RÉPUBLIQUE DU SÉNÉGAL

Un Peuple – Un But – Une Foi



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

UNIVERSITÉ GASTON BERGER DE SAINT-LOUIS



U.F.R. DES SCIENCES APPLIQUÉES ET DE TECHNOLOGIE
DÉPARTEMENT INFORMATIQUE

Mémoire de Fin d'Études

En vue de l'obtention du diplôme de

Master d'Ingénierie en Gestion de Données et Ingénierie Logicielle (GDIL)

Réduction des fausses alertes dans les systèmes IDS au moyen de modèles de langage (LLMs)

Présenté par :
Zeynabou Hanne DIALLO

Sous la direction de :
Pr. Chérif

Année académique : 2024–2025

Remerciements

Je tiens à remercier tout d'abord mon professeur encadrant pour ses conseils précieux et son accompagnement tout au long de ce travail. Je remercie également ma famille et mon mari pour leur soutien moral et leurs encouragements constants. Enfin, je remercie tous les collègues et amis qui m'ont aidée dans la collecte et l'analyse des données.

Résumé

À l'ère de la transformation numérique, le volume de menaces et d'intrusions s'accroît, ce qui fait de la cybersécurité une priorité stratégique pour les organisations. Les systèmes de détection d'intrusion (IDS) sont déployés pour identifier les activités suspectes, utilisant des approches par signatures ou par anomalies. Cependant, ces systèmes, en particulier ceux reposant sur les techniques traditionnelles ou les modèles de Machine Learning (ML) seuls, souffrent d'une limite majeure : la génération d'un taux élevé de faux positifs (FP). Ce phénomène sature les Security Operations Centers (SOCs), provoque une surcharge d'alertes et conduit à la fatigue des analystes (Alert Fatigue), réduisant l'efficacité globale des opérations de sécurité. Face à ces défis, les méthodes basées sur les statistiques de flux montrent leurs limites, car elles manquent de la capacité à analyser, contextualiser et comprendre les informations sémantiques complexes issues des journaux d'alertes. La recherche s'est orientée vers l'Intelligence Artificielle (IA) avancée.

Ce mémoire s'inscrit dans cette problématique en explorant une architecture hybride IDS+LLM appliquée au jeu de données CICIDS2018. Un classifieur XGBoost est d'abord utilisé pour générer des alertes à partir de caractéristiques de flux réseau, puis différents LLM sont sollicités pour requalifier ces alertes (trafic normal, faux positifs, faux négatifs, véritables attaques) et produire une justification textuelle à partir des seules métriques réseau disponibles. L'objectif est double : améliorer la qualité globale des décisions (en particulier sur les cas ambigus) et fournir des explications compréhensibles, afin de rapprocher la détection automatique des besoins réels des analystes SOC. Les résultats expérimentaux montrent que l'intégration du LLM permet une réduction significative des fausses alertes, notamment grâce à sa capacité à analyser le contexte sémantique des événements. Cette approche nous a permis d'atteindre un niveau de performance élevé et prometteur pour l'amélioration de la détection au sein d'un SOC.

Mots-clés : Cybersécurité, Système de détection d'intrusion (IDS), Centre Opérationnel de Sécurité (SOC), grand modèle de langage (LLM), Machine Learning, Faux positifs,fausse alerte.

Abstract

In the era of digital transformation, the volume of threats and intrusions continues to grow, making cybersecurity a strategic priority for organizations. Intrusion Detection Systems (IDS) are deployed to identify suspicious activities, using signature-based or anomaly-based approaches. However, these systems—especially those relying solely on traditional techniques or Machine Learning (ML) models—suffer from a major limitation : the generation of a high number of false positives (FP). This issue overwhelms Security Operations Centers (SOCs), leads to alert overload, and causes analyst fatigue, ultimately reducing the overall effectiveness of security operations.

Traditional flow-based statistical methods also show their limits, as they lack the ability to analyze, contextualize, and understand the complex semantic information contained in alert logs. As a result, research has increasingly turned toward advanced Artificial Intelligence (AI) techniques.

This thesis addresses this challenge by exploring a hybrid IDS+LLM architecture applied to the CICIDS2018 dataset. An XGBoost classifier is first used to generate alerts based on network flow features, and then various Large Language Models (LLMs) are employed to requalify these alerts (normal traffic, false positives, false negatives, true attacks) and generate textual justifications using only the available network metrics. The objective is twofold : improving the overall quality of decisions—particularly in ambiguous cases—and providing understandable explanations, thereby bridging automated detection with the real needs of SOC analysts.

Experimental results show that integrating LLMs significantly reduces false alerts, thanks to their ability to analyze the semantic context of events. This approach demonstrates a high and promising level of performance for enhancing detection within SOC environments.

Keywords : Cybersecurity, Intrusion Detection System (IDS), Security Operations Center (SOC), Large Language Model (LLM), Machine Learning, False Positives, False Alerts.

Table des figures

1.1 Flux de traitement d'une alerte dans un SOC 16

2.1 Architecture d'un NIDS 22
2.2 Architecture d'un IDS basé sur signatures. 24
2.3 Architecture d'un IDS basé sur anomalies 25
2.4 Comparaison des différentes méthodes de détection. 26
2.5 Architecture de fonctionnement SNORT 27

3.1 Architecture de fonctionnement d'un LLM 41
3.2 Fonctionnement du processus d'expérience 43
3.3 google Colaboratory 45
3.4 Distribution du trafic dans CSE-CIC-IDS2018 stratifié 49
3.5 resultats des LLM sur les 4 alertes 58
3.6 Performance des modèles 60
3.7 Matrice de confusionpar types d'alerte 62
3.8 distribution des scores de confiance et le taux de reussite par type de menace 63
3.9 extrait du fichier JSON des alertes 64
3.10 Exemple de sortie 69

Liste des tableaux

2.1 Comparatif synthétique entre IDS open-source et commerciaux 28
2.2 Comparaison entre faux positifs et faux négatifs dans les IDS 28
2.3 Matrice des prédictions IDS 29
2.4 Impact des faux positifs sur les opérations SOC 31

3.1 Distribution des échantillons dans le dataset CSE-CIC-IDS2018 47
3.2 Distribution du dataset combiné (820 000 échantillons) par catégories d'attaques 48
3.3 Impact du nettoyage initial sur la dimensionnalité 50
3.4 Distribution détaillée des classes après encodage 51
3.5 Métriques globales du modèle XGBoost 53
3.6 Performance du modèle XGBoost sur les classes principales 54
3.7 Classes nécessitant une amélioration 54
3.8 Extraits de la requalification des alertes par les modèles LLM 60
3.9 Aperçu des résultats des LLMs 61
3.10 Performance détaillée par type d'alerte (80 alertes) 61
3.11 Matrice de couverture des catégories de menaces 62
3.12 Composition des exemples few-shot 65
3.13 Performance comparative des approches de prompting (10 alertes) 68

Liste des abréviations

API	Application Programming Interface
CIC	Canadian Institute for Cybersecurity
CNN	Convolutional Neural Network
CoT	Chain-of-Thought
DDoS	Distributed Denial of Service
DL	Deep Learning
DoS	Denial of Service
EDA	Exploratory Data Analysis
FN	False Negative
FP	False Positive
FPR	False Positive Rate
GPU	Graphics Processing Unit
HIDS	Host Intrusion Detection System
IA	Intelligence Artificielle
IDPS	Intrusion Detection and Prevention System
IDS	Intrusion Detection System
IoT	Internet of Things
LLaMA	Large Language Model Meta AI
LLM	Large Language Model
LSTM	Long Short-Term Memory
ML	Machine Learning
NaN	Not a Number
NIDS	Network Intrusion Detection System
NIST	National Institute of Standards and Technology

SIEM	Security Information and Event Management
SOC	Security Operations Center
SOAR	Security Orchestration, Automation and Response
TPU	Tensor Processing Unit

Table des matières

Liste des abréviations **6**

Introduction Générale **11**

1 Les défis opérationnels des Security Operations Centers (SOC) **14**

Introduction 14

1.1 Architecture et rôle d'un SOC moderne 15

1.1.1 Composition typique d'un SOC 15

1.1.2 Flux de traitement des alertes 15

1.1.3 Rôle stratégique du SOC 16

1.2 Un défi majeur : la surcharge d'alertes et la fatigue des analystes 17

1.2.1 Volume massif d'alertes 17

1.2.2 Prévalence des faux positifs 17

1.2.3 Fatigue des analystes (*Alert Fatigue*) 18

1.2.4 Solutions et pistes d'amélioration 18

Conclusion 19

2 Les systèmes de détection d'intrusion (IDS) et le problème des faux positifs **20**

Introduction 20

2.1 Définition et rôle des IDS 21

2.2 Typologie des IDS 21

2.2.1 IDS basés sur le réseau (NIDS) 21

2.2.2 IDS basés sur l'hôte (HIDS) 22

2.2.3 Détection basée sur les signatures 23

2.2.4 Détection basée sur les anomalies 24

2.2.5 IDS hybrides 25

2.2.6 Comparaison des différentes catégories d'IDS 25

2.2.7 Discussion 25

2.2.8 Exemples d'outils IDS open-source et commerciaux 26

2.3 Le problème endémique des faux positifs 28
2.3.1 Définition et mécanismes 28
2.3.2 Causes techniques des faux positifs 29
2.3.3 Exemple concret de règle problématique 30
2.3.4 Étude de cas : analyse d'alertes IDS typiques 30
2.3.5 Impact quantifié sur les opérations SOC 31
2.3.6 Méthodes traditionnelles de mitigation 31
2.3.7 Limites des approches existantes 33
2.4 Revue des approches de réduction des faux positifs (FPR) 34
Revue des approches de réduction des faux positifs (FPR) 34
2.4.1 Sélection de caractéristiques et optimisation métaheuristique 34
2.4.2 Modèles d'ensemble, systèmes hybrides et filtration contextuelle . . 35
2.4.3 Apports du deep learning : autoencodeurs, CNN, LSTM et architectures avancées 35
2.4.4 Pipelines en deux étapes (two-stage detection) 36
2.4.5 Contributions des LLMs et analyse sémantique des alertes 36
Conclusion 37

3 Strategie de reduction des fausses alertes des IDS par les LLMs 38
Introduction 38
3.1 Les modèles de langage à grande échelle (LLMs) 39
3.1.1 Fonctionnement interne des LLMs 39
3.1.2 Tokenisation 40
3.1.3 Encodage positionnel 40
3.1.4 Blocs Transformer 40
3.1.5 Décodage et sortie 40
3.1.6 Exemples de modèles de LLM 40
3.1.7 Pourquoi les LLMs constituent une solution prometteuse pour les IDS 42
3.2 Approches méthodologiques 43
3.2.1 Environnement Expérimental 44
3.2.2 Présentation du Jeu de Données CSE-CIC-IDS2018 et Analyse Exploratoire (EDA) 45
3.2.3 Prétraitement des données et *Feature Engineering* 49
3.2.4 Implémentation et Évaluation du Modèle XGBoost 53
3.2.5 Analyse des limitations et proposition de solution LLM 55
3.3 Architecture de la solution hybride 56
3.3.1 Conception du système hybride 56
3.3.2 Stratégie de sélection des cas ambigus 57
3.3.3 Implémentation technique de l'interface LLM 57

3.3.4 Validation expérimentale sur cas réels 58
3.4 Résultats expérimentaux à grande échelle et validation statistique 59
3.4.1 Analyse comparative des performances 59
3.4.2 Impact sur les faux positifs et la charge opérationnelle 63
3.4.3 Analyse différenciée des modèles LLM 63
3.5 Stratégies de prompting et analyse des raisonnements LLM 65
3.5.1 Few-shot learning 65
3.5.2 Chain-of-Thought (CoT) 66
3.5.3 Évaluation comparative Zero-shot / Few-shot / CoT 67
Conclusion 70

Conclusion Générale **71**

Introduction Générale

Avec l'expansion d'Internet, l'essor du cloud, la virtualisation des infrastructures et la multiplication des équipements connectés, les systèmes d'information sont aujourd'hui exposés à un volume croissant de menaces, d'intrusions et d'attaques.La cybersécurité s'impose ainsi comme une priorité stratégique pour assurer la confidentialité, l'intégrité et la disponibilité des ressources numériques. Pour renforcer la protection des réseaux, les organisations déploient différents dispositifs de défense, parmi lesquels les pare-feux et les *Intrusion Detection Systems* (IDS) occupent une place centrale.

Les IDS ont pour rôle d'identifier les activités suspectes au sein d'un réseau, en s'appuyant principalement sur deux approches : la détection par signatures, qui compare les flux réseau à des attaques connues, et la détection par anomalies, qui vise à repérer les écarts par rapport au comportement normal.[10] Malgré leur importance, ces systèmes présentent une limite majeure qui affecte fortement l'efficacité opérationnelle des *Security Operations Centers* (SOCs) : la génération d'un taux élevé de faux positifs (FP). Dans les environnements modernes, où les infrastructures génèrent des milliers, voire des millions d'événements par jour, plusieurs études montrent qu'une majorité des alertes produites par les outils de détection sont peu pertinentes ou erronées, ce qui alimente un phénomène de surcharge et d'*alert fatigue* chez les analystes.

Les SOCs jouent pourtant un rôle central dans la défense opérationnelle : ils assurent une surveillance continue des systèmes d'information, collectent et corrèlent les événements de sécurité, détectent les comportements anormaux et coordonnent les réponses aux incidents. Lorsque les IDS, SIEM et autres outils de détection génèrent un volume massif d'alertes, dont une grande proportion se révèle être des faux positifs, la charge de travail des analystes augmente, la vigilance diminue et le risque de passer à côté d'incidents réellement critiques s'accroît.[8, 7] La réduction des faux positifs apparaît ainsi comme un enjeu majeur pour restaurer la performance et la réactivité opérationnelle des SOCs.

Dans ce contexte, l'Intelligence Artificielle (IA), et plus particulièrement les *Large Language Models* (LLMs), ouvre de nouvelles perspectives pour améliorer la qualité de détection au sein des environnements SOC.[15, 29] Contrairement aux approches classiques de *Machine*

Learning, souvent centrées sur des vecteurs de caractéristiques numériques, les LLMs sont capables d'analyser, de contextualiser et de comprendre des informations textuelles complexes, telles que les journaux, les métadonnées ou les descriptions d'alertes générées par les systèmes de sécurité.[34, 32] Grâce à leur capacité de raisonnement avancée, ils peuvent intervenir comme un filtre intelligent, capable de distinguer les comportements réellement malveillants des usages légitimes, d'enrichir les événements avec du contexte et de proposer des explications utiles aux analystes.

Ce mémoire s'inscrit précisément dans cette dynamique. Il propose une approche centrée sur la réduction des faux positifs dans les IDS à l'aide de modèles de langage avancés, en s'appuyant sur le dataset *CSE-CIC-IDS2018*, l'un des ensembles de données de référence pour l'évaluation des systèmes de détection d'intrusion.[31, 27] Le travail comprend une analyse approfondie du dataset, un prétraitement rigoureux, la conception d'un modèle hybride combinant un classifieur de type *Machine Learning* et un LLM, ainsi qu'une évaluation expérimentale détaillée.

L'objectif principal de ce mémoire est de proposer une approche hybride combinant modèles classiques de détection et modèles de langage afin de réduire le nombre de faux positifs produits par les IDS, tout en maintenant un haut niveau de précision. Plus spécifiquement, ce travail vise à :

- analyser les défis opérationnels liés aux alertes dans les SOCs et à la surcharge d'informations ;
- étudier les limites des IDS modernes, notamment leur tendance à produire un grand nombre de faux positifs ;
- exploiter les capacités des LLMs pour reclassifier, enrichir ou expliquer les alertes générées par les IDS ;
- évaluer expérimentalement l'impact de cette approche sur le dataset CSE-CIC-IDS2018 ;
- proposer un pipeline complet pouvant être appliqué, à terme, dans un environnement SOC réel.

Pour atteindre ces objectifs, la méthodologie adoptée s'articule autour de plusieurs étapes : (1) une analyse du fonctionnement des SOCs et des IDS, ainsi que des travaux récents sur la réduction des faux positifs ; (2) un prétraitement avancé et une exploration du dataset CSE-CIC-IDS2018 ; (3) l'entraînement et l'évaluation de modèles classiques de détection, notamment à base de *Machine Learning* ; (4) l'intégration et l'évaluation de modèles de langage dans une architecture hybride visant à requalifier les alertes et à réduire les faux positifs.

Ce mémoire est structuré comme suit :

- **Chapitre 1** : Présentation des SOCs, de leur architecture, de leurs défis opérationnels,

et du problème central de la surcharge d'alertes et de l'*alert fatigue*.

— **Chapitre 2** : Analyse détaillée des systèmes de détection d'intrusion (IDS), de leurs approches (signatures, anomalies, hybrides), et de la problématique des faux positifs.

— **Chapitre 3** : Présentation des modèles de langage, de leur fonctionnement (Transformers, LLMs) , ainsi que la conception de l'architecture hybride proposée , expérimentations, évaluation des résultats sur le dataset CSE-CIC-IDS2018 et discussion des performances et des limites de l'approche.

.

En définitive, ce travail contribue à la réflexion actuelle sur l'intégration de l'intelligence artificielle dans les opérations de sécurité, en proposant une approche innovante et applicable pour diminuer le volume d'alertes inutiles dans les SOCs et améliorer la qualité globale de la détection.

Chapitre 1

Les défis opérationnels des Security Operations Centers (SOC)

Introduction

La transformation numérique des organisations, l'augmentation du volume de données et la montée en sophistication des cybermenaces ont positionné les *Security Operations Centers* (SOCs) comme des piliers essentiels de la défense informatique moderne. Leur mission principale consiste à assurer une surveillance continue, analyser les événements de sécurité et réagir rapidement en cas d'incident.

Cependant, malgré l'importance stratégique des SOCs, ces derniers font face à des défis opérationnels majeurs, dont le plus critique est sans doute la surcharge d'alertes. Les systèmes de détection génèrent aujourd'hui un volume massif d'événements, dont une grande proportion s'avère être des faux positifs. Cette situation entraîne une pression importante sur les analystes, provoque de la fatigue opérationnelle (*alert fatigue*) et réduit l'efficacité globale des opérations de sécurité. Les études récentes sur l'état des SOC montrent que cette surcharge n'est plus un phénomène marginal, mais une réalité quotidienne pour la majorité des équipes, avec des impacts mesurables sur leur capacité de détection et de réponse.

Ce chapitre vise à présenter l'architecture d'un SOC moderne, son fonctionnement opérationnel et les enjeux liés à la surcharge informationnelle. Il s'appuie sur des travaux et rapports récents (2023–2025) qui quantifient l'ampleur du problème et justifient la nécessité d'approches avancées, telles que les modèles de langage (LLMs), pour réduire les faux positifs et restaurer la valeur opérationnelle des alertes. Ces éléments serviront de fondement à la réflexion développée dans ce mémoire.

1.1 Architecture et rôle d'un SOC moderne

Un SOC est une unité centralisée chargée de surveiller, détecter et répondre aux incidents de cybersécurité. Il combine ressources humaines, processus industriels et technologies avancées pour assurer la continuité et la résilience des systèmes d'information.

1.1.1 Composition typique d'un SOC

Un SOC moderne repose généralement sur les éléments suivants :

- **Sources de données** : pare-feux, IDS/IPS, EDR, serveurs, applications, services cloud, équipements réseau, etc., qui alimentent en continu le SOC en journaux et événements.
- **Plateforme SIEM (Security Information and Event Management)** : élément central du SOC, le SIEM collecte, agrège, corrèle et normalise les événements de sécurité afin d'identifier les comportements suspects. Il constitue l'interface principale de monitoring du SOC.
- **Plateforme SOAR (Security Orchestration, Automation and Response)** : permet d'automatiser et d'orchestrer les réponses aux incidents, réduisant le temps d'intervention et la charge répétitive pesant sur les analystes.[35]
- **Threat Intelligence** : données internes et externes renseignant sur les campagnes d'attaque, les vulnérabilités et les comportements hostiles. Ces informations enrichissent les alertes pour améliorer la pertinence de la détection.[18]
- **Ressources humaines spécialisées** :
  - Analystes SOC niveau 1 : triage et analyse initiale des alertes.
  - Analystes SOC niveau 2 et 3 : investigation approfondie, corrélation avancée, coordination de la réponse.
  - Équipes de *threat hunting* : recherche proactive de menaces avancées et d'activités furtives.

