

République Algérienne Démocratique et Populaire
Ministère de l'Enseignement Supérieur et de la Recherche Scientifique
Université Mouloud Mammeri de Tizi-Ouzou



Faculté De Génie Électrique et d'Informatique
Département de Télécommunications

Mémoire de Fin d'Etudes

de MASTER ACADEMIQUE

Spécialité :

Réseaux et télécommunications

Filière :

Télécommunications

Par :

KHELAFI Ibrahim

FELLAH Alaeddine

Thème :

**Stratégies et solutions pour la protection contre
les attaques DDoS (Distributed Denial of
Service)**

Soutenu le 07/07/2025.

Devant les jury :

Président : Dr. KHELIFA Mohammed Amin

Encadrant : Dr. BACHIR M'hamed Saadi

Examineur : Dr. HEDIR Abdellah

Remerciements

Avant tout, nous remercions Dieu Tout-Puissant de nous avoir accordé la force, la patience et la volonté nécessaires pour mener à bien ce travail.

*Nous tenons à exprimer notre profonde gratitude à notre encadrant, **Dr BACHIR**, pour ses conseils inestimables, sa disponibilité et son accompagnement tout au long de ce projet.*

Nous remercions également les membres du jury pour l'intérêt qu'ils ont porté à notre travail en acceptant de le lire et d'enrichir notre réflexion par leurs commentaires.

Nous sommes profondément reconnaissants envers nos familles pour leur soutien inconditionnel, leur patience, leur compréhension et leur amour tout au long de ces années d'études.

Enfin, nous exprimons nos sincères remerciements pour l'esprit de collaboration, de dévouement et de respect mutuel qui ont fait de ce travail commun une expérience académique enrichissante.

...

.....

Dédicace

Je rends tout d'abord grâce à Allah, le
Tout-Puissant, pour Sa miséricorde infinie et Son
soutien constant tout au long de ce parcours
académique.

Je dédie ce travail à ma chère famille, mes parents,
mes frères et sœurs, pour leur amour inconditionnel,
leur patience et leur soutien moral indéfectible.
J'adresse également ma sincère reconnaissance à
l'ensemble de mes professeurs, pour la qualité de
leur enseignement, leurs conseils éclairés et leur
engagement dans ma formation. Leur
accompagnement a été pour moi une source
d'inspiration précieuse.

Mes pensées vont aussi à mes amis et collègues,
pour leur bienveillance, leur entraide et les moments
de complicité partagés durant ces années.
Enfin, j'exprime ma gratitude à toutes les personnes
qui, de près ou de loin, ont contribué à la réalisation
de ce mémoire.

Ibrahim

Dédicace

Je dédie ce mémoire

À mes parents, pour leur soutien indéfectible et leur présence irremplaçable dans mon parcours.

À ma sœur, en remerciement pour sa gentillesse et son soutien.

À mes professeurs, pour l'excellence de leur enseignement, pour leurs attentes et leur disponibilité.

À mes amis, pour les merveilleux souvenirs de ces années d'études.

À toutes les personnes qui, de près ou de loin, ont contribué à la réalisation de ce travail.

Alaeddine

Résumé

Les attaques par déni de service distribué (DDoS) représentent l'une des menaces les plus sérieuses pour la stabilité des infrastructures réseau et la continuité des services en ligne. Dans ce mémoire, nous proposons une approche basée sur l'IA (Intelligence artificielle) pour la détection des attaques DDoS, en combinant deux techniques d'apprentissage non supervisé : un AE (Autoencodeur), issu du deep learning, et l'algorithme Isolation Forest, issu du machine learning. L'objectif est de concevoir un système capable de détecter automatiquement les anomalies dans le trafic réseau, sans utiliser des signatures connues.

L'étude repose sur le jeu de données CIC-IDS2019, reconnue pour sa richesse et sa pertinence dans l'évaluation des systèmes de cybersécurité. Les deux modèles ont été formés et évalués à l'aide de métriques classiques (précision, rappel, F1-score), puis intégrés dans une interface graphique interactive développée en Python, permettant à l'utilisateur de lancer la détection en temps réel et de visualiser les résultats.

Les résultats expérimentaux montrent que l'AE offre une détection très sensible mais génère davantage de faux positifs, tandis qu'Isolation Forest est plus tolérant mais laisse passer certaines attaques. Ce travail met en évidence l'intérêt de combiner ou de choisir dynamiquement les modèles selon le contexte d'application, et ouvre la voie à des systèmes de protection adaptatifs, fiables et efficaces contre les attaques DDoS.

Mots clés : Attaques DDoS, ML, IA, Apprentissage non supervisé, Auto-encodeur, Isolation Forest, DL.

Abstract

Distributed Denial of Service (DDoS) attacks remain one of the most critical threats to the stability of network infrastructures and the availability of online services. This thesis presents an artificial intelligence-based approach for DDoS detection, combining two unsupervised learning techniques : a deep learning AE (Autoencoder) and the Isolation Forest algorithm from machine learning. The goal is to design a system capable of automatically detecting anomalies in network traffic without relying on predefined signatures.

The study is based on the CIC-IDS2019 dataset, which is widely recognized for its richness and relevance in cybersecurity research. Both models were trained and evaluated using standard performance metrics (precision, recall, F1-score, accuracy), and then integrated into a Python-based interactive graphical interface, allowing real-time detection and dynamic visualization of results.

Experimental results show that the AE achieves high sensitivity but generates more false positives, while Isolation Forest offers better tolerance but may miss some attacks. This work highlights the relevance of combining or dynamically selecting detection models depending on the context, and lays the groundwork for adaptive, reliable, and efficient protection systems against DDoS attacks.

Keywords : DDoS attacks, ML, AI, Unsupervised learning, Auto-encoder, Isolation Forest, DL.

Table des matières

Remerciements	i
Dédicace	iii
Résumé	iv
Liste des figures	x
Liste des tableaux	xii
Liste des abréviations	xiii
Introduction générale	1
I Fondements des attaques DDoS	4
I.1 Introduction	4
I.2 Concepts de base en cybersécurité	4
I.2.1 Définition et enjeux des cyberattaques	4
I.2.2 La triade CIA : fondements de la sécurité informatique	4
I.2.2.1 Confidentialité	5
I.2.2.2 Intégrité	5
I.2.2.3 Disponibilité	5
I.3 Menaces ciblant la disponibilité des services	5
I.3.1 Criticité de la disponibilité des services	5
I.3.2 Secteurs particulièrement vulnérables	6
I.3.3 Typologie des attaques ciblant la disponibilité	6
I.4 Anatomie des attaques DDoS	7
I.4.1 Historique d'évolution des attaques DDoS	7
I.4.2 Le principe de distribution	8
I.4.2.1 La distribution géographique	8
I.4.2.2 La distribution temporelle	8
I.4.2.3 La distribution des ressources	8
I.4.3 Efficacité des attaques DDoS par rapport à sa version précédente .	9
I.4.4 Mécanismes d'épuisement des différentes ressources	9
I.4.4.1 Épuisement du processeur (CPU)	10
I.4.4.2 Saturation de la mémoire (RAM)	10
I.4.4.3 Congestion de la bande passante	10
I.4.4.4 Épuisement des sessions/connexions	10

I.4.4.5	Surcharge des entrées/sorties disque	10
I.4.5	Techniques d'amplification et de réflexion	10
I.4.5.1	Principe de l'amplification	11
I.4.5.2	Principe de la réflexion	11
I.4.6	Architecture et outils d'attaque.....	12
I.4.6.1	Cybercriminel (l'attaquant)	12
I.4.6.2	Les maîtres de contrôle (gestionnaires).....	12
I.4.6.3	Botnets	13
I.4.6.3.1	Programme malveillant :.....	13
I.4.6.3.2	Joindre le réseau :	13
I.4.6.3.3	Payer pour l'utiliser :	13
I.4.6.4	La cible (victime)	14
I.4.7	DDoS en tant qu'un service.....	14
I.4.8	Classification des attaques DDoS	14
I.4.8.1	Attaques volumétriques (Volume-based)	14
I.4.8.1.1	Flood UDP.....	14
I.4.8.1.2	Amplification DNS	15
I.4.8.2	Attaques protocolaires (Protocol attacks).....	15
I.4.8.2.1	SYN flood	16
I.4.8.2.2	Ping of Death.....	16
I.4.8.3	Attaques de couche application (Application layer).....	17
I.4.8.3.1	HTTP flood.....	17
I.4.8.3.2	Slowloris.....	17
I.4.9	Motivations des attaquants	18
I.5	Mécanismes de défense face aux attaques DDoS.....	18
I.5.1	Prévention.....	18
I.5.1.1	Surdimensionnement de la bande passante	18
I.5.1.2	Architecture distribuée avec répartition de charge	18
I.5.2	Détection des attaques	19
I.5.2.1	Surveillance réseau et détection comportementale	19
I.5.2.2	Détection autonome par l'IA.....	19
I.5.3	Mitigation	19
I.5.3.1	Filtrage par blacklists/whitelists	19
I.5.3.2	Rate limiting (limitation du débit).....	20
I.6	Conclusion	20
II	Détection d'anomalies à l'aide de l'intelligence artificielle	22
II.1	Introduction.....	22
II.2	Fondamentaux de l'IA.....	22
II.2.1	Définition de l'IA	22
II.2.2	Les sous-domaines de l'IA	23
II.2.2.1	L'apprentissage automatique (Machine Learning)	24
II.2.2.2	Réseaux de neurones artificiels	24
II.2.2.3	L'apprentissage profond (Deep Learning).....	25
II.2.3	Comparaison entre Deep Learning et d'autres techniques de Ma- chine Learning.....	25
II.2.3.1	Ingénierie des caractéristiques :	26
II.2.3.2	Nombre de paramètres et complexité du modèle :	26

TABLE DES MATIÈRES

II.2.3.3	Capacité d'apprentissage et volume de données :	26
II.2.3.4	Temps d'exécution :	27
II.2.3.5	Interopérabilité du modèle :	27
II.2.4	Les différents types d'apprentissage	27
II.2.4.1	Apprentissage supervisé	27
II.2.4.2	Apprentissage non supervisé	27
II.2.4.3	Apprentissage par renforcement	28
II.2.5	Exemples de domaines d'utilisation d'algorithmes de machine learning	28
II.2.6	Les étapes d'un projet d'IA	29
II.2.6.1	Définir l'objectif	30
II.2.6.2	Collecter et préparer les données	30
II.2.6.3	Choisir l'algorithme approprié	30
II.2.6.4	Entraîner le modèle	30
II.2.6.5	Évaluer le modèle	30
II.2.6.6	Mettre en production	31
II.3	Application concrète en détection d'attaques DDoS	31
II.3.1	Détermination de la base de données appropriée	32
II.3.2	Traitement des données	33
II.3.3	Sélection des fonctionnalités	33
II.3.4	Critères de sélection d'algorithmes	33
II.3.4.1	Présentation de l'algorithme Isolation Forest	33
II.3.4.1.1	Intuition et principe fondamental :	34
II.3.4.1.2	Construction des arbres d'isolation :	34
II.3.4.1.3	Formulation mathématique du score d'anomalie :	35
II.3.4.1.4	Paramétrage et complexité	35
II.3.4.1.5	Avantages et limites :	35
II.3.4.2	Présentation de l'algorithme d'AE	36
II.3.4.2.1	Architecture et principe de fonctionnement : .	36
II.3.4.2.2	Fonction de perte et d'apprentissage :	37
II.3.4.2.3	L'erreur de reconstruction :	37
II.3.4.2.4	Seuil de détection :	37
II.3.4.2.5	Avantages et limitations :	37
II.3.5	Méthodologie d'entraînement et de test	38
II.3.6	Métriques d'évaluation des performances	38
II.3.6.1	Matrice de confusion	38
II.3.6.2	Métriques de performance	38
II.3.6.2.1	Accuracy :	39
II.3.6.2.2	Précision :	39
II.3.6.2.3	Rappel (Recall) :	39
II.3.6.2.4	Score F1 :	39
II.4	Conclusion	39
III Résultats et discussions		41
III.1	Introduction	41
III.2	Objectif de notre réalisation	41
III.3	Plateforme et environnement de développement	41

TABLE DES MATIÈRES

III.3.1	Google Colab.....	42
III.3.2	Jupyter	42
III.3.3	Kaggle	42
III.3.4	Visual Studio Code	43
III.4	Outils de développement	43
III.4.1	Langage utilisé : Python	43
III.4.2	Bibliothèques utilisées.....	44
III.4.2.1	TensorFlow	44
III.4.2.2	Keras	44
III.4.2.3	Scikit-learn	44
III.4.2.4	PyQt6	45
III.5	Jeu de données utilisé	45
III.5.1	Description de la base de données	45
III.5.2	Séparation de jeu de données	47
III.6	Évaluation des performances des modèles	47
III.6.1	Résultats de l'Auto-encodeur	48
III.6.1.1	Matrice de confusion.....	49
III.6.1.2	Rapport de classification	50
III.6.1.3	Métriques globales de performance	50
III.6.2	Résultats de l'Isolation Forest	51
III.6.2.1	Matrice de confusion.....	51
III.6.2.2	Rapport de classification	52
III.6.2.3	Métriques globales de performance	53
III.6.2.4	Analyse de la séparation des classes	54
III.6.3	Comparaison globale des performances des deux modèles	55
III.6.4	Détermination du modèle optimal	55
III.7	Démonstration du service anti-DDoS.....	56
III.7.1	Conception de l'interface utilisateur Anti-DDoS	57
III.7.1.1	Création de l'interface graphique	57
III.7.1.2	Design et style visuel de l'interface	57
III.7.1.3	Implémentation des boutons de commande	57
III.7.1.4	Scripts/fonctions associés aux boutons	58
III.7.1.5	Affichage des résultats et journalisation	58
III.7.2	Exécution du système et utilisation de l'interface	59
III.7.2.1	Lancement de la plateforme	59
III.7.2.2	Choix du moteur de détection	59
III.7.2.3	Surveillance en temps réel.....	60
III.7.2.4	Détection et affichage des anomalies.....	60
III.7.2.5	Réaction de l'utilisateur face aux attaques détectées.....	61
III.7.2.6	Fin de session.....	61
III.8	Conclusion	61

Conclusion générale	62
----------------------------	-----------

References	63
-------------------	-----------

Table des figures

I.1	Triade CIA	5
I.2	comparaison Entre DoS et DDoS.....	9
I.3	Principe de réflexion et d'amplification	11
I.4	Architecture des attaques DDoS.....	12
I.5	UDP flood	15
I.6	DNS amplification	15
I.7	Syn flood.....	16
I.8	Ping death.....	16
I.9	Http flood	17
I.10	Slowloris	17
II.1	Sous-domaine de l'IA.....	23
II.2	Architecture ANN	24
II.3	Etapes de réalisation d'un projet d'IA.....	29
II.4	Organigramme de fonctionnement de notre application.....	32
II.5	Fonctionnement de l'algorithme IF.....	34
II.6	Architecture d'un AE	36
III.1	Logo de Google Colaboratory	42
III.2	Logo de Jupyter.....	42
III.3	Logo Kaggle.....	42
III.4	Logo du Visual Studio Code	43
III.5	Python logo	43
III.6	Logo de TensorFlow	44
III.7	Logo de keras	44
III.8	Logo de Scikit-learn.....	44
III.9	Logo de pyqt6	45
III.10	Contenu du Dataset	47
III.11	Division du Dataset	47
III.12	Reconstruction des données normales et anormales.....	48
III.13	Matrice de confusion.....	49
III.14	Evaluation du modèle d'Auto-encodeur.....	50
III.15	Métriques globales de performance d'Auto-encodeur	50
III.16	Matrice de confusion d'IF	51
III.17	Rapport de classification du modèle d'Isolation Forest.....	52
III.18	Métriques de performance.....	53
III.19	Analyse de la séparation des classes	54
III.20	Fonctionnement de l'application	56
III.21	Tableau de bord principal de la plateforme Anti-ddos.....	59

III.22	Interface de surveillance en attente de détection	60
III. 3	Interface de détection en temps réel des attaques DDoS via ML et DL	61

Liste des tableaux

I.1	Typologie des attaques ciblant la disponibilité	7
I.2	Comparaison entre les attaques DoS et DDoS	9
II.1	Comparaison entre le Deep Learning et Machine Learning classique	26
II.2	Applications du Machine Learning	29
II.3	Matrice de confusion.....	38
III.1	Caractéristiques du dataset CICIDS2019	45
III.2	Signification de certaines caractéristiques	46
III.3	Comparaison des performances entre l’auto-encodeur et l’Isolation Forest	55
III.4	Fonctions associées aux boutons dans l’interface graphique Anti-DDoS	58
III.5	Boutons déclenchant des scripts externes de détection dans l’interface Anti-DDoS	58

LISTE DES ABBREVIATIONS

Abréviation	Définition
AE	Autoencoder / Autoencodeur
AI / IA	Artificial Intelligence / Intelligence Artificielle
ANN	Artificial Neural Network
API	Application Programming Interface
AutoML	Automated Machine Learning
CDN	Content Delivery Network
CIC-IDS2019	Canadian Institute for Cybersecurity Intrusion Detection System 2019
CNN	Convolutional Neural Network
CPU	Central Processing Unit
CSV	Comma-Separated Values
DDoS	Distributed Denial of Service
DL	Deep Learning
DNS	Domain Name System
DoS	Denial of Service
F1-score	F-measure, moyenne harmonique précision/rappel
FN	False Negative
FNR	False Negative Rate
FP	False Positive
FPR	False Positive Rate
GPU	Graphics Processing Unit
HTTP	HyperText Transfer Protocol
ICS	Industrial Control Systems
IDE	Integrated Development Environment
IDS	Intrusion Detection System

IF	Isolation Forest
IoT	Internet of Things
KDD	Knowledge Discovery in Databases
KNN	K-Nearest Neighbors
LDAP	Lightweight Directory Access Protocol
LSTM	Long Short-Term Memory
ML	Machine Learning
MSE	Mean Squared Error
NETBIOS	Network Basic Input/Output System
NLP	Natural Language Processing
NTP	Network Time Protocol
PORTMAP	Port Mapping Protocol
PyQt6	Python Qt6 (framework d'interface graphique)
RAM	Random Access Memory
ROC	Receiver Operating Characteristic
SNMP	Simple Network Management Protocol
SVM	Support Vector Machine
SYN	Synchronize (paquet TCP SYN)
TCP	Transmission Control Protocol
TFTP	Trivial File Transfer Protocol
TN	True Negative
TNR	True Negative Rate
TP	True Positive
TPR	True Positive Rate
UDP	User Datagram Protocol

Introduction générale

L'évolution constante des technologies numériques a profondément transformé les systèmes d'information et les infrastructures réseau, les rendant plus interconnectés, plus complexes et plus exposés aux menaces. Parmi celles-ci, les attaques par déni de service distribué (DDoS) figurent aujourd'hui parmi les plus redoutables, capables de paralyser en quelques instants des services critiques, de compromettre la disponibilité des ressources et d'entraîner d'importantes pertes économiques. Face à la fréquence et à la sophistication croissantes de ces attaques, la mise en place de mécanismes de détection intelligents est devenue une nécessité stratégique [1], [2].

Dans cet objectif, les méthodes classiques de détection, souvent fondées sur des signatures ou des règles fixes, se révèlent insuffisantes face à des attaques de plus en plus variées et imprévisibles [3]. C'est dans ce contexte que l'intelligence artificielle, et plus particulièrement l'apprentissage automatique, offre des perspectives prometteuses. Capables d'apprendre à partir des données et de s'adapter à des comportements nouveaux, ces approches permettent de détecter des anomalies dans le trafic réseau de manière proactive et autonome [4], [5].

La présente étude vise à concevoir une solution de détection d'attaques DDoS fondée sur deux techniques d'apprentissage non supervisé : le modèle Isolation Forest, basé sur le principe d'isolement des points rares dans un espace de données, et le modèle AE [6], issu du deep learning, qui repose sur la reconstruction des entrées et la détection des écarts [7]. Ces deux approches, bien que différentes dans leur fonctionnement, se complètent dans leur capacité à identifier des comportements anormaux dans un environnement réseau en constante évolution [5].

La motivation principale de ce mémoire réside dans la recherche d'un équilibre entre précision de détection, faible taux de fausses alertes et simplicité d'intégration. En s'appuyant sur le jeu de données CIC-IDS2019 [8], reconnu pour sa richesse et sa représentativité, ce mémoire évalue et compare les performances des deux modèles selon des métriques standards (précision, rappel, F1-score, accuracy) [9], tout en intégrant ces outils dans une interface utilisateur interactive développée en Python. Cette interface permet d'appliquer les modèles en temps réel à des données de trafic réseau, facilitant ainsi leur exploitation dans un contexte opérationnel.

Cette démarche se veut à la fois pratique et scientifique : elle met en œuvre des concepts avancés de l'intelligence artificielle tout en proposant une solution concrète et accessible pour renforcer la cybersécurité des systèmes.

Ce mémoire est structuré en trois chapitres :

Le premier chapitre établit les fondements des attaques DDoS, en présentant leurs mécanismes, leurs typologies et les stratégies de défense existantes.

Le deuxième chapitre introduit les concepts clés de l'intelligence artificielle de l'apprentissage automatique et profond, en mettant en lumière leur pertinence dans le domaine de la cybersécurité.

Le troisième chapitre décrit la mise en œuvre et l'évaluation des deux modèles de détection, à travers une analyse comparative rigoureuse et le développement d'une interface utilisateur intuitive.

Le mémoire se conclut par une synthèse générale des résultats obtenus, accompagnée d'une réflexion sur les limites, les apports de la recherche, et les perspectives futures en matière de détection intelligente des attaques DDoS.

CHAPITRE I

Fondements des attaques DDoS

Chapitre I

Fondements des attaques DDoS

I.1 Introduction

Les attaques par déni de service distribué (DDoS) sont l'une des menaces préoccupant dans le domaine de la sécurité informatique aujourd'hui. Capables de geler les services vitaux en un clic, elles ont la capacité de rendre instantanément inopérants les services essentiels, provoquant des pertes économiques considérables et portant un coup sévère à la réputation des entreprises [1].

On va analyser dans ce chapitre les approches fondamentales de ces attaques, présente une classification des principales techniques d'attaques, et examine les stratégies de défense appropriées. Cette étude vise à fournir une compréhension approfondie de ces menaces et des moyens de s'en protéger efficacement dans un environnement en constante évolution.

I.2 Concepts de base en cybersécurité

I.2.1 Définition et enjeux des cyberattaques

Au cours des dernières années, les cybermenaces se sont considérablement évoluées, en passant d'actes de sabotage numérique à des opérations menées par des acteurs de plus en plus variés et organisés. L'interconnexion croissante des systèmes d'information et la professionnalisation du cybercrime ont créé une surface d'attaque sans précédent [10].

Les conséquences des cyberattaques ne se limitent pas au seul domaine technologique, elles touchent aussi en profondeur l'économie et le social. Chaque année, la charge totale des cyberattaques augmente alors de manière exponentielle, atteignant des sommes considérables. Économiquement, les pertes englobent la perte d'activité, le coût des remèdes, et la perte de confiance des clients.

Les cyberattaques d'aujourd'hui présentent un éventail de techniques et d'objectifs plus variés : injection, ingénierie sociale, malware, attaque réseau, exploitant chacune une spécificité d'une vulnérabilité du système d'information [1], [11].

