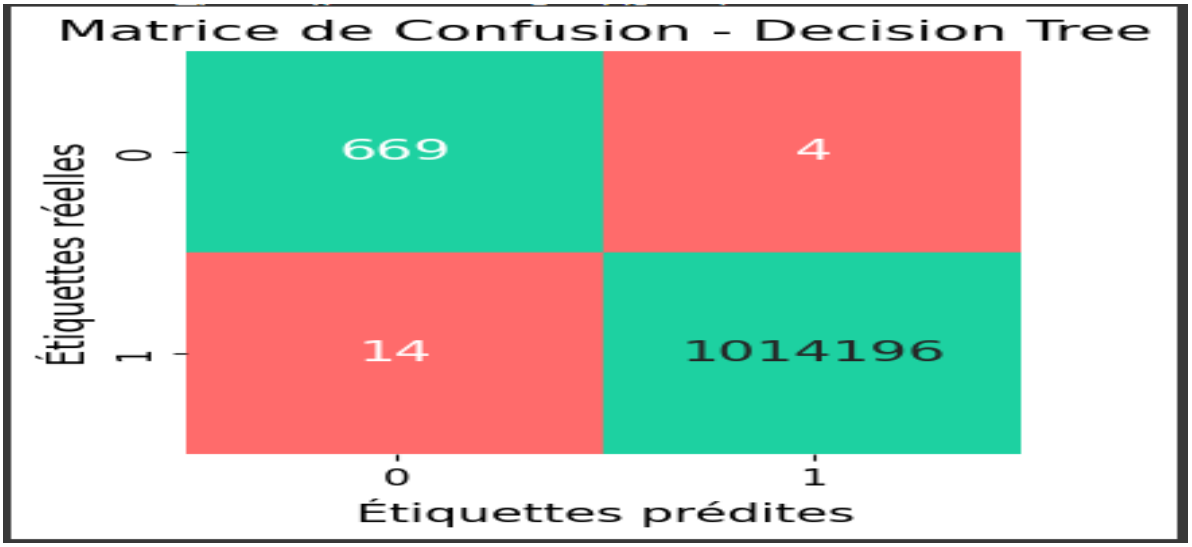


Figure 5.19 – Matrice de Confusion « Decision Tree »

- **Vrais positifs (TP) :** 1 014 196 (attaques bien détectées)
- **Faux positifs (FP) :** 4 (trafic bénin mal classé comme attaque)
- **Faux négatifs (FN) :** 14 (attaques non détectées)
- **Vrais négatifs (TN) :** 669 (trafic bénin bien détecté)

5.4.3 Application de l'algorithme « Gaussienne NB »

➢ **Résultat du test :**

```
Accuracy: 0.9992944999571379

Classification Report:
              precision    recall  f1-score   support

           0       0.41      0.15      0.22       673
           1       1.00      1.00      1.00   1014210

    accuracy                           1.00   1014883
   macro avg       0.71      0.57      0.61   1014883
weighted avg       1.00      1.00      1.00   1014883
```

Figure 5.20 – Résultat test « Gaussienne NB »

➢ **Discussions des résultats :**

Précision (Precision) :

- **Classe 0 (Bénin) : 0.41**
  La précision est très faible, signifiant que lorsque le modèle prédit un échantillon comme bénin, il n'a que 41 % de chances d'avoir raison.
  En cybersécurité, cela se traduit par un nombre élevé de faux positifs, où du trafic normal pourrait être interprété à tort comme une attaque.
- **Classe 1 (Attaque) : 1.00**
  La précision parfaite indique que presque tous les échantillons prédits comme des attaques étaient effectivement des attaques, ce qui est positif pour la détection des menaces.

Rappel (Recall) :

- **Classe 0 (Bénin) : 0.15**
  Seulement 15 % des échantillons réellement bénins sont correctement détectés comme tels.
  Cela signifie que 85 % du trafic bénin est mal classé comme une attaque, ce qui est

problématique dans un contexte de production où cela pourrait entraîner des interruptions de service inutiles.

- **Classe 1 (Attaque) : 1.00**
  Le rappel parfait pour les attaques indique que le modèle ne laisse passer aucune menace non détectée.

