République Algérienne Démocratique et Populaire

Ministère de l'Enseignement Supérieur et de la Recherche Scientifique

Université A. MIRA-BEJAIA

جامعة بجاية

Tasdawit n Bgayet

Université de Béjaïa

Faculté des Sciences Exactes

Département d'Informatique

MÈMOIRE

En vue de l'obtention du Diplôme de Master en Informatique

Domaine : Mathématiques et Informatique **Filière :** Informatique

Spécialité : Réseaux et Sécurité

Présenté par

Mlle. BOUCHEKOUT Liticia

Encadré par

M. MOKTEFI Mohand

Thème

Réalisation d'un benchmark DAPT pour les attaques APT (Advanced Persistent Threat)

Soutenu le 30 Juin 2025. **Devant le jury composé de :**

Nom et Prénom : **Mme. Zamouche Djamila**

Mme. Yaici Malika

Mme. Aloui Soraya

Mme. MAMMERI Souhila

Année Universitaire : 2024/2025

Remérciements

Avant tout, je rends grâce à Dieu Tout-Puissant qui m'a donné la force, la patience et la persévérance pour mener à bien ce travail et surmonter les épreuves.

Je tiens à exprimer ma profonde reconnaissance à mon encadrant **MOKTEFI Mohand** pour ses conseils avisés, sa disponibilité et son accompagnement précieux tout au long de ce mémoire. Son expertise et son soutien ont été déterminants dans la réalisation de ce projet.

Je remercie également tous les membres de jury pour l'intérêt qu'ils ont manifesté envers mon travail en acceptant de l'examiner et de l'enrichir par leurs précieuses suggestions

Mes sincères remerciements s'adressent aussi à mes enseignants tout au long de ma formation, qui ont su éveiller ma curiosité, me transmettre leur savoir et me guider avec bienveillance.

Je n'oublie pas mes collègues, mes ami(e)s, pour leur soutien moral, leurs encouragements, les échanges constructifs et les moments de partage inoubliables.

Enfin, un merci du fond du cœur à ma famille, pilier de ma vie, pour leur amour inconditionnel, leur patience et leurs prières constantes. Sans vous, rien n'aurait été possible.

Dédicaces

Je dédie ce travail ...

A moi même

ma propre personne, à celle qui, malgré les difficultés et les obstacles, a poursuivi son chemin avec détermination, travail et foi tout au long de son parcours académique, jusqu'à ce jour tant espéré.

A mon chère papa

À celui qui a enlevé les épines de mon chemin, le premier homme de ma vie, qui m'a toujours soutenue, s'est sacrifié pour moi, et m'a appris la valeur du travail, de la patience et de la force. à celui que je porte son nom, mon chère papa **Bouchekout Karim**.

A ma chère maman

À celle qui m'a appris à tenir la plume, qui a veillé avec moi durant de longues nuits, s'est sacrifiée pour mon bien, et a semé en moi les plus nobles valeurs. À ma mère bien-aimée, source infinie de tendresse, d'amour et de générosité.

A ma sœur et mes frères bien-aimés

À mes frères et à ma précieuse sœur, qui m'ont toujours encouragée et soutenue, présents dans les moments de doute comme dans les moments de joie.

Et à tous ceux qui m'ont aidée de près ou de loin, qui m'ont accompagnée par leur présence, leur soutien ou leurs prières, à tous ceux qui me sont chères et occupent une place dans mon cœur .
Je vous dédie le fruit de cet accomplissement que j'ai tant rêvé.

Table des matières

Table des figures

Liste des tableaux

Liste des abréviations et acronymes

ACC	Accuracy
APT	Advanced Persistent Threat
Bi-LSTM	Bidirectional Long Short-Term Memory
C2	Command and Control (Commande et Contrôle)
CIC-IDS 2017	Canadian Institute for Cybersecurity - Intrusion Detection System 2017 Dataset
CNN	Convolutional Neural Network
DAPT 2020	Dataset for Advanced Persistent Threats 2020
DARPA TC	Defense Advanced Research Projects Agency – Transparent Computing
DDoS	Distributed Denial of Service
DLP	Data Loss Prevention
DoS	Denial of Service (Déni de service)
FL	Federated Learning
FNR	False Negative Rate
FPR	False Positive Rate
GCN	Graph Convolutional Network
GNN	Graph Neural Network
GRU	Gated Recurrent Unit

GV-FL	Global Vision Federated Learning
HTTP	HyperText Transfer Protocol
HTTPS	HyperText Transfer Protocol Secure
IA	Intelligence Artificielle
IDS	Intrusion Detection System (Système de Détection d'Intrusions)
IoT	Internet of Things
LSTM	Long Short-Term Memory
ML	Machine Learning
MLP	Multi-Layer Perceptron
MITM	Man-In-The-Middle
NLP	Natural Language Processing
OpenAPT	Open-source dataset for APT scenarios
OS	Operating System
RAPID	Robust APT Detection and Investigation using Context-Aware Deep Learning
Recall	Rappel (Taux de détection)
RNN	Recurrent Neural Network
RRCF	Robust Random Cut Forest
SDN	Software Defined Network (Réseau Défini par Logiciel)
SHAP	SHapley Additive exPlanations
SQL	Structured Query Language
SQLi	SQL Injection
TPR	True Positive Rate
URL	Uniform Resource Locator
USB	Universal Serial Bus
VPN	Virtual Private Network (Réseau privé virtuel)
XFedHunter	Explainable Federated Learning Framework for APT Detection in SDN
Zero-Day	Vulnérabilité non encore connue ni corrigée au moment de l'attaque

Introduction générale

La cybersécurité constitue aujourd'hui un enjeu essentiel pour l'ensemble des acteurs, qu'il s'agisse d'institutions publiques, d'entreprises ou même de particuliers. Face à la multiplication de menaces cybernétiques sophistiquées, le repérage et la prévention des intrusions s'avèrent être des défis majeurs. Parmi ces menaces, les attaques dites persistantes avancées (APT – Advanced Persistent Threats) se distinguent en combinant des techniques complexes et discrètes pour maintenir leur présence dans les systèmes informatiques sur de longues périodes. [1]

Les APT sont généralement menées par des groupes organisés hautement qualifiés disposant de moyens humains, techniques et financiers importants. Ces groupes visent des cibles stratégiques telles que les secteurs de la défense, de l'énergie ou encore de la finance, avec pour objectif le vol de données sensibles ou la perturbation d'infrastructures vitales. Leurs conséquences peuvent s'avérer catastrophiques, tant sur le plan économique que politique et social. [2]

Dans ce contexte, les outils traditionnels de protection comme les pare-feux, les antivirus ou encore les systèmes classiques de détection d'intrusion montrent leurs limites. Il devient donc indispensable de s'orienter vers des mécanismes de défense plus intelligents et adaptatifs.