I.2.2 La triade CIA : fondements de la sécurité informatique

La triade CIA (Confidentialité, Intégrité, Disponibilité) est le cadre conceptuel de référence en cybersécurité. Ce modèle permet une classification des menaces, selon leur

impact particulier, pour établir des priorités de protection [11], comme illustré dans la figure I.1.

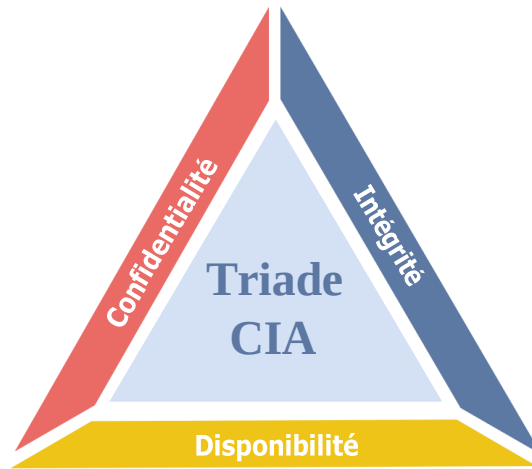


Figure I.1 – Triade CIA

I.2.2.1 Confidentialité

La confidentialité garantit que seules les personnes autorisées peuvent accéder à l'information, à partir de mesures telles que le chiffrement, les contrôles d'accès, pour protéger les données sensibles d'un accès non autorisé, espionnage ou vol de données [12].

I.2.2.2 Intégrité

L'intégrité garantit que les données sont conformes au besoin, sans altération non autorisée, par les techniques de hachage et de signature numérique pour faire les vérifications et s'assurer de l'inviolabilité des données [2].

I.2.2.3 Disponibilité

La disponibilité garantit que les systèmes et données sont accessibles en temps voulu, grâce à l'architecture redondante, à la répartition des charges ou des mécanismes de haute disponibilité pour la continuité de service [11].

Des attaques, à destination de la disponibilité comme des attaques DDoS, visent à rendre les services visés inaccessibles en saturant toutes les ressources qu'ils exploitent. Elles en mettent à contribution les limites de bande passante, de puissance de calcul, ou de mémoire afin de priver le service d'une partie de son fonctionnement, une situation qualifiée de déni de service [13].

I.3 Menaces ciblant la disponibilité des services

I.3.1 Criticité de la disponibilité des services

L'étendue des failles de sécurité les plus sophistiquées sans accès effectif au service est fondamentalement paradoxale, illustrant le fait que les investissements en confidentialité

et en intégrité deviennent inutiles si les utilisateurs légitimes n’y ont pas accès, ou si leur accès en est dénié. Cette criticité s’accroît particulièrement dans les environnements temps réel où les systèmes de contrôle industriel, les applications médicales critiques ou les services d’urgence ne tolèrent que quelques secondes d’interruption avant d’engendrer des conséquences catastrophiques sur la sécurité physique des personnes. Au-delà des aspects techniques, la disponibilité impacte directement et immédiatement la réputation d’un service, car contrairement aux violations de confidentialité qui peuvent rester invisibles, toute indisponibilité génère instantanément une méfiance durable chez les utilisateurs, amplifiée par la diffusion immédiate sur les réseaux sociaux. Cette visibilité se traduit par des conséquences économiques immédiates et quantifiables : pertes de revenus, coûts de remédiation, pénalités contractuelles et impact boursier, créant une pression financière directe qui contraste avec d’autres types d’attaques aux impacts souvent différés ou difficiles à évaluer [11], [12].

I.3.2 Secteurs particulièrement vulnérables

La portée des attaques de disponibilité est très variable, en fonction des secteurs d’activités, certains d’entre eux étant exposés dès les premières étapes du processus, en raison de leurs spécificités visant à assurer la continuité de leurs fonctions et de leur importance sociétale. Le secteur financier est emblématique de cette vulnérabilité vu sa soumission au fonctionnement en temps réel des systèmes interconnectés. De même, le commerce électronique subit un impact financier direct et immédiat de chaque minute d’indisponibilité, particulièrement critique pendant les périodes de forte affluence, cette vulnérabilité étant amplifiée par la dépendance croissante des consommateurs au commerce numérique depuis la pandémie de COVID-19. Les services publics essentiels (énergie, eau, transports, télécommunications) constituent quant à eux des cibles stratégiques dont l’interruption dépasse le cadre économique pour affecter la sécurité publique et le bien-être des citoyens, leur digitalisation croissante avec l’adoption des technologies IoT créant de nouvelles vulnérabilités exploitables à distance.

Enfin, les infrastructures critiques englobent l’ensemble des systèmes indispensables au fonctionnement national, leur interconnexion croissante amplifiant les risques d’effets en cascade, ce qui en fait des cibles particulièrement sensibles pouvant être motivées par des objectifs géopolitiques, terroristes ou de déstabilisation sociale, nécessitant ainsi des mesures de protection renforcées et une coordination entre secteurs public et privé [10], [14].

I.3.3 Typologie des attaques ciblant la disponibilité

Les offensives visant à compromettre la disponibilité s’organisent autour de trois approches essentielles : le refus d’accès aux données, qui rend les informations inaccessibles sans pour autant les anéantir physiquement [9], le déni d’accès aux infrastructures qui compromet les fondations mêmes des systèmes [11], et l’épuisement des ressources système qui sature les capacités de traitement pour empêcher le service légitime [10]. Le tableau suivant détaille cette typologie en présentant les mécanismes et impacts spécifiques de chaque type d’attaque :

Table I.1 – Typologie des attaques ciblant la disponibilité

Catégorie	Type d'attaque	Mécanisme	Impact
Déni d'accès aux données	Ransomware	Chiffrement malveillant avec demande de rançon	Données inaccessibles, récupération conditionnée au paiement ou à des sauvegardes
	Destruction / Corruption	Altération irréversible ou suppression ciblée des données critiques	Récupération uniquement possible via des sauvegardes fiables
Déni d'accès aux infrastructures	Sabotage physique / logique	Destruction d'équipements, corruption de fichiers ou configurations système	Pannes durables, nécessite intervention physique ou restauration technique complexe
	Pannes provoquées	Manipulations exploitant des failles système ou erreurs logiques	Crashes apparemment accidentels, difficilement traçables
Épuisement de ressources système	DoS	Surcharge via source unique (trafic ou requêtes massives)	Indisponibilité temporaire, saturation ciblée
	DDoS	Sources multiples coordonnées (botnets)	Impact massif, contournement des protections classiques

I.4 Anatomie des attaques DDoS

I.4.1 Historique d'évolution des attaques DDoS

Les attaques par déni de service (DoS), qui ont commencé de façon simple, sont devenues des attaques distribuées (DDoS) plus fortes. Elles utilisent des réseaux de machines compromises appelées botnets. Cette change a apporté une variation dans les buts, allant des actions de protestation au sabotage des infrastructures vitales [1], [15].

Voici les principales étapes marquant cette évolution :

- **Avant 1999 : Premières attaques DoS**

Les premières attaques visaient à rendre des services indisponibles depuis une seule machine, en exploitant des failles simples comme les *Ping of Death* ou les *SYN flood*.

- **1999 – 2005 : Émergence du DDoS**

Le DDoS apparaît avec des outils comme *Trin00* ou *TFN*, coordonnant des machines zombies pour lancer des attaques massives, avec des cas célèbres en 2000 contre Yahoo !, eBay ou CNN.

- **2006 – 2012 : Professionnalisation**

Les attaques deviennent plus organisées et motivées politiquement, à l'image des actions d'*Anonymous* ou des attaques étatiques, notamment en Estonie en 2007.

- **2013 – 2016 : Amplification massive**

Les techniques d'amplification (DNS, NTP, SSDP) permettent d'atteindre des débits records. L'apparition du botnet *Mirai* marque un tournant avec l'exploitation des objets connectés vulnérables.

- **2017 – 2020 : Démocratisation des attaques**

Les attaques DDoS deviennent accessibles via des services payants (*booter/stresser*), ciblant aussi les services cloud avec des charges atteignant plusieurs téraoctets par seconde.

- **2021 – Aujourd'hui : Sophistication et multi-couches**

Les attaques sont plus complexes, ciblant les couches applicatives (HTTP, API), et visent des secteurs critiques comme la santé ou les gouvernements, dans un contexte de tensions géopolitiques croissantes.

I.4.2 Le principe de distribution

La caractéristique essentielle qui permet de distinguer les attaques DoS et DDoS réside dans leur architecture de mise en œuvre : si une source unique limitée dans ses ressources est à l'origine, il s'agit d'une attaque DoS, avec l'attaque DDoS, c'est un ensemble de sources multiples, distribuées qui produit une puissance d'attaque plus efficace et plus complexe, qui repose donc sur la capacité à pouvoir combiner des sources non identifiables et géographiquement dispersées, ce qui rend les dispositifs de protection traditionnels face à l'attaque incapables de résoudre ce défi [13].

Cette distribution s'articule autour de trois dimensions principales : la distribution géographique, la distribution temporelle, et la distribution des ressources.

I.4.2.1 La distribution géographique

Permet de coordonner des machines situées dans différents pays, continents et réseaux. Cette répartition géographique rend impossible le blocage par zones géographiques et complique considérablement l'identification des sources d'attaque. Un botnet DDoS peut ainsi mobiliser simultanément des ordinateurs zombies en Asie, en Europe et en Amérique [10].

I.4.2.2 La distribution temporelle

Consiste à orchestrer les attaques selon des patterns temporels variables. Les sources peuvent être activées par vagues successives, créant des pics d'intensité difficiles à prévoir. Cette coordination temporelle permet de maintenir une pression constante sur la cible tout en évitant la saturation immédiate du réseau d'attaque [16].

I.4.2.3 La distribution des ressources

Exploite la diversité des capacités des machines compromises. Des serveurs haute performance, des ordinateurs personnels, des dispositifs IoT. Cette hétérogénéité crée un flux d'attaque complexe et imprévisible, où chaque source apporte sa propre signature de trafic [17].

I.4.3 Efficacité des attaques DDoS par rapport à sa version précédente

Bien que ces deux attaques visent le même objectif de déni de service, elles se distinguent par leur architecture, cette distinction fondamentale est illustrée dans la figure I.2 ci-dessous qui montre la différence d'origine entre les deux types d'attaques.

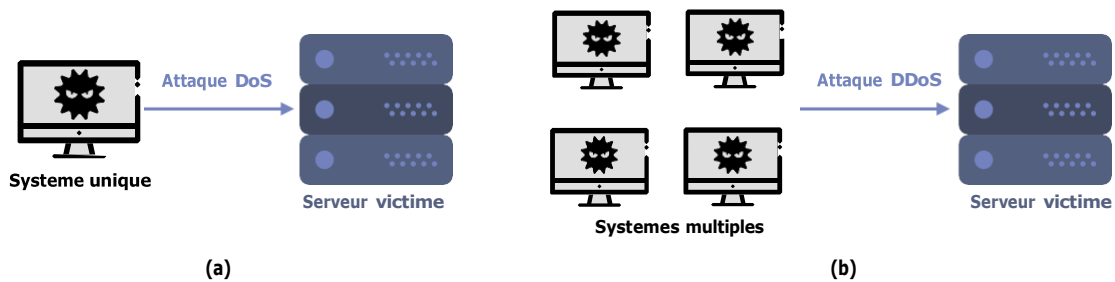


FIGURE I.2 – comparaison Entre DoS et DDoS

L'évolution des attaques DoS vers DDoS marque une transformation majeure en termes de complexité, d'efficacité et de moyens de défense, comme l'illustre le tableau I.2 comparatif suivant :

Table I.2 – Comparaison entre les attaques DoS et DDoS

Critère	DoS (Déni de Service)	DDoS (Déni de Service Distribué)
Puissance	Limitée par les ressources de l'attaquant	Amplifiée par l'effet cumulatif
Détection	Relativement facile (source identifiable)	Complexe (multiples points d'origine)
Mitigation	Blocage direct de la source	Nécessite une approche multi-couche
Coût pour l'attaquant	Élevé (utilise ses propres ressources)	Faible (exploite des ressources compromises)
Persistance	Limitée dans le temps	Peut être maintenue plus longtemps

I.4.4 Mécanismes d'épuisement des différentes ressources

Les attaques DDoS perturbent l'accès aux services en s'attaquant de manière systématique aux ressources essentielles du système visé. L'idée principale repose sur l'exploitation des limites, tant physiques que logiques, de chaque infrastructure informatique. L'efficacité de ces attaques provient de leur capacité à convertir des éléments normalement accessibles au public en véritables facteurs de limitation. Cette stratégie complexe d'épuisement

sement des ressources permet aux assaillants de cibler simultanément plusieurs points de congestion.

I.4.4.1 Épuisement du processeur (CPU)

Les attaques dirigées contre le processeur ont pour but de submerger les ressources de calcul du serveur en créant des requêtes qui demandent un traitement important. Ces attaques exploitent des fonctions qui consomment beaucoup de ressources, telles que les algorithmes de cryptage, les opérations complexes sur les bases de données ou le traitement d'images. La saturation du CPU provoque des ralentissements généralisés et peut rendre le système totalement inaccessible [18].

I.4.4.2 Saturation de la mémoire (RAM)

L'épuisement de la mémoire vive se produit par l'accumulation de processus, de connexions ouvertes ou de données temporaires en mémoire. Les attaques peuvent tirer parti des fuites de mémoire dans les applications ou maintenir de nombreuses connexions partiellement établies afin de consommer progressivement toute la mémoire disponible. Une fois la RAM saturée, le système devient instable et peut planter [1], [12].

I.4.4.3 Congestion de la bande passante

La saturation de la bande passante constitue l'approche la plus directe des attaques DDoS. En inondant les liens réseau de trafic malveillant, ces attaques empêchent le trafic légitime d'atteindre sa destination. La bande passante peut être saturée au niveau du fournisseur d'accès, des interconnexions réseau, ou de l'infrastructure locale de la victime [10].

I.4.4.4 Épuisement des sessions/connexions

Les attaques visant les connexions exploitent les limites des protocoles de communication, notamment TCP. En ouvrant un grand nombre de connexions partielles ou en maintenant des connexions actives sans les utiliser, ces attaques saturer les tables de connexions des équipements réseau et empêchent l'établissement de nouvelles connexions légitimes [19].

I.4.4.5 Surcharge des entrées/sorties disque

L'épuisement des ressources de stockage s'effectue par l'écriture excessive de données, la création de nombreux fichiers temporaires, ou la saturation des opérations d'entrée/sortie disque. Ces attaques peuvent cibler les logs système, les bases de données, ou les caches disque pour ralentir considérablement les performances du système [12].

I.4.5 Techniques d'amplification et de réflexion

Au-delà de l'épuisement direct des ressources, les attaques DDoS exploitent deux techniques fondamentales qui démultiplient considérablement leur puissance destructrice : l'amplification et la réflexion. Ces méthodes permettent aux attaquants de transformer un flux d'attaque relativement modeste en un déluge de trafic dévastateur, optimisant ainsi le rapport entre les ressources investies et l'impact obtenu.

I.4.5.1 Principe de l'amplification

L'amplification tire parti de la variante fondamentale que l'on observe dans divers protocoles réseau, où une requête de dimensions réduites peut entraîner une réponse largement plus conséquente. Cette inégalité entre la taille de la requête envoyée et celle de la réponse obtenue représente le facteur d'amplification, qui peut osciller entre quelques unités et plusieurs centaines en fonction du protocole utilisé.

Les attaquants tirent parti de cette caractéristique en générant massivement des requêtes minimales vers des services publics, créant ainsi un volume de trafic sortant disproportionné par rapport à leur investissement en bande passante. Cette technique permet à un attaquant disposant de ressources limitées de générer un impact considérable, transformant chaque octet envoyé en dizaines ou centaines d'octets dirigés vers la victime.

L'efficacité de l'amplification repose sur l'identification et l'exploitation de protocoles présentant les facteurs d'amplification les plus élevés. Les cybercriminels recherchent constamment de nouveaux services susceptibles d'être détournés, explorant les fonctionnalités de diagnostic, les services de résolution, ou les protocoles de gestion réseau qui génèrent naturellement des réponses volumineuses [19].

I.4.5.2 Principe de la réflexion

La réflexion complète le dispositif d'amplification en détournant des serveurs légitimes pour qu'ils participent involontairement à l'attaque. Cette technique s'appuie sur la falsification de l'adresse IP source (spoofing), permettant aux attaquants de rediriger les réponses amplifiées vers la victime plutôt que vers eux-mêmes.

Le mécanisme de réflexion transforme ainsi l'infrastructure Internet mondiale en un arsenal potentiel. Chaque serveur public capable de répondre à des requêtes non authentifiées devient un réflecteur involontaire, multipliant la surface d'attaque disponible. Cette approche présente l'avantage tactique de masquer la source réelle de l'attaque, les flux malveillants semblant provenir de serveurs légitimes dispersés géographiquement. La réflexion permet également de contourner certaines protections basées sur la géolocalisation ou le filtrage d'adresses IP. En utilisant des réflecteurs situés dans des zones géographiques autorisées ou en exploitant des serveurs de confiance, les attaquants peuvent contourner les mécanismes de défense périmétrique traditionnels [1], [16].

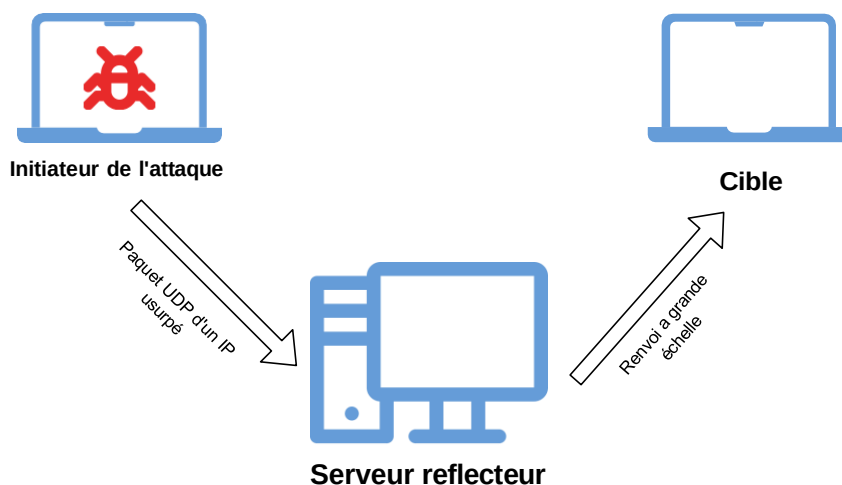


FIGURE I.3 – Principe de réflexion et d'amplification

I.4.6 Architecture et outils d'attaque

Les attaques DDoS modernes reposent sur une architecture indirecte où l'attaquant évite la confrontation directe avec sa cible. Les cybercriminels utilisent plusieurs couches d'intermédiaires pour masquer leur identité, amplifier leur puissance d'attaque et contourner les défenses. Cette approche stratifiée nécessite des outils spécialisés, notamment les gestionnaires de contrôle et les botnets [10].

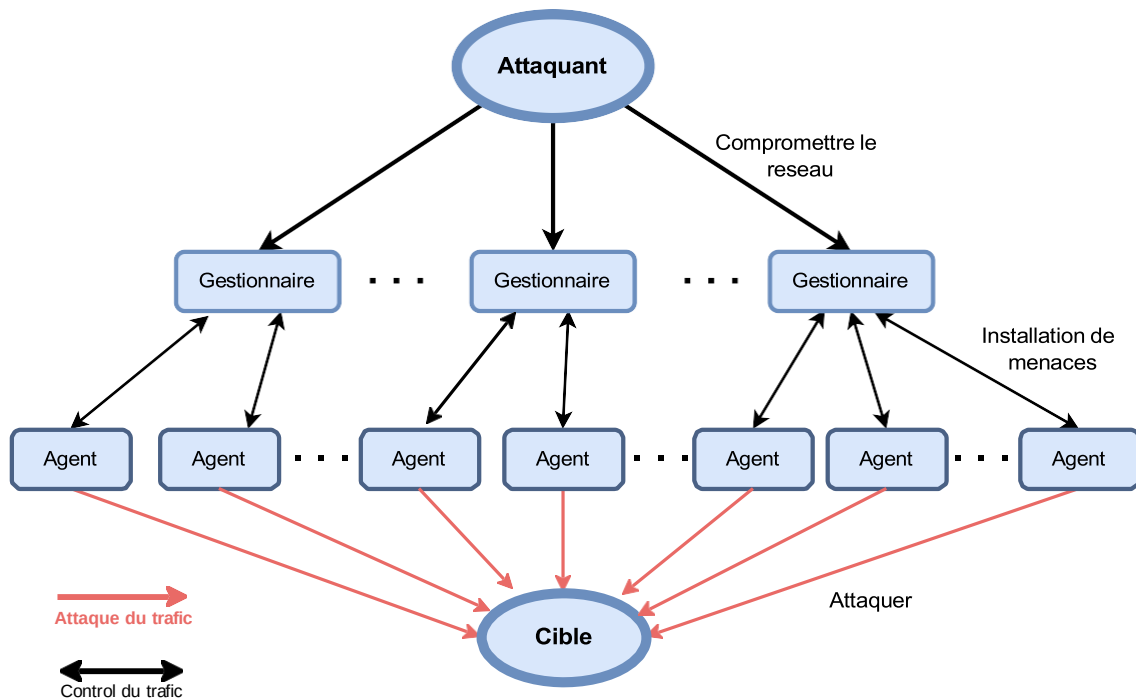


Figure I.4 – Architecture des attaques DDoS [13].

I.4.6.1 Cybercriminel (l'attaquant)

Le cybercriminel constitue le cerveau stratégique de l'attaque DDoS, orchestrant l'ensemble de l'infrastructure malveillante depuis la conception jusqu'à l'exécution. Il sélectionne les cibles, déploie les serveurs de commande et contrôle, constitue ou loue des réseaux de botnets, puis coordonne l'attaque distribuée pour saturer les ressources de la victime et rendre ses services indisponibles [16].

I.4.6.2 Les maîtres de contrôle (gestionnaires)

Les maîtres de contrôle représentent l'élément central de coordination dans l'architecture des attaques DDoS distribuées.

Ces serveurs de commande et contrôle (C&C) servent d'interface de pilotage entre l'attaquant principal et l'ensemble des machines compromises du réseau. Ils centralisent la gestion opérationnelle en recevant les instructions de l'attaquant, les traitent et les redistribuent vers les différents nœuds du botnet.

L'architecture de contrôle peut adopter différentes topologies selon les besoins de l'attaque : une structure centralisée avec un serveur C&C unique pour les opérations simples, une architecture hiérarchique avec plusieurs niveaux de gestionnaires pour les réseaux étendus, ou une approche décentralisée peer-to-peer pour maximiser la résilience face

aux contre-mesures. Ces gestionnaires intègrent souvent des mécanismes de communication chiffrée, des systèmes de sauvegarde automatique, et des protocoles de migration pour maintenir le contrôle même en cas de découverte par les autorités [19].

I.4.6.3 Botnets

Les botnets constituent l'infrastructure fondamentale des attaques DDoS modernes. Ces réseaux de machines infectées, contrôlées à distance par les cybercriminels, fournissent la puissance de calcul et la distribution géographique nécessaires pour mener des attaques de grande ampleur.

