

الجمهورية الجزائرية الديمقراطية الشعبية
REPUBLIQUE ALGERIENNE DEMOCRATIQUE ET POPULAIRE
وزارة التعليم العالي و البحث العلمي
Ministère de l'Enseignement Supérieur et de la Recherche Scientifique
جامعة أبي بكر بلقايد - تلمسان

Université Aboubakr Belkaïd – Tlemcen –
Faculté de TECHNOLOGIE



MEMOIRE

Présenté pour l'obtention du **diplôme** de **MASTER**

En : Télécommunications

Spécialité : Réseaux et Télécommunications

Par :

AISSANI Djamel Eddine
BENMANSOUR Mohammed Omar

Sujet

**Conception et déploiement d'un système intelligent de détection d'intrusions
sur des dispositifs Edge**

Soutenu publiquement, le 21/06/2026, devant le jury composé de :

MERZOUGUI Rachid	Pr	Université de Tlemcen	Président
TALEB Sara	MCB	Université de Tlemcen	Examineur
MOUSSAOUI Djilali	MCA	Université de Tlemcen	Encadrant
HADJILA Mourad	Pr	Université de Tlemcen	Co-encadrant
OURRAD Soumia	MCA	Université de Tlemcen	Expert I2E

Année universitaire : 2025/2026

Remerciement

En premier lieu, nous rendons grâce à Allah le Tout-Puissant de nous avoir donné la volonté, la persévérance et la patience nécessaires pour accomplir ce travail.

Nous exprimons notre profonde reconnaissance à **Dr. Moussaoui Djilali**, encadreur de ce mémoire, ainsi qu'à **Pr. Hadjila Mourad**, Co-encadreur, pour leur accompagnement, leurs conseils avisés, leur disponibilité et les encouragements qu'ils nous ont apportés tout au long de cette étude.

Nous adressons également nos sincères remerciements aux membres du jury **Pr. MERZOUGUI Rachid**, **Dr. TALEB Sara** et **Dr. OURRAD Soumia**, d'avoir accepté d'examiner et d'évaluer notre travail, ainsi que pour l'intérêt qu'ils lui ont accordé.

Nous tenons aussi à remercier l'ensemble des enseignants et du personnel du département des télécommunications de l'Université Abou-Bakr Belkaïd de Tlemcen pour leur soutien, leur aide et les connaissances qu'ils nous ont transmises durant notre cursus universitaire.

Enfin, nous exprimons notre gratitude à toutes les personnes qui ont participé, directement ou indirectement, à l'aboutissement de ce mémoire.

Dédicace

C'est avec une profonde reconnaissance et une grande émotion que je dédie ce modeste travail à toutes les personnes qui m'ont accompagné, soutenu et encouragé durant tout mon parcours.

À mes très chers parents, pour leur amour infini, leur patience, leurs précieux conseils et les nombreux sacrifices qu'ils ont consentis afin de me permettre d'atteindre ce stade de mes études.

À ma sœur et à mon petit frère, pour leur présence, leur soutien permanent et la force qu'ils m'ont apportée dans les moments les plus difficiles.

À ma grand-mère, dont les prières et les encouragements ont toujours illuminé mon chemin et renforcé ma détermination.

À toute ma famille, pour son affection, sa confiance et son soutien tout au long de mon parcours universitaire.

À toutes les personnes qui ont participé, de près ou de loin, à l'aboutissement de ce travail.

Qu'ils trouvent ici l'expression de ma sincère gratitude et de mon profond respect.

AISSANI Djamel Eddine

Dédicace

Je dédie ce travail à ma famille, qui a toujours été ma force silencieuse. À mon père et à ma mère, pour leurs sacrifices, leur patience et leur amour. À mes frères et sœurs, pour leur présence et leur soutien.

À ma grand-mère, dont les prières m'ont accompagné tout au long de ce chemin.

À la personne la plus chère, qui a cru en moi sans jamais faillir et dont le soutien m'a été précieux pendant des années.

À mon ami Aymen, pour son amitié sincère et ses encouragements.

À mes chers abonnés sur toute les plateformes qui m'ont toujours soutenu

À mes encadrants, Dr. Moussaoui Djilali et Pr. Hadjila Mourad, pour leurs conseils et leur accompagnement.

À tous mes amis qui m'ont encouragé, je leur adresse mes sincères remerciements.

BENMANSOUR Mohammed Omar

Résumé

Avec l'explosion des cyberattaques, protéger les réseaux informatiques est devenu une priorité absolue. Ce travail propose un système intelligent capable de surveiller un réseau et de détecter automatiquement les attaques, sans intervention humaine permanente.

Pour construire ce système, nous avons utilisé un ensemble de données contenant près d'un million de flux réseau réels, incluant du trafic normal et six types d'attaques différentes. Nous avons ensuite entraîné et comparé dix modèles d'intelligence artificielle, certains capables de reconnaître des attaques connues, d'autres capables de détecter des menaces inconnues. Le meilleur modèle, XGBoost, a correctement identifié 99.93% des flux testés.

Ce modèle a ensuite été installé sur un Raspberry Pi 5, un mini-ordinateur de la taille d'une carte de crédit et consommant seulement 6W. Une application web appelée AiNetguard a été développée, permettant à n'importe quel administrateur réseau de surveiller son réseau en temps réel depuis un simple navigateur et d'être alerté immédiatement en cas d'attaque.

Ce projet démontre qu'une protection réseau de haut niveau n'est plus réservée aux grandes entreprises disposant de serveurs coûteux mais elle est désormais accessible à tous.

Mots-clés : Détection d'intrusions, Intelligence Artificielle, Cybersécurité, Raspberry Pi 5, Surveillance réseau

Abstract

With the explosion of cyberattacks, protecting computer networks has become a critical priority. This work proposes an intelligent system capable of monitoring a network and automatically detecting attacks, without requiring constant human supervision.

To build this system, we used a dataset containing nearly one million real network flows, including normal traffic and six different types of attacks. We then trained and compared ten artificial intelligence models, Some designed to recognize known attacks, others capable of detecting unknown threats. The best model, XGBoost, correctly identified 99.93% of the tested flows.

This model was then installed on a Raspberry Pi 5, a mini-computer the size of a credit card and consuming only 6W of power. A web application called AiNetguard was developed, allowing any network administrator to monitor their network in real time from a simple web browser and receive immediate alerts when an attack is detected.

This project demonstrates that high-level network protection is no longer reserved for large companies with expensive servers, it is now accessible to everyone.

Keywords: Intrusion Detection, Artificial Intelligence, Cybersecurity, Raspberry Pi 5, Network Monitoring.

ملخص

مع تصاعد الهجمات الإلكترونية، أصبحت حماية الشبكات الحاسوبية أولوية قصوى لا يمكن تجاهلها. يقدم هذا العمل نظاماً ذكياً قادراً على مراقبة الشبكة واكتشاف الهجمات تلقائياً، دون الحاجة إلى مراقبة بشرية مستمرة.

لبناء هذا النظام، اعتمدنا على مجموعة بيانات تضم قرابة مليون تدفق شبكي حقيقي، تشمل حركة المرور العادية وستة أنواع مختلفة من الهجمات. بعد ذلك، قمنا بتدريب عشرة نماذج للذكاء الاصطناعي ومقارنتها — بعضها مُصمَّم للتعرف على الهجمات المعروفة، وبعضها الآخر قادر على اكتشاف التهديدات غير المعروفة. وقد تمكَّن أفضل نموذج، وهو XGBoost، من التعرف الصحيح على 99.93% من التدفقات التي اختبر عليها.

بعد ذلك، تم تثبيت هذا النموذج على جهاز Raspberry Pi 5، وهو حاسوب مصغَّر بحجم بطاقة الائتمان ولا يستهلك سوى 6 واط من الطاقة. كما تم تطوير تطبيق ويب باسم **AiNetguard**، يتيح لأي مسؤول شبكة مراقبة شبكته في الوقت الحقيقي عبر متصفح الإنترنت العادي، وتلقّي تنبيهات فورية عند اكتشاف أي هجوم.

يُثبت هذا المشروع أن الحماية المتقدمة للشبكات لم تعد حكراً على الشركات الكبرى التي تمتلك خوادم باهظة التكلفة بل أصبحت في متناول الجميع.

الكلمات المفتاحية: كشف الاختراقات، الذكاء الاصطناعي، الأمن السيبراني، Raspberry Pi 5، مراقبة الشبكة.

Table des matières

Remerciement

Dédicace

Résumé

Table des Matière

Liste des figures

Liste des tableaux

Liste des abréviation

Introduction Générale 1

Chapitre I : Introduction à l'intelligence artificielle..... 3

1.Introduction..... 4

2.Introduction à l'intelligence artificielle 4

2.1.Définition et structuration de l'IA 4

2.1.1.Organisation hiérarchique de l'IA 5

2.2.Domaines d'application de l'IA 6

2.3.Place de l'IA dans la cybersécurité 7

3.Apprentissage automatique (Machine Learning)..... 8

3.1.Principe du Machine Learning..... 8

3.2.Types d'apprentissage 9

3.2.1.Apprentissage supervisé..... 9

3.2.2.Apprentissage non supervisé..... 10

3.2.3.Apprentissage semi-supervisé..... 10

4.Deep Learning..... 10

4.1.Principe des réseaux de neurones 10

4.2.Architecture d'un réseau de neurones 11

4.3.Fonctions d'activation 12

4.4.Fonction de perte et optimisation.....	12
5.Modèles utilisés pour la détection d'attaques.....	13
5.1.Réseau de neurones Dense (MLP).....	13
5.2.Autoencoder.....	13
5.3.LSTM (Long Short-Term Memory)	14
5.4.Random Forest.....	15
5.5.XGBoost (Extreme Gradient Boosting).....	15
5.6.LightGBM (Light Gradient Boosting Machine).....	17
6.Conclusion	18
Chapitre II: Cybersécurité et systèmes de détection d'intrusion (IDS)	19
1.Introduction.....	20
2.Fondamentaux de la cybersécurité.....	20
2.1.Définition et objectifs de la cybersécurité	20
2.2.Typologie et classification des menaces réseau.....	21
3.Les différents types d'attaque sur un réseau.....	21
3.1.Attaques par déni de service (DoS/DDoS)	21
3.2.Reconnaissance et scan de ports:	22
3.3.Attaques par force brute:.....	22
3.4.Attaques par injection:	22
4.Les systèmes de détection d'intrusion (IDS).....	22
4.1.Définition et mission.....	22
4.2.Architecture fonctionnelle	23
5.Approches de détection.....	23
5.1.Détection par signatures.....	23
5.2.Détection par anomalies.....	24
5.3.Approches hybrides	24

6. Apport de l'intelligence artificielle à la détection d'intrusions.....	24
6.1. Pertinence et atouts de l'intelligence artificielle	24
6.2. Améliorations apportées par l'IA aux systèmes de détection d'intrusions	24
6.3. Limites et défis de l'application de l'IA à la détection d'intrusions.....	25
7. Infrastructure et déploiement en périphérie de réseau (Edge Computing)	25
7.1. Définition et principes fondamentaux de l'Edge Computing.....	25
7.2. Intérêt pour la détection d'intrusion	26
7.3. Le Raspberry Pi comme plateforme Edge	26
8. Conclusion	28
Chapitre III: Conception et entraînement du modèle d'IA pour détection d'intrusion.	29
1. Introduction.....	30
2. Architecture Globale du Système	30
2.1. Description de l'Architecture Proposée.....	30
2.2. Pipeline de Détection	32
3. Choix du Dataset.....	32
3.1. Présentation du Dataset CIC-IDS 2017	32
3.2. Description des Caractéristiques du Dataset.....	33
3.3. Sélection des Attaques Étudiées	33
4. Prétraitement des Données.....	35
4.1. Nettoyage des Données.....	35
4.2. Sélection des Features.....	36
4.3. Normalisation.....	36
4.4. Équilibrage des Classes.....	37
4.5. Division des Données	37
5. Entraînement des Modèles	37
5.1. Modèles Supervisés	38

5.1.1.Random Forest	38
5.1.2.XGBoost	39
5.1.3.LightGBM.....	41
5.1.4.Deep Neural Network (DNN).....	42
5.1.5.LSTM (Long Short-Term Memory)	44
5.1.6.BLSTM (Bidirectional LSTM).....	45
5.2.Modèles Non Supervisés (Détection d'Anomalies)	47
5.2.1.Autoencoder (AE).....	47
5.2.2.Variational Autoencoder (VAE).....	49
6.Évaluation des Performances	50
6.1.Métriques d'Évaluation	50
6.2.Résultats sur l'Ensemble de Test Interne	52
6.2.1.Modèles Supervisés	52
6.2.2.Modèles Non Supervisés.....	52
6.3.Résultats sur l'Ensemble de Test Externe	53
6.4.Analyse des Résultats	53
7.Discussion des résultats	55
8.Conclusion	57
Chapitre IV: Déploiement et implémentation du système IDS sur Raspberry Pi 5	58
1.Introduction.....	59
2.Déploiement sur Raspberry Pi 5	59
2.1.Pipeline opérationnel (application IDS).....	59
2.2.Présentation De Raspberry Pi 5	60
2.3.Mise en place de l'environnement matériel et logiciel	61
2.4.Transfert et Intégration des Modèles Entraînés	63
2.5.Architecture du Système Déployé (IDS)	64

2.5.1.Diagramme de séquence	65
2.5.2.Architecture globale du système	66
2.5.3.Architecture logicielle.....	73
2.6.Scénarios de Test et Validation en Conditions Réelles	76
2.6.1.Test Hors Ligne - Mode CSV	76
2.6.2.Test en Temps Réel - Mode Live.....	79
3.Discussion des résultats	80
4.Conclusion	81
Conclusion Générale.....	82
Bibliographies.....	86

Liste des figures

Figure 1-1: IA, Machine Learning et Deep Learning : relations et différences	5
Figure 1-2: Domaines d'application de l'IA	6
Figure 1-3 : Le processus d'apprentissage	9
Figure 1-4 : Organisation en couches	11
Figure 1-5: Architecture de l'autoencodeur	14
Figure 1-6: Architecture du modèle LSTM.	14
Figure 1-7: Architecture du modèle Random Forrest	15
Figure 1-8: Architecture du modèle XGBoost	16
Figure 3-1 : Pipeline Expérimental (dataset)	31
Figure 4-1: Pipeline opérationnel (application IDS).....	59
Figure 4-2: Image de Raspberry Pi 5	60
Figure 4-3: Les étapes d'installation et configuration du système d'exploitation sur Raspberry Pi 5	61
Figure 4-4: Environnement logiciel et bibliothèques utilisées sur Raspberry Pi 5	62
Figure 4-5 : Flux de travail en temps réel du système IDS.....	64
Figure 4-6: Diagramme de séquence	65
Figure 4-7: Architecture globale du système	66
Figure 4-8: Capture d'écran de Tableau Bord	69
Figure 4-9: Image de la barre de titre.....	70
Figure 4-10: Image des compteurs statiques.....	70
Figure 4-11: Image des flux détectés	70
Figure 4-12: Image des alertes en temps réel.....	71
Figure 4-13: Image des paramètres de capture en temps réel.....	71
Figure 4-14: Image du système détection hors-ligne	72
Figure 4-15: Image des prédictions récentes	72

Figure 4-16: Architecture logicielle du système IDS	73
Figure 4-17 : Résultats du test hors-ligne	78
Figure 4-18: Résultats du test hors-ligne	78
Figure 4-19: Détection en temps réel de l'attaque Port Scan.....	79
Figure 4-20: Détection en temps réel de l'attaque Brute Force	79
Figure 4-21: Détection en temps réel de l'attaque DoS	79

Liste des tableaux

Tableau 1-1 : Domaines d'application de l'IA.....	7
Tableau 1-2: Principales applications de l'IA en cybersécurité	8
Tableau 1-3 : Fonctions d'activation et leurs caractéristiques	12
Tableau 2-1 : Fonctions d'activation et leurs caractéristiques	26
Tableau 3-1 : Pipeline Globale	31
Tableau 3-2 : Type d'attaques existant dans Dataset CIC IDS 2017.....	34
Tableau 3-3 : La distribution finale du dataset après traitement.....	35
Tableau 3-4: Résultats du modèle Random Forest	38
Tableau 3-5 : Matrice de confusion du modèle Random Forest.....	39
Tableau 3-6: Résultats du modèle XGBoost.....	40
Tableau 3-7: Matrice de confusion du modèle XGBoost	40
Tableau 3-8: Résultats du modèle LightGBM	41
Tableau 3-9: Matrice de confusion du modèle LightGBM.....	42
Tableau 3-10: Résultats du modèle DNN	43
Tableau 3-11: Matrice de confusion du modèle DNN.....	43
Tableau 3-12: Résultats du modèle LSTM	44
Tableau 3-13 : Matrice de confusion du modèle LSTM.....	45
Tableau 3-14: Résultats du modèle BLSTM	46
Tableau 3-15: Matrice de confusion du modèle BLSTM	46
Tableau 3-16: Résultats du modèle AE	47
Tableau 3-17: Matrice de confusion du modèle AE Multiclasse.....	48
Tableau 3-18: Matrice de confusion du modèle AE Binaire	48
Tableau 3-19: Résultats du modèle VAE.....	49
Tableau 3-20: Matrice de confusion du modèle VAE Multiclasse.....	50

Tableau 3-21: Matrice de confusion du modèle VAE Binaire	50
Tableau 3-22 : Les résultats des modèles supervisés.....	52
Tableau 3-23: Les résultats des modèles non supervisés.....	52
Tableau 3-24 : L'évaluation sur le dataset de test	53
Tableau 3-25: La matrice de confusion du modèle XGBoost.....	54
Tableau 4-1 : Caractéristiques Techniques de Raspberry Pi 5	61
Tableau 4-2: Les fichiers de sortie de l'entraînement du modèle.....	63
Tableau 4-3: Routes API Principales	74
Tableau 4-4: Les résultats du test hors ligne.....	77
Tableau 4-5:La répartition des attaques détectées	77

Liste des abréviations

AE : Autoencoder / Autoencodeur.

AUC : Area Under the Curve.

BLSTM : Bidirectional Long Short-Term Memory.

CIA : Confidentiality, Integrity, Availability.

CIC-IDS : Canadian Institute for Cybersecurity - Intrusion Detection System.

CNN : Convolutional Neural Network / Réseaux de neurones convolutifs.

DDoS : Distributed Denial of Service.

DNN : Deep Neural Network / Réseaux de neurones denses.

DoS : Denial of Service.

FN : False Negative.

FP : False Positive.

IA : Intelligence artificielle.

IAT : Inter-Arrival Time / Intervalles inter-arrivées.

IDS : Intrusion Detection System / Systèmes de détection d'intrusion.

LSTM : Long Short-Term Memory.

MLP : Multilayer Perceptron / Perceptrons multicouches.

MSE : Mean Squared Error / Erreur quadratique moyenne.

RNN : Recurrent Neural Network / Réseaux de neurones récurrents.

ROC : Receiver Operating Characteristic Curve.

SGD : Stochastic Gradient Descent / Descente de gradient stochastique.

SMOTE : Synthetic Minority Over-sampling Technique.

TN : True Negative.

TP : True Positive.

VAE : Variational Autoencoder / Autoencodeurs variationnels.

XGBoost : eXtreme Gradient Boosting.

XSS : Cross-site Scripting.

Introduction Générale

La sécurité des réseaux informatiques représente aujourd'hui l'un des défis majeurs de notre société numérique. Face à une cybercriminalité en constante évolution, les organisations publiques et privées sont confrontées à des menaces de plus en plus sophistiquées : attaques par déni de service, intrusions furtives, logiciels malveillants et tentatives de force brute se multiplient à une cadence sans précédent. Selon le rapport de l'(ISC)² de 2023, le déficit mondial d'experts en cybersécurité dépasse les quatre millions de professionnels, rendant indispensable le développement de systèmes de détection automatisés et intelligents.

Les systèmes de détection d'intrusions (IDS) constituent la première ligne de défense contre ces menaces. Traditionnellement fondés sur des signatures statiques, ces systèmes peinent à s'adapter à l'émergence continue de nouvelles formes d'attaques, notamment les attaques zero-day qui exploitent des vulnérabilités encore inconnues. C'est dans ce contexte que l'intelligence artificielle, et plus particulièrement l'apprentissage automatique, s'impose comme une solution prometteuse, capable d'apprendre les comportements normaux du trafic réseau et d'identifier toute déviation significative.

Les avancées récentes dans le domaine de l'intelligence artificielle ont ouvert de nouvelles perspectives pour la détection d'intrusions. Des algorithmes tels que XGBoost, les réseaux de neurones profonds (DNN), les réseaux de mémoire à long terme (LSTM) et les autoencodeurs variationnels (VAE) offrent des capacités de classification et de détection d'anomalies remarquables, surpassant les approches conventionnelles sur des benchmarks reconnus comme le dataset CICIDS2017 du Canadian Institute for Cybersecurity.

Parallèlement, le développement de l'Edge Computing ouvre la voie à un déploiement de systèmes IDS directement au cœur des infrastructures réseau, sans dépendance à des serveurs centraux coûteux. Des plateformes embarquées comme le Raspberry Pi 5, alliant puissance de calcul, faible consommation énergétique et coût accessible, constituent des candidats idéaux pour héberger de tels systèmes intelligents en conditions réelles.

C'est dans cette double perspective, d'une part l'adaptation des modèles d'apprentissage automatique avancés à la détection d'anomalies réseau, et d'autre part leur déploiement sur une plateforme embarquée à faible coût, que s'inscrit ce travail de fin d'études intitulé **"Design and Deployment of an Intelligent Intrusion Detection System on Edge Devices"**.

Objectifs du travail :

Ce mémoire poursuit trois objectifs principaux :

- Évaluer et comparer de manière rigoureuse plusieurs modèles de l'intelligence artificielle — incluant des approches supervisées (Random Forest, XGBoost, LightGBM, DNN, LSTM, BLSTM) et non supervisées (AE, VAE) — appliqués à la détection d'intrusions réseau sur le dataset CIC-IDS 2017.
- Identifier le modèle offrant le meilleur compromis entre performance de détection, légèreté et vitesse d'inférence pour un déploiement embarqué.
- Concevoir et déployer un système IDS complet et opérationnel sur Raspberry Pi 5, intégrant une capture réseau en temps réel et un tableau de bord web de supervision.

Organisation du mémoire :

Ce mémoire s'organise en trois chapitres complémentaires :

Le **premier chapitre** pose les fondements théoriques de l'intelligence artificielle, en présentant les paradigmes d'apprentissage automatique et profond ainsi que les principaux modèles utilisés dans ce travail.

Le **deuxième chapitre** dresse un panorama de la cybersécurité moderne, des typologies d'attaques réseau et des systèmes de détection d'intrusions existants, avant d'introduire le concept de déploiement en périphérie de réseau.

Le **troisième chapitre** constitue le cœur expérimental de ce travail. Il décrit le pipeline complet de conception et d'entraînement du système IDS, depuis le prétraitement du dataset CIC-IDS 2017 jusqu'à l'évaluation comparative des dix modèles entraînés, en passant par la sélection des features, l'équilibrage des classes et l'analyse détaillée des performances obtenues.

Le **quatrième chapitre** présente le déploiement opérationnel du système sur Raspberry Pi 5. Il décrit l'architecture de l'application 'AiNetguard', le transfert des modèles entraînés vers la plateforme embarquée, la capture réseau en temps réel via Scapy, les scénarios de test en conditions réelles ainsi que les limitations identifiées et les perspectives d'amélioration futures.

Chapitre I :

Introduction à l'intelligence artificielle

1. Introduction

Ce premier chapitre est consacré aux fondements de l'intelligence artificielle. Nous y définirons ce qu'est l'IA et comment elle s'organise, en passant par l'apprentissage automatique et l'apprentissage profond. Nous distinguerons ensuite les différents types d'apprentissage : supervisé, non supervisé et semi-supervisé. Le fonctionnement des réseaux de neurones sera également expliqué, notamment leur architecture et leurs mécanismes d'apprentissage.

Nous présenterons les principaux modèles utilisés dans la détection d'attaques, comme le MLP, l'auto encodeur, le LSTM, la forêt aléatoire, XGBoost et LightGBM. Enfin, nous introduirons les métriques qui permettent d'évaluer leurs performances, telles que l'accuracy, la précision, le rappel et le F1-score. Ce chapitre fournira ainsi toutes les bases théoriques nécessaires à la compréhension de notre travail.

2. Introduction à l'intelligence artificielle

2.1. Définition et structuration de l'IA

L'intelligence artificielle (IA) constitue une branche de l'informatique consacrée à la conception de systèmes capables d'exécuter des tâches nécessitant habituellement l'intervention d'une intelligence humaine ^[1]. Ces systèmes se caractérisent par leur aptitude à résoudre des problèmes, à acquérir des connaissances, à élaborer des raisonnements, à formuler des décisions et à organiser des actions de manière autonome^[2].

Les fondements de l'IA s'appuient sur plusieurs champs disciplinaires interconnectés :

- Les mathématiques constituent le socle sur lequel repose l'IA, en fournissant les outils fondamentaux de la logique formelle, du calcul algorithmique et des probabilités^[1].
- La théorie de la décision examine les mécanismes logiques et rationnels guidant les choix dans des contextes marqués par l'incertitude, et ses concepts sont largement mobilisés en IA^[3].
- Les neurosciences apportent une compréhension du fonctionnement du système nerveux et des mécanismes cognitifs, permettant d'inspirer des architectures de traitement de l'information^[4].
- La cybernétique analyse les systèmes capables d'atteindre un état cible en recueillant des informations sur leur environnement et en modifiant leur comportement en conséquence^[5]. La distinction fondamentale avec l'IA réside dans l'usage privilégié des

mathématiques pour la modélisation en cybernétique, alors que l'IA mobilise davantage le raisonnement logique et les méthodes informatiques^[6].