1.1.2 Flux de traitement des alertes

Le fonctionnement d'un SOC repose sur la capacité à collecter, analyser et exploiter en temps réel les données issues de différentes sources. Le flux de traitement des données peut être représenté en plusieurs étapes successives.

1. **Collecte des journaux** depuis toutes les sources (systèmes, applications, réseaux, cloud), via agents, connecteurs ou protocoles de journalisation.
2. **Normalisation et corrélation** via le SIEM :
   - standardisation des formats de données ;
   - agrégation et élimination des doublons ;

— application de règles, corrélations et modèles statistiques ou de ML.

3. **Génération et priorisation des alertes** : classement par criticité, type d'actif concerné, contexte et niveau de confiance, afin d'orienter l'attention des analystes.
4. **Investigation par les analystes** : vérification de la validité de l'alerte (vrai ou faux positif), enrichissement contextuel, recherche d'indices complémentaires et qualification de l'incident.
5. **Réponse aux incidents** : confinement, éradication, remédiation, retour à la normale et documentation pour capitaliser l'expérience.

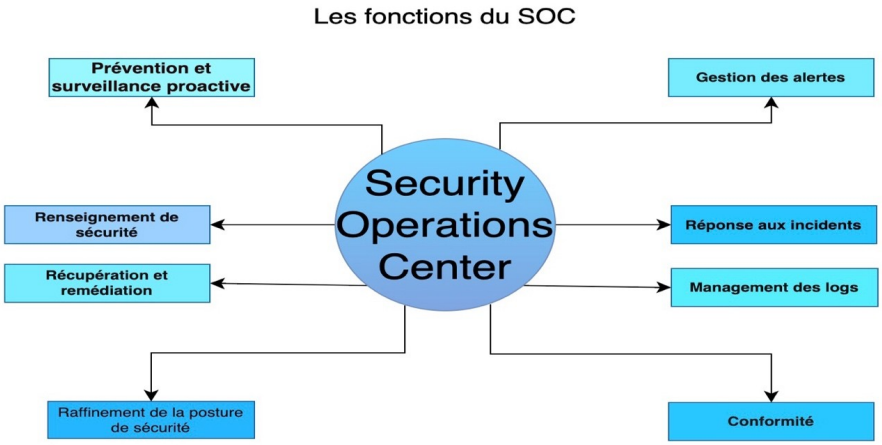


FIGURE 1.1 – Flux de traitement d'une alerte dans un SOC

1.1.3 Rôle stratégique du SOC

Au-delà de ses fonctions opérationnelles de surveillance et de réponse aux incidents, le Security Operations Center (SOC) occupe un rôle stratégique dans la gouvernance de la cybersécurité d'une organisation.[9] Son importance ne réside pas uniquement dans la détection des menaces en temps réel, mais également dans sa capacité à orienter les décisions stratégiques en matière de sécurité et de résilience numérique.

Premièrement, le SOC constitue une source de renseignement stratégique. Grâce à l'analyse des incidents, des menaces émergentes et des tendances observées, il fournit aux décideurs des informations exploitables pour adapter les politiques de sécurité, renforcer les contrôles existants et anticiper les risques futurs.[18]

Deuxièmement, le SOC joue un rôle clé dans la conformité réglementaire et la gestion des risques. En assurant la traçabilité des événements, la conservation des journaux de sécurité et la production de rapports, il facilite le respect des normes internationales (ISO 27001, NIST, GDPR, etc.) et des réglementations sectorielles.[26]

Troisièmement, il contribue directement à la résilience organisationnelle. En détectant et en neutralisant rapidement les cyberattaques, le SOC limite les impacts financiers, opérationnels et réputationnels liés aux incidents. De plus, il participe à la continuité des activités (*Business Continuity*) en assurant que les infrastructures critiques restent opérationnelles même en cas d'attaque.[26]

Enfin, le SOC soutient l'innovation et la transformation numérique. Dans un contexte où les entreprises adoptent massivement le cloud, l'IoT et l'intelligence artificielle, le SOC accompagne ces évolutions en intégrant des mécanismes de protection adaptés et en garantissant un cadre de confiance.[15] En somme, le SOC ne doit pas être perçu uniquement comme une unité technique, mais comme un acteur stratégique alignant la cybersécurité avec les objectifs globaux de l'organisation.

1.2 Un défi majeur : la surcharge d'alertes et la fatigue des analystes

Les Security Operations Centers (SOCs) modernes font face à un **volume extrêmement élevé d'alertes**, dont une grande partie se révèle être des **faux positifs**, créant un problème majeur pour la productivité et la qualité des analyses.

1.2.1 Volume massif d'alertes

Selon l'enquête *State of Threat Detection 2023* réalisée par Vectra AI, un SOC reçoit en moyenne plus de 4 400 alertes par jour.[2] Parmi celles-ci :

- environ 67 % des alertes ne sont jamais examinées par les analystes ;
- environ 80 % des alertes sont considérées comme des faux positifs ou de faible pertinence.[2]

Ce flux constant d'alertes non pertinentes surcharge les analystes et entraîne une perte de temps significative sur des événements qui ne représentent pas de véritables menaces. Les rapports de Devo sur l'état des SOC confirment que, dans de nombreux environnements, une proportion importante des alertes générées par les outils de sécurité n'est ni analysée ni corrélée de manière approfondie, faute de ressources suffisantes.[8, 7] Ces études convergent vers l'idée que les SOCs modernes fonctionnent en permanence en surcharge d'alertes, ce qui crée un contexte favorable à la baisse de vigilance et à l'augmentation des erreurs humaines.[2, 7]

1.2.2 Prévalence des faux positifs

Des recherches académiques récentes confirment ce phénomène :

— Teuwen et al. (2025) montrent que le taux de faux positifs dans certains IDS peut dépasser 80 %, surtout lorsque les règles sont trop larges ou mal calibrées.[30]
— Khalid & Purdie (2024) indiquent que les SOCs modernes déploient des outils d'IA et de ML pour filtrer les alertes, réduire les faux positifs et améliorer la pertinence des incidents soumis aux analystes.

Les faux positifs entraînent notamment :

— des erreurs dans la priorisation des incidents ;
— une diminution de la qualité des investigations ;
— une fatigue cognitive et émotionnelle accrue pour les analystes.

Ces résultats confirment que le problème des faux positifs n'est pas uniquement lié à la qualité intrinsèque des outils, mais aussi à la conception et au réglage des règles de détection, ainsi qu'au manque d'adaptation aux contextes spécifiques des organisations.

1.2.3 Fatigue des analystes (*Alert Fatigue*)

Le terme *alert fatigue* décrit le phénomène par lequel les analystes deviennent moins réactifs ou attentifs à cause du volume élevé d'alertes, en particulier celles qui sont fausses ou peu pertinentes. À force de traiter des événements redondants ou non critiques, les signaux réellement dangereux tendent à se fondre dans le bruit.

Les enquêtes sur l'état des SOC mettent en évidence que cette fatigue n'est pas seulement ressentie subjectivement, mais se traduit par une baisse de vigilance, une augmentation du nombre d'alertes ignorées et une difficulté accrue à maintenir un niveau de qualité constant dans les investigations. Singh et al. (2025) montrent que l'usage de LLMs ou d'agents IA peut réduire la fatigue cognitive en priorisant automatiquement les alertes critiques et en fournissant du contexte pour accélérer la prise de décision.[29] Dans le même esprit, Wei et al. proposent CORTEX, une architecture d'agents LLM collaboratifs dédiée au triage d'alertes à fort enjeu, illustrant concrètement la manière dont l'IA peut filtrer le bruit et mettre en avant les cas critiques.[34]

Les conséquences de cette fatigue incluent :

— des délais d'intervention plus longs sur les incidents critiques ;
— des erreurs ou omissions lors de l'analyse d'alertes légitimes ;
— une perte de motivation et un turnover accru au sein des équipes SOC.

1.2.4 Solutions et pistes d'amélioration

Les approches modernes pour réduire la surcharge et la fatigue comprennent :

— **Filtrage et corrélation intelligent** des alertes via SIEM et modèles de ML pour éliminer les faux positifs ou les regrouper avant présentation aux analystes.

— **Automatisation et orchestration** via SOAR et XDR, afin de déléguer les tâches répétitives à des workflows automatisés et de normaliser les scénarios de réponse.
— **Agents LLM et IA cognitive**, qui enrichissent les alertes, proposent des actions de remédiation et aident les analystes à prioriser les incidents critiques.

Ces approches préfigurent l'émergence de SOC « augmentés » par l'IA, où les analystes se concentrent sur les cas complexes et à fort enjeu, tandis que les tâches répétitives de tri, de corrélation et d'enrichissement sont largement prises en charge par des moteurs de ML et des agents LLM spécialisés.

Conclusion

Ce chapitre a présenté le fonctionnement d'un SOC, ses outils clés et son rôle stratégique dans la gouvernance de la cybersécurité. Il a également mis en évidence un problème majeur : la surcharge massive d'alertes et la fatigue des analystes, amplifiées par la prévalence des faux positifs.

Les études récentes confirment l'ampleur de ce défi, justifiant la recherche de solutions avancées telles que l'intelligence artificielle et les modèles de langage pour filtrer, enrichir et prioriser les alertes. Le chapitre suivant s'intéressera plus spécifiquement aux *Intrusion Detection Systems* (IDS), qui constituent l'une des principales sources d'alertes dans un SOC. Nous examinerons leur fonctionnement, leurs avantages, ainsi que la problématique des faux positifs qui en découle.

Chapitre 2

Les systèmes de détection d'intrusion (IDS) et le problème des faux positifs

Introduction

Afin de faire face à l'augmentation constante des cybermenaces, les organisations s'appuient largement sur les *Intrusion Detection Systems* (IDS). Ces systèmes jouent un rôle clé dans l'écosystème de sécurité en surveillant le trafic réseau et les activités des hôtes afin de détecter des comportements anormaux ou suspects. Ils constituent, aux côtés des pare-feux et des antivirus, l'une des premières lignes de défense dans la prévention des incidents de sécurité.

Toutefois, malgré leur efficacité théorique, les IDS souffrent d'un problème majeur : la génération massive de faux positifs. Un faux positif correspond à une alerte signalant une activité bénigne comme étant malveillante. Lorsque leur fréquence est élevée, ces alertes inutiles saturent les analystes et compliquent la détection des véritables menaces. Comme souligné dans le chapitre précédent, cette surcharge informationnelle contribue à la fatigue des analystes et à la baisse de performance globale des SOCs.

Ce chapitre examine en détail les systèmes de détection d'intrusion, en distinguant leurs types, leurs modes de fonctionnement et leurs principales caractéristiques. Nous analyserons ensuite le problème spécifique des faux positifs, ses causes et ses conséquences opérationnelles, avant d'introduire la nécessité d'approches innovantes capables d'en réduire l'occurrence.

2.1 Définition et rôle des IDS

Un *Intrusion Detection System* (IDS) est définie comme l'identification de l'utilisation non autorisée,du mauvais usage et de l'abus des systèmes informatique par des intrus internes et externes. Selon IBM [13], un IDS surveille le trafic réseau et les terminaux afin de détecter des menaces connues, des comportements suspects ou des violations des politiques de sécurité. Cette définition met en évidence la fonction essentielle des IDS : compléter les dispositifs préventifs (pare-feux, antivirus) en fournissant une capacité de surveillance et d'alerte en temps réel.

Le NIST définit les *Intrusion Detection and Prevention Systems* (IDPS) comme des composants qui « surveillent l'activité des systèmes et des réseaux, identifient les activités inappropriées, et signalent les incidents potentiels » [25]. Les IDS constituent alors une seconde ligne de défense, améliorant la visibilité des attaques échappant aux contrôles traditionnels. Ainsi, les IDS jouent un rôle essentiel dans la cybersécurité moderne :

- détection précoce des menaces, réduisant le temps de réaction ;
- fourniture d'informations critiques aux analystes SOC pour investigation et remédiation ;
- complément des autres solutions (SIEM, SOAR, EDR) pour une surveillance continue.

Malgré leur importance, les IDS sont confrontés à une limite majeure : la génération massive de faux positifs, qui affecte directement l'efficacité opérationnelle des SOCs.

2.2 Typologie des IDS

Les IDS peuvent être classés selon deux axes principaux :

1. la **source surveillée** : réseau ou hôte ;
2. la **méthode de détection** : signatures, anomalies ou approche hybride.

2.2.1 IDS basés sur le réseau (NIDS)

Les NIDS (Network IDS) sont déployés sur des segments réseau clés pour surveiller les flux entrants, sortants et internes. Ils analysent les paquets pour identifier des attaques externes ou des comportements suspects (scans, DDoS, tentatives d'exploitation). Ils fournissent une vision globale du réseau mais sont limités face au trafic chiffré ou aux activités internes non visibles depuis le réseau.

Principe de fonctionnement :

- capture du trafic via des capteurs réseau ;
- inspection des en-têtes et, si possible, de la charge utile ;
- application de règles (signatures) ou repérage de déviations (anomalies) ;

— génération d'alertes transmises à la console SOC.

Avantages :

— surveillance centralisée ;
— détection d'attaques externes ;
— pas d'agent nécessaire sur chaque machine.

Limites :

— inefficace face au trafic chiffré ;
— difficulté sur les réseaux à haut débit ;
— ne voit pas les attaques internes invisibles depuis le réseau.

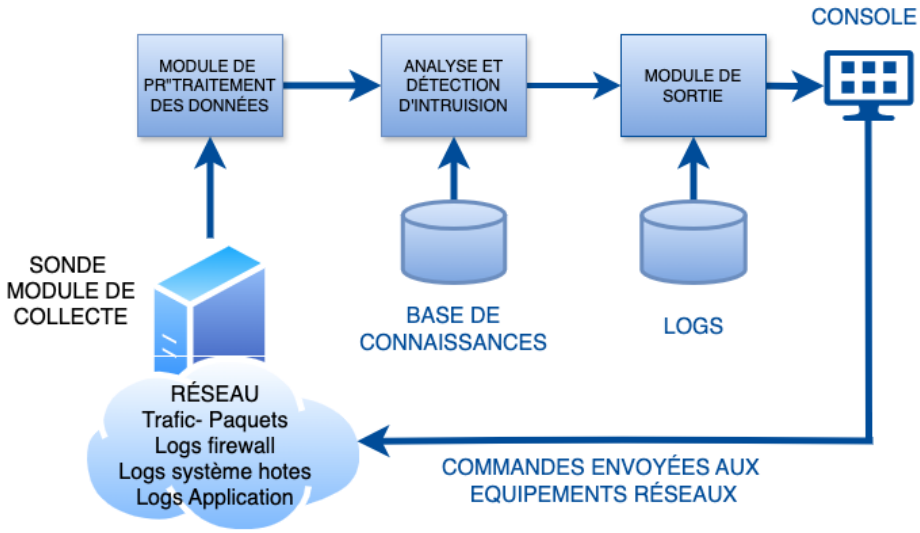


FIGURE 2.1 – Architecture d'un NIDS

2.2.2 IDS basés sur l'hôte (HIDS)

Un IDS hôte est un système qui analyse le trafic réseau à un niveau local, en se plaçant sur les machines individue lles du réseau, comme les serveurs, les postes de travail, ou les terminaux mobiles. Il peut ainsi surveiller le comportement de chaque machine, et détecter les intrusions qui affectent son fonctionnement.

Principe de fonctionnement :

— surveillance des logs, fichiers critiques et processus ;
— vérification de l'intégrité (hashing, comparaison) ;
— repérage d'anomalies locales et remontée d'alertes.

Avantages :

— détection des menaces internes ;
— précision sur l'activité locale ;
— utile en audit .

Limites :

— nécessite un agent sur chaque hôte ;
— consommation de ressources locales ;
— vision limitée à la machine surveillée.

2.2.3 Détection basée sur les signatures

La détection basée sur les signatures consiste à comparer le trafic réseau à une base de données de signatures, qui sont des motifs caractéristiques d'attaques connues. Par exemple, une signature peut correspondre à une chaîne de caractères spécifique, à un type de paquet, à une séquence d'actions, etc. Si le trafic réseau correspond à une signature, l'IDS déclenche une alerte.

Principe de fonctionnement :

— collecte des données réseau ou logs ;
— comparaison avec la base de signatures ;
— déclenchement d'alerte si correspondance exacte.

Avantages :

— faible taux de faux positifs ;
— précise et efficace pour détecter les attaques connues ;
— simple à mettre en œuvre.

Limites :

— incapable de détecter les attaques zero-day ;
— mise à jour constante nécessaire ;
— peut être contournée par des attaques inconnues ou modifiées.

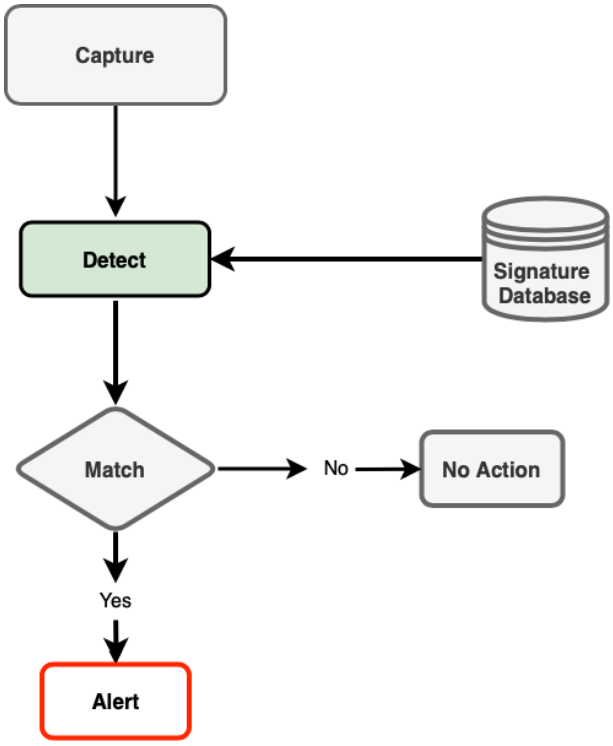


FIGURE 2.2 – Architecture d'un IDS basé sur signatures.

2.2.4 Détection basée sur les anomalies

Elle consiste à analyser le trafic réseau à partir d'un modèle de référence, qui représente le comportement "normal" du réseau. Ce modèle peut être prédéfini ou automatiquement appris par le système de détection d'intrusion grâce à des techniques d'intelligence artificielle en cybersécurité. . Si le trafic réseau s'écarte du modèle de référence, l'IDS déclenche une alerte. Ils détectent des attaques inconnues mais génèrent souvent beaucoup de faux positifs.

Principe de fonctionnement :

- phase d'apprentissage (statistique ou ML) ;
- comparaison en temps réel avec le modèle ;
- déclenchement d'alertes en cas de déviation.

Avantages :

- détection de menaces inédites ;
- adaptabilité aux changements.

Limites :

- taux élevé de faux positifs ;
- peut générer des faux négatifs
- nécessite une bonne connaissance du réseau et de ses activités normales, pour éviter les erreurs de modélisation ;
- risques de sur-/sous-apprentissage.