F1-Score :

- **Classe 0 (Bénin) : 0.22**
  Le faible F1-Score montre un déséquilibre entre la précision et le rappel pour les échantillons bénins, suggérant une mauvaise performance pour cette classe.
- **Classe 1 (Attaque) : 1.00**
  Le F1-Score parfait montre que le modèle est très performant pour détecter les attaques, avec un équilibre idéal entre précision et rappel pour cette classe.

Moyennes Macro et Pondérée :

- **Macro avg (0.71 précision, 0.57 rappel, 0.61 F1-score)**
  La moyenne macro montre clairement le déséquilibre des performances entre les deux classes, indiquant que le modèle n'est pas équilibré dans sa classification.
- **Weighted avg (1.00 pour toutes les métriques)**
  Comme la majorité des données appartiennent à la classe 1 (attaques), la moyenne pondérée donne un score parfait, mais cela masque les lacunes sur la classe minoritaire (bénin).

➢ Matrice de Confusion

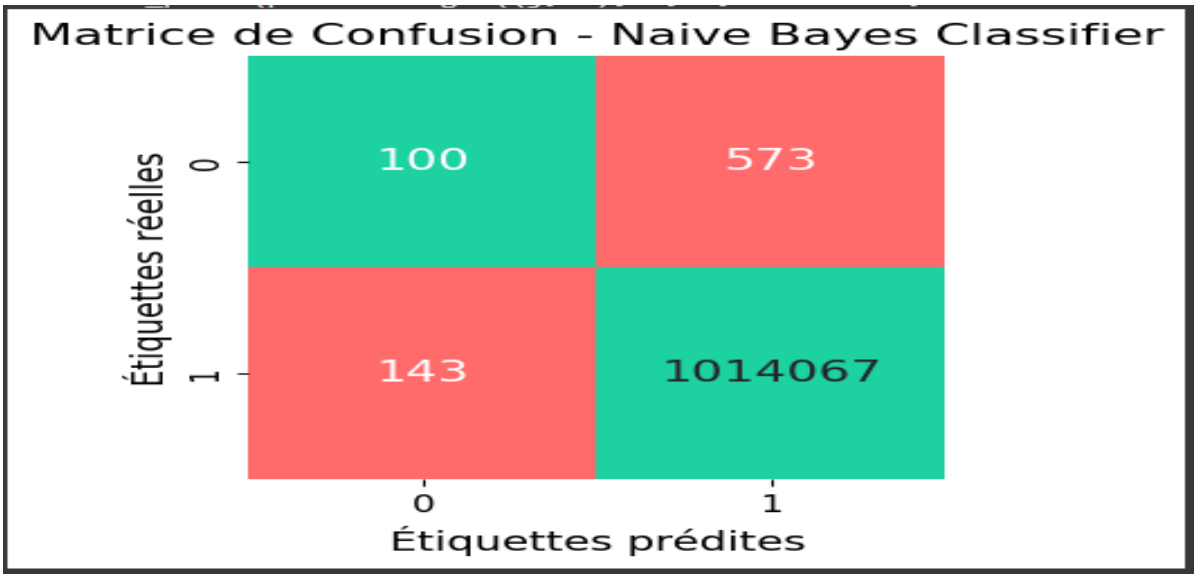


Figure 5.21 – Matrice de Confusion « Naïves Bayes »

- **Vrais positifs (TP) :** 1 014 067 (attaques bien détectées)
- **Faux positifs (FP) :** 573 (trafic bénin mal classé comme attaque)
- **Faux négatifs (FN) :** 143 (attaques non détectées)
- **Vrais négatifs (TN) :** 100 (trafic bénin bien détecté)

5.4.4 Application de l'algorithme « Regression Logistic »

➢ **Résultat du test :**

```
Accuracy: 0.9997112967701696

Classification Report:
              precision    recall  f1-score   support

           0       0.82      0.72      0.77       673
           1       1.00      1.00      1.00   1014210

    accuracy                           1.00   1014883
   macro avg       0.91      0.86      0.88   1014883
weighted avg       1.00      1.00      1.00   1014883
```

Figure 5.22 – Résultat test « Régression Logistic »

➢ **Discussions des résultats :**

Précision (Precision) :

- **Classe 0 (Bénin) : 0.82**
  La précision est relativement élevée, signifiant que lorsque le modèle prédit un échantillon comme bénin, il a 82 % de chances d'avoir raison**.**
  En cybersécurité, cela signifie que le modèle génère peu de faux positifs**,** ce qui est positif pour éviter des alertes inutiles.
- **Classe 1 (Attaque) : 1.00**
  La précision parfaite indique que tous les échantillons prédits comme attaques étaient effectivement des attaques**.**
  C'est un bon indicateur, car cela signifie que le modèle ne classe jamais du trafic bénin comme une attaque**.**