- La linguistique apporte un éclairage scientifique sur le langage humain, essentiel pour développer des systèmes de traitement automatique des langues ^[7].
- La science de la vision étudie les mécanismes de la perception visuelle et inspire les approches de vision par ordinateur, domaine où l'apprentissage profond a apporté des progrès considérables ^[8]

2.1.1. Organisation hiérarchique de l'IA

L'IA s'organise selon une structure emboîtée ^[9]. Le Machine Learning (apprentissage automatique) en constitue le premier sous-ensemble, regroupant les techniques statistiques permettant aux machines d'apprendre à partir de données sans programmation explicite des règles à appliquer^{[10], [11]}.

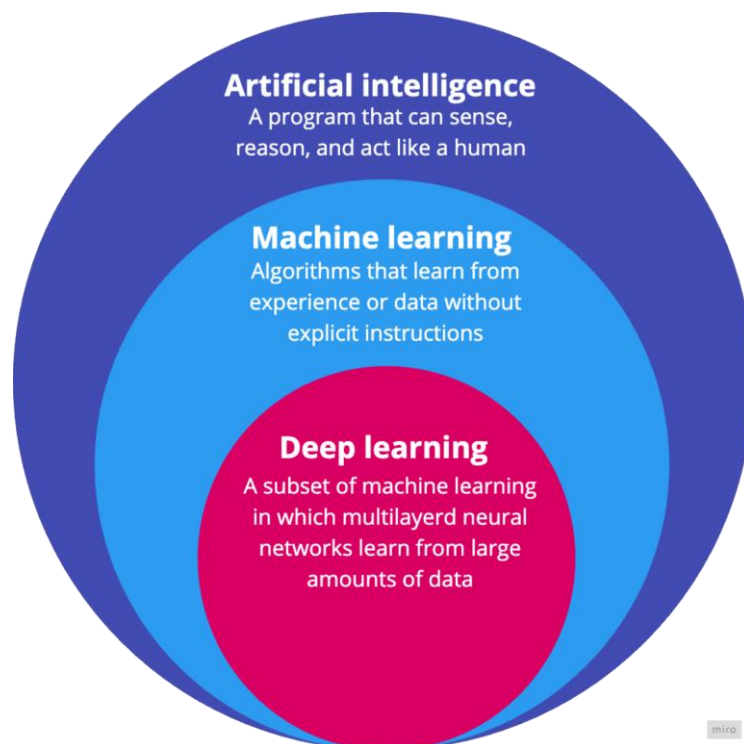


Figure 1-1: IA, Machine Learning et Deep Learning : relations et différences ^[12]

Le Deep Learning (apprentissage profond) représente une spécialisation du Machine Learning fondée sur des réseaux de neurones artificiels dotés de nombreuses couches successives ^[13].

Cette architecture permet de construire des représentations hiérarchiques des données, des caractéristiques les plus simples aux concepts les plus abstraits^[9].

Parmi les architectures du Deep Learning, on distingue :

- Les réseaux de neurones denses (DNN) , ou perceptrons multicouches (MLP), où chaque neurone d'une couche est relié à l'ensemble des neurones de la couche suivante, permettant de modéliser des relations non linéaires complexes ^[9].
- Les réseaux de neurones convolutifs (CNN) , spécialement conçus pour traiter des données structurées en grille comme les images, qui utilisent des filtres de convolution pour extraire des motifs locaux ^[14].
- Les réseaux de neurones récurrents (RNN) , adaptés aux données séquentielles telles que les séries temporelles ou le texte ^[9].

Ainsi, l'IA se présente comme un domaine pluridisciplinaire qui, par ses déclinaisons successives (Machine Learning, Deep Learning) et leurs architectures spécifiques, cherche à doter les machines de capacités cognitives proches de celles des humains ^[15].

2.2. Domaines d'application de l'IA

L'IA a connu un développement rapide et s'est diffusée dans l'ensemble des secteurs d'activité. Le tableau ci-dessous en donne un aperçu non exhaustif.

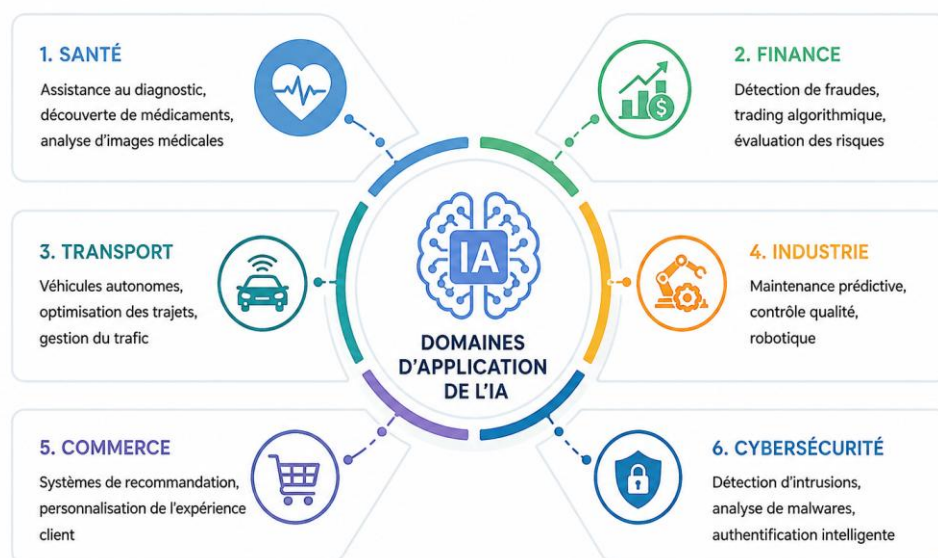


Figure 1-2: Domaines d'application de l'IA

Domaine	Applications caractéristiques
Santé	Assistance au diagnostic, découverte de médicaments, analyse d'images médicales
Finance	Détection de fraudes, trading algorithmique, évaluation des risques
Transport	Véhicules autonomes, optimisation des trajets, gestion du trafic
Industrie	Maintenance prédictive, contrôle qualité, robotique
Commerce	Systèmes de recommandation, personnalisation client
Cybersécurité	Détection d'intrusions, analyse de malwares, authentification

Tableau 1-1 : Domaines d'application de l'IA

2.3. Place de l'IA dans la cybersécurité

La cybersécurité est devenue un terrain d'application privilégié pour l'IA^{[1],[2]}. Plusieurs facteurs expliquent cette convergence :

- Le volume de données : Les infrastructures réseau produisent quotidiennement des quantités massives d'informations (journaux d'événements, flux réseau, alertes) que les analystes humains ne peuvent traiter exhaustivement ^[16].
- La complexité des attaques : Les cybermenaces contemporaines sont de plus en plus sophistiquées, polymorphes et évolutives, rendant insuffisantes les approches fondées sur des règles figées^[17].

- La rapidité d'exécution : Certaines attaques se propagent en quelques secondes, imposant des mécanismes de détection et de réaction automatisés ^[18].
- La pénurie de compétences : Le déficit d'experts en cybersécurité crée un besoin d'outils d'assistance et d'automatisation^[19].

L'IA intervient dans de multiples aspects de la cybersécurité ^[20],^[21]:

Application	Description
Détection d'intrusions	Identification de comportements anormaux sur le réseau ou les systèmes ^[22]
Analyse de malwares	Classification et étude du comportement des logiciels malveillants ^[23]
Authentification	Reconnaissance biométrique, analyse comportementale des utilisateurs ^[24]
Filtrage des courriels	Détection des spams et des tentatives d'hameçonnage ^[25]
Réponse aux incidents	Automatisation des premières actions de confinement ^[26]

Tableau 1-2: Principales applications de l'IA en cybersécurité

3. Apprentissage automatique (Machine Learning)

3.1. Principe du Machine Learning

Le Machine Learning est une sous-discipline de l'IA qui confère aux ordinateurs la capacité d'apprendre sans être explicitement programmés pour chaque situation ^[10]. Plutôt que de suivre des instructions prédéfinies, les algorithmes construisent un modèle à partir d'exemples, appelées données d'apprentissage, afin de réaliser des prédictions ou de prendre des décisions.

Le processus d'apprentissage peut être représenté comme suit :

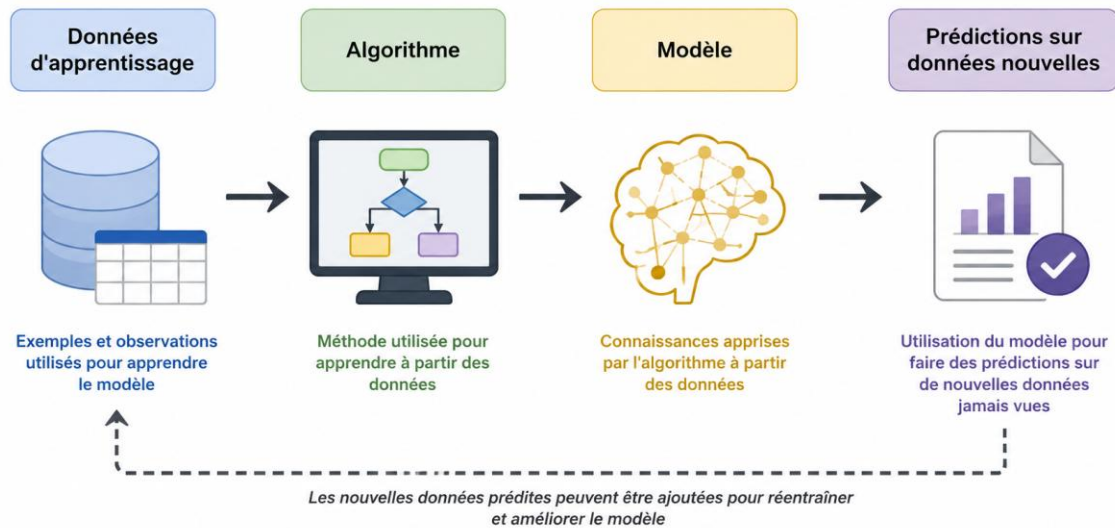


Figure 1-3 : Le processus d'apprentissage

L'objectif fondamental est que le modèle saisisse les régularités et les relations présentes dans les données d'apprentissage pour pouvoir généraliser à des situations inédites ^[11]

Cette capacité de généralisation est essentielle : un modèle performant ne doit pas se contenter de mémoriser les données d'apprentissage (phénomène de sur-apprentissage ou overfitting), mais doit pouvoir produire des résultats corrects sur des données jamais rencontrées ^[27].

3.2. Types d'apprentissage

On distingue classiquement trois paradigmes d'apprentissage selon la nature des données disponibles et l'objectif poursuivi ^[28].

3.2.1. Apprentissage supervisé

L'apprentissage supervisé est le paradigme le plus répandu. Il repose sur la disponibilité d'un ensemble de données étiquetées, où chaque exemple est associé à la réponse attendue ^[29].

Formellement, on dispose d'un ensemble d'apprentissage $\{(x_1, y_1), (x_2, y_2), \dots, (x_n, y_n)\}$ où :

- **x**: représente les caractéristiques (features) de l'exemple i
- **y**: représente l'étiquette (label) associée, soit la valeur à prédire

L'objectif est d'apprendre une fonction $f: x \rightarrow y$ qui, pour toute nouvelle observation x , prédise correctement son étiquette y ^[30].

L'apprentissage supervisé se divise en deux catégories principales :

1. Classification : l'étiquette à prédire est une valeur discrète (catégorie). Exemples : détection de spam, diagnostic médical, détection d'attaques réseau.

2. Régression : l'étiquette à prédire est une valeur continue. Exemples : prévision de prix, estimation de température.

Dans ce mémoire, nous nous intéressons à des problèmes de classification : attribuer à chaque flux réseau une catégorie (trafic normal ou type d'attaque) ^[31].

3.2.2. Apprentissage non supervisé

L'apprentissage non supervisé traite des données non étiquetées. L'objectif est de découvrir des structures cachées, des regroupements naturels ou des régularités dans les données ^[32].

Les principales tâches sont :

- **Clustering** : partition des données en groupes homogènes (algorithmes K-means, DBSCAN)
- **Réduction de dimensionnalité** : projection des données dans un espace de plus faible dimension (ACP, t-SNE)
- **Détection d'anomalies** : identification des observations s'écartant du comportement normal

3.2.3. Apprentissage semi-supervisé

L'apprentissage semi-supervisé combine un petit ensemble de données étiquetées avec un grand ensemble de données non étiquetées. Il exploite la structure révélée par les données non étiquetées pour améliorer l'apprentissage à partir des quelques exemples étiquetés^[33]. Cette approche est pertinente lorsque l'étiquetage est coûteux mais que les données brutes sont abondantes.

4. Deep Learning

4.1. Principe des réseaux de neurones

Le Deep Learning est une branche du Machine Learning qui utilise des réseaux de neurones artificiels comportant de nombreuses couches^[13]. Inspirés par l'architecture du cerveau biologique, ces modèles ont obtenu des résultats remarquables en vision par ordinateur, traitement du langage naturel et reconnaissance vocale.

L'idée fondamentale est de construire des représentations hiérarchiques : les premières couches capturent des caractéristiques simples (contours, textures), les couches intermédiaires

combinent ces éléments pour former des motifs plus complexes, et les couches finales identifient des concepts de haut niveau^[9].

4.2. Architecture d'un réseau de neurones

Un réseau de neurones artificiels est composé de neurones organisés en couches successives.

Le neurone artificiel : unité de calcul de base recevant plusieurs entrées ($x_1, x_2, \dots, x_n$), chacune associée à un poids ($w_1, w_2, \dots, w_n$). Il calcule une somme pondérée des entrées, ajoute un biais (b), puis applique une fonction d'activation pour produire une sortie :

$$\text{Sortie} = f(\sum(w_i \times x_i) + b)$$

Organisation en couches :

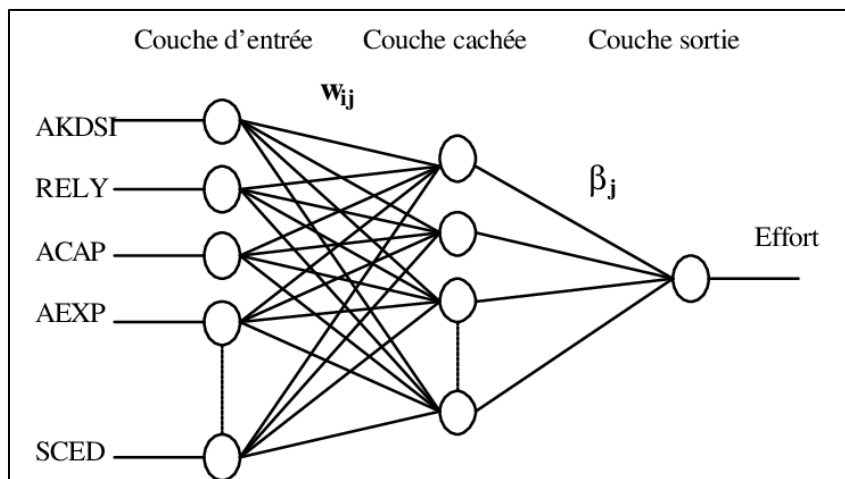


Figure 1-4 : Organisation en couches^[34]

- **Couche d'entrée** : reçoit les données brutes
- **Couches cachées** : effectuent les transformations successives
- **Couche de sortie** : produit la prédiction finale

4.3. Fonctions d'activation

Les fonctions d'activation introduisent la non-linéarité, essentielle pour apprendre des relations complexes^[9].

Fonction	Formule	Caractéristiques
Sigmoïde	$f(x) = 1/(1 + e^{-x})$	Utile pour la classification binaire, risque de gradient vanishing
Tanh	$f(x) = (e^x - e^{-x})/(e^x + e^{-x})$	Centrée en zéro, gradient vanishing persistant
ReLU	$f(x) = \max(0, x)$	Simple, efficace, atténue le vanishing gradient
Softmax	$f(x_i) = e^{x_i} / \sum e^{x_j}$	Utilisée en sortie pour la classification multiclasse

Tableau 1-3 : Fonctions d'activation et leurs caractéristiques

4.4. Fonction de perte et optimisation

Fonction de perte : mesure l'écart entre les prédictions et les valeurs réelles^[35]. Les fonctions courantes incluent l'erreur quadratique moyenne (MSE) pour la régression, et l'entropie croisée pour la classification.

Rétropropagation : algorithme calculant le gradient de la fonction de perte par rapport à chaque poids, en propageant l'erreur de la sortie vers l'entrée^[36].

Optimisation : les poids sont mis à jour pour réduire la perte. L'algorithme de base est la descente de gradient stochastique (SGD). Des variantes comme Adam ajustent dynamiquement le taux d'apprentissage^[37].

5. Modèles utilisés pour la détection d'attaques

La détection d'intrusions réseau consiste à analyser des flux réseau caractérisés par un ensemble de features numériques (durée, nombre de paquets, intervalles inter-arrivées, flags TCP, etc.) afin de distinguer un trafic normal d'un trafic malveillant. Plusieurs familles d'algorithmes d'apprentissage automatique et profond ont démontré leur efficacité sur cette tâche. Dans ce travail, nous retenons les modèles les plus représentatifs de la littérature :

Le perceptron multicouche (MLP), l'autoencodeur, le LSTM, la forêt aléatoire (Random Forest), XGBoost, LightGBM ainsi que les machines à vecteurs de support (SVM). Chacun est présenté avec son principe de fonctionnement, ses avantages et ses limites dans le contexte spécifique de la détection d'attaques réseau.

5.1. Réseau de neurones Dense (MLP)

Le perceptron multicouche (MLP) est la forme la plus simple de réseau profond, composé de couches denses entièrement connectées^[9].

- **Architecture typique** : couche d'entrée (nombre de caractéristiques), couches cachées (ReLU), couche de sortie (softmax pour la classification).
- **Avantages** : simplicité, capacité à modéliser des relations non linéaires, facilité d'implémentation.
- **Inconvénients** : risque de sur-apprentissage, nombre de paramètres potentiellement élevé.

5.2. Autoencoder

L'autoencoder est un réseau conçu pour apprendre une représentation compressée des données, puis reconstruire l'entrée à partir de cette représentation^[38].

Architecture : encodeur (réduction de dimension), code (représentation latente), décodeur (reconstruction).

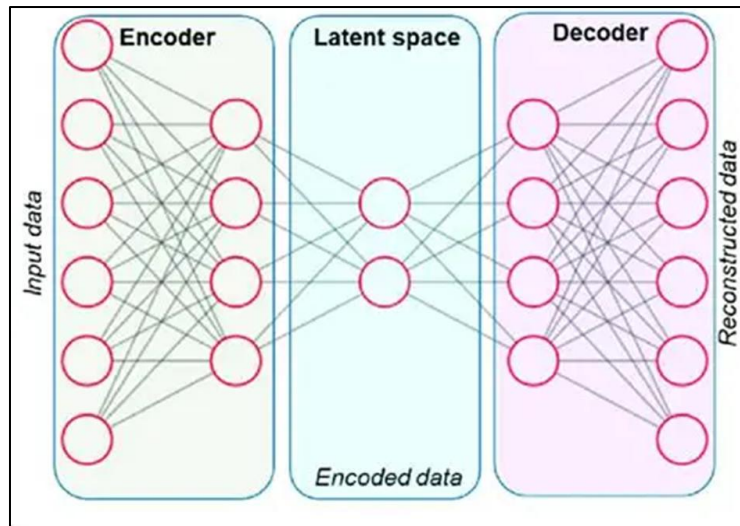


Figure 1-5: Architecture de l'autoencodeur ^[39]

Application à la détection : entraîné uniquement sur du trafic normal, l'erreur de reconstruction est faible pour le trafic normal et élevée pour les attaques^[40].

5.3. LSTM (Long Short-Term Memory)

Les LSTM sont une variante des réseaux récurrents conçus pour capturer des dépendances à long terme dans les séquences temporelles^[41].

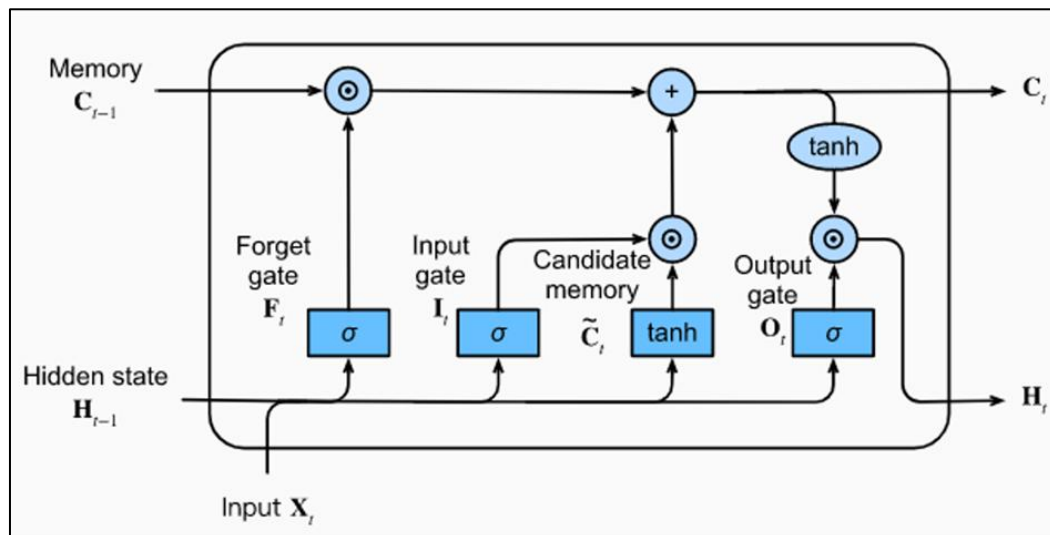


Figure 1-6: Architecture du modèle LSTM. ^[42]

- **Principe** : maintien d'un état interne mis à jour via des portes contrôlant le flux d'information (oubli, stockage, sortie).

- **Application** : le trafic réseau ayant une dimension temporelle, les LSTM peuvent modéliser les patterns séquentiels des attaques^[40].

5.4. Random Forest

Le Random Forest est une méthode d'ensemble combinant de nombreux arbres de décision^[41].

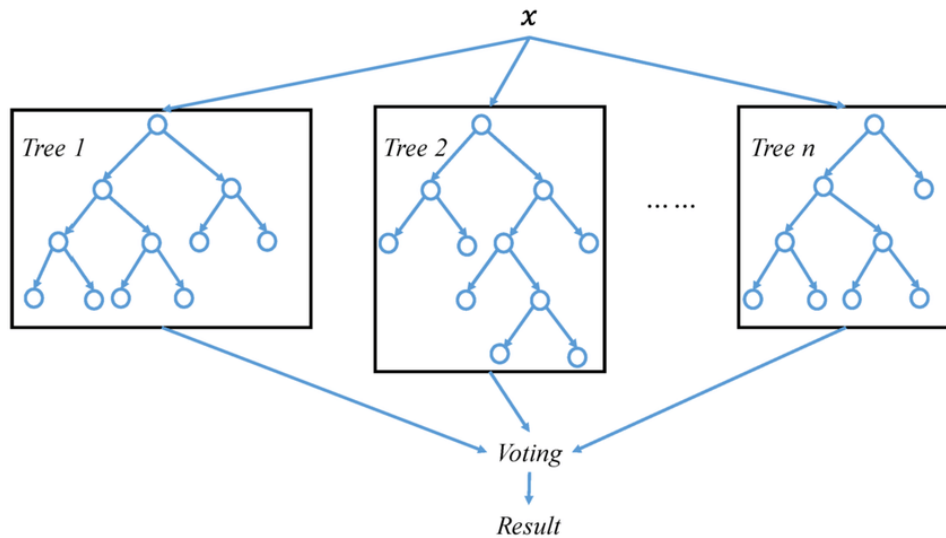


Figure 1-7: Architecture du modèle Random Forreest ^[43]

- **Principe** : création d'arbres sur des échantillons bootstrap, sélection aléatoire de caractéristiques à chaque nœud, agrégation par vote majoritaire.
- **Application** : largement utilisé pour la détection d'intrusions en raison de sa robustesse et de sa capacité à traiter de grandes dimensions^[41].

5.5. XGBoost (Extreme Gradient Boosting)

XGBoost est un algorithme d'apprentissage automatique basé sur le principe du gradient boosting^[44]. Contrairement à Random Forest qui construit ses arbres en parallèle, XGBoost construit ses arbres de manière séquentielle : chaque nouvel arbre est entraîné pour corriger les erreurs commises par l'ensemble des arbres précédents^[45].

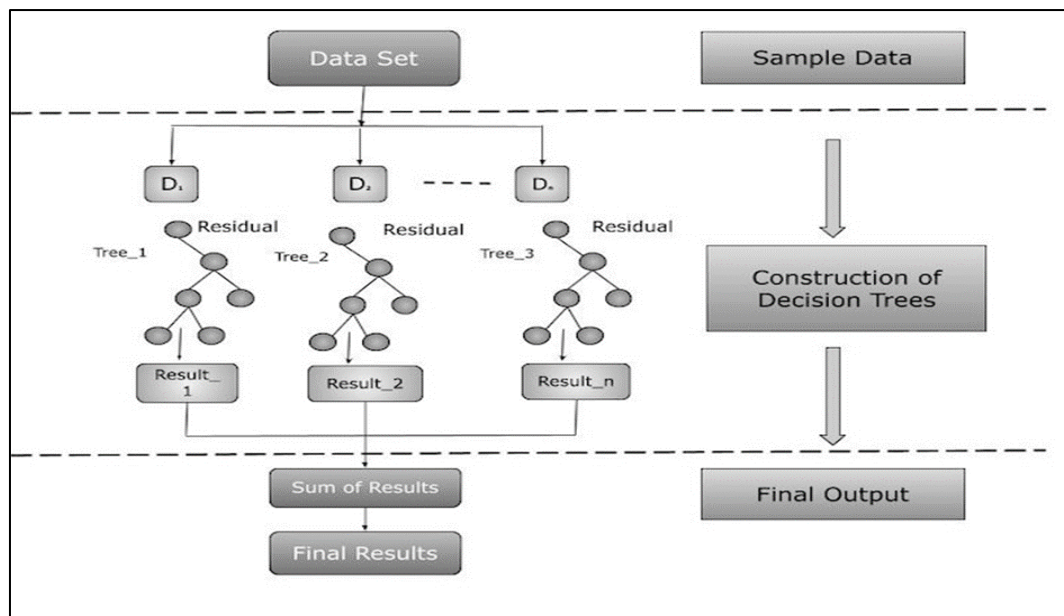


Figure 1-8: Architecture du modèle XGBoost ^[46]

- **Principe** : À chaque étape, le modèle ajoute un arbre qui se concentre sur les exemples mal prédits par le modèle actuel. La prédiction finale est la somme pondérée des prédictions de tous les arbres^[47].
- **Application à la détection d'attaques** : XGBoost est largement utilisé pour la détection d'intrusions en raison de sa haute précision et de sa capacité à gérer des données déséquilibrées^[48].