Les attaquants ont plusieurs options d'accès pour avoir ce réseau de zombies :

I.4.6.3.1 Programme malveillant : La création d'un botnet par programme malveillant constitue la méthode la plus courante et la plus autonome. Les cybercriminels développent ou achètent des malwares spécialisés dans l'infection de masse, capables de se propager automatiquement à travers différents vecteurs d'attaque.

Ces malwares exploitent les vulnérabilités des systèmes d'exploitation, des navigateurs web, ou des applications pour s'installer silencieusement sur les machines cibles. Une fois installés, ils établissent une communication avec les serveurs de commande et contrôle (C&C) et transforment la machine infectée en zombie obéissant aux instructions reçues. Les techniques de propagation incluent l'exploitation de vulnérabilités zero-day, l'utilisation de kits d'exploitation automatisés, l'infection par clés USB, ou la compromission de sites web légitimes pour distribuer le malware. Les dispositifs IoT constituent des cibles particulièrement attractives en raison de leurs protections souvent insuffisantes et de leurs mots de passe par défaut rarement modifiés [20].

I.4.6.3.2 Joindre le réseau : Certains individus rejoignent volontairement des réseaux de botnets existants, soit par idéologie (hacktivisme), soit pour participer à des actions collectives de protestation numérique. Cette approche implique le téléchargement et l'installation volontaire d'outils d'attaque qui connectent l'ordinateur de l'utilisateur au réseau de commande.

Cette méthode est particulièrement utilisée dans le contexte d'actions coordonnées comme celles menées par des groupes d'hacktivistes. Les participants téléchargent des outils comme LOIC (Low Orbit Ion Cannon) et configurent leur machine pour participer aux attaques dirigées contre des cibles spécifiques.

Bien que l'adhésion soit volontaire, les participants s'exposent aux mêmes risques juridiques que les opérateurs de botnets malveillants, car la participation à des attaques DDoS reste illégale dans la plupart des juridictions [21].

I.4.6.3.3 Payer pour l'utiliser : Le modèle "botnet-as-a-Service" permet aux cybercriminels de louer l'accès à des réseaux de machines compromises. Les fournisseurs proposent des packages selon la durée, le nombre de machines, leur localisation et capacités techniques.

Les tarifs varient selon l'ampleur et la durée des attaques, allant de montants modiques pour des attaques courtes à des sommes importantes pour des campagnes prolongées avec des centaines de milliers de zombies.

Cette commercialisation démocratise l'accès aux attaques DDoS via des interfaces conviviales, un support technique et des garanties de performance, permettant même aux non-experts de mener des attaques sophistiquées [10], [15].

I.4.6.4 La cible (victime)

La victime représente le point de convergence final de l'architecture d'attaque. Elle peut être un serveur web, une infrastructure réseau, ou tout système accessible via Internet, où converge tout le trafic malveillant généré par les botnets. Positionnée en bout de chaîne, elle doit simultanément traiter les requêtes légitimes de ses utilisateurs et absorber le flot massif orchestré par l'attaquant. Cette double sollicitation provoque la saturation de ses ressources, conduisant à l'indisponibilité du service visé [13].

I.4.7 DDoS en tant qu'un service

Le modèle DDoS-as-a-Service commercialise les attaques par déni de service via des plateformes web accessibles, démocratisant l'accès aux cyberattaques pour tout utilisateur sans compétences techniques. Ces services proposent des interfaces simplifiées permettant de sélectionner une cible, définir la durée et l'intensité de l'attaque, effectuer le paiement en cryptomonnaies, puis lancer l'offensive en quelques clics. Contrairement à l'utilisation directe de botnets qui nécessite encore des connaissances techniques, ces plateformes éliminent totalement cette barrière en offrant une couche d'abstraction complète. L'utilisateur final n'a qu'à payer et spécifier son objectif tandis que les opérateurs gèrent toute la complexité technique en exploitant leur accès aux réseaux de botnets. Cette approche transforme l'attaque DDoS en produit de consommation standard, avec des tarifs débutant à quelques dollars, des garanties de performance, un support technique, et une disponibilité continue 24h/24 [15], [21].

I.4.8 Classification des attaques DDoS

Les attaques DDoS se classifient selon trois catégories principales, chacune ciblant différentes couches du modèle réseau et exploitant des vulnérabilités spécifiques.

I.4.8.1 Attaques volumétriques (Volume-based)

Les attaques volumétriques visent à saturer la bande passante réseau en générant un volume de trafic massif vers la cible. Ces attaques exploitent la capacité limitée des liens réseau et des infrastructures de connectivité.

Les attaques volumétriques se mesurent en bits par seconde (bps), reflétant leur objectif de saturation de bande passante. Les attaques modernes peuvent atteindre plusieurs téraoctets par seconde (Tbps), dépassant largement la capacité de la plupart des infrastructures réseau [22].

I.4.8.1.1 Flood UDP Les attaques UDP flood exploitent le protocole UDP (User Datagram Protocol) pour inonder la cible de paquets UDP vers des ports aléatoires ou spécifiques. L'absence de mécanisme de contrôle de flux dans UDP permet aux attaquants de générer un volume de trafic important sans établir de connexions, rendant ces attaques particulièrement efficaces pour saturer la bande passante [19].

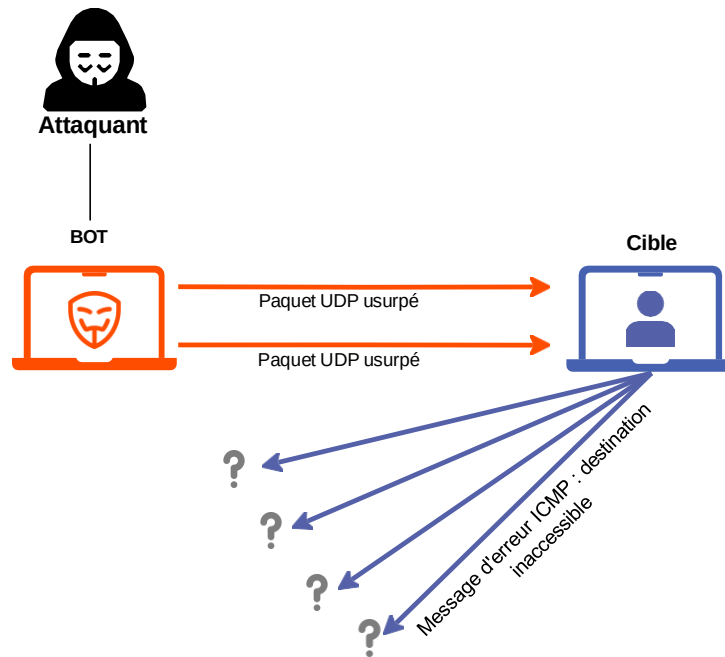


Figure I.5 – UDP flood

I.4.8.1.2 Amplification DNS Les attaques par amplification DNS exploitent des serveurs DNS publics en envoyant de petites requêtes avec l'adresse IP source falsifiée (spoo-fée) de la victime. Les serveurs DNS répondent avec des réponses volumineuses directement à la victime, créant un effet d'amplification pouvant atteindre un facteur de 50 à 100 [21].

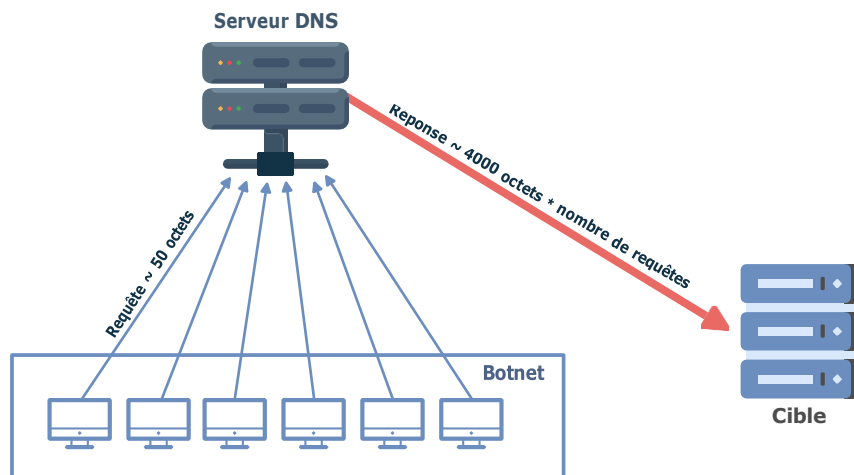


Figure I.6 – DNS amplification

I.4.8.2 Attaques protocolaires (Protocol attacks)

Les attaques protocolaires exploitent les faiblesses des protocoles de communication réseau pour épuiser les ressources des équipements d'infrastructure.

Les attaques protocolaires se mesurent en paquets par seconde (pps), car leur efficacité dépend du nombre de paquets traités plutôt que de leur volume total. Un équipement peut être saturé par quelques millions de paquets par seconde [10].

I.4.8.2.1 SYN flood L'attaque SYN flood exploite le mécanisme d'établissement de connexion TCP en envoyant de nombreuses requêtes SYN sans répondre aux accusés de réception SYN-ACK du serveur. Cette technique sature la table des connexions semi-ouvertes du serveur, l'empêchant d'accepter de nouvelles connexions légitimes [21].

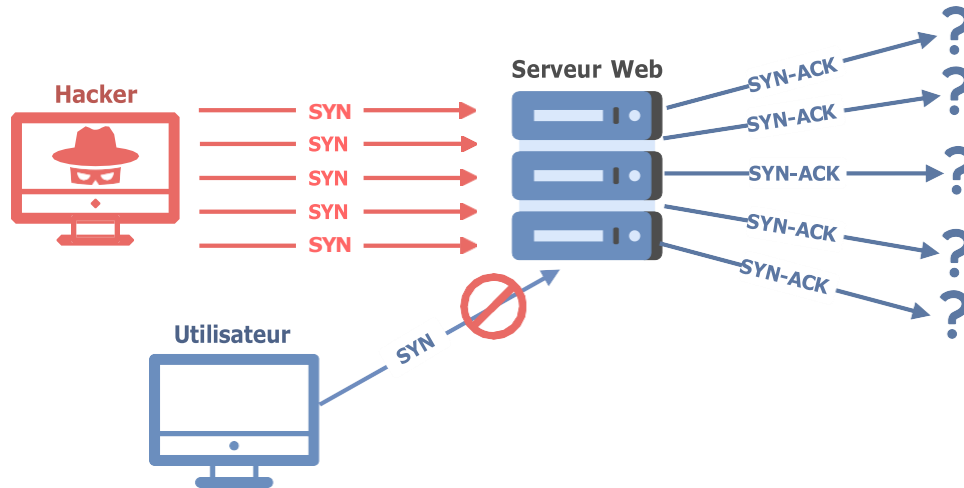


Figure I.7 – Syn flood

I.4.8.2.2 Ping of Death L'attaque Ping of Death détourne le mécanisme légitime du protocole ICMP, normalement utilisé pour tester la connectivité via des requêtes Echo Request/Reply de petite taille. L'attaquant crée des paquets ICMP surdimensionnés, fragmentés en segments individuellement conformes aux standards. Lors de la reconstitution par le système cible pour générer sa réponse, la taille excessive provoque un débordement de mémoire pouvant crasher le système [1].

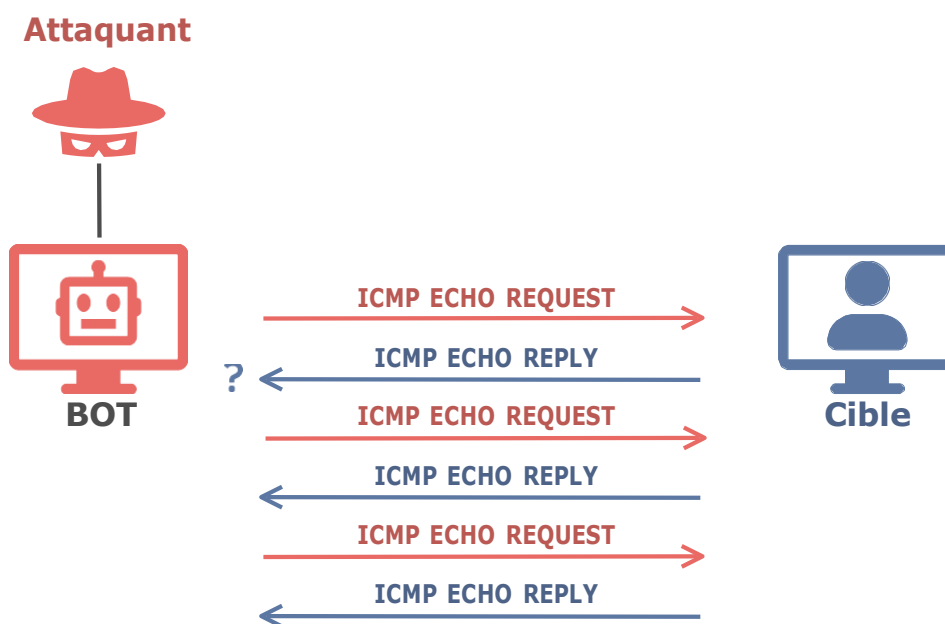


Figure I.8 – Ping death

I.4.8.3 Attaques de couche application (Application layer)

Les attaques de couche application ciblent spécifiquement les services et applications web en exploitant leurs mécanismes de fonctionnement.

Les attaques de couche application se mesurent en requêtes par seconde (rps), reflétant leur focus sur la saturation des capacités de traitement applicatif [10].

I.4.8.3.1 HTTP flood Les attaques HTTP flood inondent les serveurs web de requêtes HTTP apparemment légitimes. Ces requêtes peuvent cibler des pages statiques pour saturer la bande passante ou des pages dynamiques nécessitant un traitement intensif pour épuiser les ressources serveur [1], [21].

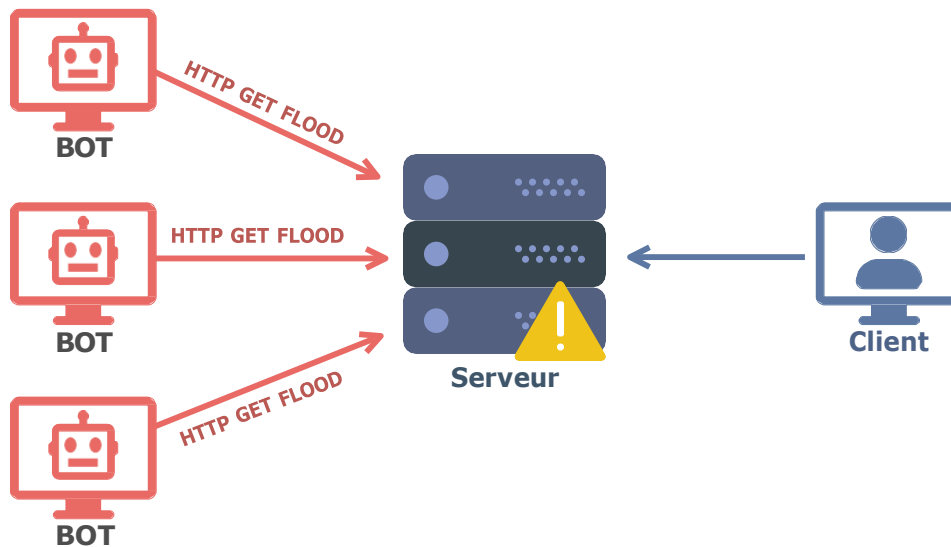


Figure I.9 – Http flood

I.4.8.3.2 Slowloris Slowloris exploite la gestion des connexions HTTP en maintenant de nombreuses connexions partiellement ouvertes vers le serveur cible. L'attaque envoie des en-têtes HTTP incomplets puis maintient chaque connexion active en envoyant périodiquement des données supplémentaires, saturant le pool de connexions disponibles [20].

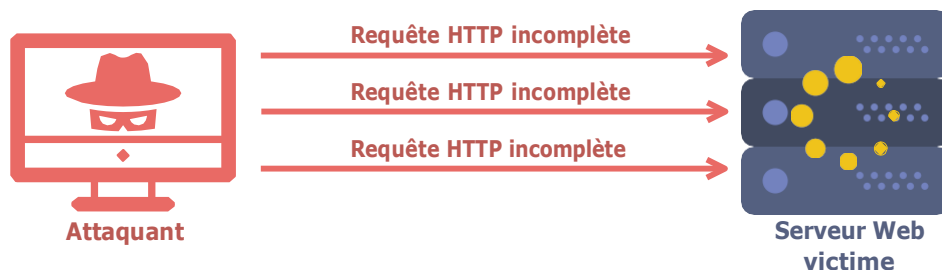


Figure I.10 – Slowloris

I.4.9 Motivations des attaquants

Les raisons qui poussent à ces attaques DDoS varient largement, allant du profit financier à des considérations politiques.

L'extorsion apparaît comme la motivation principale, où les cybercriminels menacent de mettre hors service des services essentiels d'entreprises fortement dépendantes de leur activité en ligne en particulier celles de commerce électronique et de services financiers en échange d'une rançon [20].

L'hacktivisme représente une autre dimension importante, utilisant les attaques comme moyen de protestation numérique contre des organisations ou gouvernements contestés, cherchant à attirer l'attention médiatique sur leurs revendications - comme l'illustre l'attaque "Tuflo" contre Mastercard en représailles au blocage des dons WikiLeaks [15].

La concurrence déloyale motive également certaines offensives, ciblant stratégiquement les concurrents directs lors de périodes critiques comme les lancements de produits pour maximiser l'impact économique [1].

Enfin, la démonstration de force technique permet aux groupes cybercriminels d'établir leur réputation dans la communauté underground ou de servir de vitrine préalable à d'autres activités malveillantes plus lucratives [10], [16].

I.5 Mécanismes de défense face aux attaques DDoS

Les attaques DDoS constituent une menace sérieuse pour la disponibilité des services. Pour y faire face efficacement, les organisations s'appuient sur une combinaison de stratégies préventives, de techniques de mitigation, et de mécanismes de détection.

I.5.1 Prévention

La prévention constitue la première ligne de défense contre les attaques DDoS. Elle vise à renforcer l'infrastructure et l'architecture des systèmes pour minimiser les vulnérabilités et améliorer la résistance aux attaques avant qu'elles ne se produisent [16].

I.5.1.1 Surdimensionnement de la bande passante

Prévoir une capacité réseau bien supérieure à la consommation habituelle permet d'amortir l'impact d'une attaque en absorbant temporairement le trafic excédentaire. Cette méthode offre un temps de réaction crucial, mais ne constitue pas une solution à long terme face aux attaques massives [22].

I.5.1.2 Architecture distribuée avec répartition de charge

La distribution des services sur plusieurs sites géographiques et autonomes, combinée à une répartition intelligente du trafic entre plusieurs serveurs, améliore considérablement la tolérance aux pannes. Des algorithmes tels que round-robin ou la répartition par charge permettent d'éviter la saturation d'un point unique et de rediriger le trafic vers des zones non affectées lors d'une attaque [1], [21].

I.5.2 Détection des attaques

La détection précoce des attaques DDoS permet de déclencher rapidement les mécanismes de protection et de limiter l'impact sur les services. Cette capacité repose sur des systèmes de monitoring sophistiqués et des techniques d'analyse avancées comme :

I.5.2.1 Surveillance réseau et détection comportementale

L'analyse continue du trafic réseau (volume, fréquence, géolocalisation, protocoles) permet d'identifier des anomalies révélatrices d'attaque. Les systèmes de détection d'intrusion (IDS), de prévention (IPS), et les solutions SIEM centralisent et croisent les données de logs, équipements et applications afin de révéler des comportements anormaux, même si les signatures ne sont pas connues à l'avance [8].

- Outils spécifiques : Des solutions SIEM comme Splunk, IBM QRadar, et LogRhythm sont couramment utilisées pour la surveillance et l'analyse des logs. Les IDS/IPS comme Snort et Suricata sont également largement déployés pour la détection des intrusions.
- Exemples de comportements anormaux : Une augmentation soudaine du trafic en provenance d'une seule source, des requêtes répétitives sur une courte période, ou des schémas de trafic inhabituels peuvent indiquer une attaque DDoS.
- Métriques surveillées : Les métriques incluent le nombre de paquets par seconde, la bande passante utilisée, les requêtes par seconde, et les taux d'erreur des connexions.

I.5.2.2 Détection autonome par l'IA

L'IA, et plus particulièrement l'apprentissage automatique (machine learning), occupe désormais une place centrale dans la détection des attaques DDoS. Contrairement aux systèmes classiques qui reposent sur des signatures ou des règles statiques, les solutions basées sur l'IA sont capables de détecter de manière autonome des menaces inconnues. En analysant en continu les flux réseau, ces systèmes apprennent les comportements normaux à partir des données historiques, identifient de manière proactive les anomalies même subtiles, et déclenchent automatiquement des alertes ou des mécanismes de protection sans intervention humaine. Cette capacité d'adaptation en temps réel permet de faire face à des attaques évolutives et sophistiquées qui échappent aux outils traditionnels [23].

I.5.3 Mitigation

Une fois l'attaque détectée, les techniques de mitigation visent à réduire son impact et à restaurer la disponibilité des services. Ces techniques peuvent être appliquées automatiquement ou manuellement selon la criticité de la situation [22].

I.5.3.1 Filtrage par blacklists/whitelists

Le blocage des sources connues d'attaques via des listes noires (blacklists) ou l'autorisation exclusive de sources fiables via des listes blanches (whitelists) permet de réduire significativement le volume de trafic malveillant [16].

I.5.3.2 Rate limiting (limitation du débit)

Cette technique contrôle le nombre de requêtes autorisées par source ou par session, selon des règles statiques ou dynamiques. Elle est particulièrement efficace pour contrer les attaques par saturation (flooding) sans nuire à l'expérience des utilisateurs légitimes [20].

I.6 Conclusion

Ce premier chapitre a permis d'explorer les différentes approches de détection des attaques DDoS. Nous avons identifié plusieurs méthodes existantes, notamment les techniques de détection d'anomalies qui visent à identifier tout comportement anormal dans le trafic réseau.

Cette analyse préliminaire souligne la diversité des solutions disponibles et met en évidence l'importance de disposer de mécanismes de détection efficaces face à l'évolution constante des menaces DDoS.

Le chapitre suivant abordera l'intégration de l'IA dans les systèmes de détection d'anomalies, ouvrant ainsi la voie à des approches plus sophistiquées et adaptatives pour contrer les attaques DDos.

CHAPITRE II

Détection d'anomalies à l'aide de l'intelligence artificielle

Chapitre II

Détection d'anomalies à l'aide de l'intelligence artificielle

II.1 Introduction

Avec le développement rapide de la technologie numérique, des nouveaux systèmes sont apparus. Ces derniers peuvent faire des tâches seuls sans intervention de l'être humain. Et parmi ces développements on trouve apprentissage automatique ou ce qui est connu sous le nom des systèmes qui apprennent par l'expérience, , sans être spécifiquement programmé pour chaque cas ou situation, comme cela était auparavant nécessaire. L'IA contient plusieurs domaines, tels que le traitement du langage naturel (pour comprendre ou générer des textes), le traitement d'images (la reconnaissance d'objets ou les caractères du visages) et la gestion des bases de données, dans le but d'améliorer l'intelligence, la flexibilité, la réactivité et le jugement des machines. Dans ce chapitre on va présenter d'abord les principaux concepts de l'apprentissage automatique et on va expliquer pourquoi il est devenu une technologie très importante dans le cas de moderniser les différents domaines (ou en va présenter la possibilité d'apprentissage et d'amélioration des systèmes au cours du temps dans le but de l'exploiter pour aider les humains à gérer les situations difficiles ou les défis posés par le monde actuel monde actuel) actuel [24], [25], [26].