Rappel (Recall) :

- **Classe 0 (Bénin) : 0.72**
  72 % des échantillons réellement bénins sont correctement détectés comme tels.
  Cependant, 28 % du trafic bénin est mal classé comme une attaque, ce qui peut entraîner des perturbations dans un environnement de production avec des interruptions de service inutiles.
- **Classe 1 (Attaque) : 1.00**
  Le rappel parfait pour les attaques montre que le modèle détecte toutes les menaces sans exception**.**
  C'est un atout majeur en cybersécurité, où il est primordial de ne laisser passer aucune attaque**.**

F1-Score :

- **Classe 0 (Bénin) : 0.77**
  Bien que le score soit correct, il montre un déséquilibre entre la précision et le rappel pour les échantillons bénins.
  Cela indique que le modèle a encore des difficultés à bien classifier cette classe**.**

- **Classe 1 (Attaque) : 1.00**
  Le F1-Score parfait montre que le modèle est très performant pour détecter les attaques, avec un équilibre idéal entre précision et rappel pour cette classe.

Moyennes Macro et Pondérée :

- **Macro avg (0.91 précision, 0.86 rappel, 0.88 F1-score)**
  La moyenne macro montre que le modèle est globalement performant, mais qu'il existe une différence de performance entre les deux classes.
- **Weighted avg (1.00 pour toutes les métriques)**
  Comme la majorité des données appartiennent à la classe 1 (attaques), la moyenne pondérée donne un score parfait, ce qui masque les lacunes du modèle sur la classe minoritaire (bénin)**.**

➢ Matrice de Confusion :

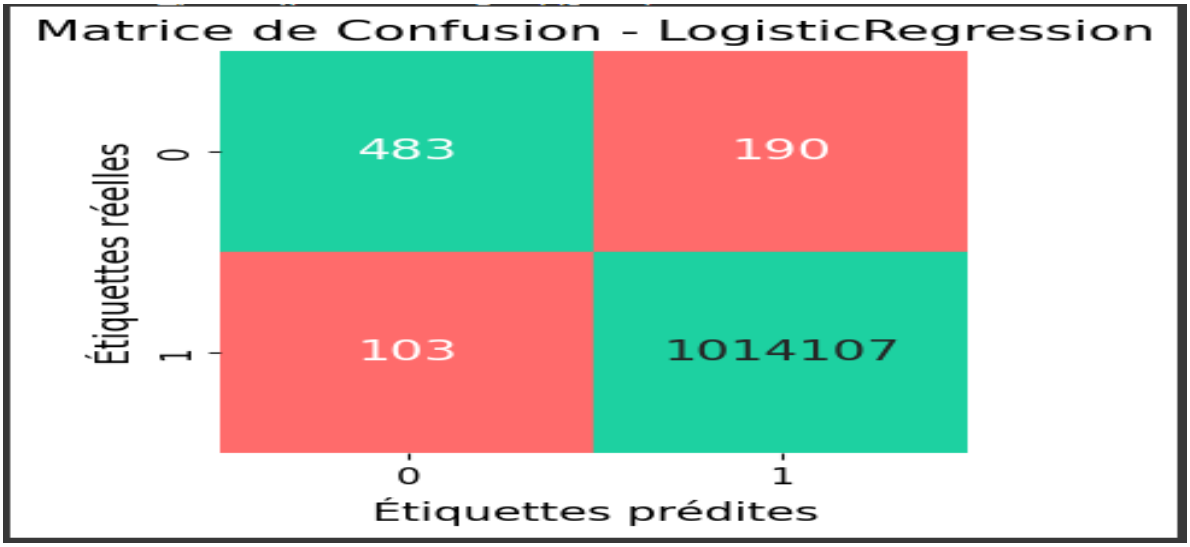


Figure 5.23 – Matrice de Confusion « Logistic Regression »

- **Vrais positifs (TP) :** 1 014 107 (attaques bien détectées)
- **Faux positifs (FP) :** 190 (trafic bénin mal classé comme attaque)
- **Faux négatifs (FN) :** 103 (attaques non détectées)
- **Vrais négatifs (TN) :** 483 (trafic bénin bien détecté)