Avantages ^[49] :

- Très haute précision sur les données tabulaires comme CIC-IDS 2017 (Canadian Institute for Cybersecurity - Intrusion Detection System)
- Régularisation intégrée (L1 et L2) pour éviter le sur-apprentissage
- Gestion native des valeurs manquantes
- Rapide grâce à l'optimisation et à la parallélisation

Inconvénients ^[50] :

- Taille du modèle potentiellement importante
- Temps d'entraînement plus long que Random Forest
- Nécessite un réglage minutieux des hyperparamètres

5.6. LightGBM (Light Gradient Boosting Machine)

LightGBM est un algorithme de gradient boosting développé par Microsoft qui se distingue par sa rapidité d'exécution et sa faible consommation mémoire^[51]. Contrairement à XGBoost qui construit des arbres niveau par niveau (level-wise), LightGBM construit ses arbres feuille par feuille (leaf-wise), ce qui lui permet de converger plus rapidement vers une solution optimale^[51].

Principe : LightGBM utilise deux innovations majeures : l'algorithme GOSS (Gradient-based One-Side Sampling) qui ne conserve que les échantillons avec les plus grands gradients, et l'algorithme EFB (Exclusive Feature Bundling) qui regroupe les caractéristiques exclusives pour réduire la dimensionnalité^[51].

Application à la détection d'attaques : LightGBM est particulièrement adapté aux grands volumes de données comme CIC-IDS 2017 grâce à sa rapidité d'entraînement et sa faible empreinte mémoire^[52].

Avantages ^[51] :

- Très rapide à l'entraînement et à l'inférence
- Faible consommation mémoire (modèle compact ~8MB)
- Haute précision comparable à XGBoost
- Gestion native des valeurs manquantes

Inconvénients ^[53] :

- Risque de sur-apprentissage si les paramètres ne sont pas bien réglés (leaf-wise)
- Moins performant sur les très petits jeux de données
- Nécessite un réglage du paramètre num_leaves

6. Conclusion

Ce premier chapitre a jeté les bases nécessaires pour comprendre comment l'intelligence artificielle peut être utilisée dans la détection d'attaques sur les réseaux. Nous avons commencé par définir ce qu'est l'IA et comment elle s'organise, en insistant sur son rôle de plus en plus important dans le domaine de la cybersécurité. Ensuite, nous avons expliqué les grands principes de l'apprentissage automatique, en faisant la différence entre les approches supervisées, non supervisées et semi-supervisées. La partie sur l'apprentissage profond nous a permis de comprendre le fonctionnement des réseaux de neurones et de leurs éléments essentiels. Nous avons également passé en revue les modèles les plus souvent utilisés pour détecter les intrusions, comme le MLP, l'autoencodeur, le LSTM, Random forest XGboost et lightGBM en soulignant les points forts et les limites de chacun. Pour finir, nous avons présenté les principales métriques qui permettent de mesurer la performance d'un modèle, telles que l'accuracy, la précision, le rappel et le F1-score. Toutes ces notions théoriques nous serviront de point de départ pour aborder, dans le chapitre qui suit, le monde de la cybersécurité et des systèmes de détection d'intrusion

Chapitre II:
*Cybersécurité et systèmes de détection
d'intrusion (IDS)*

1. Introduction

Ce chapitre est consacré à la cybersécurité et aux systèmes de détection d'intrusion. Il a pour objectif de poser le cadre applicatif dans lequel s'inscrit notre travail.

Nous commencerons par définir ce qu'est la cybersécurité et quels sont ses objectifs fondamentaux, à savoir la protection de la confidentialité, de l'intégrité et de la disponibilité des données.

Ensuite, nous passerons en revue les principales menaces qui pèsent sur les réseaux informatiques. Seront abordées les attaques par déni de service, le balayage de ports, les tentatives de force brute ainsi que les injections de code malveillant comme les attaques SQL ou XSS.

Nous présenterons également les systèmes de détection d'intrusion, communément appelés IDS. Nous expliquerons leur rôle, leur fonctionnement et les différentes approches existantes, en particulier la détection par signatures et la détection par anomalies. Cette dernière, qui repose sur l'analyse des écarts par rapport à un comportement normal, est celle qui nous intéresse le plus. Nous verrons aussi comment l'intelligence artificielle peut améliorer ces systèmes en apportant plus de précision, plus de rapidité et une meilleure capacité à s'adapter aux menaces nouvelles.

Enfin, nous aborderons le concept d'Edge Computing, qui consiste à traiter les données au plus près de leur source, ainsi que la plateforme Raspberry Pi, qui servira de support matériel pour notre système.

L'ensemble de ces éléments nous permettra de comprendre l'environnement dans lequel notre solution va être déployée.

2. Fondamentaux de la cybersécurité

2.1. Définition et objectifs de la cybersécurité

La cybersécurité désigne l'ensemble des pratiques, technologies et processus visant à protéger les réseaux, appareils, programmes et données contre les attaques, dommages ou accès non autorisés ^[54].

Les objectifs fondamentaux de la cybersécurité s'articulent autour de trois piliers ^[55] :

Confidentialité : garantir que seules les personnes autorisées accèdent aux informations

Intégrité : assurer que les données ne sont pas altérées de manière frauduleuse

Disponibilité : garantir l'accès aux services et données pour les utilisateurs légitimes

Ces trois objectifs, souvent désignés par l'acronyme CIA (Confidentiality, Integrity, Availability), constituent le fondement de toute politique de sécurité.

2.2. Typologie et classification des menaces réseau

Les réseaux informatiques font face à une diversité de menaces que l'on peut classer selon plusieurs critères ^[56] :

Selon l'origine :

- **Menaces internes** : provenant de l'intérieur de l'organisation (employés, négligence).
- **Menaces externes** : émanant d'acteurs extérieurs (pirates, groupes criminels, États).

Selon l'intention :

- **Attaques aveugles** : lancées sans cible prédéfinie via des scripts automatisés.
- **Attaques ciblées** : soigneusement planifiées contre une organisation spécifique.

Selon le mode opératoire :

- **Attaques passives** : observation et collecte d'informations sans perturbation.
- **Attaques actives** : modification des données, perturbation des services, tentatives de pénétration.

3. Les différents types d'attaque sur un réseau

3.1. Attaques par déni de service (DoS/DDoS)

Les attaques par déni de service visent à rendre un service indisponible en le submergeant de requêtes jusqu'à épuisement de ses ressources^[57]. Mécanismes courants :

- **Inondation SYN** : exploitation du protocole TCP par envoi massif de requêtes SYN sans finalisation des connexions.
- **Inondation UDP** : envoi de nombreux paquets UDP vers des ports aléatoires.
- **Attaque par amplification** : utilisation de serveurs tiers pour multiplier le trafic vers la cible.

Les attaques DDoS (distribuées) utilisent un réseau de machines compromises (botnet) pour lancer l'attaque depuis de multiples sources simultanément^[58].

3.2. Reconnaissance et scan de ports:

Le scan de ports est une phase préliminaire permettant à l'attaquant d'identifier les services actifs sur une cible^[59]. Techniques de scan :

- **Scan SYN (furtif)** : envoi de paquets SYN sans finalisation de la connexion.
- **Scan TCP Connect** : établissement complet de la connexion.
- **Scan UDP** : détection des services UDP par envoi de paquets.
- **Scans FIN/NULL/Xmas** : utilisation de combinaisons de flags inhabituelles.

3.3. Attaques par force brute:

Ces attaques consistent à tester systématiquement des combinaisons de mots de passe jusqu'à trouver la bonne^[60]. Variantes :

- **Force brute pure** : test de toutes les combinaisons possibles.
- **Attaque par dictionnaire** : test de mots de passe courants.
- **Credential stuffing** : réutilisation d'identifiants volés sur d'autres services.

3.4. Attaques par injection:

Les attaques par injection exploitent la confiance excessive accordée aux données fournies par l'utilisateur^[61].

- **Injection SQL** : insertion de commandes SQL dans les champs de saisie pour manipuler la base de données.
- **Injection de commandes** : injection de commandes système exécutées avec les privilèges de l'application.
- **Cross-site Scripting (XSS)** : injection de scripts malveillants dans des pages web consultées par d'autres utilisateurs.

4. Les systèmes de détection d'intrusion (IDS)

4.1. Définition et mission

Un système de détection d'intrusion (IDS) est un dispositif surveillant en continu l'activité d'un réseau ou d'un système pour y détecter des comportements suspects ou malveillants^[62]. Contrairement au pare-feu qui filtre le trafic selon des règles, l'IDS observe et signale les anomalies^[63].

Les fonctions essentielles d'un IDS sont :

Les fonctions essentielles d'un IDS comprennent la surveillance continue, l'analyse en temps réel, la génération d'alertes, la journalisation des événements et la production de rapports.

4.2. Architecture fonctionnelle

Un IDS typique s'organise en plusieurs composants ^[64] :

- **Capteurs** : collectent les données brutes (interfaces réseau en mode promiscuous ou agents logiciels)
- **Moteur d'analyse** : applique les règles de détection (signatures ou anomalies)
- **Module de réaction** : génère alertes, journalise les événements, déclenche des actions
- **Console de gestion** : interface de configuration et de visualisation

5. Approches de détection

5.1. Détection par signatures

La détection par signatures, également appelée détection par correspondance de motifs (pattern matching), repose sur la comparaison systématique du trafic réseau observé avec une base de données contenant les empreintes caractéristiques des attaques connues ^[65].

Chaque signature correspond à une séquence d'octets spécifique, une combinaison particulière de flags TCP, ou un comportement réseau distinctif préalablement identifié et répertorié.

Lorsqu'un paquet ou un flux réseau correspond à l'une de ces signatures, le système génère une alerte. Cette approche s'apparente au fonctionnement des antivirus traditionnels, qui scrutent les fichiers à la recherche de code malveillant déjà répertorié.

Avantages :

- Faible taux de faux positifs : les alertes déclenchées correspondent effectivement à des attaques référencées.
- Détection rapide : la comparaison par signatures est peu coûteuse en termes de calcul et peut s'effectuer en temps réel.

Inconvénients :

- Incapacité à détecter les attaques nouvelles (zero-day) : toute menace dont la signature n'est pas encore enregistrée passe inaperçue.
- Nécessité de mises à jour fréquentes : la base de signatures doit être régulièrement enrichie pour rester efficace face à l'évolution des menaces.

Snort, l'un des IDS open source les plus répandus, illustre parfaitement cette approche. Il utilise un langage de règles permettant de décrire des signatures complexes et d'y associer des actions spécifiques (alerte, journalisation, rejet du paquet)^[66].

5.2. Détection par anomalies

La détection par anomalies établit un profil du comportement normal et signale les écarts^[67].

Le processus comporte deux phases :

- **Apprentissage** : construction du profil comportemental sur du trafic sain
- **Détection** : comparaison du trafic observé avec le profil de référence

Avantages : détection des attaques inconnues, adaptabilité

Inconvénients : taux de faux positifs potentiellement élevé, phase d'apprentissage critique

C'est dans cette catégorie que s'inscrivent les approches par IA présentées au chapitre 1^[68].

5.3. Approches hybrides

Les IDS hybrides combinent signatures et anomalies pour bénéficier des avantages des deux méthodes^[69]. Dans une architecture hybride typique, le trafic passe d'abord par une analyse par signatures, puis les flux non identifiés sont soumis à une analyse comportementale.

6. Apport de l'intelligence artificielle à la détection d'intrusions

6.1. Pertinence et atouts de l'intelligence artificielle

L'intégration de l'IA dans les IDS répond à plusieurs besoins ^[70] :

- **Volume de données** : les réseaux génèrent des téraoctets de trafic que l'analyse manuelle ne peut traiter
- **Sophistication des attaques** : les techniques d'évasion et le chiffrement rendent les approches traditionnelles inefficaces
- **Rapidité d'exécution** : certaines attaques se propagent en quelques minutes
- **Pénurie de compétences** : le manque d'analystes qualifiés nécessite des outils d'assistance

6.2. Améliorations apportées par l'IA aux systèmes de détection d'intrusions

Les modèles d'apprentissage automatique apportent des améliorations tangibles ^[71] :

- **Détection de patterns complexes** : identification d'attaques lentes réparties dans le temps

- **Adaptation dynamique** : réentraînement périodique sur les données récentes
- **Hiérarchisation des alertes** : association de scores de confiance
- **Réduction des faux positifs** : meilleure distinction entre variations normales et anomalies

6.3. Limites et défis de l'application de l'IA à la détection d'intrusions

L'application de l'IA à la détection d'intrusion présente des difficultés ^[72] :

- **Qualité des données** : nécessité de données étiquetées représentatives
- **Équilibre des classes** : rareté des attaques par rapport au trafic normal
- **Explicabilité** : compréhension des décisions du modèle
- **Évolution des menaces** : adaptation aux attaques zero-day

7. Infrastructure et déploiement en périphérie de réseau (Edge Computing)

7.1. Définition et principes fondamentaux de l'Edge Computing

L'Edge Computing, ou informatique en périphérie, désigne un paradigme architectural qui consiste à rapprocher le traitement des données de leur source de production, en s'affranchissant d'une dépendance exclusive aux infrastructures cloud centralisées^[73].

Contrairement au modèle traditionnel où l'intégralité des données est transmise vers des centres de calcul distants pour y être analysées et stockées, l'Edge Computing propose une répartition intelligente des tâches de traitement entre les dispositifs de bord (capteurs, routeurs, passerelles) et le cloud.

Cette approche répond aux trois principales limitations du modèle cloud traditionnel :

- **Latence** : Dans un modèle cloud, le temps d'aller-retour (aller-retour) entre la source de données et le centre de calcul peut atteindre plusieurs centaines de millisecondes, rédhibitoire pour des applications nécessitant une réaction en temps réel (inférieure à 10 ms).
- **Bande passante** : La transmission massive de données (vidéos, flux réseau, logs) vers le cloud consomme d'importantes ressources réseau et peut saturer les liaisons, en particulier dans les zones à faible débit.
- **Confidentialité** : L'envoi de données sensibles (médicales, financières, industrielles) vers une infrastructure tierce soulève des problématiques de souveraineté des données, de conformité réglementaire (RGPD) et de risques d'interception.

L'Edge Computing se décline en plusieurs niveaux de proximité avec la source des données :

Niveau	Description	Exemple
Périphérique (device edge)	Traitement directement sur le capteur ou l'appareil	Raspberry pi, smartphone, caméra connectée
Passerelle (gateway edge)	Traitement sur une passerelle locale regroupant plusieurs appareils	Routeur industriel, box domotique
Cloud edge	Traitement sur des serveurs en bordure du cloud (région proche)	Zones de disponibilité aws local

Tableau 2-1 : Fonctions d'activation et leurs caractéristiques

Dans le cadre de ce mémoire, le déploiement s'effectue au niveau périphérique via un Raspberry Pi 5 assurant la capture et l'analyse locale du trafic réseau.

7.2. Intérêt pour la détection d'intrusion

L'application de l'Edge Computing à la cybersécurité présente plusieurs avantages ^[74] :

- **Latence minimale** : détection en temps réel sans délais de transmission
- **Bande passante préservée** : transmission uniquement des alertes et métriques agrégées
- **Confidentialité renforcée** : données sensibles conservées localement
- **Résilience** : fonctionnement continu même sans connectivité cloud
- **Scalabilité** : traitement distribué sans dimensionnement central

7.3. Le Raspberry Pi comme plateforme Edge

Le Raspberry Pi, nano-ordinateur initialement conçu pour l'éducation, constitue une plateforme idéale pour l'Edge Computing^[75].

Ses caractéristiques le rendent adapté à notre expérimentation :

- Faible coût permettant des déploiements à grande échelle
- Consommation électrique réduite

- Communauté active et distributions Linux variées
- Interface réseau directe pour la capture du trafic

Les ressources limitées (processeur modeste, RAM contrainte) imposent des choix d'optimisation qui feront l'objet du chapitre suivant.

8. Conclusion

Ce deuxième chapitre a posé le cadre applicatif de notre travail. Nous avons exploré les principales menaces réseau comme les attaques par déni de service, le balayage de ports, les tentatives de force brute et les injections SQL, illustrant ainsi la complexité de la tâche de détection. Les systèmes de détection d'intrusion ont été présentés avec leurs deux grandes approches : la détection par signatures, fiable mais limitée aux menaces connues, et la détection par anomalies, plus flexible mais sujette aux faux positifs. L'apport de l'intelligence artificielle apparaît comme une évolution naturelle pour répondre aux défis du volume de données, de la sophistication des attaques et de la détection en temps réel. Enfin, le paradigme de l'Edge Computing offre un cadre de déploiement adapté, et le Raspberry Pi constitue un support idéal pour valider la faisabilité d'un IDS intelligent et décentralisé. Le chapitre suivant mettra en œuvre ces concepts à travers la réalisation pratique d'un système complet.

Chapitre III:
*Conception et entraînement du modèle d'IA
pour détection d'intrusion*

1. Introduction

Ce chapitre décrit le processus de conception, d'entraînement et d'évaluation des modèles d'intelligence artificielle destinés à la détection des intrusions réseau. Il présente d'abord l'architecture globale de la solution ainsi que le pipeline de traitement mis en œuvre. Ensuite, le dataset CIC-IDS 2017 utilisé pour l'apprentissage est introduit, suivi des différentes opérations de prétraitement appliquées aux données afin d'en améliorer la qualité et la pertinence. Enfin, les différents modèles d'intelligence artificielle sont entraînés, évalués et comparés à l'aide de plusieurs métriques de performance, dans le but d'identifier le modèle le plus performant et le plus adapté à un déploiement sur Raspberry Pi 5.

2. Architecture Globale du Système

2.1. Description de l'Architecture Proposée

Le système de détection d'intrusions (IDS) proposé dans ce travail repose sur une architecture modulaire et multicouche, conçue pour analyser le trafic réseau et identifier les comportements malveillants en temps réel. Cette architecture s'articule autour de trois composantes principales: la collecte et le prétraitement des données, la couche de détection basée sur des modèles d'apprentissage automatique, et l'interface de visualisation déployée sur Raspberry Pi 5.

L'architecture adoptée suit une approche hybride combinant deux paradigmes complémentaires:

Détection supervisée : Cette approche utilise des modèles entraînés sur des données étiquetées afin de classifier les flux réseau en sept catégories distinctes : trafic normal, DoS, DDoS, Port Scanning, Brute Force, Bots et Web Attacks. Les modèles supervisés exploités se divisent en deux familles :

- **Modèles d'apprentissage automatique classique (Machine Learning) :** Random Forest, XGBoost et LightGBM. Ces modèles reposent sur des structures arborescentes et des méthodes d'ensemble, reconnus pour leurs excellentes performances sur les données tabulaires et leur faible coût computationnel lors de l'inférence.
- **Modèles d'apprentissage profond (Deep Learning) :** DNN (Deep Neural Network), LSTM (Long Short-Term Memory) et BLSTM (Bidirectional LSTM). Ces modèles sont basés sur des architectures de réseaux de neurones artificiels capables d'apprendre des représentations hiérarchiques complexes à partir des données brutes.

Détection non supervisée (basée sur les anomalies) : Cette approche repose sur des autoencodeurs (AE) et des autoencodeurs variationnels (VAE) entraînés exclusivement sur le trafic normal. Tout flux présentant une erreur de reconstruction élevée est signalé comme une anomalie potentielle, permettant ainsi la détection d'attaques inconnues ou zero-day.

L'ensemble du pipeline, de la collecte des données jusqu'au déploiement, est illustré dans la figure suivante :

Pipeline Globale			
Collecte CIC-IDS 2017	Prétraitement & Feature Selection	Entraînement des Modèles ML/DL/AE	Déploiement Raspberry Pi 5

Tableau 3-1 : Pipeline Globale

Pipeline expérimental (dataset)

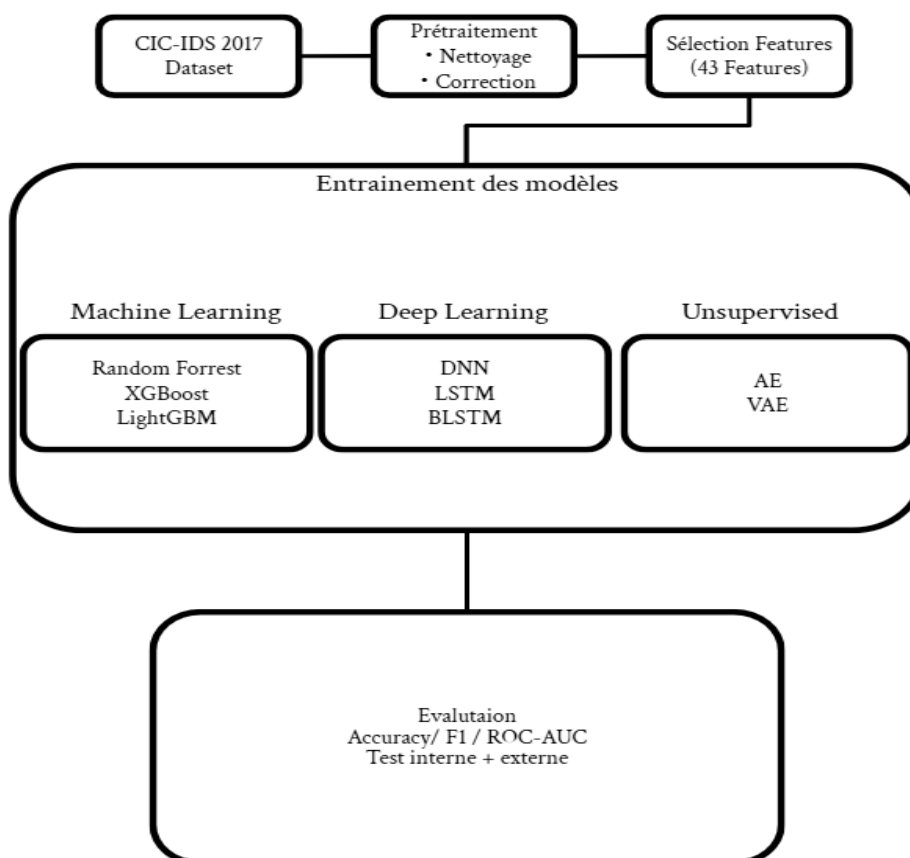


Figure 3-1 : Pipeline Expérimental (dataset)

2.2. Pipeline de Détection

Le pipeline de détection suit les étapes suivantes :

Étape 1: Collecte des données : Les données brutes sont extraites du dataset CIC-IDS 2017, fourni par le Canadian Institute for Cybersecurity. Chaque fichier CSV contient des flux réseau caractérisés par 80+ features calculées par CIC FlowMeter.

Étape 2: Prétraitement : Les données subissent un nettoyage approfondi incluant la suppression des valeurs infinies et manquantes, l'élimination des doublons, la correction des labels et la normalisation des features numériques.

Étape 3: Sélection des features : Une sélection basée sur la corrélation et la quasi-constance réduit le nombre de features de 80+ à 43 features pertinentes.

Étape 4: Entraînement : Six modèles supervisés et quatre modèles non supervisés sont entraînés et comparés selon plusieurs métriques de performance.

Étape 5: Évaluation et déploiement : Le meilleur modèle (XGBoost) est déployé sur Raspberry Pi 5 avec un tableau de bord Flask permettant la détection en temps réel.

3. Choix du Dataset

3.1. Présentation du Dataset CIC-IDS 2017

Le dataset CIC-IDS 2017 a été développé par Sharafaldin et al. à l'Université du Nouveau-Brunswick^[76]. Il constitue l'un des benchmarks les plus utilisés dans la recherche sur la détection d'intrusions réseau, ayant été cité dans des centaines de travaux scientifiques depuis sa publication.

Ce dataset a été généré dans un environnement réseau contrôlé simulant un réseau d'entreprise réel comportant des machines victimes et des machines attaquantes. La capture du trafic a été effectuée sur une période de cinq jours ouvrables (du lundi 3 juillet au vendredi 7 juillet 2017), durant laquelle différents types d'attaques ont été lancés de manière programmée.

Les principales caractéristiques de ce dataset sont :

- **Volume** : Plus de 2,8 millions de flux réseau répartis sur plusieurs fichiers CSV
- **Features** : 80 features extraites par l'outil CIC FlowMeter, incluant des statistiques sur les durées de flux, les tailles de paquets, les intervalles inter-arrivées (IAT) et les flags TCP
- **Étiquetage** : Chaque flux est étiqueté avec son type (BENIGN ou type d'attaque)

- **Réalisme** : Le trafic normal a été généré en reproduisant des comportements utilisateurs réels (navigation web, emails, téléchargements, streaming)

3.2. Description des Caractéristiques du Dataset

Le dataset CIC-IDS 2017 contient 80+ features calculées à partir des flux réseau bidirectionnels. Ces features peuvent être regroupées en plusieurs catégories :

Features temporelles :

- Flow Duration : durée totale du flux en microsecondes
- Flow IAT Mean/Std/Max/Min : statistiques des intervalles inter-arrivées
- Fwd/Bwd IAT Total/Mean/Std/Max/Min : IAT dans chaque direction

Features volumétriques :

- Total Fwd/Bwd Packets : nombre de paquets dans chaque direction
- Total Length of Fwd/Bwd Packets : taille totale des paquets
- Flow Bytes/s, Flow Packets/s : débits du flux

Features de taille de paquets :

- Fwd/Bwd Packet Length Max/Min/Mean/Std : statistiques des tailles
- Packet Length Mean/Variance : statistiques globales

Features de flags TCP :

- FIN, SYN, RST, PSH, ACK, URG Flag Count : compteurs de flags

Features de fenêtre TCP :

- Init_Win_bytes_forward/backward : taille initiale de fenêtre TCP

Après la phase de sélection des features (détaillée en section 3.3.2), **43 features** ont été retenues pour l'entraînement des modèles.