L'apprentissage automatique (Machine Learning) offre justement une réponse innovante à cette problématique. En exploitant la capacité des algorithmes à traiter et analyser d'importants volumes de données, ces techniques permettent de détecter des comportements suspects ou des schémas inhabituels pouvant trahir la présence d'une attaque. [3]

La problématique que nous abordons dans ce mémoire est donc la suivante : Quelles sont les méthodes récentes efficaces et capables de détecter ce gere d'attaques,et comment remédier aux limites de la méthode existente choisie tout en exploitant les techniques de Machine Learning ?

Ce mémoire s'inscrit dans cette démarche et vise à développe une des approches récentes existantes de la détection des attaques APT en se basant sur les techniques du machine leraning et dans le but de remédier aux limites.

La structure du mémoire est la suivante :

Le premier chapitre présente les notions de bases de la Cyber criminalité, les fondements de la sécurité ainsi qu'un panorama des différentes attaques informatiques.

Le deuxième chapitre étudie le processus d'évolution des APT, leurs particularités en analysant certains groupes connus et les méthodes qu'ils emploient pour leurs attaques, ensuite une revue approfondie des approches récentes existantes basées sur le Machine Learning pour détecter ce type d'attaques. Après avoir introduit les concepts fondamentaux de cette discipline, il analyse plusieurs travaux récents qui mettra en lumière les forces et les faiblesses.

Le troisième chapitre est consacré à la conception, la mise en œuvre et l'évaluation de notre propre solution. nous détaillerons l'architecture du système proposé, les jeux de données exploités, les outils utilisés, ainsi que les étapes essentielles allant du prétraitement à l'entraînement du modèle. Les performances de la solution seront ensuite évaluées et comparées à celles d'approches existantes.

Enfin, une conclusion viendra résumer les apports de ce travail et proposer des pistes pour de futures recherches.

Chapitre 1

Généralités sur la cybercriminalité et la sécurité informatique.

1.1 Introduction

Dans un monde, de plus en plus interconnecté,la révolution technologique et la transformation numérique a profondément modifié nos modes de vie,tout en exposant de nouvelles menaces et risques, ce qui fait de la cybercriminalité un des plus grands défis de notre époque. Elle englobe une variété de menaces allant du piratage individuel aux attaques massives orchestrées par des groupes organisés. La cyber sécurité, quant à elle, vise à protéger les données, les systèmes et les infrastructures critiques contre ces menaces,Toutefois, certaines formes d'attaques, telles que les attaques persistantes avancées (APT), se distinguent par leur sophistication, leur caractère persistant et leur ciblage précis, représentant un danger majeur pour les organisations et les nations.

Ce chapitre vise à explorer les fondements de la cybercriminalité et de la sécurité informatique, tout en mettant en lumière les mécanismes et les impacts des APTs. Il constitue une introduction essentielle pour appréhender la complexité croissante des menaces cybernétiques et les réponses nécessaires pour y faire face.

1.2 Cybercriminalité

1.2.1 Définition

La cybercriminalité désigne l'ensemble des actes délictueux ou criminels qui sont commis à l'aide des technologies de l'information,ayant des comportements malveillants tels que : le piratage informatique, la diffusion de logiciels malveillants ,l'usurpation d'identité, les escroqueries en ligne ainsi que l'espionnage numérique,tout en exploitant les failles technologiques ou humaines pour nuire aux individus, aux entreprises ou aux institutions.

1.2.2 Caractéristiques :

Transnationalité :

Les cybercrimes, peuvent être commis depuis n'importe quel endroit (différent de celui de la victime) dans le monde, ils ne connaissent pas les frontières géographique, ce qui rend la traçabilité et l'application des lois complexes.

Coût réduit :

Les cybercrimes , ne nécessitent pas de grandes ressources finanières pour être commis, les acteurs peuvent utiliser un simple ordinateur, une connexion Internet et certaines compétences techniques pour mener des attaques sérieuses.

Anonymat :

Les cybercriminels , s'appuient sur plusieurs techniques avancées pour cacher leur identité notamment les VPN et l'accès au dark web ce qui complique leur suivi et leur localisation.

Évolution rapide :

Les cyber menaces évoluent en parallèle des innovations technologiques,à mesure que de nouveaux outils de sécurité émergent, les cyberattaques trouvent de nouveaux moyens de les contourner, obligeant les entités de sécurité à mettre à jour leurs outils constamment.

1.2.3 Types de cybercriminalité

Crimes ciblant les individus :

Hameçonnage :

Utilisé pour tromper la victime au moyen de faux messages semblant provenir de sources fiables, pour but de la pousser à révéler des données secrètes et sensibles telles que des mots de passe ou des informations de cartes bancaires... .

Usurpation d'identité :

Elle s'agit de l'utilisation des informations personnelles d'une autre personne pour des activités illégales.

Cyber harcèlement :

comportement hostile exercé via les réseaux sociaux, e-mail, ou à travers les autres plateforme en ligne, visant à nuire psychologiquement à la victime ou à sa réputation.

Ransomware :

Un type de cyberattaque dans lequel les données d'un utilisateur sont cryptées et une rançon est exigée pour les récupérer.

Crimes ciblant les organisations :

Intrusion informatique :

Il s'agit de piratage des systèmes et réseaux afin de voler des informations confidentielles ou provoquer des perturbations d'activités.

Espionnage industriel :

Une pratique criminelle visant à récupérer illégalement des secrets commerciaux ou technologiques pour un avantage concurrentiel.

Déni de service (DoS) :

Cela se fait en inondant le serveur de l'organisation avec un grand nombre de fausses requêtes pour le rendre inaccessible.

Fraude en ligne :

Exploitation des vulnérabilités électroniques et manipulation des transactions financières en ligne pour voler ou transférer de l'argent illégalement.

Crimes ciblant les infrastructures critiques :

Attaques contre les réseaux électriques, les systèmes de transport et les hôpitaux .

Cyberterrorisme :

En utilisant des cyberattaques afin de provoquer des destructions et des perturbations à grande échelle et de semer la panique au sein de la population.

1.2.4 Acteurs de la cybercriminalité

Hackers indépendants :

motivés soit par le gain financier, soit par un désir de divertissement et de défi, ou par la curiosité de tester leurs capacités techniques.