II.2 Fondamentaux de l'IA

Avant d'aborder les méthodes de détection d'anomalies utilisées dans ce travail, il est essentiel de commencer par une présentation générale de l'IA, domaine fondamental sur lequel reposent ces approches.

II.2.1 Définition de l'IA

L'IA (IA) est une discipline qui a pour but de concevoir une intelligence de type humain dans les machines, c'est-à-dire qu'il s'agit d'une technologie qui permet de faire quelque chose qu'un être humain est capable de faire, par exemple apprendre, comprendre, analyser ou décider [5], [27], [28].

Un programme de ce type est classiquement nommé agent intelligent, qui est un système qui interagit avec son environnement par l'intermédiaire de capteurs pour percevoir

son état et d'actionneurs pour l'affecter [29].

L'objectif final de cette IA consiste à concevoir cette intelligence de type humain dans les machines, notamment par des algorithmes d'apprentissage qui tentent d'imiter le fonctionnement du cerveau humain [30].

Le sous-champ de l'IA tâche d'inventorier les façons possibles, et finalement réalistes, d'attribuer à nos machines une intelligence semblable à la nôtre.

Ainsi se dessine un domaine de recherche, dit apprentissage automatique (Machine Learning) : l'apprentissage et l'intelligence des machines y sont acquis non plus par programmation explicite [24].

II.2.2 Les sous-domaines de l'IA

Les grands sous-domaines de l'IA comprennent la vision artificielle pour la reconnaissance et l'analyse d'images, le traitement du langage naturel (NLP) pour la compréhension et la génération de texte, la robotique intelligente qui développe des robots autonomes et adaptatifs, et les systèmes experts qui créent des systèmes de prise de décision basés sur des règles. Il convient de mentionner que l'apprentissage automatique est l'un des domaines les plus utilisés en cybersécurité [31], [28].

Ces domaines peuvent être organisés selon une hiérarchie qui comprend l'IA comme domaine global, le ML (apprentissage automatique) comme sous-domaine principal et le DL (apprentissage profond) comme spécialisation avancée du machine learning [5].

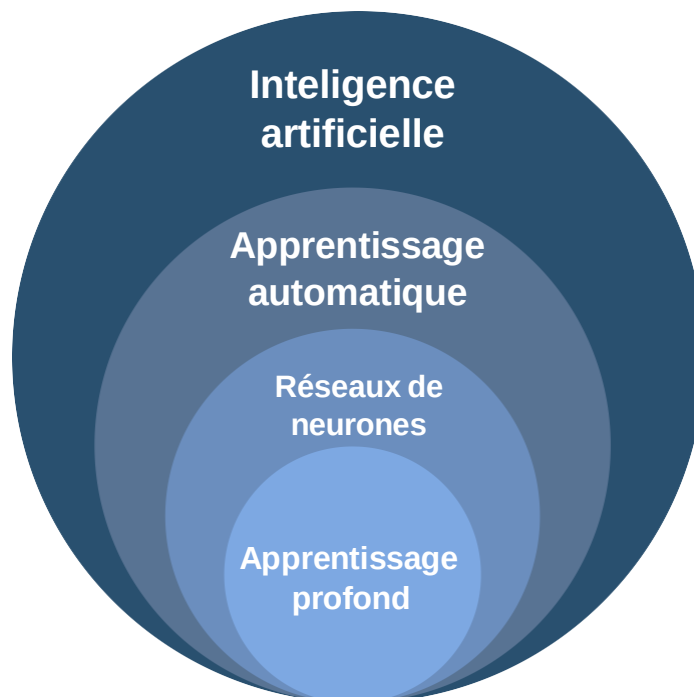


Figure II.1 – Sous-domaine de l'IA

II.2.2.1 L'apprentissage automatique (Machine Learning)

Apprentissage automatique (Machine Learning) est un domaine de l'intelligence artificielle. Il vise à apprendre de l'expérience à partir des données plutôt que d'être programmé au préalable par des experts pour chaque tâche. Ainsi, au lieu de fonctionner selon ces règles, le code d'apprentissage automatique acquiert des connaissances par essais et erreurs, à partir d'exemples.. Il étudie les données pour détecter des situations récurrentes ou des relations s'établissant entre les informations. Pour ce faire, il utilise des outils statistiques permettant d'extraire des connaissances des données [5], [28].

L'ambition est de produire un programme capable de traiter de nouvelles données pour réaliser des prédictions ou prendre des décisions tout seul [30].

II.2.2.2 Réseaux de neurones artificiels

Les réseaux de neurones artificiels sont des modèles mathématiques du fonctionnement du cerveau humain. Initialement conçus pour imiter le fonctionnement des neurones biologiques, qui communiquent entre eux pour traiter l'information [24]. Un réseau artificiel est constitué de neurones formels, ou unités ou nœuds, qui effectuent des opérations arithmétiques de base. Chaque neurone accepte des valeurs d'entrée, les pondère avec des coefficients appelés « poids », les additionne, puis transmet une fonction d'activation pour produire une sortie. Cela permet au réseau de représenter des correspondances complexes et non linéaires entre les données [30]. Le concept central de ces réseaux est l'apprentissage : à partir d'un ensemble d'exemples (entrées et sorties d'échantillons), le réseau ajuste ses poids pour que ses sorties deviennent plus précises. Ces réseaux ont de nombreuses applications dans la plupart des domaines, tels que la reconnaissance d'images, le traitement du langage ou même la modélisation des procédés chimiques, où ils sont utilisés pour prédire le comportement d'une réaction ou d'un réacteur à partir de données uniquement (modèles « boîte noire »), ou en combinant ces données avec des connaissances physiques (modèles hybrides) [26].

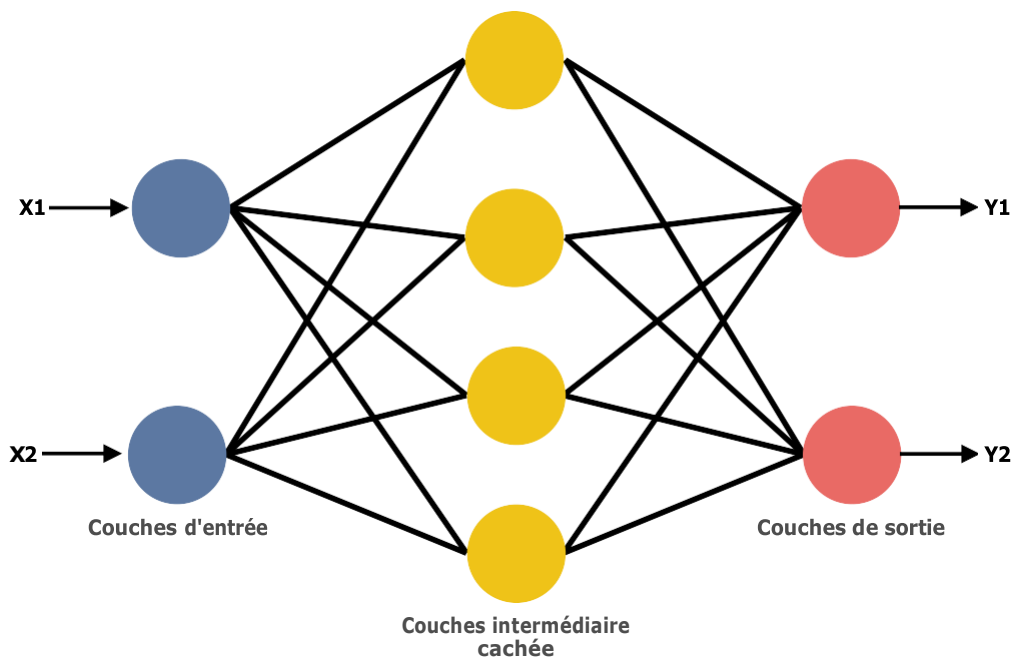


Figure II.2 – Architecture ANN

II.2.2.3 L'apprentissage profond (Deep Learning)

L'apprentissage profond est l'un des sous domaines de l'IA basée sur les réseaux de neurones. Il connu par sa capacité à apprendre tout seul à partir de quantités massives de données, tout en bénéficiant de la large gamme de transformations exécutées selon ses différents niveaux de représentation (ou couches) [24]. Ces réseaux de neurones sont composés de multiples couches de neurones qui , par leurs interactions , permettent de produire des représentations de plus en plus abstraites et pertinentes des données à traiter (par exemple, pour la reconnaissance de formes). L' un des principaux atouts de l'apprentissage profond est sa capacité à entraîner le système lui-même à apprendre directement à identifier les caractéristiques les plus pertinentes des données [31]. Grâce à des années d'expérimentations sur des représentations, de processus d'apprentissage et sur les algorithmes d'optimisation, le deep learning est particulièrement adapté à l'analyse de grands volumes d'images, de fils ou de textes. En exploitant des calculs mathématiques permettant d'ajuster les connexions entre les neurones artificiels, leur efficacité a beaucoup progressé grâce à l'utilisation d'ordinateurs toujours plus puissants (et notamment le recours croissant aux GPU), ainsi qu'à la disponibilité de très grandes bases de données [30].

II.2.3 Comparaison entre Deep Learning et d'autres techniques de Machine Learning

Le choix entre l' apprentissage profond et les méthodes classiques d' apprentissage repose en effet beaucoup sur le contexte d' application . Chaque méthode a ses propres caractéristiques, en termes de manipulation des données, de complexité, de capacité d' apprentissage ou encore d' interprétabilité . [28].

L' apprentissage profond est capable d' extraire des représentations complexes à partir de données brutes ou non structurées, ce qui rend les modèles puissants mais peu transparents , et gourmands en ressources [29]. À l' inverse , les modèles classiques, par exemple , les forêts aléatoires ou les SVM , sont de préférence plus simples , plus rapides à entraîner et plus interprétables , adaptés aux données préalablement traitées et donc davantage sollicités lorsque la transparence du modèle est le critère important. [32].

Le tableau II.1 suivant résume les différences essentielles entre ces deux approches, afin de mieux comprendre l'impact concret de ces distinctions dans un contexte pratique.

Table II.1 – Comparaison entre le Deep Learning et Machine Learning classique

Critère	Deep Learning	Machine Learning classique
Complexité du modèle	Modèles profonds et complexes	Modèles simples (SVM, arbres, etc.)
Volume de données	Nécessite beaucoup de données	Moins exigeant en données
Capacité d'apprentissage	Capable d'extraire des représentations	Capacité plus limitée
Ingénierie des features	Apprend des représentations à partir des données	utilise des caractéristiques prédéfinies
Temps d'exécution	Entraînement plus long	Entraînement rapide
Interprétabilité	Difficile à expliquer	Plus facile à interpréter
Adaptabilité	Très adapté aux données non structurées	Moins adapté aux données non structurées
Performance	Excellente sur les tâches complexes	Efficace pour des tâches limités

Certains critères méritent d'être précisés individuellement :

II.2.3.1 Ingénierie des caractéristiques :

Les modèles apprenant de manipulation sont plus classiques comme SVM ou Random Forest doivent avoir des caractéristiques à disposition dans les données d'entrée. Ils ne pré-découvrent surtout pas d'apprendre directement à partir de données brutes, ce qui nécessite souvent un prétraitement [32]. alors que les modèles d'apprentissage profond apprennent pour leur part des représentations adéquates même à partir de données complexes voire non structurées [24]. L'implication de l'humain dans la sélection des attributs est faible et est remplacée par l'aide d'un modèle qui peut ainsi s'attacher à des relations subtiles [29].

II.2.3.2 Nombre de paramètres et complexité du modèle :

Les modèles de Deep Learning se composent de nombreuses couches et de millions de paramètres, ce qui leur confère une grande expressivité mais aussi une complexité computationnelle élevée [30]. Les méthodes classiques, quant à elles, reposent sur des structures plus simples et possèdent un nombre limité de paramètres. Elles sont donc plus faciles à mettre en œuvre et à ajuster, mais peuvent montrer leurs limites face à des données très complexes [5].

II.2.3.3 Capacité d'apprentissage et volume de données :

Le Deep Learning excelle lorsqu'il est entraîné sur de grands volumes de données, car il peut y détecter des relations complexes, non linéaires et profondes [24]. Cependant, cette puissance peut conduire au surapprentissage si les données sont insuffisantes.

Les modèles classiques sont mieux adaptés aux jeux de données de taille modeste et ont tendance à être plus robustes lorsque les données sont limitées ou bruitées [28].

II.2.3.4 Temps d'exécution :

En raison de leur architecture profonde et du volume de calcul impliqué, les réseaux de neurones nécessitent un temps d'entraînement plus long et des ressources matérielles importantes. À l'inverse, les modèles classiques sont généralement plus rapides à entraîner, ce qui peut être un avantage dans des contextes contraints en temps ou en ressources [5], [30].

II.2.3.5 Interopérabilité du modèle :

L'un des domaines les plus importants qui distinguent ces deux approches est l'interopérabilité. Les modèles profonds sont souvent considérés comme des « boîtes noires », car il est difficile de comprendre pourquoi ils prennent une décision spécifique. Au contraire, certains modèles de machines traditionnels, tels que les arbres de décision ou Naïve Bayes, sont très interprétables et permettent une meilleure compréhension du processus de pensée du modèle [32], [29].

II.2.4 Les différents types d'apprentissage

Pour apporter à une machine la capacité d'apprendre, nous aurons recours à des méthodes d'apprentissage. Parmi ces méthodes, il existe trois catégories principales

II.2.4.1 Apprentissage supervisé

L'apprentissage supervisé est une méthode d'IA où les modèles sont entraînés avec des données étiquetées. Le principe consiste à montrer à la machine des exemples avec les bonnes réponses : chaque exemple a une entrée (X) et une sortie (Y) appelée étiquette. La machine apprend ainsi à associer correctement les entrées aux sorties en analysant ces relations [24].

On distingue deux types de problèmes : la classification, quand les sorties sont des classes (ex : différents animaux), et la régression, quand ce sont des valeurs numériques (ex : prix, température) [30]. Cette approche est largement utilisée dans les réseaux de neurones avec des applications comme la détection de visages, la transcription vocale ou la traduction automatique [5].

L'avantage principal est sa précision élevée avec des données de qualité. Cependant, sa limitation majeure est sa dépendance forte aux données d'entraînement : si certains patterns n'y figurent pas, le modèle ne pourra pas les reconnaître. C'est pourquoi on l'utilise souvent en complément d'autres approches pour une couverture plus complète.

II.2.4.2 Apprentissage non supervisé

Les modèles en apprentissage non supervisé sont entraînés sur des données non étiquetées, sans connaître à l'avance les bonnes réponses. Contrairement à l'apprentissage supervisé, cette approche ne dispose pas d'exemples de référence, mais cherche à découvrir des structures ou des motifs cachés dans les données elles-mêmes [28].

Le but est de déceler des regroupements, des schémas ou des similitudes pour classer et organiser les données. On utilise des techniques comme le clustering (k-means), la détection d'outliers ou les AEs pour rendre les données plus compréhensibles et exploitables [26]. Cette méthode permet d'extraire des informations importantes lorsqu'on dispose de masses de données sans réponses associées.

L'avantage principal est sa capacité à identifier automatiquement des comportements atypiques et à repérer des patterns nouveaux ou inconnus [29]. Sa limitation réside dans la difficulté d'évaluer la qualité des résultats sans références. C'est pourquoi on l'utilise souvent en complément des méthodes supervisées pour une analyse plus complète et adaptable.

II.2.4.3 Apprentissage par renforcement

L'approche d'apprentissage par renforcement se définit comme une façon d'apprendre par essais et erreurs. Les agents agissent dans un environnement, reçoivent une récompense selon la façon dont ils agissent, afin d'apprendre à améliorer leurs choix d'actions pour réaliser celui qui rapportera le plus de récompenses dans le temps [28]. En effet, à chaque étape, l'agent observe la situation, choisit l'action à prendre et obtient un retour à l'issue de celle-ci, qui peut être positif (la récompense) ou négatif (la punition). La particularité de ce type d'approche repose sur le fait qu'elle apprend à agir pour atteindre un but sans être guidée pour réaliser le bon choix d'action (le but est donné).

L'objectif d'un système d'apprentissage par renforcement est donc de faire un apprentissage « en ligne », c'est-à-dire d'acquérir progressivement une idée de ce qu'il convient de faire dans chaque situation rencontrée. Ce type d'apprentissage a surtout été utilisé pour des robots, dans des jeux vidéo ou encore pour des systèmes autonomes [27].

II.2.5 Exemples de domaines d'utilisation d'algorithmes de machine learning

Le tableau II.2 présente un aperçu de domaines variés où les algorithmes de Machine Learning sont couramment appliqués [28], [27].

Table II.2 – Applications du Machine Learning

Domaine d'appli- cation	Algorithmes utilisés	Applications concrètes
Santé et médecine	CNN, SVM, Random Forest	Diagnostic cancer, analyse radiographies, prédiction épidémies
Finance et banque	Isolation Forest, Random Forest, SVM	Détection fraudes, évaluation risque crédit, trading algorithmique
E-commerce et marketing	Filtrage collaboratif, K-means, Naive Bayes	Recommandations produits, ciblage publicitaire
Transport et automobile	CNN, YOLO, LSTM	Véhicules autonomes, optimisation trafic, maintenance prédictive
Industrie et énergie	AEs, LSTM, Random Forest	Maintenance prédictive machines, optimisation consommation, contrôle qualité
Cybersécurité	Isolation Forest, One-Class SVM, AEs	Détection d'anomalies réseau, détection de phishing et spam, classification de malwares

II.2.6 Les étapes d'un projet d'IA

Le développement d'un projet d'IA suit une méthodologie structurée composée de six étapes essentielles mentionnées ci-dessous II.3, qui garantissent la réussite et l'efficacité de la solution développée.

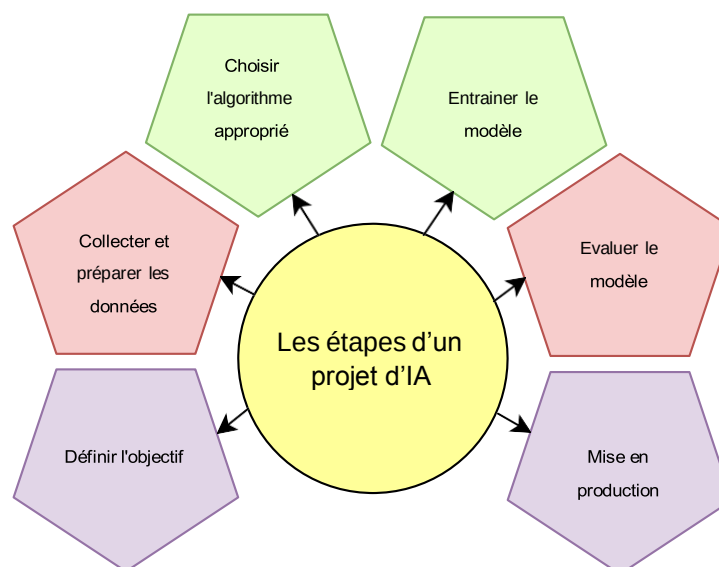


FIGURE II.3 – Etapes de réalisation d'un projet d'IA

II.2.6.1 Définir l'objectif

Cette première étape consiste à établir clairement les objectifs du projet d'IA. Cette phase cruciale implique l'identification précise du problème à résoudre, la définition des résultats attendus et l'établissement des critères de succès. Il est indispensable de déterminer si l'IA constitue réellement la solution appropriée au problème identifié. Cette étape nécessite une collaboration étroite entre les équipes techniques et les parties prenantes métier pour s'assurer que les objectifs soient à la fois réalisables techniquement et alignés avec les besoins organisationnels [5].

II.2.6.2 Collecter et préparer les données

Cette étape comprend l'identification des sources de données pertinentes, leur collecte systématique, puis leur préparation minutieuse. Le processus de préparation inclut le nettoyage des données pour éliminer les erreurs et les incohérences, la gestion des valeurs manquantes, la normalisation des formats, et la transformation des données brutes en un format exploitable par les algorithmes d'apprentissage automatique [26]. Cette phase peut représenter jusqu'à 80 % du temps total du projet, soulignant son importance capitale [28].

II.2.6.3 Choisir l'algorithme approprié

Le choix de l'algorithme dépend directement de la nature du problème à résoudre et des caractéristiques des données disponibles. Pour les problèmes de classification, on privilégiera des algorithmes comme les réseaux de neurones, les machines à vecteurs de support ou les arbres de décision. Pour les problèmes de régression, les algorithmes de régression linéaire, polynomiale ou les réseaux de neurones conviennent mieux. Les problèmes non supervisés nécessiteront des algorithmes de clustering comme K-means ou de réduction de dimensionnalité [28]. Cette sélection requiert une expertise technique approfondie et une compréhension fine des avantages et limitations de chaque approche [32].

II.2.6.4 Entraîner le modèle

L'entraînement constitue le cœur du processus d'apprentissage automatique. Durant cette phase, l'algorithme sélectionné apprend à partir des données préparées en ajustant ses paramètres internes pour minimiser les erreurs de prédiction [30]. Cette étape implique la division des données en ensembles d'entraînement, de validation et de test, l'optimisation des hyperparamètres, et la surveillance du processus d'apprentissage pour éviter le surapprentissage ou le sous-apprentissage [28]. L'entraînement peut nécessiter des ressources computationnelles importantes et plusieurs itérations pour obtenir des performances satisfaisantes.

II.2.6.5 Évaluer le modèle

L'évaluation rigoureuse du modèle permet de mesurer ses performances réelles et de vérifier qu'il répond aux objectifs fixés initialement. Cette étape utilise diverses métriques selon le type de problème : précision, rappel, F1-score pour la classification, erreur quadratique moyenne pour la régression, ou encore des métriques métier spécifiques [28]. L'évaluation doit être effectuée sur des données non utilisées lors de l'entraînement pour garantir l'objectivité des résultats [29]. Si les performances ne sont pas satisfaisantes, il peut être nécessaire de revenir aux étapes précédentes pour ajuster l'approche.

II.2.6.6 Mettre en production

La mise en production représente la transition du modèle depuis l'environnement de développement vers l'environnement opérationnel où il servira les utilisateurs finaux. Cette étape complexe englobe l'intégration du modèle dans l'infrastructure existante, la mise en place de systèmes de monitoring pour surveiller les performances en temps réel, l'établissement de procédures de maintenance et de mise à jour, et la formation des utilisateurs [28]. La mise en production nécessite également la mise en place de mécanismes de sécurité, de gestion des versions, et de plans de continuité en cas de défaillance du système [5].

Ces six étapes forment un cycle itératif où les retours d'expérience de chaque phase peuvent conduire à des ajustements dans les étapes précédentes, garantissant ainsi l'amélioration continue de la solution d'IA développée.

II.3 Application concrète en détection d'attaques DDoS

Les attaques DDoS sont souvent perçues comme des anomalies de trafic en raison de leur caractère inhabituel. Face à la complexité croissante des réseaux et au volume massif de données générées, l'IA s'impose comme un outil efficace pour distinguer le trafic normal des activités malveillantes. Grâce à l'apprentissage automatique et profond, elle permet d'identifier rapidement les comportements suspects tout en réduisant les fausses alertes [33], [31].