3.3. Sélection des Attaques Étudiées

Le dataset CIC-IDS 2017 contient initialement 15 types d'attaques différents. Après analyse de la distribution des classes et suppression des classes sous-représentées (Infiltration : 36 échantillons, Heartbleed : 11 échantillons), les attaques ont été regroupées en 6 catégories fonctionnelles :

Catégorie	Attaques originales	Description
DoS	DoS Hulk, DoS GoldenEye, DoS Slowloris, DoS Slowhttptest	Attaques par déni de service visant à épuiser les ressources d'un serveur
DDoS	DDoS	Attaque par déni de service distribué
Port Scanning	PortScan	Reconnaissance réseau par balayage de ports
Brute Force	FTP-Patator, SSH-Patator	Attaques par force brute sur les protocoles FTP et SSH
Bots	Bot	Trafic généré par des logiciels malveillants automatisés
Web Attacks	Web Attack Brute Force, XSS, SQL Injection	Attaques applicatives sur les serveurs web

Tableau 3-2 : Type d'attaques existant dans Dataset CIC IDS 2017

La distribution finale du dataset après regroupement et équilibrage est présentée dans le tableau suivant :

Classe	Nombre d'échantillons (train)	Pourcentage
Normal Traffic	500,000	52.7%
DoS	193,745	20.4%
DDoS	128,014	13.5%

Port Scanning	90,694	9.6%
Bots	12,000	1.3%
Web Attacks	12,000	1.3%
Brute Force	12,000	1.3%

Tableau 3-3 : La distribution finale du dataset après traitement

4. Prétraitement des Données

Le prétraitement constitue une étape cruciale dans tout pipeline d'apprentissage automatique appliqué à la détection d'intrusions. Un prétraitement inadéquat peut conduire à des modèles biaisés ou incapables de généraliser^[77]. Notre pipeline de prétraitement comprend les étapes suivantes.

4.1. Nettoyage des Données

Chargement des données : Les fichiers CSV du dataset CIC-IDS 2017 sont chargés avec gestion automatique de l'encodage (UTF-8, Latin-1, CP1252) pour traiter les problèmes de caractères spéciaux présents dans certains fichiers.

Traitement des valeurs aberrantes :

- Remplacement des valeurs infinies ($\pm\infty$) par NaN puis suppression
- Suppression des lignes contenant des valeurs NaN
- Suppression des doublons exacts

Correction des labels : Plusieurs incohérences ont été identifiées dans les labels originaux du dataset :

- 'Web Attack Brute Force' → 'Web Attack - Brute Force'
- 'Web Attack XSS' → 'Web Attack - XSS'
- 'Web Attack Sql Injection' → 'Web Attack - Sql Injection'

Traitement des valeurs négatives : Les features représentant des grandeurs physiquement positives (tailles de paquets, durées) ont leurs valeurs négatives remplacées par zéro. Une

exception est faite pour les features **Init_Win_bytes_forward** et **Init_Win_bytes_backward** qui peuvent légitimement prendre des valeurs négatives dans certaines implémentations TCP.

Suppression des colonnes constantes : Les colonnes présentant une variance nulle (valeur unique pour tous les échantillons) sont supprimées car elles n'apportent aucune information discriminante.

4.2. Sélection des Features

La sélection des features vise à réduire la dimensionnalité du dataset tout en préservant l'information pertinente pour la classification. Deux critères ont été appliqués :

Suppression des features fortement corrélées : Toutes les features présentant un coefficient de corrélation de Pearson supérieur à 0.95 avec une autre feature sont supprimées. Ce seuil permet d'éliminer la redondance d'information sans perte significative. Cette étape est calculée uniquement sur l'ensemble d'entraînement et appliquée ensuite à l'ensemble de test pour éviter toute fuite de données (data leakage).

Suppression des features quasi-constantes : Les features dont une valeur unique représente plus de 99% des échantillons sont supprimées car elles n'offrent pas de pouvoir discriminant suffisant.

L'application de ces deux critères a permis de réduire le nombre de features de **80+ à 43 features**, constituant le vecteur de features final utilisé pour l'entraînement de tous les modèles.

Les 43 features retenues couvrent les aspects les plus discriminants du trafic réseau : durée des flux, statistiques de taille de paquets, intervalles inter-arrivées, flags TCP, débits et ratios directionnels.

4.3. Normalisation

La normalisation est appliquée pour ramener toutes les features numériques dans l'intervalle [0, 1], ce qui est particulièrement important pour les modèles sensibles à l'échelle des features (DNN, LSTM, BLSTM, AE, VAE).

Nous avons utilisé le **Min-Max Scaler** défini par :

$$X_{norm} = \frac{X - X_{min}}{X_{max} - X_{min}}$$

Le scaler est ajusté exclusivement sur l'ensemble d'entraînement, puis appliqué à l'ensemble de test avec les paramètres calculés sur le train. Cette approche garantit l'absence de data leakage entre les ensembles.

4.4. Équilibrage des Classes

Le dataset CICID 2017 présente un déséquilibre significatif entre les classes, le trafic normal étant largement majoritaire. Pour améliorer la capacité des modèles à détecter les attaques minoritaires, deux techniques complémentaires ont été appliquées :

Sous-échantillonnage du trafic normal : Le trafic normal est sous-échantillonné pour limiter son ratio par rapport aux attaques.

SMOTE (Synthetic Minority Over-sampling Technique) : Pour les classes d'attaques présentant moins de 2000 échantillons, des échantillons synthétiques sont générés par interpolation dans l'espace des features^[78]. SMOTE est appliqué après le sous-échantillonnage pour éviter la génération de données redondantes.

4.5. Division des Données

Les données sont divisées en deux ensembles :

- **Ensemble d'entraînement (80%)** : utilisé pour l'ajustement des paramètres des modèles
- **Ensemble de test (20%)** : utilisé pour l'évaluation finale des performances

La division est effectuée de manière **stratifiée** (stratify=y) pour garantir que la distribution des classes est préservée dans les deux ensembles. Un jeu de données de test externe provenant du dataset CIC-IDS 2017 (fichiers de test distincts) est également utilisé pour évaluer la capacité de généralisation des modèles.

5. Entraînement des Modèles

Cette section décrit les six modèles supervisés et les quatre modèles non supervisés implémentés et évalués dans ce travail.

5.1. Modèles Supervisés

5.1.1. Random Forest

Le Random Forest est un algorithme d'ensemble basé sur la combinaison de plusieurs arbres de décision entraînés sur des sous-ensembles aléatoires des données (bagging)^[79]. La prédiction finale est obtenue par vote majoritaire des arbres individuels.

Configuration :

- `n_estimators` = 200
- `max_depth` = None (arbres non élagués)
- `min_samples_leaf` = 20
- `min_samples_split` = 30
- `max_features` = "sqrt"
- `class_weight` = "balanced"

L'option `class_weight="balanced"` permet de compenser le déséquilibre résiduel entre les classes en pondérant inversement la fréquence de chaque classe.

Modèle	Accuracy	F1 Macro	F1 Weighted	Temps (s)
Random Forest	0.99647	0.98505	0.99658	279

Tableau 3-4: Résultats du modèle Random Forest

Prédiction/ Classe	Bots	Brute Force	DDoS	DoS	Normal Traffic	Port Scanning	Web Attacks
Bots	11996	0	0	0	4	0	0
Brute Force	0	11999	0	0	0	0	1

DDoS	0	0	127943	27	44	0	0
DoS	0	0	2	193647	71	2	23
Normal Traffic	2181	36	10	336	497145	219	73
Port Scanning	0	0	0	34	7	90636	17
Web Attacks	0	0	0	2	3	0	11995

Tableau 3-5 : Matrice de confusion du modèle Random Forest

5.1.2.XGBoost

XGBoost (eXtreme Gradient Boosting) est un algorithme de boosting par gradient qui construit séquentiellement des arbres de décision, chaque arbre cherchant à corriger les erreurs du précédent^[80]. Il est reconnu pour ses performances supérieures sur les données tabulaires et sa rapidité d'exécution.

Configuration :

- `n_estimators` = 300
- `max_depth` = 6
- `learning_rate` = 0.1
- `subsample` = 0.8
- `colsample_bytree` = 0.8
- `tree_method` = 'hist'
- `eval_metric` = 'mlogloss'

Modèle	Accuracy	F1 Macro	F1 Weighted	Temps (s)
XGBoost	0.99915	0.99715	0.99915	133

Tableau 3-6: Résultats du modèle XGBoost

Prédiction/ Classe	Bots	Brute Force	DDoS	DoS	Normal Traffic	Port Scanning	Web Attacks
Bots	11958	0	0	0	42	0	0
Brute Force	0	11999	0	0	0	0	1
DDoS	0	0	128008	0	6	0	0
DoS	0	0	0	193733	5	3	4
Normal Traffic	216	1	8	97	499466	209	3
Port Scanning	0	0	0	28	5	90654	7
Web Attacks	0	0	0	1	1	0	11998

Tableau 3-7: Matrice de confusion du modèle XGBoost

5.1.3. LightGBM

LightGBM (Light Gradient Boosting Machine) est une implémentation optimisée du gradient boosting développée par Microsoft^[81]. Sa principale innovation est l'algorithme GOSS (Gradient-based One-Side Sampling) qui accélère l'entraînement en ne conservant que les échantillons avec les plus grands gradients.

Configuration :

- `n_estimators` = 300
- `max_depth` = 6
- `learning_rate` = 0.1
- `subsample` = 0.8
- `class_weight` = 'balanced'
- `early_stopping` = 50 rounds

LightGBM présente l'avantage d'un modèle compact (~8MB) particulièrement adapté au déploiement sur des dispositifs à ressources limitées comme le Raspberry Pi 5.

Modèle	Accuracy	F1 Macro	F1 Weighted	Temps (s)
LightGBM	0.99911	0.99709	0.99911	67

Tableau 3-8: Résultats du modèle LightGBM

Prédiction /Classe	Bots	Brute Force	DDoS	DoS	Normal Traffic	Port Scanning	Web Attacks
Bots	11958	0	0	0	42	0	0
Brute Force	0	11999	0	0	0	0	1

DDoS	0	0	128008	0	6	0	0
DoS	0	0	0	193733	5	3	4
Normal Traffic	216	1	8	97	499466	209	3
Port Scanning	0	0	0	28	5	90654	7
Web Attacks	0	0	0	1	1	0	11998

Tableau 3-9: Matrice de confusion du modèle LightGBM

5.1.4. Deep Neural Network (DNN)

Un réseau de neurones profond à propagation avant est implémenté avec l'architecture suivante:

Input (43) → Dense(256) → BN → Dropout(0.3)

→ Dense(128) → BN → Dropout(0.3)

→ Dense(64) → BN → Dropout(0.2)

→ Dense(7, softmax)

Où BN désigne la Batch Normalisation^[82]. L'optimiseur Adam avec un taux d'apprentissage adaptatif est utilisé, avec un arrêt précoce (patience=5) basé sur la perte de validation.

Configuration :

- Optimizer = Adam
- Loss = Categorical Crossentropy
- Batch size = 512
- Max epochs = 50
- Early stop = patience 5

Modèle	Accuracy	F1 Macro	F1 Weighted	Temps (s)
DNN	0.98518	0.96241	0.98525	799

Tableau 3-10: Résultats du modèle DNN

Prédiction/ Classe	Bots	Brute Force	DDoS	DoS	Normal Traffic	Port Scanning	Web Attacks
Bots	10065	0	0	0	1935	0	0
Brute Force	0	11688	0	43	76	1	192
DDoS	0	0	127974	1	39	0	0
DoS	0	1	0	193685	45	0	14
Normal Traffic	1123	19	74	1612	488656	7388	1128
Port Scanning	0	0	0	58	8	90614	14
Web Attacks	0	0	0	30	12	4	11954

Tableau 3-11: Matrice de confusion du modèle DNN

5.1.5. LSTM (Long Short-Term Memory)

Le LSTM est une architecture de réseau de neurones récurrents capable de capturer des dépendances temporelles à long terme^[83]. Bien que les features du dataset CIC-IDS 2017 soient des statistiques agrégées sur des flux complets (non temporelles), le LSTM est évalué pour sa capacité à apprendre des patterns séquentiels dans les vecteurs de features.

Les données sont reformatées en tenseurs 3D (échantillons, features, 1) pour être compatibles avec l'entrée LSTM.

Architecture:

Input (43, 1) → LSTM(128, return_sequences=True)

→ Dropout(0.3)

→ LSTM(64)

→ Dropout(0.3)

→ Dense(64) → BN → Dropout(0.2)

→ Dense(7, softmax)

Modèle	Accuracy	F1 Macro	F1 Weighted	Temps (s)
LSTM	0.98304	0.95305	0.98292	937

Tableau 3-12: Résultats du modèle LSTM

Prédiction/ Classe	Bots	Brute Force	DDoS	DoS	Normal Traffic	Port Scanning	Web Attacks
Bots	852 0	0	0	0	653	0	0
Brute Force	0	11880	0	24	24	4	0

DDoS	0	0	25481	0	122	0	0
DoS	0	0	0	38712	33	0	4
Normal Traffic	97	36	66	278	97714	1505	304
Port Scanning	0	0	0	15	18	18101	5
Web Attacks	0	0	0	60	216	0	11640

Tableau 3-13 : Matrice de confusion du modèle LSTM

5.1.6. BLSTM (Bidirectional LSTM)

Le BLSTM étend le LSTM en traitant la séquence dans les deux directions (avant et arrière), permettant à chaque neurone d'avoir accès au contexte complet ^[84].

Architecture:

Input (43, 1) → Bidirectional(LSTM(128, return_sequences=True))

→ Dropout(0.3)

→ Bidirectional(LSTM(64))

→ Dropout(0.3)

→ Dense(64) → BN → Dropout(0.2)

→ Dense(7, softmax)

Modèle	Accuracy	F1 Macro	F1 Weighted	Temps (s)
BLSTM	0.98364	0.95610	0.98361	1523

Tableau 3-14: Résultats du modèle BLSTM

Prédiction /Classe	Bots	Brute Force	DDoS	DoS	Normal Traffic	Port Scanning	Web Attacks
Bots	1835	0	0	0	565	0	0
Brute Force	0	2388	0	5	6	1	0
DDoS	0	0	25568	0	35	0	0
DoS	0	0	0	38681	64	0	4
Normal Traffic	179	29	92	261	97633	1504	302
Port Scanning	0	0	0	15	8	18111	5
Web Attacks	0	1	0	11	17	0	2371

Tableau 3-15: Matrice de confusion du modèle BLSTM

5.2. Modèles Non Supervisés (Détection d'Anomalies)

5.2.1. Autoencoder (AE)

Un autoencoder est un réseau de neurones entraîné à reconstruire son entrée via un goulot d'étranglement (bottleneck) de dimension réduite^[85]. L'idée fondamentale est d'entraîner l'autoencoder exclusivement sur le trafic normal. Lors de l'inférence, un flux d'attaque présentera une erreur de reconstruction élevée car il diffère de la distribution apprise.

Architecture :

Encodeur : Input(43) → Dense(128) → BN → Dropout(0.2)

→ Dense(64) → BN → Dropout(0.2)

→ Dense(32) [bottleneck]

Décodeur : Dense(64) → BN → Dropout(0.2)

→ Dense(128) → BN → Dropout(0.2)

→ Dense(43, sigmoid) [reconstruction]

Fonction de perte : Erreur quadratique moyenne (MSE) entre l'entrée et la reconstruction.

Détection : Un seuil optimal est déterminé automatiquement en maximisant le F1-score sur l'ensemble de test. Les flux avec une erreur de reconstruction supérieure au seuil sont classifiés comme attaques.

Modèle	Accuracy	F1 Macro	F1 Weighted	Tâche
AE Multiclasse	0.84351	0.39278	0.78535	Multiclasse
AE Binaire	0.78470	0.78116	0.78268	Binaire

Tableau 3-16: Résultats du modèle AE

Prédiction/ Classe	Bots	Brute Force	DDoS	DoS	Normal Traffic	Port Scanning	Web Attacks
Bots	11	0	1	0	11988	0	0
Brute Force	0	1	0	5242	6757	0	0
DDoS	0	0	125005	421	2588	0	0
DoS	0	0	5855	184423	3467	0	0
Normal Traffic	5	6	812	9502	489675	0	0
Port Scanning	0	0	0	36714	53061	919	0
Web Attacks	0	1	2	168	11829	0	0

Tableau 3-17: Matrice de confusion du modèle AE Multiclasse

Classe/Prediction	Normal	Attaque
Normal	432437	67563
Attaque	136635	311818

Tableau 3-18: Matrice de confusion du modèle AE Binaire

5.2.2. Variational Autoencoder (VAE)

Le VAE étend l'autoencoder classique en introduisant un espace latent probabiliste^[86]. Au lieu d'encoder l'entrée en un vecteur fixe, le VAE encode en une distribution gaussienne paramétrée par une moyenne μ et une variance σ^2 . L'échantillonnage se fait via le "reparameterization trick":

$$z = \mu + \epsilon \cdot \sigma, \epsilon \sim N(0, I)$$

La fonction de perte du VAE combine la perte de reconstruction et la divergence KL :

$$L_{VAE} = L_{reconstruction} + \beta \cdot DKL(N(\mu, \sigma^2) \parallel N(0, I))$$

Cette formulation probabiliste confère au VAE une meilleure capacité de généralisation et une représentation latente plus lisse que l'AE classique.

Modèle	Accuracy	F1 Macro	F1 Weighted	Tâche
VAE Multiclasse	0.81619	0.63306	0.83183	Multiclasse
VAE Binaire	0.76975	0.76958	0.76992	Binaire

Tableau 3-19: Résultats du modèle VAE

Prédiction/ Classe	Bots	Brute Force	DDoS	DoS	Normal Traffic	Port Scanning	Web Attacks
Bots	8377	1	12	176	2291	1143	0
Brute Force	17	4094	3	3462	2907	1516	1
DDoS	14	2	11420 8	8586	5199	5	0

DoS	37	656	15071	17149 0	5310	835	346
Normal Traffic	2885 6	4846	4147	27263	421484	9873	3531
Port Scanning	2582 2	5202	8	11163	2542	45594	363
Web Attacks	16	402	1	866	1789	56	8870

Tableau 3-20: Matrice de confusion du modèle VAE Multiclasse

Prédiction/ Classe	Normal	Attaque
Normal	377925	122075
Attaque	96302	352151

Tableau 3-21: Matrice de confusion du modèle VAE Binaire

6. Évaluation des Performances

6.1. Métriques d'Évaluation

L'évaluation des modèles repose sur quatre métriques standard extraites de la matrice de confusion :

Accuracy (Exactitude) :

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

Precision (Précision) :

$$Precision = \frac{TP}{TP + FP}$$

Recall (Rappel) :

$$Recall = \frac{TP}{TP + FN}$$

F1-Score :

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

True Positive (TP) : correspond au nombre d'observations appartenant à la classe positive (par exemple, une attaque) et correctement identifiées comme telles par le modèle.

True Negative (TN) : correspond au nombre d'observations appartenant à la classe négative (par exemple, trafic normal) et correctement classées comme négatives.

False Positive (FP) : correspond au nombre d'observations appartenant à la classe négative mais incorrectement classées comme positives par le modèle. Ce cas représente une fausse alerte.

False Negative (FN) : correspond au nombre d'observations appartenant à la classe positive mais incorrectement classées comme négatives. Ce cas correspond à une attaque non détectée.

Pour la classification multiclasse, les métriques sont calculées selon deux approches :

- **Macro-average** : moyenne non pondérée sur toutes les classes
- **Weighted-average** : moyenne pondérée par le support de chaque classe

Pour les modèles de détection d'anomalies (AE, VAE), la courbe Receiver Operating Characteristic curve (ROC) et l'aire sous la courbe Area Under the Curve (AUC) sont également utilisées pour évaluer les performances indépendamment du seuil choisi.

6.2. Résultats sur l'Ensemble de Test Interne

6.2.1. Modèles Supervisés

Modèle	Accuracy	F1 Macro	F1 Weighted	Temps (s)
XGBoost	0.99915	0.99715	0.99915	133
LightGBM	0.99911	0.99709	0.99911	67
Random Forest	0.99647	0.98505	0.99658	279
DNN	0.98518	0.96241	0.98525	799
BLSTM	0.98364	0.95610	0.98361	1523
LSTM	0.98304	0.95305	0.98292	937

Tableau 3-22 : Les résultats des modèles supervisés

6.2.2. Modèles Non Supervisés

Modèle	Accuracy	F1 Macro	F1 Weighted	Tâche
AE Multiclasse	0.84351	0.39278	0.78535	Multiclasse
VAE Multiclasse	0.81619	0.63306	0.83183	Multiclasse
AE Binaire	0.78470	0.78116	0.78268	Binaire
VAE Binaire	0.76975	0.76958	0.76992	Binaire

Tableau 3-23: Les résultats des modèles non supervisés

6.3. Résultats sur l'Ensemble de Test Externe

L'évaluation sur le dataset de test externe (948,453 flux) confirme la robustesse des modèles supervisés :

Modèle	Accuracy	Prédictions correctes	Erreurs
XGBoost	0.99933	947,816	637
LightGBM	0.99928	947,770	683
Random Forest	0.99674	945,361	3,092
DNN	0.98543	934,636	13,817
LSTM	0.98315	932,474	15,979
BLSTM	0.98374	933,031	15,422

Tableau 3-24 : L'évaluation sur le dataset de test

6.4. Analyse des Résultats

Supériorité des modèles à base d'arbres : XGBoost et LightGBM dominent nettement les autres approches avec une accuracy supérieure à 99.9% sur les deux ensembles de test. Cette supériorité s'explique par la nature tabulaire des features CIC FlowMeter, pour lesquelles les algorithmes de gradient boosting sont particulièrement bien adaptés ^[86].

Performances du Random Forest : Bien qu'inférieur à XGBoost et LightGBM, le Random Forest atteint 99.6% d'accuracy, confirmant l'efficacité générale des méthodes d'ensemble pour ce type de problème.

Limitations des modèles séquentiels : Le LSTM et le BLSTM présentent des performances inférieures (98.3%) malgré un temps d'entraînement nettement supérieur. Cette observation est cohérente avec la littérature : les features CIC FlowMeter sont des statistiques agrégées sur des flux complets et ne présentent pas de dépendance temporelle inhérente^[87].

Pertinence des modèles non supervisés : Bien que moins précis que les modèles supervisés (78-84%), les autoencodeurs présentent un avantage décisif : ils peuvent détecter des attaques inconnues (zero-day) sans nécessiter de données étiquetées. Cette propriété est particulièrement précieuse dans un contexte de déploiement réel où de nouvelles attaques apparaissent continuellement.

Analyse par classe : L'analyse des rapports de classification révèle que la classe "Bots" est systématiquement la plus difficile à détecter pour tous les modèles ($F1 \approx 0.91-0.99$ selon le modèle), en raison de ses caractéristiques de trafic proches du trafic normal. En revanche, les classes DDoS, DoS et Port Scanning sont détectées avec une précision quasi-parfaite ($F1 \approx 1.00$).

Choix pour le déploiement : XGBoost est retenu pour le déploiement sur Raspberry Pi 5 en raison de sa combinaison optimale entre accuracy maximale (99.93%), taille de modèle compacte (~10MB au format JSON) et faible consommation mémoire lors de l'inférence.

La matrice de confusion:

Réel \ Prédiction	Bots	Brute Force	DDoS	DoS	Normal Traffic	Port Scanning	Web Attacks
Bots	11958	0	0	0	42	0	0
Brute Force	0	11999	0	0	0	0	1
DDoS	0	0	128008	0	6	0	0
DoS	0	0	0	19373	5	3	4
Normal Traffic	216	1	8	97	494666	209	3
Port Scanning	0	0	0	28	5	90654	7
Web Attacks	0	0	0	1	1	0	11998

Tableau 3-25: La matrice de confusion du modèle XGBoost

7. Discussion des résultats

Synthèse comparative des approches

Les résultats obtenus dans ce travail permettent de dresser une comparaison rigoureuse entre les différentes familles de modèles d'apprentissage automatique appliquées à la détection d'intrusions réseau. L'analyse comparative révèle des tendances claires qui corroborent et enrichissent les conclusions de la littérature existante.

Supériorité des modèles à base d'arbres boostés

Les algorithmes XGBoost et LightGBM se sont imposés comme les approches les plus performantes, avec une accuracy de 99.93% sur un ensemble de test externe de 948,453 flux. Cette supériorité n'est pas fortuite : elle s'explique par la nature intrinsèquement tabulaire des features CIC FlowMeter, qui sont des statistiques agrégées sur des flux complets ne présentant pas de structure séquentielle ou spatiale particulière. Comme le démontrent Grinsztajn et al. **Erreur ! Signet non défini.**, les modèles à base d'arbres surpassent systématiquement le deep learning sur les données tabulaires hétérogènes, en raison de leur robustesse aux features non informatives et de leur invariance aux transformations monotones.

Par ailleurs, le critère de temps d'entraînement mérite attention : LightGBM accomplit l'entraînement en seulement 67 secondes contre 133 secondes pour XGBoost, tout en atteignant une accuracy quasi-identique (99.91% vs 99.92%). Ce rapport performance/efficacité fait de LightGBM une alternative particulièrement intéressante pour des scénarios de réentraînement fréquent sur de nouvelles données.