➢ Résultats des différents modèles

Cette capture d'écran présente les performances de plusieurs modèles de classification entraînés pour la détection d'attaques. La colonne "Model_de_classification_Entrainé" liste les algorithmes utilisés : Random Forest, Logistic Regression, Decision Tree et Naive Bayes. La colonne "Score_de_Prediction_Obtenu" indique la précision obtenue pour chaque modèle après l'entraînement et l'évaluation. Les résultats montrent que Random Forest (0.999991) et Decision Tree (0.999982) obtiennent des scores très élevés, suggérant une excellente capacité de prédiction. Logistic Regression (0.999711) est légèrement en dessous, tandis que Naive Bayes (0.999294) obtient la précision la plus basse parmi les modèles testés. Ces scores proches de 1 indiquent une forte performance, mais il est essentiel d'analyser d'autres métriques comme la précision, le rappel et la matrice de confusion pour évaluer leur efficacité réelle, notamment face aux faux positifs et faux négatifs.

	Model_de_classification_Entrainé	Score_de_Prediction_Obtenu
0	Random Forest	0.999991
1	Logistic Regression	0.999711
2	Decision Tree	0.999982
3	Naive Bayes	0.999294

Figure 5.24 – résultats

5.5 Comparaison et conclusion sur les algorithmes

Table 5.1 – Comparaison entre les algorithmes

Algorithme	**Precision**		**Recall**		**F1-score**		**Accuracy**
	Attaque (1)	Benin (0)	Attaque (1)	Benin (0)	Attaque (1)	Benin (0)	
Random Forest (RF)	1.00	0.99	1.00	1.00	1.00	0.99	0.999991
Decision Tree (DT)	1.00	0.98	1.00	0.99	1.00	0.99	0.999711
Naives Baye (NB)	1.00	0.41	1.00	0.15	1.00	0.22	0.999982
Regresion Logistic (RL)	1.00	0.82	1.00	0.72	1.00	0.77	0.999294

- **Random Forest (RF) : Le meilleur choix pour détecter les attaques DDoS**

Random Forest est le modèle le plus performant parmi ceux testés. Il affiche une précision parfaite pour les attaques DDoS (1.00) et une excellente précision pour le trafic bénin (0.99). Son recall est de 1.00 pour toutes les classes, garantissant une détection complète des attaques DDoS et du trafic normal. Il minimise les faux positifs et faux négatifs, ce qui est idéal pour la détection des attaques DDoS. De plus, il évite l'overfitting grâce à l'agrégation de plusieurs arbres de décision, assurant une meilleure robustesse que le Decision Tree.

- **Decision Tree (DT) : Un bon modèle, mais avec des limites**

Decision Tree offre des performances très proches de Random Forest avec une précision de 1.00 pour les attaques DDoS et 0.98 pour le trafic bénin. Son recall est légèrement inférieur

(0.99 pour le trafic bénin), ce qui signifie qu'il pourrait ne pas détecter toutes les connexions normales aussi bien que RF. De plus, il est plus sujet à l'overfitting, ce qui peut rendre ses performances instables sur de nouvelles attaques DDoS aux caractéristiques différentes.

- **Régression Logistique (RL) : Un compromis acceptable mais moins fiable**

La régression logistique fonctionne bien pour la détection des attaques DDoS (1.00 en précision et recall), mais elle échoue partiellement dans la détection du trafic bénin avec un recall de 0.72 et une précision de 0.82. Cela signifie qu'elle génère plus de faux positifs, ce qui peut être problématique dans un environnement réel où une mauvaise classification du trafic bénin pourrait entraîner des perturbations.

- **Naïve Bayes (NB) : Le moins performant pour la détection des attaques DDoS**

Naïve Bayes est totalement inadapté à la détection des attaques DDoS. Bien qu'il détecte correctement les attaques (1.00 en précision et recall), il échoue complètement à identifier le trafic bénin, avec une précision de seulement 0.41 et un recall catastrophique de 0.15. Cela signifie qu'il génère énormément de faux positifs, rendant une solution basée sur NB inutilisable en pratique pour la détection des attaques DDoS.