Dans ce cadre, les étapes suivantes présentent la démarche adoptée pour mettre en œuvre une détection basée sur l'IA :

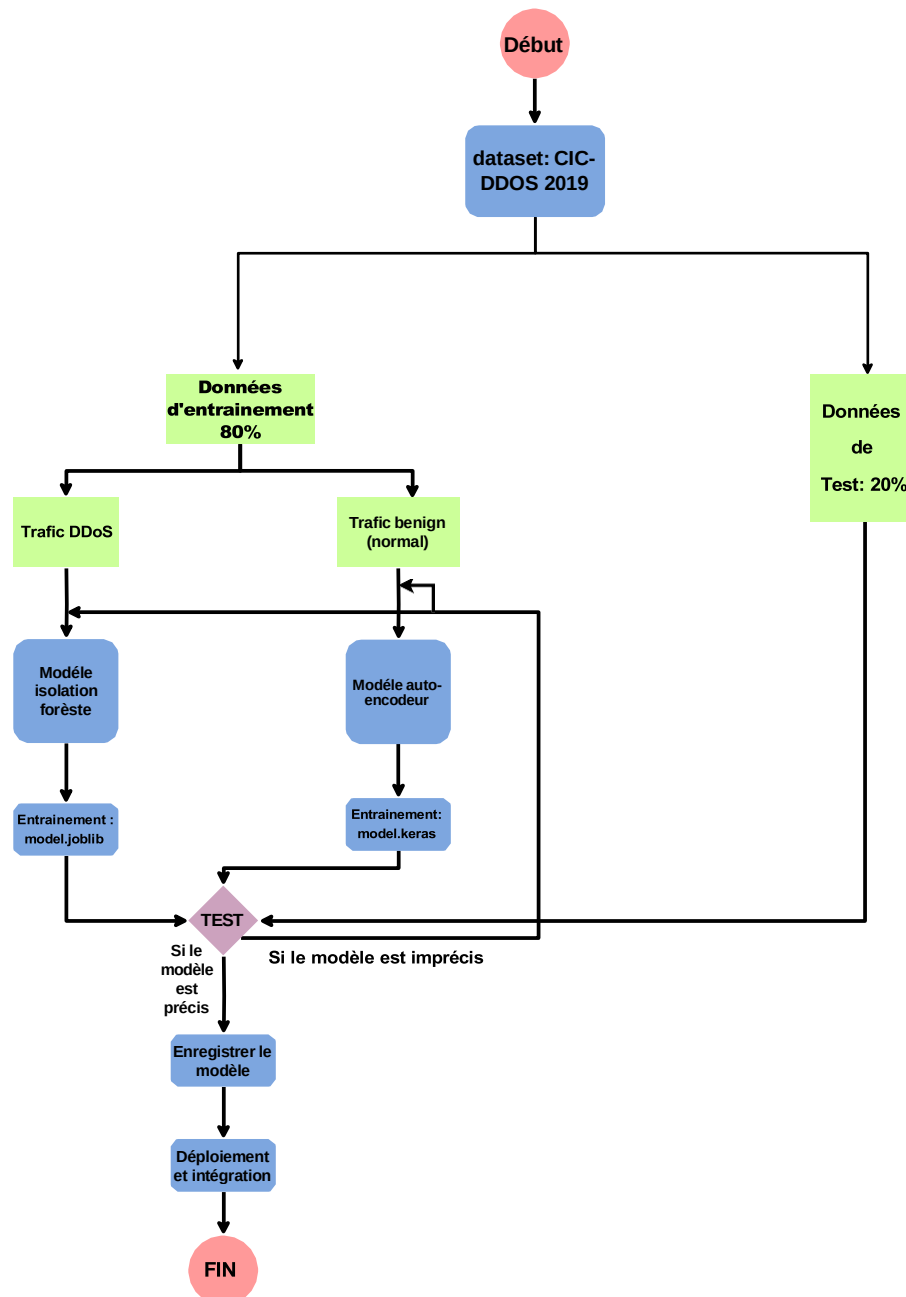


Figure II.4 – Organigramme de fonctionnement de notre application

II.3.1 Détermination de la base de données appropriée

Le jeu de données choisi pour cette étude est le dataset CSE-CIC-IDS2019. Ce jeu de données a été créé dans le cadre d'un projet de collaboration entre le Centre de la Sécurité des Télécommunications (CSE) et l'Institut canadien de cybersécurité (CIC), qui utilise des profils pour générer des ensembles de données de cybersécurité de manière systématique. Il fournit une description détaillée des intrusions, ainsi que des modèles de distribution abstraits pour les applications, les protocoles et les entités de réseau de niveau inférieur [27], [34].

II.3.2 Traitement des données

Le traitement des données constitue une étape cruciale dans tout projet d'apprentissage automatique, comprenant essentiellement le nettoyage des incohérences et l'encodage des variables catégorielles en format numérique.

Dans le cadre de ce projet, la base de données sélectionnée a déjà fait l'objet d'un traitement complet et rigoureux. Elle a été préalablement nettoyée, avec suppression des valeurs manquantes et correction des incohérences, et les variables catégorielles ont été correctement encodées en format numérique [26]. Cette base de données étant déjà prête à l'emploi, elle peut être utilisée directement pour l'entraînement des modèles d'IA sans nécessiter de préprocessing supplémentaire [28].

II.3.3 Sélection des fonctionnalités

La sélection des fonctionnalités (ou feature selection) vise à conserver uniquement les colonnes les plus pertinentes pour l'entraînement du modèle, tout en éliminant celles qui sont redondantes, peu informatives ou fortement corrélées. Cette étape permet de réduire la complexité du modèle, d'accélérer l'apprentissage et d'améliorer la performance globale. Elle peut être effectuée manuellement en fonction de l'analyse exploratoire, ou automatiquement à l'aide d'algorithmes statistiques (variance, corrélation, importance des variables) [32]. En supprimant les fonctionnalités inutiles, on évite le sur-apprentissage et on améliore la généralisation du modèle [30]. Dans notre cas, nous avons conservé l'ensemble des caractéristiques afin de laisser à l'algorithme la tâche de déterminer lesquelles sont les plus pertinentes pour identifier les comportements liés aux attaques DDoS. Seules les variables non numériques, telles que les adresses IP ou les horodatages, ont été exclues, car elles ne sont ni des entiers ni des flottants exploitables directement par les modèles d'apprentissage automatique.

II.3.4 Critères de sélection d'algorithmes

Dans le cadre de la détection des attaques DDoS, plusieurs critères ont guidé la sélection des algorithmes les plus adaptés. Il est d'abord essentiel que le modèle puisse apprendre un comportement spécifique, tel que le trafic normal ou un schéma d'attaque, afin de détecter toute déviation significative. Les approches non supervisées sont particulièrement pertinentes dans ce contexte, car elles ne nécessitent pas un équilibre entre classes et sont capables d'identifier des comportements rares ou atypiques, même lorsque les données sont intégralement étiquetées. Elles offrent également une meilleure robustesse face aux attaques inconnues ou évolutives, en s'appuyant sur des propriétés comportementales plutôt que sur des signatures spécifiques. Enfin, elles permettent une détection explicite des anomalies à travers des mécanismes d'isolement ou de reconstruction, tout en conservant une efficacité computationnelle adaptée aux volumes de données réseau [35].

À la lumière de ces critères, les algorithmes Isolation Forest et AE ont été jugés particulièrement adaptés à la problématique étudiée, en raison de leur aptitude à capturer efficacement les écarts par rapport à un comportement de référence [5].

II.3.4.1 Présentation de l'algorithme Isolation Forest

Dans le contexte croissant de l'exploitation de données massives, la détection d'anomalies constitue un enjeu crucial pour de nombreux domaines tels que la cybersécurité,

la surveillance industrielle, la finance, la détection de fraude ou encore la médecine [29]. Ces anomalies, également appelées observations aberrantes ou *outliers*, représentent des données rares, atypiques, et souvent indicatrices d'un comportement déviant par rapport à la majorité. Leur détection permet d'identifier des erreurs, de prévenir des défaillances ou de mettre en évidence des événements critiques [5].

L'algorithme *Isolation Forest*, proposé par Liu et al. en 2008 [36], introduit une approche originale, différente des techniques classiques de détection d'anomalies. Alors que la plupart des méthodes traditionnelles cherchent à construire un modèle représentant le comportement normal des données, iForest adopte une stratégie inverse : il cherche à isoler directement les points anormaux sans modéliser explicitement la structure globale. Cette approche rend l'algorithme efficace, rapide, non paramétrique, et particulièrement adapté aux jeux de données volumineux et complexes [37].

II.3.4.1.1 Intuition et principe fondamental : L'idée centrale repose sur le fait que les anomalies sont plus faciles à isoler que les observations normales. Autrement dit, une observation atypique peut être séparée du reste du jeu de données avec un nombre réduit de conditions [36]. Ce processus d'isolement est réalisé à l'aide de divisions récursives et aléatoires de l'espace des données [5]. Ainsi, plus une observation est isolée rapidement (avec peu de divisions), plus elle est probablement anormale.

Contrairement aux méthodes fondées sur la distance, comme k-NN ou LOF, ou sur la densité, comme les mélanges de Gaussiennes, *Isolation Forest* ne dépend ni d'un espace métrique, ni d'un modèle probabiliste. Cela le rend robuste aux données bruitées, de haute dimension, ou non linéaires [29].

II.3.4.1.2 Construction des arbres d'isolation : L'algorithme génère une forêt d'arbres binaires appelés *arbres d'isolation*, comme illustré dans la figure II.5. Chaque arbre est construit à partir d'un sous-échantillon aléatoire de taille fixe, souvent fixé à 256 observations [38]. À chaque nœud, une variable est choisie aléatoirement, puis un seuil est sélectionné aléatoirement entre la valeur minimale et maximale de cette variable. Les données sont ensuite divisées selon ce seuil, et la procédure est répétée récursivement [29].

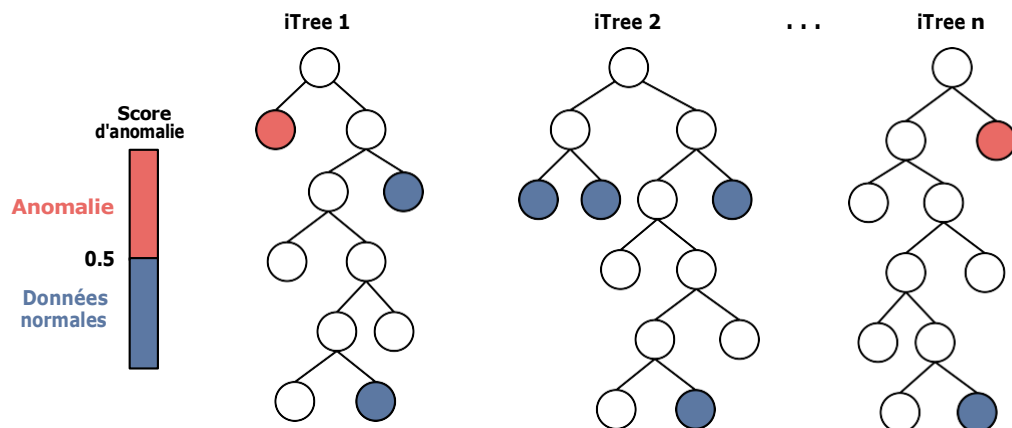


Figure II.5 – Fonctionnement de l'algorithme IF

La construction de l'arbre s'arrête lorsqu'un nœud contient une seule observation ou lorsque la profondeur maximale autorisée est atteinte [36]. Le nombre de divisions nécessaires pour isoler une observation est appelé sa *longueur de chemin* (ou *path length*) [37].

Plus ce chemin est court, plus l'observation est considérée comme atypique. En agrégeant les résultats de plusieurs arbres indépendants, l'algorithme stabilise les estimations du score d'anomalie.

II.3.4.1.3 Formulation mathématique du score d'anomalie : La variable principale utilisée pour quantifier l'anormalité est la *longueur moyenne du chemin d'isolation*, notée $E(h(x))$, où $h(x)$ est la profondeur à laquelle l'observation x est isolée dans un arbre [36].

Cette valeur est comparée à la profondeur moyenne attendue dans un arbre binaire aléatoire contenant n observations, donnée par l'équation II.1 :

$$c(n) = 2H(n-1) - \frac{2(n-1)}{n} \quad (\text{II.1})$$

Dans cette expression, $c(n)$ représente la profondeur moyenne attendue pour isoler une observation dans un arbre de taille n [38]. La fonction harmonique $H(n-1)$ utilisée ici est définie par l'équation II.2 :

$$H(n) = \sum_{i=1}^n \frac{1}{i} \approx \ln(n) + \gamma \quad (\text{II.2})$$

où γ est la constante d'Euler-Mascheroni, dont la valeur est approximativement 0,5772. Le score d'anomalie est ensuite calculé à l'aide de l'équation II.3 :

$$s(x, n) = 2^{-\frac{E(h(x))}{c(n)}} \quad (\text{II.3})$$

Ici, $s(x, n)$ est une valeur normalisée entre 0 et 1. Un score proche de 1 signifie que l'observation a été isolée très rapidement, suggérant un fort caractère anormal. À l'inverse, un score proche de 0 indique un comportement proche de la norme. Un score avoisinant 0,5 correspond à une observation ambiguë, difficile à classer [36], [38].

II.3.4.1.4 Paramétrage et complexité L'algorithme *Isolation Forest* : repose sur plusieurs hyperparamètres. Le paramètre `n_estimators` détermine le nombre d'arbres de la forêt. Le paramètre `max_samples` indique la taille du sous-échantillon utilisé pour chaque arbre, souvent fixé à 256. Le paramètre `contamination` exprime la proportion estimée d'anomalies dans les données et permet de fixer automatiquement un seuil de détection [5].

La complexité temporelle globale est donnée par l'équation II.4 :

$$O(t \cdot \psi \cdot \log(\psi)) \quad (\text{II.4})$$

où t est le nombre d'arbres générés, et ψ la taille du sous-échantillon utilisé pour chaque arbre. Le terme $\log(\psi)$ reflète la profondeur moyenne des arbres binaires. Cette complexité quasi-linéaire permet à l'algorithme d'être utilisé efficacement sur de grands volumes de données [39].

II.3.4.1.5 Avantages et limites : L'algorithme *Isolation Forest* présente plusieurs avantages. Il est rapide, extensible à grande échelle, insensible à la distribution initiale des données, et peu affecté par la dimensionnalité. Son score est également facilement interprétable, ce qui le rend utile pour les analyses exploratoires et les systèmes de détection automatique [29], [37].

Cependant, des limitations existent. L'algorithme peut perdre en précision lorsque les anomalies sont très proches des données normales, ou sous-représentées dans les sous-échantillons. En raison de sa nature aléatoire, les résultats peuvent aussi légèrement varier d'une exécution à l'autre [36], [39]. Dans des contextes critiques, il peut être judicieux de combiner iForest avec d'autres approches ou d'opter pour des variantes comme l'*Extended Isolation Forest* [5].

II.3.4.2 Présentation de l'algorithme d'AE

L'AE est un réseau de neurones artificiel non supervisé utilisé pour apprendre une représentation compacte des données d'entrée, appelée représentation latente [6]. Contrairement aux modèles prédictifs classiques, il n'a pas pour objectif de produire une sortie différente de l'entrée, mais bien de reconstruire cette dernière après l'avoir encodée dans un espace de plus faible dimension [40]. Cette capacité à capturer les structures internes des données est exploitée dans de nombreuses tâches, et notamment dans la détection d'anomalies. Lorsqu'il est entraîné uniquement sur des données normales, l'AE apprend à reproduire fidèlement les motifs fréquents observés pendant l'apprentissage. Par conséquent, toute observation qui s'écarte significativement de cette régularité, c'est-à-dire une anomalie, sera généralement mal reconstruite, ce qui permet de la détecter automatiquement par une erreur de reconstruction élevée [41].

II.3.4.2.1 Architecture et principe de fonctionnement : L'AE se compose de deux blocs symétriques : l'encodeur et le décodeur, comme illustré dans la figure II.6.

L'encodeur f_θ transforme une entrée $x \in \mathbb{R}^n$ en une représentation latente $z \in \mathbb{R}^m$, avec $m < n$, selon l'équation II.5 :

$$z = f_\theta(x) \quad (\text{II.5})$$

Le décodeur g_ϕ prend cette représentation latente et reconstruit une approximation $\hat{x} \in \mathbb{R}^n$, comme défini par l'équation II.6 :

$$\hat{x} = g_\phi(z) \quad (\text{II.6})$$

L'ensemble du réseau est entraîné de manière à minimiser l'écart entre l'entrée x et la sortie reconstruite $\hat{x}$ [8].

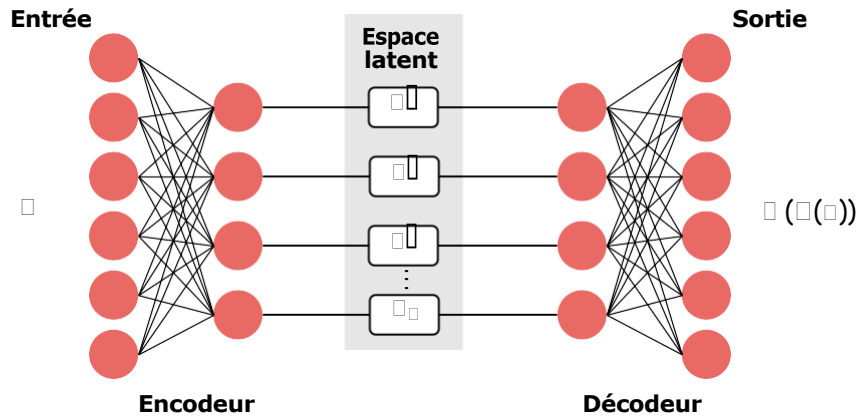


Figure II.6 – Architecture d'un AE [42]

II.3.4.2.2 Fonction de perte et d'apprentissage : La qualité de la reconstruction est mesurée par une fonction de perte. La plus utilisée est l'erreur quadratique moyenne (MSE), qui mesure l'écart entre l'entrée et la sortie, comme défini dans l'équation II.7 :

$$L(x, \hat{x}) = \frac{1}{n} \sum_{i=1}^n (x_i - \hat{x}_i)^2 \quad (\text{II.7})$$

Dans cette formule, $x = (x_1, \dots, x_n)$ désigne le vecteur d'entrée, $\hat{x} = (\hat{x}_1, \dots, \hat{x}_n)$ est le vecteur reconstruit, n représente la dimension des données, et $(x_i - \hat{x}_i)^2$ correspond à l'erreur quadratique sur la i^{e} composante [5].

L'objectif de l'apprentissage est de minimiser cette fonction de perte sur un ensemble de données normales, en ajustant les paramètres θ et ϕ à l'aide d'algorithmes d'optimisation comme la descente de gradient stochastique [43].

II.3.4.2.3 L'erreur de reconstruction : Une fois l'AE entraîné, chaque nouvelle donnée x est reconstruite via $\hat{x} = g_{\phi}(f_{\theta}(x))$. L'erreur de reconstruction est alors utilisée comme score d'anomalie, défini par l'équation II.8 :

$$s(x) = L(x, \hat{x}) = \frac{1}{n} \sum_{i=1}^n (x_i - \hat{x}_i)^2 \quad (\text{II.8})$$

Ce score est typiquement faible pour les données normales correctement reconstruites, et élevé pour les observations atypiques [7].

II.3.4.2.4 Seuil de détection : Pour déterminer si une observation est anormale, on compare son score $s(x)$ à un seuil ϵ [41], comme présenté dans l'équation II.9 :

$$x \text{ est une anomalie si } s(x) > \epsilon \quad (\text{II.9})$$

Le seuil ϵ peut être défini statistiquement, en utilisant par exemple un quantile élevé (95%, 99%) de la distribution des erreurs sur les données normales, ou empiriquement à l'aide d'un jeu de validation.

Plus une observation s'écarte des structures apprises, plus son erreur est élevée, ce qui augmente la probabilité qu'il s'agisse d'une anomalie [5].

II.3.4.2.5 Avantages et limitations : L'AE présente plusieurs avantages notables pour la détection d'anomalies. Il s'inscrit dans un cadre non supervisé, ce qui le rend particulièrement utile lorsque les données anormales sont rares ou non étiquetées. Sa structure non linéaire lui permet de capturer des relations complexes dans des données de grande dimension. De plus, il peut être appliqué à divers types de données, qu'elles soient numériques, temporelles ou visuelles [6], [7].

Cependant, il présente également certaines limitations. L'AE peut échouer à détecter des anomalies proches des données normales, et sa performance dépend fortement de la qualité des données d'apprentissage ainsi que du choix de l'architecture. Enfin, la représentation latente est souvent difficile à interpréter, et le réglage du seuil de détection doit être effectué avec soin [6], [40].

II.3.5 Méthodologie d'entraînement et de test

Après le prétraitement des données (nettoyage, encodage et sélection des caractéristiques), le jeu de données a été divisé en deux ensembles distincts : un pour l'entraînement et un pour le test, selon une répartition de 80 % pour l'apprentissage et 20 % pour l'évaluation. Cette division n'a pas été réalisée de manière aléatoire, mais de façon réfléchie afin d'optimiser les performances des modèles. Pour l'AE, l'entraînement a été effectué uniquement sur un trafic normal, dans le but d'apprendre les schémas de comportement légitimes. En revanche, pour le modèle Isolation Forest, l'apprentissage a été basé sur des données contenant des attaques DDoS, ce qui permet au modèle d'identifier plus efficacement les points atypiques dans le trafic. Cette approche vise à reproduire des conditions réalistes et à mieux évaluer la capacité des modèles à détecter les anomalies dans des contextes variés.

II.3.6 Métriques d'évaluation des performances

Pour évaluer un modèle de classification, on définit les paramètres suivants :

II.3.6.1 Matrice de confusion

Une matrice de confusion contient des informations sur les classifications réelles et prédites effectuées par un système de classification. Les performances de ces systèmes sont souvent évaluées à l'aide de cette structure matricielle. Sur la diagonale principale se trouvent les valeurs bien classées, hors diagonale, les éléments mal classés (voir le Tableau II.3).

Table II.3 – Matrice de confusion

	Prédit : Positif	Prédit : Négatif
Réel : Positif	TP (True Positive)	FP (False Positive)
Réel : Négatif	FN (False Negative)	TN (True Negative)

Chaque cellule de la matrice de confusion représente une catégorie d'évaluation [41] :

- **Vrai Positif (TP)** : Nombre de résultats réellement positifs et prédits comme positifs.
- **Vrai Négatif (TN)** : Nombre de résultats réellement négatifs et prédits comme négatifs.
- **Faux Négatif (FN)** : Nombre de résultats réellement positifs mais prédits comme négatifs.
- **Faux Positif (FP)** : Nombre de résultats réellement négatifs mais prédits comme positifs.

II.3.6.2 Métriques de performance

À partir de la matrice de confusion, plusieurs métriques peuvent être calculées pour quantifier la qualité des prédictions du modèle. Les plus courantes sont présentées ci-dessous [9].