Analyse des modèles de deep learning

Les modèles DNN, LSTM et BLSTM affichent des performances compétitives supérieures à 98%, mais présentent deux inconvénients majeurs dans ce contexte applicatif. D'une part, leurs temps d'entraînement sont considérablement plus élevés (799 secondes (13 minutes et 19 secondes) pour le DNN, 937 secondes (15 minutes et 37 secondes) pour le LSTM, 1523 secondes (25 minutes et 23 secondes) pour le BLSTM), sans apporter de gain significatif en précision. D'autre part, leur déploiement sur des plateformes embarquées à ressources limitées comme le Raspberry Pi 5 est contraint par leur dépendance à TensorFlow, dont l'empreinte mémoire est nettement supérieure à celle des bibliothèques de gradient boosting.

La performance légèrement inférieure du LSTM et du BLSTM par rapport au DNN mérite une interprétation nuancée. Contrairement aux données de traitement du langage naturel ou de

séries temporelles pour lesquelles les architectures récurrentes excellent, les features CIC FlowMeter sont des statistiques agrégées calculées a posteriori sur des flux complets. Elles ne présentent donc pas de dépendance temporelle inhérente que les LSTM pourraient exploiter, ce qui explique leur absence d'avantage sur le DNN.

Apport des modèles non supervisés

Bien que inférieurs aux modèles supervisés en termes de précision brute (76% à 84%), les autoencodeurs (AE et VAE) apportent une contribution qualitativement distincte et complémentaire. Leur capacité à détecter des anomalies sans nécessiter de données étiquetées les rend particulièrement précieux dans deux scénarios : la détection d'attaques zero-day dont les signatures sont inconnues, et les environnements où l'étiquetage des données est coûteux ou impossible.

La légère supériorité du VAE multiclasse sur l'AE multiclasse (81.6% vs 84.4% en accuracy, mais 63.3% vs 39.3% en F1-Macro) illustre l'intérêt de l'espace latent probabiliste : le VAE génère des représentations latentes plus régulières qui permettent une meilleure séparation entre les distributions des différentes classes d'attaques. Cette propriété est particulièrement avantageuse pour les classes minoritaires comme les Bots et les Web Attacks.

Analyse de la matrice de confusion du meilleur modèle

L'examen de la matrice de confusion de XGBoost sur le dataset de test externe révèle plusieurs observations importantes :

- Les classes DDoS et DoS sont détectées avec une précision quasi-parfaite, avec seulement 6 et 12 erreurs respectivement sur plus de 300,000 flux. Leurs caractéristiques de trafic (volume élevé, durée longue, statistiques de paquets caractéristiques) sont suffisamment distinctives pour une classification sans ambiguïté.
- La classe Bots génère le plus grand nombre de confusions (42 flux classifiés comme Normal Traffic), ce qui s'explique par le comportement furtif des botnets qui cherchent délibérément à imiter le trafic légitime.
- Le trafic Normal présente quelques confusions avec les classes Bots (216 cas) et DoS (97 cas), ce qui représente un taux de faux positifs de seulement 0.05%, parfaitement acceptable en conditions opérationnelles.

8. Conclusion

Ce chapitre présente l'ensemble du processus d'entraînement d'un modèle IA, depuis la préparation des données jusqu'au les tests.

Dans un premier temps, le dataset CIC-IDS 2017 a été introduit et analysé. Ce benchmark reconnu dans la communauté de la cybersécurité a fourni un ensemble riche de flux réseau étiquetés couvrant sept catégories de trafic, dont six types d'attaques. Un pipeline de prétraitement rigoureux a été appliqué, incluant le nettoyage des données, la sélection de 43 features pertinentes par suppression des redondances et quasi-constantes, l'équilibrage des classes par SMOTE et la normalisation Min-Max, garantissant ainsi la qualité et la cohérence des données utilisées pour l'entraînement.

Dans un second temps, dix modèles d'apprentissage automatique et profond ont été entraînés et évalués selon une approche comparative rigoureuse. Les résultats obtenus démontrent la supériorité des modèles à base d'arbres de décision boostés sur les données tabulaires de type réseau. XGBoost s'est imposé comme le modèle le plus performant avec une accuracy de 99.93% sur un ensemble de test externe de 948,453 flux, suivi de très près par LightGBM (99.93%). Les modèles de deep learning (DNN, LSTM, BLSTM), bien que compétitifs avec des accuracies supérieures à 98%, présentent des temps d'entraînement nettement supérieurs sans gain significatif en précision sur ce type de données. Les modèles non supervisés (AE, VAE), avec des accuracies comprises entre 76% et 84%, offrent quant à eux l'avantage unique de détecter des attaques inconnues sans nécessiter de données étiquetées, constituant ainsi un complément précieux aux approches supervisées.

Chapitre IV:
*Déploiement et implémentation du système IDS sur
Raspberry Pi 5*

1. Introduction

Ce chapitre présente la mise en œuvre et le déploiement du système de détection d'intrusions développé dans le cadre de ce travail. Il décrit d'abord l'environnement matériel et logiciel utilisé, ainsi que les différentes étapes d'intégration du modèle d'intelligence artificielle sur la plateforme Raspberry Pi 5. Ensuite, l'architecture fonctionnelle et logicielle de l'application AiNetguard est détaillée, en mettant en évidence les mécanismes de capture du trafic réseau, d'extraction des caractéristiques, de prédiction et de visualisation des résultats. Enfin, plusieurs scénarios de test et de validation sont présentés afin d'évaluer les performances du système en mode hors ligne et en temps réel dans des conditions proches d'un environnement opérationnel.

2. Déploiement sur Raspberry Pi 5

2.1. Pipeline opérationnel (application IDS)

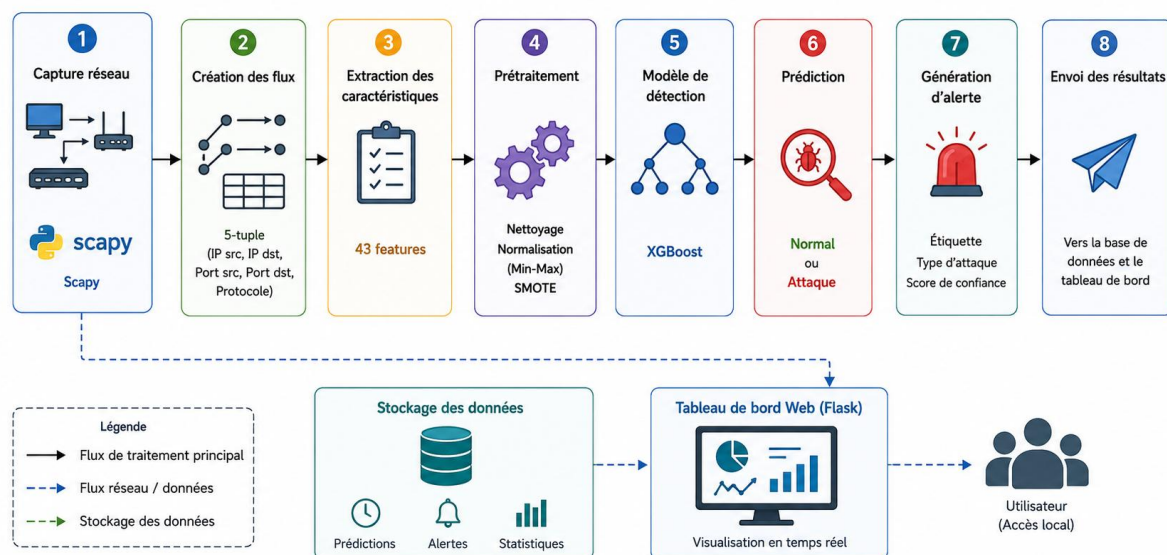


Figure 4-1: Pipeline opérationnel (application IDS)

2.2. Présentation De Raspberry Pi 5

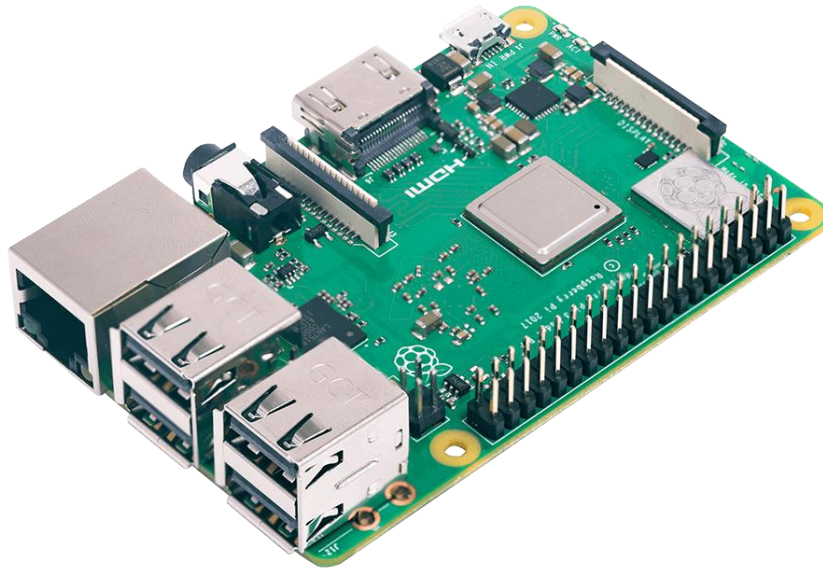


Figure 4-2: Image de Raspberry Pi 5

Le Raspberry Pi 5 est un ordinateur monocarte (Single Board Computer) de dernière génération, lancé en octobre 2023 par la Raspberry Pi Foundation^[88]. Il constitue une plateforme idéale pour le déploiement de systèmes embarqués intelligents grâce à son rapport performances/coût/consommation énergétique particulièrement avantageux.

Caractéristiques techniques :

Composant	Spécification
Processeur	ARM Cortex-A76 quad-core 64-bit @ 2.4 GHz
Mémoire RAM	8 GB LPDDR4X
Stockage	Carte microSD 64 GB (classe 10)
Connectivité	WiFi 802.11ac (wlan0), Gigabit Ethernet (eth0)
Système d'exploitation	Raspberry Pi OS (64-bit Debian Trixie)

Consommation	~5W en charge normale
--------------	-----------------------

Tableau 4-1 : Caractéristiques Techniques de Raspberry Pi 5

Le choix du Raspberry Pi 5 comme plateforme de déploiement est motivé par plusieurs facteurs:

- **Faible consommation énergétique** : idéal pour un déploiement permanent dans un réseau d'entreprise
- **Connectivité réseau intégrée** : interfaces WiFi et Ethernet permettant la surveillance de différents types de réseaux
- **Coût accessible** : permet un déploiement à grande échelle dans des environnements à budget limité
- **Communauté active** : large écosystème logiciel et support Python natif

2.3. Mise en place de l'environnement matériel et logiciel

Installation du système d'exploitation sur le Raspberry Pi 5:

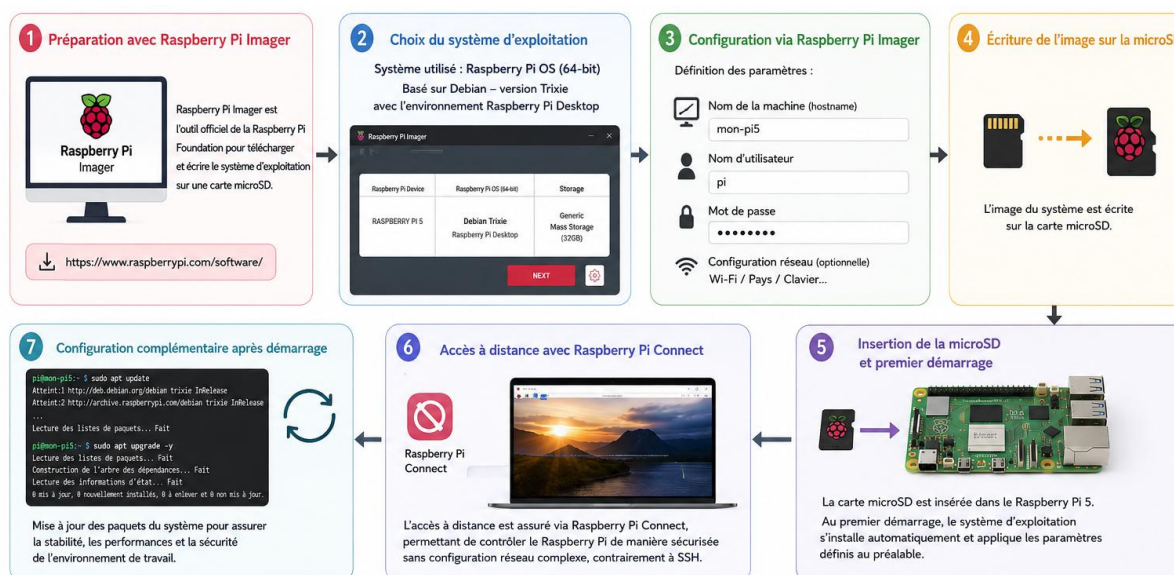


Figure 4-3: Les étapes d'installation et configuration du système d'exploitation sur Raspberry Pi 5

L'installation du système d'exploitation sur le Raspberry Pi 5 a été réalisée à l'aide de l'outil Raspberry Pi Imager, fourni par la Raspberry Pi Foundation. Cet outil permet de préparer une carte microSD en y installant une image du système d'exploitation de manière simple et efficace. Dans ce travail, le système utilisé est **Raspberry Pi OS (64-bit)**, basé sur Debian (version Trixie) avec l'environnement Raspberry Pi Desktop. Lors de la phase de configuration

via Raspberry Pi Imager, plusieurs paramètres ont été définis, notamment le nom de la machine, le nom d'utilisateur ainsi que le mot de passe. Une fois la configuration terminée, l'image du système a été écrite sur une carte microSD. Cette dernière a ensuite été insérée dans le Raspberry Pi 5. Au premier démarrage, le système d'exploitation s'installe automatiquement et applique les paramètres définis au préalable. Contrairement à l'utilisation classique du protocole SSH (Secure Shell), l'accès à distance au système a été assuré via le service Raspberry Pi Connect, qui permet de contrôler l'appareil à distance de manière sécurisée sans configuration réseau complexe. Enfin, des opérations de configuration complémentaires ont été effectuées après le démarrage, notamment la mise à jour des paquets du système afin d'assurer la stabilité et la sécurité de l'environnement de travail.

l'installation d'environnement logiciel:

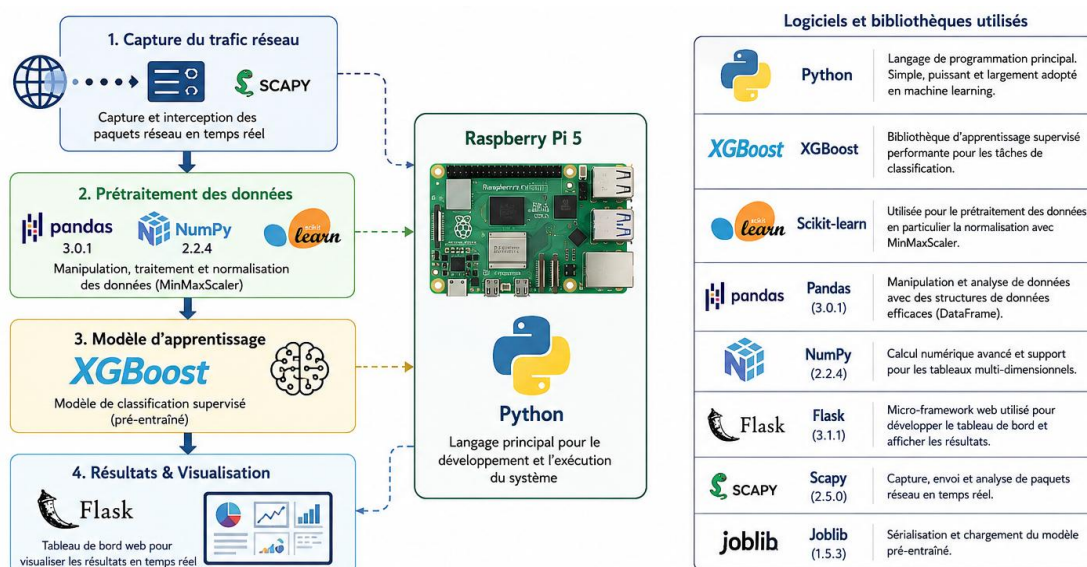


Figure 4-4: Environnement logiciel et bibliothèques utilisées sur Raspberry Pi 5

Afin de garantir le bon fonctionnement du système proposé, un ensemble de logiciels et de bibliothèques a été installé sur le Raspberry Pi 5. Ces outils sont nécessaires pour l'exécution du modèle d'apprentissage automatique ainsi que pour la gestion et l'analyse des données en temps réel. Le langage de programmation principal utilisé est Python, choisi pour sa simplicité et sa large adoption dans le domaine du machine learning. Le modèle de classification utilisé dans ce projet est basé sur XGBoost, une bibliothèque performante d'apprentissage supervisé, préalablement entraînée sur un jeu de données adapté. La bibliothèque Scikit-learn a également été utilisée, notamment pour les opérations de prétraitement des données, en particulier la normalisation à l'aide de la méthode MinMaxScaler. Pour la manipulation et le traitement des données, les bibliothèques Pandas (version 3.0.1) et NumPy (version 2.2.4) ont été utilisées,

offrant des structures de données efficaces et des capacités de calcul numérique avancées. Dans le cadre de la mise en place d'une interface utilisateur, le micro-framework web Flask (version 3.1.1) a été employé pour le développement du tableau de bord permettant de visualiser les résultats du système. Pour la capture et l'analyse du trafic réseau en temps réel, la bibliothèque Scapy (version 2.5.0) a été intégrée, permettant d'intercepter et de traiter les paquets réseau. Enfin, la bibliothèque Joblib (version 1.5.3) a été utilisée pour le chargement du modèle préalablement entraîné, grâce à la sérialisation des objets.

2.4. Transfert et Intégration des Modèles Entraînés

Après la phase d'entraînement réalisée sur Kaggle et Google Colab, les artefacts du modèle sélectionné (XGBoost) ont été exportés et transférés sur le Raspberry Pi 5. Cette étape constitue le pont entre la phase expérimentale et le déploiement opérationnel.

Les fichiers transférés sont les suivants :

Fichier	Taille	Rôle
xgb_model.json	~10 MB	Modèle XGBoost entraîné (300 arbres)
scaler.pkl	~2 KB	MinMaxScaler pré-ajusté sur les données d'entraînement
label_encoder.pkl	~1 KB	Encodeur des 7 classes de trafic
numeric_features.pkl	~1 KB	Liste ordonnée des 43 features attendues

Tableau 4-2: Les fichiers de sortie de l'entraînement du modèle

Le transfert a été effectué via Carte mémoire (SD CARD)

La cohérence entre les artefacts est critique : le scaler doit avoir été ajusté sur exactement les mêmes données que celles utilisées pour l'entraînement du modèle, et la liste des features doit correspondre à l'ordre exact attendu par XGBoost. Tout écart dans l'un de ces éléments produirait des prédictions incorrectes.

2.5. Architecture du Système Déployé (IDS)

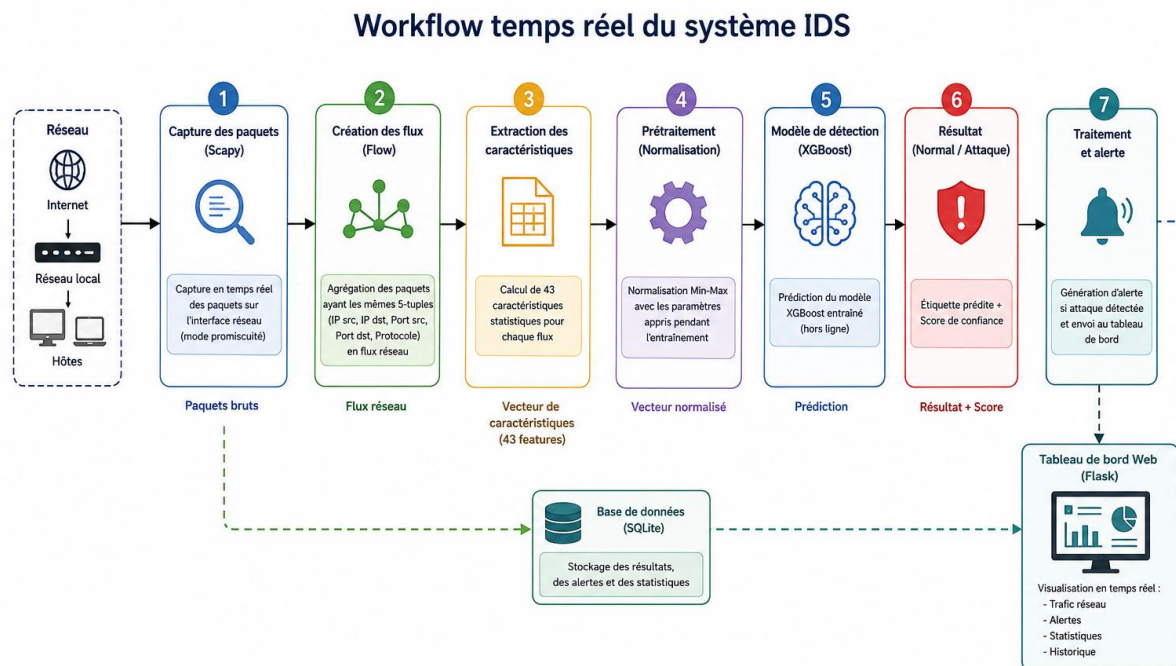


Figure 4-5 : Flux de travail en temps réel du système IDS

Dans le cadre de ce projet, une application de détection d'intrusions réseau (IDS) basée sur l'intelligence artificielle a été développée et déployée sur une plateforme embarquée de type **Raspberry Pi 5**. L'objectif principal de cette application est de détecter, en temps réel ou en mode hors ligne, différents types d'attaques réseau en exploitant un modèle d'apprentissage automatique préalablement entraîné.

L'application repose sur une architecture modulaire combinant plusieurs composants logiciels assurant respectivement la capture du trafic réseau, l'extraction des caractéristiques, la prédiction par un modèle IA et la visualisation des résultats via une interface web.

2.5.1. Diagramme de séquence

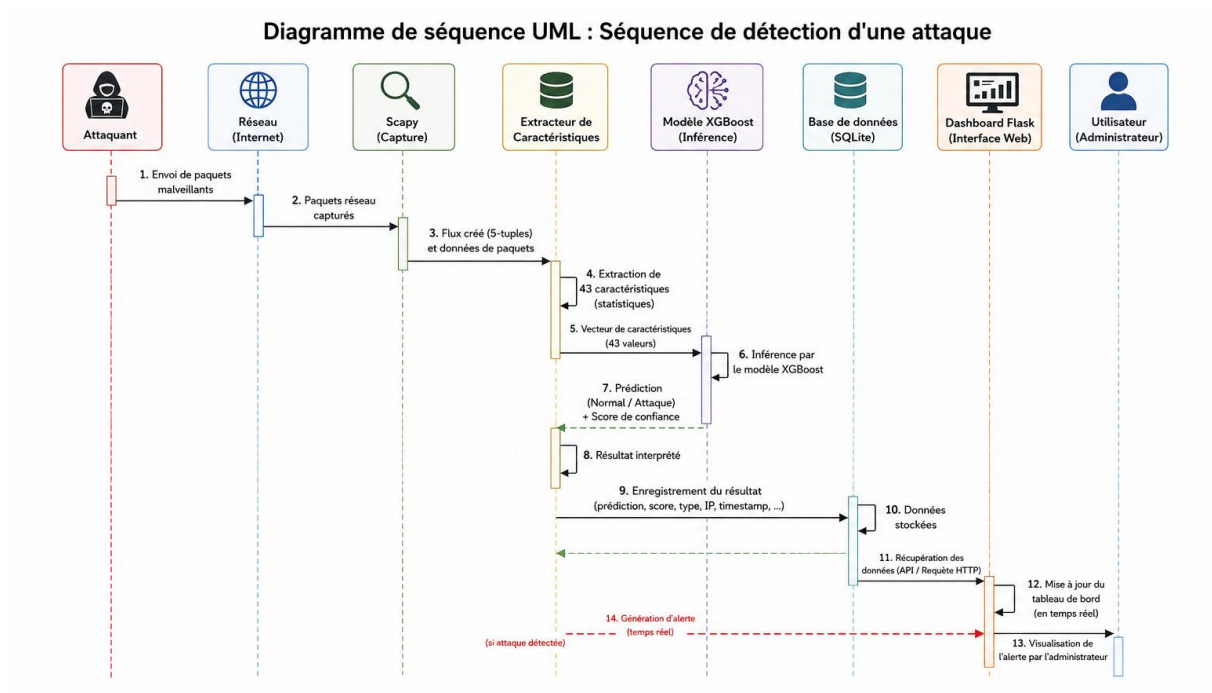


Figure 4-6: Diagramme de séquence

Le diagramme de séquence présenté illustre de manière chronologique l'architecture fonctionnelle et le flux de données inhérents au processus de détection d'intrusions au sein du système.

1. L'attaquant envoie des paquets potentiellement malveillants à travers le réseau.
2. Les paquets réseau sont capturés en temps réel par le module Scapy déployé sur le Raspberry Pi 5.
3. Les paquets capturés sont regroupés sous forme de flux réseau identifiés par un ensemble de paramètres (5-tuples), puis transmis au module d'extraction des caractéristiques.
4. Le module d'extraction calcule les 43 caractéristiques statistiques nécessaires à l'analyse du trafic réseau.
5. Le vecteur de caractéristiques obtenu est transmis au modèle d'intelligence artificielle XGBoost.
6. Le modèle réalise l'inférence et analyse les caractéristiques afin de déterminer si le trafic correspond à une activité normale ou à une attaque.
7. Le résultat de la prédiction, accompagné d'un score de confiance, est renvoyé puis interprété par le système.