5.6 Utilisation du modèle Random Forest

Pour cet algorithme après avoir effectué les étapes préliminaires sur le dataset notamment : l'importation des données, le prétraitement, l'encodage, notre algorithme de Random Forest va découper les données reçues en plusieurs arbres de décision indépendants lors de la phase d'entraînement :

- **Construction des Arbres de Décision**
  - ✓ **Échantillonnage Bootstrap :** Le modèle de forêt aléatoire commence par créer plusieurs sous-ensembles de données d'entraînement à partir de l'ensemble de données original. Chaque sous-ensemble est obtenu en échantillonnant avec remplacement, ce qui signifie que certains échantillons peuvent être sélectionnés plusieurs fois, tandis que d'autres peuvent être omis.
  - ✓ **Construction des Arbres :** Pour chaque sous-ensemble, un arbre de décision est construit. Cependant, contrairement aux arbres de décision traditionnelle,

à chaque nœud de l'arbre, une sélection aléatoire d'un sous-ensemble de caractéristiques est utilisée pour déterminer la meilleure scission. Cela introduit de la variance et réduit la corrélation entre les arbres, améliorant ainsi la performance globale du modèle.

- **Prédiction avec la Forêt Aléatoire**

  - ✓ **Passage des Données à Travers les Arbres :** Lorsqu'une nouvelle observation est donnée pour la prédiction, elle est passée à travers chaque arbre de la forêt. Chaque arbre produit une prédiction de classe pour cette observation.
  - ✓ **Agrégation des Prédictions :** Les prédictions de tous les arbres sont ensuite agrégées pour déterminer la classe finale. Pour les problèmes de classification, comme c'est le cas ici, la forêt aléatoire utilise un vote majoritaire pour déterminer la classe prédite. La classe qui reçoit le plus de votes parmi tous les arbres est choisie comme prédiction finale.

- **Sauvegarde du model**

Sur cette capture d'écran, nous avons mis en place un code permettant de sauvegarder le modèle de Random Forest que nous avons utilisé lors de l'entraînement. Pour ce faire, nous avons utilisé la bibliothèque joblib, qui permet de sérialiser et de sauvegarder le modèle dans un fichier avec l'extension. joblib. Ce fichier contient l'ensemble des informations nécessaires pour réutiliser le modèle ultérieurement, sans avoir besoin de le réentraîner. Nous avons ensuite choisi un chemin spécifique où ce fichier sera stocké, ce qui nous permettra de le charger facilement à l'avenir pour effectuer des prédictions sur de nouvelles données sans avoir à repasser par le processus d'entraînement.

```
import joblib

fille_path ='C:\joblibsauvegarde\model.joblib'
# Sauvegarder le modèle Random Forest dans un fichier .joblib
joblib.dump(RF, fille_path)
```

```
['C:\\joblibsauvegarde\\model.joblib']
```

Figure 5.25 – Sauvegarde du model « Random Forest »

➢ Chargement du model

Sur cette capture d'écran, nous effectuons le chargement du modèle **Random Forest** que nous avions précédemment sauvegardé dans un fichier avec l'extension. joblib. Grâce à la bibliothèque joblib, nous restaurons ce modèle sans avoir à le réentraîner, ce qui nous permet de l'utiliser directement pour analyser et détecter d'éventuelles attaques. Cette approche facilite l'utilisation du modèle sur de nouvelles données tout en optimisant le temps de traitement.

```
import joblib
import pandas as pd
import numpy as np
import smtplib
from email.mime.multipart import MIMEMultipart
from email.mime.text import MIMEText
from sklearn.preprocessing import LabelEncoder

# 📌 Charger le modèle et les encodeurs
model = joblib.load("random_forest_ddos_detector.joblib")
```

Figure 5.26 – Chargement du model « Random Forest »

➢ Prédiction

Sur cette capture d'écran, nous avons introduit de nouvelles données afin de tester la capacité de notre modèle à détecter une attaque DDoS. Le modèle analyse ces données et effectue une prédiction pour déterminer si elles correspondent à un trafic malveillant (attaque DDoS) ou à un état normal (BENIGN). En fonction du résultat, il affiche la classe prédite, ce qui nous permet d'évaluer son efficacité dans la détection des attaques.

```python
# 📌 Exemple de nouvelles données
new_data = pd.DataFrame([{
    "Flow Bytes/s": 500.0,
        "Flow Packets/s": 5.0,
        "Total Fwd Packets": 5,
        "Total Backward Packets": 2,
        "Protocol": 1,  # ICMP
        "Source Port": 0,
        "Destination Port": 0,
        "Flow Duration": 50000,
        "Fwd Packet Length Mean": 50.0,
        "Fwd IAT Mean": 30000.0,
        "Bwd IAT Mean": 25000.0,
        "SYN Flag Count": 0,
        "ACK Flag Count": 0
}])

# 📌 Prédiction
prediction = model.predict(new_data)
classe_predite = "DDoS" if prediction[0] == 1 else "BENIGN"

print(f"Classe prédite : {classe_predite}")
```