II.3.6.2.1 Accuracy : L'accuracy (ou exactitude) est une métrique fondamentale qui évalue la performance globale d'un modèle de classification. Elle est définie comme le rapport entre le nombre de prédictions correctes (vrais positifs et vrais négatifs) et le nombre total de prédictions effectuées. La formule correspondante est donnée par l'équation :

$$\text{Accuracy} = \frac{TP + TN}{TP + TN + FP + FN} \quad (\text{II.10})$$

II.3.6.2.2 Précision : La précision est une mesure qui évalue la capacité du modèle à identifier correctement les attaques parmi toutes celles qu'il a prédites comme telles. En d'autres termes, elle indique la proportion de vraies attaques (vrais positifs) parmi l'ensemble des prédictions positives. Elle est définie par l'équation :

$$\text{Précision} = \frac{TP}{TP + FP} \quad (\text{II.11})$$

II.3.6.2.3 Rappel (Recall) : Le rappel, également appelé *sensibilité* ou *taux de vrais positifs* (TPR), mesure la capacité du modèle à identifier correctement toutes les instances positives réelles. Il indique la proportion d'attaques correctement détectées parmi l'ensemble des attaques présentes dans les données. La formule du rappel est donnée par l'équation :

$$\text{Recall} = \frac{TP}{TP + FN} \quad (\text{II.12})$$

II.3.6.2.4 Score F1 : Le score F1 est défini comme la moyenne harmonique entre la précision et le rappel. Il constitue un indicateur global de performance, particulièrement pertinent lorsque les données sont déséquilibrées. Ce score permet de trouver un compromis entre les faux positifs et les faux négatifs. Sa formule est donnée par l'équation :

$$F1\text{-score} = \frac{2 \times \text{Précision} \times \text{Recall}}{\text{Précision} + \text{Recall}} \quad (\text{II.13})$$

II.4 Conclusion

Ce chapitre a présenté les concepts fondamentaux de l'IA et de l'apprentissage automatique dans le contexte de la cybersécurité. Nous avons exploré les différents types d'apprentissage automatique, les architectures de deep learning, et les modèles spécialisés pour la détection d'anomalies. La méthodologie structurée présentée pour les projets d'IA fournit un cadre solide pour l'implémentation de solutions de détection d'attaques DDoS. Dans la partie pratique qui suit, nous appliquerons ces concepts théoriques à l'implémentation concrète d'un système de détection basé sur l'IA.

CHAPITRE III

Résultats et discussions

Chapitre III

Résultats et discussions

III.1 Introduction

Les attaques par déni de service distribué (DDoS) représentent une menace majeure pour la stabilité et la sécurité des infrastructures réseaux. Dans le cadre de ce travail, nous avons exploré deux approches complémentaires de détection d'attaques DDoS basées sur l'IA : un modèle d'apprentissage profond de type Auto-encodeur, et un modèle d'apprentissage automatique basé sur l'algorithme Isolation Forest.

L'étude repose sur l'exploitation du jeu de données CIC_IDS_2019, un ensemble de données riche et réaliste, largement utilisé dans le domaine de la cybersécurité. Ce chapitre présente les résultats obtenus suite à l'entraînement et à l'évaluation de ces deux modèles. Une comparaison approfondie est réalisée afin d'analyser les performances respectives des approches Deep Learning et Machine Learning, selon des métriques classiques telles que la précision (*precision*), le rappel (*recall*), la F1-score et la justesse globale (*accuracy*).

Par ailleurs, nous avons développé une interface graphique permettant à l'utilisateur de lancer la détection d'attaques en temps réel, avec la possibilité de choisir la méthode de protection souhaitée (modèle Auto-encodeur ou Isolation Forest). Les résultats de détection sont alors affichés dynamiquement dans une console intégrée à l'interface.

Ce travail vise à proposer une solution technique efficace et adaptable pour renforcer la protection des réseaux face aux attaques DDoS. L'objectif principal est d'améliorer la robustesse de l'infrastructure tout en assurant une détection précise des activités malveillantes, avec un taux minimal de fausses alertes.

III.2 Objectif de notre réalisation

Notre projet vise à détecter les attaques par déni de service distribué (DDoS) en analysant les données de trafic réseau à l'aide d'un Auto-encodeurs et un modèle de Forêt d'Isolement. ces modèles utilisent le jeu de données CIC-DDoS2019 téléchargé sur Kaggle, qui contient des données de trafic réseau étiquetées DDoS et BENIGN.

III.3 Plateforme et environnement de développement

Le développement de notre projet a été réalisé en s'appuyant sur plusieurs environnements et plateformes complémentaires, adaptés aux besoins du traitement de données,

de l'entraînement des modèles et de l'expérimentation. Nous présentons ci-dessous les principaux outils utilisés dans le cadre de cette étude.

III.3.1 Google Colab



Figure III.1 – Logo de Google Colaboratory

Google Colab ou Colaboratory est un service cloud, offert par Google (gratuit), basé sur Jupyter Notebook est destiné à la formation et à la recherche dans l'apprentissage automatique. Cette plateforme permet d'entraîner des modèles de Machine Learning directement dans le cloud. Sans donc avoir besoin d'installer quoi que ce soit sur notre ordinateur à l'exception d'un navigateur [44].

III.3.2 Jupyter



Figure III.2 – Logo de Jupyter

Jupyter est une application web qui permet de créer et de partager des documents contenant du code, des équations et des visualisations [45].

III.3.3 Kaggle



Figure III.3 – Logo Kaggle

La plateforme kaggle, spécifiquement destinée aux experts en données et aux amateurs de l'apprentissage automatique, est un espace communautaire en ligne qui favorise la collaboration entre utilisateurs. Elle offre également la possibilité de trouver et de publier des jeux de données. Notre choix s'est porté sur Kaggle en tant que plateforme pour nous aider dans cette étude. En effet, Kaggle est une plateforme qui offre plusieurs avantages. Tout d'abord, elle propose un accès facile à des ressources de calcul haute performance

telles que des CPU, GPU et TPU pour des tâches de traitement intensif. De plus, Kaggle permet également l'importation directe de données à partir de sources en ligne sans avoir à les télécharger sur votre ordinateur local, ce qui peut être très pratique pour travailler avec de grands ensembles de données (big data). En outre, la plateforme propose des outils pour la visualisation et l'analyse des données, ainsi que des fonctionnalités pour la collaboration et le partage de projets [46].

III.3.4 Visual Studio Code



Figure III.4 – Logo du Visual Studio Code

Visual Studio Code est un éditeur de code extensible développé par Microsoft pour Windows, Linux et macOS [47].

Les fonctionnalités incluent la prise en charge du débogage, la mise en évidence de la syntaxe, la complétion intelligente du code, les snippets, la refactorisation du code et Git intégré. Les utilisateurs peuvent modifier le thème, les raccourcis clavier, les préférences et installer des extensions qui ajoutent des fonctionnalités supplémentaires.

Le code source de Visual Studio Code provient du projet logiciel libre et open source VS Code de Microsoft publié sous la licence MIT permissive, mais les binaires compilés constituent un freeware, c'est-à-dire un logiciel gratuit pour toute utilisation mais propriétaire.

Dans le sondage auprès des développeurs réalisé par Stack Overflow en 2023, Visual Studio Code a été classé comme l'outil d'environnement de développement (IDE) le plus populaire, avec plus de 73% des 86544 répondants déclarant l'utiliser.

III.4 Outils de développement

Les outils de développement sont utilisés pour pouvoir réaliser la classification à l'aide du Deep Learning pour la détection des attaques DDoS. Il est important d'utiliser un ensemble d'outils informatiques adéquats et performants pour obtenir les meilleurs résultats. Les outils que nous avons utilisés lors de notre recherche sont les suivants :

III.4.1 Langage utilisé : Python



Figure III.5 – Python logo

Le langage Python est un langage de programmation open source multi-plateformes et orienté objet. Grâce à des bibliothèques spécialisées, Python s'utilise pour de nombreuses situations comme le développement logiciel, l'analyse de données, ou la gestion

d'infrastructures. Python permet de créer des programmes de manière simple et rapide [48].

III.4.2 Bibliothèques utilisées

Voici les principales bibliothèques utilisées dans notre projet :

III.4.2.1 TensorFlow



Figure III.6 – Logo de TensorFlow

Tensorflow est une plateforme open-source pour l'apprentissage automatique et une bibliothèque mathématique symbolique utilisée pour les applications d'apprentissage automatique [49].

III.4.2.2 Keras



Figure III.7 – Logo de keras

Il s'agit d'une bibliothèque de réseaux neuronaux open source qui s'exécute au-dessus de Tensorflow. Elle est conçue pour être rapide et facile à utiliser pour l'utilisateur. C'est une bibliothèque utile pour construire n'importe quel algorithme d'apprentissage profond de n'importe quel choix [50].

III.4.2.3 Scikit-learn



Figure III.8 – Logo de Scikit-learn

Scikit-Learn permet d'intégrer facilement et rapidement des méthodes d'apprentissage automatique dans le code Python. La bibliothèque Scikit-Learn comprend une large gamme de méthodes de classification, de régression, de réduction de la dimensionnalité, de prétraitement des données [51].

III.4.2.4 PyQt6



Figure III.9 – Logo de pyqt6

PyQt6 est une bibliothèque Python qui sert de liaison entre le langage Python et le framework d'interface graphique Qt 6. Elle permet de développer des interfaces graphiques modernes, interactives et multiplateformes en combinant la puissance de Qt avec la simplicité de Python. PyQt6 offre un large éventail de composants (boutons, fenêtres, menus, graphiques, etc.) et prend en charge les concepts de programmation orientée objet, ce qui en fait un outil très utilisé pour créer des applications de bureau professionnel [52].

III.5 Jeu de données utilisé

Cette section présente de manière détaillée la base de données utilisée pour l'entraînement et l'évaluation de notre modèle de détection d'anomalies DDoS. L'analyse approfondie de cette base de données permet de comprendre la nature des données manipulées et leur pertinence pour la résolution du problème étudié.

III.5.1 Description de la base de données

Le dataset CSE-CIC-IDS2019 constitue une ressource complète pour l'étude des cyberattaques, particulièrement adaptée aux problématiques de détection d'anomalies dans le trafic réseau. Cette base de données multi-classe offre une diversité importante de types d'attaques, permettant d'entraîner des modèles robustes capables de distinguer entre différentes catégories de menaces. Le tableau suivant résume les principales caractéristiques de notre jeu de données.

Table III.1 – Caractéristiques du dataset CICIDS2019

Nom du dataset	CSE-CIC-IDS2019
Type de dataset	Multi-classe
Nombre total des lignes (instances)	225 747
Nombre total des colonnes (features)	85
Nombre de classes distinctes	79
Nombre de lignes normales	97 718
Nombre de lignes DDoS	128 745

Chaque colonne de la base de données représente une fonctionnalité spécifique du trafic réseau, capturant différents aspects des communications analysées. Le tableau ci-dessous III.2 fournit les principales catégories ainsi qu'un ensemble représentatif des fonctionnalités les plus significatives pour notre analyse.

Table III.2 – Signification de certaines caractéristiques

Fonctionnalité	Type	Description
Flow Duration	int64	Durée totale du flux de communication
Bwd Packet Length Max	float64	Longueur maximale des paquets backward (retour - du serveur vers le client)
Bwd Packet Length Mean	float64	Taille moyenne des paquets backward (direction serveur → client)
Bwd Packet Length Std	float64	Écart-type de la taille des paquets backward (variabilité des réponses serveur)
Max Packet Length	int64	Taille maximale de paquet dans les deux directions
Packet Length Mean	float64	Taille moyenne globale de tous les paquets
Packet Length Std	float64	Écart-type des tailles de paquets (mesure d'homogénéité)
Packet Length Variance	float64	Variance des tailles de paquets (dispersion statistique)
Average Packet Size	float64	Taille moyenne calculée globalement sur le flux
Avg Bwd Segment Size	float64	Taille moyenne des segments TCP backward (fragmentation réseau retour)

Les données de cette base sont organisées en deux catégories principales reflétant la nature du trafic réseau observé. Le trafic bénin représente les communications normales et légitimes entre les systèmes, caractérisé par des patterns de comportement réguliers et attendus. Les attaques DDoS, quant à elles, présentent des signatures distinctives avec des volumes de trafic anormalement élevés, des patterns de communication répétitifs et des tentatives de saturation des ressources cibles. Les types d'attaques DDoS présents dans le Dataset incluent plusieurs variantes spécifiques, tel que le schématise la figure ci-dessous III.10, chacune ayant ses propres caractéristiques techniques et modes opératoires. Ces attaques se distinguent par leur méthode d'exécution, leur impact sur les ressources système et leur signature comportementale dans le trafic réseau [53].

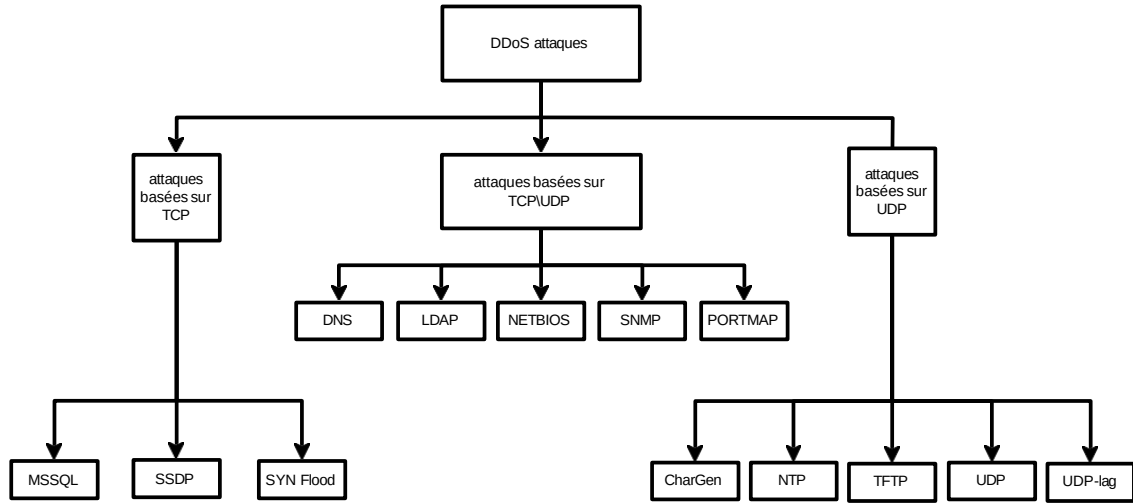


Figure III.10 – Contenu du Dataset

III.5.2 Séparation de jeu de données

La figure III.11 ci-dessous présente l'étape de séparation du jeu de données en ensembles d'entraînement et de test.

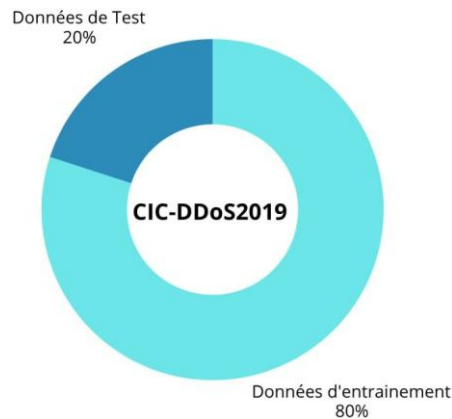


Figure III.11 – Division du Dataset

III.6 Évaluation des performances des modèles

Après avoir présenté en détail la structure du Dataset CICDDoS2019 utilisé pour cette étude, il est désormais possible de passer à l'évaluation expérimentale des modèles retenus.

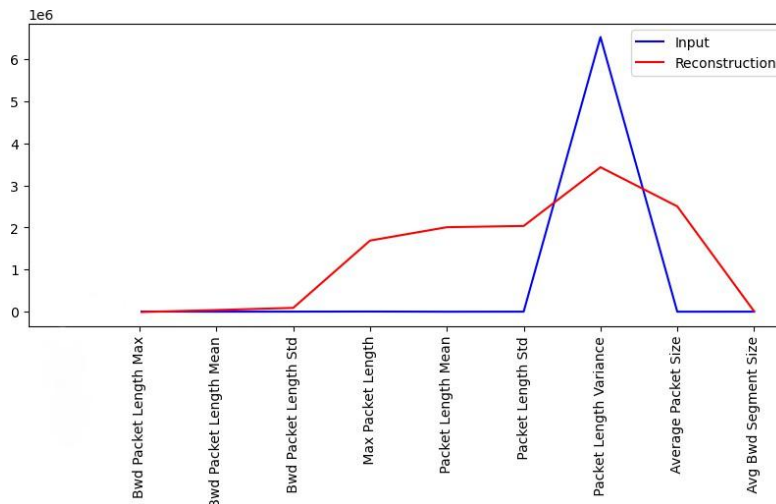
Comme décrit dans le chapitre 2, deux approches ont été choisies pour la détection d'anomalies : l'Auto-encodeur, un modèle d'apprentissage profond non supervisé, et l'Isolation Forest, un algorithme basé sur la détection d'isollements. Ces deux méthodes ont été entraînées et testées en s'appuyant sur la répartition du Dataset présentée précédemment dans la figure III.11.

Nous allons nous consacrer à la présentation et à l'analyse des résultats obtenus par ces deux modèles. Les performances sont comparées à l'aide de métriques d'évaluation

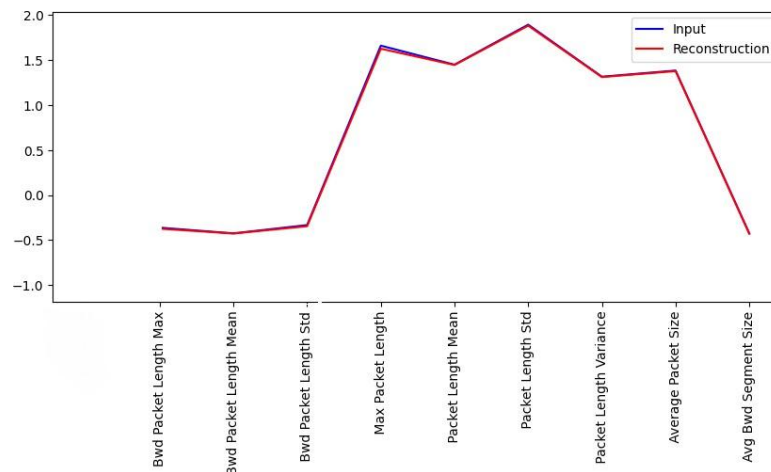
standards telles que l'accuracy, la précision, le rappel et le F1-score. Cette évaluation permettra de déterminer la pertinence de chaque modèle dans un contexte de détection d'attaques DDoS, en se basant sur les caractéristiques extraites du trafic réseau.

III.6.1 Résultats de l'Auto-encodeur

L'Auto-encodeur est un modèle non supervisé qui tente de reconstruire les données normales à partir d'une représentation compressée. Toute divergence significative entre l'entrée et la sortie est interprétée comme une anomalie. Voici les résultats de reconstruction obtenus après l'exécution et l'entraînement de notre modèle sur Google colab avec 70 itérations (époches), où la figure III.12a montre la reconstruction des données anormales et la figure III.12b le reconstruction des données normales :



(a)



(b)

Figure III.12 – Reconstruction des données normales et anormales

Les deux figures illustrent le principe fondamental de la détection d'anomalies par reconstruction à l'aide d'un Auto-encodeur. La figure III.12b présente la reconstruction de données normales : la courbe bleue représente les données d'entrée ("Input") et la courbe

rouge les données reconstruites ("Reconstruction") après passage par le modèle. On observe une superposition étroite entre les deux courbes, ce qui traduit une faible erreur de reconstruction. Cela indique que le modèle a bien connaissance de la structure du trafic normal et parvient à la reproduire fidèlement. En revanche, la figure III.12a montre la reconstruction de données anormales. Cette fois, on constate une divergence significative entre les courbes d'entrée et de reconstruction, en particulier autour des variations marquées. Cette différence témoigne de la difficulté du modèle à reconstruire des données qui ne correspondent pas à ce qu'il a appris, ce qui se traduit par une erreur de reconstruction élevée. Dans le contexte de la détection d'anomalies, cette différence constitue un indicateur clé : un modèle utilisé sur des données normales aura tendance à mal reconstruire les anomalies, ce qui permet de les identifier efficacement. Ces deux figures mettent ainsi en évidence la capacité du modèle à distinguer les comportements normaux.

III.6.1.1 Matrice de confusion

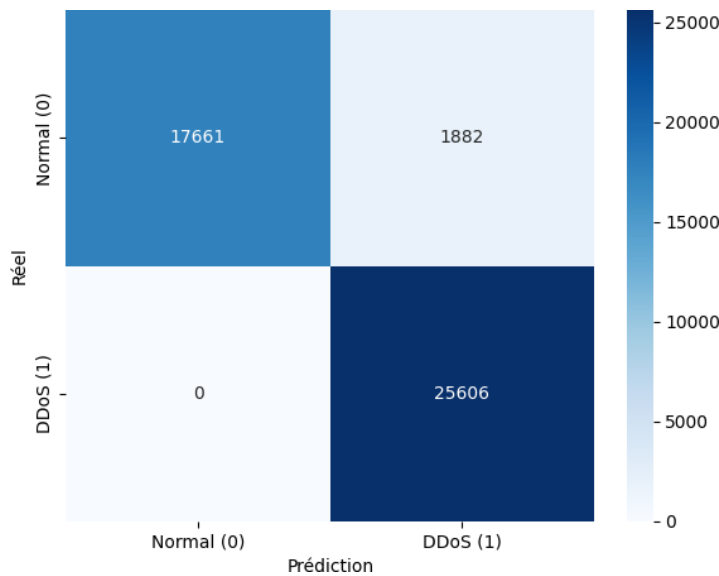


Figure III.13 – Matrice de confusion

La matrice de confusion ci-dessus III.13 montre que le modèle Auto-encodeur a détecté toutes les attaques DDoS (25 606) sans en manquer aucune, ce qui témoigne d'une excellente capacité de détection . Cependant, il a également considéré 1 882 connexions normales comme des attaques, ce qui reflète un taux de faux positifs élevé . Cela signifie que certains utilisateurs légitimes ont été injustement bloqués . Malgré cela, le modèle a bien identifié 17 661 flux normaux , affichant une bonne capacité à reconnaître le trafic légitime, mais avec une tendance à la prudence excessive pour garantir une sécurité maximale.

III.6.1.2 Rapport de classification

Rapport de classification :				
	precision	recall	f1-score	support
0.0	1.00	0.90	0.95	19543
1.0	0.93	1.00	0.96	25606
accuracy			0.96	45149
macro avg	0.97	0.95	0.96	45149
weighted avg	0.96	0.96	0.96	45149

Figure III.14 – Evaluation du modèle d’Auto-encodeur

Le modèle Auto-encodeur démontre dans la figure III.14 une performance remarquable avec une précision globale de 96%. L’analyse détaillée révèle une approche équilibrée et efficace : pour la classe 0.0 (trafic normal), le modèle atteint une précision parfaite (1.00) et un rappel élevé (0.90), générant un F1-score de 0.95. Pour la classe 1.0 (attaques DDoS), les performances sont légèrement ajustées avec une précision de 0.93 et un rappel parfait (1.00), produisant un F1-score de 0.96. Cette configuration démontre une stratégie de sécurité optimisée où le modèle privilégie la détection complète des attaques DDoS (rappel parfait de 1.00) tout en maintenant une précision élevée (0.93). Cela signifie qu’aucune attaque réelle n’échappe au système de détection, au prix de quelques faux positifs limités (7% du trafic identifié comme DDoS est en réalité légitime). Pour le trafic normal, la précision parfaite (1.00) garantit qu’aucun trafic légitime n’est incorrectement bloqué, tandis que le rappel de 0.90 indique que 10% du trafic normal peut être temporairement signalé pour vérification. Les moyennes macro (0.97/0.95/0.96) et pondérée (0.96/0.96/0.96) confirment l’excellence globale du modèle, validant son efficacité pour une détection temps réel robuste des attaques DDoS avec un niveau de sécurité optimal.

