RÉPUBLIQUE DU BÉNIN
MINISTÈRE DE L'ENSEIGNEMENT SUPÉRIEUR
ET DE LA RECHERCHE SCIENTIFIQUE

UNIVERSITÉ D'ABOMEY-CALAVI

INSTITUT DE FORMATION ET DE RECHERCHE EN INFORMATIQUE

BP 526 Cotonou Tel : +229 21 14 19 88
http://www.ifri-uac.net Courriel : contact@ifri.uac.bj

MÉMOIRE

pour l'obtention du

Diplôme de Licence en Informatique

Option : Sécurité Informatique

Présenté par :
Emmanuel **ATTANASSO**

Approche basée sur le machine learning (ML) pour la détection proactive et la réponse aux incidents IT

Sous la supervision :
Dr. Abdou **AMBARKA**

Membres du jury :

HOUNDJI Ratheil	Docteur	IFRI	Président
GAHOU Maryse	Ingénieur	IFRI	Examinateur
AMBARKA Abdou	Docteur	IFRI	Rapporteur

Ce 30/05/2025
Ok pour dépôt.
HOUNDJI
V. Ratheil

Année Académique : **2023 - 2024**

Sommaire

Dédicace ii

Remerciements iii

Résumé iv

Abstract v

Liste des figures vi

Liste des tableaux vii

Acronymes viii

Introduction 1

1 Revue de littérature 3

2 Méthodes de détection d'anomalies 8

3 Déploiement de la solutions et test de fonctionnalité 29

Conclusion 38

Bibliographie 39

Webographie 41

Table des matières 43

Dédicace

Je dédie ce travail,

- À ma mère, Gisèle DOSSOU
- À mon père, Sébastien ATTANASSO
- À mon frère, Chrislain ATTANASSO,
- À ma sœur, Bénia ATTANASSO.
- À mes proches et à ceux qui me sont chers,

Pour vos sacrifices et votre soutien tout au long de mon cursus.

Remerciements

Je suis profondément reconnaissant envers l'Éternel Dieu, pour sa protection, l'inspiration et la force qu'il me donne tous les jours.

Je souhaite adresser mes remerciements les plus sincères à mes parents qui tout au long de ce travail, m'ont apporté leur soutien et leurs encouragements.

Mes sincères remerciements s'adressent au **Professeur Eugène EZIN** mon Directeur, à toute l'administration et au corps professoral d'IFRI pour la qualité de la formation que j'ai reçu durant ces trois années.

J'exprime ma profonde gratitude à **Mme Claude BORNA**, qui m'a accordé ce stage et offert cette précieuse opportunité.

Je remercie le **Dr. Abdou AMBARKA**, mon maître de stage, pour ses idées, sa disponibilité et son précieux accompagnement tout au long de mon mémoire.

Je remercie également **Mr Rodolphe AGANI** et **Mr Danis KIKISSAGBE**, dont l'assistance et les conseils tout au long de mon mémoire ont été d'une aide précieuse. Je remercie aussi tous les employés de l'Agence de Développement Sèmè City pour leur soutien.

Mes vifs remerciements vont également aux membres du jury qui ont accepté d'examiner mon travail et de l'enrichir par leurs observations et remarques.

Résumé

Dans un contexte où les cyberattaques deviennent de plus en plus complexes, ce mémoire présente le développement d'un outil de détection d'anomalies appliqué aux logs réseaux, basé sur le machine learning. Le système conçu collecte automatiquement les journaux d'événements générés par les équipements réseaux (pare-feux, systèmes IDS, etc.), les centralise dans un fichier unifié, puis les nettoie et les transforme pour extraire des caractéristiques exploitables. Ces données sont ensuite analysées par un modèle de machine learning, notamment l'algorithme Isolation Forest, afin de détecter des comportements anormaux.
L'approche mise en œuvre permet d'identifier plusieurs types d'attaques comme les DDoS, le scan de ports ou les injections SQL. Une interface de visualisation et un flux de réponse permettent d'alerter les analystes et de générer des actions automatiques comme le blocage d'IP ou la journalisation d'incidents. L'outil s'intègre ainsi dans une logique de sécurité proactive et contribue à alléger la charge de travail des centres opérationnels de sécurité (SOC). Ce travail combine des techniques de traitement de données, d'apprentissage automatique et de réponse aux incidents dans une base de données structurée, visant à renforcer la surveillance des réseaux et à améliorer la réactivité face aux menaces.

Mots clés : détection d'anomalies, machine learning, DDoS, injection SQL.

Abstract

In a context where cyberattacks are becoming more and more complex, this thesis presents the development of an anomaly detection tool applied to network logs, based on machine learning. The designed system automatically collects event logs generated by network equipment (firewalls, IDS systems, etc.), centralizes them into a unified file, and then cleans and transforms them to extract usable characteristics. This data is then analyzed by a machine learning model, including the Isolation Forest algorithm, to detect anomalous behavior.
The approach implemented makes it possible to identify several types of attacks such as DDoS, port scanning or SQL injections. A visualization interface and response flow allow you to alert analysts and generate automatic actions such as IP blocking or incident logging. The tool is thus integrated into a proactive security logic and helps to lighten the workload of security operations centers (SOCs). This work combines data processing, machine learning, and incident response techniques into a structured database, aimed at strengthening network monitoring and improving responsiveness to threats.

Key words: anomaly detection, machine learning, DDoS, SQL injection.

Liste des figures

1.1 HIDS 5
1.2 NIDS 6

2.1 Méthodes de détection d'anomalies 11
2.2 LOF 13
2.3 One-class Support Vector Machine 14
2.4 Exemple Isolation forest 14
2.5 Découpages aléatoires 15
2.6 La méthode des K-means 16
2.7 Cas d'utilisation 21
2.8 Diagramme de séquence du Processus de détection d'incident 22
2.9 Diagramme de séquence du Processus de réponse à un incident 23
2.10 Diagramme de classe 24
2.11 Log d'événement et logs d'attaque 25
2.12 Dataset post traitements 27
2.13 test modèle iforest 28

3.1 Article TOGBE Mauras et Lakshmi 30
3.2 Page d'accueil dashboard 33
3.3 Page d'accueil dashboard 34
3.4 Page d'affichage des anomalies 34
3.5 Page d'affichage des anomalies : logs inconnus 35
3.6 Page d'affichage des anomalies : logs d'évènements 35
3.7 Page d'affichage des alertes 35
3.8 Page d'affichage des alertes : description 36
3.9 Affichage des alertes 36
3.10 Chat 36

Liste des tableaux

2.1 Synthèse de 10 revues existantes : 1- Hodge et Austin (2004), 2- Patcha et Park (2007), 3- Chandola et al. (2009), 4- Aggarwal (2017), 5- M Gupta, Gao, Aggarwal et al. (2014), 6- Soudien et al. (2016), 7- Tellis et D'Souza (2018), 8- Salehi et Rashidi (2018), 9- J Zhang (2013), 10- Chalapathy et S Chawla (2019). 10
2.2 Comparaison des jeux de données utilisés 17
2.3 Résultats des performances des modèles sur différents jeux de données 18
2.4 Comparaison d'outils de détection d'incidents informatiques basés sur le Machine Learning 19
2.5 Exemple de log d'attaque et explication des champs clés 26
2.6 Exemple de log d'événement et description des champs 26

Acronymes

ASIN :

Agence des Systèmes d'information et du numérique viii, 1

DDOS :

Distributed Denial of Service viii, 1

EDR :

Endpoint detection and response viii, 4

HIDS :

Host based Intrusion Detection System viii, 5

IDS :

Intrusion Detection System viii, 5, 6

IPS :

Intrusion Prevention System viii, 5

NIDS :

Network Host Based Intrusion Detection System viii, 5

SIEM :

Security Information and Event Management viii, 5

SOC :

Security Operations Center viii, 1

Introduction Générale

1. contexte

Le système d'information est un élément clé de toute organisation moderne. Il peut être défini comme un ensemble de composants interconnectés visant à collecter, stocker, traiter et diffuser l'information dans une organisation. Ses objectifs vont de la facilitation de la prise de décision à l'optimisation des processus internes, en passant par l'amélioration de la communication et de la collaboration. Mais, face à l'évolution des technologies, l'intégrité des systèmes d'information se voit menacé. Il est indispensable de pouvoir surveiller ces ressources qui sont sensible en temps réel.

2. Problématique

En 2ème année, j'ai eu à faire mon stage au sein de l'Agence des Systèmes d'Information et du Numérique (ASIN), qui avec son pôle sécurité joue un rôle de centrale de traitement et de réponse aux menaces informatique de plusieurs entités du Bénin, où je travaillais en tant qu'Analyste SOC junior. A l'ASIN, il est utilisé de puissants outils propriétaires de Détection et de Réponses aux incidents sur les terminaux du marché qui permettent de très vite détecter une menace sur ces derniers et qui surpassent la performance des antivirus classiques que nous rencontrons sur nos ordinateurs. En effet, la transformation numérique amorcée il y a plusieurs décennies a amplifié les attaques informatiques dans le monde entier (Attaque de ransomware, vol de donnée, attaque DDOS, etc.) par des pirates sur des postes clients, des serveurs ou sur tous types d'actif essentiel dans une entreprise.

Etant du côté des analystes, j'ai pu remarquer le nombre très important d'alertes qui surviennent par jours, ces alertes doivent être traité afin de vérifier si oui ou non elles traduisent de réelles menaces, ce qui devient vite une tâche des plus fastidieuses pour les analystes. C'est pour cela que l'implémentation d'un outil basé sur le machine Learning de détection et de réponse contre les menaces au sein même des entités est recommandée afin de diminuer la charge de travail des analystes.

3. Organisation du document

Le travail est structuré en trois chapitres. Le chapitre 1 présente une revue de littérature concernant la problématique. Nous abordons ensuite dans le chapitre 2 la question des choix techniques ainsi que les étapes de conception de notre projet puis dans le chapitre 3 nous présenterons notre solution et les résultats obtenus.

4. Objectifs

L'objectif global de ce projet est de mettre en œuvre des techniques de machine learning pour améliorer la détection des incidents informatique grâce aux anomalies dans les logs réseau et optimiser la réponse à ces incidents. Les objectifs spécifiques de ce projet consistent à :

- Développer un modèle de machine learning pour la détection d'anomalies dans les logs réseau : Mettre en œuvre et entraîner divers algorithmes de machine learning (par exemple, K-means, Isolation Forest, etc.) pour identifier des comportements anormaux dans les logs de réseau.
- Élaborer des mécanismes de réponse aux incidents basés sur les résultats de la détection d'anomalies : Développer des scripts et des outils automatisés pour répondre rapidement et efficacement aux incidents détectés (par exemple, blocage d'adresses IP suspectes, alerte aux administrateurs système, etc.).

5. Organisation du mémoire

Ce document s'articule autour de trois chapitres. D'abord, le premier chapitre est consacré à la revue de littérature. Ensuite, le deuxième chapitre aborde la présentation des méthodes de détection d'anomalies. Enfin, le chapitre 3 aborde le déploiement et les tests de fonctionnalité de notre solution.

Chapitre 1

Revue de littérature

Introduction

La protection des réseaux informatique devient un challenge pour les entreprises au vu des dernières avancées technologiques. Déployer un outil de détection et de réponse aux menaces devient très important.

Dans ce chapitre, il s'agira de faire une clarification conceptuelle des thèmes liés au concept de machine Learning, de monitoring des systèmes d'information.

1.1 La sécurité informatique

1.1.1 Définition

La sécurité informatique protège l'intégrité des technologies de l'information comme les systèmes, les réseaux et les données informatiques contre les attaques, les dommages ou les accès non autorisés. Autrement dit, qu'elle s'assure que les exigences de sécurité soient respectées dans un environnement [18].

Les objectifs de la Sécurité Informatique visent généralement cinq principaux objectifs :

- **Intégrité** : garantir que les données soient bien celles que l'on croit être ;
- **Disponibilité** : maintenir le bon fonctionnement du système d'information ;
- **Confidentialité** : rendre l'information inintelligible à d'autres personnes que les seuls acteurs d'une transaction ;
- **Non-répudiation** : garantir qu'une transaction ne peut être niée ;
- **Authentification** : assurer que seules les personnes autorisées aient accès aux ressources.

1.1.2 Les Menaces informatiques

La sécurité informatique protège contre différents types de menaces. On distingue notamment :

- **Failles de sécurité** : Les failles de sécurité sont constituées par les défauts d'un logiciel ou de configuration de sécurité, qui fragilisent la sécurité globale du réseau.
- **Logiciels espions** : Un logiciel espion peut être téléchargé à partir de sites Web, d'e-mails, de messages instantanés et de connexions directes de partage de fichiers. Par ailleurs, un utilisateur peut, sans le savoir, recevoir un logiciel espion en acceptant un contrat de licence utilisateur final d'un programme informatique.
- **Courrier indésirable** : Le courrier indésirable consiste à recevoir des messages non sollicités, la plupart du temps de la publicité, à un grand nombre de destinataires. Le courrier indésirable pose un sérieux problème de sécurité, car il peut être utilisé pour envoyer un e-mail susceptible de contenir des chevaux de Troie, des virus, des vers, des logiciels espions et des attaques ciblées qui visent à obtenir des informations d'identification personnelles et confidentielles.
- **Programmes malveillants** : Les programmes malveillants font partie des programmes hostiles, comprenant les virus, les vers et les chevaux de Troie. Ils tentent également d'exploiter les failles existantes des systèmes en s'introduisant de façon aisée et transparente.
- **L'employé mécontent** : C'est une personne de mauvaise intention ayant eu a travaillé dans une entreprise ou le départ de ce dernier ne s'est pas bien passée dont les accès sont toujours valides. Ce genre d'individu peut représenter une menace très importante pour une entreprise, car elle peut causer des conséquences énormes d'un point de vue informationnel, financier ou autres ; on peut donc assister à des vols et vente d'informations confidentiels, des dégradations ou suppression de données importantes, injection de logiciel malveillants dans le réseau informatique de l'entreprise à distances.

1.2 Le monitoring

1.2.1 Définition

Le monitoring informatique est une des importantes fonctionnalités d'un EDR, qui est une opération consistant à faire un contrôle de chacun des équipements physiques et logiciels qui composent votre SI. L'objectif est de détecter les éventuels dysfonctionnements et de garantir son fonctionnement optimal à tout moment.

1.2.2 Objectifs du monitoring

Le monitoring informatique permet d'anticiper les défaillances et donc d'éviter les pannes en étant alerté des dysfonctionnements. La supervision a pour objectif la surveillance du système d'information. Elle permet à l'administrateur d'être au courant en temps réel de l'état du système informatique et d'agir en cas d'anomalie [19]. C'est donc pour répondre aux préoccupations suivantes :

- **Technique** : surveillance du réseau et des systèmes (elle permet d'avoir un œil sur la santé du réseau en tenant compte des éléments qui peuvent nuire à sa performance tels que les nouvelles charges et la latence), détecter les indicateurs de compromission ;

- **Applicative** : surveillance des applications et des processus métiers. (Elle permet de connaître tout ce qui est comme logiciels ou programme qui tourne en temps réel sur les points de terminaisons ou sur le réseau)

1.3 Les IDS

Un système de détection d'intrusion (IDS) est un outil de sécurité des réseaux qui surveille le trafic réseau et les appareils pour détecter les activités malveillantes connues, les activités suspectes ou les violations des politiques de sécurité. Un IDS peut aider à accélérer et à automatiser la détection des menaces sur le réseau en alertant les administrateurs de la sécurité des menaces connues ou potentielles, ou en envoyant des alertes à un outil de sécurité centralisée. Un outil de sécurité centralisée, tel qu'un système de gestion des informations et des événements de sécurité (SIEM), peut combiner des données provenant d'autres sources pour aider les équipes de sécurité à identifier et à répondre aux cybermenaces susceptibles d'échapper aux autres mesures de sécurité [20].

Un IDS ne peut pas arrêter les menaces de sécurité à lui seul. Aujourd'hui, les fonctionnalités d'IDS sont généralement intégrées aux systèmes de prévention contre les intrusions (IPS), qui peuvent détecter les menaces de sécurité et prendre automatiquement des mesures pour les éviter.

Il existe deux principaux types d'IDS, nous avons les HIDS (Host based Intrusion Detection System) et NIDS (Network Host Based Intrusion Detection System).