Figure 5.27 – Prédiction

➢ Mail d'alerte

Sur cette capture d'écran, nous configurons un serveur **SMTP** afin de permettre l'envoi automatique d'e-mails d'alerte à un utilisateur en cas de détection d'une attaque DDoS. La fonction envoyer_alerte permet d'envoyer un e-mail contenant les informations d'alerte dès qu'une attaque est prédite. Pour cela, nous utilisons **SMTP** avec le serveur Gmail (smtp.gmail.com) et une connexion sécurisée via **STARTTLS**. L'authentification se fait avec un nom d'utilisateur et un mot de passe d'application pour garantir la sécurité de l'envoi des notifications.

```python
# 📌 Fonction pour envoyer un e-mail d'alerte
def envoyer_alerte(destinataire, sujet, message):
    try:
        msg = MIMEMultipart()
        msg["From"] = USERNAME
        msg["To"] = destinataire
        msg["Subject"] = sujet
        msg.attach(MIMEText(message, "plain"))

        server = smtplib.SMTP(SMTP_SERVER, SMTP_PORT)
        server.starttls()
        server.login(USERNAME, PASSWORD)
        server.sendmail(USERNAME, destinataire, msg.as_string())
        server.quit()

        print("📧 Alerte envoyée avec succès à")
    except Exception as e:
        print("❌ Erreur lors de l'envoi de l'e-mail :", str(e))
```

```python
# 📌 Configuration SMTP pour Gmail
SMTP_SERVER = "smtp.gmail.com"
SMTP_PORT = 587
USERNAME = "maty75mbengue@gmail.com"
PASSWORD = "unua igsb ghqq mziu"  # Utilise un mot de passe d'application pour éviter les problèmes de sécurité
destinataire = "cherif.diallo@ugb.edu.sn"
```

Figure 5.28 – Envoie de mail d'alerte

Sur cette capture d'écran, nous avons mis en place un mécanisme permettant d'envoyer automatiquement un e-mail d'alerte à l'utilisateur lorsqu'une attaque **DDoS** est détectée. Si le modèle prédit une attaque, un message de sécurité est généré et transmis à l'utilisateur via e-mail. Ce message contient des recommandations pour atténuer l'impact de l'attaque et solliciter une assistance si nécessaire.

```python
if classe_predite == "DDoS":
    sujet = "🚨 Alerte Sécurité : Attaque DDoS détectée !"
    message = f"""
    Cher(e) utilisateur,

Nous venons de détecter une attaque de type DDoS sur votre réseau.
🚨 Nous vous recommandons de prendre des mesures immédiatement pour éviter tout dommage potentiel. ⚡

🔧 Actions recommandées :
👉 Contactez votre fournisseur d'accès Internet pour qu'il mette en place les filtres nécessaires afin de limiter l'impact de cette attaque. 💻

⚙️ Besoin d'aide ?
N'hésitez pas à nous contacter pour toute assistance. Notre équipe est disponible pour vous aider à résoudre cette situation rapidement. 📞

Protégez votre réseau dès maintenant ! 🔐

Cordialement,
Maty MBENGUE
Master RETEL
    """
    envoyer_alerte(destinataire, sujet, message)
```

```
Classe prédite : DDoS
📧 Alerte envoyée avec succès à
```

Figure 5.29 – Envoie de mail d'alerte

➢ Mail d’alerte reçu par l’utilisateur

Sur cette capture d’écran, on observe que l’utilisateur a bien reçu l’e-mail d’alerte après la détection d’une attaque **DDoS**. Une fois l’attaque identifiée par le modèle, un e-mail de notification a été automatiquement envoyé à l’utilisateur, confirmant ainsi la transmission de l’alerte de sécurité.