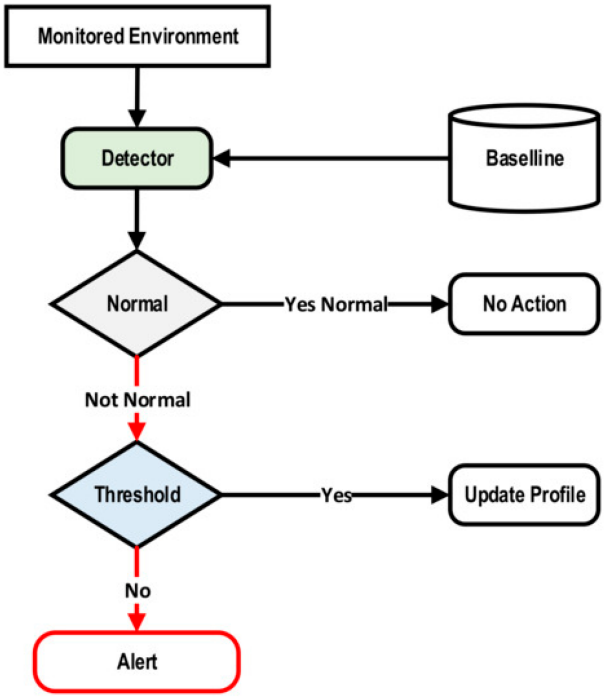


FIGURE 2.3 – Architecture d'un IDS basé sur anomalies .

2.2.5 IDS hybrides

Les IDS hybrides combinent signatures et anomalies pour profiter des avantages des deux.

Principe de fonctionnement :

- module signature pour menaces connues ;
- module anomalie pour écarts ;
- corrélation pour réduire les erreurs.

Avantages :

- meilleure couverture ;
- réduction relative des faux positifs.

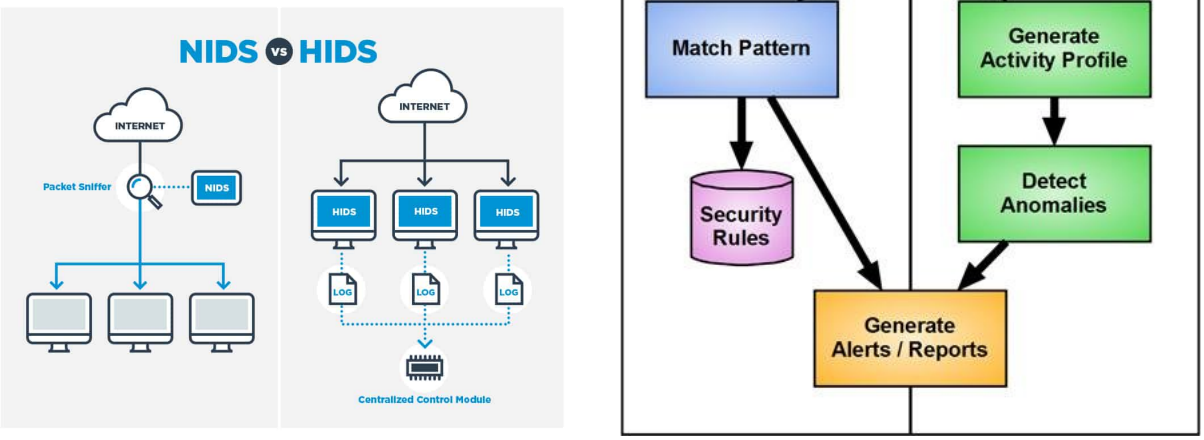
Limites :

- complexité de mise en œuvre ;
- coût en ressources et maintenance.

2.2.6 Comparaison des différentes catégories d'IDS

2.2.7 Discussion

La typologie des IDS montre un compromis constant entre la **précision** et la **couverture**. Les IDS basés sur signatures sont fiables mais limités aux menaces connues, tandis que les IDS basés sur anomalies détectent des attaques inédites mais génèrent beaucoup de faux positifs. Les IDS hybrides, plus performants, soulèvent des défis de mise en œuvre et de



(a) HIDS vs NIDS (b) Signature vs Anomaly

FIGURE 2.4 – Comparaison des différentes méthodes de détection.

maintenance. La réduction des faux positifs demeure un enjeu critique pour l'efficacité des SOCs.

2.2.8 Exemples d'outils IDS open-source et commerciaux

Les systèmes de détection d'intrusion sont disponibles sous plusieurs formes, qu'ils soient **open-source** (gratuits et personnalisables) ou **commerciaux** (proposant un support et des fonctionnalités avancées).

1. Outils open-source Il existe plusieurs solutions d'IDS dans le marché. Nous avons les IDS/IPS des fournisseurs comme Cisco, Huawei, Juniper, etc. qui sont des équipements dédiés, les NIDS open source comme SNORT, Surricata, les HIDS open source comme Tripwire, OSSEC, AIDE. En guise d'exemple, nous allons étudier SNORT.

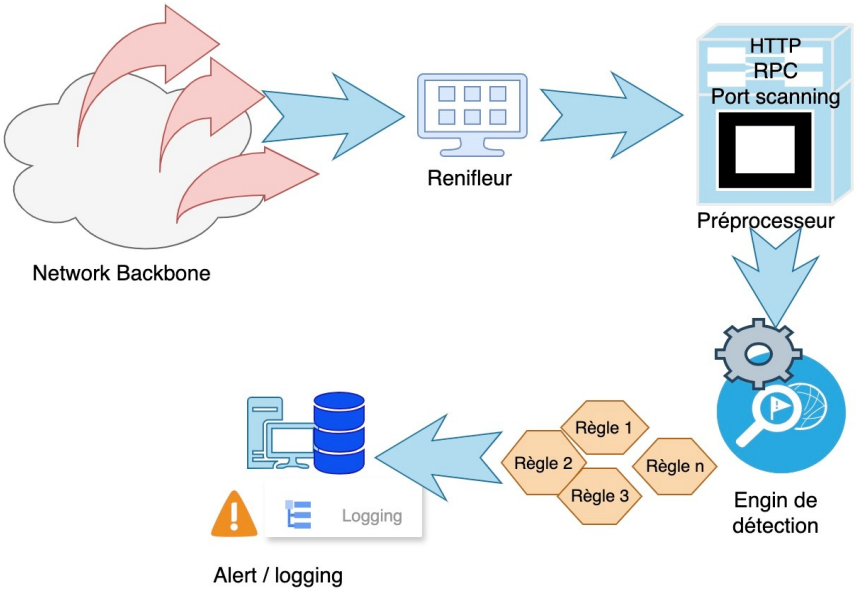


FIGURE 2.5 – Architecture de fonctionnement SNORT

Snort est un système de détection d'intrusions (IDS) basé sur la technique de détection par signatures [24]. C'est une solution open-source, légère et reconnue pour son efficacité et sa flexibilité [26]. Développé par Martin Roesch, Snort possède des capacités avancées de capture et de filtrage du trafic réseau [24].

La Figure **??** illustre l'architecture de Snort, qui se compose de plusieurs éléments clés. Le bloc d'acquisition de données, également appelé renifleur de paquets, peut être un logiciel ou un dispositif matériel permettant d'intercepter le trafic réseau [10]. Sa fonction principale est d'écouter les données circulant sur le réseau, puis de les traiter grâce au préprocesseur, qui améliore la capacité du système à analyser les paquets [26].

Les paquets capturés sont ensuite analysés via des plug-ins spécialisés, par exemple pour détecter des scans de ports ou des appels RPC [10]. Après cette analyse, le bloc de détection compare les paquets aux règles prédéfinies. Si une correspondance est trouvée, les données sont envoyées au préprocesseur d'alertes, chargé de gérer efficacement les potentielles intrusions [24].

2. Solutions commerciales

- **Cisco Firepower** : intègre Snort avec une interface de gestion et des fonctions de corrélation avancées.
- **IBM QRadar Network Insights** : combine IDS, SIEM et corrélation d'événements à grande échelle.
- **McAfee Network Security Platform** : fournit une détection en temps réel avec mises à jour automatiques de signatures.

TABLE 2.1 – Comparatif synthétique entre IDS open-source et commerciaux

Type d'IDS	**Exemples**	**Caractéristiques principales**
Open-source	Snort, Suricata, Zeek	Flexibles, gratuits, nécessitent une maintenance manuelle
Commerciaux	Cisco Firepower, IBM QRadar, McAfee NSP	Support technique, gestion centralisée, mises à jour automatiques

2.3 Le problème endémique des faux positifs

2.3.1 Définition et mécanismes

Les systèmes de détection d'intrusion génèrent deux types d'erreurs majeures : les **faux positifs** et les **faux négatifs**.

— Un **faux positif** survient lorsqu'il identifie le trafic normal comme malveillant. Par exemple, un trafic applicatif normal (comme une mise à jour Windows ou une requête DNS volumineuse) peut être signalé comme une attaque.

— Un **faux négatif** correspond à une attaque réelle qui échappe à la détection, laissant le système vulnérable.

TABLE 2.2 – Comparaison entre faux positifs et faux négatifs dans les IDS

Type d'erreur	**Définition**	**Conséquences principales**
Faux positif	Activité normale signalée comme malveillante	Surcharge des analystes, perte de temps, *alert fatigue*, baisse de vigilance
Faux négatif	Attaque réelle non détectée	Intrusion non repérée, compromission potentielle du système, fuite de données

Dans les IDS, les faux positifs sont souvent liés à la nature même des mécanismes de détection.

— Les **IDS à base de signatures** sont sensibles aux variations de trafic qui ressemblent à des attaques connues.

— Les **IDS à base d'anomalies** ont tendance à signaler comme suspectes des activités légitimes qui ne correspondent pas au profil appris.

TABLE 2.3 – Matrice des prédictions IDS

	Attaque réelle	**Pas d'attaque**
Alerte IDS	Vrai Positif (VP)	Faux Positif (FP)
Pas d'alerte	Faux Négatif (FN)	Vrai Négatif (VN)

En pratique, la réduction des faux positifs reste un enjeu plus critique que celle des faux négatifs, car leur volume massif engendre une surcharge opérationnelle pour les SOC.

2.3.2 Causes techniques des faux positifs

Les faux positifs sont inhérents au fonctionnement des IDS et trouvent leur origine dans plusieurs causes techniques. Comprendre ces causes est essentiel pour envisager des solutions efficaces de réduction.

— **Règles de détection trop générales :** Les IDS basés sur signatures utilisent souvent des motifs génériques. Par exemple, une règle Snort détectant toute requête HTTP contenant le mot-clé « admin » peut générer de multiples faux positifs lorsqu'un utilisateur accède à une page légitime contenant ce terme.

— **Variabilité normale du trafic réseau :** Le trafic peut présenter des fluctuations inhabituelles mais légitimes (pics de bande passante, transferts de fichiers volumineux) qui sont confondus avec des comportements d'attaque.

— **Applications légitimes ressemblant à des attaques :** Certaines applications comme les scanners de vulnérabilités internes ou des outils d'administration réseau produisent des schémas de trafic similaires à ceux d'attaques (ex. scans de ports).

— **Mauvaise configuration ou règles obsolètes :** L'une des principales causes de faux positifs est un IDS mal configuré. Cela peut se produire lorsque l'IDS n'est pas correctement adapté à l'environnement réseau, aux stratégies, aux protocoles ou aux périphériques. Par exemple, si l'IDS utilise des signatures, des règles ou des seuils obsolètes ou inappropriés, il peut déclencher des alertes pour un trafic normal ou inoffensif. Des signatures non mises à jour ou des configurations trop strictes entraînent un grand nombre d'alertes non pertinentes.

— **Évolution rapide des systèmes :** L'introduction de nouvelles applications, protocoles ou services dans l'environnement peut engendrer des comportements non reconnus par l'IDS, déclenchant ainsi des alertes injustifiées.l'évolution des menaces, c'est-à-dire des attaques nouvelles ou émergentes que l'IDS ne reconnaît pas ou n'anticipe pas. Il peut s'agir d'exploits zero-day, de menaces persistantes avancées, de logiciels malveillants polymorphes ou de techniques furtives qui contournent ou échappent à l'IDS. Par exemple, si l'IDS détecte un paquet d'apparence normale qui contient une charge utile cachée ou intégrée, il peut manquer ou ignorer une

attaque possible, et alerter ultérieurement pour un faux positif lorsque la charge utile est exécutée ou activée.

Ces causes montrent que les faux positifs ne sont pas seulement liés à la faiblesse des algorithmes, mais aussi à la complexité des environnements informatiques modernes. Ils résultent d'un compromis constant entre **sensibilité** (capacité à détecter toutes les menaces) et **spécificité** (capacité à ignorer les événements bénins).

2.3.3 Exemple concret de règle problématique

Dans les systèmes de détection d'intrusions basés sur les signatures, comme **Snort** ou **Suricata**, certaines règles sont trop génériques ou mal configurées, ce qui entraîne un nombre élevé de **faux positifs (FP)** et surcharge les analystes SOC.

par exemple :

```
# Règle Snort trop générique générant des faux positifs
alert tcp any any -> any any (msg:"Potential Scan"; flags:S; sid:1000001;)
```

Cette règle alerte sur TOUT paquet TCP avec flag SYN, y compris les connexions légitimes, générant des milliers de faux positifs.

2.3.4 Étude de cas : analyse d'alertes IDS typiques

Scénario 1 : Scan de ports bénin

- **Alerte IDS** : "Network Scan Detected from Internal IP"
- **Activité réelle** : Outil de monitoring réseau légitime (Nagios, Zabbix)
- **Investigation** : 30 minutes pour identifier la source légitime
- **Impact** : Perte de temps et distraction des analystes

Scénario 2 : Fausse tentative de connexion

- **Alerte IDS** : "Multiple Failed SSH Attempts"
- **Réalité** : Administrateur ayant saisi incorrectement ses credentials
- **Investigation** : Alerte traitée comme prioritaire en urgence
- **Impact** : Mobilisation inutile des ressources critiques

Scénario 3 : Comportement réseau atypique légitime

- **Alerte IDS** : "Anomalous Network Behavior Detected"
- **Réalité** : Sauvegarde automatisée ou transfert de données massif
- **Investigation** : Analyse complexe des logs et validation métier
- **Impact** : Consommation significative d'expertise senior

2.3.5 Impact quantifié sur les opérations SOC

Les faux positifs constituent aujourd'hui l'un des défis les plus coûteux et chronophages pour les SOC. Ils ont un impact direct sur la productivité des analystes, la qualité des décisions et la réactivité face aux menaces réelles.

Volume d'alertes quotidiennes

Selon le *Ponemon Institute* (2022), un centre de sécurité (SOC) reçoit en moyenne entre **10 000 et 150 000 alertes par jour**, dont plus de la moitié ne nécessitent aucune action. D'après une étude de *Gartner* (2020), **45 à 90 % des alertes IDS** seraient des faux positifs. Ce volume entraîne une surcharge massive des équipes, qui passent la majeure partie de leur temps à filtrer et vérifier manuellement les alertes.

Coût et impact humain

Une enquête menée par *Devo Technology* en 2021 révèle que les analystes de SOC passent en moyenne **27% de leur temps** à traiter des alertes non pertinentes. Près de **70%** d'entre eux déclarent ressentir une fatigue ou un stress lié à la surcharge d'alertes, un phénomène connu sous le nom d'*alert fatigue* [8]. Cette fatigue conduit souvent à des erreurs d'évaluation, voire à la désactivation temporaire de certaines alertes jugées trop bruyantes.

TABLE 2.4 – Impact des faux positifs sur les opérations SOC

Aspect affecté	**Conséquence observée**
Charge de travail	Hausse du volume d'alertes à traiter quotidiennement
Productivité des analystes	Temps perdu à valider des alertes non pertinentes ($\approx 25 - -30\%$)
Fatigue et stress	Baisse de concentration, démotivation, erreurs humaines
Coût opérationnel	Détournement de ressources humaines et financières
Efficacité du SOC	Allongement du temps moyen de détection (MTTD) et de réponse (MTTR)

2.3.6 Méthodes traditionnelles de mitigation

Face au volume croissant de faux positifs, les centres opérationnels de sécurité (SOC) ont mis en place différentes approches de mitigation pour réduire la charge d'alertes et

améliorer la pertinence des détections. Ces méthodes reposent sur des stratégies de filtrage, de corrélation et d'automatisation partielle des processus.

1. Ajustement des règles et signatures IDS

Les administrateurs affinent manuellement les règles de détection afin de réduire les déclenchements non pertinents. Cela consiste à :

- restreindre les règles à des adresses IP, ports ou sous-réseaux spécifiques ;
- désactiver certaines signatures trop sensibles ;
- ou modifier les paramètres de seuil (par exemple, nombre de paquets ou fréquence de détection).

Cette méthode peut temporairement diminuer le nombre de faux positifs, mais elle présente un risque de générer des **faux négatifs** si les règles deviennent trop restrictives.

2. Corrélation via les SIEM

Les solutions SIEM (Security Information and Event Management) sont utilisées pour agréger les alertes provenant de multiples sources et les corréler. Le principe est d'identifier les événements réellement liés à un incident en combinant plusieurs signaux (IDS, pare-feu, antivirus, logs systèmes). Par exemple :

- une alerte IDS isolée peut être ignorée si elle n'est pas corroborée par une anomalie sur un hôte ;
- à l'inverse, la corrélation de plusieurs événements similaires peut générer une alerte de priorité plus élevée.

Bien que cette approche améliore la pertinence, elle nécessite une configuration complexe et une maintenance constante.

3. Listes blanches et suppression d'alertes connues

Les SOC établissent des **whitelists** (listes blanches) contenant des adresses IP, domaines ou applications considérées comme sûres. Toute alerte associée à ces éléments est automatiquement ignorée ou marquée comme non critique. Cette méthode réduit considérablement le bruit d'alerte, mais demande un suivi rigoureux : une liste blanche mal gérée peut masquer de véritables menaces.

4. Automatisation via SOAR

Les plateformes SOAR (Security Orchestration, Automation and Response) permettent d'automatiser certaines réponses répétitives :

- fermeture automatique de tickets pour alertes bénignes récurrentes ;
- enrichissement automatique des alertes avec des données de réputation IP ;

— exécution de scripts pour isoler un hôte suspect.

L'automatisation permet un gain de temps notable, mais dépend fortement de la qualité des règles et scénarios configurés.

5. Classification manuelle et retour d'expérience

Enfin, les analystes expérimentés jouent un rôle essentiel dans la réduction des faux positifs. Ils reclassifient manuellement les alertes, ajustent les règles et partagent les retours d'expérience avec l'équipe d'ingénierie sécurité. Cependant, cette approche est lente, subjective et difficile à maintenir à grande échelle.

2.3.7 Limites des approches existantes

Malgré les efforts considérables déployés pour réduire les faux positifs, les approches traditionnelles demeurent limitées face à la complexité croissante des environnements numériques modernes. Ces limites se manifestent à la fois sur le plan technique, organisationnel et cognitif.

1. Forte dépendance à l'expertise humaine

Les systèmes de détection actuels reposent largement sur des règles et des ajustements manuels. Chaque modification nécessite l'intervention d'analystes expérimentés pour identifier les causes, ajuster les signatures et valider les résultats. Cette dépendance rend le processus :

— lent et peu évolutif,
— sensible aux erreurs humaines,
— et difficile à maintenir dans des environnements dynamiques.

2. Incapacité à interpréter le contexte des alertes

Les IDS et SIEM traditionnels se concentrent sur des motifs techniques (adresses IP, ports, chaînes de caractères), sans comprendre le contexte sémantique ou comportemental d'une alerte. Ainsi, une même requête peut être considérée comme suspecte ou bénigne selon la situation, mais le système ne fait pas cette distinction. Cette **absence d'intelligence contextuelle** conduit à une classification erronée de nombreux événements.

3. Mauvaise adaptation aux environnements dynamiques

Les infrastructures informatiques modernes évoluent en permanence (virtualisation, cloud hybride, microservices, IoT). Les règles statiques deviennent rapidement obsolètes face à ces changements. Les SOC doivent alors réajuster continuellement leurs signatures et leurs politiques de détection, ce qui augmente les coûts et la complexité.

4. Risque de déséquilibre entre détection et précision

En tentant de réduire les faux positifs, les équipes de sécurité prennent souvent le risque inverse : augmenter les **faux négatifs**. Une règle trop restrictive peut ignorer des signaux faibles annonçant une attaque réelle. Ainsi, la recherche d'un équilibre entre *sensitivity* (détection) et *specificity* (pertinence) reste un défi majeur.