Groupes organisés :

Entités opérant de manière organisée, à la manière des mafias numériques,pour but de générer des gains financiers importants ou de commettre des fraudes à grande échelle.

Hacktivistes :

Des groupes ou des individus motivés par des idéologies ou des causes politiques.

États-nations :

Des entités officielles qui ont recours aux cyberattaques dans le cadre de leurs stratégies de sécurité nationale dans le but de Cyber espionnage ou de guerre hybride.

1.2.5 Techniques utilisées par les cybercriminels

Malwares (logiciels malveillants) :

Incluent les virus, chevaux de Troie, ransomwares, et spywares qui sont utilisés pour endommager les appareils, voler des informations sensibles ou perturber les systèmes.

Exploitations de failles :

Exploitation des vulnérabilités logicielles ou matérielles pour compromettre les systèmes ou exécuter du code malveillant à l'insu des utilisateurs.

Social engineering :

C'est une méthode basée sur la tromperie et la manipulation psychologique des utilisateurs pour obtenir des informations sensibles.

Botnets :

Un groupe d'ordinateurs compromis utilisés pour mener des attaques massives et coordonnées.

1.2.6 Impacts de la cybercriminalité

Les cybercrimes ne se limitent pas à des pertes immédiates uniquement ; leur impact s'étend à de nombreux aspects de la vie.

Sur le plan économique, ils entraînent des pertes financières directes résultant de certaines attaques telles que la fraude et les rançongiciels, en plus de pertes indirectes qui se manifestent par les coûts de réparation des dommages, l'érosion de la réputation des entreprises et la perte de confiance des clients.

Au niveau social, les cybercrimes Atteint à la vie privée et impactent l'état psychologique des victimes ; anxiété, stress et sentiments de menace constante.

Sur le plan politique, ils déstabilisent les gouvernements à travers la fuite des données confidentielles et les attaques sur les infrastructures.

Techniquement,malgré la gravité de ces menaces, elles ont cependant donné une puissante accélération de la course à l'innovation en cybersécurité.

1.2.7 Cybercriminalité et l'avenir

Évolution des menaces :

Les cyberattaques deviennent de plus en plus sophistiquées grâce à l'essor de l'intelligence artificielle, de l'IoT et des technologies block Chain.

Cyberguerre :

Les cyberattacques entre nations se multiplient à mesure que les tensions géopolitiques entre eux augmentent.

Rôle de l'intelligence artificielle :

Elle peut être en même temps une menace (attaques automatisées) et une solution (détection proactive des menaces).

1.3 Sécurité informatique

1.3.1 Définition

La sécurité informatique fait référence à l'ensemble des technologies, pratiques et stratégies mises en œuvre pour protéger les systèmes, les réseaux et les données contre les cyberattaques et les accès non autorisés.

1.3.2 Objectifs de la sécurité

Les principaux objectifs de la sécurité informatique sont :

L'intégrité :

s'assurer que les données ne sont pas altérées, modifiées ou détruites sans autorisation.

Disponibilité :

garantir que les systèmes, réseaux et services sont accessibles aux utilisateurs autorisés sans interruption.

Non-répudiation :

Ne pas nier avoir effectué une action (ex : un paiement, un envoi d'email).

Confidentialité :

Garantir que les informations sensibles ne sont accessibles qu'aux personnes autorisées.

Authentification :

Vérifier l'identité d'un utilisateur avant d'y accèder au système et aux données.

FIGURE 1.1 – Les objectifs de la sécurité

1.3.3 Évolution des Mécanismes de Sécurité

Sécurité Périmétrique (Pare-feu et Antivirus) :

Autrefois, la sécurité informatique était axée sur la protection du périmètre externe des systèmes, où l'objectif était de bloquer les menaces avant qu'elles n'atteignent le réseau.

parmi les outils utilisés :

Pare-feu (Firewall) :

surveillez et filtrez le trafic entre le réseau et les tiers pour empêcher les connexions malveillantes.

Antivirus :

Détecte et supprime les logiciels malveillants tels que les virus et les chevaux de Troie.

Bien que ces méthodes aient été efficaces à leur époque, elles sont devenues insuffisantes face aux attaques modernes qui exploitent les utilisateurs et les failles internes.

Sécurité Basée sur les Données :

Avec l'accélération de la transformation numérique et la dépendance croissante aux données, la cybersécurité est passée à la protection des données elles mêmes.

Chiffrement :

Transforme les données en code illisible sans clé de déchiffrement.

DLP (Data Loss Prevention) :

Empêche la fuite d'informations sensibles (blocage des e-mails contenant des données confidentielles).

1.3.4 Enjeux Actuels de la Sécurité Informatique

Avec l'évolution de l'infrastructure informatique, la dépendance croissante au cloud computing, la prolifération des appareils intelligents connectés (Internet des objets) et la multiplicité des centres de données,les systèmes deviennent plus compliqués, et Cette complexité a créé un grand défi dans la surveillance et la sécurisation de tous les points d'accès, encourageant les organisations à adopter des solutions de sécurité complètes et adaptables à différents environnements. Dans ce contexte, les cybercriminels ne cessent de développer leurs méthodes ; De nouvelles menaces apparaissent, telles que les attaques zero-day qui exploitent des vulnérabilités inconnues, le phishing et des ransomwares de plus en plus sophistiqués et répandus. Pour lutter contre ces menaces avancées, il est devenu crucial d'adopter un modèle de sécurité intelligent et réactif qui s'appuie sur des outils d'Intelligence Artificielle, permettant une détection précoce et une réponse rapide en temps réel à ces menaces.

1.4 Attaques informatiques

1.4.1 Définition

Une attaque informatique est une tentative visant à endommager un système, un réseau ou des données. Elle est menée par des cybercriminels, des pirates informatiques ou même par des employés au sein de l'entreprise. Ils exploitent souvent les vulnérabilités des logiciels ou le manque de sensibilisation des utilisateurs.

1.4.2 Catégories des attaques informatiques

Les attaques informatiques peuvent être divisées en deux grandes catégories principales selon leur impact sur les systèmes et leur mode opératoire.

Attaques passives :

visent à espionner les données secrètement pendant leur transfert à l'insu de la victime sans les altérer ni affecter le fonctionnement du système.

Attaques actives :

Elles impliquent la modification et suppression des données, ou bien la désactivation des systèmes et des applications, causant des dommages directs et évidents au système ciblé.