- **Les HIDS** : Ils analysent le fonctionnement et l'état des machines sur lesquels ils sont installés afin de détecter les attaques en se basant sur des daemons. L'intégrité des systèmes est alors vérifiée périodiquement et des alertes peuvent être levées. Par nature, ces IDS sont limités et ne peuvent détecter les attaques provenant des couches réseaux (tels que les attaques de type DOS) ;

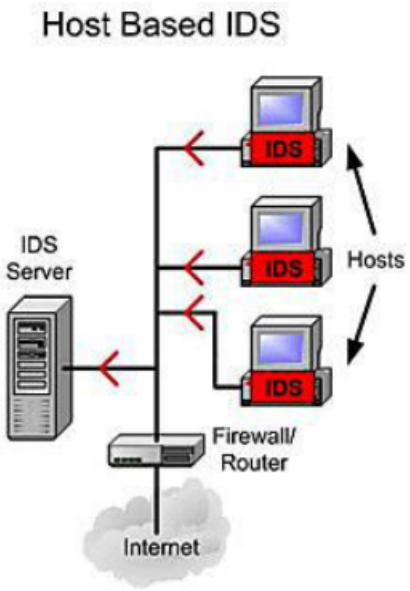


FIGURE 1.1 : HIDS

Source Figure : [1]

- **Les NIDS** : Ils analysent en temps réel les trafics entrant et sortant du réseau qu'ils aspirent à l'aide d'une sonde afin d'y déceler les attaques. Ensuite, les paquets sont décortiqués puis analysés. En cas, de détection d'intrusion, des alertes peuvent être envoyées.

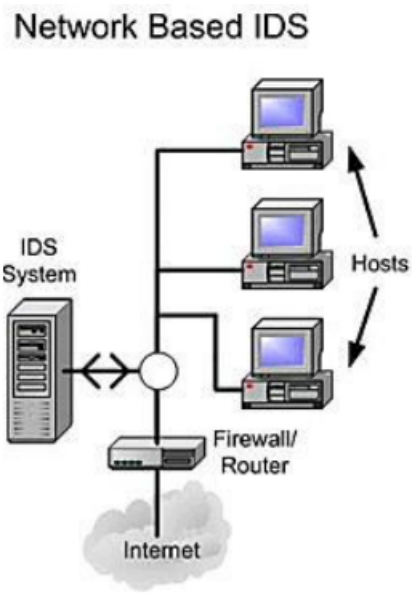


FIGURE 1.2 : NIDS

Source Figure : [2]

1.3.1 Fonctionnement des systèmes de détection d'intrusion

Les IDS peuvent être des applications logicielles installées sur des terminaux ou des unités matérielles dédiées, connectées au réseau. Certaines solutions d'IDS sont disponibles sous forme de services cloud. Quelle que soit la forme utilisée, un IDS utilise une ou deux des deux principales méthodes de détection des menaces : la **détection basée sur les signatures** ou la **détection basée sur les anomalies**.

1.3.1.1 Détection basée sur les signatures

La détection basée sur les signatures analyse les paquets réseau pour détecter les caractéristiques ou comportements d'attaque propres à une menace spécifique. Une séquence de code apparaissant dans une variante particulière de logiciel malveillant est un exemple de signature d'attaque.

Un IDS basé sur les signatures maintient une base de données de signatures d'attaque avec laquelle il compare les paquets réseau. Si un paquet déclenche une correspondance avec l'une des signatures, l'IDS le signale. Pour être efficaces, les bases de données de signatures doivent être régulièrement mises à jour avec de nouveaux renseignements sur les menaces à mesure que de nouvelles cyberattaques émergent et que les attaques existantes évoluent. Les toutes nouvelles attaques qui n'ont pas encore été analysées pour les signatures peuvent échapper à un IDS basé sur les signatures.

[1] https://www-igm.univ-mlv.fr/~dr/XPOSE2004/IDS/IDSPres.html
[2] https://www-igm.univ-mlv.fr/~dr/XPOSE2004/IDS/IDSPres.html

1.3.1.2 Détection basée sur les anomalies

Les méthodes de détection basées sur les anomalies font appel au machine Learning pour créer et affiner en permanence un modèle de référence de l'activité réseau normale. Elles comparent ensuite l'activité réseau au modèle et signalent les écarts, comme un processus qui utilise plus de bande passante que d'habitude ou un appareil qui ouvre un port.

Étant donné qu'ils signalent tout comportement anormal, les IDS basés sur les anomalies peuvent souvent détecter de nouvelles cyberattaques susceptibles d'échapper à la détection basée sur les signatures. Par exemple, les IDS basés sur les anomalies peuvent détecter des attaques « **zero-day** », c'est-à-dire des attaques qui exploitent les vulnérabilités des logiciels avant que le développeur ne s'en aperçoive ou n'ait eu le temps d'appliquer un correctif.

Toutefois, les IDS basés sur les anomalies peuvent également être plus sujets aux faux positifs. Même une activité bénigne, comme un utilisateur autorisé accédant à une ressource réseau sensible pour la première fois, peut déclencher un IPS basé sur des anomalies.

Chapitre 2

Méthodes de détection d'anomalies

Introduction

La détection d'anomalies est un sujet qui intéresse de plus en plus de chercheurs de nos jours. Dans la littérature, on trouve plusieurs définitions des anomalies encore appelée **outlier**. Dans **'Identification of outliers'** [1] en 1980, un **outlier** est défini par **Douglas M. Hawkins** comme une observation qui dévie considérablement du reste des autres observations comme si elle était générée par un processus différent. Dans ses travaux, **Aristote (384-322 av. J.-C.)** utilisait le terme d'anomalie pour décrire des déviations par rapport à la norme ou à l'ordre naturel. Pour lui, une **anomalie** était quelque chose qui ne suivait pas les règles habituelles de la nature. La détection d'anomalies peut servir à l'amélioration de la qualité des données par suppression ou remplacement des données anormales. L'exploitation de données sans anomalies permet d'aboutir à des résultats et des décisions plus sûres pour plusieurs applications. Elle trouve son application dans plusieurs autres domaines comme la santé, la cybersécurité, la finance, la prédiction des catastrophes naturelles, et bien d'autres domaines.

2.1 Etat de l'art des méthodes

La détection d'anomalies est un sujet qui a fait l'objet de plusieurs travaux. La revue de Hodge et Austin (2004)[2] fournit une vue d'ensemble complète des méthodologies de détection des anomalies et souligne l'importance de choisir la méthode appropriée en fonction du contexte et des spécificités des données. Patcha et Park (2007)[3] ont réalisé une revue des méthodes utilisées pour la détection d'intrusion. Une revue plus générale des techniques existantes couvrant plusieurs approches est proposée dans Aggarwal (2017)[4] et Chandola et al. (2009)[5]. M Gupta, Gao, Aggarwal et al. (2014)[6] fait l'état de l'art des méthodes en fonction du type de données considérées : les données temporelles telles que les séries temporelles, les données spatio-temporelles et les flux de données. Salehi et Rashidi (2018)[9], Souiden et al. (2016)[10], J. Zhang (2013)[11], Tellis et D'Souza (2018)[12] présentent également des méthodes applicables aux flux de données. La revue de Chalapathy et Chawla (2019)[13] offre un aperçu complet des applications de l'apprentissage profond pour la détection des anomalies. Dans le Tableau 2.1, nous présentons une synthèse qui s'appuie sur 10 revues majeures dans la littérature. Nous identifions respectivement les techniques de détection d'anomalies, les types de jeux de données et les domaines d'application abordés dans chacune de ces 10 revues.

Le but de cette section est de fournir un état de l'art complet en agrégeant plusieurs informations sur les différentes méthodes de détection d'anomalies, les jeux de données et les domaines d'applications.

L'émergence des nouvelles technologies a mené à la génération de différents types de données. On trouve plusieurs jeux de données ayant différentes caractéristiques et apportant de nouveaux challenges dans la détection d'anomalies. Nous présentons dans le Tableau 2.1 des types de jeux de données abordés dans les 10 revues étudiées. Différents algorithmes de détection d'anomalies sont appliqués en fonction du type de jeux de données considéré [21].

2.1.1 Domaines d'application

La détection d'anomalies est présente dans tout domaine qui exploite les données. Ainsi, elle a de nombreuses applications possibles. Les domaines d'application ayant leur spécificité en fonction des données générées ou exploitées, toutes les méthodes de la détection d'anomalies ne sont pas adaptées à tous les domaines d'application. Chandola et al. (2009)[5] et Aggarwal (2017)[4] présentent un état de l'art couvrant plusieurs domaines d'application. Dans M Gupta, Gao, Aggarwal et al. (2014)[6], les auteurs ont fait la revue des méthodes de détection d'anomalies temporelles applicables dans différents domaines. La détection d'intrusion, décrite dans Chandola et al. (2009)[5], se base sur l'analyse d'une cible, généralement un réseau ou un hôte, pour détecter les comportements anormaux. Il s'agit en effet de tentatives frauduleuses d'accès à une ressource par violation de la sécurité mise en place pour la cible en question Aggarwal (2017)[4]. La détection de fraude permet de repérer les activités suspectes menées par un individu généralement sous une fausse identité (identité usurpée). Le Tableau 2.1 liste plusieurs revues ayant abordé différents domaines d'application tels que la détection d'intrusion, la détection de fraude, la santé, la maintenance prédictive dans l'industrie, les réseaux de capteurs, le traitement d'images, le traitement de texte, les données biologiques, l'astronomie et l'économie [21]. La détection d'anomalies dans les données textuelles (comme les logs) intéresse de nombreux chercheurs surtout avec l'utilisation accrue des réseaux sociaux. Différentes méthodes existent pour la détection d'anomalies dans les données textuelles. Après avoir fait un état de l'art détaillé des méthodes existantes, Mohotti et Nayak proposent dans Mohotti et Nayak (2020)[14] une méthode qui se base sur la fréquence et le rang des mots pour détecter les anomalies. Le tableau 2.1 présente différentes techniques de détection d'anomalies et les références associées. Il inclut :

- **Techniques :** Statistique, Clustering, Méthode des plus proches voisins, Classification, Régression, Approche spectrale, Motifs fréquents, Deep learning.
- **Type de jeux de données :** Flux de données, Séries temporelles, Graphes, Grande dimension, Séquentielles, Spatio-temporelles, Spatiales.
- **Domaines d'application :** Détection d'intrusion, Détection de fraude, Santé, Maintenance prédictive, Réseaux de capteurs, Traitement d'images, Traitement de texte, Données biologiques, Astronomie, Économie.

Chaque méthode est associée à différentes références bibliographiques.

Techniques	**Références**									
	1	2	3	4	5	6	7	8	9	10
Statistique	✓	✓						✓		
Clustering	✓		✓		✓	✓			✓	
Plus proches voisins		✓	✓	✓				✓		
Classification	✓		✓			✓		✓		
Régression				✓		✓			✓	
Approche spectrale			✓			✓			✓	
Motifs fréquents			✓							
Deep learning									✓	✓
Type de jeux de données	**Références**									
	1	2	3	4	5	6	7	8	9	10
Flux de données	✓	✓			✓	✓	✓		✓	✓
Séries temporelles	✓					✓		✓		
Graphes										✓
Grande dimension		✓					✓		✓	✓
Séquentielles			✓						✓	
Spatio-temporelles									✓	✓
Spatiales										✓
Domaines d'application	**Références**									
	1	2	3	4	5	6	7	8	9	10
Détection d'intrusion	✓		✓		✓	✓	✓		✓	
Détection de fraude		✓		✓	✓			✓		✓
Santé								✓	✓	
Maintenance prédictive								✓	✓	
Réseaux de capteurs								✓	✓	
Traitement d'images										✓
Traitement de texte									✓	
Données biologiques								✓	✓	
Astronomie								✓		
Économie									✓	✓

TABLE 2.1 : Synthèse de 10 revues existantes : 1- Hodge et Austin (2004), 2- Patcha et Park (2007), 3- Chandola et al. (2009), 4- Aggarwal (2017), 5- M Gupta, Gao, Aggarwal et al. (2014), 6- Soudien et al. (2016), 7- Tellis et D'Souza (2018), 8- Salehi et Rashidi (2018), 9- J Zhang (2013), 10- Chalapathy et S Chawla (2019).

2.1.2 Techniques de détection d'anomalies

Les techniques de détection d'anomalies existantes se basent sur deux propriétés importantes d'anomalies : elles sont rares et elles présentent un comportement très différent de la majorité des données. Parmi ces techniques, nous distinguons : les méthodes statistiques, les méthodes du machine Learning classique, celles du deep learning et d'autres méthodes [21].

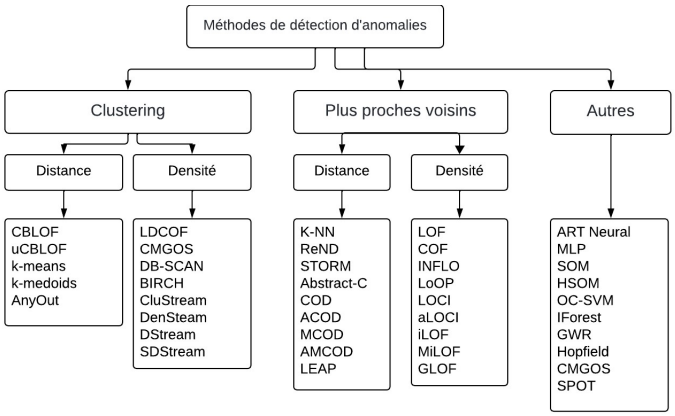


FIGURE 2.1 : Méthodes de détection d'anomalies

Source Figure et tableau : [1]

2.1.2.1 Méthodes statistiques

Les **méthodes statistiques** pour la détection d'anomalies sont parmi les plus anciennes et les plus fondamentales techniques utilisées pour identifier des points de données qui s'écartent significativement du comportement attendu. Ces méthodes reposent sur l'idée que les anomalies sont des points de données qui sont peu probables selon un modèle statistique des données normales. Elles peuvent être paramétriques ou non paramétriques. A la différence de l'approche non paramétrique, l'approche paramétrique suppose une connaissance a priori de la distribution des données. Elles utilisent des paramètres estimés à partir des données pour définir un modèle, et les anomalies sont identifiées comme des observations qui dévient de ce modèle. Les méthodes non-paramétriques quant à elles ne supposent pas que les données suivent une distribution spécifique, ce qui les rend plus flexibles pour détecter des anomalies dans des ensembles de données réels qui ne suivent pas de distribution prédéfinie

2.1.2.2 Deep Learning

Elles représentent une classe d'algorithmes d'apprentissage automatique supervisé, semi-supervisé ou non supervisé basés sur l'utilisation de plusieurs couches d'unité de traitement non linéaire. Parmi ces méthodes, on peut citer celles basées sur les auto-encoders (AE), Recurrent Neural Network (RNN) et Long short-term memory (LSTM). Les méthodes de deep Learning sont aussi performantes pour la détection d'anomalies bien qu'elles nécessitent une grande disponibilité de ressources (mémoire, puissance de calcul, temps pour traiter les données). Plusieurs Framework et outils sont conçus pour faciliter l'utilisation de ces méthodes. Les méthodes de deep Learning seront donc privilégiées pour des jeux de données de très grande dimension avec à disposition les ressources nécessaires.

2.1.2.3 Machine Learning

Le **Machine Learning (ML)** est un sous-domaine de l'**intelligence artificielle (IA)** qui permet aux systèmes informatiques d'apprendre et de s'améliorer automatiquement à partir de l'expérience sans être explicitement programmés. Le ML se base sur la création d'algorithmes capables de traiter de

[1] https://theses.hal.science/tel-03966951v1/file/TOGBE_Maurras_Ulbricht_2022.pdf

grandes quantités de données, d'en extraire des modèles, et d'utiliser ces modèles pour faire des prédictions ou prendre des décisions. La différence entre le deep Learning et le machine Learning réside dans la manière dont chaque algorithme apprend. Le machine Learning « **profond** » (ou deep Learning) peut s'appuyer sur des jeux de données étiquetés, également connus sous le nom d'apprentissage supervisé, pour informer son algorithme, mais il n'a pas nécessairement besoin de ce jeu de données. Le processus d'apprentissage profond peut ingérer des données non structurées sous leur forme brute (comme du texte ou des images) et déterminer automatiquement l'ensemble des fonctionnalités qui distinguent différentes catégories de données les unes des autres. Cela permet d'éliminer une partie de l'intervention humaine nécessaire et d'utiliser des ensembles de données plus volumineux. On peut considérer le deep Learning comme un « machine Learning évolutif »

Les modèles de machine Learning se répartissent en trois catégories principales.

- **Machine Learning supervisé** Les techniques d'apprentissage supervisé utilisent des données d'entrée et de sortie du monde réel pour détecter les anomalies. Ces types de systèmes de détection d'anomalies nécessitent qu'un analyste de données étiquette les points de données comme normaux ou anormaux pour être utilisés comme données d'entraînement. Un modèle de machine Learning entraîné avec des données étiquetées sera capable de détecter les valeurs aberrantes en fonction des exemples qui lui sont donnés. Ce type d'apprentissage automatique est utile pour la détection des valeurs aberrantes connues, mais n'est pas capable de découvrir des anomalies inconnues ou de prédire des problèmes futurs.