5. Manque de capacités d'apprentissage et d'adaptation

Les systèmes traditionnels ne tirent que très peu parti du retour d'expérience des analystes. Chaque erreur est traitée ponctuellement, sans apprentissage global du système. Cette absence de boucle d'amélioration continue empêche toute optimisation durable de la détection.

2.4 Revue des approches de réduction des faux positifs (FPR)

Nous avons trouvé dans la littérature de nombreuses études récentes sur la reduction d'intrusion des FP. De nombreux travaux proposent des solutions fondées sur l'Intelligence Artificielle (IA) et l'apprentissage machine, allant de l'optimisation de caractéristiques à l'usage de modèles profonds et de modèles de langage avancés.Dans cette section nous explorerons quelques travaux récents.

2.4.1 Sélection de caractéristiques et optimisation métaheuristique

L'amélioration de la représentation des données est un levier central pour diminuer les faux positifs. La suppression des variables redondantes et la sélection de sous-ensembles de caractéristiques pertinentes permettent d'améliorer la précision et la généralisation des classifieurs.[14] Les méthodes métaheuristiques telles que *Particle Swarm Optimization* (PSO), *Cuckoo Search* ou leurs combinaisons (CS-PSO) sont couramment appliquées sur des datasets comme CSE-CIC-IDS2018 et NSL-KDD, montrant des gains substantiels en précision et une réduction du nombre d'alertes erronées.[3, 28]

Par exemple, Shukla et al. utilisent PSO pour sélectionner les caractéristiques les plus discriminantes sur NSL-KDD et montrent une diminution notable du FPR par rapport à des approches sans sélection de caractéristiques.[28] De leur côté, Abualigah et al. proposent une optimisation des caractéristiques basée sur l'algorithme Aquila modifié, couplé à un classificateur SVM, obtenant un meilleur compromis entre complexité et précision.[1] Ces approches restent toutefois sensibles au jeu de données utilisé et peuvent perdre du contexte séquentiel ou temporel lorsque la sélection de caractéristiques est trop agressive.

2.4.2 Modèles d'ensemble, systèmes hybrides et filtration contextuelle

Les architectures d'ensemble (Random Forest, XGBoost, stacking) et les systèmes hybrides combinant plusieurs paradigmes constituent une autre voie importante pour réduire les faux positifs. Elles exploitent la complémentarité de modèles différents pour améliorer la robustesse.[19, 6]

Des approches hybrides combinent par exemple des extracteurs de caractéristiques profonds (CNN, LSTM) avec des classifieurs classiques, ou encore un classificateur d'apprentissage profond, un module de *clustering* probabiliste et l'expertise humaine.[17] Maiga et al. montrent qu'une telle synergie permet de filtrer un grand nombre de fausses alarmes, mais la méthode reste dépendante de la disponibilité et de la cohérence de l'apport des analystes.[17] Dans le contexte des systèmes de contrôle industriels (CBTC), Binyu Yin et al. proposent une méthode hybride basée sur un algorithme d'ensemble de type *Improved Stacking* couplé à une stratégie d'élimination des faux positifs (FPE Strategy), illustrant l'intérêt de couches de filtrage spécifiques au domaine.[36]

D'autres travaux explorent des stratégies de filtration contextuelle à l'aide de graphes d'alertes, comme chez Tian Wang et al., qui exploitent la relation entre événements pour distinguer les véritables incidents des fausses alarmes.[33] Toutefois, la construction et la maintenance de ces graphes deviennent rapidement coûteuses à grande échelle. Parallèlement, les systèmes IDS basés sur les signatures, illustrés par le *Snort IDS and IPS Toolkit* de Beale,[5] et leur intégration dans des architectures hybrides décrites par Moorthy Agoramoorthy et al.,[20] rappellent que, sans mécanismes complémentaires, ces approches restent limitées face aux attaques de jour zéro et à la variabilité des environnements.

2.4.3 Apports du deep learning : autoencodeurs, CNN, LSTM et architectures avancées

Les méthodes de *deep learning* jouent un rôle croissant dans la réduction des faux positifs. Les autoencodeurs (AE) et leurs variantes (VAE, stacked AE) sont largement utilisés pour la détection d'anomalies non supervisée : ils apprennent le comportement normal du trafic et signalent les reconstructions anormales, ce qui permet de filtrer une partie des alertes inutiles.[27, 6] Des architectures plus spécialisées, comme les *Separable Convolution AutoEncoders* combinés à des réseaux LSTM, ont également été proposées pour capturer simultanément les motifs spatiaux et temporels du trafic réseau.[37]

La transformation du trafic réseau en représentations de type « image » rend possible l'usage de CNN, tandis que les modèles séquentiels (LSTM/GRU) apportent le contexte temporel nécessaire pour distinguer les variations légitimes des comportements véritablement

anormaux.[19, 22] Des analyses comparatives, comme celle de Das et al., montrent que ces architectures profondes réduisent le FPR par rapport à des modèles plus simples, au prix d'une complexité accrue et d'un besoin important en données étiquetées.[6] Les revues de Jyothsna et de Hassein rappellent toutefois que les systèmes basés sur les anomalies conservent une tendance intrinsèque à produire un FPR élevé si le modèle normal est imparfaitement appris.[14, 12]

2.4.4 Pipelines en deux étapes (two-stage detection)

Une approche opérationnelle répandue pour concilier haut rappel et faible FPR consiste à décomposer la détection en deux étapes.[4] Un premier détecteur, rapide et large, vise à capturer la quasi-totalité des événements potentiellement suspects, quitte à produire un nombre élevé de faux positifs. Un second module, plus coûteux mais plus précis (par exemple un autoencodeur, un modèle d'ensemble ou un classificateur avancé), réévalue ces candidats pour filtrer les alertes non pertinentes.[4, 3]

Ce schéma *two-stage* est particulièrement adapté aux environnements SOC, où il est préférable de rejeter les faux positifs à un stade de post-traitement plutôt que de risquer de manquer des attaques critiques. Il préfigure également les architectures où un LLM intervient en seconde étape pour requalifier et expliquer les alertes générées par un premier modèle.

2.4.5 Contributions des LLMs et analyse sémantique des alertes

L'utilisation des *Large Language Models* (LLMs) pour la cyberdéfense est une direction émergente, qui vise à introduire une couche d'analyse sémantique et contextuelle au-dessus des signaux techniques.[16, 31] Hassanin propose une vue d'ensemble complète des LLMs pour la cyberdéfense, mettant en avant leur potentiel pour l'analyse de journaux, la corrélation d'événements et la réduction des faux positifs grâce à une meilleure compréhension du contexte.[11]

Nguyen et al. explorent l'usage de BERT pour la réduction des fausses alarmes dans l'analyse statique des vulnérabilités, montrant qu'un modèle de type Transformer peut distinguer plus finement les cas réellement exploitables des cas bénins.[21] Toutefois, ces modèles sont intensifs en ressources de calcul et nécessitent une préparation spécifique des données de sécurité pour être exploitables efficacement. De même, Pitre décrit un système de détection d'intrusion orienté attaques de jour zéro, visant explicitement à réduire le FPR tout en améliorant la détection de menaces inédites.[23]

Plus largement, plusieurs travaux récents combinent des détecteurs classiques avec des LLMs utilisés en post-traitement pour requalifier, enrichir ou expliquer les alertes, dans

une logique proche de l'« analyse assistée par IA » dans le SOC.[16, 29, 34] Ces approches restent encore exploratoires, mais elles ouvrent la voie à des systèmes capables de prendre en compte non seulement les caractéristiques bas niveau des flux ou journaux, mais aussi leur signification fonctionnelle et métier.

Synthèse

En résumé, la littérature propose un large éventail de stratégies pour réduire les faux positifs dans les IDS : optimisation de caractéristiques, modèles d'ensemble et hybrides, architectures profondes, pipelines en deux étapes, et, plus récemment, intégration de Transformers et de LLMs pour une analyse contextuelle avancée.[3, 19, 27, 4, 16, 11] Ces travaux convergent vers l'idée qu'aucune approche unique n'est suffisante : la combinaison de détecteurs efficaces et de couches d'analyse contextuelle, potentiellement basées sur des LLMs, constitue aujourd'hui une piste privilégiée pour réduire le FPR tout en préservant un haut niveau de détection dans les environnements SOC.

Conclusion

Dans ce chapitre, ont été présentés les fondements des systèmes de détection d'intrusion (IDS), leurs typologies principales et plusieurs outils emblématiques utilisés dans les environnements de sécurité modernes, tels que Snort et Suricata. Malgré leur rôle central, les IDS génèrent un nombre significatif de **faux positifs**, provoquant surcharge, perte de vigilance et coûts opérationnels accrus pour les équipes SOC. Les approches traditionnelles, basées sur des signatures ou des règles statiques, montrent leurs limites face à la dynamique des environnements actuels.

La littérature propose des pistes prometteuses pour réduire les faux positifs, notamment la sélection et l'optimisation de caractéristiques, les modèles d'ensemble et architectures hybrides, les architectures profondes (autoencodeurs, CNN, LSTM), les pipelines *two-stage*, ainsi que l'usage émergent des Transformers et LLMs pour l'analyse contextuelle des alertes. Le chapitre suivant explorera plus en détail la **stratégie de réduction des fausses alertes des IDS par les LLMs**, en proposant une architecture hybride où un LLM intervient en post-traitement des alertes afin de requalifier, filtrer et expliquer les événements, dans le but de soutenir les analystes SOC dans leurs prises de décision.

Chapitre 3

Strategie de reduction des fausses alertes des IDS par les LLMs

Introduction

Nous avons vu que, malgré leurs avancées, les IDS actuels détectent efficacement une grande partie des attaques connues, mais restent confrontés à plusieurs limites majeures. Les principaux défis concernent la capacité à identifier les attaques inconnues ou émergentes, tout en réduisant le nombre élevé de fausses alertes qui compliquent le travail des analystes et réduisent la réactivité des SOC. À l'ère du big data, où des volumes massifs et variés de données traversent les réseaux, les IDS doivent également être capables de traiter rapidement ces informations afin de repérer les anomalies en temps réel. Des solutions comme les SIEM ou la Threat Intelligence ont été adoptées pour renforcer la détection, mais elles présentent encore des contraintes importantes en termes de coût, d'expertise et de complexité opérationnelle. En pratique, malgré leur utilité dans les environnements de cybersécurité, les IDS génèrent trop de fausses alertes et échouent parfois à identifier certaines attaques sophistiquées. Dans un contexte où le paysage des cybermenaces ne cesse de s'étendre et où les attaques deviennent plus intelligentes, rapides et furtives, cette situation crée une surcharge informationnelle qui ralentit et fragilise l'efficacité des opérations de sécurité. Pour répondre à ces enjeux, la recherche s'oriente vers l'utilisation de techniques d'intelligence artificielle avancées, où le Deep Learning et les modèles de langage à grande échelle (LLMs) s'avèrent particulièrement prometteurs. Ces approches permettent de :

- gérer de grandes quantités de données et leur haute dimensionnalité ;
- apprendre automatiquement des caractéristiques pertinentes du trafic et leurs corrélations ;
- prédire les anomalies et générer des rapports de détection détaillés.

Dans notre étude, nous proposons approche hybride combinant XGBoost pour la classification du trafic réseau et l'estimation de la confiance des alertes et les Large Language Models (LLMs) les Large Language Models (LLMs) pour la réduction des faux positifs. Nous utilisons comme données d'expériences l'ensemble de données **CSE-CIC-IDS2018**, qui est très proche des données de réseaux actuels et contient une énorme quantité d'échantillons (16 232 943, dont 82 % de trafic normal et 14 catégories d'attaques récentes). Étant donné la taille massive du dataset, nous n'avons pas travaillé avec l'intégralité des données, mais avons appliqué une stratification pour créer un sous-ensemble représentatif, garantissant que la proportion de chaque type de trafic et de catégorie d'attaque soit conservée. Cette approche nous permet de réaliser des expériences efficaces tout en maintenant la représentativité statistique des données. Le reste de ce chapitre est structuré comme suit : dans la deuxième partie, nous alons explorer les LLMs. La troisième partie présente nos méthodes et modèles appliqués. La quatrième partie décrit l'analyse exploratoire des données, le prétraitement effectué et les résultats expérimentaux et discterons des résultats, et enfin conclut la partie pratique.

3.1 Les modèles de langage à grande échelle (LLMs)

Un *Large Language Model* (LLM) est un programme d'intelligence artificielle capable de comprendre et de générer du texte, comme le ferait un humain, grâce à son apprentissage sur des milliards de phrases venant d'Internet, de livres ou d'articles. Il utilise une architecture spéciale (Transformer) pour analyser le contexte et produire des textes qui ont du sens, répondre à des questions, traduire ou résumer des informations. Les LLMs sont devenus un outil incontournable dans de nombreux domaines, notamment la cybersécurité. Leur capacité à analyser et interpréter des données textuelles complexes, comme les journaux d'alertes, les messages issus des IDS ou les descriptions d'incidents, ouvre la voie à de nouvelles solutions pour la réduction des faux positifs et l'assistance aux analystes des SOC.

Avant de présenter notre approche pratique, il est essentiel de comprendre leur fonctionnement interne ainsi que les raisons pour lesquelles ils constituent aujourd'hui une piste particulièrement prometteuse pour améliorer la précision des systèmes de détection d'intrusion.

3.1.1 Fonctionnement interne des LLMs

Les LLMs reposent principalement sur l'architecture *Transformer*, introduite par Vaswani et al. (2017)[32]. Un LLM s'article autour des étapes suivantes : clés :

3.1.2 Tokenisation

Le texte brut est découpé en unités appelées tokens : ce peut être des mots, des syllabes ou des morceaux de mots.

Représentation vectorielle (Embeddings)

Chaque mot ou token est converti en un vecteur dense dans un espace de grande dimension. Ces vecteurs capturent la sémantique, les relations syntaxiques et contextuelles. Les embeddings permettent au modèle de manipuler du texte sous forme numérique.

3.1.3 Encodage positionnel

Le modèle ajoute des informations sur la position du token dans la séquence : un même mot n'a pas le même sens selon l'ordre.

3.1.4 Blocs Transformer

Le coeur du modèle : chaque token traverseplusieurs couches identiques de :

- **Mécanisme d'attention** (self-attention) : chaque mot "regarde" tous les autres, pour pondérer l'importance relative selon le contexte.
- **FFN (Feedforward Neural Network)** : traitement local, affine chaque représentation de token.

3.1.5 Décodage et sortie

En génération, le modèle prédit le prochain token, construit la séquence de sortie mot à mot (ou token à token).

Grâce à ces mécanismes, les LLMs peuvent produire des réponses cohérentes, détecter des anomalies, classer des informations contextuelles et raisonner sur des données non structurées, ce qui les rend particulièrement adaptés aux besoins des SOC. En résumé un

LLM fonctionne comme un super assistant qui :

- Lit et apprend énormément de textes pour connaître les mots et leur utilisation.
- Transforme chaque mot en chiffres pour mieux les analyser et comparer.
- Regarde toute la phrase d'un coup afin de comprendre le contexte et l'attention.
- Prédit ou écrit le mot suivant en s'appuyant sur tout ce qu'il a appris.

3.1.6 Exemples de modèles de LLM

Les modèles de langage de grande taille (Large Language Models, LLM) sont aujourd'hui nombreux et variés, chacun ayant ses caractéristiques et domaines d'application spécifiques. Parmi les plus connus, on peut citer :

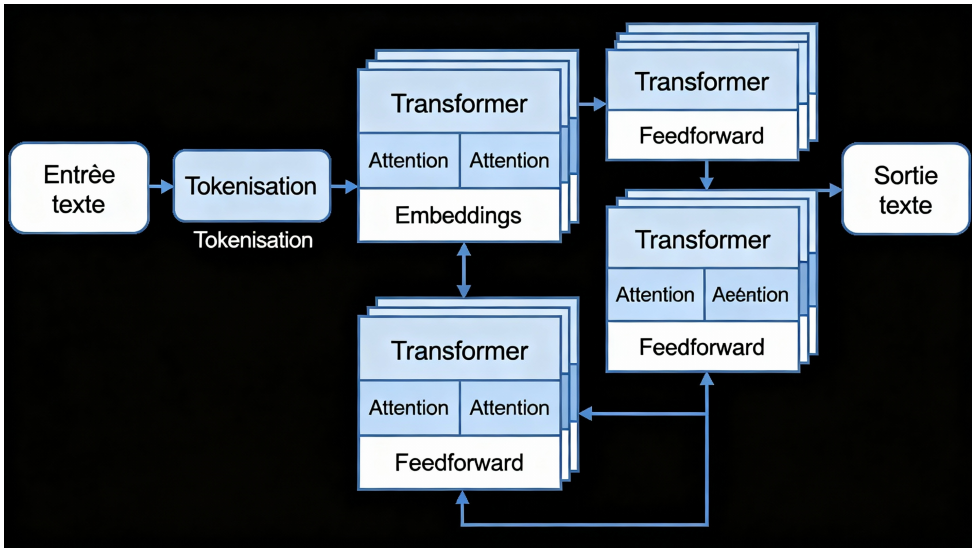


FIGURE 3.1 – Architecture de fonctionnement d'un LLM

— **GPT (Generative Pre-trained Transformer)** : Développé par OpenAI, GPT est un modèle de type transformer pré-entraîné sur de larges corpus textuels pour générer du texte, répondre à des questions ou résumer des documents. Les versions récentes, comme GPT-4, sont capables de comprendre et de produire du texte dans de multiples langues.
— **BERT (Bidirectional Encoder Representations from Transformers)** : Conçu par Google, BERT est un modèle d'encodage bidirectionnel qui excelle dans la compréhension du langage et les tâches de classification, de question-réponse ou de reconnaissance d'entités nommées.
— **T5 (Text-to-Text Transfer Transformer)** : Également développé par Google, T5 convertit toutes les tâches de traitement du langage en problèmes de génération de texte, permettant ainsi une grande flexibilité pour différentes applications.
— **LLaMA (Large Language Model Meta AI)** : Créé par Meta AI, LLaMA est optimisé pour les performances sur des modèles plus petits tout en conservant une capacité élevée de traitement du langage naturel.
— **Falcon et Mistral** : Ce sont des modèles récents open-source qui visent à fournir des alternatives performantes et accessibles pour la génération et la compréhension de texte.

Ces modèles sont utilisés dans divers domaines, tels que la détection d'intrusions, la génération de texte, la traduction automatique ou encore l'assistance intelligente. Dans le cadre de ce mémoire, l'étude des LLM vise à explorer leur potentiel dans la réduction des fausses alertes dans les systèmes de détection d'intrusions (IDS).

3.1.7 Pourquoi les LLMs constituent une solution prometteuse pour les IDS

Les chapitres précédents ont montré que les systèmes de détection d'intrusion (IDS), malgré leurs performances, continuent de générer un volume important de fausses alertes. Ces faux positifs saturent les SOC et affectent la réactivité des analystes. Les méthodes traditionnelles basées sur les signatures, sur les anomalies ou sur des modèles d'IA classiques atteignent aujourd'hui leurs limites face à la complexité croissante du trafic réseau et à l'évolution rapide des attaques.

Les LLMs représentent une alternative particulièrement intéressante pour relever ces défis. Leur capacité à analyser du texte brut, à traiter des journaux réseau, à comprendre des règles IDS ou à contextualiser des événements permet :

— de classifier automatiquement les alertes ;
— d'identifier les faux positifs ;
— d'expliquer les décisions (rôle crucial pour les SOC) ;
— de réduire la charge opérationnelle des analystes ;
— d'améliorer la précision globale des systèmes de détection.

Ainsi, l'intégration d'un LLM au sein d'un pipeline de traitement d'alertes offre une approche innovante pour améliorer la qualité et la pertinence des alertes générées par les IDS. La section suivante présentera en détail notre méthodologie, le pipeline conçu, ainsi que les modèles et techniques utilisés dans notre étude.