Type d'attaques	Objectifs	Exemples	Protection
Malware	Infecter, espionner, chiffrer	Virus, Ransomware	Antivirus, mise à jour
Attaques réseaux	Perturber, intercepter	DDoS, MITM	Pare-feu, VPN
Sur les données	Voler, modifier	Phishing, injection SQL	Chiffrement
Attaques logiques	Exploiter les failles	Zero-Day, Backdoor	Patchs de sécurité
Attaques physiques	Accès direct aux systèmes	Clé USB piégée, vol	Contrôle d'accès

TABLE 1.1 – Classifications de certaines attaques

1.5 Conclusion

Dans ce premier chapitre, nous avons initialement introduit les concepts fondamentaux de la cybercriminalité en examinant diverses attaques.

Par la suite, nous nous sommes concentrés sur les attaques et la sécurité informatiques en discutant leurs objectifs , leurs défis et de leurs caractéristiques.

Le chapitre suivant présentera en détail les attaques APT qui font objet de notre études en discutant de leurs cycles de vie et de l'utilisation de l'apprentissage automatique pour les détecter.

Chapitre 2

État de l'art sur l'application du L'IA pour la détection des menaces persistantes avancées.

2.1 Introduction

Dans le domaine de la cybersécurité informatique, la détection efficace des menaces persistantes avancées est considérée toujours comme un défi majeur.Ces attaques font partie des types d'attaques les plus complexes et les plus dangereuses, elles sont menées par des acteurs malveillants dotés de ressources et de capacités étendues dans le but d'infiltrer en permanence les systèmes pour accéder à des informations sensibles ou perturber des services vitaux.

À mesure que ces menaces évoluent,les approches traditionnelles de détection d'intrusion montrent leurs limite,d'où l'évolution vers des solutions plus intelligentes et plus efficaces notamment les techniques d'apprentissage automatique qui offrent des perspectives prometteuses pour renforcer la détection de ce type d'attaque.

Ce chapitre dresse un état de l'art détaillé sur l'application du Machine Learning à la problématique des APT. Après avoir présenté les concepts de base de ce dernier, nous analyserons les travaux récents les plus pertinents dans ce domaine,Les différentes stratégies suivies et les résultats obtenus seront également mis en évidence.

Enfin, Une discussion critique qui comprend une analyse des points forts et des points faibles des solutions actuelles, avec des suggestions d'amélioration future.

2.2 Advanced Persistent Threat

2.2.1 Définitions

Une attaque persistante avancée (APT) est un type de cyberattaque développée menée secrètement par un groupe de pirates informatiques hautement qualifiés, pendant une longue durée de temps, visant à compromettre les systèmes d'organisations gouvernementales ou privées et voler des informations sensibles sans être détecté.

L'appellation "Advanced Persistent Threat" a été introduite la première fois dans le secteur militaire, dont chaque terme signifie :

Advanced :

fait référence à l'ensemble des moyens avancés qui permettent aux acteurs de mener des attaques complexes, en exploitant les différentes vulnérabilités trouvées comme : zero-day, les courriels d'hameçonnage et les injections Structured Query Language (SQL) pour maximiser leurs chances d'intrusion dans le système cible et échapper à la détection.

Persistent :

Présence sur le réseau ciblé pour une langue durée (des mois ou des années) sans être détecté.

Threat :

reflète des intentions malveillantes souvent liées à l'espionnage, le sabotage ou la déstabilisation.

- Le tableau ci-dessus présente une comparaison de caractéristiques entre les attaques APT et les attaques traditionnelles

Critères	**Attaques traditionnelles**	**Attaques APT**
Objectifs	Déstabiliser, endommager un système, extorquer de l'argent	Espionnage, vol de données stratégiques, sabotage à long terme
Durée	Courte (quelques heures à quelques jours)	Longue (mois, voire années)
Ciblage	Large et souvent aléatoire (attaques de masse, campagnes de phishing)	Très ciblé (gouvernements, entreprises stratégiques, infrastructures critiques)
Méthodes d'attaque	Phishing, ransomware, virus, DDoS, brute-force	Exploits Zero-Day, ingénierie sociale, mouvement latéral, persistance furtive
Niveau de sophistication	Basique à modéré (outils accessibles aux cybercriminels)	Très avancé (développé par des groupes d'élite, souvent soutenus par des États)
Mode opératoire	Attaque rapide et destructrice	Infiltration lente et discrète, recherche d'informations précieuses
Dissimulation	Peu ou pas d'efforts pour masquer l'attaque	Techniques avancées pour rester caché (backdoors, rootkits, chiffrement des communications)

Propagation	Se propage rapidement (ex. virus, botnets, vers informatiques)	Progression furtive, mouvement latéral à l'intérieur du réseau
Impact	Dommages immédiats visibles (systèmes bloqués, demandes de rançon)	Dommages invisibles à court terme, mais critiques à long terme (vol d'informations, sabotage)
Exemples d'attaques	Attaque DDoS sur un site e-commerce, ransomware WannaCry	Stuxnet, SolarWinds
Détection	Facile avec un antivirus et un pare-feu	Difficile, nécessite une surveillance avancée
Mode de défense	Pare-feu, antivirus	Analyse comportementale, segmentation du réseau, détection avancée des anomalies

TABLE 2.1 – Comparaison entre une attaque traditionnelle et une attaque APT

2.2.2 Caractéristiques des APT

Les attaques APT contient plusieurs caractéristiques qui les rendent plus dangereuses et plus complexes que les traditionnelles,notamment :

Ciblage précis et stratégique :

Contrairement aux autres attaques telles que le phishing ou les ransomwares, les attaques APT ciblent soigneusement des organisations spécifiques, selon leur valeur stratégique. elles sont précédée par une période de collecte d'informations en se basant sur des sources publiques, des réseaux sociaux et des techniques OSINT (intelligence open source).

Persistance à long terme :

L'un des aspects les plus redoutables des APT est la capacité de rester dans le système cible pendant des mois, voire des années, sans être détectées. Pour y parvenir, les attaquants mettent en place des portes dérobées, le chiffrement des communications et l'utilisation de serveurs proxy pour masquer leurs traces.

Contournement des défenses :

Les APT s'appuient sur des techniques avancées pour éviter d'être détectées par les

systèmes de sécurité traditionnels comme les antivirus, les pare-feu et les IDS/IPS.

Mouvement latéral et escalade des privilèges :

Les attaquants se déplacent au sein du réseau pour accéder à d'autres systèmes discrètement, en utilisant des outils qui leur permettent d'obtenir des privilèges élevés (tels que les informations d'identification d'administrateur).