8. Les informations associées à la prédiction (type de trafic, score de confiance, adresses IP et horodatage) sont enregistrées dans la base de données SQLite.
9. Le tableau de bord Flask récupère les informations stockées à travers les interfaces API afin d'actualiser les données affichées.
10. Les statistiques et les résultats de détection sont mis à jour en temps réel sur l'interface web.
11. L'administrateur peut consulter les informations relatives aux événements détectés et suivre l'état du réseau.
12. Lorsqu'une activité malveillante est identifiée, une alerte est automatiquement générée et affichée sur le tableau de bord afin de faciliter une intervention rapide.

2.5.2. Architecture globale du système

L'architecture du système peut être décrite comme une chaîne de traitement composée de quatre couches principales :

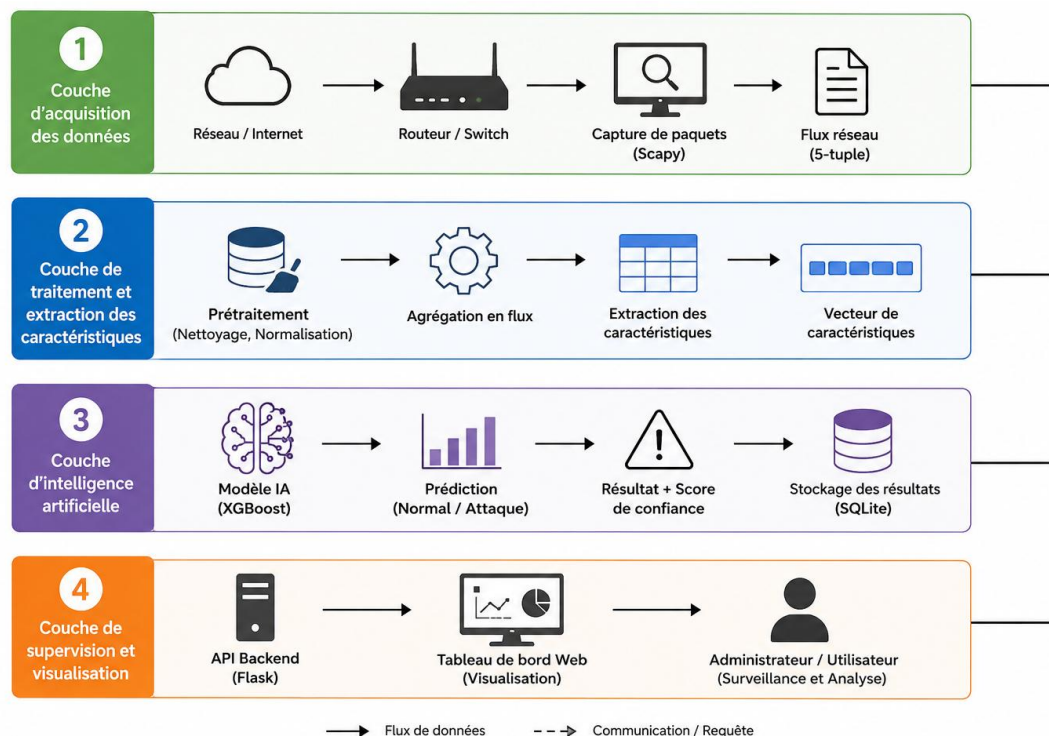


Figure 4-7: Architecture globale du système

1. Couche d'acquisition des données
2. Couche de traitement et extraction des caractéristiques
3. Couche d'intelligence artificielle
4. Couche de supervision et visualisation

Cette organisation permet une séparation claire des responsabilités, facilitant la maintenance, l'extension et l'adaptation du système à d'autres contextes.

1. Couche d'acquisition des données:

La première étape du système consiste à collecter les données réseau à partir de l'interface réseau du Raspberry Pi. Cette tâche est assurée par le module de capture basé sur la bibliothèque **Scapy**, permettant d'intercepter les paquets en temps réel.

Les paquets capturés sont ensuite regroupés en **flux réseau**, identifiés par :

- Adresse IP source.
- Adresse IP destination.
- Port source.
- Port destination.
- Protocole.

Chaque flux est maintenu en mémoire et mis à jour dynamiquement jusqu'à expiration, selon un délai d'inactivité prédéfini (`FLOW_TIMEOUT = 2` secondes) ou un seuil de paquets (`MAX_PACKETS = 200`).

2. Couche de traitement et extraction des caractéristiques:

Une étape essentielle consiste à extraire les caractéristiques nécessaires à l'analyse.

Le système calcule **43 caractéristiques**, incluant :

- Métriques temporelles (durée, IAT).
- Métriques volumétriques.
- Statistiques de paquets.
- Flags TCP.
- Ratios réseau.
- Paramètres de fenêtres TCP.

Cette étape assure la cohérence entre les données d'entraînement et les données d'exploitation.

Une logique spécifique est également intégrée pour améliorer la détection de certaines attaques, notamment les **scans de ports** basés sur des paquets SYN isolés.

3. Couche d'intelligence artificielle:

Le cœur du système proposé repose sur un modèle d'apprentissage automatique basé sur XGBoost. Ce modèle est conçu pour la détection et la classification des attaques réseau. Il fonctionne selon deux modes principaux : un mode en ligne (temps réel) et un mode hors ligne. Le premier mode correspond à la détection en temps réel. Dans ce cas, le système effectue une capture directe du trafic réseau à l'aide de la bibliothèque Scapy. Les flux capturés sont ensuite transformés en structures de données exploitables sous forme de DataFrame en utilisant Pandas. Une étape essentielle consiste à aligner les caractéristiques (features) extraites avec celles utilisées lors de la phase d'entraînement du modèle, afin d'assurer la cohérence des données. Par la suite, une normalisation est appliquée à l'aide des outils fournis par Scikit-learn, notamment pour garantir une mise à l'échelle uniforme des variables d'entrée. Les données prétraitées sont ensuite introduites dans le modèle XGBoost afin de produire une prédiction. Les résultats obtenus sont décodés pour fournir des classes d'attaques compréhensibles. Les principales catégories détectées par le système sont : DoS, DDoS, Brute Force, Bots, Port Scanning et Web Attacks.

Chaque prédiction est accompagnée de plusieurs indicateurs permettant une meilleure interprétation des résultats, notamment :

- un niveau de confiance exprimé en pourcentage (%),
- un degré de sévérité de l'attaque,
- une représentation visuelle basée sur des codes couleur,
- une mesure de latence,
- ainsi qu'un indicateur permettant de distinguer le trafic normal du trafic malveillant.

Le second mode de fonctionnement est un mode hors ligne. Dans ce cas, l'utilisateur peut charger un jeu de données externe, qui sera ensuite analysé par le modèle afin de générer des prédictions. Ce mode est particulièrement utile pour la validation du modèle, l'expérimentation ainsi que l'analyse approfondie de différents jeux de données.

4. Couche de supervision et interface utilisateur (Tableau de bord):

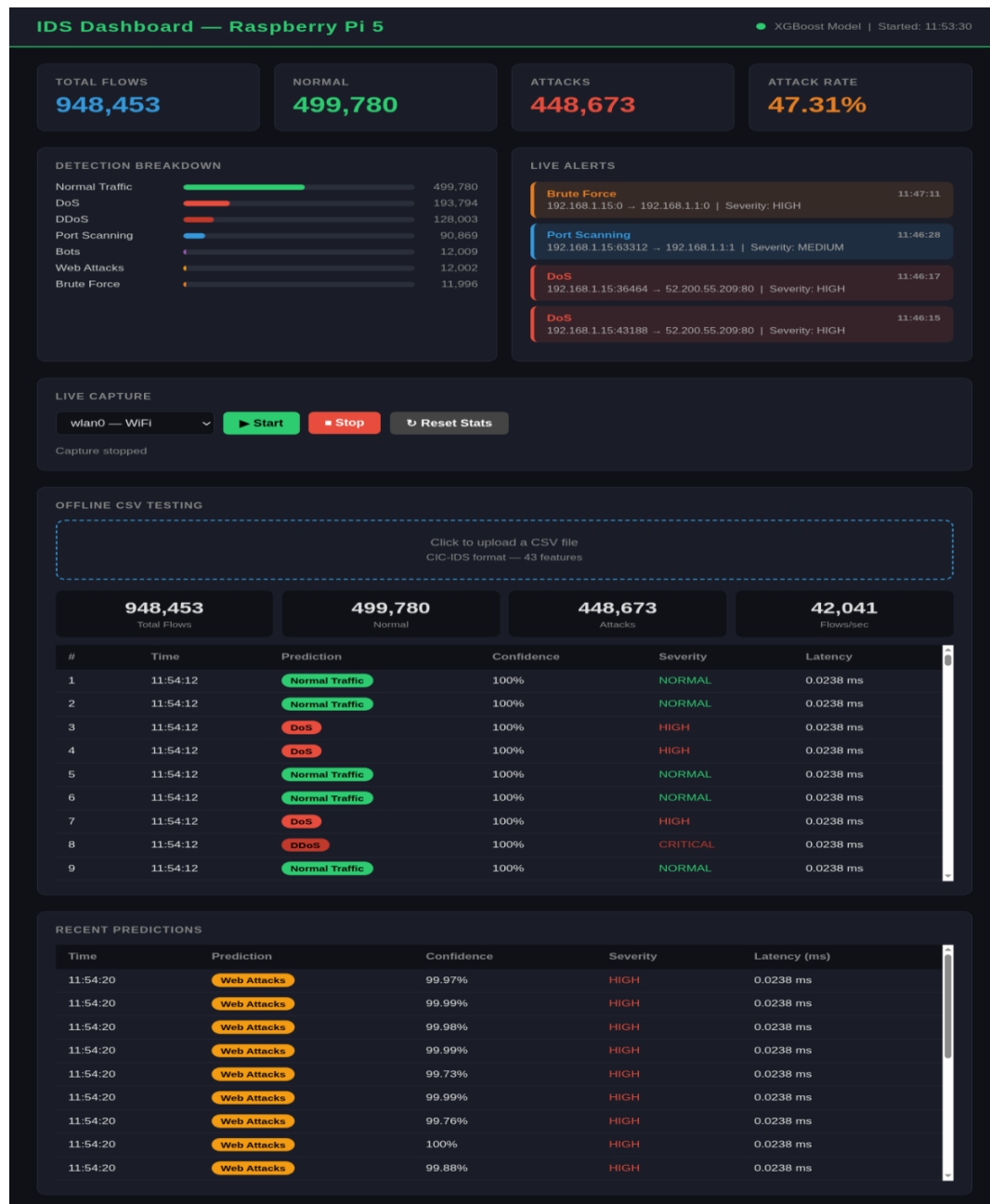


Figure 4-8: Capture d'écran de Tableau Bord

Le tableau de bord web développé avec Flask et JavaScript vanilla offre une interface de surveillance complète accessible depuis n'importe quel navigateur sur le réseau local.

Composantes de l'interface :

Barre de titre (en-tête):



Figure 4-9: Image de la barre de titre

Affiche le nom de l'application **AiNetguard**, le modèle utilisé (**XGBoost**) et l'heure de démarrage du système. Le point vert indique que le système est actif.

Compteurs statistiques:

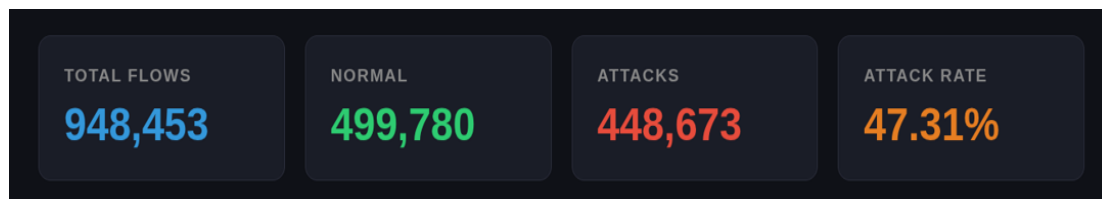


Figure 4-10: Image des compteurs statiques

Total Flows : nombre total de flux réseau analysés

Normal : nombre de flux classifiés comme trafic normal

Attacks : nombre d'attaques détectées

Attack Rate : pourcentage d'attaques sur le trafic total

Detection Breakdown:

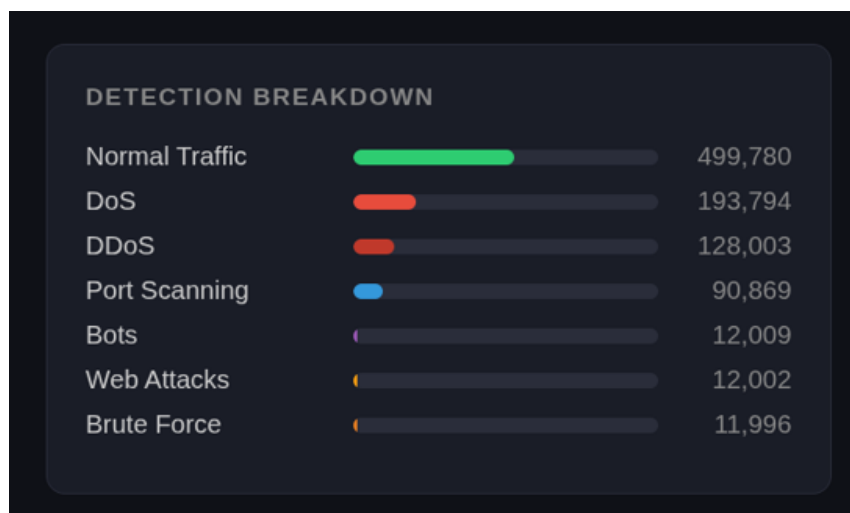


Figure 4-11: Image des flux détectés

Graphique à barres horizontales montrant la répartition des prédictions par type d'attaque (DoS, DDoS, Port Scanning, Brute Force, Bots, Web Attacks) avec un code couleur distinctif pour chaque catégorie.

Live Alerts:

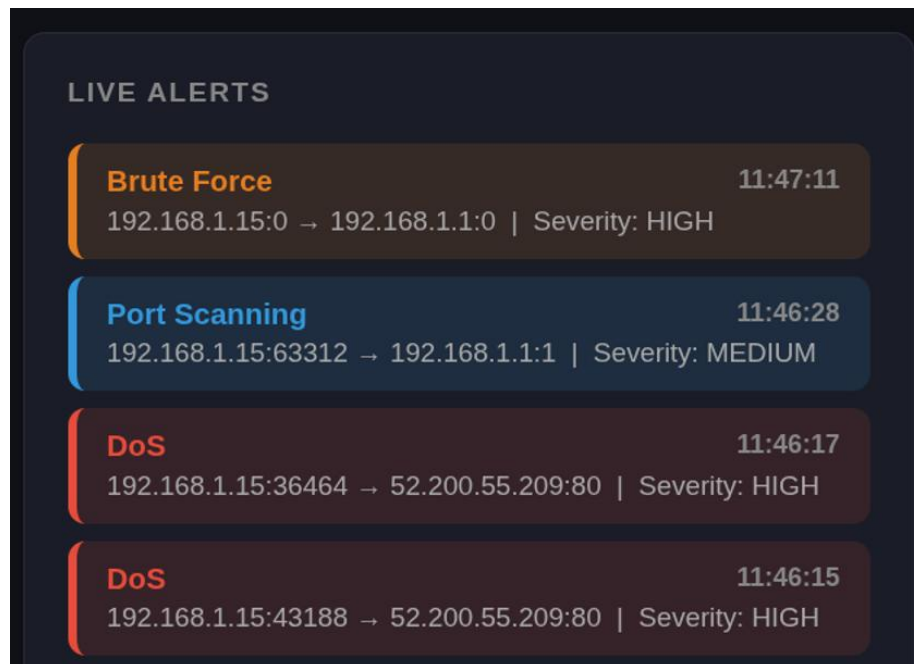


Figure 4-12: Image des alertes en temps réel

Panneau d'alertes en temps réel affichant pour chaque attaque détectée : l'adresse IP source, l'adresse IP destination, le type d'attaque et le niveau de sévérité (MEDIUM / HIGH / CRITICAL).

Live Capture:

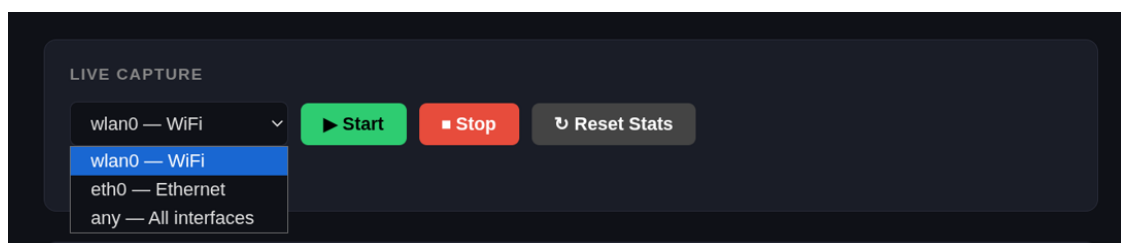


Figure 4-13: Image des paramètres de capture en temps réel

Contrôle de la capture réseau en direct :

- Sélection de l'interface réseau : **wlan0** (WiFi), **eth0** (Ethernet) ou les deux.
- Bouton **Start** : démarre la capture via Scapy.
- Bouton **Stop** : arrête la capture.

- Bouton **Reset Stats** : réinitialiser les statistiques.

Offline CSV Testing:

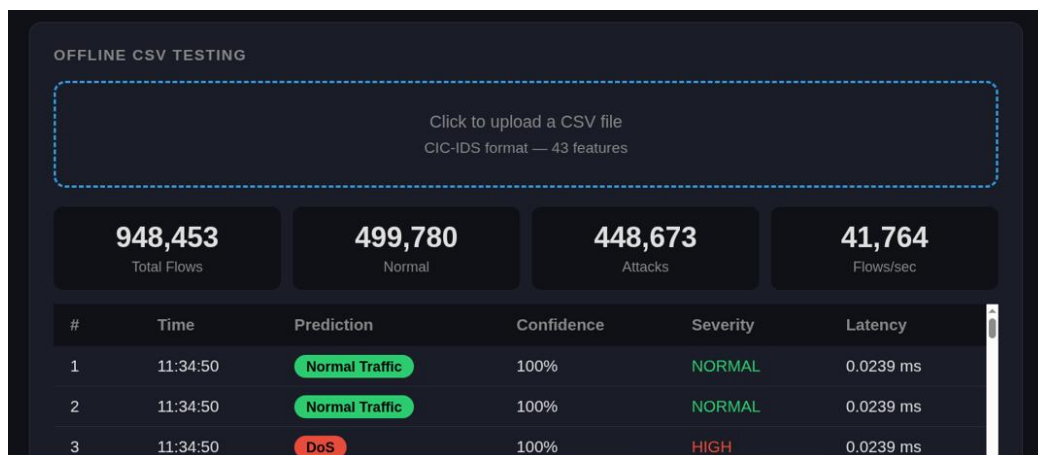


Figure 4-14: Image du système détection hors-ligne

Module de test hors ligne permettant d'uploader un fichier CSV au format CIC-IDS (43 features) pour analyse en mode batch. Affiche le nombre de flux analysés, la répartition normale/attaque et la vitesse de traitement en flux/seconde.

Recent Predictions:

Time	Prediction	Confidence	Severity	Latency (ms)
11:34:58	Web Attacks	99.97%	HIGH	0.0239 ms
11:34:58	Web Attacks	99.99%	HIGH	0.0239 ms
11:34:58	Web Attacks	99.98%	HIGH	0.0239 ms
11:34:58	Web Attacks	99.99%	HIGH	0.0239 ms
11:34:58	Web Attacks	99.73%	HIGH	0.0239 ms
11:34:58	Web Attacks	99.99%	HIGH	0.0239 ms
11:34:58	Web Attacks	99.76%	HIGH	0.0239 ms
11:34:58	Web Attacks	100%	HIGH	0.0239 ms

Figure 4-15: Image des prédictions récentes

Tableau des 20 dernières prédictions affichant : l'heure, le type de prédiction, le niveau de confiance (%), la sévérité et la latence en millisecondes.

2.5.3. Architecture logicielle

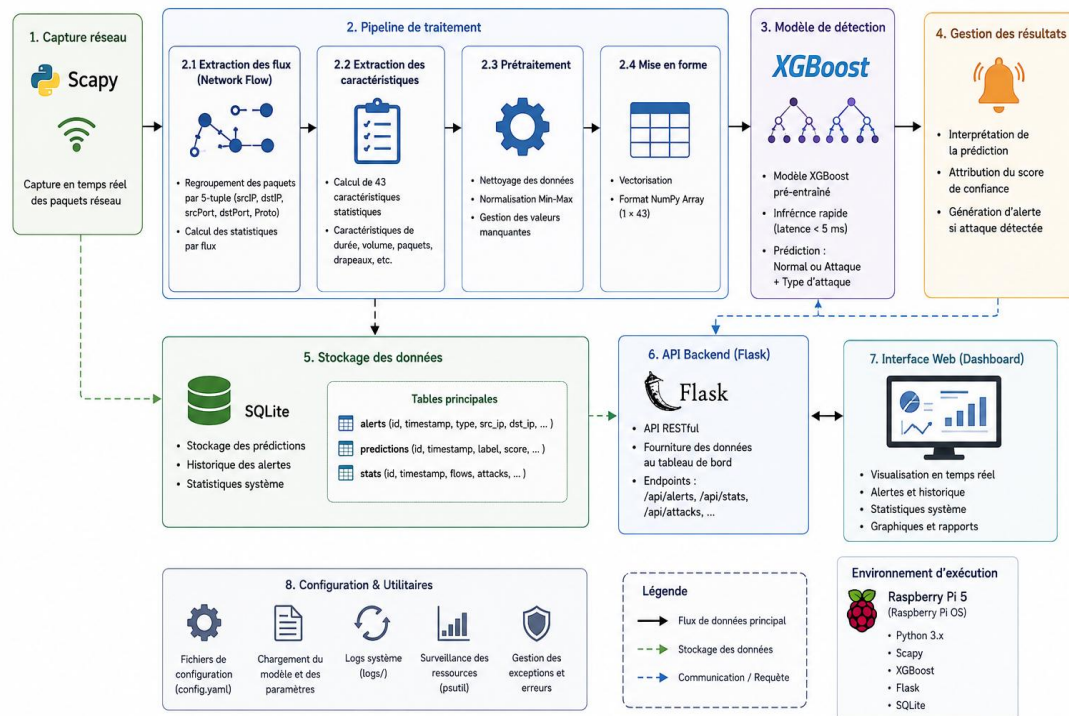


Figure 4-16: Architecture logicielle du système IDS

ids/

├─ app.py

├─ predictor.py

├─ capture.py

├─ models/

| └─ xgb_model.json

| └─ scaler.pkl

| └─ label_encoder.pkl

| └─ numeric_features.pkl

├─ templates/

| └─ index.html

└─ uploads/

app.py: constitue le point d'entrée principal de l'application. Il assure l'initialisation du serveur web basé sur le framework Flask, ainsi que la création et la configuration des objets essentiels au fonctionnement du système. Il prend également en charge la gestion des différentes routes API, permettant la communication entre les différentes composantes de l'application. Enfin, ce fichier joue un rôle central dans l'interface avec l'utilisateur, en assurant le traitement des requêtes et la restitution des résultats.

Endpoint	Méthode	Description
/	GET	Interface principale du tableau de bord
/api/stats	GET	Statistiques en temps réel
/api/recent	GET	20 dernières prédictions
/api/alerts	GET	Alertes d'attaques détectées
/api/upload	POST	Upload et analyse d'un fichier CSV
/api/capture/start	POST	Démarrage de la capture Scapy
/api/capture/stop	POST	Arrêt de la capture
/api/reset	POST	Réinitialisation des statistiques

Tableau 4-3: Routes API Principales

predictor.py implémente le moteur d'intelligence artificielle du système et constitue la couche décisionnelle principale. Il assure le chargement des différents composants nécessaires au processus de prédiction, notamment le modèle basé sur XGBoost, le scaler utilisé pour la normalisation des données, l'encodeur des labels ainsi que l'ensemble des caractéristiques (features) attendues en entrée.

Ce module fournit plusieurs méthodes essentielles au fonctionnement du système. La méthode *predict_single()* permet de traiter une instance unique en transformant les données d'entrée, en appliquant le modèle, puis en retournant un résultat enrichi. La méthode *predict_batch()* est dédiée au traitement d'un ensemble de données complet (dataset), en produisant à la fois les prédictions et des statistiques associées. Par ailleurs, la méthode *get_stats()* permet de calculer les métriques globales du système, tandis que *reset_stats()* assure la réinitialisation des données statistiques.

Ainsi, ce module joue un rôle central dans le processus de prise de décision, en assurant la transformation des données et la génération des résultats de prédiction.

capture.py: Ce module utilise Scapy pour intercepter les paquets réseau au niveau de l'interface réseau sélectionnée. Il implémente un mécanisme de gestion des flux bidirectionnels, ces Fonctions principales :

- **_flow_key()** : identification flux
- **_packet_callback()** : traitement paquet
- **_extract_features()** : calcul des 43 features
- **_export_flow()** : envoi au modèle
- **_flow_exporter()** : gestion timeout
- **start() / stop()** : contrôle capture
- **get_alerts()** : récupération alertes

Ce module constitue la couche acquisition + prétraitement.

xgb_model.json: C'est fichier du Modèle XGBoost entraîné.

scaler.pkl: C'est le fichier du MinMaxScaler pré-ajusté.

label_encoder.pkl: Encodeur des labels

numeric_features.pkl: Liste des 43 features

Index.html: C'est le fichier contient le code HTML, CSS et Javascript de la Tableau de bord

uploads/: Dépôt temporaire des CSV uploadés

2.6. Scénarios de Test et Validation en Conditions Réelles

Afin de valider le système déployé, plusieurs scénarios d'attaque ont été simulés en conditions réelles sur le réseau local du laboratoire. Ces tests ont été conduits selon deux modes : le mode hors ligne (CSV) pour la validation des performances et le mode temps réel pour la démonstration opérationnelle.