- **Machine Learning non supervisé**
  Les techniques d'apprentissage non supervisé ne nécessitent pas de données étiquetées et peuvent gérer des ensembles de données plus complexes. L'apprentissage non supervisé s'appuie sur l'apprentissage profond et les réseaux neuronaux ou les encodeurs automatiques qui imitent la façon dont les neurones biologiques se signalent. Ces puissants outils peuvent trouver des modèles à partir des données d'entrée et formuler des hypothèses sur les données perçues comme normales. Ces techniques peuvent contribuer grandement à la découverte d'anomalies inconnues et à la réduction du travail de tri manuel de grands ensembles de données. Cependant, les data scientistes doivent surveiller les résultats recueillis grâce à l'apprentissage non supervisé. Étant donné que ces techniques émettent des hypothèses sur les données saisies, il leur est possible d'étiqueter les anomalies de manière incorrecte.

- **Machine Learning semi-supervisé**
  Les méthodes de détection d'anomalies semi-supervisées combinent les avantages des deux méthodes précédentes. Les ingénieurs peuvent appliquer des méthodes d'apprentissage non supervisé pour automatiser l'apprentissage des fonctionnalités et travailler avec des données non structurées. Cependant, en le combinant avec une supervision humaine, ils ont la possibilité de surveiller et de contrôler le type de modèles que le modèle apprend. Cela permet généralement de rendre les prédictions du modèle plus précises.

2.1.2.4 Local Outlier Factor (LOF)

L'idée principale du LOF est de comparer la densité de chaque point avec celle de ses voisins proches. Si un point est dans une région moins dense que ses voisins, il est considéré comme une anomalie. Pour chaque point dans le jeu de données, on trouve ses k-voisins les plus proches. Les k-voisins

sont simplement les k Points les plus proches de ce point. La distance de portée entre un point et son voisin est la distance entre eux ou la distance au k-ième voisin du voisin, selon la plus grande des deux distances. Cela aide à éviter les effets des points très proches. On calcule la densité locale pour chaque point. La densité locale est inversement proportionnelle à la moyenne des distances de portée de ce point à ses k-voisins. En d'autres termes, un point dans une région dense aura une densité locale élevée, et un point dans une région moins dense aura une densité locale faible. Le score LOF d'un point est une comparaison de sa densité locale avec celle de ses voisins. Si la densité locale d'un point est beaucoup plus faible que celle de ses voisins, le score LOF sera élevé, indiquant que ce point est une anomalie [22].

Score LOF ≈ 1 : La densité locale du point est similaire à celle de ses voisins. Ce point n'est pas une anomalie.

Score LOF > 1 : La densité locale du point est inférieure à celle de ses voisins. Plus le score est élevé, plus le point est considéré comme une anomalie.

Score LOF < 1 : (rare) La densité locale du point est supérieure à celle de ses voisins. Cela peut indiquer que ce point est dans une région plus dense que ses voisins immédiats.

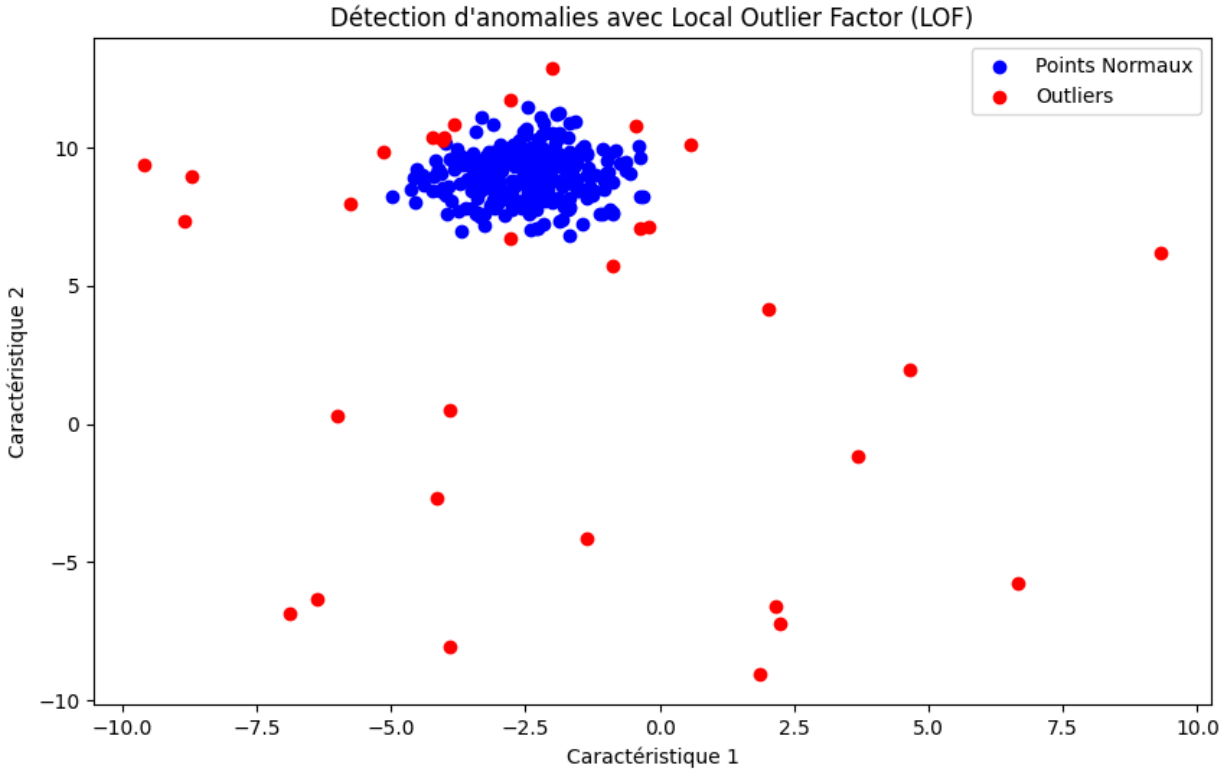


FIGURE 2.2 : LOF

2.1.2.5 One-class Support Vector Machine (OC-SVM)

Cette technique de détection d'anomalies utilise des données d'apprentissage pour définir des limites autour de ce qui est considéré comme normal. Les points groupés dans les limites définies sont considérés comme normaux et ceux situés à l'extérieur sont étiquetés comme des anomalies [23].

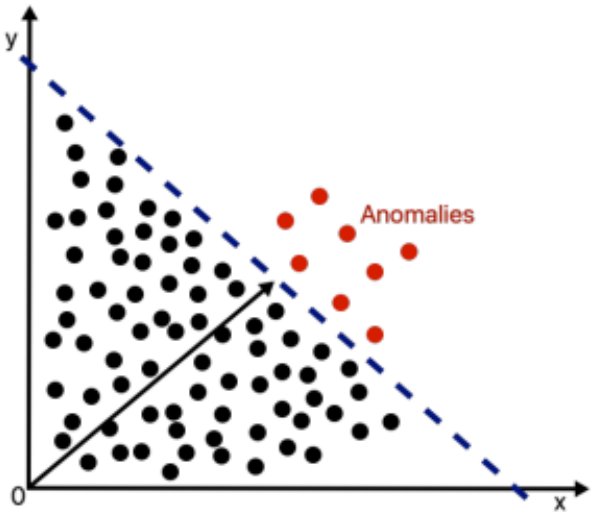


FIGURE 2.3 : One-class Support Vector Machine

Source Figure 2.3 : [2]

2.1.2.6 Forêt d'isolement (Isolation forest)

Ce type d'algorithme de détection d'anomalies utilise des données non supervisées. Contrairement aux techniques de détection d'anomalies supervisées, qui fonctionnent à partir de points de données normaux étiquetés, cette technique tente d'isoler les anomalies dans un premier temps [23].

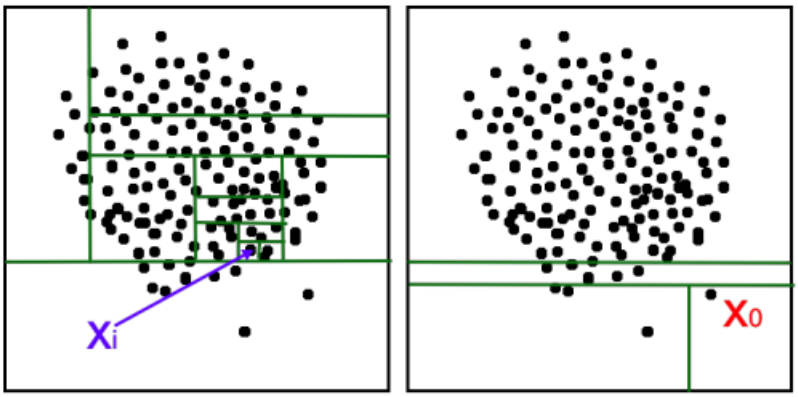


FIGURE 2.4 : Exemple Isolation forest

Source Figure 2.4 : [3]

En pratique, l'algorithme Isolation Forest réalise une série de splits (découpages aléatoires) du Data Set et compte le nombre de découpages nécessaires pour isoler des individus. L'algorithme construit dans le même temps un arbre ou chaque individu est au bout d'une des branches de cet arbre. Plus le nombre de découpages est réduit et plus il y a de chances que cet individu constitue réellement une anomalie par rapport au reste de la population du Data Set. Inversement, un grand nombre de découpage nécessaire pour isoler un individu laisse penser que cet individu n'est pas une anomalie mais est proche d'un grand nombre d'autres individus. Ce processus est répété et chaque point reçoit

[2] https://theses.hal.science/tel-03966951v1/file/TOGBE_Maurras_Ulbricht_2022.pdf
[3] https://theses.hal.science/tel-03966951v1/file/TOGBE_Maurras_Ulbricht_2022.pdf

un score d'anomalie compris entre 0 et 1, en fonction de sa position par rapport aux autres points ; les valeurs inférieures à 0,5 sont généralement considérées comme normales, tandis que les valeurs qui dépassent ce seuil sont plus susceptibles d'être anormales.

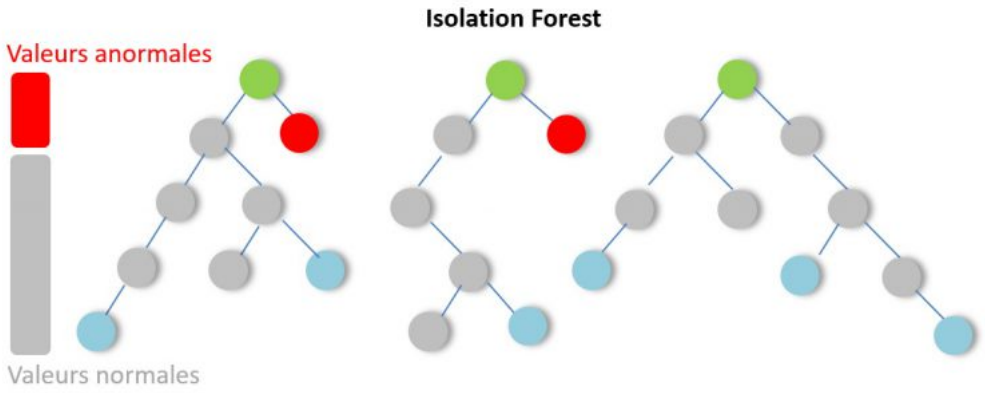


FIGURE 2.5 : Découpages aléatoires

Source Figure 2.5 : [4]

2.1.2.7 La méthode des K-means

K-means est l'un des algorithmes de clustering les plus utilisés. Dans cette méthode, les points de données sont assignés à K groupes. Dans ce cas, K représente le nombre de groupes créés en fonction de la distance entre le noyau de chaque groupe. Ce noyau est aussi appelé le centroïde et est soit choisi au hasard, soit spécifié par le data scientiste, en fonction des données. Une fois que le nombre de groupes (K) et les centroïdes ont été identifiés, le modèle assigne chaque nouveau point au noyau le plus proche et le groupe dans le cluster correspondant. La méthode la plus répandue pour calculer la distance entre un point et un noyau est le carré de la distance Euclidienne. L'un des principaux défis avec la méthode des K-means est le fait que le nombre de clusters doit être spécifié avant le début du modèle. Ceci peut être compliqué par moment, surtout lorsque la quantité de données est très importante. Si une grande valeur pour K est choisie, le modèle rendra de plus petits groupes. A l'inverse, si K prend une petite valeur, alors le modèle rendra des grands groupes. Cette méthode est la plus souvent utilisée pour la classification de documents, la segmentation d'images et la segmentation marketing ce qui n'est pas d'une très grande utilité dans notre cas particulier [24].

[4] https://metalblog.ctif.com/2022/10/03/la-detection-danomalies-en-machine-learning-non-supervise/

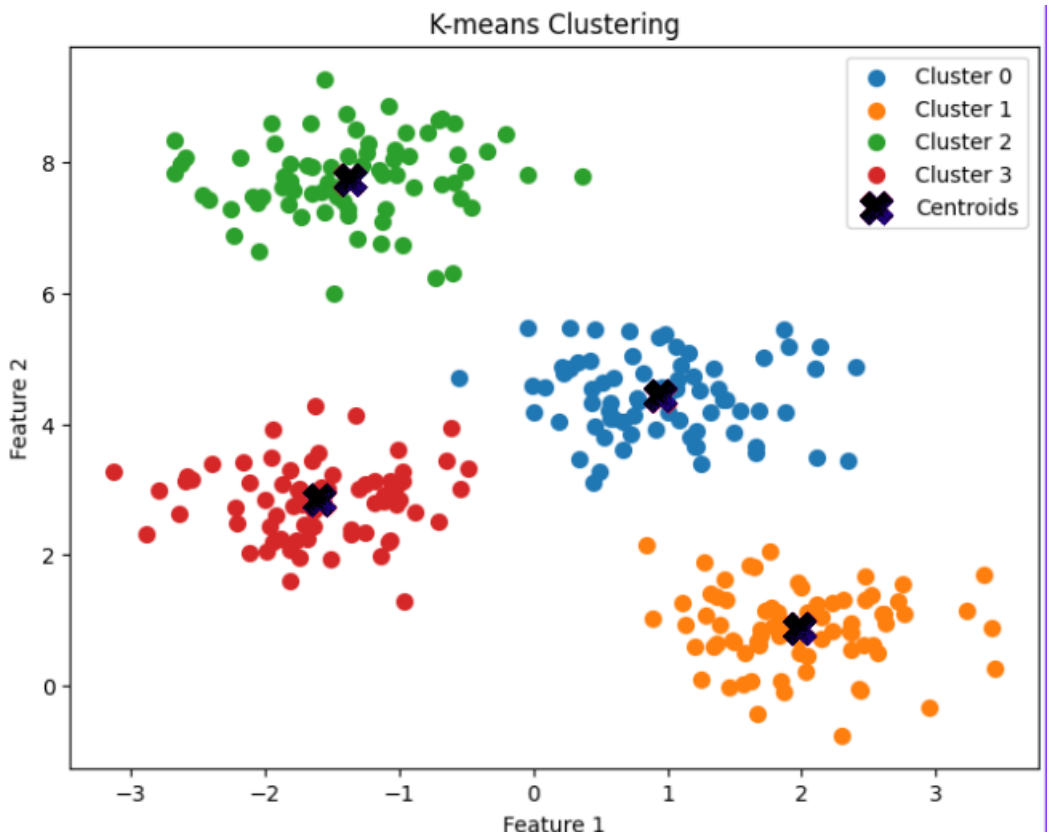


FIGURE 2.6 : La méthode des K-means

2.1.3 Choix du modèle

Le modèle de détection mis en œuvre repose sur une approche non supervisée, adaptée aux environnements où les données disponibles (logs réseaux) ne sont pas labellisées. Pour évaluer la pertinence du modèle sans compromettre sa nature non supervisée, une méthode **d'évaluation supervisée a posteriori** a été utilisée : un jeu de données annoté manuellement a permis de mesurer la capacité du modèle à identifier les anomalies connues. Cette méthode est largement reconnue dans la littérature scientifique. Elle est notamment utilisée dans des benchmarks classiques comme le KDD Cup 1999 (Stolfo et al., 2000) [7] ou UNSW-NB15 (Moustafa et Slay, 2015) [8], où des modèles non supervisés sont évalués à l'aide de labels ajoutés après l'apprentissage. De même, dans les travaux de Chandola et al. (2009) [5] sur la détection d'anomalies, il est explicitement recommandé d'évaluer les méthodes non supervisées en comparant leurs résultats avec des vérités terrain disponibles. Aussi, Mejri et al. (2022) [16] ont mené une évaluation approfondie de méthodes de détection d'anomalies non supervisées sur des séries temporelles, en utilisant des données annotées pour mesurer des métriques telles que la précision, le rappel et le score F1. De même, Idan (2024) [17] souligne les défis de la validation des modèles non supervisés et propose des approches pour évaluer ces modèles en l'absence de labels, tout en reconnaissant l'importance des évaluations supervisées lorsque des annotations sont disponibles. Ainsi, l'approche adoptée dans ce travail respecte les pratiques méthodologiques éprouvées et garantit une évaluation rigoureuse de la performance du système de détection.