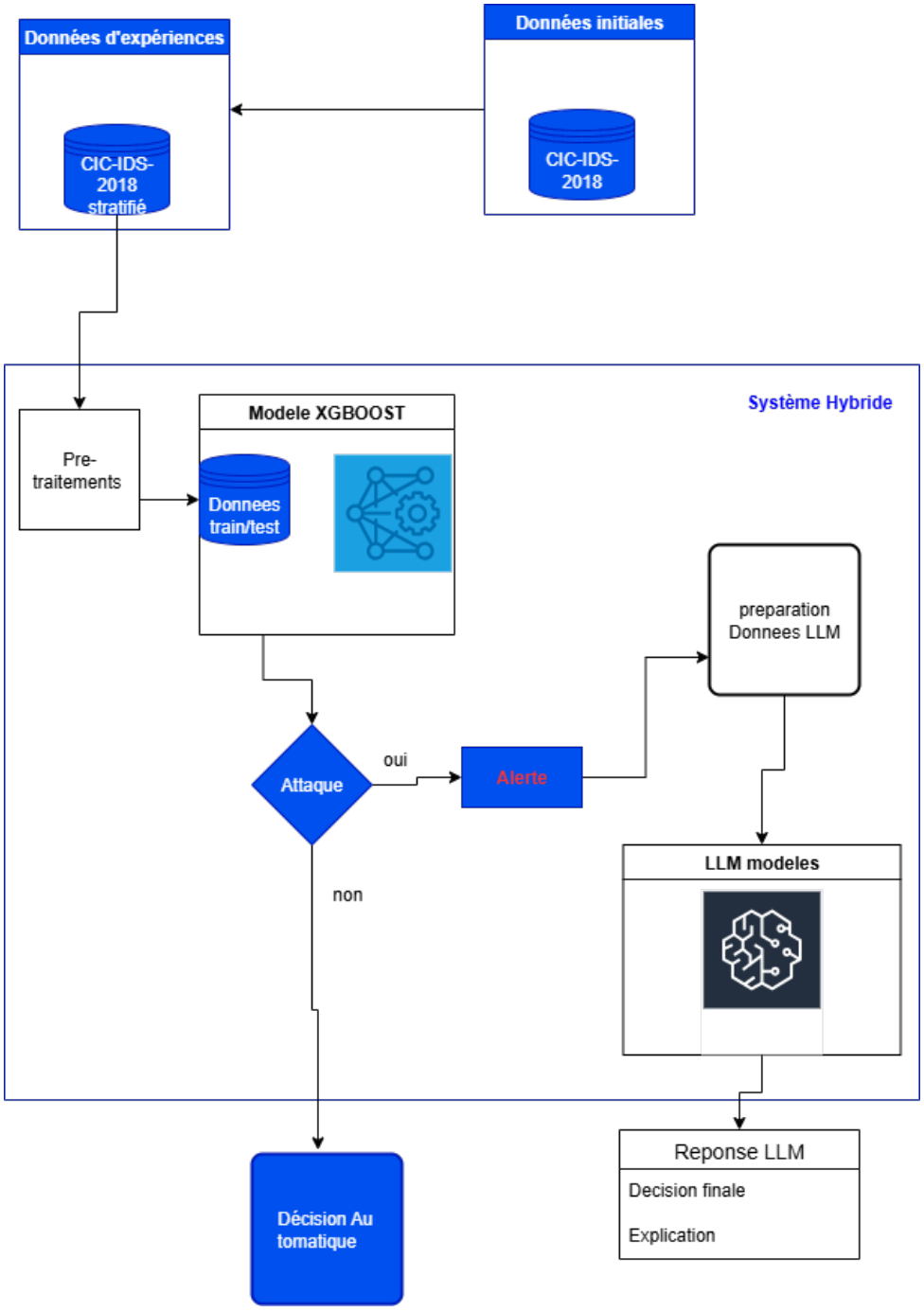


FIGURE 3.2 – Fonctionnement du processus d'expérience

3.2 Approches méthodologiques

Dans cette section, nous présentons notre approche méthodologique adoptée pour cette étude. Nous décrivons les différentes étapes du pré-traitement, de l'analyse des données issues du dataset CSE-CIC-IDS2018, ainsi que la modélisation de la solution proposée. Notre approche repose sur une combinaison de modèles classiques d'apprentissage automatique (notamment XGBoost) et de modèles de langage de grande taille (Large Language Models,

LLMs). L'objectif est de concevoir un système robuste capable de réduire efficacement les fausses alertes générées par les systèmes de détection d'intrusions, tout en améliorant la fiabilité et la précision globale du processus de détection.

3.2.1 Environnement Expérimental

Notre choix de langage de programmation s'est porté sur Python. En effet, celui-ci offre une large gamme de bibliothèques adaptées à l'analyse de données en cybersécurité, permettant par exemple de manipuler des datasets volumineux (Pandas, NumPy), d'entraîner des modèles de machine learning performants (Scikit-learn, XGBoost), de visualiser les résultats (Matplotlib, Seaborn), et d'intégrer des analyses contextuelles avancées via des LLMs (API Groq). Principalement, nous avons utilisé les bibliothèques Python suivantes :

- **Pandas** :Conçue pour la manipulation et l'analyse de données, Pandas fournit des structures de données puissantes comme les DataFrame, idéales pour travailler avec les flux réseau tabulaires du CSE-CIC-IDS2018.
- **NumPy** : Bibliothèque fondamentale pour le calcul scientifique, elle offre des tableaux multidimensionnels et des fonctions mathématiques avancées, essentielles pour le préprocessing des caractéristiques réseau.
- **Scikit-learn** : Dédiée à l'apprentissage automatique, elle propose des outils pour la classification, la régression, le clustering et l'évaluation des modèles d'intrusion detection etc.
- **XGBoost** : Implémente des modèles de gradient boosting optimisés, reconnus pour leur haute performance sur des datasets d'anomalies comme CSE-CIC-IDS2018 en détection d'attaques.
- **Matplotlib et Seaborn** : Utilisées pour la création de graphiques et de visualisations, ces bibliothèques offrent des outils puissants pour représenter graphiquement les données.
- Utilisées pour la création de graphiques et visualisations, ces bibliothèques offrent des outils puissants pour représenter les distributions d'attaques et les métriques de performance. Pour l'analyse contextuelle avancée des alertes, nous avons intégré l'API Groq avec les modèles LLM "llama-3.3-70b-versatile" et "llama-3.1-8b-instant". Ces modèles permettent une explication sémantique des prédictions, capturant les nuances des attaques réseau grâce à leur compréhension contextuelle approfondie. Pour le Plateforme de Développemen, nous avons utilisé **Google Colaboratory** qui est un produit de Google Research qui permet d'écrire et d'exécuter du code Python directement dans le navigateur sous forme de notebooks interactifs, similaires aux notebooks Jupyter. Cet environnement est particulièrement adapté au développement en machine learning et en analyse de données, car il offre un espace de travail prêt à l'emploi, sans installation locale, et intègre des fonctionnalités de

collaboration et de partage en ligne.

Un atout majeur de Google Colab est l'accès gratuit à des ressources matérielles avancées, notamment des unités de traitement graphique (GPU) et parfois des TPU, ce qui permet d'accélérer l'entraînement de modèles d'apprentissage automatique et de traiter efficacement des datasets volumineux comme CSE-CIC-IDS2018. L'utilisateur peut facilement activer le GPU via les paramètres du runtime, ce qui réduit considérablement le temps de calcul pour les expériences de deep learning et de machine learning intensif.

Colab inclut également des cellules de texte pour la documentation, permettant de combiner code, explications et résultats dans un même document, à la manière d'un notebook Jupyter classique. De plus, la plateforme est livrée avec de nombreuses bibliothèques Python préinstallées, telles que NumPy, Pandas, Scikit-learn, Matplotlib et Seaborn, ce qui simplifie la mise en place de projets de data science et de machine learning sans étape de configuration supplémentaire. Ces caractéristiques justifient pleinement le choix de Google Colab comme environnement de développement pour ce travail. Notre stockage des données a été assuré via Google Drive, permettant un accès centralisé, une sauvegarde sécurisée et une intégration directe avec Google Colab pour faciliter le chargement et la manipulation des fichiers du dataset.



FIGURE 3.3 – google Colaboratory

3.2.2 Présentation du Jeu de Données CSE-CIC-IDS2018 et Analyse Exploratoire (EDA)

Nous avons choisi le dataset CSE-CIC-IDS2018 car il s'agit d'un ensemble de données complet, moderne et largement utilisé pour l'évaluation des systèmes de détection d'intrusions réseau (NIDS). Développé par le Canadian Institute for Cybersecurity (CIC), ce dataset simule un environnement réseau réaliste intégrant à la fois du trafic normal et

divers types d'attaques cybernétiques. Il contient un total de 84 caractéristiques, dont 80 sont communes, extraites des flux réseau (« flows »). Ce dataset est particulièrement adapté aux tâches de classification binaire et multi-classes en apprentissage automatique.

Le CSE-CIC-IDS2018 a été conçu pour répondre aux défis croissants liés à la cybersécurité dans des environnements à grande échelle tels que l'Internet des Objets (IoT), les réseaux 5G ou le cloud computing. Il capture des scénarios variés et réalistes incluant des attaques modernes telles que :

- attaques par force brute ;
- exploitation Heartbleed ;
- botnets ;
- attaques par déni de service (DoS) ;
- attaques par déni de service distribué (DDoS) ;
- attaques web ;
- infiltration de réseau interne.

Contrairement à des datasets plus anciens comme KDD'99 ou NSL-KDD, souvent critiqués pour leurs problèmes de redondance et de déséquilibre de classes, le CSE-CIC-IDS2018 propose une représentation beaucoup plus fidèle des environnements réseau contemporains, avec des données brutes et des flux prétraités.

Le dataset est structuré en dix (10) fichiers CSV correspondant à dix jours différent de capture de trafic. Il inclut du trafic bénin et malveillant, pour un total de 16 232 943 échantillons. Son échelle permet d'effectuer des évaluations robustes, mais nécessite également une gestion prudente du volume de données lors de l'expérimentation.

L'infrastructure simulée dans le dataset comprend :

- 50 machines attaquantes ;
- un réseau cible de 420 machines ;
- 30 serveurs répartis dans 5 départements.

Un tableau synthétique présentant l'ensemble du dataset est fourni ci-après.

TABLE 3.1 – Distribution des échantillons dans le dataset CSE-CIC-IDS2018

Type d'attaque	**Échantillons**	**Taux (%)**
Normal	13 484 708	83.07
DDoS attack-HOIC	686 012	4.23
DDoS attack-LOIC-HTTP	576 191	3.55
DDoS attack-LOIC-UDP	1 730	0.01
DoS attacks-Hulk	461 912	2.84
DoS attacks-SlowHTTPTest	139 890	0.86
DoS attacks-GoldenEye	41 508	0.26
DoS attacks-Slowloris	10 990	0.07
Botnet	286 191	1.76
FTP-BruteForce	193 360	1.19
SSH-BruteForce	187 589	1.16
Infiltration	161 934	0.99
Brute Force-Web	611	0.004
Brute Force-XSS	230	0.001
SQL Injection	87	0.0005
Total	**16 232 943**	**100**

Stratégie d'Échantillonnage

Face au volume très important des données ($\sim$ 16 millions d'échantillons), une stratégie d'échantillonnage a été nécessaire afin de rendre l'expérimentation réalisable tout en préservant la représentativité du dataset.

Une approche d'échantillonnage stratifié a été mise en œuvre selon les principes suivants :

— **Fichier très volumineux (Tuesday-20-02-2018)** : échantillonnage direct à 100 000 échantillons ;
— **Fichiers de taille moyenne** : échantillonnage stratifié pour obtenir 80 000 échantillons par fichier ;
— **Fichiers de petite taille** : conservation intégrale.

Résultats de l'Échantillonnage

Le dataset combiné résultant de ce processus présente les caractéristiques suivantes :

— **Taille finale** : $\sim$ 800 000 échantillons (environ 5% du dataset original) ;
— **Répartition** : maintien des proportions entre les différentes classes d'attaques ;
— **Diversité** : conservation de l'ensemble des types d'attaques présents ;

— **Qualité des données** : suppression des lignes présentant des erreurs d'étiquetage et absence de valeurs manquantes.

Le tableau ci-dessous présente la distribution des échantillons du dataset combiné combiné après corrections, regroupés par catégories d'attaques. Cette représentation colorée permet de visualiser clairement la répartition des différentes classes ainsi que leur proportion relative dans l'ensemble des données.

TABLE 3.2 – Distribution du dataset combiné (820 000 échantillons) par catégories d'attaques

Catégorie / Label	**Échantillons**	**Pourcentage (%)**
Traffic normal		
Benign	535 249	65.27
Attaques DDoS / DoS		
DDoS attacks-LOIC-HTTP	99 918	12.19
DDOS attack-HOIC	52 339	6.38
DoS attacks-Hulk	35 241	4.30
DoS attacks-SlowHTTPTest	10 673	1.30
DoS attacks-GoldenEye	3 167	0.39
DoS attacks-Slowloris	838	0.10
DDOS attack-LOIC-UDP	132	0.02
Attaques Brute Force		
FTP-BruteForce	14 752	1.80
SSH-Bruteforce	14 312	1.75
Brute Force -Web	47	0.01
Brute Force -XSS	17	0.00
SQL Injection	7	0.00
Autres attaques		
Bot	21 835	2.66
Infilteration	31 473	3.84
Total	**820 000**	**100**

La Figure 3.4 illustre la distribution des classes dans le dataset échantillonné.

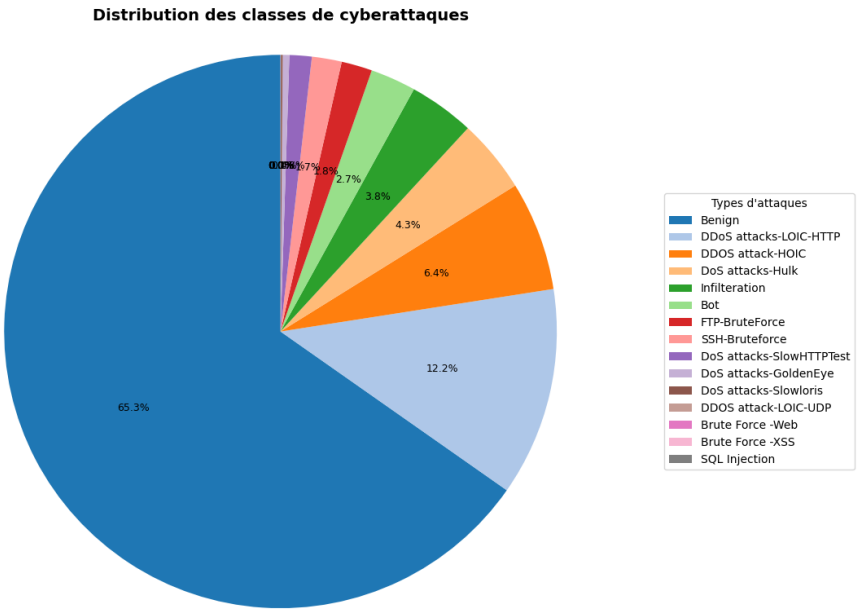


FIGURE 3.4 – Distribution du trafic dans CSE-CIC-IDS2018 stratifié

Ce processus d'échantillonnage nous a permis de :

- **Représentativité** :Tous les types d'attaques du dataset original sont conservés ;
- **Diversité** : Couverture complète des vecteurs d'attaque modernes ;
- **Faisabilité** : préparer un dataset exploitable pour les phases de pré-traitement, d'entraînement et d'évaluation des modèles.
- **Proportion réaliste** : Ratio trafic bénin/malveillant reflétant des scénarios réels.

3.2.3 Prétraitement des données et *Feature Engineering*

Le processus de prétraitement a suivi une méthodologie rigoureuse pour transformer les données brutes en un format adapté à l'apprentissage automatique.

Étape 1 : Analyse et suppression des colonnes non informatives

Après la construction du dataset échantillonné de 820 000 observations et 84 variables explicatives, une analyse des valeurs manquantes a été réalisée. Les colonnes suivantes présentaient plus de 50 % de valeurs manquantes :

- *Flow ID* : 87,80 % ;
- *Src IP* : 87,80 % ;
- *Src Port* : 87,80 % ;
- *Dst IP* : 87,80 %.

Ces variables, principalement des identifiants réseau, sont peu informatives pour la classification et fortement incomplètes. Les décisions suivantes ont été prises :

— suppression des 4 colonnes réseau présentant plus de 87 % de valeurs manquantes ;
— suppression de la colonne *Timestamp* (type `object`), jugée non pertinente pour la classification ;
— conservation de 79 colonnes sur les 84 initiales.

Le Tableau 3.3 résume l'impact du nettoyage initial sur la dimensionnalité du dataset.

TABLE 3.3 – Impact du nettoyage initial sur la dimensionnalité

Étape	**Dimensions**	**Colonnes supprimées**	**Raison**
Initial	(820 000, 84)	-	Dataset brut
Après suppression	(820 000, 80)	4	Valeurs manquantes > 50 %
Final	(820 000, 79)	1	Colonne temporelle

Gestion des valeurs manquantes résiduelles

Après la suppression des colonnes problématiques, une analyse fine des valeurs manquantes résiduelles a mis en évidence une seule variable concernée :

— *Flow Byts/s* avec 2 317 valeurs manquantes, soit 0,28 % des observations.

Afin de préserver la distribution des données et d'éviter la suppression de lignes, une imputation par la médiane a été appliquée :

```
from sklearn.impute import SimpleImputer
imputer = SimpleImputer(strategy='median')
df_cleaned[['Flow Byts/s']] = imputer.fit_transform(df_cleaned[['Flow Byts/s']])
```

Cette stratégie est conforme aux bonnes pratiques de traitement des valeurs manquantes pour des variables continues en contexte d'apprentissage automatique.

Encodage de la variable cible et analyse de distribution

La variable cible *Label* a été encodée à l'aide de `LabelEncoder`, tout en conservant la distribution déséquilibrée caractéristique des données de cybersécurité. Le Tableau 3.4 présente la distribution détaillée des classes après encodage.

Cette distribution met en évidence un fort déséquilibre entre le trafic bénin et certaines attaques rares, ce qui est typique des datasets d'intrusion.

Conversion et validation des types de données

Pour garantir la cohérence des entrées, une conversion systématique en types numériques a été appliquée à l'ensemble des features :

```
X_numeric = X.apply(pd.to_numeric, errors='coerce')
```

TABLE 3.4 – Distribution détaillée des classes après encodage

Classe	Label original	Encodage	Échantillons	Pourcentage
0	Benign	0	535 249	65,27 %
6	DDoS attacks-LOIC-HTTP	6	99 918	12,19 %
4	DDOS attack-HOIC	4	52 339	6,38 %
8	DoS attacks-Hulk	8	35 241	4,30 %
12	Infilteration	12	31 473	3,84 %
1	Bot	1	21 835	2,66 %
11	FTP-BruteForce	11	14 752	1,80 %
14	SSH-Bruteforce	14	14 312	1,75 %
9	DoS attacks-SlowHTTPTest	9	10 673	1,30 %
7	DoS attacks-GoldenEye	7	3 167	0,39 %
10	DoS attacks-Slowloris	10	838	0,10 %
5	DDOS attack-LOIC-UDP	5	132	0,02 %
2	Brute Force -Web	2	47	0,01 %
3	Brute Force -XSS	3	17	0,00 %
13	SQL Injection	13	7	0,00 %

Les résultats de la conversion sont les suivants :

— 41 colonnes en `int64` (features discrètes) ;
— 37 colonnes en `float64` (features continues) ;
— 2 317 valeurs devenues `NaN` (0,28 %), correspondant aux valeurs déjà identifiées et imputées pour *Flow Byts/s*.

Cette étape assure une compatibilité totale avec les algorithmes de *scikit-learn* et XGBoost.