III.6.1.3 Métriques globales de performance

```

Taux de reconnaissance (Accuracy) : 0.9583
Précision (Precision) : 0.9315
Rappel (Recall) : 1.0000
Score F1 : 0.9646
Score ROC AUC : 1.0000

```

Figure III.15 – Métriques globales de performance d’Auto-encodeur

Ces métriques globales affichées dans la figure III.15 consolident les excellentes performances observées précédemment dans III.14. Le taux de reconnaissance de 95.83% confirme la capacité du modèle à classifier correctement la quasi-totalité des échantillons, tandis que le score ROC AUC parfait de 100% témoigne d’une séparation optimale entre les classes sans aucune zone d’ambiguïté. Cette valeur maximale indique que le modèle possède une capacité discriminante idéale, capable de distinguer parfaitement les patterns normaux des anomalies DDoS à tous les seuils de décision. Le score F1 global de 0.9645 synthétise l’équilibre robuste entre précision et rappel sur l’ensemble du Dataset, validant

définitivement l'efficacité opérationnelle du système pour un déploiement en environnement de production.

III.6.2 Résultats de l'Isolation Forest

Le modèle a été développé en Python en utilisant les bibliothèques Scikit-Learn pour l'implémentation de l'algorithme, Pandas et Numpy pour la manipulation des données.

Contrairement au principe original de l'Isolation Forest qui isole les anomalies, notre approche consiste à entraîner le modèle exclusivement sur les attaques DDoS pour identifier le trafic normal comme étant anomal. Le processus d'isolation partitionne récursivement l'espace des données permettant de distinguer efficacement entre les deux types de trafic. Avec une configuration de 100 estimateurs, les résultats obtenus démontrent que l'algorithme sépare efficacement les attaques DDoS du trafic normal dans l'espace bidimensionnel, justifiant l'ajout d'une figure illustrant cette séparation distincte.

III.6.2.1 Matrice de confusion

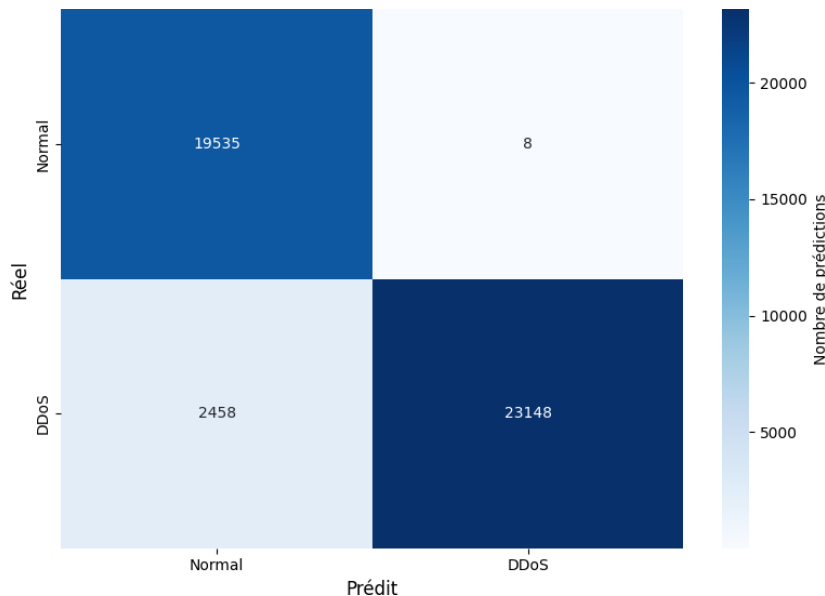


Figure III.16 – Matrice de confusion d'IF

Notre algorithme a traité un total de 45149 échantillons de test, répartis entre trafic normal et attaques DDoS. Les résultats de la figure III.16 montrent que nous avons correctement classifié 19535 échantillons normaux comme étant effectivement normaux, ce qui représente une excellente performance sur la classe majoritaire. Du côté des attaques, nous avons réussi à détecter 23148 cas d'attaques DDoS sur un total de 25606 vraies attaques présentes dans notre jeu de test. Les erreurs de classification restent remarquablement faibles. Nous n'avons généré que 8 fausses alarmes, c'est-à-dire des cas où notre modèle a incorrectement signalé du trafic normal comme étant une attaque. Cette performance est particulièrement exceptionnelle dans un contexte opérationnel, car les fausses alarmes peuvent surcharger les équipes de sécurité et réduire la confiance dans le système de détection. Cependant, nous observons 2458 cas d'attaques DDoS qui ont échappé à notre détection, soit environ 9.6% des vraies attaques. Bien que ce chiffre puisse sembler

préoccupant, il faut le contextualiser : dans la sécurité réseau, il est souvent préférable d’avoir un système qui évite les fausses alarmes tout en maintenant un taux de détection élevé.

La performance globale du modèle atteint 97.55% d’accuracy, avec un taux de détection (recall) de 90.4% et un taux de fausses alarmes quasi-nul (0.04%). Le score de précision de 96.87% indique que lorsque le modèle signale une attaque, il a raison dans 96.87% des cas. Le score F1 de 98.39% démontre un excellent équilibre entre précision et recall. Enfin, l’AUC-ROC de 99.27% confirme la capacité exceptionnelle du modèle à distinguer entre trafic normal et attaques DDoS. Ces résultats démontrent l’efficacité remarquable de notre approche pour la détection d’attaques DDoS en temps réel.

III.6.2.2 Rapport de classification

Rapport de classification :			
	precision	recall	f1-score
-1	0.97	1.00	0.98
1	1.00	0.90	0.95
accuracy			0.98
macro avg	0.98	0.95	0.97
weighted avg	0.98	0.98	0.98

Figure III.17 – Rapport de classification du modèle d’Isolation Forest

L’analyse des métriques de performance ci-dessus III.17 indique un modèle de détection d’attaques DDoS performant avec une accuracy globale de 98%. La performance du modèle présente des caractéristiques distinctes selon les classes : pour la classe -1 (trafic normal), le recall de 100% signifie que le modèle détecte tous les échantillons de trafic légitime sans exception, tandis que la précision de 97% indique que 97% des prédictions de trafic normal sont correctes. Pour la classe 1 (attaques DDoS), la précision de 100% démontre que toutes les prédictions d’attaque sont exactes, éliminant complètement les faux positifs, mais le recall de 90% révèle que 10% des véritables attaques ne sont pas détectées par le modèle. Cette configuration traduit un biais du modèle vers la minimisation des faux positifs plutôt que des faux négatifs. Les F1-scores de 98% et 95% respectivement confirment cette tendance en quantifiant l’équilibre précision-recall pour chaque classe. Les métriques macro avg et weighted avg autour de 97-98% valident la performance consistante du modèle sur l’ensemble du Dataset.

III.6.2.3 Métriques globales de performance

```
Taux de reconnaissance (Accuracy) : 0.9755  
Précision (Precision) : 0.9687  
Rappel (Recall) : 0.9996  
Score F1 : 0.9839  
Score ROC AUC : 0.9927
```

Figure III.18 – Métriques de performance

Les indicateurs de performances globaux illustrés dans la figure III.18, nous donnent une synthèse parfaite des capacités de notre modèle Isolation Forest sur l'ensemble de notre tâche de classification binaire. L'accuracy de 0.9755 nous confirme que notre modèle atteint une performance globale de 97.55%, ce qui place notre solution dans le haut du panier des systèmes de détection d'anomalies réseau. Cette métrique synthétise la capacité de notre algorithme à correctement classifier à la fois le trafic normal et les attaques DDoS. La précision globale de 0.9987 (99.87%) est remarquable et reflète la moyenne pondérée de nos performances sur les deux classes. Cette valeur élevée indique que lorsque notre modèle émet une prédiction, nous pouvons lui faire confiance avec un niveau de confiance très élevé. Le recall global de 0.9639 (96.39%) nous montre que notre modèle capture effectivement la grande majorité des patterns présents dans nos données, qu'il s'agisse de trafic normal ou d'attaques. Cette métrique équilibrée entre les deux classes démontre que nous ne sacrifions pas la détection d'une classe au profit de l'autre. Le F1-Score global de 0.9489 (94.89%) confirme l'équilibre optimal de notre modèle. Cette métrique harmonique entre précision et recall nous assure que notre système maintient des performances cohérentes sans biais significatif vers l'une ou l'autre des métriques. Enfin, notre ROC AUC de 0.9927 représente un score exceptionnel, très proche de la perfection. Cette métrique nous indique que notre modèle possède une capacité discriminante exceptionnelle, capable de séparer efficacement les deux classes à travers pratiquement tout les seuils de classification possibles.

III.6.2.4 Analyse de la séparation des classes

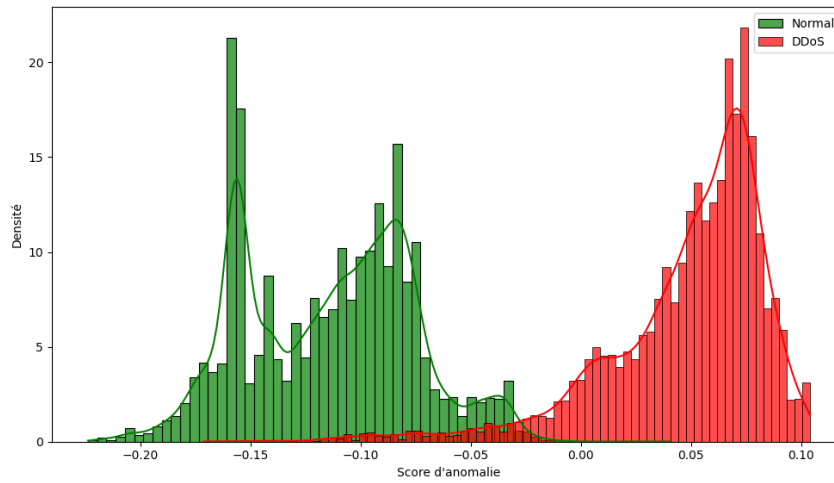


Figure III.19 – Analyse de la séparation des classes

Cette distribution des scores d'anomalie, figurée ci-dessus III.19, nous révèle des informations cruciales sur le comportement de notre modèle Isolation Forest et sa capacité à discriminer entre le trafic normal et les attaques DDoS. Nous observons une séparation remarquablement nette entre les deux classes. Le trafic normal (en vert) se concentre massivement dans la partie gauche du graphique, avec des scores d'anomalie négatifs centrés autour de -0.15. Cette distribution présente un pic très prononcé, indiquant que notre modèle attribue de manière cohérente des scores faibles (peu anormaux) au trafic légitime. La courbe de densité superposée montre une distribution quasi-normale avec une légère asymétrie, ce qui est typique pour cette classe. À l'opposé, les attaques DDoS (en rouge) se regroupent distinctement dans la partie droite, avec des scores d'anomalie positifs centrés autour de 0.06-0.08. Cette distribution est également bien définie, avec un pic clair qui démontre que notre algorithme identifie systématiquement les patterns d'attaque comme anormaux. La forme de cette distribution, légèrement plus étalée que celle du trafic normal, reflète la diversité des techniques d'attaque DDoS présentes dans notre Dataset. Le point de séparation naturel se situe approximativement autour de -0.02, où nous observons un chevauchement minimal entre les deux distributions. Cette zone de chevauchement limité explique les excellentes performances que nous avons observées dans nos métriques précédentes. La quasi-absence d'overlap entre les deux classes confirme pourquoi notre modèle atteint une précision si élevée et génère si peu de fausses alarmes. Cette visualisation valide également notre choix de l'Isolation Forest, qui démontre sa capacité intrinsèque à capturer les différences fondamentales entre comportement normal et anormal dans le trafic réseau. La séparation claire des distributions explique nos scores ROC AUC exceptionnels et confirme que notre modèle a appris des représentations discriminantes robustes pour notre tâche de détection d'anomalies.

III.6.3 Comparaison globale des performances des deux modèles

Table III.3 – Comparaison des performances entre l’auto-encodeur et l’Isolation Forest

Critère	Auto-encodeur	Isolation Forest
Rappel	100 %	99.96 %
Accuracy	95.83 %	97.55 %
Précision	93.15 %	96.87 %
Score ROC AUC	100 %	99.27 %
Score F1	96.46 %	98.39 %

L’analyse comparative des deux modèles révèle des profils de performance distincts avec des caractéristiques complémentaires, comme en témoigne le tableau III.3. En matière de rappel, l’Auto-encodeur démontre une performance parfaite de 100%, tandis qu’Isolation Forest atteint 99.96%, soit une différence marginale de 0.04% qui traduit une capacité quasi-équivalente de détection des attaques DDoS. Cependant, cette tendance s’inverse concernant la précision où Isolation Forest surpasse significativement l’Auto-encodeur avec 96.87% contre 93.15%, marquant un écart de 3.72% qui indique une réduction notable des fausses alarmes. Cette différence de précision se répercute directement sur l’accuracy globale où Isolation Forest maintient son avantage avec 97.55% versus 95.83% pour l’Auto-encodeur. Parallèlement, le score ROC AUC révèle une capacité discriminante supérieure de l’Auto-encodeur avec un score parfait de 100% contre 99.27% pour Isolation Forest, suggérant une séparabilité optimale entre les classes sur l’ensemble des seuils de classification. Cette performance de l’Auto-encodeur en termes de discrimination ne se traduit néanmoins pas dans le F1-score où Isolation Forest obtient 98.39% contre 96.46%, reflétant l’impact de sa précision supérieure sur cette métrique d’équilibre. Au final, l’Auto-encodeur privilégie une stratégie de sensibilité maximale acceptant davantage de faux positifs, tandis qu’Isolation Forest optimise l’équilibre global en minimisant les fausses alarmes tout en conservant une détection quasi-parfaite des menaces.

III.6.4 Détermination du modèle optimal

Le choix du modèle dépend fondamentalement du contexte d’application et des priorités en matière de cybersécurité :

- Si l’objectif est d’assurer qu’aucune attaque ne passe inaperçue, même au prix de quelques alertes inutiles, le modèle Auto-encodeur est préférable. Il maximise le rappel et garantit une sécurité maximale, ce qui est essentiel dans les environnements critiques (banques, infrastructures nationales, etc.).
- À l’inverse, dans des contextes où la qualité des alertes prime sur leur quantité (c’est-à-dire où il est important de ne pas perturber le fonctionnement normal du système par des alertes fausses), le modèle Isolation Forest est plus adapté. Il filtre très bien les données normales, réduisant presque totalement les faux positifs.

III.7 Démonstration du service anti-DDoS

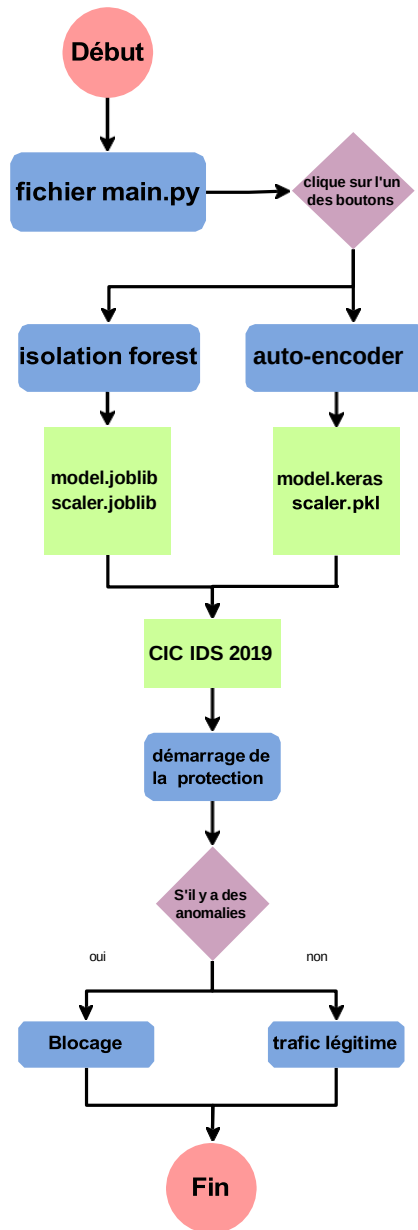


Figure III.20 – Fonctionnement de l'application

L'organigramme présenté ci-dessus III.20 offre une vue d'ensemble de l'architecture logique du service Anti-DDoS, en illustrant les différents modules impliqués ainsi que leur interaction.

Pour mieux comprendre comment cette architecture a été concrètement traduite dans l'outil développé, nous allons à présent détailler deux volets complémentaires : d'abord, la conception de l'interface utilisateur, puis l'exécution pratique du système, illustrant comment l'utilisateur interagit avec la plateforme pour surveiller, détecter et réagir face aux attaques DDoS en temps réel.

III.7.1 Conception de l'interface utilisateur Anti-DDoS

Pour rendre notre système plus accessible et facile à utiliser, nous avons conçu une interface graphique simple et intuitive. Elle permet de lancer la détection, de visualiser les alertes en temps réel et d'interagir avec les modèles IA sans passer par le code. Voici comment nous avons construit cette interface, étape par étape.

III.7.1.1 Création de l'interface graphique

L'interface graphique a été développée en Python à l'aide de la bibliothèque PyQt6, largement reconnue pour sa compatibilité avec les systèmes d'intelligence artificielle et sa capacité à produire des interfaces modernes et réactives. Elle a été conçue de manière modulaire afin de garantir une navigation intuitive et une séparation claire des fonctionnalités.

La structure de l'interface repose sur un agencement en deux zones : un menu latéral à gauche regroupe les boutons de navigation et d'action, tandis que la zone principale à droite est réservée à l'affichage des résultats et au journal des événements en temps réel.

Trois écrans principaux composent cette interface : un tableau de bord central servant d'écran d'accueil, une vue de surveillance en attente d'événements, ainsi qu'une interface de détection active permettant de visualiser les connexions suspectes au fur et à mesure de leur apparition. affichées en temps réel.

III.7.1.2 Design et style visuel de l'interface

Pour renforcer l'identité visuelle du projet, une charte graphique cohérente a été adoptée :

- Palette de couleurs : nuances de vert pastel et foncé, symbolisant la cybersécurité, la vigilance et la stabilité.
- Typographie lisible et boutons aux coins arrondis pour un rendu moderne.
- Logo bouclier + icône de sécurité : positionné en haut à droite, il renforce le sentiment de protection continue.

Le design a été pensé pour être sobre, intuitif et fonctionnel, évitant toute surcharge visuelle, afin que les utilisateurs se concentrent sur les résultats affichés.

III.7.1.3 Implémentation des boutons de commande

L'interface graphique intègre plusieurs boutons de commande organisés verticalement dans le panneau latéral gauche. Chaque bouton est conçu pour assurer une interaction intuitive, avec un code couleur et une icône spécifique facilitant leur identification, y compris pour des utilisateurs non spécialistes en sécurité réseau.

- **Boutons principaux :**
 - **Démarrer** : lance la surveillance du trafic réseau.
 - **Arrêter** : interrompt immédiatement l'analyse en cours.
 - **Filtrage @IP** : active le blocage automatique des adresses IP considérées comme malveillantes.
 - **Machine Learning / Deep Learning** : accessibles via une fenêtre secondaire, ces options permettent à l'utilisateur de choisir entre deux types de modèles de détection.

- **Boutons de navigation :**
 - **Retour** : permet de revenir à l'écran précédent.
 - **Quitter** : ferme proprement l'application.

III.7.1.4 Scripts/fonctions associés aux boutons

Le tableau suivant présente les différents boutons intégrés à l'interface utilisateur, accompagnés d'une description synthétique de leur rôle fonctionnel dans le système de détection Anti-DDoS.

TABLE III.4 – Fonctions associées aux boutons dans l'interface graphique Anti-DDoS

Bouton	Texte affiché	Fonction liée	Description
Démarrer	"Démarrer"	show detection methods()	Affiche les choix entre Deep Learning et Machine Learning
Arrêter	"Arrêter"	stop protection thread() → stop protection()	Affiche "Protection arrêtée"
Filtrage @IP	"Filtrage @IP"	show blocked ips()	Affiche une liste d'IP bloquées
Quitter	"Quitter"	QApplication.quit()	Ferme l'application
Retour	"Retour"	show_home_page()	Recharge l'interface principale

Le tableau ci-dessous récapitule les boutons de l'interface Anti-DDoS qui déclenchent l'exécution de scripts externes de détection d'attaques, selon le type de modèle choisi par l'utilisateur.

TABLE III.5 – Boutons déclenchant des scripts externes de détection dans l'interface Anti-DDoS

Boutons	Texte affiché	Fonction appelée	Script
Deep Learning	"Deep Learning"	run autoencoder detection()	ddos detection auto.py
Machine Learning	"Machine Learning"	run isolationforest detection()	detection isolation.py

III.7.1.5 Affichage des résultats et journalisation

La zone principale située à droite de l'interface est dédiée à l'affichage en temps réel des événements détectés par le système. Deux états distincts peuvent s'y manifester :

- **État inactif** : une simple ligne « Journal des événements. . . » est affichée pour indiquer l'absence d'activité suspecte.

- **État actif** : à chaque détection d'anomalie, une entrée est automatiquement ajoutée sous forme de tableau contenant les informations suivantes :
 - Horodatage de l'événement,
 - Adresse IP source et adresse IP cible,
 - Port impliqué,
 - Protocole utilisé,
 - Type d'alerte identifiée (ex. : « DDoS »).

Ce journal est mis à jour dynamiquement sans nécessiter de rechargement manuel, garantissant ainsi une surveillance fluide et continue pendant toute la session d'analyse.

III.7.2 Exécution du système et utilisation de l'interface

Une fois la configuration terminée, l'interface Anti-DDoS est prête à être utilisée. L'objectif de cette partie est de montrer comment un utilisateur interagit avec la plateforme, du démarrage du système jusqu'à l'apparition des résultats dans la zone de journalisation.

III.7.2.1 Lancement de la plateforme

Dès l'ouverture de l'application, l'utilisateur accède au tableau de bord principal. Il peut alors cliquer sur le bouton « Démarrer » pour lancer la surveillance du trafic réseau. En arrière-plan, le système initialise le modèle d'intelligence artificielle préalablement choisi (Machine Learning ou Deep Learning), et commence à analyser en temps réel les flux captés.