2.1.4 Etude comparative de quelques méthodes

A ce niveau, nous allons comparer de façon expérimentale trois méthodes de détection d'anomalies : LOF, OC-SVM et IForest. En effet, il s'agit de méthodes performantes appartenant à des catégories

différentes. Ces trois méthodes donnent de très bons résultats comparés aux autres méthodes de leurs approches respectives. De plus, elles sont souvent utilisées comme références de comparaison pour évaluer les performances des nouvelles méthodes de détection d'anomalies [21].
Afin d'évaluer les performances des méthodes de détection d'anomalies utilisées dans ce travail, nous avons opté pour un jeu de données directement issu des journaux traités par notre plateforme. Ce jeu de données représente un échantillon réaliste du contexte d'application, constitué de 1328 entrées de logs, parmi lesquelles 151 ont été identifiées comme des anomalies connues. Contrairement à l'approche classique reposant sur des jeux publics comme KDD-Cup99 ou Shuttle, nous avons choisi de baser notre évaluation sur des données spécifiques à notre environnement, ce qui renforce la pertinence et la fiabilité des résultats obtenus. Chaque observation dans ce jeu contient des attributs techniques tels que l'adresse IP source, les ports, les types d'attaques, les niveaux de sévérité ou encore les politiques de sécurité. Ces données ont été utilisées pour comparer la performance de différents algorithmes de détection non supervisée (Isolation Forest, Local Outlier Factor, One-Class SVM) en termes de rappel, spécificité et score AUC.

Jeu de données	**Taille (observations)**	**Anomalies (observations)**
Jeu de donnée	1328	151

TABLE 2.2 : Comparaison des jeux de données utilisés

2.1.5 Résultats

Plusieurs critères peuvent être considérés pour comparer les performances des méthodes de détection d'anomalies. Pour qualifier la précision de la détection, nous nous intéressons aux trois (3) mesures suivantes :

- L'aire sous la courbe ROC (ROC AUC) qui est un standard dans la comparaison des performances des méthodes de détection d'anomalies,
- La Spécificité : Spécificité = $$\frac{VN}{VN + FP}$$ qui représente la proportion de données normales correctement identifiées.
- Le Rappel : Rappel = $$\frac{VP}{VP + FN}$$ représente la proportion d'anomalies correctement détectées.

Dans ce contexte, les positifs représentent les données anormales et les négatifs représentent les données normales. Nous avons également mesuré le temps d'exécution qui dépend à la fois de la complexité de l'algorithme et de la taille du jeu de données considéré. Le tableau résume les résultats des 3 méthodes sur les deux jeux de données considérés. *Source tableau 2.2 2.3* : [5]

IForest a une durée d'exécution plus petite que LOF et OC-SVM pour les deux jeux de données de tailles et de dimensions différentes. Cela s'explique par le fait que IForest ne calcule ni la distance entre les points ni la densité des points. IForest détecte un grand nombre d'anomalies avec beaucoup moins de fausses alertes que OC-SVM qui dans cet exercice est dépassé par LOF. LOF laisse passer

[5] https://theses.hal.science/tel-03966951v1/file/TOGBE_Maurras_Ulbricht_2022.pdf

Jeu de données	Rappel	Spécificité	ROC AUC	CPU Time (s)
IForest	**0.98**	**0.86**	**0.93**	**2.86**
LOF	0.28	0.29	0.14	21.6
OC-SVM)	0.91	0.71	0.83	815.74

TABLE 2.3 : Résultats des performances des modèles sur différents jeux de données

beaucoup d'anomalies mais fait beaucoup moins de fausses alertes que OC-SVM . Il faut noter également qu'en terme de temps d'exécution, LOF est beaucoup plus rapide que OC-SVM. En général, les performances de toutes ces méthodes régressent (mais pas de la même manière) au fur et à mesure que la taille du jeu de données et ses dimensions augmentent. OC-SVM est adaptée aux jeux de données complexes de grandes dimensions, mais n'est pas adaptée aux jeux de données de grande taille. LOF est adaptée à la détection d'anomalies locales, car l'observation n'est comparée qu'à ses k plus proches voisins sans avoir de vision globale sur la totalité des données. Le mode de fonctionnement d'Isolation Forest (pas de calcul de distance, pas de calcul de densité ni recherche de modèle) fait que le traitement est rapide même pour un jeu de données de grande taille ou de grande dimension. La contribution de différents arbres indépendants dans la prise de décision fournit également une très bonne précision dans la détection.
L'étude comparative que nous avons menée a montré que IForest est plus performant que LOF et OC-SVM en termes de détection et de durée d'exécution sur deux jeux de données de différentes tailles et dimensions. La détection d'anomalies dans les flux de données fait face à des défis liés aux caractéristiques mêmes des flux de données. Cependant, Isolation Forest (IForest) présente des performances intéressantes pour cette tâche, avec une complexité linéaire et une rapidité d'exécution qui en font une méthode particulièrement adaptée au contexte des flux de données.

2.1.6 Justification du choix final du modèle Isolation Forest :

Parmi les algorithmes testés, le modèle **Isolation Forest** a été retenu comme solution finale pour plusieurs raisons. Tout d'abord, il présente une excellente capacité de détection d'anomalies dans des jeux de données non labellisés, ce qui correspond parfaitement au contexte des logs réseau traités dans notre projet. Il est particulièrement adapté pour détecter des points rares sans nécessiter de modèle statistique global, contrairement à OC-SVM ou LOF qui sont plus sensibles aux choix de paramètres.

De plus, l'algorithme Isolation Forest est **rapide, scalable** et nécessite moins de mémoire, ce qui en fait une option idéale pour un déploiement en environnement réel ou en temps réel. Lors des tests, il a obtenu un bon compromis entre rappel et spécificité, avec une courbe ROC-AUC compétitive. Enfin, sa simplicité d'implémentation avec Scikit-learn et sa capacité à traiter efficacement les valeurs extrêmes justifient son adoption dans notre pipeline de détection automatique.

2.1.7 État de l'art des plateformes de détection d'anomalies existantes

Dans le cadre de la détection d'incidents de sécurité, plusieurs outils intègrent aujourd'hui des techniques de machine learning pour améliorer la précision de détection et réduire les faux positifs. Le tableau ci-dessous présente une comparaison de quelques solutions populaires, en mettant en évidence leur description, leurs avantages ainsi que leurs principales limitations.

TABLE 2.4 : Comparaison d'outils de détection d'incidents informatiques basés sur le Machine Learning

Outil	Description	Avantages	Limites
Splunk (avec MLTK)	Plateforme SIEM populaire intégrant un Toolkit ML pour analyser les logs en temps réel.	Visualisation avancée, intégration de modèles ML, forte compatibilité avec systèmes tiers.	Coût élevé, complexité de déploiement, dépendance à la qualité des logs collectés.
ELK Stack (ElasticSearch, Logstash, Kibana) + ML	Stack open-source pour l'analyse de logs ; offre des modules ML via X-Pack pour détecter des anomalies.	Open-source, hautement personnalisable, bonne scalabilité.	Requiert configuration avancée, ML disponible seulement via abonnement (X-Pack).
Wazuh (fork de OSSEC)	Système de détection d'intrusion (HIDS) open-source intégrant règles, détection ML via intégration avec ELK.	Intégration native avec ELK, gestion centralisée des agents, open-source.	Détection ML dépendante de l'intégration externe, pas de moteur ML natif.
Darktrace	Outil propriétaire basé sur l'IA, utilisant des algorithmes non supervisés pour détecter des comportements anormaux sur le réseau.	Très performant, visuels intuitifs, autonome, auto-apprenant.	Très coûteux, opaque (black-box), peu personnalisable.
Snort + ML Wrapper	IDS open-source, couplé à un module externe utilisant le ML pour détecter des comportements anormaux.	Communauté large, open-source, détection en temps réel.	L'ajout du ML demande des compétences avancées, performances dépendantes du wrapper.

Comme le montre le tableau 2.4, plusieurs solutions de détection d'incidents informatiques basées sur le Machine Learning existent sur le marché. Toutefois, ces solutions présentent généralement des limites importantes telles que leur coût élevé, leur complexité de déploiement, ou encore leur manque de transparence dans les processus de détection.

Afin de répondre à ces limitations et d'offrir une alternative plus accessible, ce travail propose le développement d'une **plateforme web de détection d'anomalies et de gestion des incidents**, conçue spécifiquement pour fonctionner avec des logs générés par des équipements réseau tels que les pare-feu Fortinet. Cette solution a été pensée pour être à la fois :

- **Accessible** : via une interface web intuitive, consultable en temps réel depuis un navigateur.
- **Automatisée** : elle intègre un pipeline de traitement automatique des logs, basé sur des modèles d'apprentissage automatique.
- **Interprétable** : les résultats sont affichés de manière lisible et actionnable, avec la possibilité de bloquer automatiquement ou manuellement les adresses IP malveillantes.

- **Légère et adaptable** : cette solution ne nécessite pas d'infrastructure lourde, et peut être personnalisée pour différents environnements réseaux.

Dans les sections suivantes, nous présenterons en détail le fonctionnement de cette plateforme, son architecture logicielle, ainsi que les performances des modèles utilisés sur les données collectées lors du stage.

2.2 Modélisation

Afin de modéliser les fonctionnalités de notre plateforme, nous avons porté notre choix sur le langage UML, un langage de modélisation visuelle standard destiné à être utilisé pour spécifier, visualiser et documenter des modèles de systèmes logiciels, y compris leur structure et leur conception de manière à répondre à toutes ses exigences.

Identification des acteurs

- **Administrateur** : L'utilisateur qui interagit avec le Dashboard et prend des actions sur les alertes.
- **Firewall** : Système qui reçoit les actions et applique les décisions de blocage ou autorisation.
- **Système de fichier** : Système qui représente le processus automatisé (script, modèle).

2.2.1 Diagramme de cas d'utilisation

Le diagramme de cas d'utilisation est un diagramme UML qui résume la façon dont les utilisateurs interagissent avec un système. Il permet de recueillir, d'analyser et d'organiser les besoins et de recenser les grandes fonctionnalités d'un système. Grâce au diagramme de cas d'utilisation, nous aurons une vue globale du comportement fonctionnel du système. Trois acteurs interagissent dans le système : l'administrateur (ici l'analyste), le Firewall, le pipeline. La figure illustre le diagramme de cas d'utilisation.

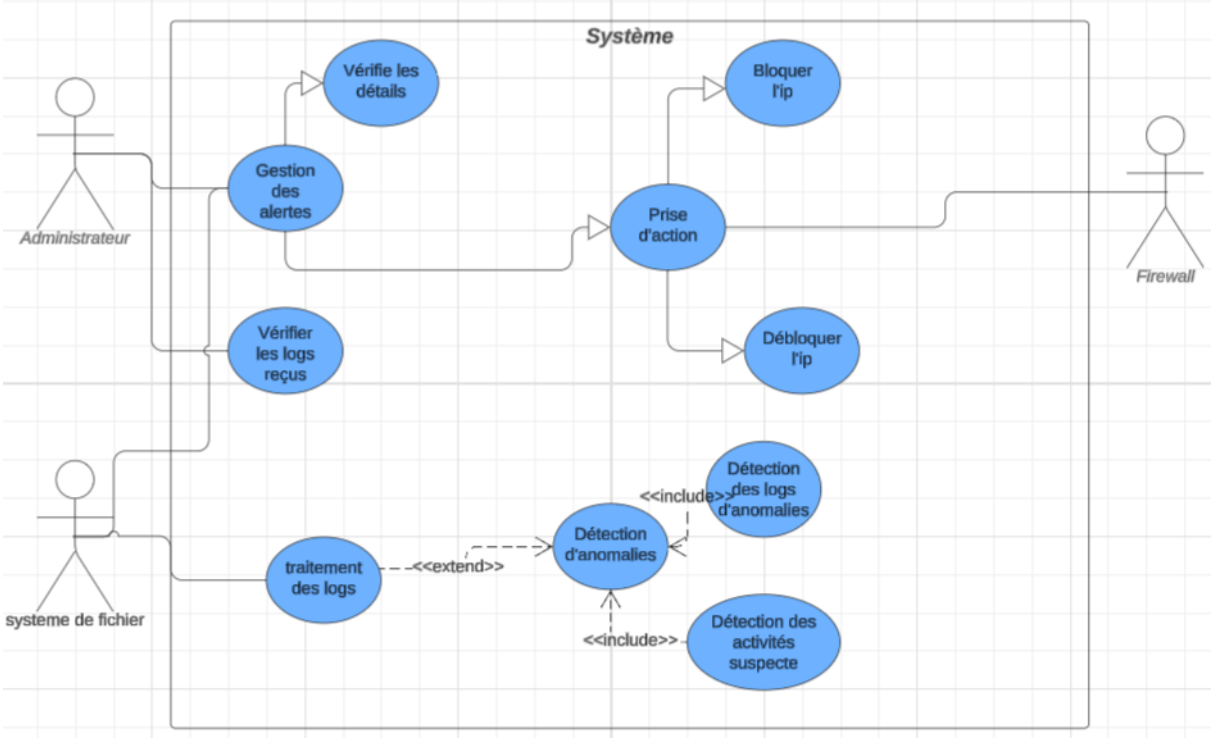


FIGURE 2.7 : Cas d'utilisation

2.2.2 Diagramme de séquence

Les diagrammes de séquence sont la représentation graphique des interactions entre les acteurs et le système selon un ordre chronologique dans la formulation UML. Ici, nous allons illustrer les diagrammes de séquence de plusieurs interactions dans notre système.

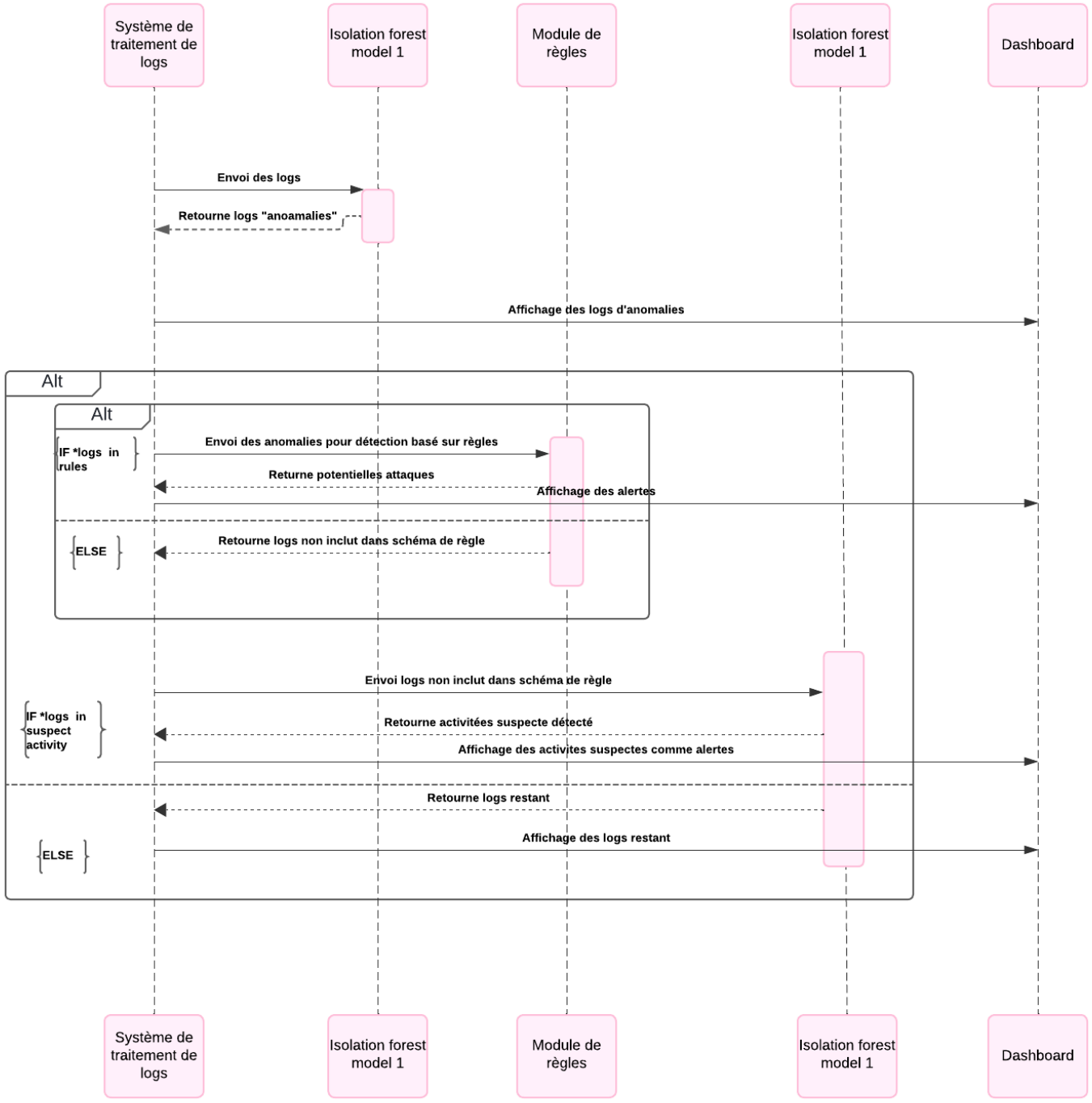


FIGURE 2.8 : Diagramme de séquence du Processus de détection d'incident

Ce diagramme illustre le processus de détection d'attaque ou incident à partir des logs. En effet, les logs subissent d'abord un traitement afin d'être rendu sous forme exploitable par le modèle puis envoyé à celui-ci; le premier modèle utilise un pré-entrainement réalisé préalablement pour détecter les anomalies dans ces logs. Ces anomalies sont remises sous forme de logs et affiché sur le dashboard, aussi ces anomalies passent ensuite par le module de règles afin d'identifié des schémas d'attaques et générer des alertes qui seront affichées sur le dashboard. Les logs qui seront passé au travers les mailles du module de règle subiront une vérification du deuxième modèle d'isolation forest pour y faire ressortir et afficher au Dashboard de potentielles activités suspectes qui peuvent traduire de nouveaux types d'attaques. A la fin de ce processus, les logs restants, n'étant pas reconnu

suspect ou liés à une attaque, sont affichés sur le Dashboard afin de recevoir une dernière analyse, elle, faite par l'humain lui-même.