Exfiltration de Données sensibles :

L'objectif principal est de voler les informations sensibles, telles que les secrets commerciaux, les bases de données, les documents gouvernementaux ou les données militaires, qui sont ensuite transmises progressivement et de manière cryptée vers des destinations externes.

2.2.3 Cycle de Vie d'une APT

Une attaque APT suit généralement six grandes étapes :

Reconnaissance :

Les attaquants rassemblent des informations sur leur cible en analysant les employés, les logiciels utilisés, et les failles potentielles.

Infection Initiale :

L'attaque commence par une tentative d'infiltration, qui peut passer par un email frauduleux contenant un fichier infecté, une vulnérabilité logicielle. Une fois exécuté, le malware s'installe sur le système et ouvre un accès aux attaquants.

Établissement du Contrôle :

La mise en place d'un C2 (Command Control) par les attaquants , afin de pouvoir piloter l'attaque à distance. Ils installent des backdoors, pour capturer les données et surveiller l'activité du réseau.

Mouvement Latéral :

Utilisation des identifiants volés pour se propager discrètement dans l'entreprise victime et accédant progressivement aux serveurs les plus sensibles.

Exfiltration des Données :

Utilisation des techniques comme les tunnels DNS et les communications chiffrées afin d'exfiltrer discrètement les informations critiques identifiées.

Effacement des Traces :

Pour rendre la détection extrêmement difficile, les attaquants suppriment leurs logs,effacent les fichiers malveillants et désactivent les alertes de sécurité.

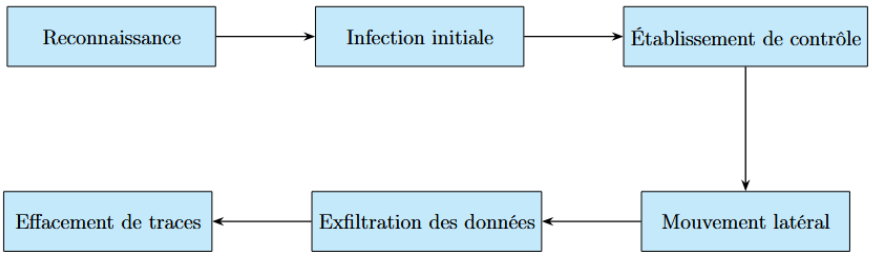

FIGURE 2.1 – Cycle de vie d'une attaque APT

2.2.4 Groupes APT Célèbres

Plusieurs groupes spécialisés dans l'exécution d'attaques (APT) sont liés à des gouvernements ou à des organisations criminelles sophistiquées. Parmi ces groupes on trouve :

APT28 (Fancy Bear) :

Un groupe d'origine Russe, il a été impliqué dans des attaques contre l'OTAN et l'élection présidentielle américaine.

APT29 (Cozy Bear) :

Un autre groupe russe, connu pour ses cyberattaques ciblant les institutions gouvernementales sensibles.

Groupe Lazarus :

Originaire de Corée du Nord, il a été impliqué dans des cyberattaques financières et des ransomwares.

Groupe Equation :

Considéré comme affilié à l'Agence de sécurité nationale américaine (NSA), spécialisé dans l'espionnage de haut niveau.

Stuxnet :

Il s'agit d'un ver informatique découvert en 2010 ,spécifiquement conçu pour saboter les infrastructures industrielles, il a été développé par les États-Unis et Israël pour cibler le programme nucléaire iranien.

2.2.5 Conséquences des APT

Les attaques APT peuvent avoir des conséquences catastrophiques pour les gouvernements et les entreprises.

Elles entraînent des fuites massives de données sensibles, menaçant la sécurité nationale et mettant en danger les infrastructures critiques et la stabilité économique.

Dans le secteur privé, ce type d'attaques peut entraîner des pertes financières importantes, une chute de confiance des investisseurs et une atteinte irréparable à la réputation d'une entreprise.

Les pirates informatiques peuvent également revendre les données volées sur le dark web, exposant ainsi les clients à la fraude ou à l'extorsion.

2.2.6 Moyens de Défense contre une APT

Les protections traditionnelles telles que les antivirus et les pare-feu ne suffisent plus à contrer les menaces APT sophistiquées. Il est plutôt devenu nécessaire d'adopter stratégie de défense multicouche comprenant :

Principe de confiance zéro :

ne jamais faire confiance automatiquement à un utilisateur ou à un appareil, même ceux qui sont internes.

Détection des comportements anormaux :

Surveillance des activités suspectes telles que des tentatives de connexion inhabituelles ou des connexions étranges.

Segmentation du réseau :

Limiter la propagation d'un attaquant s'il parvient à s'infiltrer, en séparant les parties du réseaux pour réduire les dommages.

Cryptage des données sensibles :

Protéger les informations volées en les rendant inexploitables.

Sensibilisation et formation des employés :

Réduire le risque d'attaques de phishing et ingénierie sociale en améliorant la sensibilisation des utilisateurs.

2.3 Lien entre Cybercriminalité, Sécurité Informatique et APT

La cybercriminalité comprend toutes les activités illégales commises à l'aide de la technologie numérique, du vol de données au sabotage des infrastructures. Parmi ces menaces, les APT (Advanced Persistent Threats) qui sont des attaques hautement sophistiquées et discrètes, et elles ciblent des entités stratégiques telles que les gouvernements, les grandes entreprises et les institutions sensibles.

Face à ces menaces, la sécurité de l'information constitue la première ligne de défense, englobant les technologies et stratégies permettant de protéger les systèmes d'information contre les cyberattaques, basé sur les principes fondamentaux de confidentialité, d'intégrité, de disponibilité et d'authentification.

Les APT utilisent des techniques de cybercriminalité telles que le phishing, les logiciels malveillants et les exploits zero-day pour s'infiltrer durablement dans un système dans le but de voler et exfiltrer des données sensibles . Il recourt également à des mouvements latéraux sur le réseau pour étendre la portée de l'intrusion, ce qui la rend difficile à surveiller et à détecter.

En réponse, Face à cela, les entreprises et gouvernements renforcent leur cyber sécurité avec des solutions avancées.

2.4 Inteligence artificielle

L'intelligence artificielle (IA) est un domaine de l'informatique et des mathématiques rassemblant un ensemble de techniques algorithmiques et de théories permettant de réaliser des machines qui imitent L'intelligence humaine , afin d'être capable de résoudre des problèmes complexes.