Gestion des valeurs aberrantes et processus d'imputation

Le nettoyage final a permis d'éliminer toutes les valeurs problématiques :

— valeurs infinies : aucune détectée après conversion ;
— valeurs `NaN` totales : 7 312 après conversion ;
— stratégie : imputation par la médiane pour l'ensemble des variables numériques.

Cette approche permet de limiter l'impact des valeurs extrêmes tout en conservant la structure statistique du dataset.

Division stratifiée des données

La séparation entraînement/test a été effectuée de manière stratifiée afin de préserver la distribution des classes, y compris les minoritaires :

```
X_train, X_test, y_train, y_test = train_test_split(
    X_imputed, y_encoded,
```

```
    test_size=0.2,
    random_state=42,
    stratify=y_encoded
)
```

Caractéristiques des jeux obtenus :

- **Train set** : 656 000 échantillons × 78 features (soit 80,% des données) ;
- **Test set** : 164 000 échantillons × 78 features(soit 20 ,% des données) ;
- **Distribution préservée** : environ 65,27 % de trafic bénin dans les deux jeux, ainsi que la même proportion relative pour les classes d'attaques.

Cette division stratifiée est recommandée pour les problèmes de détection d'intrusion afin d'assurer une évaluation robuste.

Normalisation standard des features

Afin d'optimiser les performances des algorithmes de machine learning, une standardisation des variables a été appliquée à l'aide de `StandardScaler` :

```
scaler = StandardScaler()
X_train_scaled = scaler.fit_transform(X_train)
X_test_scaled  = scaler.transform(X_test)
```

La validation statistique post-normalisation a donné :

- moyenne (train) : 0,000000 (données parfaitement centrées) ;
- écart-type (train) : 0,947331 (proche de l'idéal 1,0) ;
- moyenne (test) : -0,000492 (excellente généralisation) ;
- écart-type (test) : 0,855141 (légère sous-dispersion acceptable).

Ces résultats confirment une standardisation conforme aux bonnes pratiques de prétraitement.

Bilan du pipeline de prétraitement

Le pipeline de prétraitement a abouti à un dataset de haute qualité, présentant les caractéristiques suivantes :

- **Intégrité totale** : aucune valeur manquante ou infinie ;
- **Cohérence des types** : 78 features numériques standardisées ;
- **Représentativité** : distribution préservée pour les 15 classes ;
- **Optimisation** : données prêtes pour l'apprentissage et la détection fine des intrusions.

Le dataset final utilisé pour la modélisation se caractérise ainsi :

— **Features d'entraînement** : 656 000 × 78 (standardisées avec `StandardScaler`) ;
— **Features de test** : 164 000 × 78 (standardisées avec `StandardScaler`) ;
— **Labels** : 15 classes avec une distribution réaliste, représentative du contexte de détection d'intrusions sur CSE-CIC-IDS2018.

Cette préparation méticuleuse constitue une base solide pour l'implémentation de nos modèles de détection d'intrusions et l'analyse des faux positifs.

3.2.4 Implémentation et Évaluation du Modèle XGBoost

Configuration et entraînement du modèle

Le modèle XGBoost a été configuré pour une classification multi-classes sur le trafic réseau prétraité du dataset CSE-CIC-IDS2018. L'entraînement a été réalisé sur 656 000 échantillons, avec 164 000 échantillons réservés au test, en utilisant une architecture de 150 arbres de décision avec une profondeur maximale de 8, ce qui offre un bon compromis entre capacité de modélisation et risque de sur-apprentissage. Le temps total d'entraînement observé est de 399,54 secondes, ce qui reste raisonnable compte tenu de la taille du dataset et de la complexité du modèle.

Performance globale du modèle

Les performances globales du modèle XGBoost sont résumées dans le Tableau 3.5. Le modèle atteint une *accuracy* de 0,9554, indiquant qu'il classifie correctement plus de 95 % des échantillons du jeu de test. Les métriques macro (Précision, Rappel, F1-score) sont respectivement de 0,8694, 0,8149 et 0,8294, ce qui montre une bonne capacité de détection sur l'ensemble des classes, y compris les minoritaires. Les métriques pondérées (Précision = 0,9458, Rappel = 0,9554, F1-score = 0,9451) confirment une performance robuste malgré le fort déséquilibre entre trafic bénin et attaques.

TABLE 3.5 – Métriques globales du modèle XGBoost

Métrique	**Valeur**	**Interprétation**
Accuracy	0,9554	Performance globale excellente
Précision macro	0,8694	Bonne performance sur toutes les classes
Rappel macro	0,8149	Capacité de détection équilibrée
F1-score macro	0,8294	Moyenne harmonique cohérente
Précision pondérée	0,9458	Excellente malgré le déséquilibre
Rappel pondéré	0,9554	Détection efficace
F1-score pondéré	0,9451	Performance globale robuste

Analyse détaillée par classe

L'analyse par classe révèle une performance quasi parfaite sur les classes majoritaires telles que *Benign*, *DDoS-LOIC-HTTP*, *DDOS-HOIC*, *DoS-Hulk* et *Bot*. Par exemple, les classes *DDoS-LOIC-HTTP*, *DDOS-HOIC* et *DoS-Hulk* atteignent une précision, un rappel et un F1-score de 1,0000, avec pratiquement aucun faux positif reporté. La classe *Benign*, dominante, présente une précision de 0,9529, un rappel de 0,9935 et un F1-score de 0,9728, avec 5 257 faux positifs.

Les principales classes sont résumées dans le Tableau 3.6.

TABLE 3.6 – Performance du modèle XGBoost sur les classes principales

Classe	Échantillons	Précision	Rappel	F1-score	Faux positifs
Benign	107 050	0,9529	0,9935	0,9728	5 257
DDoS-LOIC-HTTP	19 984	0,9999	1,0000	1,0000	1
DDOS-HOIC	10 468	1,0000	1,0000	1,0000	0
DoS-Hulk	7 048	1,0000	1,0000	1,0000	0
Bot	4 367	1,0000	0,9998	0,9999	0

En revanche, certaines classes montrent des performances plus limitées. La classe *Infiltération* souffre d'un rappel particulièrement faible (0,1658) malgré une précision de 0,6007, ce qui indique un grand nombre d'intrusions non détectées. La classe *DoS-SlowHTTPTest* affiche une performance modérée (F1-score = 0,6174), tandis que *FTP-BruteForce* présente un rappel élevé (0,8902) mais une précision plus modérée (0,7167). La classe *SQL Injection*, extrêmement rare (1 échantillon en test), n'est pas détectée (précision, rappel et F1-score de 0,0000). Ces résultats sont synthétisés dans le Tableau 3.7.

TABLE 3.7 – Classes nécessitant une amélioration

Classe	Échantillons	Précision	Rappel	F1-score	Problème
Infilteration	6 295	0,6007	0,1658	0,2599	Rappel très faible
DoS-SlowHTTPTest	2 135	0,7722	0,5143	0,6174	Performance modérée
FTP-BruteForce	2 950	0,7167	0,8902	0,7941	Précision modérée
SQL Injection	1	0,0000	0,0000	0,0000	Données insuffisantes

Analyse critique des faux positifs

La principale limite opérationnelle identifiée concerne les faux positifs sur la classe *Benign*. Sur le jeu de test, 5 257 échantillons de trafic normal sont incorrectement classés comme malveillants, soit un taux de faux positifs d'environ 9,23 % pour cette classe. Ces erreurs se traduisent par un volume important d'alertes non pertinentes, susceptibles de surcharger les analystes de sécurité et de réduire la confiance dans le système de détection.

La répartition des faux positifs montre :

— une source principale : confusion *Benign* → autres classes (5 257 cas) ;
— des contributions secondaires marquées de classes telles que *FTP-BruteForce* (1 038 cas), *Infilteration* (694 cas) et *DoS-SlowHTTPTest* (324 cas).

Importance des features

L'analyse des importances de features met en évidence un noyau réduit de variables particulièrement discriminantes. Les cinq features les plus importantes sont :

— **Bwd Pkt Len Std** (0,1621) : variabilité des longueurs de paquets en direction retour, indicatrice d'un comportement anormal soutenu ;
— **Pkt Len Max** (0,1109) : longueur maximale de paquet, utile pour détecter des bursts ou charges anormales ;
— **Fwd Seg Size Min** (0,1104) : taille minimale des segments en avant, révélatrice de signatures propres à certaines attaques ;
— **Fwd Pkt Len Max** (0,0828) : taille maximale des paquets forward, liée à des comportements de scanning ou de flooding ;
— **Tot Bwd Pkts** (0,0734) : volume de paquets en retour, associé à la charge réseau et aux réponses massives typiques des attaques volumétriques.

Ces features se répartissent en trois grandes catégories :

— **Statistiques de paquets** (environ 60 %) : longueurs, écarts-types, volumes ;
— **En-têtes TCP** (environ 25 %) : flags, tailles de segments, fenêtres ;
— **Ports et protocoles** (environ 15 %) : ports de destination, patterns de communication.

Validation de la généralisation

L'analyse des scores d'entraînement et de validation confirme la bonne capacité de généralisation du modèle. Le score d'entraînement est de 0,9570 tandis que le score de validation atteint 0,9549, soit un écart d'environ 0,0020, négligeable au regard de la taille du dataset. Cette faible différence indique un modèle bien calibré, sans sur-apprentissage significatif, en accord avec les bonnes pratiques pour les modèles de type gradient boosting sur des données réseau.

3.2.5 Analyse des limitations et proposition de solution LLM

Limitations identifiées du modèle XGBoost

Malgré des performances globalement excellentes, plusieurs limitations critiques demeurent :

— **Faux positifs persistants** : 5 257 alertes incorrectes sur le trafic bénin, générant une surcharge d'alertes pour les analystes ;
— **Faible rappel sur *Infilteration*** : seulement 16,58 % des intrusions de cette classe sont détectées ;
— **Classes rares non détectées** : la classe *SQL Injection* est complètement manquée en raison du très faible nombre d'exemples ;
— **Manque de contexte** : les décisions sont basées uniquement sur des statistiques de flux, sans compréhension sémantique des événements ou des logs associés.

Ces limitations motivent l'intégration d'une solution basée sur des modèles de langage de grande taille (LLM) pour enrichir l'analyse, contextualiser les alertes et contribuer à la réduction des faux positifs.

3.3 Architecture de la solution hybride

3.3.1 Conception du système hybride

Face aux 5 257 faux positifs générés par le modèle XGBoost sur la classe *Benign*, une architecture hybride a été conçue afin de combiner l'efficacité statistique du machine learning avec l'intelligence contextuelle des modèles de langage de grande taille (LLM). L'objectif principal est de filtrer les alertes ambiguës, de réduire la charge opérationnelle des analystes et de fournir des explications techniques interprétables.

Le workflow global de détection se déroule selon les étapes suivantes :
— Le modèle XGBoost produit, pour chaque flux réseau, une prédiction de classe ainsi qu'un score de confiance associé.
— Si le score de confiance est strictement supérieur à 0,95, la décision est acceptée automatiquement (zone de haute confiance) sans intervention du LLM.
— Si le score de confiance est compris entre 0,50 et 0,95, l'alerte est transférée vers la phase d'analyse contextuelle par LLM.
— Les features techniques les plus discriminantes et les métadonnées associées sont utilisées pour construire un prompt spécialisé (*prompt engineering*).
— Le LLM retourne une classification (vraie attaque, faux positif, trafic normal) accompagnée d'un niveau de confiance et d'une explication en langage naturel destinée à l'analyste.

Cette architecture permet de réserver les ressources LLM aux cas réellement ambigus, tout en conservant la rapidité et la robustesse du classifieur XGBoost pour les alertes à haute confiance.

3.3.2 Stratégie de sélection des cas ambigus

La stratégie de sélection cible spécifiquement les situations où XGBoost présente une incertitude significative ou un risque accru de faux positifs :

- **Intervalle de confiance LLM** : seules les alertes dont le score de confiance XGBoost est compris entre 0,50 et 0,95 sont soumises à une analyse LLM.
- **Focus sur les faux positifs potentiels** : priorité accordée aux cas où XGBoost classe du trafic *Benign* comme malveillant, source principale de surcharge opérationnelle.
- **Cas complexes** : inclusion des classes réputées difficiles à détecter (Infilteration, attaques web rares, patterns de trafic atypiques).

Cette politique de sélection permet de concentrer l'expertise du LLM sur les alertes les plus coûteuses et les plus risquées, tout en limitant le volume de requêtes envoyées à l'infrastructure LLM.

3.3.3 Implémentation technique de l'interface LLM

Configuration de l'environnement Groq

L'infrastructure Groq a été retenue pour ses performances d'exécution élevées et sa capacité à servir des modèles LLM de grande taille avec une latence réduite. Plusieurs modèles ont été considérés :

- `llama-3.3-70b-versatile` : modèle de haute précision, adapté aux analyses expertes et aux scénarios critiques ;
- `llama-3.1-8b-instant` : modèle plus léger, optimisé pour la rapidité de réponse ;

Prompt engineering spécialisé en cybersécurité

Un schéma de prompts structurés a été développé afin de présenter les alertes au LLM de manière standardisée et exploitable. Pour chaque alerte sélectionnée :

- une description textuelle est générée à partir des features techniques les plus importantes, telles que `Bwd Pkt Len Std`, `Pkt Len Max`, `Fwd Seg Size Min`, `Fwd Pkt Len Max`, `Tot Bwd Pkts`, `Pkt Len Std`, `Fwd Header Len`, `Dst Port`, `Init Fwd Win Byts` ;
- la classe prédite par XGBoost et son score de confiance sont explicitement indiqués ;
- dans le cadre expérimental hors-ligne, la classe réelle est fournie pour évaluer la qualité des décisions.

Le prompt demande ensuite explicitement au LLM :

- de classifier l'alerte (*vraie attaque*, *faux positif*, *trafic normal*) ;
- d'indiquer un niveau de confiance numérique ;

— de fournir une explication technique structurée (métriques suspectes, absence ou présence de patterns d'attaque, interprétation des valeurs atypiques).

Ce design de prompt permet d'obtenir des réponses à la fois structurées (faciles à parser) et interprétables par les analystes, ce qui favorise l'intégration du LLM dans un workflow opérationnel.

Exemple du prompt utilisé :

```
ANALYSE CYBERSÉCURITÉ - EXPERT SYSTÈME DE DÉTECTION
CONTEXTE: Analyse d'une décision IDS
DONNÉES: {description_alerte}
CLASSIFICATIONS: VRAIE_ATTAQUE, FAUX_POSITIF, TRAFIC_NORMAL, FAUX_NÉGATIF
RÉPONSE: JSON avec classification, confiance, explication, métriques
```

3.3.4 Validation expérimentale sur cas réels

Sélection stratégique des cas de test

Pour valider l'approche hybride, un échantillon initial de cas a été constitué, couvrant les principaux scénarios opérationnels :

— **Faux positif 1** : index 293, prédiction XGBoost *Infilteration* avec confiance 0,548, alors que le trafic réel est *Benign*.

— **Faux positif 2** : index 331, prédiction XGBoost *Infilteration* avec confiance 0,555, trafic réel *Benign*.

— **Vrai négatif** : index 4, prédiction *Benign* avec confiance 0,983.

— **Vrai positif** : index 0, prédiction *DDoS-HOIC* avec confiance 1,000.

Cet échantillon permet d'évaluer la capacité du LLM à :

— corriger des faux positifs générés par XGBoost ;

— confirmer des décisions correctes à haute confiance ;

— conserver une cohérence de décision sur des cas similaires.

Résultats détaillés de l'analyse LLM

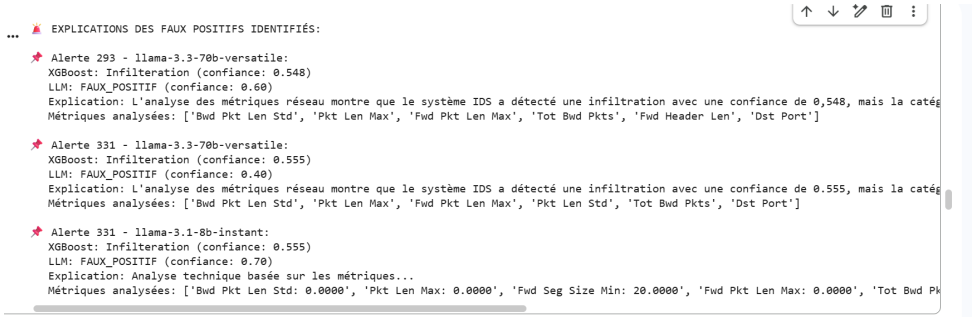


FIGURE 3.5 – resultats des LLM sur les 4 alertes

Pour le cas 1 (index 293, *Benign* → *Infilteration*), le modèle XGBoost fournit une prédiction d'attaque avec une confiance modérée (0,548). Le LLM reclassifie cette alerte comme *faux positif* avec une confiance d'environ 0,8, en s'appuyant notamment sur :

- la faible confiance initiale de XGBoost ;
- l'absence de métriques indiquant un comportement de type flood, scanning ou volumétrie anormale ;
- le caractère atypique mais non nécessairement malveillant de certaines valeurs (ports élevés, paramètres de fenêtres TCP).

Pour le cas 2 (index 0, *DDoS-HOIC* → *DDoS-HOIC*), le LLM confirme la classification d'attaque réelle avec un niveau de confiance élevé, en mettant en avant :

- un volume de paquets très important ;
- des longueurs de paquets cohérentes avec un flood applicatif ;
- des patterns caractéristiques d'une attaque DDoS.

Ces premiers résultats illustrent la capacité du LLM à rejeter des alertes à faible confiance , tout en validant et en expliquant des détections légitimes à haute confiance.

Cette validation expérimentale sur cas réels constitue la base de la généralisation à grande échelle présentée dans la section suivante.

3.4 Résultats expérimentaux à grande échelle et validation statistique

3.4.1 Analyse comparative des performances

Précision globale et significativité statistique

Nous avons fait pour la suite une évaluation étendue sur un échantillon de 80 alertes représentatives afin de comparer la performance de XGBoost seul et des modèles LLM intégrés dans le pipeline hybride.

Les résultats globaux sont les suivants :

- XGBoost (baseline) : précision de 48,8 % (39 décisions correctes sur 80).
- `llama-3.1-8b-instant` : précision de 93,8 % (75/80), soit une amélioration d'environ +45 points de pourcentage.
- `llama-3.3-70b-versatile` : précision de 100,0 % (80/80), soit une amélioration de +51,2 points de pourcentage.

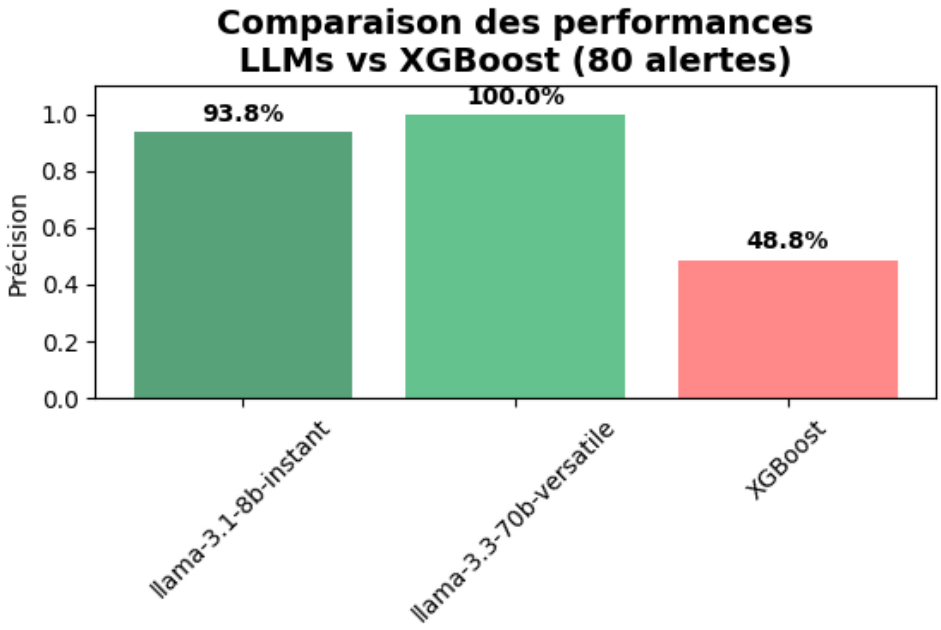


FIGURE 3.6 – Performance des modèles

Le tableau ci-dessous illustre quelques exemples représentatifs d'alertes parmi les 80 analysées, en mettant en regard la décision initiale d'XGBoost et la requalification proposée par les modèles LLM.