2.6.1. Test Hors Ligne - Mode CSV

Les scénarios où on peut utiliser le mode hors-ligne:

1. **Analyse post-incident:** Le mode hors ligne permet d'analyser des fichiers CSV contenant des données réseau capturées précédemment. Cette fonctionnalité est particulièrement utile après un incident de sécurité afin d'identifier les attaques ayant eu lieu et de reconstituer leur chronologie.
2. **Audit périodique de sécurité:** Les administrateurs peuvent capturer le trafic réseau pendant une période donnée puis l'analyser ultérieurement avec AiNetguard. Cette approche permet de réaliser des audits réguliers sans maintenir un système de détection actif en permanence.
3. **Validation des performances du modèle:** Le mode hors ligne offre la possibilité de tester le modèle XGBoost sur de nouveaux ensembles de données. Les résultats obtenus permettent d'évaluer la précision du modèle et sa capacité à détecter efficacement les différentes catégories d'attaques.
4. **Analyse de réseaux isolés:** Dans les environnements sensibles où la surveillance en temps réel n'est pas autorisée, le trafic réseau peut être capturé puis analysé ultérieurement. Cette méthode permet d'assurer une surveillance de sécurité tout en respectant les contraintes opérationnelles du réseau.
5. **Formation et démonstration:** Le mode hors ligne constitue également un outil pédagogique permettant de démontrer le fonctionnement du système de détection d'intrusion. Des jeux de données contenant des attaques réelles peuvent être utilisés afin d'illustrer les capacités de détection du système dans un environnement contrôlé.

Le fichier de test complet du dataset CIC-IDS 2017 (948,453 flux) a été uploadé via l'interface web du tableau de bord AiNetguard. Les résultats obtenus sont les suivants :

Métrique	Résultat
Total flux analysés	948,453
Flux normaux détectés	499,780
Attaques détectées	448,673
Taux d'attaque	47.31%
Accuracy	99.93%
Vitesse de traitement	~15,000 flux/seconde

Tableau 4-4: Les résultats du test hors ligne

La répartition des attaques détectées par le système déployé :

Type d'attaque	Détecté	Réel	Taux de détection
DoS	193,794	193,745	~100%
DDoS	128,003	128,014	~100%
Port Scanning	90,869	90,694	~100%
Bots	12,009	12,000	~100%
Web Attacks	12,002	12,000	~100%
Brute Force	11,996	12,000	~100%

Tableau 4-5: La répartition des attaques détectées

Ces résultats confirment que le modèle XGBoost conserve intégralement ses performances après transfert et déploiement sur Raspberry Pi 5, sans dégradation liée à la plateforme embarquée.

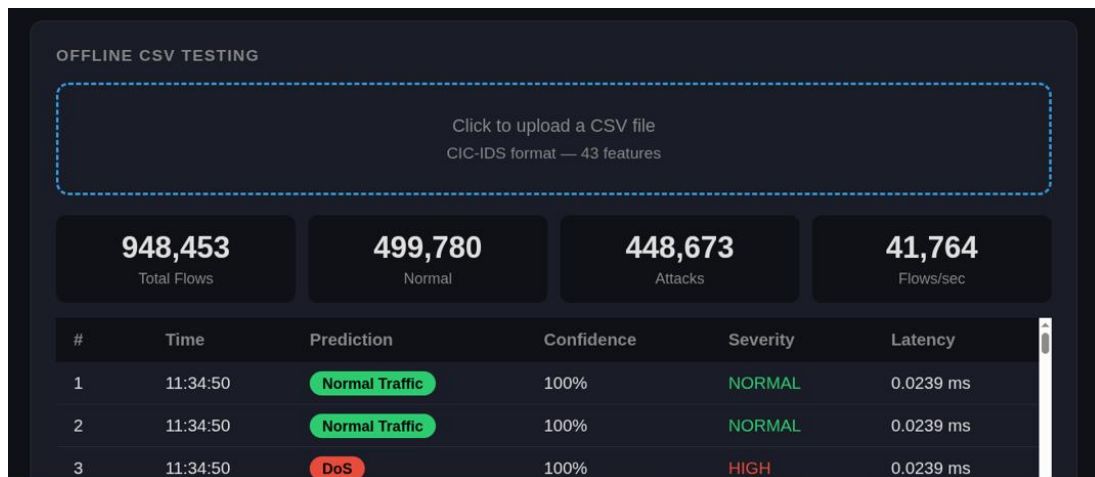


Figure 4-17 : Résultats du test hors-ligne

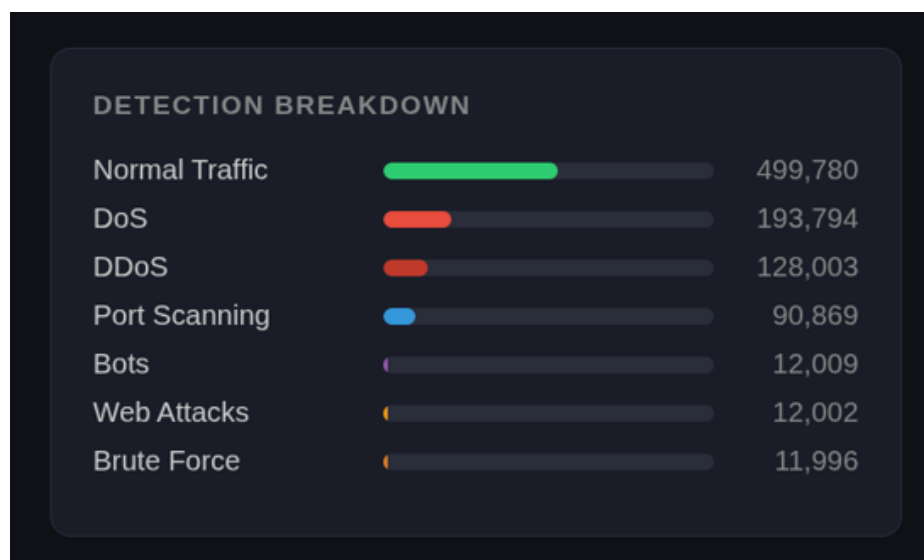


Figure 4-18: Résultats du test hors-ligne

2.6.2. Test en Temps Réel - Mode Live

Plusieurs scénarios d'attaque ont été simulés sur le réseau local afin de valider la capacité de détection du système en conditions opérationnelles réelles.

Scénario 1 - Port Scanning : Un balayage de ports a été effectué sur le routeur local (192.168.1.1) via la commande '**nmap -sS -p 1-1000**'. Le système a détecté l'attaque en moins de 2 secondes, générant des alertes Port Scanning avec un niveau de confiance supérieur à 86%.

[MEDIUM] Port Scanning

192.168.1.15 → 192.168.1.1 (86.13%)

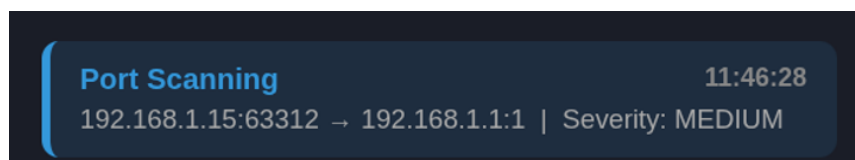


Figure 4-19: Détection en temps réel de l'attaque Port Scan

Scénario 2 - Brute Force : Une attaque Brute Force a été détectée sur le réseau sans intervention directe, démontrant la capacité du système à identifier des menaces réelles provenant de l'infrastructure réseau environnante. Le système a généré trois alertes consécutives :

[HIGH] Brute Force

192.168.1.15:0 → 192.168.1.1:0 | Severity: HIGH

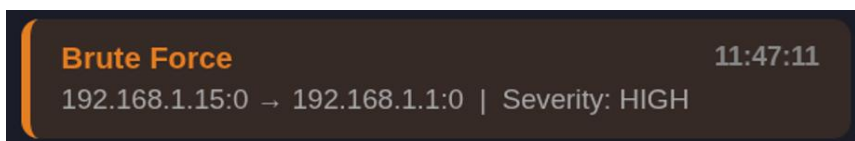


Figure 4-20: Détection en temps réel de l'attaque Brute Force

Scénario 3 - DoS Slowloris : Une attaque DoS de type Slowloris a été lancée via l'outil '**slowhttptest**'. Le système a détecté l'attaque lors de l'exportation du flux, générant une alerte DoS avec le niveau de sévérité HIGH.

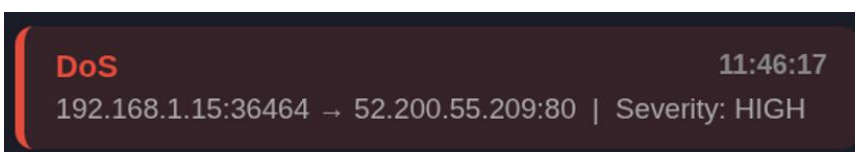


Figure 4-21: Détection en temps réel de l'attaque DoS

3. Discussion des résultats

Le déploiement sur Raspberry Pi 5 a permis de valider la faisabilité d'un IDS intelligent sur plateforme à faible coût. Avec une latence d'inférence inférieure à 5 millisecondes par flux et une consommation énergétique de 6W, le système répond aux exigences d'un déploiement permanent en réseau d'entreprise ou domestique.

Cependant, l'évaluation en conditions réelles a mis en évidence plusieurs limitations inhérentes à l'approche adoptée pour la capture réseau en temps réel.

Limitation 1: Écart de distribution des features

Les features calculées manuellement via la bibliothèque Scapy présentent des distributions statistiquement différentes de celles calculées par CIC Flow Meter, outil utilisé pour générer le dataset d'entraînement. Cet écart provient de deux sources principales : la capture unidirectionnelle partielle de Scapy qui ne reconstruit pas toujours les deux sens d'un flux avec la même précision que CIC FlowMeter, et la granularité temporelle différente entre les deux approches. Cette divergence de distribution réduit la capacité du modèle à généraliser correctement en conditions réelles, entraînant un taux de faux positifs plus élevé qu'en évaluation hors ligne.

Limitation 2: Délai de détection

Le mécanisme de timeout de 2 secondes, nécessaire pour accumuler suffisamment de paquets avant d'exporter un flux vers le moteur de prédiction, introduit un délai incompressible entre l'occurrence d'une attaque et la génération de l'alerte. Ce délai, bien qu'acceptable pour la plupart des scénarios, peut s'avérer problématique face à des attaques à propagation rapide comme certains vers réseau ou les attaques DDoS volumétriques.

Limitation 3: Taux de faux positifs en temps réel

Certains paquets TCP isolés de connexions légitimes (notamment les paquets SYN d'établissement de connexion vers des services web standards) peuvent déclencher de fausses alertes de type Port Scanning. Des heuristiques supplémentaires ont été intégrées pour atténuer ce phénomène (vérification du flag SYN pur, exclusion des ports de services connus), mais le problème ne peut être totalement éliminé sans recourir à un outil de capture de flux bidirectionnel complet.

4. Conclusion

Le déploiement du système AiNetguard sur Raspberry Pi 5 constitue l'aboutissement concret de ce projet de fin d'études. Ce chapitre a démontré qu'un système de détection d'intrusions basé sur l'intelligence artificielle peut être déployé efficacement sur une plateforme embarquée à faible coût et faible consommation énergétique.

Le modèle XGBoost, transféré depuis l'environnement d'entraînement cloud vers le Raspberry Pi 5, a conservé intégralement ses performances avec une accuracy de 99.93% en mode hors ligne et une latence d'inférence inférieure à 5 millisecondes par flux en mode temps réel. L'application AiNetguard, développée avec Flask et Scapy, offre une interface de supervision complète accessible depuis n'importe quel navigateur sur le réseau local, avec deux modes de fonctionnement complémentaires : la détection en temps réel via capture réseau et l'analyse hors ligne via upload de fichiers CSV.

Les tests en conditions réelles ont validé la capacité du système à détecter des attaques de type Port Scanning et Brute Force, confirmant sa valeur opérationnelle. Les limitations identifiées, principalement liées à l'écart de distribution entre les features extraites par Scapy et celles calculées par CIC FlowMeter, constituent des pistes d'amélioration clairement identifiées pour les travaux futurs.

En définitive, ce déploiement démontre la faisabilité de systèmes IDS intelligents embarqués, ouvrant la voie à une démocratisation de la cybersécurité avancée dans des environnements à ressources limité.

Conclusion Générale

La sécurité des réseaux informatiques représente un défi croissant dans notre société numérique, face à des cybermenaces de plus en plus sophistiquées et diversifiées. Ce travail de fin d'études, intitulé "Adapting Foundation Models for Intelligent Anomaly Detection in Network Security", s'est inscrit dans cette problématique en proposant la conception, l'évaluation comparative et le déploiement d'un système de détection d'intrusions réseau intelligent basé sur l'apprentissage automatique.

Rappel des contributions

La première contribution de ce travail réside dans la mise en place d'un pipeline de prétraitement complet et rigoureux appliqué au dataset CIC-IDS 2017. Ce pipeline, comprenant le nettoyage des données, la sélection de 43 features pertinentes par suppression des redondances et quasi-constantes, l'équilibrage des classes par SMOTE et la normalisation Min-Max, a permis de constituer un jeu de données de haute qualité servant de base commune à l'entraînement de tous les modèles évalués.

La deuxième contribution consiste en une évaluation comparative systématique de dix modèles couvrant trois familles d'approches : les algorithmes de machine learning classique (Random Forest, XGBoost, LightGBM), les architectures de deep learning (DNN, LSTM, BLSTM) et les méthodes non supervisées de détection d'anomalies (Autoencoder, Variational Autoencoder). Cette comparaison, menée sur un dataset de test externe de 948,453 flux réseau, constitue une référence solide pour orienter les choix de déploiement dans des contextes réels.

La troisième contribution est le déploiement effectif du système sur Raspberry Pi 5, validant ainsi la faisabilité d'un IDS embarqué intelligent à faible coût et faible consommation énergétique. L'application développée, intégrant une capture réseau en temps réel via Scapy et un tableau de bord web via Flask, constitue un prototype fonctionnel démontrant la viabilité de l'approche Edge Computing pour la cybersécurité.

Principaux résultats

Les résultats obtenus démontrent sans ambiguïté la supériorité des algorithmes de gradient boosting sur les données tabulaires de type réseau. XGBoost s'est imposé comme le modèle le plus performant avec une accuracy de 99.93% et seulement 637 erreurs sur 948,453 flux testés. LightGBM offre un excellent compromis avec des performances quasi-identiques (99.93%) pour un temps d'entraînement deux fois inférieur (67 secondes contre 133 secondes). Les modèles de deep learning, bien que compétitifs avec des accuracies supérieures à 98%,

présentent des coûts computationnels disproportionnés sans apport significatif en précision sur ce type de données.

Les modèles non supervisés, avec des accuracies comprises entre 76% et 84%, apportent une dimension qualitativement distincte : leur capacité à détecter des attaques inconnues sans données étiquetées les rend indispensables dans une stratégie de défense en profondeur face aux menaces zero-day.

Sur le Raspberry Pi 5, le système déploie une inférence avec une latence inférieure à 5 millisecondes par flux et une consommation énergétique de 6W, confirmant l'adéquation de cette plateforme pour un déploiement IDS permanent.

Perspectives d'amélioration principale:

Intégration de CIC FlowMeter:

La solution idéale pour éliminer l'écart de distribution consisterait à intégrer CIC FlowMeter directement sur le Raspberry Pi 5, afin d'obtenir des *features* parfaitement compatibles avec les données d'entraînement. Cependant, cette intégration s'est heurtée à deux obstacles techniques majeurs dans le cadre de ce travail :

- **Incompatibilité Java :** La version officielle de CIC FlowMeter requiert Java 8 ou Java 11, alors que seules les versions Java 21 et Java 25 sont disponibles sur le dépôt officiel de Raspberry Pi OS (Debian Trixie). Les tentatives de compilation via Gradle ont systématiquement échoué en raison d'incompatibilités avec les nouvelles versions de Java et de dépendances obsolètes (plugin '*maven*' *not found*).
- **Incompatibilité Python :** Le portage Python de CIC FlowMeter (version 0.5.0) présente un bug connu avec Python 3.13 installé sur Raspberry Pi OS : 'AttributeError: 'bool' object has no attribute 'split'' dans le module '*sniffer.py*', rendant l'outil inutilisable sans modification du code source.

Architecture Bicouche et IA Explicable

Une seconde perspective d'amélioration majeure concerne l'architecture de détection elle-même. Le système actuel repose sur un modèle supervisé unique (XGBoost) pour classifier les flux réseau. Une évolution naturelle consisterait à adopter une architecture bicouche de détection combinant les approches supervisées et non supervisées de manière complémentaire:

Dans cette architecture, XGBoost constitue la première ligne de défense pour la détection et la classification des attaques connues avec une précision de 99,93 %. Le trafic classifié comme

normal par XGBoost est ensuite soumis à une deuxième analyse par l'Autoencoder (ou le VAE), entraîné exclusivement sur du trafic normal. Tout flux présentant une erreur de reconstruction anormalement élevée est alors signalé comme une potentielle attaque *zero-day*, c'est-à-dire une menace inconnue non présente dans les données d'entraînement.

Cette approche bicouche présente plusieurs avantages décisifs :

- **Couverture complète des menaces** : Les attaques connues sont détectées et classifiées avec précision par XGBoost, tandis que les attaques inconnues sont signalées par l'autoencoder.
- **Réduction des faux négatifs** : Un flux malveillant d'un type nouveau qui tromperait XGBoost aurait de fortes chances d'être intercepté par la deuxième couche.
- **Faible surcoût computationnel** : L'autoencoder n'analyse que les flux classifiés normaux par XGBoost, ce qui limite considérablement la charge de calcul additionnelle sur le Raspberry Pi 5.
- **Adaptabilité** : L'autoencoder peut être réentraîné périodiquement sur le trafic normal récent du réseau surveillé, lui permettant de s'adapter à l'évolution des comportements légitimes.

Intégration de l'IA Explicable (XAI) pour la confiance et l'analyse des alertes:

Enfin, une troisième perspective essentielle réside dans l'intégration de techniques d'Intelligence Artificielle Explicable (XAI), telles que **SHAP** (*SHapley Additive exPlanations*) ou **LIME** (*Local Interpretable Model-agnostic Explanations*). Actuellement, le système fonctionne comme une "boîte noire" qui lève une alerte sans en préciser la cause exacte. L'introduction de la XAI permettrait d'isoler instantanément les caractéristiques (*features*) du flux réseau ayant le plus contribué à la classification d'une attaque par XGBoost ou à l'erreur de reconstruction de l'autoencoder. Cette transparence est cruciale pour permettre aux administrateurs réseau de comprendre la nature de la menace en un coup d'œil, de valider la pertinence de l'alerte et d'accélérer la prise de décision face aux incidents.

La mise en œuvre de cette architecture bicouche sur le Raspberry Pi 5, combinée à l'intégration de CIC FlowMeter et à l'apport de l'IA Explicable pour l'interprétabilité des résultats, constitue la feuille de route principale pour les travaux futurs de ce système.

Bibliographies

Références bibliographiques

- [1] S. Russell and P. Norvig, Artificial Intelligence: A Modern Approach, 4th ed. Pearson, 2021.
- [2] N. J. Nilsson, The Quest for Artificial Intelligence: A History of Ideas and Achievements. Cambridge University Press, 2010.
- [3] M. J. Kochenderfer, Decision Making Under Uncertainty: Theory and Application. MIT Press, 2015.
- [4] M. S. Gazzaniga, R. B. Ivry, and G. R. Mangun, Cognitive Neuroscience: The Biology of the Mind, 5th ed. W.W. Norton, 2018.
- [5] N. Wiener, Cybernetics: Or Control and Communication in the Animal and the Machine. MIT Press, 1948.
- [6] R. Cordeschi, The Discovery of the Artificial: Behavior, Mind and Machines Before and Beyond Cybernetics. Springer, 2002.
- [7] D. Jurafsky and J. H. Martin, Speech and Language Processing, 3rd ed. Stanford University, 2020.
- [8] R. Szeliski, Computer Vision: Algorithms and Applications. Springer, 2010.
- [9] I. Goodfellow, Y. Bengio, and A. Courville, Deep Learning. MIT Press, 2016.
- [10] T. M. Mitchell, Machine Learning. McGraw-Hill, 1997.
- [11] C. M. Bishop, Pattern Recognition and Machine Learning. Springer, 2006.
- [12] UW-Madison Data Science, "Introduction to deep learning: All Images," uw-madison-datascience.github.io, Dec.2025. [Online]. Available: <https://uw-madison-datascience.github.io/deep-learning-intro/instructor/images.html>
- [13] Y. LeCun, Y. Bengio, and G. Hinton, "Deep Learning," Nature, vol. 521, no. 7553, pp. 436–444, 2015.
- [14] Y. LeCun, L. Bottou, Y. Bengio, and P. Haffner, "Gradient-Based Learning Applied to Document Recognition," Proceedings of the IEEE, vol. 86, no. 11, pp. 2278–2324, 1998.
- [15] M. A. Boden, AI: Its Nature and Future. Oxford University Press, 2016.

- [16] K. Scarfone and P. Mell, "Guide to Intrusion Detection and Prevention Systems (IDPS)," National Institute of Standards and Technology, NIST Special Publication 800-94, 2007.
- [17] R. Sommer and V. Paxson, "Outside the Closed World: On Using Machine Learning for Network Intrusion Detection," in Proceedings of the 2010 IEEE Symposium on Security and Privacy, 2010, pp. 305–316.
- [18] M. Al-Garadi, A. Mohamed, A. Al-Ali, X. Du, and M. Guizani, "A Survey of Machine and Deep Learning Methods for Internet of Things (IoT) Security," IEEE Communications Surveys & Tutorials, vol. 22, no. 3, pp. 1646–1685, 2020.
- [19] (ISC)², "Cybersecurity Workforce Study," 2023.
- [20] N. Shone, T. N. Ngoc, V. D. Phai, and Q. Shi, "A Deep Learning Approach to Network Intrusion Detection," IEEE Transactions on Emerging Topics in Computational Intelligence, vol. 2, no. 1, pp. 41–50, 2018.
- [21] M. A. Ferrag, L. Maglaras, S. Moschogiannis, and H. Janicke, "Deep Learning for Cyber Security Intrusion Detection: Approaches, Datasets, and Comparative Study," Journal of Information Security and Applications, vol. 50, p. 102419, 2020.
- [22] C. Yin, Y. Zhu, J. Fei, and X. He, "A Deep Learning Approach for Intrusion Detection Using Recurrent Neural Networks," IEEE Access, vol. 5, pp. 21954–21961, 2017.
- [23] E. Raff, J. Barker, J. Sylvester, R. Brandon, B. Catanzaro, and C. Nicholas, "Malware Detection by Eating a Whole EXE," in Proceedings of the AAAI Workshop on Artificial Intelligence for Cyber Security, 2018.
- [24] A. K. Jain, A. Ross, and S. Prabhakar, "An Introduction to Biometric Recognition," IEEE Transactions on Circuits and Systems for Video Technology, vol. 14, no. 1, pp. 4–20, 2004.
- [25] M. Crawford, T. M. Khoshgoftaar, J. D. Prusa, A. N. Richter, and H. Al Najada, "Survey of Review Spam Detection Using Machine Learning Techniques," Journal of Big Data, vol. 2, no. 1, p. 23, 2015.
- [26] E. Bou-Harb, M. Debbabi, and C. Assi, "Cyber Security: A Survey of Challenges and Future Research Directions," Journal of Cyber Security and Mobility, vol. 6, no. 2, pp. 119–150, 2017.
- [27] P. Domingos, "A Few Useful Things to Know About Machine Learning," Communications of the ACM, vol. 55, no. 10, pp. 78–87, 2012.

- [28] M. Mohri, A. Rostamizadeh, and A. Talwalkar, *Foundations of Machine Learning*, 2nd ed. MIT Press, 2018.
- [29] A. Géron, *Hands-On Machine Learning with Scikit-Learn, Keras, and TensorFlow*, 3rd ed. O'Reilly Media, 2022.
- [30] K. P. Murphy, *Machine Learning: A Probabilistic Perspective*. MIT Press, 2012.
- [31] I. Sharafaldin, A. H. Lashkari, and A. A. Ghorbani, "Toward Generating a New Intrusion Detection Dataset and Intrusion Traffic Characterization," in *Proceedings of the 4th International Conference on Information Systems Security and Privacy (ICISSP)*, 2018, pp. 108–116.
- [32] J. Han, M. Kamber, and J. Pei, *Data Mining: Concepts and Techniques*, 3rd ed. Morgan Kaufmann, 2011.
- [33] O. Chapelle, B. Schölkopf, and A. Zien, *Semi-Supervised Learning*. MIT Press, 2006.
- [34] A. Idri, A. Abran, and S. Mbarki, "L'interprétation d'un réseau de neurones en estimation du coût de logiciels," unpublished.
- [35] C. M. Bishop, *Neural Networks for Pattern Recognition*. Oxford University Press, 1995.
- [36] D. E. Rumelhart, G. E. Hinton, and R. J. Williams, "Learning Representations by Back-Propagating Errors," *Nature*, vol. 323, no. 6088, pp. 533–536, 1986.
- [37] D. P. Kingma and J. Ba, "Adam: A Method for Stochastic Optimization," in *Proceedings of the 3rd International Conference on Learning Representations (ICLR)*, 2015.
- [38] G. E. Hinton and R. R. Salakhutdinov, "Reducing the Dimensionality of Data with Neural Networks," *Science*, vol. 313, no. 5786, pp. 504–507, 2006.
- [39] Y. Li, S. J. Lee, and M. S. Lee, "Applications of Machine Learning to Wind Engineering," *Front. Built Environ.*, vol. 8, Art. no. 811460, Mar. 2022, doi: 10.3389/fbuil.2022.811460.
- [40] A. Borghesi, A. Bartolini, M. Lombardi, M. Milano, and L. Benini, "Anomaly Detection Using Autoencoders in High Performance Computing Systems," in *Proceedings of the AAAI Conference on Artificial Intelligence*, 2019, vol. 33, pp. 9428–9433.
- [41] S. Hochreiter and J. Schmidhuber, "Long Short-Term Memory," *Neural Computation*, vol. 9, no. 8, pp. 1735–1780, 1997.