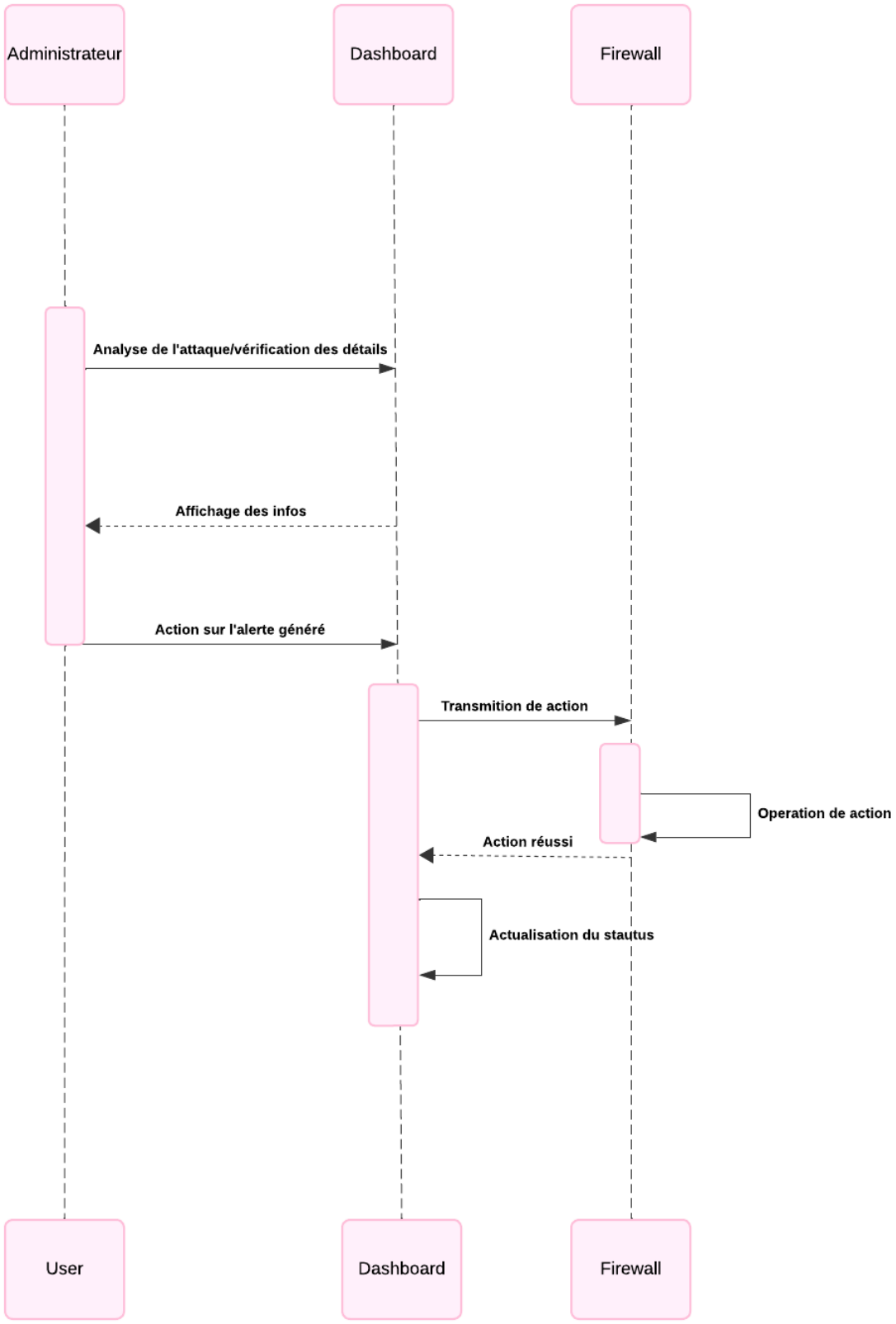


FIGURE 2.9 : Diagramme de séquence du Processus de réponse à un incident

Ce diagramme illustre le processus de réponse à un incident par l'administrateur. En effet, lors-

qu'une alerte est générée, l'administrateur qui est ici notre analyste aura à consulter les détails de l'alerte afin de confirmer sa véracité. Si l'incident s'avère réel, l'administrateur va prendre une action de réponse depuis le Dashboard qui sera directement réaliser en commun accord avec le dispositif de sécurité, ici le firewall, l'action réussi alors le statu de l'incident passera de "En cours" à "Résolu".

2.2.3 Diagramme de classe

Le diagramme de classes présenté ci-dessous modélise la structure logique de la plateforme de détection d'anomalies et de réponse aux incidents développée dans ce travail. Il est constitué de plusieurs entités principales : la classe `Log` représente les journaux collectés à partir des équipements réseau. Chaque `Log` peut être analysé et conduire à la détection d'une `Anomalie`, laquelle est ensuite classifiée en un `Incident`. Un `Incident` peut entraîner une ou plusieurs `Actions` (comme le blocage ou la levée d'une alerte). L'utilisateur interagit avec ces incidents via une interface centralisée représentée par la classe `Dashboard`, qui permet de visualiser les menaces détectées et de prendre des décisions de réponse. Enfin, la classe `Utilisateur` représente les administrateurs ou analystes en sécurité utilisant la plateforme. Ce diagramme formalise les relations fonctionnelles entre les composants clés de la solution et facilite sa compréhension ainsi que son implémentation.

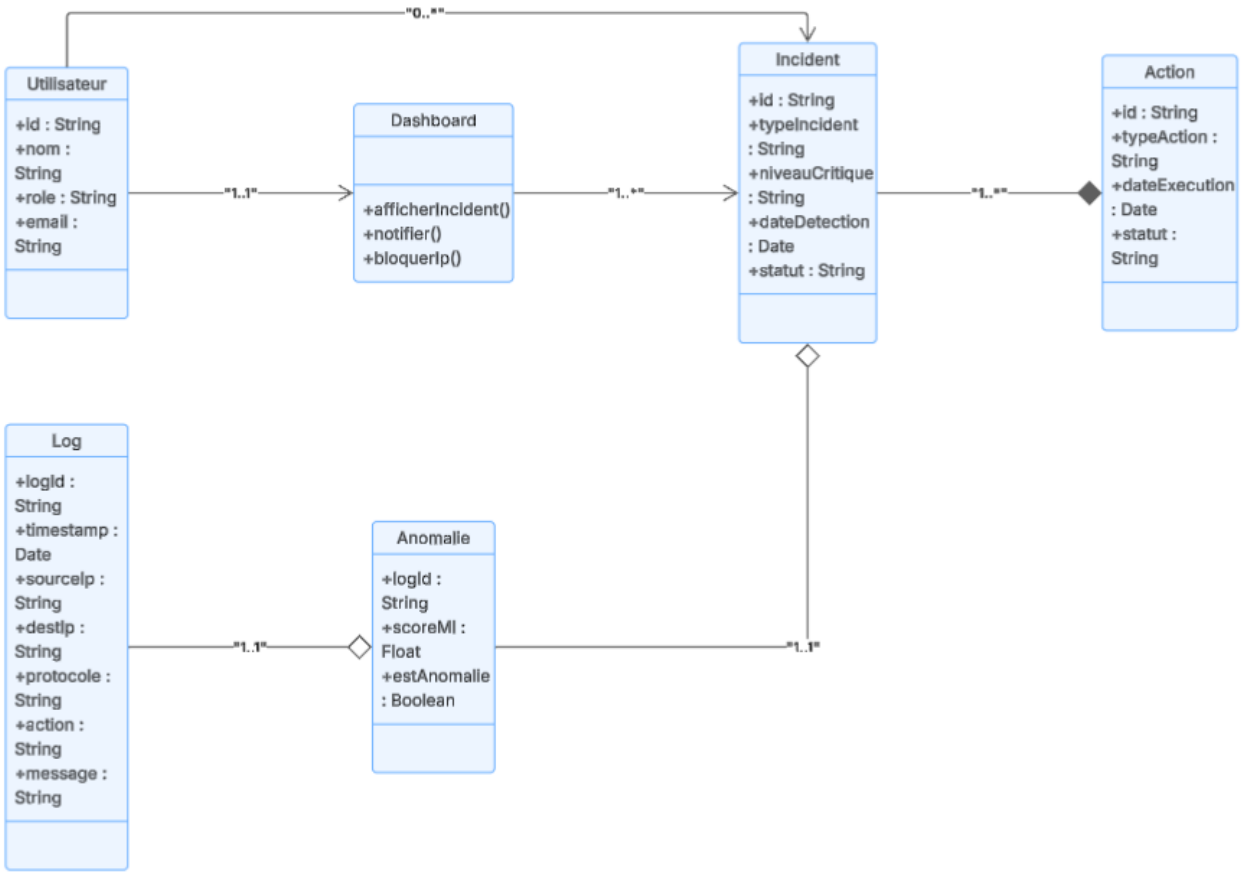


FIGURE 2.10 : Diagramme de classe

2.2.4 Méthodologie et approche de réalisation du projet

2.2.4.1 Jeux de donnée

Pour l'entraînement du modèle, nous avons utilisé un dataset fourni par l'organisation constitué de 17092 logs de firewall dont 14555 logs d'évènements normaux et 2536 logs d'incident.


```
v014xxxxdate=2024-04-29 time=09:48:23 log_id=20000052 msg_id=000000037796 device_id=FV100ET223000047 eventtime=1714380503079387147 vd="root"
timezone="(GMT+1:00)West Central Africa" timezone_dayst="GMTf-1" type=attack pri=alert main_type="Client Management" sub_type="N/A"
trigger_policy="N/A" severity_level=Low proto=tcp service=http backend_service=tcp action=Alert_Deny policy="WebFarmPolicy" src=
185.220.101.167 src_port=59211 dst=10.61.21.148 dst_port=443 http_method=none http_url="none" http_host="none" http_agent="none"
http_session_id=none msg="Client ID 9C23E7DB8B1E842E2F25F4665F901F87A1F0 has exceeded threat score limit.[1714380503-22-65-1|
1713228441-22-65-1|]" signature_subclass="N/A" signature_id="N/A" signature_cve_id="N/A" srccountry="Germany" content_switch_name="none"
server_pool_name="none" false_positive_mitigation="none" user_name="Unknown" monitor_status="Disabled" http_refer="none"
http_version="Unknown" dev_id="9C23E7DB8B1E842E2F25F4665F901F87A1F0" es=0 threat_weight=0 history_threat_weight=0 threat_level=Off
ftp_mode="N/A" ftp_cmd="N/A" cipher_suite="none" ml_log_hmm_probability=0.000000 ml_log_sample_prob_mean=0.000000 ml_log_sample_arglen_mean=
0.000000 ml_log_arglen=0 ml_svm_log_main_types=0 ml_svm_log_match_types="none" ml_svm_accuracy="none" ml_domain_index=0 ml_url_dbid=0
ml_arg_dbid=0 ml_allow_method="none" owasp_top10="N/A" bot_info="none" client_level="Unidentified" x509_cert_subject="none" owasp_api_top10
="N/A"
v014xxxxdate=2024-05-16 time=14:28:31 log_id=11005605 msg_id=000000041276 device_id=FV100ET223000047 eventtime=1715866111225426185 vd="root"
timezone="(GMT+1:00)West Central Africa" timezone_dayst="GMTf-1" type=event subtype="system" pri=notice trigger_policy="N/A" user=daemon
ui=daemon action=check-resource status=failure msg="mem usage keep too high,mem(66)"
```


FIGURE 2.11 : Log d'événement et logs d'attaque

2.2.4.2 Description du Jeu de Données

Le jeu de données utilisé dans ce mémoire regroupe des journaux système (*logs*) capturés à partir d'équipements de sécurité réseau. Ces logs ont été extraits dans le but d'entraîner et d'évaluer des modèles d'apprentissage automatique pour la détection d'anomalies et d'incidents de sécurité.

Le fichier contient deux types de logs distincts :

A. Logs d'attaque (`type=attack`)

Ces logs sont générés lorsqu'une activité considérée comme malveillante est détectée. Ils contiennent des informations riches liées aux menaces et au comportement réseau suspect.

Champ	Signification	Exemple
type	Type de log, ici attack signifie une alerte de sécurité.	attack, event
main_type	Catégorie principale de l'attaque détectée	Client Management
pri	Niveau de priorité de l'événement	alert
severity_level	Niveau de gravité de l'attaque	low, medium, high, etc.
action	Action prise par le système	Alert_Deny
policy	Politique de sécurité ayant déclenché l'alerte	WebFarmPolicy
proto	Protocole réseau utilisé	tcp
src	Adresse IP source de l'attaque	192.168.16.19
dst	Adresse IP de destination visée.	192.168.78.10
msg	Message descriptif de l'événement.	Client ID EH278U0EUSBYG7 has exceeded threat score limit
srccountry	Pays d'origine de l'adresse IP source.	Italy
ml_svm_accuracy	Précision du modèle SVM appliqué au log.	93.75%
bot_info	Informations sur les comportements de type bot détectés.	suspicious

TABLE 2.5 : Exemple de log d'attaque et explication des champs clés

B. Logs d'événements système (type=event)

Ces logs correspondent à des événements internes non malveillants, liés au fonctionnement normal ou à des erreurs système. Ils permettent de suivre l'état de santé du système.

Champ	Signification	Exemple
type	Type de log, ici event signifie un événement système ou administratif	attack, event
subtype	Sous-catégorie de l'événement	system
pri	Niveau de priorité de l'événement	notice
user	Utilisateur ou service à l'origine de l'événement	daemon
action	Action réalisée	check-resource
status	Résultat de l'action effectuée	failure
msg	Message associé à l'événement, contenant des détails utiles	surcharge mémoire

TABLE 2.6 : Exemple de log d'événement et description des champs

Objectif du Jeu de Données

L'objectif est de combiner les deux types de logs dans un même pipeline d'analyse afin :

- d'entraîner un modèle d'apprentissage automatique capable de distinguer les comportements normaux des comportements anormaux,
- de renforcer la détection proactive d'incidents,
- et d'améliorer la visibilité sur l'état du réseau.

Afin d'être utilisé par notre algorithme d'isolation forest, nos logs subissent un traitement comme suit :

- identifier les champs uniques à chaque type d logs.
- Ajouter l'indicateur "existe" et "notexiste" pour suivre la présence ou l'absence d'un champ dans un log donné.
- Ajoute les colonnes absentes avec des valeurs par défaut.