2.4.1 Classification de l'IA

IA faible

L'IA faible,ou autrement dit L'IA étroite, est une forme de L'IA conçue pour effectuer une tâche spécifique sans la capacité de penser ou d'apprendre au-delà de cette tâche. Cette dernière est largement présente dans le quotidien et est utilisé dans plusieurs applications

telles que les assistants numériques (comme Google Assistant), les voitures autonomes et d'autres systèmes intelligents. Malgré l'efficacité de l'IA faible mais elle ne possède pas de conscience ni de perception, elle s'appuie plutôt sur des algorithmes préprogrammés. À ce jour, toutes les applications d'IA disponibles, y compris les systèmes avancés comme 'ChatGPT', relèvent de ce type d'IA.

IA forte

L'IA forte consiste à développer une machine dotée de capacités cognitives similaires à celles des humains, capable de résoudre des problèmes nouveaux et inconnus de manière autonome. Bien que ce domaine soit au centre de nombreuses recherches scientifiques actives, il n'est pas encore disponible.

IA symbolique

C' est une approche basée sur une logique formelle pour le traitement de l'information et la prise de décision en manipulant des symboles et en appliquant des bases de connaissances structurées. Cette approche a été largement utilisée dans les systèmes experts, notamment dans le domaine médical pour fournir des diagnostics automatisés, ainsi que dans certains chatbots qui s'appuient sur des scripts préprogrammés. Cependant, l'IA symbolique est limitée par son manque de flexibilité : elle ne peut pas apprendre seule et doit être continuellement mise à jour par des experts humains. Cette rigidité a conduit à son déclin progressif au profit de nouvelles approches plus dynamiques.

IA connective :

L'IA connective, également connue sous le nom de réseaux neuronaux artificiels, s'inspire du fonctionnement du cerveau humain. Il se caractérise par sa capacité à traiter d'énormes quantités de données et à s'adapter à une variété de situations. Cette approche est aujourd'hui dominante et est à la base des avancées récentes en intelligence artificielle, notamment dans la reconnaissance faciale et la traduction automatique. contrairement à l'IA symbolique, l'IA connexionniste peut apprendre à partir d'exemples, ce qui la rend plus flexible et performante dans des tâches complexes.

2.4.2 Domaines d'applications de l'IA :

FIGURE 2.2 – Les domaines d'application de L'IA

2.5 Apprentissage automatique

2.5.1 Définition :

L'apprentissage automatique (Machine Learning) est un sous-domaine de l'intelligence artificielle (IA) qui permet aux machines d'apprendre à partir de données sans être explicitement programmées. Il repose sur des algorithmes capables d'identifier des modèles et de faire des prédictions ou prendre des décisions en fonction de nouvelles données.

2.5.2 Approches de l'apprentissage automatique :

Il existe trois grandes approches en apprentissage automatique : l'apprentissage supervisé, l'apprentissage non supervisé et l'apprentissage par renforcement.

Apprentissage supervisé :

L'apprentissage supervisé s'inspire des expériences précédentes pour produire des sorties de données. Le système s'entraîne sur un ensemble de données étiquetées, avec les informations qu'il est censé déterminer. Les données peuvent même être déjà classifiées de la manière dont le système est supposé le faire. L'apprentissage supervisé comprend des algorithmes de régression et des algorithmes de classification binaire ou multiclasses. [18]

Apprentissage non supervisé :

L'apprentissage non supervisé utilise une approche plus indépendante dans laquelle un ordinateur apprend à identifier des processus et des schémas complexes sans aucun guidage humain constant et rigoureux. C'est l'algorithme lui-même qui classe et analyse les données pour aboutir aux résultats corrects. Les algorithmes d'apprentissage non supervisé sont nombreux ; parmi eux, nous citons le K-means, l'analyse en composantes principales (PCA) et les règles d'association. [18]

Apprentissage par renforcement :

L'apprentissage par renforcement est une méthode qui permet d'entraîner des modèles d'IA de manière spécifique. Un agent ou un algorithme apprend des stratégies de manière autonome pour ainsi faire des choix de manière correcte sur la base des informations que l'agent pourrait percevoir.

[18]

2.5.3 Différentes étapes de l'apprentissage automatique :

Collecte des données :

La première étape du ML consiste à récupérer des données pertinentes pour entraîner un modèle. Ces données peuvent provenir de diverses sources, notamment des bases de données relationnelles (SQL), des bases NoSQL, des fichiers CSV, JSON ou Excel, ou encore des jeux de données. La qualité des données est cruciale : un modèle ne peut être efficace que si les données utilisées sont fiables, complètes et représentatives du problème étudié.

Exploration des données :

Une fois les données collectées, une exploration est primordiale, pour analyser leur structure, leurs tendances, les valeurs manquantes, les corrélations, etc..., et ceci en passant par des visualisations de données à travers des histogrammes, nuages de points, courbes

de densité, graphiques. Cette étape permet de comprendre le comportement des données et d'adapter les choix du modèle en conséquence.

Nettoyage des données :

Les données brutes contiennent souvent des erreurs qui peuvent nuire à la performance du modèle, et le nettoyage des données consiste à supprimer ou corriger ces erreurs, gérer les valeurs manquantes (soit en les supprimant, soit en les imputant avec des moyennes, des médianes ou d'autres méthodes), supprimer les doublons, et corriger les incohérences. Cette étape est essentielle pour obtenir des résultats précis et significatifs dans les analyses suivantes.

Transformation des données :

Les algorithmes de ML nécessitent souvent une transformation des données pour fonctionner correctement. Cela peut inclure l'encodage des variables catégorielles en utilisant le One-Hot Encoding ou le Label Encoding), la normalisation (Min-Max) ou la standardisation, pour garantir que les variables numériques sont comparables. De plus, si le jeu de données contient un grand nombre de variables, une réduction de dimensionnalité peut être appliquée pour réduire la complexité tout en conservant l'essentiel de l'information.

Division des données :

Pour évaluer la performance d'un modèle, il est crucial de diviser les données en plusieurs ensembles : un ensemble d'entraînement (généralement 70 à 80 % des données) utilisé pour entraîner le modèle, un ensemble de test (20 à 30 %) utilisé pour évaluer la performance finale et, parfois, un ensemble de validation qui permet d'ajuster les hyper paramètres qui ne sont pas appris par le modèle lui-même, mais qui influencent ses performances.

Entrainement du modèle :

Cette phase consiste à ajuster les paramètres du modèle pour minimiser l'erreur entre les prédictions obtenus du modèle et les résultats réels. Cette optimisation utilise généralement un ensemble d'entraînement comme l'algorithme de la descente de gradient, dans le but de rendre le modèle capable de généraliser ses connaissances pour de nouvelles données et obtenir des résultats satisfaisants sur des exemples inconnus.