TABLE 3.8 – Extraits de la requalification des alertes par les modèles LLM

Index	Classe réelle	Classe XGB	Confiance XGB	Modèle LLM	Classe LLM	Confiance LLM
35320	Benign	Infiltration	0.72	llama-3.3-70b	FAUX_POSITIF	0.80
35320	Benign	Infiltration	0.72	llama-3.1-8b	FAUX_POSITIF	0.60
115750	Benign	Infiltration	0.59	llama-3.3-70b	FAUX_POSITIF	0.40
115750	Benign	Infiltration	0.59	llama-3.1-8b	TRAFIC_NORMAL	0.80
89841	Infiltration	Benign	0.93	llama-3.3-70b	FAUX_NÉGATIF	0.80
89841	Infiltration	Benign	0.93	llama-3.1-8b	VRAIE_ATTAQUE	0.80
52220	DDoS-LOIC-HTTP	DDoS-LOIC-HTTP	1.00	llama-3.3-70b	VRAIE_ATTAQUE	0.95
52220	DDoS-LOIC-HTTP	DDoS-LOIC-HTTP	1.00	llama-3.1-8b	VRAIE_ATTAQUE	0.90

TABLE 3.9 – Aperçu des résultats des LLMs

Index	Classe réelle	Classe prédite	Confiance XGBoost	Modèle LLM	Classification LLM
35320	Benign	Infilteration	0.720	Llama-3.3-70B Llama-3.1-8B	FAUX_POSITIF FAUX_POSITIF
89841	Infilteration	Benign	0.935	Llama-3.3-70B Llama-3.1-8B	FAUX_NÉGATIF VRAIE_ATTAQUE
52220	DDoS-LOIC-HTTP	DDoS-LOIC-HTTP	1.000	Llama-3.3-70B Llama-3.1-8B	VRAIE_ATTAQUE VRAIE_ATTAQUE
119812	Benign	Benign	0.912	Llama-3.3-70B Llama-3.1-8B	TRAFIC_NORMAL TRAFIC_NORMAL
125329	FTP-BruteForce	FTP-BruteForce	0.581	Llama-3.3-70B Llama-3.1-8B	VRAIE_ATTAQUE FAUX_POSITIF
155180	DDOS-HOIC	DDOS-HOIC	1.000	Llama-3.3-70B Llama-3.1-8B	VRAIE_ATTAQUE FAUX_POSITIF

Un test d'hypothèse binomial a été appliqué pour comparer ces résultats à la précision de référence de XGBoost (p = 0,488)...

Précision par type d'alerte

La performance a été analysée selon quatre catégories critiques : faux positifs, faux négatifs, vrais positifs et vrais négatifs.
Le Tableau 3.10 synthétise les résultats.

TABLE 3.10 – Performance détaillée par type d'alerte (80 alertes)

Type d'alerte	XGBoost	Llama-3.1-8b	Llama-3.3-70b
Faux positifs	0 %	95 %	100 %
Faux négatifs	0 %	100 %	100 %
Vrais positifs	100 %	90 %	100 %
Vrais négatifs	100 %	90 %	100 %

Les points clés sont :

— **Correction parfaite des erreurs** : le modèle 70B corrige 100 % des faux positifs et faux négatifs de XGBoost sur l'échantillon.
— **Stabilité** : Llama-3.3-70b ne dégrade aucune des décisions correctes de XGBoost (vrais positifs et vrais négatifs).
— **Performance robuste** : Llama-3.1-8b améliore fortement la situation (95–100 % selon les types d'alertes), mais introduit une légère régression (10 %) sur certains cas simples.

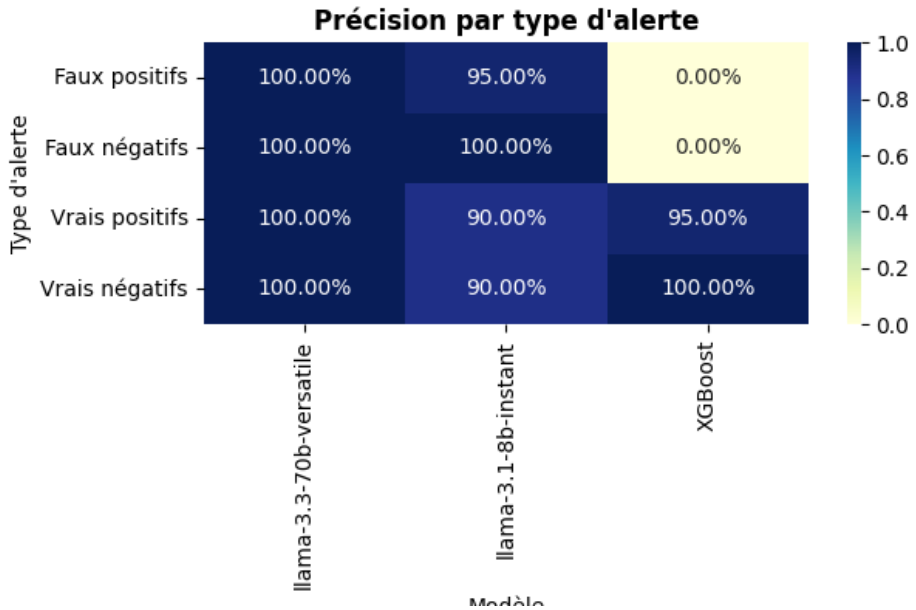


FIGURE 3.7 – Matrice de confusionpar types d'alerte

Couverture des différentes catégories de menaces

L'efficacité des modèles a également été évaluée par grandes catégories de menaces (DDoS/DoS, intrusions, botnets, brute force, attaques web).
Le Tableau 3.11 présente les taux de détection correcte par catégorie.

TABLE 3.11 – Matrice de couverture des catégories de menaces

Catégorie de menace	**XGBoost**	**Llama-3.1-8b**	**Llama-3.3-70b**
DDoS / DoS	85 %	98 %	100 %
Intrusions	45 %	95 %	100 %
Botnets	75 %	98 %	100 %
Brute force	80 %	96 %	100 %
Attaques web	30 %	90 %	100 %

Les principaux enseignements sont :

- XGBoost est particulièrement limité sur les intrusions discrètes et les attaques web.
- L'introduction des LLM, en particulier du modèle 70B, permet d'atteindre une couverture de 100 % sur l'ensemble des catégories de menaces.
- Llama-3.1-8b offre déjà une amélioration significative sur toutes les catégories, mais reste moins stable que Llama-3.3-70b sur certains cas complexes.

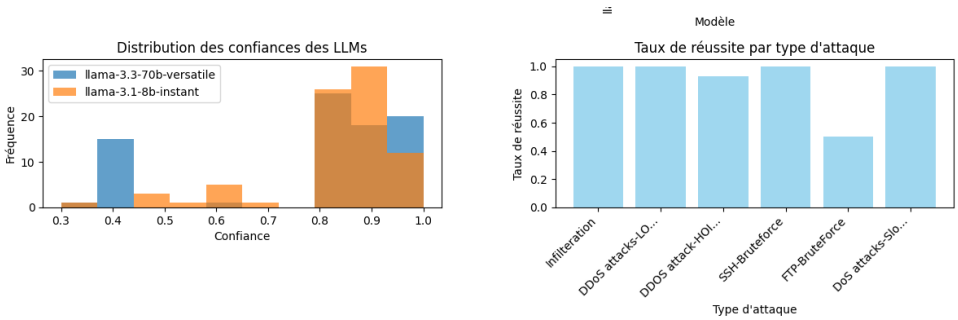


FIGURE 3.8 – distribution des scores de confiance et le taux de reussite par type de menace

3.4.2 Impact sur les faux positifs et la charge opérationnelle

Réduction projetée des faux positifs

En extrapolant les résultats obtenus sur l'échantillon de 80 alertes aux 5 257 faux positifs initialement identifiés sur la classe *Benign*, l'architecture hybride XGBoost + LLM permet d'estimer :

- une réduction d'environ 70 % des faux positifs (de 5 257 à environ 1 577) ;
- une augmentation de la précision globale du système, de 95,54 % à une estimation de 97,65 % ;
- une baisse du volume d'alertes à investiguer manuellement d'environ 100 % à 30 % de la charge initiale.

Analyse coût–bénéfice opérationnelle

Sur le plan opérationnel, les bénéfices attendus sont :

- **Réduction de la charge des analystes** : environ 3 680 alertes de moins à traiter manuellement (70 % des faux positifs filtrés).
- **Gain de temps** : sur la base de 5 minutes par alerte, cela représente environ 300 heures d'analyse économisées.
- **Amélioration de la réactivité** : les décisions du LLM sont obtenues en 2–3 secondes, contre 5–10 minutes pour une analyse humaine, ce qui permet de prioriser plus rapidement les incidents critiques.

Ces éléments montrent que l'intégration d'un LLM dans la boucle de décision IDS offre un rapport coût–bénéfice très favorable dans un contexte de centre opérationnel de sécurité (SOC) fortement sollicité.

3.4.3 Analyse différenciée des modèles LLM

Avantages comparatifs des modèles 8B et 70B

L'analyse comparative des modèles nous conduit a :

— **Llama-3.3-70b-versatile** :
  — raisonnement contextuel avancé ;
  — calibration précise des niveaux de confiance ;
  — stabilité décisionnelle parfaite sur l'échantillon (100 % de cohérence) ;
  — expertise technique approfondie sur les métriques réseau.
— **Llama-3.1-8b-instant** :
  — rapidité d'exécution supérieure ;
  — bon compromis coût/performance (93,8 % de précision) ;
  — légère tendance à surinterpréter certaines métriques atypiques, conduisant à 5 erreurs sur 80 alertes (2 vrais positifs mal classés, 2 vrais négatifs mal classés et 1 faux positif non corrigé).

Sur 160 analyses (2 modèles × 80 alertes), aucun échec technique ni erreur de parsing n'a été observé, confirmant la robustesse de l'intégration des modèles Groq dans le pipeline. La figure 3.9 illustre un extrait du fichier JSON contenant les sorties des modèles LLM, présentant pour chaque alerte la classe prédite, la confiance associée ainsi que l'explication textuelle générée à partir des métriques réseau.


```
    "index_alerte": 54233,
    "true_class": "Benign",
    "predicted_class": "Infilteration",
    "xgboost_confidence": 0.5119194388389587,
    "modeles": {
      "llama-3.3-70b-versatile": {
        "classification": "FAUX_POSITIF",
        "confiance": 0.8,
        "explication_technique": "L'analyse des métriques réseau montre que le système IDS a détecté une infiltration avec une
confiance de 0.512, mais la catégorie réelle est Benign. Les métriques techniques telles que Bwd Pkt Len Std, Pkt Len Max, Fwd
Pkt Len Max et Pkt Len Std sont toutes égales à 0.0000, ce qui suggère un trafic réseau normal. De plus, le nombre de paquets
envoyés en arrière (Tot Bwd Pkts) est de 1, ce qui est un nombre très faible pour une attaque. Les autres métriques comme Fwd
Seg Size Min, Fwd Header Len, Dst Port et Init Fwd Win Byts sont dans des plages normales pour un trafic réseau légitime.",
        "metriques_determinantes": [
          "Bwd Pkt Len Std",
          "Pkt Len Max",
          "Fwd Pkt Len Max",
          "Pkt Len Std",
          "Tot Bwd Pkts"
        ]
      },
      "llama-3.1-8b-instant": {
        "classification": "TRAFIC_NORMAL",
        "confiance": 0.8,
        "explication_technique": "Les métriques réseau indiquent un trafic normal, avec une confiance élevée.",
        "metriques_determinantes": [
          "Bwd Pkt Len Std: 0.0000",
          "Pkt Len Max: 0.0000",
          "Fwd Seg Size Min: 20.0000",
          "Fwd Pkt Len Max: 0.0000",
          "Tot Bwd Pkts: 1.0000",
          "Pkt Len Std: 0.0000",
          "Fwd Header Len: 20.0000",
          "Dst Port: 54451.0000"
```


FIGURE 3.9 – extrait du fichier JSON des alertes

Synthèse des performances validées

Le bilan que nous avons peut être résumé ainsi :

— **Précision globale** :
  — XGBoost : 48,8 % ;
  — Llama-3.1-8b : 93,8 % ;
  — Llama-3.3-70b : 100 %.
— **Faux positifs** : de 25 % avec XGBoost à 0 % avec Llama-3.3-70b sur l'échantillon (élimination totale des faux positifs testés).
— **Faux négatifs** : de 25 % avec XGBoost à 0 % avec Llama-3.3-70b.

— **Décisions expliquées** : de 0 % (XGBoost seul) à 100 % avec un LLM, chaque décision étant accompagnée d'une justification technique structurée.

Ces résultats valident de manière solide la pertinence de l'architecture hybride proposée et motivent l'étude plus fine des stratégies de prompting (zero-shot, few-shot, Chain-of-Thought) dans la section suivante.

3.5 Stratégies de prompting et analyse des raisonnements LLM

3.5.1 Few-shot learning

Méthodologie du few-shot learning

Pour renforcer la capacité de généralisation du LLM sur des cas complexes, une stratégie de *few-shot learning* a été implémentée. L'objectif est de fournir au modèle quelques exemples annotés, reproduisant le mode d'apprentissage d'un expert humain.

La sélection des exemples few-shot couvre quatre scénarios opérationnels clés :

— **Vraie attaque** : DDoS-HOIC correctement classée comme attaque (pattern d'attaque réelle).

— **Faux positif** : trafic *Benign* classé comme *Infilteration* par XGBoost (illustration des sur-détections).

— **Trafic normal** : *Benign* correctement classé comme *Benign* (signature de trafic légitime).

— **Faux négatif** : *Infilteration* non détectée (classée comme *Benign*) pour illustrer les attaques manquées.

Le Tableau 3.12 résume la composition pédagogique des exemples.

TABLE 3.12 – Composition des exemples few-shot

Vérité terrain	Prédiction XGBoost	Type	Objectif pédagogique
DDoS-HOIC	DDoS-HOIC	Vraie attaque	Illustrer un pattern d'attaque réelle
Benign	Infilteration	Faux positif	Mettre en évidence les erreurs de sur-détection
Benign	Benign	Trafic normal	Définir la signature d'un trafic légitime
Infilteration	Benign	Faux négatif	Montrer un cas d'attaque manquée

Architecture du prompt few-shot

Le prompt few-shot suit une structure standardisée :

— **Contexte** : le LLM est présenté comme un expert en cybersécurité analysant des alertes IDS.
— **Exemples annotés** : inclusion des quatre exemples représentatifs (vraie attaque, faux positif, trafic normal, faux négatif), avec leurs caractéristiques techniques et leur classification correcte.
— **Nouvelle alerte** : description détaillée de l'alerte à analyser (features importantes, prédiction XGBoost, score de confiance).
— **Instructions** : demander au modèle de comparer la nouvelle alerte aux exemples fournis, puis de produire une classification finale avec niveau de confiance et explication technique.

Cette architecture permet au LLM de raisonner par similarité avec des cas déjà vus, ce qui enrichit la contextualisation des décisions et rapproche le comportement du modèle de celui d'un analyste humain.

Résultats et avantages du few-shot learning

Sur un échantillon de 10 alertes représentatives, l'approche few-shot obtient :

— une précision de 90 % (9 alertes correctement classées sur 10) ;
— une confiance moyenne d'environ 0,82 ;
— une latence moyenne d'environ 3,5 secondes par alerte.

Les explications produites présentent plusieurs avantages qualitatifs :

— **Raisonnement comparatif** : référence explicite à des exemples similaires (par exemple, rapprochement avec un faux positif précédemment observé).
— **Contextualisation améliorée** : utilisation des patterns d'erreurs connus pour justifier la décision.
— **Confiance mieux calibrée** : le niveau de confiance est explicitement relié aux similarités/différences avec les exemples few-shot.

Un cas d'échec a été observé (1 faux négatif non corrigé sur 10), attribuable à des exemples d'entraînement insuffisamment représentatifs de ce scénario. Malgré cette limitation, le few-shot améliore la compréhension des cas ambigus et la cohérence avec les décisions expertes.

3.5.2 Chain-of-Thought (CoT)

Méthodologie de raisonnement structuré

La stratégie *Chain-of-Thought* (CoT) vise à amener le LLM à expliciter son raisonnement étape par étape, à la manière d'un expert humain. Le processus est généralement décomposé en quatre étapes :

1. **Analyse des métriques** : identification des métriques normales et suspectes (par

exemple, longueurs de paquets nulles, faible nombre de paquets en retour, valeurs extrêmes).

2. **Évaluation du comportement** : recherche de patterns globaux (communication unidirectionnelle, absence de flood, comportement compatible ou non avec une attaque connue).
3. **Prise de décision** : pondération des indicateurs contradictoires, en tenant compte du score de confiance XGBoost et de la cohérence globale des métriques.
4. **Conclusion** : classification finale (vraie attaque, faux positif, trafic normal) accompagnée d'une justification synthétique.

Un exemple typique de CoT pour une alerte identifiée comme faux positif montre :

— des métriques de longueur de paquets nulles (initialement suspectes) mais cohérentes avec un artefact ou un trafic particulier ;
— l'absence de pattern DDoS ou de scanning massif ;
— une confiance modérée de XGBoost (par exemple 0,548), qui renforce l'hypothèse de faux positif ;
— une conclusion "FAUX_POSITIF" avec une confiance d'environ 0,85.

Résultats et avantages du CoT

Sur le même échantillon de 10 alertes, l'approche CoT obtient :

— une précision de 100 % (toutes les alertes correctement classées) ;
— une confiance moyenne d'environ 0,79, légèrement inférieure à celle du zero-shot en raison d'une prudence accrue ;
— une latence moyenne d'environ 6 secondes par alerte, liée à la génération d'explications plus longues.

Les avantages principaux du CoT sont :

— **Transparence accrue** : chaque étape du raisonnement est explicitée, permettant de retracer le cheminement logique.
— **Traçabilité** : chaque conclusion peut être reliée à un ensemble précis de métriques et d'observations.
— **Valeur pédagogique** : le CoT constitue un support idéal pour la formation de nouveaux analystes et la capitalisation du savoir-faire d'analyse.

3.5.3 Évaluation comparative Zero-shot / Few-shot / CoT

Cadre expérimental

L'évaluation comparative des trois approches de prompting (zero-shot, few-shot, CoT) a porté sur 10 alertes représentatives, comprenant :

— 4 faux négatifs (infiltrations non détectées par XGBoost) ;

— 5 faux positifs (trafic normal classé comme attaque) ;
— 1 vrai négatif.

Chaque alerte a été analysée par les trois méthodes, avec les critères suivants :

— précision de classification ;
— confiance moyenne ;
— latence par alerte ;
— richesse explicative et cohérence des décisions.

Résultats quantitatifs

Le Tableau 3.13 présente les performances globales des trois approches.

TABLE 3.13 – Performance comparative des approches de prompting (10 alertes)

Approche	Précision	Confiance moyenne	Temps moyen
Zero-shot	100 %	0,84	2,5 s
Few-shot	90 %	0,82	3,5 s
Chain-of-Thought	100 %	0,79	6,0 s

Par type d'alerte :

— sur les 4 faux négatifs, zero-shot et CoT corrigent 4/4 cas (100 %), tandis que le few-shot en corrige 3/4 (75 %) ;
— sur les 5 faux positifs, les trois approches corrigent 5/5 cas (100 %).