Après ces modifications les données résultante sont sous cette forme

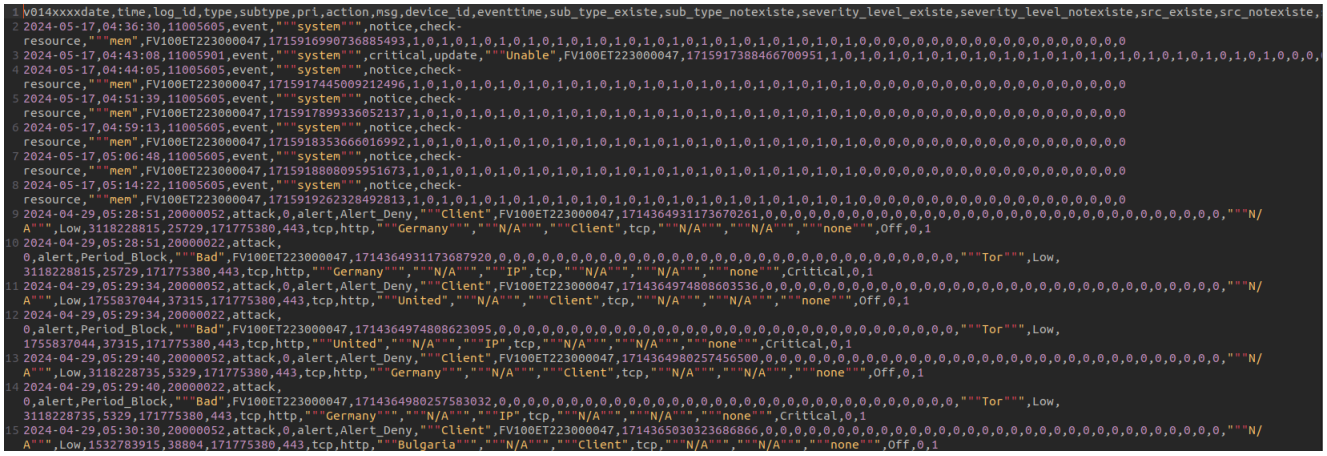


FIGURE 2.12 : Dataset post traitements

2.2.4.3 Développement du modèle d'isolation Forest

Nous avons développé un modèle d'isolation forest dont la fonction est de détecter des anomalies parmi les logs représentant des logs liés à des incidents et attaques. Pour affûter la précision du modèle dans une situation en temps réel, nous avons décidé de partir d'un modèle pré-entraîné. Le modèle fut entraîné et évaluer sur la totalité des données soit exactement 17092 logs. Le résultat de l'entraînement est sauvegardé dans un fichier qui sera utilisé lors du test. Le test est alors réalisé sur 18,68% des données soit 3194 logs constitués de 870 anomalies et de 2324 logs légitimes.
Les résultats du test montrent la détection de 1838 anomalies, la vérification a prouvé que la totalité des anomalies avaient été détectées, bien que 30 % des éléments signalés ne correspondaient pas à de réelles menaces, comme l'illustre cette figure :

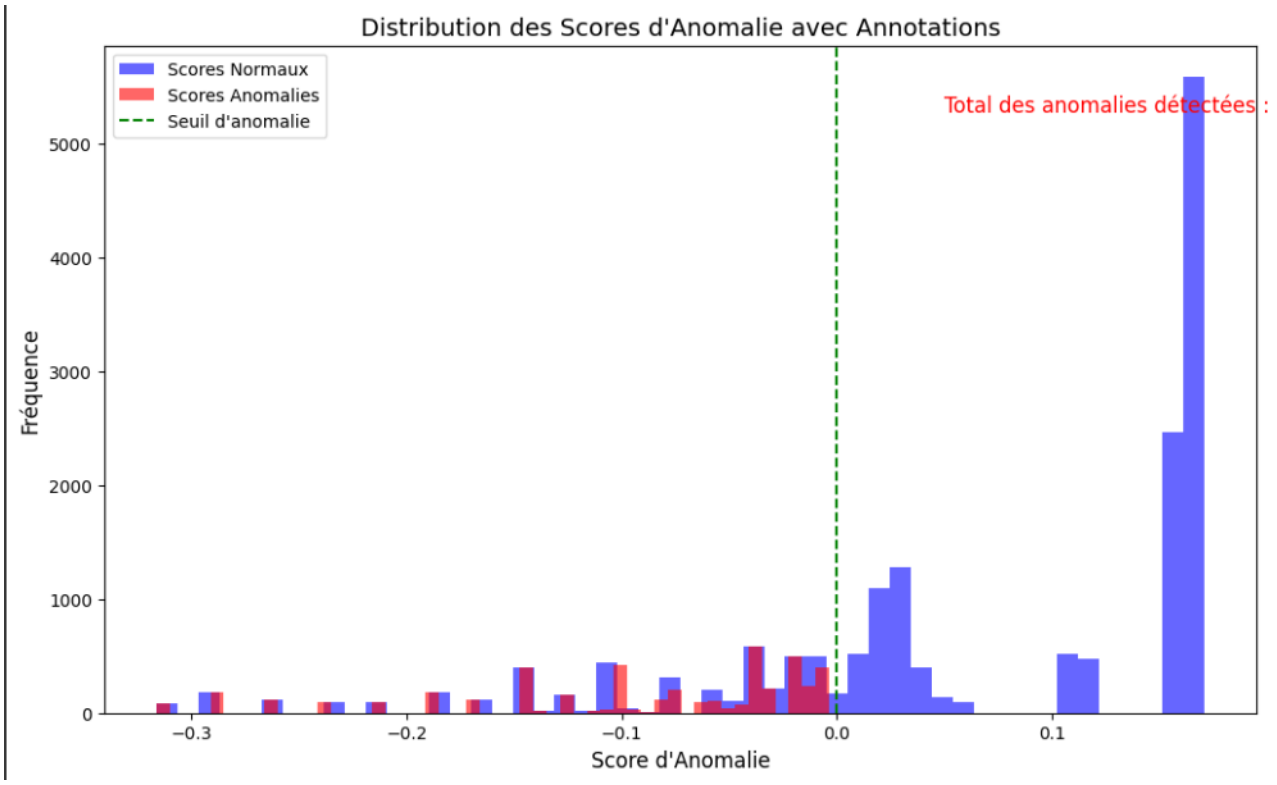


FIGURE 2.13 : test modèle iforest

En variant les tests nous avons remarqué que plus les données d'entrainement son importante, plus notre modèle à la capacité de détecté même les anomalies ne figurant pas dans son dataset d'entraînement. Comme approche de solution à ce problème, nous avons configurer notre modèle pour qu'au fur et à mesure que les logs seront reçus en temps réel, ils soient ajoutés au modèle pré-entrainé, ce qui aura pour effet d'affiner continuellement au fil du temps la détection.

Conclusion

Ce chapitre nous a permis de présenter en détails nos choix techniques et également les étapes de la conception de notre projet. Ces étapes ont consisté à la modélisation et à la présentation de notre approche et méthodologie pour le projet.

Chapitre 3

Déploiement de la solutions et test de fonctionnalité

Introduction

Ce chapitre constitue le corps essentiel de ce mémoire dans lequel nous allons clôturer la mise en place de notre solution. Nous présenterons dans un premier temps l'interfaces de notre application ensuite nous exposerons quelques limites de notre solution.

3.1 Méthodologie de gestion du projet

Pour le développement de notre solution, nous avons opté pour la méthode scrum, une des méthodes de la famille des méthodes agiles.
La méthode **SCRUM** est une méthode agile qui s'adapte aux projets Web, et permet aux développeurs d'être au plus près des besoins du marché, en leur évitant de développer des choses qui ne seront pas utilisées. Le principe de base est simple. L'équipe avance ensemble et reste prête à réorienter le projet au fur et à mesure de sa progression. Il s'agit d'une approche dynamique, participative et empirique de la gestion de projet, "empirique" en cela qu'elle se base sur l'expérience. Comme pour toute méthode Agile, l'équipe Scrum est auto-organisée et pluridisciplinaire. Elle choisit la meilleure façon d'accomplir son travail et possède toutes les compétences nécessaires à l'accomplissement du projet. La flexibilité, la créativité et la productivité de l'équipe sont ainsi optimisées. L'organisation **SCRUM** repose sur trois piliers essentiels

- **Product Owner** le porte-parole des utilisateurs, des clients et des différentes parties prenantes.Il est responsable de la définition et de la mise à jour d'une vision globale du produit.
- **Scrum Master** est le garant de la mise en application de la méthode.
- **Equipe de développement** Collectif responsable de la concrétisation technique du projet

Dans le cadre de mon projet, mon maître mémoire cumule les rôles de product owner et de Scrum master, tandis que j'assume le rôle de développeur.

3.2 Technologies utilisées

Dans le but de faire des choix technique en adéquation avec notre projet, nous avons mis en parallèle deux revue, respectivement de Maurras Ulbricht Togbe[21] et de Lakshmi Geethanjali Mandagondi[15].

	Maurras Ulbricht Togbe	**Lakshmi Geethanjali**
Thème	Détection distribuée d'anomalies dans les flux de données	Utilisation du machine learning pour la détection d'anomalies dans les fichiers logs
Choix Technique	MOA, SAMOA, StreamDm ,Scikit-Multiflow, Pytorch, tensorflow, google colab,Apache spark	Scikit,scipy,Numpy,Padas,, Matplolib, seaborn
Algorithme	Isolation Forest	TFIDF with K-means and PCA

FIGURE 3.1 : Article TOGBE Mauras et Lakshmi

Ce tableau présente une synthèse des choix techniques issus de deux approches différentes. Afin de répondre aux besoins spécifiques de notre projet, nous avons retenu certaines technologies en fonction de leurs avantages pratiques et de leur adéquation avec notre processus de traitement des logs.

Actuellement pour développer un modèle d'IA, plusieurs possibilités s'offrent aux Data Scientist. On peut citer **Tensorflow**, **Kera**, **Pytorch**, **Scikit-learn**, **Caffe**, **Sonnet**, **MXNet**... Chacune de ces solutions possède ses avantages et inconvénients et surtout sont propice à des cas d'utilisations spécifiques.
Pour notre projet, nous avons choisi d'utiliser la bibliothèque **scikit-learn** pour le modèle de machine Learning, **Pandas** pour la structure des données, **Flask** pour l'application web,

- **Scikit-learn** [25] est une bibliothèque libre Python destinée à l'apprentissage automatique. Elle est développée par de nombreux contributeurs notamment dans le monde académique par des instituts français d'enseignement supérieur et de recherche comme l'Inria. Elle propose dans son Framework de nombreuses bibliothèques d'algorithmes à implémenter, clé en main. Ces

bibliothèques sont à disposition notamment des data scientists. Elle comprend notamment des fonctions pour estimer des forêts aléatoires, des régressions logistiques, des algorithmes de classification, et les machines à vecteurs de support. Elle est conçue pour s'harmoniser avec d'autres bibliothèques libres Python, notamment **NumPy** et **SciPy**.
Nous avons choisi d'utiliser **scikit-learn** pour l'implémentation du modèle de détection d'anomalies. Ce choix s'explique par la richesse de sa documentation, sa compatibilité avec Pandas, ainsi que sa capacité à intégrer des algorithmes adaptés à notre problématique, tels que les modèles de détection non supervisés.

- **Flask** est un micro Framework open-source de développement web en Python. Il est classé comme micro Framework car il est très léger. Flask a pour objectif de garder un noyau simple mais extensible. Il n'intègre pas de système d'authentification, pas de couche d'abstraction de base de données, ni d'outil de validation de formulaires. Cependant, de nombreuses extensions permettent d'ajouter facilement des fonctionnalités. Il est distribué sous licence BSD.

- **Pandas** [26] est une bibliothèque écrite pour le langage de programmation Python permettant la manipulation et l'analyse de données. Elle propose en particulier des structures de données et des opérations de manipulation de tableaux numériques et de séries temporelles. Pandas est un logiciel libre sous licence BSD2. Son nom est dérivé du terme Panel Data (en français "données de panel", un terme d'économétrie pour les jeux de données qui comprennent des observations sur plusieurs périodes de temps pour les mêmes individus). Son nom est également un jeu de mots sur l'expression "**Python Data Analysis**".
Pandas a été utilisé pour la manipulation et la structuration des logs centralisés. Grâce à ses structures de données flexibles (DataFrame), il a facilité les étapes de nettoyage, de transformation et d'encodage nécessaires avant l'analyse.

Environnement de développement

Google Colab, ou "Colaboratory" [27], est un service hébergé qui vous permet d'écrire et d'exécuter du code Python directement dans votre navigateur, sans nécessiter de configuration préalable.

3.3 Fonctionnement de la plateforme

La plateforme développée dans le cadre de ce projet a pour objectif de permettre une **surveillance en temps réel des journaux (logs)** issus du réseau et d'assurer une **détection automatique des anomalies et des incidents**, ainsi qu'une **réponse assistée ou automatisée** via une interface web centralisée. Elle repose sur une architecture modulaire composée de plusieurs scripts spécialisés, d'un modèle de machine learning et d'une interface web Flask interconnectée.

3.3.1 Pipeline global de traitement

Le fonctionnement de la plateforme suit les étapes suivantes :

1. **Surveillance des fichiers de logs** : un composant d'écoute analyse en continu les fichiers de logs générés par les équipements réseau (principalement un pare-feu Fortinet). Il détecte toute

nouvelle ligne ajoutée et déclenche automatiquement les traitements.

2. **Prétraitement et nettoyage** : chaque log est analysé, parsé et nettoyé. Les informations non pertinentes sont supprimées, tandis que les champs essentiels (comme les adresses IP, les ports, la sévérité, le protocole, etc.) sont conservés ou encodés.

3. **Détection d'anomalies par Machine Learning** : les logs prétraités sont ensuite transmis à un modèle **Isolation Forest** préalablement entraîné. Ce modèle détecte les logs anormaux (logs considérés comme déviants) et les marques comme anomalies.

4. **Classification par règles** : les anomalies détectées sont ensuite analysées par un script de détection d'incidents basé sur des règles métiers définies (nombre de requêtes par minute, nombre de tentatives de connexion échouées, type de sous-protocoles, etc.). Cela permet de caractériser l'incident : DDoS, brute-force, scan, etc.

5. **Détection d'activitées suspectes** : les anomalies ne vérifiant aucune règle sont ensuite envoyé vers un second modèle d'iforest. Ce modèle détecte les comportements anormaux (connexion hors heures règlementaire, nombre de tentative de connection à un port précis, etc...) et les marques comme activitées suspecte.

6. **Affichage dans le dashboard web** : les incidents sont envoyés dans une **interface web développée avec Flask** où l'utilisateur peut visualiser, filtrer, analyser et prendre des décisions sur les incidents détectés.

7. **Réponse manuelle ou automatisée** : en fonction du type d'incident détecté, une action peut être automatiquement proposée ou exécutée (comme le blocage temporaire d'une adresse IP). L'utilisateur peut également décider manuellement de bloquer ou autoriser une adresse IP directement depuis l'interface.

8. **Interaction avec le pare-feu** : les actions de blocage ou déblocage sont relayées vers le pare-feu via des commandes système exécutées côté serveur, simulant une réponse automatique à l'incident.

3.3.2 Synthèse des modules

La solution est donc composée de :

- Scripts de parsing et de nettoyage (Python)
- Modèle IForest pré-entraîné et dynamique
- Scripts de règles métiers pour classification (détection d'attaques connues)
- Dashboard web (Flask, HTML/CSS/JS)
- Scripts d'actions système pour le blocage réseau

Cette architecture permet une **surveillance continue**, une **détection d'anomalies connues et inconnues**, ainsi qu'une **réponse rapide** à travers une interface accessible et intuitive.