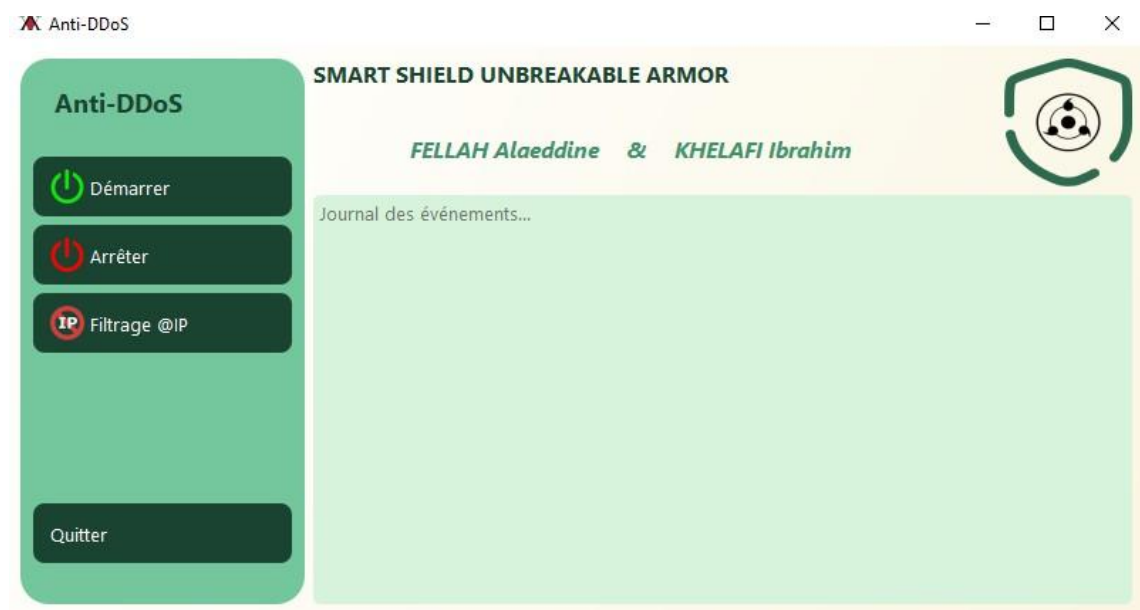


Figure III.21 – Tableau de bord principal de la plateforme Anti-ddos

III.7.2.2 Choix du moteur de détection

Avant ou après le démarrage, l'utilisateur peut choisir entre deux moteurs de détection :

Le modèle de Machine Learning, basé par exemple sur Isolation Forest, recommandé pour sa rapidité ;

Le modèle de Deep Learning, tel qu'un Auto-encodeur, plus adapté à l'analyse de comportements complexes.

Ce choix se fait simplement en cliquant sur le bouton correspondant dans l'interface (Deep Learning ou Machine Learning), selon le besoin.

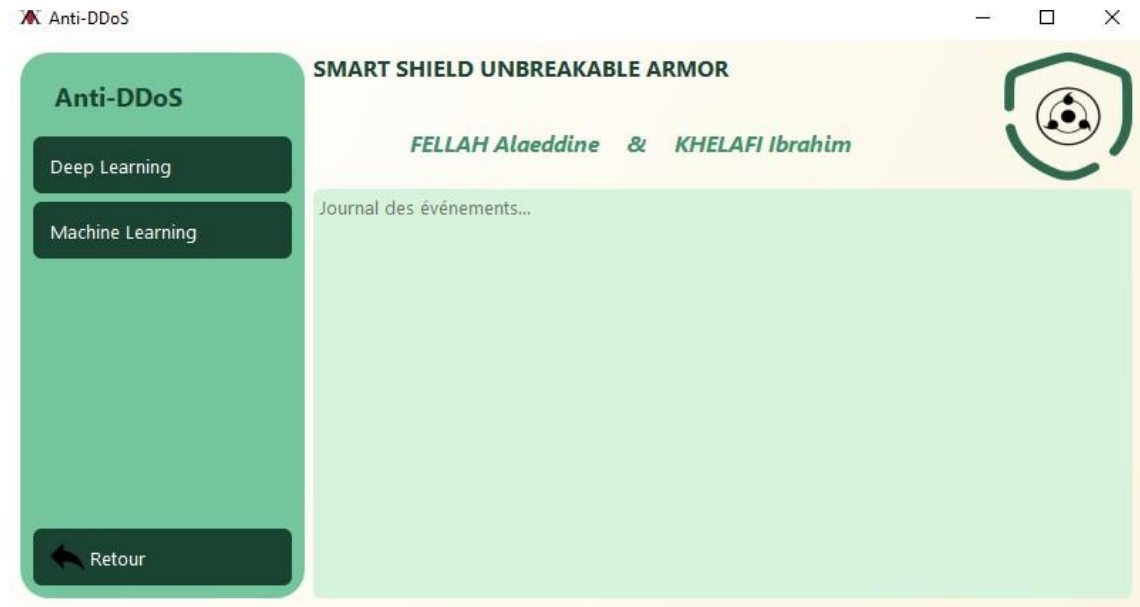


Figure III.22 – Interface de surveillance en attente de détection

III.7.2.3 Surveillance en temps réel

Une fois le modèle activé et la surveillance lancée, l'interface reste en attente dans un état passif. À ce stade, aucune alerte n'est affichée tant qu'aucune anomalie n'a été détectée. C'est ce qu'on appelle l'état neutre, visible sur l'interface « Journal des événements vide ».

III.7.2.4 Détection et affichage des anomalies

Dès qu'un comportement suspect est identifié dans le trafic réseau, une alerte est automatiquement générée et affichée dans la zone centrale de l'interface. Cette alerte prend la forme d'une ligne ajoutée au journal d'événements en temps réel.

Chaque alerte contient les informations suivantes :

- L'heure précise de la détection,
- L'adresse IP source de l'activité suspecte,
- L'adresse IP de destination visée,
- Le port réseau concerné,
- Le protocole utilisé (par exemple TCP ou UDP),
- Le type d'attaque détectée (ex. : « DDoS »).

L'affichage est instantané et mis à jour dynamiquement, permettant à l'utilisateur de suivre les événements au fil de leur apparition, sans intervention manuelle.

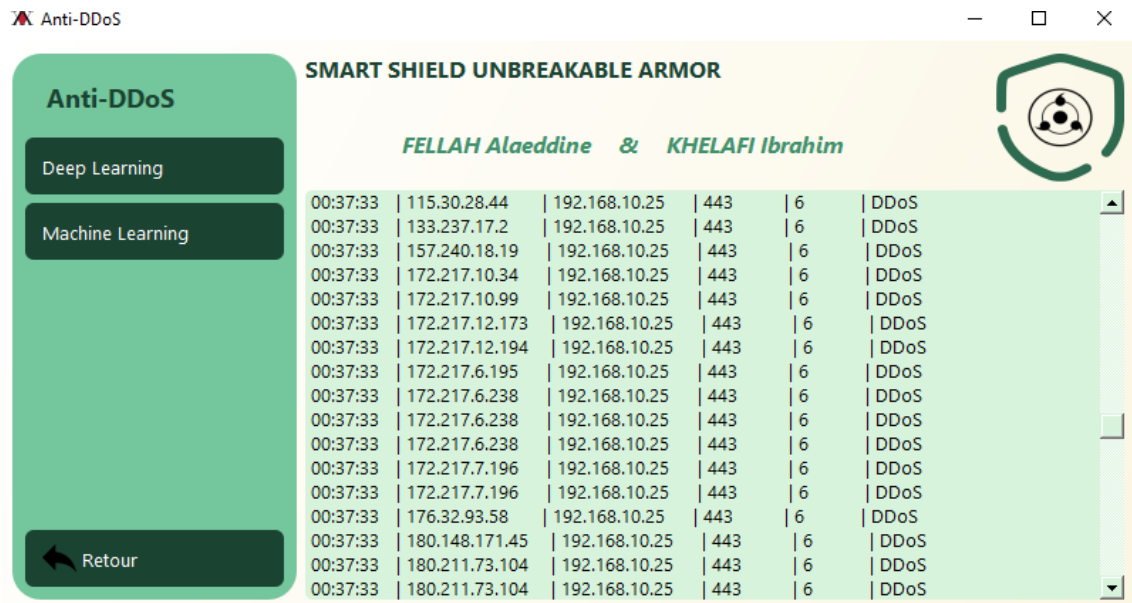


Figure III.23 – Interface de détection en temps réel des attaques DDoS via ML et DL

III.7.2.5 Réaction de l'utilisateur face aux attaques détectées

Dès qu'une attaque est détectée, l'utilisateur dispose de plusieurs options d'intervention en fonction du contexte ou de la stratégie souhaitée. Il peut choisir d'interrompre immédiatement le processus de surveillance en cliquant sur le bouton « *Arrêter* », ce qui met fin à l'analyse en cours. Si une réaction plus ciblée est requise, il peut activer le blocage automatique de l'adresse IP suspecte via le bouton « *Filtrage @IP* », ce qui neutralise instantanément la source de l'anomalie. Enfin, dans une logique d'observation passive ou d'analyse différée, l'utilisateur peut simplement suivre l'évolution des alertes sans effectuer d'intervention directe, en s'appuyant sur l'affichage en temps réel du journal d'événements.

III.7.2.6 Fin de session

À la fin de la session, l'utilisateur peut quitter proprement l'application en cliquant sur « *Quitter* », ce qui arrête tous les processus en cours et libère les ressources du système.

III.8 Conclusion

Dans ce chapitre, nous avons comparé deux méthodes non supervisées de détection d'attaques DDoS Isolation Forest et auto-encodeur. L'Auto-encodeur a détecté toutes les attaques (aucun faux négatif), mais a généré beaucoup de faux positifs, bloquant ainsi des utilisateurs légitimes. En revanche, Isolation Forest a produit moins de faux positifs, mais a laissé passer certaines attaques (faux négatifs), ce qui le rend plus permissif. Chaque modèle a donc ses forces et faiblesses : l'Auto-encodeur est plus sûr, mais moins tolérant. Isolation Forest est plus souple, mais moins rigoureux en matière de sécurité. Les deux modèles ont été intégrés dans une interface Python interactive, permettant de tester et visualiser les résultats en temps réel. Ce travail montre que le choix du modèle dépend du contexte sécurité maximale ou accessibilité utilisateur.

Conclusion générale

Ce mémoire a visé à répondre à un enjeu fondamental en cybersécurité moderne : la détection efficace des attaques par déni de service distribué (DDoS), qui compromettent la disponibilité des services et la stabilité des infrastructures réseau. L'objectif principal était d'explorer l'intégration de deux techniques d'apprentissage non supervisé, l'AE (deep learning) et l'algorithme Isolation Forest (machine learning), dans un système de détection capable d'identifier les comportements anormaux dans le trafic réseau avec précision et fiabilité.

Les résultats obtenus ont montré que les deux approches présentent des performances significatives, mais avec des caractéristiques distinctes. Le modèle AE a permis de détecter la totalité des attaques, ne laissant passer aucun faux négatif, mais au prix d'un nombre élevé de faux positifs, pouvant bloquer des utilisateurs légitimes. À l'inverse, le modèle Isolation Forest s'est montré plus tolérant, générant moins de faux positifs, mais laissant échapper certaines attaques, ce qui soulève des questions de sécurité dans des contextes critiques.

L'intégration des deux modèles dans une interface graphique interactive, développée en Python, constitue un apport concret et fonctionnel. Elle permet à l'utilisateur de lancer une détection en temps réel, en sélectionnant la méthode de protection souhaitée, et de visualiser les résultats dynamiquement. Ces résultats offrent une contribution utile à la problématique de détection d'anomalies dans le domaine de la cybersécurité. En démontrant qu'il est possible d'appliquer des modèles intelligents à des flux réseau réels, avec un bon équilibre entre précision, flexibilité et intégration, ce travail ouvre la voie à des systèmes de protection plus adaptatifs et autonomes. Une observation notable concerne la complémentarité des deux modèles, ce qui suggère l'intérêt d'approches hybrides pour renforcer la robustesse des systèmes de défense.

Cependant, certaines limites subsistent. L'étude s'est basée sur des simulations à partir du jeu de données CIC-IDS2019, sans déploiement dans un environnement réseau réel. De plus, seuls deux modèles ont été testés, alors que d'autres approches d'apprentissage non supervisé pourraient être explorées.

Les travaux futurs pourraient s'orienter vers l'implémentation matérielle, l'évaluation sur des données en temps réel, ou encore le développement de systèmes hybrides capables de basculer dynamiquement entre plusieurs modèles selon le contexte. Dans l'ensemble, cette recherche fournit une base solide pour le développement de solutions de détection intelligentes et adaptatives, et ouvre des perspectives prometteuses pour la protection des réseaux face aux menaces DDoS.

Bibliographie

- [1] R. Kabanda, B. Byera, H. Emeka, and K. T. Mohiuddin, “The history, trend, types, and mitigation of distributed denial of service attacks,” *Journal of Information Security*, vol. 14, no. 4, pp. 464–471, 2023.
- [2] S. Samonas and D. Coss, “The cia strikes back : Redefining confidentiality, integrity and availability in security.” *Journal of Information System Security*, vol. 10, no. 3, 2014.
- [3] F. O. Catak and A. F. Mustacoglu, “Distributed denial of service attack detection using autoencoder and deep neural networks,” *Journal of Intelligent & Fuzzy Systems*, vol. 37, no. 3, pp. 3969–3979, 2019.
- [4] G. Rätsch, “A brief introduction into machine learning,” *Friedrich Miescher Laboratory of the Max Planck Society*, pp. 1–6, 2004.
- [5] M. Almansoori and M. Telek, “Anomaly detection using combination of autoencoder and isolation forest,” in *1st workshop on intelligent infocommunication networks, systems and services (WI2NS2)*. Budapest University of Technology and Economics, 2023, pp. 25–30.
- [6] U. Michelucci, “An introduction to autoencoders,” *arXiv preprint arXiv :2201.03898*, 2022.
- [7] K. Berahmand, F. Daneshfar, E. S. Salehi, Y. Li, and Y. Xu, “Autoencoders and their applications in machine learning : a survey,” *Artificial Intelligence Review*, vol. 57, no. 2, p. 28, 2024.
- [8] H. Liu and B. Lang, “Machine learning and deep learning methods for intrusion detection systems : A survey,” *applied sciences*, vol. 9, no. 20, p. 4396, 2019.
- [9] M. Alkasassbeh, G. Al-Naymat, A. B. Hassanat, and M. Almseidin, “Detecting distributed denial of service attacks using data mining techniques,” *International Journal of Advanced Computer Science and Applications*, vol. 7, no. 1, 2016.
- [10] M. N. Rajkumar, “A survey on latest dos attacks : classification and defense mechanisms,” *International Journal of Innovative Research in Computer and Communication Engineering*, vol. 1, no. 8, pp. 1847–1860, 2013.
- [11] O. M. C. Osazuwa, O. Mitchell, and C. Osazuwa, “Confidentiality ; integrity, and availability in network systems : A review of related literature,” *International Journal of Innovative Science and Research Technology*, vol. 8, no. 12, pp. 1946–1953, 2023.
- [12] S. Mohanty, M. Ganguly, and P. K. Pattnaik, “Cia triad for achieving accountability in cloud computing environment,” *Int J Comput Sci Mob Appl*, 2018.

- [13] C. Douligeris and A. Mitrokotsa, “Ddos attacks and defense mechanisms : a classification,” in *Proceedings of the 3rd IEEE International Symposium on Signal Processing and Information Technology (IEEE Cat. No. 03EX795)*. IEEE, 2003, pp. 190–193.
- [14] Z. He, T. Zhang, and R. B. Lee, “Machine learning based ddos attack detection from source side in cloud,” in *2017 IEEE 4th International Conference on Cyber Security and Cloud Computing (CSCloud)*. IEEE, 2017, pp. 114–120.
- [15] R. R. Brooks, L. Yu, I. Ozcelik, J. Oakley, and N. Tusing, “Distributed denial of service (ddos) : a history,” *IEEE Annals of the History of Computing*, vol. 44, no. 2, pp. 44–54, 2021.
- [16] J. Mirkovic and P. Reiher, “A taxonomy of ddos attack and ddos defense mechanisms,” *ACM SIGCOMM Computer Communication Review*, vol. 34, no. 2, pp. 39–53, 2004.
- [17] C. S. Kalutharage, X. Liu, C. Chrysoulas, N. Pitropakis, and P. Papadopoulos, “Explainable ai-based ddos attack identification method for iot networks,” *Computers*, vol. 12, no. 2, p. 32, 2023.
- [18] N. Kilari and R. Sridaran, “A survey on security threats for cloud computing,” *IJERT*, 03 2021.
- [19] F. Lau, S. H. Rubin, M. H. Smith, and L. Trajkovic, “Distributed denial of service attacks,” in *Smc 2000 conference proceedings. 2000 ieee international conference on systems, man and cybernetics. 'cybernetics evolving to systems, humans, organizations, and their complex interactions' (cat. no. 0, vol. 3)*. IEEE, 2000, pp. 2275–2280.
- [20] N. Hoque, D. K. Bhattacharyya, and J. K. Kalita, “Botnet in ddos attacks : trends and challenges,” *IEEE Communications Surveys & Tutorials*, vol. 17, no. 4, pp. 2242–2270, 2015.
- [21] E. Alomari, S. Manickam, B. B. Gupta, S. Karuppayah, and R. Alfari, “Botnet-based distributed denial of service (ddos) attacks on web servers : classification and art,” *arXiv preprint arXiv :1208.0403*, 2012.
- [22] A. Asosheh and N. Ramezani, “A comprehensive taxonomy of ddos attacks and defense mechanism applying in a smart classification,” *WSEAS Transactions on Computers*, vol. 7, no. 4, pp. 281–290, 2008.
- [23] T. A. Tuan, H. V. Long, L. H. Son, R. Kumar, I. Priyadarshini, and N. T. K. Son, “Performance evaluation of botnet ddos attack detection using machine learning,” *Evolutionary Intelligence*, vol. 13, no. 2, pp. 283–294, 2020.
- [24] H. Fan, F. Zhang, and Z. Li, “Anomalydae : Dual autoencoder for anomaly detection on attributed networks,” *arXiv preprint arXiv :2009.06516*, 2020.
- [25] H. Guo, X. Fan, A. Cao, F.-Y. Liang *et al.*, “Peek inside the closed world : Evaluating autoencoder-based detection of ddos to cloud,” *arXiv preprint arXiv :1907.12300*, 2019.
- [26] M. Neri *et al.*, “Unsupervised network anomaly detection with autoencoders and image-based representation,” *arXiv preprint arXiv :2501.12345*, 2025.
- [27] A. Haque and H. Soliman, “A transformer-based autoencoder with isolation forest and xgboost for malfunction and intrusion detection in wireless sensor networks for forest fire prediction,” *Future Internet*, vol. 17, no. 4, p. 164, 2025.

- [28] E. Agyemang, “Anomaly detection using unsupervised machine learning algorithms : A simulation study,” 04 2024.
- [29] H. Xu, G. Pang, Y. Wang, and Y. Wang, “Deep isolation forest for anomaly detection,” *arXiv preprint arXiv :2203.00001*, 2022.
- [30] A. Singh and J. Jang-Jaccard, “Autoencoder-based unsupervised intrusion detection using multi-scale convolutional recurrent networks,” *arXiv preprint arXiv :2208.12345*, 2022.
- [31] I. Ortega Fernández, M. Sestelo, J. Burguillo, and C. Piñón-Blanco, “Network intrusion detection system for ddos attacks in ics using deep autoencoders,” *Wireless Networks*, vol. 30, pp. 1–17, 01 2023.
- [32] T. Smolen and L. Benova, “Comparing autoencoder and isolation forest in network anomaly detection,” *ResearchGate*, 2022.
- [33] M. Roopak, S. Parkinson, G. Y. Tian, Y. Ran, S. Khan, and B. Chandrasekaran, “An unsupervised approach for the detection of zero-day distributed denial of service attacks in internet of things networks,” *IET Networks*, vol. 13, no. 5-6, pp. 513–527, 2024.
- [34] M. Fagerlund, “Anomaly detection for insider threats : Comparing lstm autoencoder, isolation forest and elasticsearch,” Master’s thesis, KTH Royal Institute of Technology, 2023.
- [35] M. S. Lakshmi, G. Rajavikram, and P. Kumar, “Evaluating the isolation forest method for anomaly detection in software-defined networking,” *Journal of Electrical Systems*, 2023.
- [36] F. T. Liu, K. M. Ting, and Z.-H. Zhou, “Isolation forest,” pp. 413–422, 2008.
- [37] O. Mykhaylova, A. Shtypka, and T. Fedynyshyn, “An isolation forest - based approach for brute force attack detection,” O. Orobchuk and M. Petryk, Eds., vol. 3842. CEUR-WS.org, 2024, pp. 43–54.
- [38] J. Lesouple, C. Baudoin, M. Spigai, and J.-Y. Tournet, “Generalized isolation forest for anomaly detection,” *Pattern Recognition Letters*, vol. 149, pp. 109–119, 2021.
- [39] H. Lu, “Evaluating the performance of svm, isolation forest, and dbscan for anomaly detection,” in *ITM Web of Conferences*, vol. 70. EDP Sciences, 2025, p. 04012.
- [40] Z. Cheng, S. Wang, P. Zhang, S. Wang, X. Liu, and E. Zhu, “Improved autoencoder for unsupervised anomaly detection,” *International Journal of Intelligent Systems*, vol. 36, no. 12, pp. 7103–7125, 2021.
- [41] W. Xu, J. Jang-Jaccard, A. Singh, Y. Wei, and F. Sabrina, “Improving performance of autoencoder-based network anomaly detection on nsl-kdd dataset,” *IEEE Access*, vol. 9, pp. 140 136–140 146, 2021.
- [42] F. O. Catak and A. F. Mustacoglu, “Distributed denial of service attack detection using autoencoder and deep neural networks,” *Journal of Intelligent & Fuzzy Systems*, vol. 37, no. 3, pp. 3969–3979, 2019.
- [43] L. Zhang, S. Wang, and B. Liu, “Deep learning for sentiment analysis : A survey,” *Wiley interdisciplinary reviews : data mining and knowledge discovery*, vol. 8, no. 4, p. e1253, 2018.

- [44] P. G. Naik, *Conceptualizing python in google colab : Hands-on practical sessions*. Shashwat Publication, 2023.
- [45] “Project Jupyter — jupyter.org,” <https://jupyter.org/>, [Accessed 02-07-2025].
- [46] “Kaggle : Your Machine Learning and Data Science Community,” <https://www.kaggle.com/>, [Accessed 02-07-2025].
- [47] “Visual Studio Code,” <https://code.visualstudio.com/>, [Accessed 02-07-2025].
- [48] “Welcome to Python,” <https://www.python.org/>, [Accessed 02-07-2025].
- [49] “TensorFlow — tensorflow.org,” <https://www.tensorflow.org/>, [Accessed 02-07-2025].
- [50] K. Team, “Keras : Deep Learning for humans — keras.io,” <https://keras.io/>, [Accessed 02-07-2025].
- [51] F. Pedregosa, G. Varoquaux, A. Gramfort, V. Michel, B. Thirion, O. Grisel, M. Blondel, P. Prettenhofer, R. Weiss, V. Dubourg, J. Vanderplas, A. Passos, D. Cournapeau, M. Brucher, M. Perrot, and E. Duchesnay, “Scikit-learn : Machine learning in Python,” *Journal of Machine Learning Research*, vol. 12, pp. 2825–2830, 2011.
- [52] “PyQt6 — pypi.org,” <https://pypi.org/project/PyQt6/>, [Accessed 02-07-2025].
- [53] I. Sharafaldin, A. H. Lashkari, S. Hakak, and A. A. Ghorbani, “Developing realistic distributed denial of service (ddos) attack dataset and taxonomy,” in *2019 international carnaahan conference on security technology (ICCST)*. IEEE, 2019, pp. 1–8.