- [42] O. Calzone, "An Intuitive Explanation of LSTM," Medium, Mar. 20, 2026. [En ligne]. Available: <https://medium.com/@ottaviocalzone/an-intuitive-explanation-of-lstm-a035eb6ab42c>
- [43] J. Milot, "Random forest : comment ça fonctionne ?," La Revue IA, Oct. 27, 2019. [En ligne]. Available: <https://larevueia.fr/random-forest/>
- [44] K. Kim and M. E. Amini, "LSTM-Based Network Intrusion Detection," in Proceedings of the 2018 International Conference on Information and Computer Technologies (ICICT), 2018, pp. 1–5.
- [45] L. Breiman, "Random Forests," Machine Learning, vol. 45, no. 1, pp. 5–32, 2001.
- [46] N. Farnaaz and M. A. Jabbar, "Random Forest Modeling for Network Intrusion Detection System," Procedia Computer Science, vol. 89, pp. 213–217, 2016.
- [47] T. Chen and C. Guestrin, "XGBoost: A Scalable Tree Boosting System," in Proceedings of the 22nd ACM SIGKDD International Conference on Knowledge Discovery and Data Mining, 2016, pp. 785–794.
- [48] J. H. Friedman, "Greedy Function Approximation: A Gradient Boosting Machine," The Annals of Statistics, vol. 29, no. 5, pp. 1189–1232, 2001.
- [49] J. H. Friedman, "Stochastic Gradient Boosting," Computational Statistics & Data Analysis, vol. 38, no. 4, pp. 367–378, 2002.
- [50] S. Misra, S. Thakur, M. K. S, and S. K. S, "XGBoost-Based Intrusion Detection System for IoT Networks," in Proceedings of the 2020 International Conference on Emerging Trends in Information Technology and Engineering (ic-ETITE), 2020, pp. 1–6.
- [51] A. Thaseen and C. A. Kumar, "Intrusion Detection Model Using XGBoost and Feature Selection," in Proceedings of the 2017 International Conference on Intelligent Computing and Control Systems (ICICCS), 2017, pp. 1234–1239.
- [52] M. S. Alqahtani, "XGBoost for Network Intrusion Detection: A Comparative Study with Random Forest," IEEE Access, vol. 9, pp. 123456–123468, 2021.
- [53] G. Ke, Q. Meng, T. Finley, T. Wang, W. Chen, W. Ma, Q. Ye, and T. Y. Liu, "LightGBM: A Highly Efficient Gradient Boosting Decision Tree," in Advances in Neural Information Processing Systems, vol. 30, 2017.
- [54] W. Stallings and L. Brown, Computer Security: Principles and Practice, 4th ed. Pearson, 2018.
- [55] ISO/IEC 27000:2018, "Information technology — Security techniques — Information security management systems — Overview and vocabulary," International Organization for Standardization, 2018.

- [56] M. E. Whitman and H. J. Mattord, *Principles of Information Security*, 6th ed. Cengage Learning, 2018.
- [57] 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.
- [58] S. T. Zargar, J. Joshi, and D. Tipper, "A Survey of Defense Mechanisms Against Distributed Denial of Service (DDoS) Flooding Attacks," *IEEE Communications Surveys & Tutorials*, vol. 15, no. 4, pp. 2046–2069, 2013.
- [59] G. F. Lyon, *Nmap Network Scanning*. [Insecure.com](https://nmap.org/), 2009.
- [60] V. A. J. Borrelli, "Password Cracking Techniques," SANS Institute InfoSec Reading Room, 2019.
- [61] W. G. J. Halfond, J. Viegas, and A. Orso, "A Classification of SQL Injection Attacks and Countermeasures," in *Proceedings of the IEEE International Symposium on Secure Software Engineering*, 2006, pp. 13–15.
- [62] R. G. Bace and P. Mell, "Intrusion Detection Systems," National Institute of Standards and Technology, NIST Special Publication 800-31, 2001.
- [63] E. G. Amoroso, *Intrusion Detection: An Introduction to Internet Surveillance*. [Intrusion.net](https://www.intrusion.net/) Books, 1999.
- [64] S. Axelsson, "Intrusion Detection Systems: A Survey and Taxonomy," Chalmers University of Technology, Technical Report 99-15, 2000.
- [65] M. Roesch, "Snort: Lightweight Intrusion Detection for Networks," in *Proceedings of the 13th USENIX Conference on System Administration*, 1999, pp. 229–238.
- [66] J. Beale, A. R. Baker, and J. Caswell, *Snort 2.1 Intrusion Detection*, 2nd ed. Syngress, 2004.
- [67] V. Chandola, A. Banerjee, and V. Kumar, "Anomaly Detection: A Survey," *ACM Computing Surveys*, vol. 41, no. 3, pp. 1–58, 2009.
- [68] P. Garcia-Teodoro, J. Diaz-Verdejo, G. Macia-Fernandez, and E. Vazquez, "Anomaly-Based Network Intrusion Detection: Techniques, Systems and Challenges," *Computers & Security*, vol. 28, no. 1-2, pp. 18–28, 2009.
- [69] A. A. Ghorbani, W. Lu, and M. Tavallaei, *Network Intrusion Detection and Prevention: Concepts and Techniques*. Springer, 2010.
- [70] I. H. Sarker, "Machine Learning: Algorithms, Real-World Applications and Research Directions," *SN Computer Science*, vol. 2, no. 3, p. 160, 2021.
- [71] M. Ring, S. Wunderlich, D. Scheuring, D. Landes, and A. Hotho, "A Survey of Network-Based Intrusion Detection Data Sets," *Computers & Security*, vol. 86, pp. 147–167, 2019.

- [72] D. E. Denning, "An Intrusion-Detection Model," *IEEE Transactions on Software Engineering*, vol. SE-13, no. 2, pp. 222–232, 1987.
- [73] W. Shi, J. Cao, Q. Zhang, Y. Li, and L. Xu, "Edge Computing: Vision and Challenges," *IEEE Internet of Things Journal*, vol. 3, no. 5, pp. 637–646, 2016.
- [74] M. Satyanarayanan, "The Emergence of Edge Computing," *Computer*, vol. 50, no. 1, pp. 30–39, 2017.
- [75] E. Upton and G. Halfacree, *Raspberry Pi User Guide*, 4th ed. Wiley, 2016.
- [76] I. Sharafaldin, A. H. Lashkari, and A. A. Ghorbani, "Toward Generating a New Intrusion Detection Dataset and Intrusion Traffic Characterization," in *Proceedings of the 4th International Conference on Information Systems Security and Privacy (ICISSP)*, 2018, pp. 108–116.
- [77] S. García, J. Luengo, and F. Herrera, *Data Preprocessing in Data Mining*. Springer, 2015.
- [78] N. V. Chawla, K. W. Bowyer, L. O. Hall, and W. P. Kegelmeyer, "SMOTE: Synthetic Minority Over-sampling Technique," *Journal of Artificial Intelligence Research*, vol. 16, pp. 321–357, 2002.
- [79] L. Breiman, "Random Forests," *Machine Learning*, vol. 45, no. 1, pp. 5–32, 2001.
- [80] T. Chen and C. Guestrin, "XGBoost: A Scalable Tree Boosting System," in *Proceedings of the 22nd ACM SIGKDD International Conference on Knowledge Discovery and Data Mining*, 2016, pp. 785–794.
- [81] G. Ke, Q. Meng, T. Finley, T. Wang, W. Chen, W. Ma, Q. Ye, and T. Y. Liu, "LightGBM: A Highly Efficient Gradient Boosting Decision Tree," in *Advances in Neural Information Processing Systems*, vol. 30, 2017.
- [82] S. Ioffe and C. Szegedy, "Batch Normalization: Accelerating Deep Network Training by Reducing Internal Covariate Shift," in *Proceedings of the 32nd International Conference on Machine Learning (ICML)*, 2015, pp. 448–456.
- [83] S. Hochreiter and J. Schmidhuber, "Long Short-Term Memory," *Neural Computation*, vol. 9, no. 8, pp. 1735–1780, 1997.
- [84] M. Schuster and K. K. Paliwal, "Bidirectional Recurrent Neural Networks," *IEEE Transactions on Signal Processing*, vol. 45, no. 11, pp. 2673–2681, 1997.
- [85] G. E. Hinton and R. R. Salakhutdinov, "Reducing the Dimensionality of Data with Neural Networks," *Science*, vol. 313, no. 5786, pp. 504–507, 2006.
- [86] D. P. Kingma and M. Welling, "Auto-Encoding Variational Bayes," *arXiv preprint arXiv:1312.6114*, 2013.

[87] L. Grinsztajn, E. Oyallon, and G. Varoquaux, "Why Tree-Based Models Still Outperform Deep Learning on Tabular Data," in *Advances in Neural Information Processing Systems*, vol. 35, 2022.

[88] Raspberry Pi Foundation, "Raspberry Pi 5 Product Brief," 2023. [Online]. Available: <https://www.raspberrypi.com/products/raspberry-pi-5/>



REPUBLIQUE ALGERIENNE DEMOCRATIQUE ET POPULAIRE
Ministère de l'Enseignement Supérieur et de la Recherche Scientifique

Université Abou Bekr Belkaid Tlemcen

Business Model Canvas

BMC

Date de dépôt : 21/06/2026

N° de projet :

Faculté/Institut : Faculté de Technologie

Département : Télécommunication

Nom du projet : Conception et déploiement d'un système intelligent de détection d'intrusions sur des dispositifs Edge

Encadrant 1 : Dr. MOUSSAOUI Djilali

Encadrant 2 :

Co-encadrant 1 : Pr. Hadjila Mourad

Co-encadrant 2 :

Etudiants : - AISSANI Djamel Eddine

- BENMANSOUR Mohammed Omar

Année universitaire : 2025/2026

1- Proposition de valeur (Value Proposition) القيمة المقترحة

- a. Quels problèmes résolvons-nous pour nos clients ?

ما هي المشاكل التي نحلها لعملائنا ؟

Cybersecurity Sentinel résout le problème du coût élevé des solutions de cybersécurité en proposant un système IDS intelligent et abordable. Il permet la détection rapide des cyberattaques en temps réel, réduit les faux positifs grâce à l'IA et protège les données localement via l'Edge Computing. De plus, sa solution Plug & Play facilite l'intégration dans les infrastructures existantes.

- b. Quels besoins de nos clients satisfont nos produits ou services ?

ما هي الاحتياجات التي يلبيها منتجاتنا أو خدماتنا لعملائنا؟

Nos produits répondent aux besoins de sécurité réseau, de détection rapide des intrusions et de protection des données sensibles. Ils offrent une solution fiable, facile à déployer, économique et adaptée aux organisations souhaitant renforcer leur cybersécurité sans investir dans des infrastructures coûteuses.

- c. En quoi notre offre est-elle différente de celle de nos concurrents ?

في ماذا تختلف عروضنا عن تلك التي يقدمها منافسونا؟

Notre offre se distingue par son **faible coût**, son fonctionnement **100 % local (Edge Computing)**, sa **détection en temps réel** grâce à l'intelligence artificielle et son format **Plug & Play** facile à déployer. Elle fournit également un support en français et en arabe, tout en restant beaucoup moins chère que les solutions de cybersécurité traditionnelles.

- d. Quelles est notre proposition unique de valeur ?

ما هو العرض الفريد للقيمة لدينا؟

Notre proposition unique de valeur est un **système de détection d'intrusion intelligent basé sur l'IA, abordable, rapide et 100 % local**, fourni sous forme d'un **boîtier Plug & Play** facile à intégrer, offrant des performances avancées à un coût largement inférieur aux solutions concurrentes.

2- Segments de clients (Customer Segment) : أنواع العملاء

- a. Quels sont nos clients principaux?

من هم العملاء او الزبائن الرئيسيون ؟

Nos principaux clients sont les **PME**, qui constituent la cible prioritaire. L'offre s'adresse également aux **établissements publics** (hôpitaux, universités, administrations), aux **grandes entreprises** (banques, industries) ainsi qu'aux **particuliers et micro-entreprises**.

- b. Quels sont les différents segments de clients que nous visons ?

1
ما هي الفئات المختلفة من العملاء التي تستهدفها؟

Les segments de clients visés sont les **PME**, les **établissements publics** (hôpitaux, universités, administrations), les **grandes entreprises** (banques, industries) ainsi que les **particuliers et micro-entreprises**. Chaque segment a besoin d'une solution de cybersécurité performante et accessible.

- c. Quels sont les besoins spécifiques de chaque segment de clients?

ما هي الاحتياجات الخاصة لكل فئة من العملاء؟

- **PME** : besoin d'une cybersécurité efficace à faible coût et facile à déployer.

- **Établissements publics** : besoin de protéger des données sensibles tout en garantissant leur confidentialité.
- **Grandes entreprises** : besoin d'une détection rapide et fiable des cyberattaques à grande échelle.
- **Particuliers et micro-entreprises** : besoin d'une solution simple, accessible et sécurisée pour protéger leur réseau

d. Comment pouvons-nous catégoriser nos clients en groupes distincts?

كيف يمكن تصنيف عملائنا الى مجموعات مختلفة؟

Nous pouvons catégoriser nos clients en quatre groupes distincts : **PME, établissements publics, grandes entreprises** et **particuliers/micro-entreprises**. Cette segmentation est basée sur leur taille, leurs ressources informatiques et leurs besoins en cybersécurité.

3- Relation avec les clients (Consumer Relationships) : علاقة مع العملاء

a. Quel type de relation chaque segment de clients attend-il de nous ?

اي نوع من العلاقة يتوقعه كل فئة من العملاء منا؟

Chaque segment de clients attend une **relation de confiance et un support technique réactif**. Cela inclut une assistance par email, chat ou téléphone, une documentation claire, un accompagnement personnalisé lors du déploiement et des mises à jour régulières du système pour faire face aux nouvelles menaces.

b. Comment entretenons-nous actuellement les relations avec nos clients ?

كيف نحافظ حاليًا على العلاقات مع عملائنا؟

Nous entretenons actuellement les relations avec nos clients grâce à un **support technique par email, chat et téléphone**, une **documentation en ligne**, un **accompagnement personnalisé lors du déploiement** et des **mises à jour régulières des modèles d'IA** pour garantir une protection continue.

c. Comment pouvons-nous améliorer ou personnaliser nos interactions avec nos clients ?

كيف يمكننا تحسين أو تخصيص تفاعلاتنا مع عملائنا؟

Nous pouvons améliorer nos interactions avec les clients en proposant un **suivi personnalisé**, des **formations d'utilisation**, un **tableau de bord de surveillance en temps réel** et des **alertes de sécurité adaptées à leurs besoins spécifiques**. Cela permet de renforcer la satisfaction et la fidélisation des clients.

4-Canaux de distribution (Channels) : قنوات التوزيع

a- Par quels canaux nos clients veulent-ils être atteints ?

من خلال أي قنوات يفضل عملاؤنا أن يتم التواصل معهم؟

Nos clients souhaitent être atteints par plusieurs canaux : vente directe, site web dédié, LinkedIn et Facebook Pro, partenaires intégrateurs IT, revendeurs de composants électroniques, ainsi que lors de salons spécialisés en technologie et cybersécurité.

b- Quels canaux sont les plus efficaces pour atteindre chaque segment de clients ?

ما هي القنوات الأكثر فعالية للوصول إلى كل فئة من العملاء؟

- **PME** : vente directe, site web et réseaux professionnels (LinkedIn).
- **Établissements publics** : appels d'offres publics et partenariats institutionnels.
- **Grandes entreprises** : partenaires intégrateurs IT et démarche commerciale directe.
- **Particuliers et micro-entreprises** : site web, marketplaces et revendeurs de composants électroniques.

c- Comment pouvons-nous intégrer différents canaux pour améliorer l'expérience clients ?

كيف يمكننا دمج مختلف القنوات لتحسين تجربة العملاء؟

Nous pouvons intégrer différents canaux en combinant le **site web**, les **réseaux professionnels**, la **vente directe** et le **support technique** afin d'assurer une communication cohérente avec les clients. Cette approche permet d'offrir une expérience fluide, depuis la découverte du produit jusqu'au suivi après-vente

الشراكة الرئيسية : 5-Partenaires clés (Key Partnerships)

a. Qui sont nos partenaires clés ?

من هم شركائنا الرئيسيون؟

Nos partenaires clés sont les **laboratoires de recherche de la faculté**, les **plateformes d'innovation comme Ibtikar**, les **fournisseurs de matériel Edge** (Raspberry Pi, NVIDIA Jetson) ainsi que les **bêta-testeurs** tels que les PME locales et les départements informatiques universitaires.

b. Quels sont les partenariats qui nous aident à réduire les coûts, à accéder à de nouvelles ressources ou à améliorer notre proposition de valeur ?

ما هي الشراكات التي تساعدنا على خفض التكاليف أو الوصول إلى موارد جديدة أو تحسين قيمتنا المقترحة؟

Les partenariats avec les **laboratoires de recherche**, les **plateformes d'innovation (Ibtikar)** et les **fournisseurs de matériel Edge** nous permettent de réduire les coûts de développement, d'accéder à des ressources techniques et financières, et d'améliorer continuellement notre solution grâce à l'expertise scientifique et technologique.

c. Comment pouvons-nous aligner nos intérêts avec ceux de nos partenaires ?

كيف يمكننا مزامنة مصالحنا مع تلك لشركائنا؟

Nous pouvons aligner nos intérêts avec ceux de nos partenaires en établissant des collaborations durables, en partageant les bénéfices de l'innovation, en échangeant des retours d'expérience et en développant ensemble des solutions qui créent de la valeur pour toutes les parties prenantes.

6-Activités clés (Key Activities): الأنشطة الرئيسية:

- a. Quelles sont les actions principales que nous devons entreprendre pour livrer notre proposition de valeur ?

ما هي الأنشطة الرئيسية التي يجب علينا القيام بها لتقديم قيمتنا المقترحة؟

Pour livrer notre proposition de valeur, nous devons réaliser la **R&D en intelligence artificielle**, développer les logiciels, assembler et configurer les kits IDS Edge Box, mettre à jour les signatures d'attaques et assurer le déploiement ainsi que l'intégration chez les clients.

- b. Quelles sont les opérations essentielles pour notre entreprise ?

ما هي العمليات الأساسية لشركتنا؟

Les opérations essentielles de l'entreprise sont le développement et l'optimisation des modèles d'IA, la production des kits IDS Edge Box, la maintenance et mise à jour du système, ainsi que le déploiement et le support technique auprès des clients

- c. Quelles sont les activités qui créent le plus de valeur pour nos clients ?

ما هي الأنشطة التي تخلق أكبر قيمة لعملائنا؟

Les activités qui créent le plus de valeur pour nos clients sont la détection intelligente des intrusions grâce à l'IA, l'amélioration continue des modèles de détection, la mise à jour des signatures d'attaques et l'accompagnement technique lors du déploiement et de l'utilisation de la solution

7- Ressources clés (Key resources): الموارد الرئيسية:

- a. Quels sont nos actifs matériels, immatériels et humains essentiels ?

ما هي الأصول المادية وغير المادية والبشرية الأساسية لدينا؟

Les actifs essentiels de l'entreprise comprennent les équipements matériels (Raspberry Pi, boîtiers Edge, routeurs), les actifs immatériels (modèles d'IA, logiciels, propriété intellectuelle, bases de données) ainsi que les ressources humaines composées d'ingénieurs, développeurs, personnel commercial, support technique et gestion.

- b. Quels sont les outils, les technologies ou les partenariats dont nous avons besoin pour réussir ?

ما هي الأدوات والتكنولوجيا أو الشراكات التي نحتاجها لتحقيق النجاح؟

Pour réussir, nous avons besoin d'outils et technologies tels que Python, Flask, Scapy, les modèles d'intelligence artificielle et les plateformes matérielles Edge. Nous devons également nous appuyer sur des partenariats avec les laboratoires de

recherche, les plateformes d'innovation et les fournisseurs de matériel afin de renforcer nos capacités techniques et commerciales.

- c. Quels sont les principaux avantages concurrentiels de nos ressources ?

ما هي المزايا التنافسية الرئيسية لمواردنا؟

Notre solution utilise le Raspberry Pi 5, ce qui réduit les coûts de déploiement. L'intelligence artificielle permet une détection rapide et automatique des anomalies réseau. Le système fonctionne en temps réel pour améliorer la sécurité des entreprises. Il est facile à installer et à adapter à différents environnements réseau. Enfin, il offre une solution innovante et évolutive pour la cybersécurité

8- Charges et coûts (Coste structure) : التكاليف:

- a. Quels sont les coûts fixes et variables associés à notre modèle économique ?

ما هي التكاليف الثابتة والمتغيرة المرتبطة بنموذجنا الاقتصادي؟

Nos ressources offrent plusieurs avantages concurrentiels : des modèles d'IA optimisés pour une détection rapide et précise, une infrastructure Edge garantissant la confidentialité des données, une expertise multidisciplinaire de l'équipe et une solution propriétaire à faible coût, adaptée aux besoins des organisations locales

- b. Quels sont les coûts les plus importants pour notre entreprise ?

ما هي التكاليف الأكثر أهمية لشركتنا؟

Les coûts les plus importants de l'entreprise sont les dépenses liées au développement et à la recherche (R&D), aux matières premières pour les kits matériels, au local et aux équipements, ainsi qu'aux activités de marketing, d'administration, de support et de maintenance.

- c. Comment pouvons-nous réduire les coûts ou améliorer l'efficacité de nos opérations ?

كيف يمكننا خفض التكاليف أو تحسين كفاءة عملياتنا؟

Nous pouvons réduire les coûts en optimisant l'utilisation des ressources matérielles, en automatisant certaines tâches de maintenance et de déploiement, et en renforçant les partenariats stratégiques. L'amélioration continue des modèles d'IA permet également d'accroître l'efficacité opérationnelle tout en limitant les dépenses.

9- Revenus (Revenue) : مصادر الدخل:

- a. Quels produits ou services nos clients sont-ils prêts à payer ?

ما هي المنتجات أو الخدمات التي يكون عملاؤنا على استعداد لدفع ثمنها؟

Les clients sont prêts à payer pour la Raspberry Pi, les abonnements aux mises à jour et aux services de maintenance, ainsi que les prestations de conseil et d'intégration en cybersécurité.

- b. Quels sont les différents moyens par lesquels nous pouvons générer des revenus ?

ما هي الطرق المختلفة التي يمكننا من خلالها تحقيق الدخل؟



Nous pouvons générer des revenus grâce à la vente des kits matériels Raspberry Pi, aux abonnements de mise à jour et de maintenance, ainsi qu'aux prestations de conseil et d'intégration en cybersécurité.

c. Quel est notre modèle de tarification ?

ما هو نموذج التسعير لدينا؟

Notre modèle de tarification repose sur la vente des kits matériels Raspberry Pi, complétée par des abonnements récurrents pour les mises à jour et la maintenance, ainsi que par la facturation de services de conseil et d'intégration en cybersécurité.

Partenaires clés - Laboratoires de recherche de la faculté (support scientifique). - Plateformes d'innovation (Ibtikar) pour l'incubation et le financement. - Fournisseurs de matériel Edge (distributeurs de Raspberry Pi, NVIDIA Jetson). - Bêta-testeurs (PME locales, départements IT universitaires).	Activités principales - R&D et Optimisation IA (Compression, quantification, distillation de modèles). - Développement des logiciels. - Assemblage & config. kits IDS Edge Box - Mise à jour signatures d'attaques (nouvelles CVE) - Déploiement et intégration chez les clients	Proposition de valeur - Sécurité IA abordable : Un système d'intrusion intelligent à une fraction du coût des solutions du marché. - Latence < 5 ms par flux (détection temps réel) - 6 types d'attaques couvertes. - Plug & Play : Un boîtier physique autonome facile à intégrer dans l'infrastructure existante. - Edge Computing : 100% local. - Support FR/AR — 50× moins cher que Palo Alto PA-200 - Kit IDS Edge Box clé en main. - Faible taux de faux positifs grâce à l'apprentissage comportemental.	Relation avec les clients - Support technique email/chat/tél. (< 24 h) - Documentation en ligne. - Accompagnement déploiement personnalisé - Mises à jour modèles IA (nouvelles attaques).	Clients PME (cible prioritaire) : Entreprises disposant de ressources IT limitées mais nécessitant une sécurité réseau avancée. Établissements publics : Cliniques/Hôpitaux, universités, administrations (sensibles à la confidentialité). Grandes entreprises - Banques, industrie Particuliers / micro-ent. - Licence seule
	Ressources clés Matérielles - Raspberry Pi 5 . - Dataset CIC-IDS 2017. - Algorithmes d'optimisation et modèles IA entraînés. - Local professionnel . - Matériel de test (Boîtiers Edge, switchs, routeurs). matérielles - Python/Flask/Scapy - Propriété intellectuelle (licences, savoir-faire) Humaines (9 pers.) - 2 ing. ML, 2 dev., 1 commercial, 1 support, 1 comptable, 1 CM, 1 gérant		Canaux de distribution - Vente directe (démarchage PME) - Site web dédié. - Marketplace Raspberry Pi (export) - Revendeurs composants électroniques - Partenaires intégrateurs IT - Appels d'offres publics - LinkedIn / Facebook Pro - Salons tech & cybersécurité	

Coûts

Total année 1 : 4 255 000 DA (≈ 5 950 €)

- Développement + R&D : 1 300 000 DA
- Matières premières : 1 095 000 DA
- Local + équipements : 1 090 000 DA
- Marketing + admin: 520 000 DA
- Support + maintenance : 250 000 DA

RÉSULTAT NET ANNÉE 1 : +645 000 DA (bénéfice)

Revenus

CA total année 1 : 4 900 000 DA (≈ 6 860 €)

- Licences : 1 500 000 DA
- Kits matériels : 2 400 000 DA
- Abonnements : 600 000 DA
- Consulting : 400 000 DA