3.4 Interfaces

- **Page d'accueil**
  Sur la page d'accueil

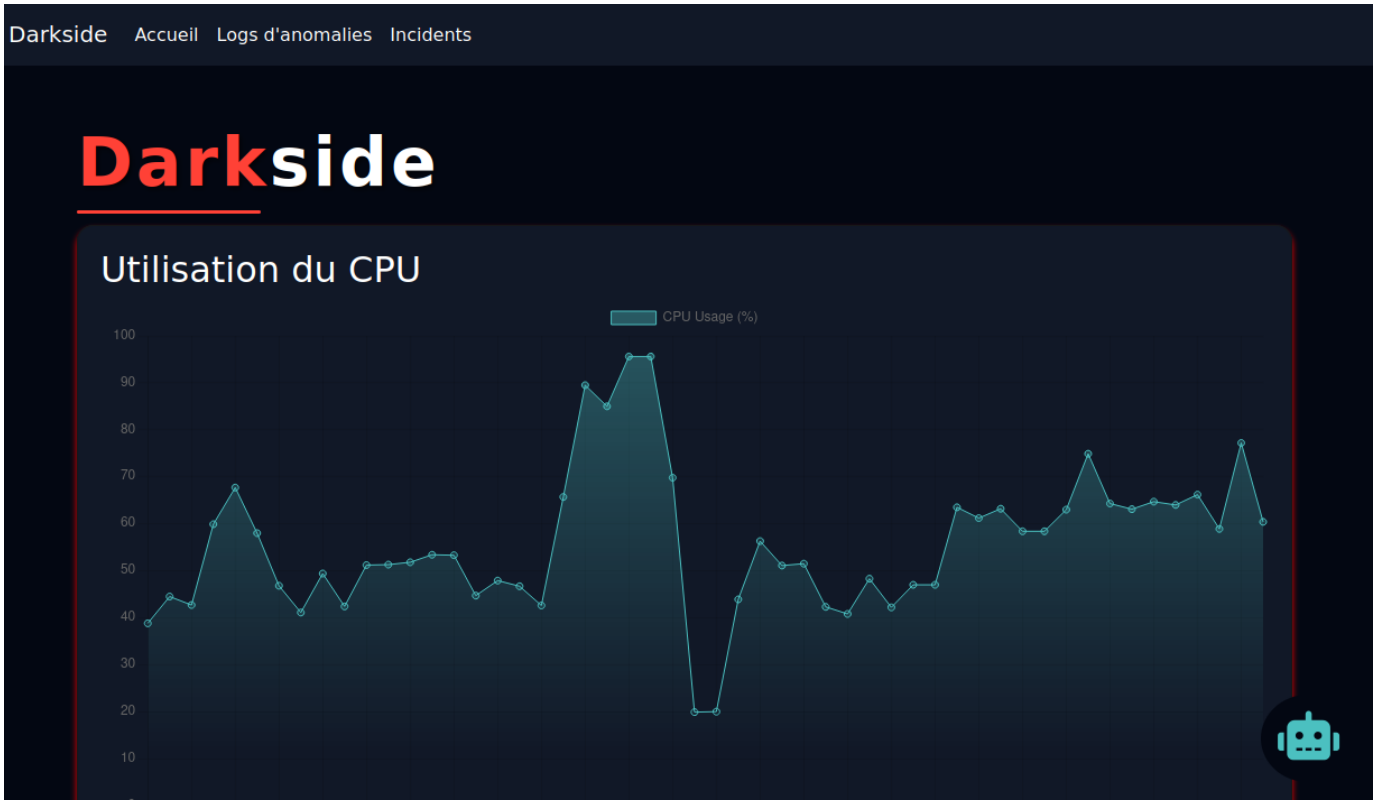


FIGURE 3.2 : Page d'accueil dashboard

Nous avons pour commencer, le graphique traduisant l'utilisation du CPU (Central Processing Unit), qui est essentiel pour surveiller la santé du système, optimiser les ressources, détecter les anomalies et améliorer les performances globales. Aussi nous avons, un diagramme affichant, la proportion de logs reçu et celle des logs d'anomalies détectées.

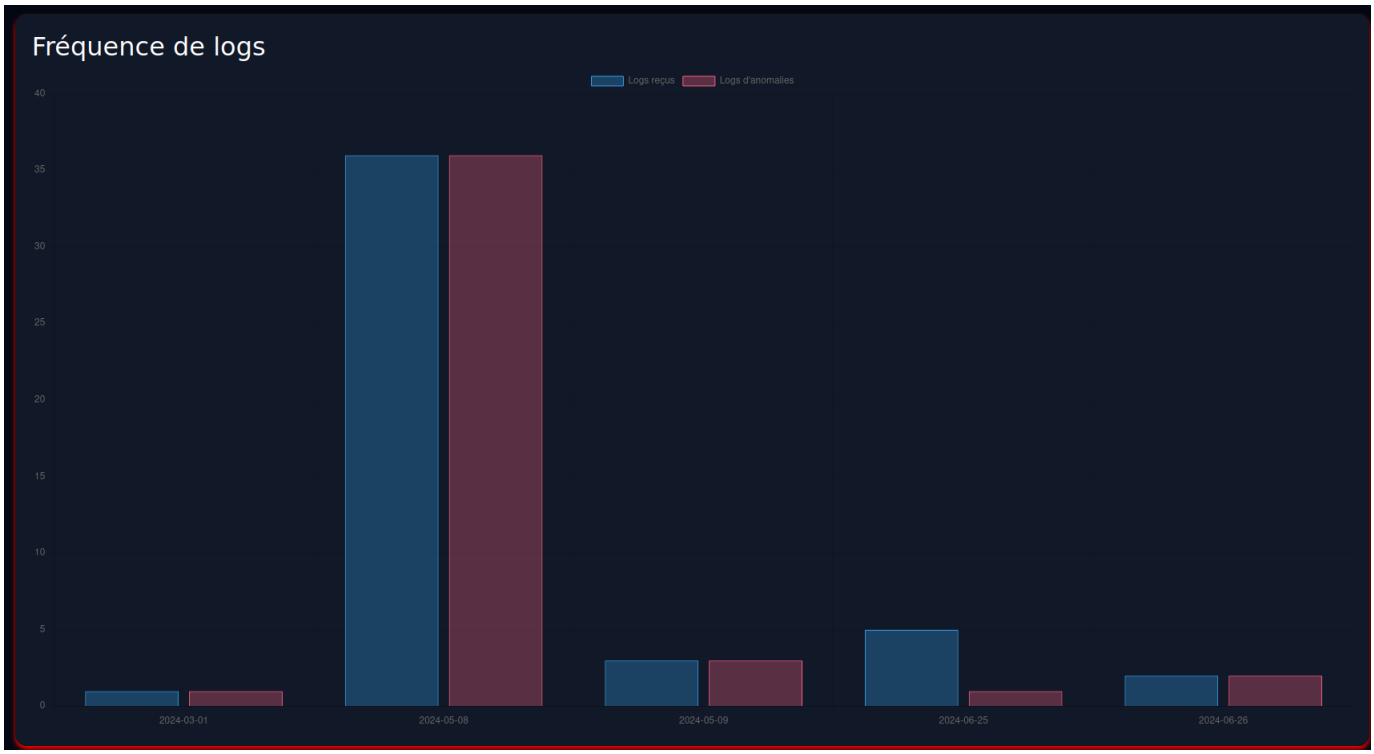
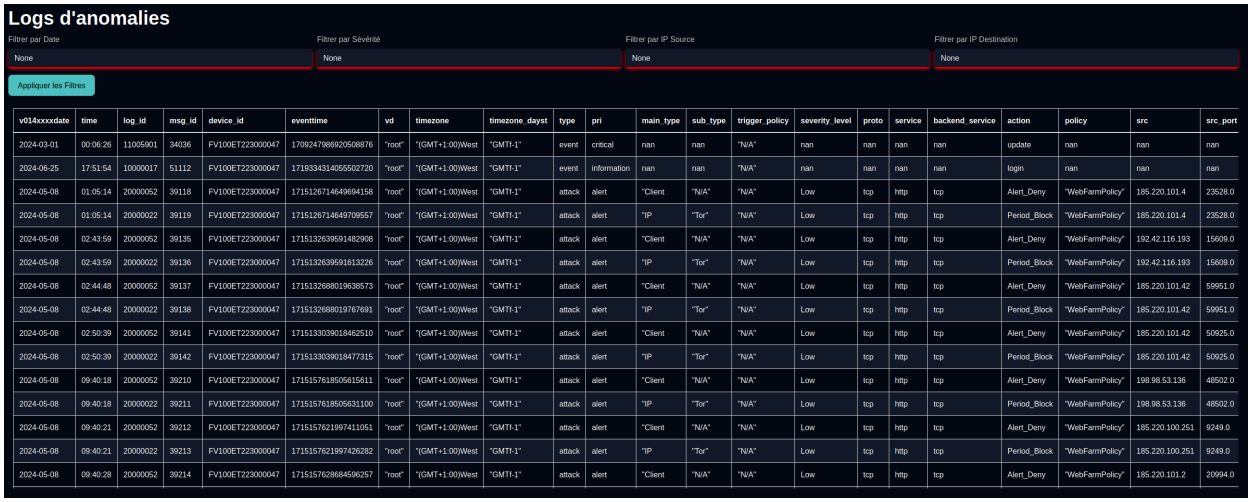


FIGURE 3.3 : Page d'accueil dashboard

- **Page d'affichage des anomalies** Sur la page des anomalies détecté nous avons trois box, le box d'affichage des logs d'anomalies détectés par notre premier modèle de machine Learning. C'est à dire, les logs classés d'attaque par notre premier modèle de machine learning.



v014xxxxdate	time	log_id	msg_id	device_id	eventtime	vd	timezone	timezone_dayst	type	pri	main_type	sub_type	trigger_policy	severity_level	proto	service	backend_service	action	policy	src	src_port
2024-03-01	00:06:26	11005901	34036	FV100ET223000047	1709247986920508876	"root"	"(GMT+1:00)West	"GMTf-1"	event	critical	nan	nan	"N/A"	nan	nan	nan	nan	update	nan	nan	nan
2024-06-25	17:51:54	10000017	51112	FV100ET223000047	1719334314055502720	"root"	"(GMT+1:00)West	"GMTf-1"	event	information	nan	nan	"N/A"	nan	nan	nan	nan	login	nan	nan	nan
2024-05-08	01:05:14	20000052	39118	FV100ET223000047	1715126714649694158	"root"	"(GMT+1:00)West	"GMTf-1"	attack	alert	"Client	"N/A"	"N/A"	Low	tcp	http	tcp	Alert_Deny	"WebFarmPolicy"	185.220.101.4	23528.0
2024-05-08	01:05:14	20000022	39119	FV100ET223000047	1715126714649709557	"root"	"(GMT+1:00)West	"GMTf-1"	attack	alert	"IP	"Tor"	"N/A"	Low	tcp	http	tcp	Period_Block	"WebFarmPolicy"	185.220.101.4	23528.0
2024-05-08	02:43:59	20000052	39135	FV100ET223000047	1715132639591482908	"root"	"(GMT+1:00)West	"GMTf-1"	attack	alert	"Client	"N/A"	"N/A"	Low	tcp	http	tcp	Alert_Deny	"WebFarmPolicy"	192.42.116.193	15609.0
2024-05-08	02:43:59	20000022	39136	FV100ET223000047	1715132639591613226	"root"	"(GMT+1:00)West	"GMTf-1"	attack	alert	"IP	"Tor"	"N/A"	Low	tcp	http	tcp	Period_Block	"WebFarmPolicy"	192.42.116.193	15609.0
2024-05-08	02:44:48	20000052	39137	FV100ET223000047	1715132688019638573	"root"	"(GMT+1:00)West	"GMTf-1"	attack	alert	"Client	"N/A"	"N/A"	Low	tcp	http	tcp	Alert_Deny	"WebFarmPolicy"	185.220.101.42	59951.0
2024-05-08	02:44:48	20000022	39138	FV100ET223000047	1715132688019767691	"root"	"(GMT+1:00)West	"GMTf-1"	attack	alert	"IP	"Tor"	"N/A"	Low	tcp	http	tcp	Period_Block	"WebFarmPolicy"	185.220.101.42	59951.0
2024-05-08	02:50:39	20000052	39141	FV100ET223000047	1715133039018462510	"root"	"(GMT+1:00)West	"GMTf-1"	attack	alert	"Client	"N/A"	"N/A"	Low	tcp	http	tcp	Alert_Deny	"WebFarmPolicy"	185.220.101.42	50925.0
2024-05-08	02:50:39	20000022	39142	FV100ET223000047	1715133039018477315	"root"	"(GMT+1:00)West	"GMTf-1"	attack	alert	"IP	"Tor"	"N/A"	Low	tcp	http	tcp	Period_Block	"WebFarmPolicy"	185.220.101.42	50925.0
2024-05-08	09:40:18	20000052	39210	FV100ET223000047	1715157618505615611	"root"	"(GMT+1:00)West	"GMTf-1"	attack	alert	"Client	"N/A"	"N/A"	Low	tcp	http	tcp	Alert_Deny	"WebFarmPolicy"	198.98.53.136	48502.0
2024-05-08	09:40:18	20000022	39211	FV100ET223000047	1715157618505631100	"root"	"(GMT+1:00)West	"GMTf-1"	attack	alert	"IP	"Tor"	"N/A"	Low	tcp	http	tcp	Period_Block	"WebFarmPolicy"	198.98.53.136	48502.0
2024-05-08	09:40:21	20000052	39212	FV100ET223000047	1715157621097411051	"root"	"(GMT+1:00)West	"GMTf-1"	attack	alert	"Client	"N/A"	"N/A"	Low	tcp	http	tcp	Alert_Deny	"WebFarmPolicy"	185.220.100.251	9249.0
2024-05-08	09:40:21	20000022	39213	FV100ET223000047	1715157621097426282	"root"	"(GMT+1:00)West	"GMTf-1"	attack	alert	"IP	"Tor"	"N/A"	Low	tcp	http	tcp	Period_Block	"WebFarmPolicy"	185.220.100.251	9249.0
2024-05-08	09:40:28	20000052	39214	FV100ET223000047	1715157628684596257	"root"	"(GMT+1:00)West	"GMTf-1"	attack	alert	"Client	"N/A"	"N/A"	Low	tcp	http	tcp	Alert_Deny	"WebFarmPolicy"	185.220.101.2	20994.0

FIGURE 3.4 : Page d'affichage des anomalies

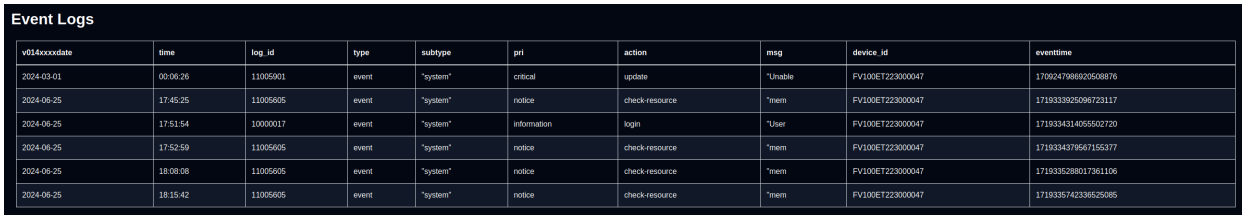
Les logs d'anomalies ne traduisant ou n'étant lié à aucun incident à la fin de tous les processus de détection.



v014xxxxdate	time	log_id	msg_id	device_id	eventtime	vd	timezone	timezone_dayst	type	pri	main_type	sub_type	trigger_policy	severity_level	proto	service	backend_service	action	policy	src	src_port
2024-06-25	17:51:54	10000017	51112	FV100ET223000047	1719334314055502720	"root"	"(GMT+1:00)West	"GMTI-1"	event	information	nan	nan	"N/A"	nan	nan	nan	nan	login	nan	nan	nan
2024-05-08	09:40:18	20000052	39210	FV100ET223000047	1715157618505615611	"root"	"(GMT+1:00)West	"GMTI-1"	attack	alert	"Client	"N/A"	"N/A"	Low	tcp	http	tcp	Alert_Deny	"WebFarmPolicy"	198.98.53.136	48502.0
2024-05-08	09:40:18	20000022	39211	FV100ET223000047	1715157618505631100	"root"	"(GMT+1:00)West	"GMTI-1"	attack	alert	"IP	"Tor"	"N/A"	Low	tcp	http	tcp	Period_Block	"WebFarmPolicy"	198.98.53.136	48502.0
2024-05-08	09:40:21	20000052	39212	FV100ET223000047	1715157621997411051	"root"	"(GMT+1:00)West	"GMTI-1"	attack	alert	"Client	"N/A"	"N/A"	Low	tcp	http	tcp	Alert_Deny	"WebFarmPolicy"	185.220.100.251	9249.0
2024-05-08	09:40:21	20000022	39213	FV100ET223000047	1715157621997426282	"root"	"(GMT+1:00)West	"GMTI-1"	attack	alert	"IP	"Tor"	"N/A"	Low	tcp	http	tcp	Period_Block	"WebFarmPolicy"	185.220.100.251	9249.0
2024-05-08	09:40:28	20000052	39214	FV100ET223000047	1715157628684596257	"root"	"(GMT+1:00)West	"GMTI-1"	attack	alert	"Client	"N/A"	"N/A"	Low	tcp	http	tcp	Alert_Deny	"WebFarmPolicy"	185.220.101.2	20994.0
2024-05-08	09:40:28	20000022	39215	FV100ET223000047	1715157628684610753	"root"	"(GMT+1:00)West	"GMTI-1"	attack	alert	"IP	"Tor"	"N/A"	Low	tcp	http	tcp	Period_Block	"WebFarmPolicy"	185.220.101.2	20994.0
2024-05-08	09:40:31	20000052	39216	FV100ET223000047	1715157631283718118	"root"	"(GMT+1:00)West	"GMTI-1"	attack	alert	"Client	"N/A"	"N/A"	Low	tcp	http	tcp	Alert_Deny	"WebFarmPolicy"	185.220.101.78	27151.0
2024-05-08	09:40:31	20000022	39217	FV100ET223000047	1715157631283733317	"root"	"(GMT+1:00)West	"GMTI-1"	attack	alert	"IP	"Tor"	"N/A"	Low	tcp	http	tcp	Period_Block	"WebFarmPolicy"	185.220.101.78	27151.0
2024-05-08	09:40:35	20000052	39218	FV100ET223000047	1715157635591351327	"root"	"(GMT+1:00)West	"GMTI-1"	attack	alert	"Client	"N/A"	"N/A"	Low	tcp	http	tcp	Alert_Deny	"WebFarmPolicy"	109.70.100.65	5624.0