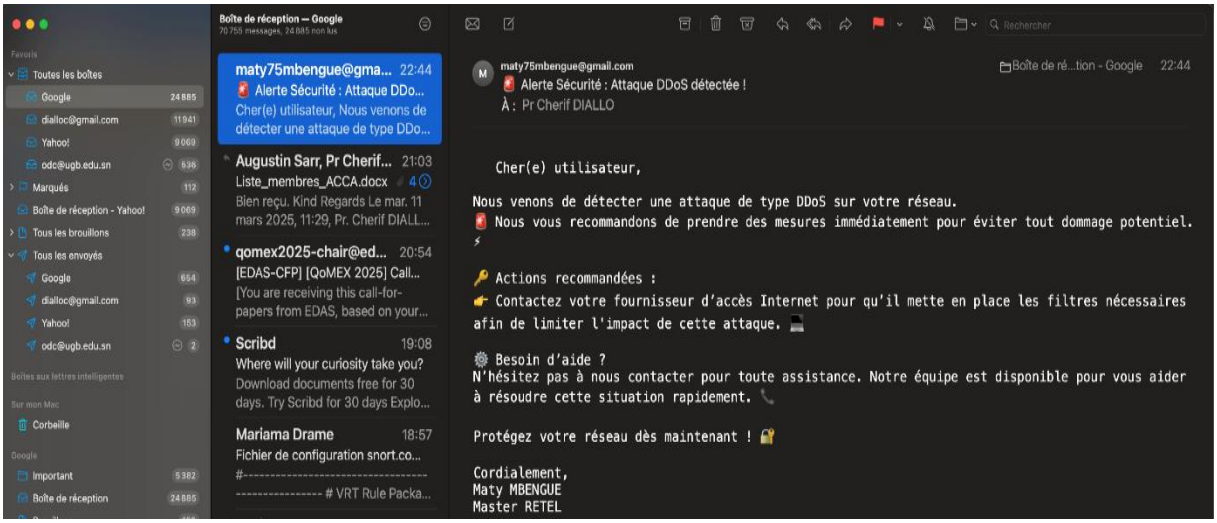


Figure 5.30 – mail reçu par l’utilisateur

5.7 Conclusion

Dans ce chapitre, nous avons exploré les différentes étapes de la mise en œuvre de notre approche pour la détection précoce des attaques DDoS visant les services internet. Nous avons détaillé les méthodes de prétraitement des données, les choix algorithmiques, ainsi que les critères d’évaluation de performance. L’expérimentation réalisée a permis de valider l’efficacité de notre modèle, mettant en évidence sa capacité à identifier rapidement et précisément les attaques DDoS tout en minimisant les faux positifs. Les résultats obtenus confirment la pertinence de notre approche, soulignant son efficacité dans la détection des menaces en temps réel. Cependant, certaines limites subsistent, notamment en ce qui concerne la généralisation du modèle face à des attaques DDoS de grande échelle ou des attaques à fort degré de sophistication. L’adaptation du modèle aux environnements réels, où les attaques peuvent varier et évoluer rapidement, reste un défi important. Ces observations ouvrent des perspectives d’amélioration, telles que l’intégration de mécanismes d’adaptation aux nouvelles variantes d'attaques DDoS et l’optimisation de la détection en temps réel pour mieux protéger les services internet contre ces menaces émergentes.

Conclusion générale et perspectives

La prolifération des attaques par déni de service distribué (DDoS) représente une menace croissante pour la disponibilité et la sécurité des services Internet, qui sont au cœur des infrastructures critiques telles que l'e-commerce, l'e-gouvernement et les plateformes éducatives. Face aux limites des approches traditionnelles de détection, qui reposent principalement sur des signatures et des règles statiques, l'adoption de solutions basées sur l'intelligence artificielle (IA) et l'apprentissage automatique (machine Learning) s'impose comme une nécessité pour faire face à l'évolution rapide des cybermenaces.

Dans ce travail, nous avons exploré diverses techniques de détection précoce des attaques DDoS en mettant en avant l'utilisation de modèles d'apprentissage automatique tels que la régression logistique, les forêts aléatoires, les arbres de décision et Naïve Bayes. Ces approches permettent une analyse dynamique et en temps réel des flux de trafic, afin de repérer les anomalies caractéristiques d'une attaque DDoS en cours, ce qui améliore ainsi la réactivité et la résilience des systèmes de défense.