Evaluation du modèle :

Aprèsl'étape d'entraînement, il est indispensable d'évaluer la performance du modèle sur l'ensemble de test. Différentes métriques d'évaluation sont utilisées en fonction du type

de problème. Pour la classification, des indicateurs comme l'accuracy (précision globale), le Recall, la Précision, le F1-score et la matrice de confusion. Pour les problèmes de régression, des métriques comme l'erreur quadratique moyenne (MSE) et le coefficient de détermination (R^2) permettent d'évaluer la qualité des prédictions. Une bonne évaluation est essentielle pour détecter les erreurs systématiques du modèle et identifier les éventuelles améliorations à apporter.

Sous-apprentissage (Underfitting) et Sur-apprentissage (Overfitting) :

Un modèle peut être trop simple ou trop complexe pour bien généraliser aux nouvelles données.

Sous-apprentissage (Underfitting) :

Cela se produit lorsque le modèle est trop simple pour capturer les tendances des données. Il présente une mauvaise performance aussi bien sur les données d'entraînement que sur celles de test. Ce problème peut être résolu en ajoutant plus de caractéristiques, en augmentant la complexité du modèle ou en prolongeant l'entraînement.

Sur-apprentissage (Overfitting) :

Ce problème survient lorsque le modèle s'adapte trop aux données d'entraînement, au point de perdre en capacité de généralisation. Il affiche une très bonne performance sur l'entraînement, mais une mauvaise performance sur les données de test.

2.5.4 Relation entre L'IA et le ML :

Le machine Learning et L'intelligence artificielle ne sont pas identiques.

L'IA est un terme générique qui englobe toutes les techniques visant à rendre les machines intelligentes.

quant au ML est l'une des nombreuses branches de l'intelligence artificielle où les machines apprennent à partir des données sans être explicitement programmées. Si le machine learning relève de l'IA, toutes les activités d'IA ne peuvent pas être appelées machine learning.

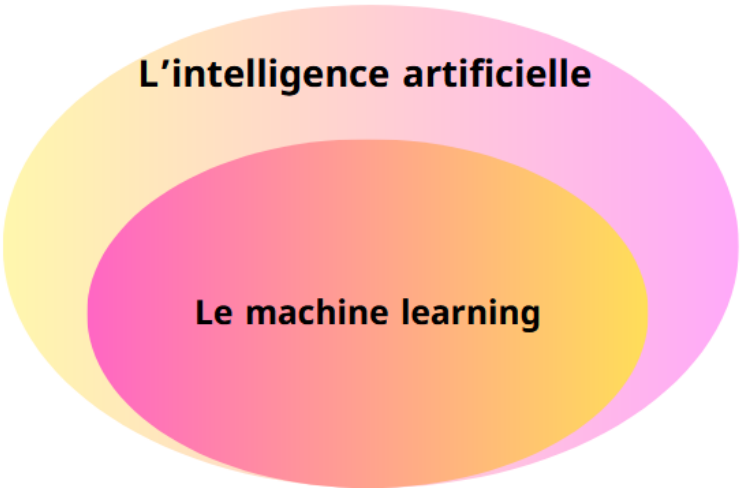

FIGURE 2.3 – Relation entre l'IA et ML

2.5.5 Algorithmes récents appliqués pour la détection des APT

La détection des attaques APT représente un défi majeur en cybersécurité, en raison de leur furtivité, de leur persistance, et de leur capacité à contourner les systèmes de sécurité traditionnels. Pour remédier à ce problème, la recherche s'est orientée vers l'intelligence artificielle qui se base sur l'identification des comportements anormaux et des motifs complexes dans les données système et réseau.

Random Cut Forest (RRCF) :

C'est un algorithme non supervisé, particulièrement adapté à la détection d'anomalies dans des séries temporelles, comme les journaux système. Il repose sur la construction de forêts de "coupes aléatoires" afin d'identifier les points de données inhabituels.

AdaBoost :

Un algorithme supervisé, qui combine plusieurs modèles faibles souvent des arbres de décision pour créer un classificateur robuste. Il est utilisé dans la détection des attaques APT à partir des données réseau, en identifiant si un paquet ou une session est normale ou malveillante. il est performant dans les environnements complexes et bruyants car il est apte à s'adapter et corriger aux erreurs des modèles précédentes.

Isolation Forest :

Il isole les observations en construisant des arbres de partition. Il a l'avantage d'être rapide et efficace, même avec de grandes quantités de données, ce qui le rend utile pour la détection des APT.

Autoencoders :

Les auto encodeurs sont des réseaux de neurones un peu particuliers qui possèdent exactement le même nombre de neurones sur leur couche d'entrée et leur couche de sortie. Le but pour un auto encodeur est d'avoir une sortie la plus proche de l'entrée !

L'apprentissage est donc « auto-supervisé » car la loss à minimiser est le coût de reconstruction entre la sortie et l'entrée. Les données n'ont donc pas à être étiquetées, car elles sont leurs propres étiquettes, ce qui fait donc de ce modèle un modèle non supervisé. [23]

Réseaux de neurones récurrents (RNN, LSTM) :

Grâce à leur capacité à modéliser des séquences temporelles, les RNN (Recurrent Neural Network) et les LSTM (Long Short-Term Memory), sont particulièrement efficaces pour analyser les journaux système ou les flux réseau tout en capturant les dépendances temporelles des événements.

Federated Learning (Apprentissage fédéré) :

L'apprentissage fédéré est un paradigme d'apprentissage automatique de pointe qui facilite la formation de modèles d'intelligence artificielle dans un réseau d'appareils ou de serveurs décentralisés. Cette approche permet à chaque nœud de contribuer au processus d'apprentissage du modèle en utilisant ses données locales sans qu'il soit nécessaire d'échanger ou de centraliser ces échantillons de données. Cette méthode permet l'apprentissage collaboratif tout en conservant les données brutes à la source, ce qui répond aux préoccupations en matière de protection de la vie privée et de sécurité. [24]

2.6 Synthèse des travaux connexes récents pour la détection des APT

La figure ci-dessus présente une taxonomie des principales méthodes récentes de détection des APT utilisant , classées en deux catégories distinctes : les approches basées sur les logs systèmes et celles axées sur l'analyse du trafic réseau et l'apprentissage fédéré. Cette classification permet de comprendre les différentes stratégies adoptées par les chercheurs pour aborder ce problème complexe et met en évidence l'évolution des techniques de détection au fil du temps.