FIGURE 3.5 : Page d'affichage des anomalies : logs inconnus

Et enfin, les logs identifiés par mon modèle de machine Learning comme étant des logs d'évènements normaux du système.



v014xxxxdate	time	log_id	type	subtype	pri	action	msg	device_id	eventtime
2024-03-01	00:06:26	11005901	event	"system"	critical	update	"Unable	FV100ET223000047	1709247986920508876
2024-06-25	17:45:25	11005605	event	"system"	notice	check-resource	"mem	FV100ET223000047	1719333925096723117
2024-06-25	17:51:54	10000017	event	"system"	information	login	"User	FV100ET223000047	1719334314055502720
2024-06-25	17:52:59	11005605	event	"system"	notice	check-resource	"mem	FV100ET223000047	1719334379567155377
2024-06-25	18:08:08	11005605	event	"system"	notice	check-resource	"mem	FV100ET223000047	1719335288017361106
2024-06-25	18:15:42	11005605	event	"system"	notice	check-resource	"mem	FV100ET223000047	1719335742336525085

FIGURE 3.6 : Page d'affichage des anomalies : logs d'évènements

- **Page d'affichage des alertes**
  Sur cette page, sont affichées les alertes générées par le système



FIGURE 3.7 : Page d'affichage des alertes

Chaque alerte générée, a plusieurs champs :

- La **criticité** allant de warning à high selon l'attaque ou l'incident détecté,
- Le **datetime** qui est la combinaison de la date et de l'heure de détection,
- L'**adresse ip** lié à l'incident ou l'attaque détecté,

- Le **type** de l'attaque ou de l'incident,
- Une brève **description** de l'incident
- L'**Action** proposé à l'administrateur
- Le **statut "en cours" ou "résolu"** de chaque incident

En cliquant sur le niveau de criticité d'un incident, on peut afficher des détails sur cet incident, plus particulièrement, les logs impliqués dans la détection de cet incident.



FIGURE 3.8 : Page d'affichage des alertes : description

Lorsqu'une action est prise sur un incident, en appuyant sur "**Envoyer**", le statut de l'alerte passe à "**Résolu**", comme dans l'exemple ci-dessous

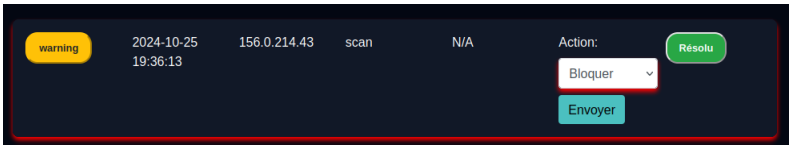


FIGURE 3.9 : Affichage des alertes

En bas, à droite du Dashboard, se trouve un bouton "**Chat**" faisant apparaitre une discussion avec l'IA générative Gemini AI.

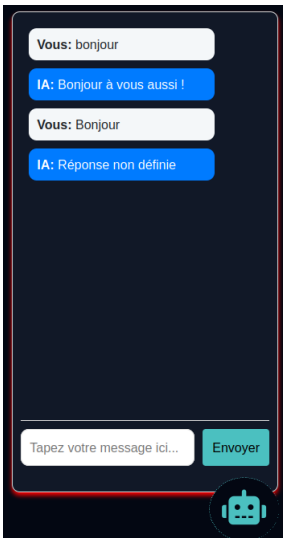


FIGURE 3.10 : Chat

Cette IA générative peut servir d'assistance aux administrateurs lors de l'analyse de l'incident ou des logs désignés, ne traduisant pas un incident, par le système.

3.5 Limites de l'outil

Malgré ses avantages, nous avons relevé quelques limites à notre solution. Il s'agit de :

- Notre modèle n'est pas encore très performant vu les résultats sur l'ensemble de test, aussi nous n'avons pas eu assez un dataset assez conséquent et varié en incident et évènement nous permettant d'affiner la détection.
- Nous ne pouvons pour l'instant que se basé principalement sur un modèle de règle prédéfini par nous même pour la détection d'incidents.,
- Notre outil n'arrive pas encore à fournir une analyse complète et transparente lors des incidents.
- Nous n'arrivons pas encore à corréler une plus grande variété de logs pour avoir plus d'information lors de la survenance d'un incident.

Conclusion

Dans ce chapitre, nous avons vu le fonctionnement de notre outil, de la détection des logs d'anomalies jusqu'à la réponse à un incident. Nous avions pu constater que notre outil a pu détecter et stopper une attaque de scan de port avec succès. Cela permettra aux entreprises de pouvoir bien protéger leurs actifs contre plusieurs attaques grâce au machine learning.

Conclusion Générale

Grâce à ce projet de mémoire, qui se concentre sur l'utilisation de l'intelligence artificielle pour détecter des anomalies et résoudre les incidents de sécurité, des techniques novatrices ont été explorées dans le domaine de la cybersécurité. J'ai pu montrer l'efficacité de ces méthodes en utilisant des modèles de machine Learning, comme l'Isolation Forest, pour repérer en temps réel des comportements anormaux dans des environnements réseau complexes.

Un des résultats majeurs de cette étude est la création d'un système qui peut traiter et analyser des journaux, repérer des menaces potentielles et classer diverses formes d'attaques. Avec ce système, la réactivité face aux incidents de sécurité est améliorée, tout en diminuant les faux positifs, un défi majeur dans ce secteur. Outre le côté technique, ce projet m'a donné l'opportunité d'approfondir ma compréhension des enjeux liés à la gestion de la sécurité des systèmes d'information, en particulier en ce qui concerne la gestion de données massives et la surveillance en temps réel.

Malgré les résultats obtenus, il reste encore quelques possibilités d'amélioration. Il est possible que l'incorporation de nouvelles méthodes d'apprentissage non supervisé et l'utilisation de réseaux neuronaux puissent encore améliorer la précision et les performances du système. En outre, la recherche d'autres types de données, tels que les flux d'événements en temps réel, pourrait favoriser une meilleure prévision des incidents.

En résumé, cette étude a non seulement démontré l'importance de l'intelligence artificielle dans la détection d'anomalies, mais a également posé les fondations d'améliorations futures visant à renforcer la sécurité des systèmes d'information. Ce projet marque un tournant majeur dans ma carrière, et m'a offert de nouvelles opportunités dans le domaine de la sécurité informatique et de l'intelligence artificielle.

Bibliographie

[1] Douglas M. Hawkins *"Identification of outliers"*,1980,Springer, Dordrecht

[2] Victoria Hodge & Jim Austin *"A Survey of Outlier Detection Methodologies.Artificial intelligence review"*,2004

[3] Patcha Animesh et Park Jung-Min *"An overview of anomaly detection techniques : Existing solutions and latest technological trends. Computer networks"*, 2007

[4] Aggarwal Charu C. *Outlier Analysis. Second Edition. Springer International*, 2017,AG

[5] Chandola Varun, Banerjee Arindam et Kumar Vipin *" Anomaly detection : A survey"*,2009, ACM computing surveys (CSUR)

[6] Gupta Manish, Gao Jing, Aggarwal Charu C et Han Jiawei *"Outlier detection for temporal data : A survey. IEEE Transactions on Knowledge and Data Engineering"*,2014

[7] Stolfo, Salvatore J and Fan, Wei and Lee, Wenke and Prodromidis, Andreas L and Chan, Philip K *Cost-based modeling for fraud and intrusion detection : Results from the JAM project ; DARPA Information Survivability Conference and Exposition. DISCEX'00. Proceedings*, 130–144,2000

[8] Moustafa, Nour and Slay, Jill, *UNSW-NB15 : a comprehensive data set for network intrusion detection systems (UNSW-NB15 network data set) ; Military Communications and Information Systems Conference (MilCIS)*, 1–6, 2015, IEEE

[9] Salehi Mahsa et Rashidi Lida *"A survey on anomaly detection in evolving data :[with application to forest fire risk prediction]"*,2018 ,ACM SIGKDD Explorations Newsletter

[10] Souiden Imen, Brahmi Zaki et Toumi Hajer *"A survey on outlier detection in the context of stream mining : review of existing approaches and recommadations. International Conference on Intelligent Systems Design and Applications"*,2016

[11] Zhang Ji *"Advancements of outlier detection : A survey. ICST Transactions on Scalable Information Systems"*,2013

[12] Tellis Venisha Maria et D'Souza Divya Jennifer *"Detecting Anomalies in Data Stream Using Efficient Techniques : A Review. 2018 International Conference on Control, Power, Communication and Computing Technologies (ICCPCCT)."*,2018

[13] Chalapathy Raghavendra et Chawla Sanjay *"Deep learning for anomaly detection : A survey"*,2019

[14] Mohotti Wathsala Anupama et Nayak Richi *"Efficient outlier detection in text corpus using rare frequency and ranking. ACM Transactions on Knowledge Discovery from Data (TKDD)"*,2020

[15] Lakshmi Geethanjali Mandagondi, *"AnomalyDetection in log files using Machine learning Techniques"*, Consulté le 11 mars 2025

[16] Mejri, Nesryne and Lopez-Fuentes, Laura and Roy, Kankana and Chernakov, Pavel and Ghorbel, Enjie and Aouada, Djamila, *Unsupervised Anomaly Detection in Time-series : An Extensive Evaluation and Analysis of State-of-the-art Methods*, arXiv preprint arXiv :2212.03637, 2022

[17] Idan, Lihi, *Towards Unsupervised Validation of Anomaly-Detection Models*, arXiv preprint arXiv :2410.14579 ; 2024

Webographie

[18] Red Hat, Comprendre la sécurité informatique,https://www.redhat.com/fr/topics/security, Consulté le 20 octobre 2024

[19] Groupe MDSI, A QUOI SERT LE MONITORING INFORMATIQUE ?,https://mdsi.re/a-quoi-sert-le-monitoring-informatique/, Consulté le 20 octobre 2024

[20] Qu'est-ce qu'un IDS ?, https://www.ibm.com/fr-fr/topics/intrusion-detection-system#:~:text=Un%20syst%C3%A8me%20de%20d%C3%A9tection%20d%E2%80%99intrusion%20%28IDS%29%20est%20un,suspectes%20ou%20les%20violations%20des%20politiques%20de%20s%C3%A9curit%C3%A9., Consulté le 20 octobre 2024

[21] Etude comparative des méthodes de détection d'anomalies, Maurras Ulbricht Togbe, Yousra Chabchoub, Aliou Boly, Raja Chiky ,https://hal.science/hal-02874904/document, Consulté le 20 octobre 2024

[22] Détecter des anomalies avec de l'Intelligence Artificielle ,https://www.avisia.fr/actualites/blog/data/detecter-des-anomalies-avec-de-lintelligence-artificielle, Consulté le 20 octobre 2024

[23] La détection d'anomalies en Machine Learning non supervisé by Patrick Hairy ,https://metalblog.ctif.com/2022/10/03/la-detection-danomalies-en-machine-learning-non-supervise/, Consulté le 21 octobre 2024

[24] K-Means : algorithme de clustering et cas d'usage ,https://cyberinstitut.fr/k-means-algorithme-clustering-cas-usage/#:~:text=Le%20K-Means%20est%20un%20algorithme%20de%20clustering%20sp%C3%A9cifique,des%20K%20groupes%20en%20minimisant%20la%20variance%20intra-cluster. ,Consulté le 21 octobre 2024

[25] scikit-learn Machine Learning in Python ,https://www.bing.com/ck/a?!&&p=50101c0300a07ce2JmltdHM9MTcyOTgxNDQwMCZpZ3VpZD0wMWQyOWJiOS1jODVjLTYwNGMtMTJjZi04ZjA0Yzk0OTYxNDMmaW5zaWQ9NTE4Nw&ptn=3&ver=2&hsh=3&fclid=01d29bb9-c85c-604c-12cf-8f04c9496143&psq=Scikit-learn&u=a1aHR0cHM6Ly9zY2lraXQtbGVhcm4ub3JnL3N0YWJsZS9pbmRleC5odG1s&ntb=1, Consulté le 21 octobre 2024

[26] Pandas ,https://pandas.pydata.org/, Consulté le 21 octobre 2024

[27] Google Colab ,https://www.bing.com/ck/a?!&&p=09eb1a40a765c412JmltdHM9MTcyOTgxNDQwMCZpZ3VpZD0wMWQyOWJiOS1jODVjLTYwNGMtMTJjZi04ZjA0Yzk0OTYxNDMmaW5zaWQ9NTE4NA&ptn=3&ver=2&hsh=3&fclid=01d29bb9-c85c-604c-12cf-8f04c9496143&psq=Google+Colab&u=a1aHR0cHM6Ly9jb2xhYi5yZXNlYXJjaC5nb29nbGUuY29tLw&ntb=1, Consulté le 21 octobre 2024

Table des matières

Dédicace ii

Remerciements iii

Résumé iv

iv

Abstract v

v

Liste des figures vi

Liste des tableaux vii

Acronymes viii

Introduction 1

Introduction . . . 1

1 Revue de littérature 3

Introduction . . . 3

1.1 La sécurité informatique . . . 3

1.1.1 Définition . . . 3

1.1.2 Les Menaces informatiques . . . 4

1.2 Le monitoring . . . 4

1.2.1 Définition . . . 4

1.2.2 Objectifs du monitoring . . . 4

1.3 Les IDS . . . 5

1.3.1 Fonctionnement des systèmes de détection d'intrusion . . . 6

1.3.1.1 Détection basée sur les signatures . . . 6

1.3.1.2 Détection basée sur les anomalies . . . 7

2 Méthodes de détection d'anomalies 8

Introduction . . . 8

2.1 Etat de l'art des méthodes . . . 8

2.1.1 Domaines d'application . . . 9

2.1.2 Techniques de détection d'anomalies . . . 10

2.1.2.1 Méthodes statistiques . . . 11

2.1.2.2 Deep Learning . . . 11

2.1.2.3 Machine Learning . . . 11
2.1.2.4 Local Outlier Factor (LOF) . . . 12
2.1.2.5 One-class Support Vector Machine (OC-SVM) . . . 13
2.1.2.6 Forêt d'isolement (Isolation forest) . . . 14
2.1.2.7 La méthode des K-means . . . 15
2.1.3 Choix du modèle . . . 16
2.1.4 Etude comparative de quelques méthodes . . . 16
2.1.5 Résultats . . . 17
2.1.6 Justification du choix final du modèle Isolation Forest : . . . 18
2.1.7 État de l'art des plateformes de détection d'anomalies existantes . . . 18
2.2 Modélisation . . . 20
2.2.1 Diagramme de cas d'utilisation . . . 20
2.2.2 Diagramme de séquence . . . 21
2.2.3 Diagramme de classe . . . 24
2.2.4 Méthodologie et approche de réalisation du projet . . . 25
2.2.4.1 Jeux de donnée . . . 25
2.2.4.2 Description du Jeu de Données . . . 25
2.2.4.3 Développement du modèle d'isolation Forest . . . 27
Conclusion . . . 28

3 Déploiement de la solutions et test de fonctionnalité 29
Introduction . . . 29
3.1 Méthodologie de gestion du projet . . . 29
3.2 Technologies utilisées . . . 30
3.3 Fonctionnement de la plateforme . . . 31
3.3.1 Pipeline global de traitement . . . 31
3.3.2 Synthèse des modules . . . 32
3.4 Interfaces . . . 33
3.5 Limites de l'outil . . . 37
Conclusion . . . 37

Conclusion 38

Bibliographie 39

Webographie 41

Table des matières 43