Les résultats obtenus sur des jeux de données de référence tels que CIC-DDoS2019 ont montré que ces techniques peuvent significativement améliorer la détection précoce des attaques par rapport aux méthodes classiques basées sur des signatures. L'utilisation de ces modèles permet de réduire les faux positifs et d'augmenter l'efficacité des systèmes de détection.

Cependant, plusieurs défis demeurent. L'évolution rapide des techniques d'attaque DDoS, la gestion de volumes massifs de données en temps réel et l'optimisation des performances des modèles sont des aspects nécessitant encore des améliorations. En outre, la mise à jour régulière des modèles, pour tenir compte des nouvelles stratégies d'attaque, est indispensable pour maintenir leur efficacité. À l'avenir, l'intégration de techniques d'apprentissage profond (deep Learning), combinée à l'utilisation de systèmes distribués et de solutions basées sur l'Edge computing, pourrait offrir des solutions encore plus performantes en matière de détection et de prévention des attaques DDoS.

En conclusion, ce mémoire apporte une contribution significative à la compréhension et à la mise en œuvre de solutions de détection précoce des attaques DDoS. Il souligne le potentiel des approches basées sur l'intelligence artificielle pour anticiper et atténuer les menaces pesant sur les infrastructures critiques d'Internet. Les recherches futures dans ce domaine permettront de

continuer à améliorer la cybersécurité des services en ligne et de renforcer la résilience des systèmes face à l'évolution constante des cybermenaces.

Références

[1] Signification d'une attaque DDoS

https://www.futura-sciences.com/tech/definitions/internet-deni-service-2433/

[2] Destruction ou modification des informations de configuration

http ://www.cert.org/historical/advisories/CA-1996-01.cfm. Accessed on Sept 2014

[3] Attaques basées sur le volume

https ://softwarelab.org/fr/attaque-ddos/

[4] L'attaque par amplification DNS

https ://www.cloudflare.com/fr-fr/learning/ddos/what-is-a-ddos-attack/

[5] Les botnets

Curt Wilson, Arbor Networks ASERT, « Attack of the Shuriken: Many Hands, Many Weapons »., fév. 2012.

[6] Définition de L'intelligence artificielle

https ://experiences.microsoft.fr/business/intelligenceartificielle-iabusiness/comprendre-utiliser-intelligence-artificielle/

[7] Définition de Machine Learning

https ://www.lebigdata.fr/machine-learning-et-big-data

[8] Apprentissage supervisé

https ://larevuedesmedias.ina.fr/linformation-face-lintelligence-artificiellepromesses-et-dangers

[9] Apprentissage supervisé

A. Géron, « Machine Learning avec Scikit-Learn », Dunod: Paris, 2017.
[10] Apprentissage supervisé

M. Taffar, « Initiation à l'apprentissage automatique », Cours Master, Ment InformatiqueFaculté des Sciences Exactes et de l'Informatique.

[11] Classification naïve de Bayes

https://www.geeksforgeeks.org/naive-bayes-classifiers/

[12] Indépendance Conditionnelle

https://chatgpt.com/

[13] Machine Learning and Deep Learning
Raschka, S., & Mirjalili, V. (2019). Python Machine Learning: Machine Learning and Deep Learning with Python, scikit-learn, and TensorFlow 2. Packt Publishing

[14] Les arbres de décision
https://cedric.cnam.fr/vertigo/cours/ml2/coursArbresDecision.html.

[15] Random Forest
https://medium.com/@denizgunay/random-forest-af5bde5d7e1e

[16] Random Forest
https://medium.com/@dishantkharkar9/about-random-forest-algorithms-62163357db25

[17] Random Forest
https://medium.com/@mrmaster907/introduction-random-forest-classification-by-example-6983d95c7b91

[18] Définition Précision
https://www.analyticsvidhya.com/blog/2020/09/precision-recall-machinelearning/

[19] Définition Matplotlib
Apprendre le ML en une semaine.pdf

[20] Définition Seaborn
https://datascientest.com/seaborn

[21] Définition et Choix du data set CIC-DDOS-2019
Le modèle avec le NSL-KDD'99 à cinq classes - Hamouda Djallel

[22] Le lien pour la pratique sur google colab

https://colab.research.google.com/drive/1OlePdiq8TZhIMyabIVXooUYEu79eQYkq#scrollTo=M6nWzwt9_CwO