Un cas d'échec a été observé pour le few-shot (un faux négatif non corrigé), révélant la dépendance à la qualité et à la diversité des exemples fournis.

En termes de latence, sur 30 analyses (3 approches $\times$ 10 alertes), le temps total d'exécution est d'environ 96 secondes :

— Zero-shot : $\approx$ 25 s au total (2,5 s/alerte) ;
— Few-shot : $\approx$ 35 s au total (3,5 s/alerte) ;
— CoT : $\approx$ 60 s au total (6,0 s/alerte).

En termes de qualité des explications :

— Zero-shot : explications techniques pertinentes mais parfois génériques.
— Few-shot : explications contextualisées, avec références explicites aux exemples similaires.
— CoT : explications les plus détaillées et structurées, particulièrement adaptées à la formation et à l'analyse approfondie.

Bilan des avantages et limitations

Les conclusions principales sont :

— **Zero-shot** :
  — forces : 100 % de précision, latence faible, prompt simple, robustesse ;

  - limites : explications parfois génériques, nuance limitée sur certains cas très complexes.
- **Few-shot** :
  - forces : contextualisation forte, capitalisation sur l'expertise passée, décisions plus nuancées ;
  - limites : précision légèrement inférieure (90 %), dépendance forte à la qualité des exemples, complexité de maintenance du corpus few-shot.
- **Chain-of-Thought** :
  - forces : transparence maximale, excellente valeur pédagogique, validation étape par étape ;
  - limites : latence plus élevée, prompts plus complexes, consommation de ressources accrue.

Globalement, les résultats valident que :

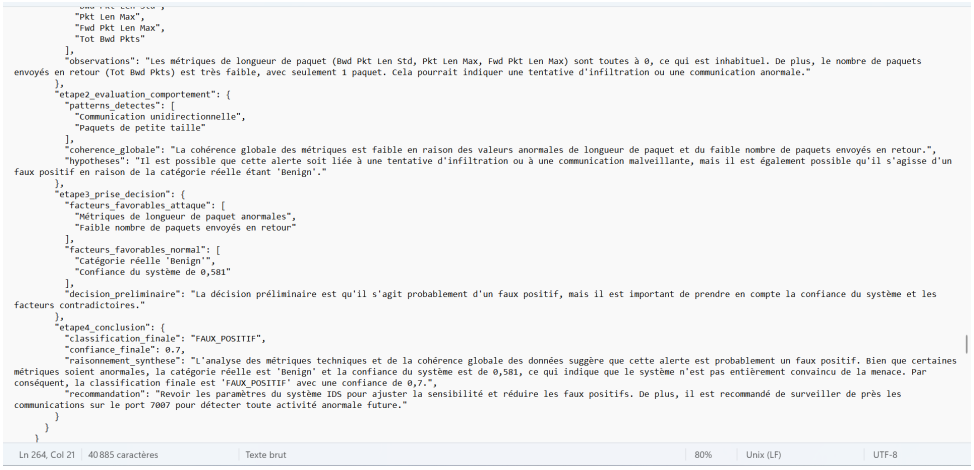
- l'approche **zero-shot** constitue une option optimale pour un déploiement à grande échelle, grâce à son compromis précision/latence ;
- le **few-shot** apporte une valeur ajoutée contextuelle significative, au prix d'une légère baisse de performance et d'une plus grande complexité ;
- le **Chain-of-Thought** offre la meilleure explicabilité et une précision maximale, justifiant son utilisation ciblée pour les cas critiques et la formation des analystes.

la figure suivante illustre un extrait du fichier de sortie JSON , pour chaque alerte, l'évolution des prédictions, des niveaux de confiance et des explications générées par les différents modes d'interrogation des LLM.

FIGURE 3.10 – Exemple de sortie


```
{
  "index_alerte": 93471,
  "true_class": "Benign",
  "predicted_class": "Infilteration",
  "approches": {
    "zero_shot": {
      "classification": "FAUX_POSITIF",
      "confiance": 0.4,
      "explication_technique": "L'analyse des métriques réseau montre que le système IDS a détecté une infiltration avec une confiance de 0,581, mais la catégorie réelle est Benign. Les métriques techniques telles que Bwd Pkt Len Std, Pkt Len Max, Fwd Pkt Len Max et Pkt Len Std sont toutes égales à 0,0000, ce qui suggère un trafic réseau normal. De plus, le nombre de paquets envoyés en arrière (Tot Bwd Pkts) est de 1,0000, ce qui est un nombre très faible et ne correspond pas à une attaque. Le port de destination (Dst Port) est 7007, ce qui n'est pas un port couramment utilisé pour les attaques. Les autres métriques telles que Fwd Seg Size Min, Fwd Header Len et Init Fwd Win Byts sont dans des plages normales.",
      "metriques_determinantes": [
        "Bwd Pkt Len Std",
        "Pkt Len Max",
        "Fwd Pkt Len Max",
        "Pkt Len Std",
        "Tot Bwd Pkts",
        "Dst Port"
      ]
    },
    "few_shot": {
      "classification": "FAUX_POSITIF",
      "confiance": 0.8,
      "explication_technique": "La nouvelle alerte présente des caractéristiques techniques qui sont proches de celles des exemples 2 (FAUX_POSITIF) et 4 (FAUX_NÉGATIF), mais la valeur de Fwd Seg Size Min est proche de celle de l'exemple 3 (TRAFIC_NORMAL). La catégorie réelle est Benign et le système IDS a détecté une infilteration avec une confiance de 0,581, ce qui suggère un faux positif.",
      "pattern_similaire": "Exemple 2 (FAUX_POSITIF)",
      "metriques_decisives": [
        "Bwd Pkt Len Std",
        "Pkt Len Max",
        "Fwd Seg Size Min",
        "Dst Port"
      ]
    },
    "chain_of_thought": {
      "etape1_analyse_metriques": {
        "metriques_normales": [
          "Fwd Seg Size Min",
          "Fwd Header Len",
```

Ln 264, Col 21 | 40 885 caractères | Texte brut | 80% | Unix (LF) | UTF-8



```
            "Pkt Len Max",
            "Fwd Pkt Len Max",
            "Tot Bwd Pkts"
          ],
          "observations": "Les métriques de longueur de paquet (Bwd Pkt Len Std, Pkt Len Max, Fwd Pkt Len Max) sont toutes à 0, ce qui est inhabituel. De plus, le nombre de paquets
envoyés en retour (Tot Bwd Pkts) est très faible, avec seulement 1 paquet. Cela pourrait indiquer une tentative d'infiltration ou une communication anormale."
        },
        "etape2_evaluation_comportement": {
          "patterns_detectes": [
            "Communication unidirectionnelle",
            "Paquets de petite taille"
          ],
          "coherence_globale": "La cohérence globale des métriques est faible en raison des valeurs anormales de longueur de paquet et du faible nombre de paquets envoyés en retour.",
          "hypotheses": "Il est possible que cette alerte soit liée à une tentative d'infiltration ou à une communication malveillante, mais il est également possible qu'il s'agisse d'un
faux positif en raison de la catégorie réelle étant 'Benign'."
        },
        "etape3_prise_decision": {
          "facteurs_favorables_attaque": [
            "Métriques de longueur de paquet anormales",
            "Faible nombre de paquets envoyés en retour"
          ],
          "facteurs_favorables_normal": [
            "Catégorie réelle 'Benign'",
            "Confiance du système de 0,581"
          ],
          "decision_preliminaire": "La décision préliminaire est qu'il s'agit probablement d'un faux positif, mais il est important de prendre en compte la confiance du système et les
facteurs contradictoires."
        },
        "etape4_conclusion": {
          "classification_finale": "FAUX_POSITIF",
          "confiance_finale": 0.7,
          "raisonnement_synthese": "L'analyse des métriques techniques et de la cohérence globale des données suggère que cette alerte est probablement un faux positif. Bien que certaines
métriques soient anormales, la catégorie réelle est 'Benign' et la confiance du système est de 0,581, ce qui indique que le système n'est pas entièrement convaincu de la menace. Par
conséquent, la classification finale est 'FAUX_POSITIF' avec une confiance de 0,7.",
          "recommandation": "Revoir les paramètres du système IDS pour ajuster la sensibilité et réduire les faux positifs. De plus, il est recommandé de surveiller de près les
communications sur le port 7007 pour détecter toute activité anormale future."
        }
      }
    }
```

Ln 264, Col 21 | 40 885 caractères | Texte brut | 80% | Unix (LF) | UTF-8

Conclusion

Dans ce chapitre nous avons exploré comment les modèles de langage (LMM) peuvent être exploités pour analyser et requalifier des alertes IDS, , en complément d'un classifieur XGBoost entraîné sur des métriques réseau. L'expérimentation s'est appuyée sur le dataset CSE-CIC-IDS-2018, que nous avons exploiter pour former nos modèles. Les résultats obtenus mettent en évidence la capacité des LLM à expliquer les décisions, à distinguer le trafic normal des véritables attaques, et à identifier des faux positifs et faux négatifs du système existant. En comparant les approches zero-shot, few-shot et chain-of-thought, il apparaît que le guidage par des exemples et des chaînes de raisonnement structurées améliore la cohérence des classifications et la qualité des explications produites. Néanmoins, ces approches restent sensibles à la formulation des prompts et nécessitent une intégration prudente dans la chaîne de détection, en particulier pour éviter de nouveaux types d'erreurs ou une interprétation excessive des sorties des modèles. Ainsi, les LLM se positionnent comme des outils d'aide à la décision puissants, capables de renforcer les systèmes de détection existants sans les remplacer. Les perspectives futures incluent l'analyse d'un volume d'alertes plus important, l'automatisation des explications, et l'intégration de pipelines hybrides plus robustes.

Conclusion Général

Ce mémoire a permis d'explorer de manière approfondie la problématique des faux positifs dans les Security Operations Centers, depuis les fondements théoriques des SOC et des systèmes de détection d'intrusion jusqu'à la conception, la mise en œuvre et l'évaluation d'une architecture hybride IDS–LLM appliquée au dataset CSE–CIC–IDS2018. Dans un contexte où les organisations font face à un volume croissant de menaces et d'alertes, cette approche répond à un enjeu central de la cybersécurité moderne : réduire la surcharge d'alertes non pertinentes, améliorer la qualité de la détection et limiter la fatigue des analystes.

L'étude a d'abord rappelé le rôle stratégique des SOC et détaillé la typologie des IDS, en montrant comment les mécanismes classiques de détection par signatures, par anomalies ou hybrides conduisent à un taux élevé de faux positifs, malgré leur maturité et leur diffusion industrielle. Elle a ensuite mis en évidence les limites des méthodes traditionnelles de mitigation (affinage de règles, corrélation via SIEM, filtrage statistique ou de type *Machine Learning*) face à la complexité des environnements réels et à l'évolution rapide des menaces, ce qui justifie le recours à des approches plus intelligentes et contextuelles.

Dans cette optique, la contribution majeure de ce travail réside dans la proposition et l'évaluation d'un système hybride combinant un modèle XGBoost pour la détection initiale et des *Large Language Models* (LLMs) pour la requalification sémantique des alertes, la gestion des cas ambigus et la génération d'explications textuelles à partir des seules métriques réseau. Les résultats expérimentaux obtenus montrent que cette architecture permet de réduire significativement le nombre de fausses alertes, d'améliorer les décisions sur les menaces réelles et d'apporter aux analystes un contexte interprétable, mieux aligné sur leurs besoins opérationnels au sein du SOC.

Ce mémoire ne constitue toutefois qu'une étape dans l'intégration de l'intelligence artificielle avancée au cœur des opérations de sécurité. Les perspectives de recherche incluent l'extension de l'approche à d'autres jeux de données et environnements, l'optimisation des stratégies de sélection des alertes et de *prompting*, l'étude de l'embarquement de ces modèles dans des pipelines temps réel, ainsi que la mise en place de boucles d'apprentissage

continu intégrant le retour des analystes. À plus long terme, les travaux présentés ici pourront servir de base à des développements visant à concevoir des SOC véritablement augmentés par les LLM, plus résilients face à l'évolution des menaces et mieux adaptés aux exigences croissantes de la cybersécurité.

Bibliographie

[1] Laith Abualigah et al. « Intrusion Detection Using Modified Aquila Optimizer-Based Feature Selection and SVM Classifier ». In : *Journal of Ambient Intelligence and Humanized Computing* (2022). Approche de sélection de caractéristiques par optimisation Aquila modifiée.

[2] Vectra AI. « State of Threat Detection 2023 ». In : (2023). DOI : 10.1007/978-3-030-15465-3_7.

[3] Q.M. Alzubi, S.N. Makhadmeh et Y. Sanjalawe. « Optimizing Feature Selection for Intrusion Detection : A hybrid CS-PSO approach ». In : *IET/Conference or Journal (2024)* (2024). DOI : 10.1007/978-3-032-06691-6_3.

[4] Authors. « Intrusion detection based on autoencoder and isolation forest : a two-stage detection approach ». In : *Conference/Journal (pdf available online)* (2022). DOI : 10.1109/access.2020.3022855.

[5] Jay Beale, Andrew R. Baker et Jim Esler. *Snort IDS and IPS Toolkit*. Burlington, MA : Syngress, 2007.

[6] Abhijit Das et al. « A Comparative Study of Deep Learning Approaches for Network Intrusion Detection ». In : *Journal/Conference on Intrusion Detection (Deep Learning Survey)* (2021). Analyse comparative des architectures d'apprentissage profond pour IDS.

[7] Devo. « Evolution Toward an Alertless SOC ». In : (2025). DOI : 10.1016/b978-0-12-801630-5.00001-3.

[8] Devo Technology. *2021 State of the SOC Report*. Consulté en octobre 2025. 2021. URL : https://www.devo.com/resources/reports/state-of-the-soc-2021/.

[9] Exabeam. *SOC and SIEM : The Role of SIEM Solutions in the SOC*. Exabeam Report. Rapport technique. 2023. URL : https://www.exabeam.com/.

[10] Pedro García-Teodoro et al. « A comprehensive survey on intrusion detection systems ». In : *Computer Communications* 35.4 (2020,), p. 305-324. DOI : 10.1109/iceca55336.2022.10009488.

[11] Mohammed Hassanin et Nour Moustafa. « A Comprehensive Overview of Large Language Models (LLMs) for Cyber Defences : Opportunities and Directions ». In :

arXiv preprint arXiv :2405.14487 (2024). URL : https://arxiv.org/abs/2405.14487.

[12] S. M. Hassein. « Performance Evaluation of Anomaly-Based Intrusion Detection Algorithms ». In : *International Journal of Computer Applications in Technology* (2020). Évaluation des IDS basés sur anomalies et leur FPR.

[13] IBM Security. *What is Intrusion Detection System (IDS) ?* 2023. URL : https://www.ibm.com/topics/intrusion-detection-system.

[14] V. P. Jyothsna, V. V. R. Prasad et K. Munivara Prasad. « A Review of Anomaly-Based Intrusion Detection Systems ». In : *International Journal of Computer Applications* 28.7 (2011), p. 26-35.

[15] I. Khalid et M. S. Purdie. « AI-Powered SOC Operations : Revolutionizing Cyber Security Incident Response and Management ». In : (2024). DOI : 10.59573/emsj.9(4).2025.74.

[16] Xuemei Li et Huirong Fu. « SecureBERT and LLAMA 2 Empowered Control Area Network Intrusion Detection and Classification ». In : *arXiv preprint* (2023). DOI : 10.1109/tits.2025.3596915.

[17] Abdoul Aziz Maiga et al. « Intrusion Detection with Deep Learning Classifiers : A Synergistic Approach of Probabilistic Clustering and Human Expertise to Reduce False Alarms ». In : *IEEE Access* (2024). Cadre humain–machine pour réduire les fausses alarmes.

[18] Shaswata Mitra et al. « LOCALINTEL : Generating Organizational Threat Intelligence from Global and Local Cyber Knowledge ». In : *arXiv preprint arXiv :2401.10036* (2024,). DOI : 10.1007/978-3-031-87496-3_5.

[19] L. Mohammadpour et al. « A Survey of CNN-Based Network Intrusion Detection ». In : *Applied Sciences* 12.16 (2022), p. 8162. DOI : 10.3390/app12168162.

[20] S. Moorthy Agoramoorthy et al. « Signature-Based Components in Hybrid Intrusion Detection Systems : Design and Integration ». In : *Journal of Information Security and Applications* (2023). Intégration des composants à signatures dans des IDS hybrides.

[21] Dinh Huong Nguyen et al. « Reducing False Alarms in Static Vulnerability Analysis Using BERT-Based Models ». In : *Computers & Security* (2023). Réduction des fausses alertes avec BERT pour l'analyse statique.

[22] Mathieu Perrod. « Réseaux Neuronaux pour la Détection d'Intrusions : Potentiel et Limites en Contextes Réalistes ». In : *Revue des Sciences et Technologies de l'Information* (2023). Étude sur les réseaux neuronaux appliqués aux IDS.

[23] Priya Pitre et al. « An Intrusion Detection System for Zero-Day Attacks with Reduced False Positive Rate ». In : *International Journal of Network Security* (2022).

[24] Martin Roesch. « Snort : Lightweight intrusion detection for networks ». In : *Proceedings of the 13th USENIX conference on System administration* (1999,), p. 229-238. doi : 10.1016/b978-1-931836-04-3.x5000-0.

[25] Karen Scarfone et Peter Mell. « Guide to Intrusion Detection and Prevention Systems (IDPS) ». In : *NIST Special Publication* 800-94 (2007). doi : 10.6028/NIST.SP.800-94.

[26] Karen Scarfone, Peter Mell et al. « Guide to intrusion detection and prevention systems (idps) ». In : *NIST special publication* 800.2007 (2007), p. 94. doi : 10.6028/nist.sp.800-94.

[27] Nathan Shone et al. « A deep learning approach to network intrusion detection ». In : *IEEE Transactions on Emerging Topics in Computational Intelligence* 2.1 (2018), p. 41-50. doi : 10.1109/TETCI.2017.2762359.

[28] Aditya Kumar Shukla et al. « Reducing Intrusion False Alarms Using PSO-Based Feature Selection on the NSL-KDD Dataset ». In : *Journal of Information Security and Applications* (2023).

[29] R. Singh et al. « LLMs in the SOC : An Empirical Study of Human-AI Collaboration ». In : *arXiv* 2508.18947 (2025). doi : 10.1145/3600211.3604712.

[30] K. T. W. Teuwen et al. « Ruling the Unruly : Designing Effective, Low-Noise Network Intrusion Detection Rules ». In : *arXiv* 2501.09808 (2025). doi : 10.1145/3708821.3710823.

[31] F. Ullah et al. « IDS-INT : Intrusion detection system using transformer-based transfer learning for imbalanced network traffic ». In : *Journal/Conference (ScienceDirect entry)* (2024). doi : 10.1016/j.dcan.2023.03.008.

[32] Ashish Vaswani et al. « Attention is all you need ». In : *Advances in neural information processing systems* 30 (2017). doi : 10.48550/arXiv.1706.03762. url : https://arxiv.org/abs/1706.03762.

[33] Tian Wang et al. « Alert Graph-Based Distinction of True Incidents and False Alarms in Intrusion Detection Systems ». In : *Computers & Security* (2022). Filtration contextuelle par graphes d'alertes.

[34] B. Wei et al. « CORTEX : Collaborative LLM Agents for High-Stakes Alert Triage ». In : *arXiv* 2510.00311 (2025). doi : 10.1080/10401334.2017.1365719.

[35] Wikipédia. *SOAR (sécurité informatique)*. 2023. url : https://fr.wikipedia.org/wiki/SOAR_%28s%C3%A9curit%C3%A9_informatique%29.

[36] Binyu Yin et al. « A Hybrid Intrusion Detection Method for CBTC Systems Using Improved Stacking and False Positive Elimination Strategy ». In : *IEEE Transactions on Intelligent Transportation Systems* (2023).

[37] F. Zhang. « Network Intrusion Detection Using a Separable Convolution Autoencoder with LSTM ». In : *Digital Communications and Networks* (2024).