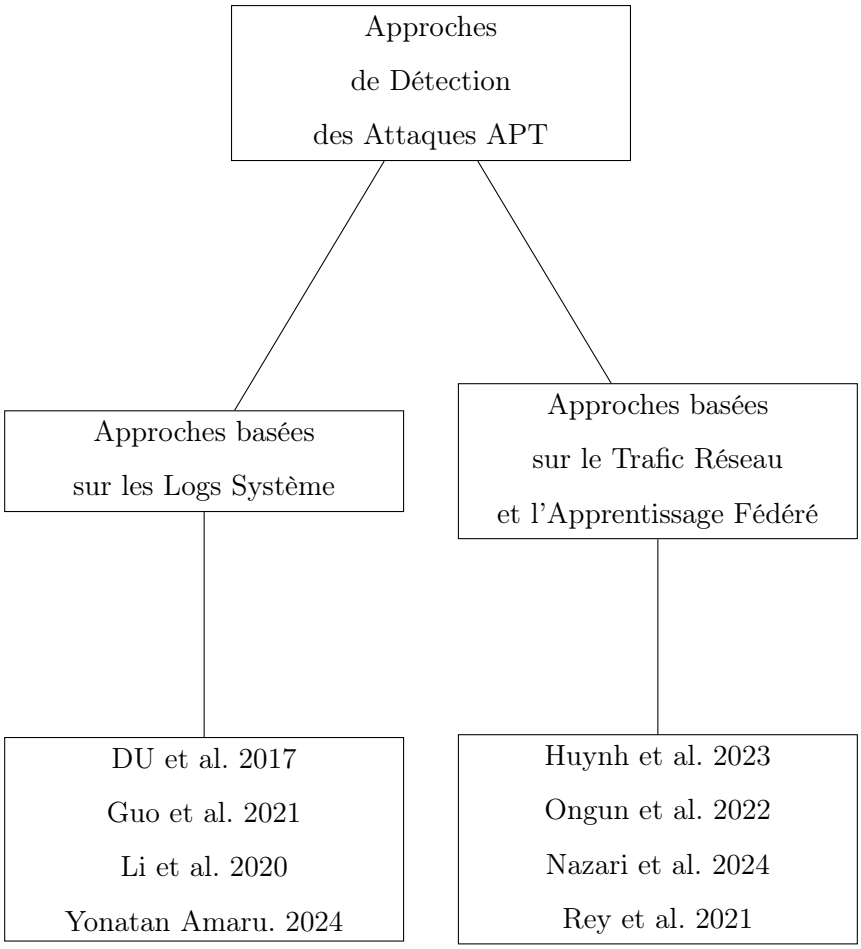

FIGURE 2.4 – Taxonomie des approches de détection des attaques APT.

2.6.1 Méthode RAPID : Détection Basée sur les Logs Système

Historique et Contexte :

Le système RAPID (Robust APT Detection and Investigation using Context-Aware Deep Learning) se repose sur l'analyse des journaux système (logs) pour identifier des comportements anormaux en temps réel, il a été conçu pour répondre aux faiblesses des (IDS) classiques, en particulier ceux qui reposent sur des signatures statiques. son développement a été proposé en 2024 par une équipe de chercheurs de l'Université Ben-Gurion du Néguev, dirigée par Yonatan Amaru, dans un contexte où les méthodes de détection comportementale gagnaient en importance face aux limitations des systèmes basés sur des signatures. L'idée principale derrière RAPID est d'utiliser un entraînement supervisé/auto-supervisé, notamment avec des Bi-LSTM.

Principe de Fonctionnement :

le système RAPID collecte et analyse en temps réel les logs générés par les systèmes

d'exploitation afin d'identifier toute activité anormale ou potentiellement malveillante. le type de données traité est :

-THEIA (graph-based provenance dataset) :Système open-source pour la capture et l'analyse des logs système sous forme de graphe, qui inclut des scénarios réalistes d'APT simulés dans des environnements Linux et Fournit des événements système détaillés : créations de fichiers, connexions réseau, exécutions de processus, etc.

- DARPA TC (plus anciens) : Utilisé pour valider la généralisation de RAPID dans des contextes variés.

Avantages et Limites :

Avantages :

- Capable de traiter jusqu'à 36 000 logs/seconde sur certaines machines, grâce à des optimisations.

- Il peut détecter des attaques lentes et fragmentées, typiques des APT.

Limites :

- Principalement testé sous Linux, la portabilité vers Windows ou d'autres OS nécessite des adaptations importantes.

- Se base uniquement sur les logs système, ce qui rend aveugle aux attaques sur les réseaux externes ou anomalies de trafic.

Classification des travaux basés sur la détection des APT basés sur les logs systèmes :

Auteurs	**Année**	**Dataset**	**Modèle utilisé**	**Résultats (ACC, FPR, TPR, FNR)**
Du et al.[26]	2017	Logs système (HDFS, BGL, etc.)	LSTM (DeepLog)	Precision = 96.6%, Recall = 92.6%, F1-score = 94.6%
Guo et al. [27]	2021	HDFS logs	BERT (NLP + Transformer)	Precision = 98.9%, Recall = 98.3%, F1 = 98.6%
Yonatan Amaru.[28]	2024	Journaux système réels (production + test)	Bi-LSTM	amélioration > +8 % du taux de détection

TABLE 2.2 – Tableau récapitulatif des travaux de détection des APT basés sur l'analyse des logs système

2.6.2 Méthode XFedHunter (eXplainable Federated Hunter) : Détection Basée sur le Trafic Réseau et l'Apprentissage Fédéré

Historique et Contexte :

XFedHunter a été introduite en 2024 pendant le besoin de détecter les attaques APT de manière collaborative et distribuée sans partager les données sensibles. C'est une approche qui utilise l'apprentissage fédéré, qui s'agit d'une technique qui permet à plusieurs organisations d'entraîner un modèle localement, puis de partager uniquement les mises à jour du modèle. Cela garantit la confidentialité des informations sensibles.

Principe de Fonctionnement :

XFedHunter fonctionne en analysant le trafic réseau pour détecter des comportements anormaux liés aux attaques APT Où plusieurs organisations entraînent localement des modèles de détection, sans échanger leurs données sensibles , en partageant uniquement les paramètres appris avec un serveur central qui à son tour agrège les contributions de tous les participants pour introduire un modèle global et enfin le redistribué à chaque organisation , ce processus est répété jusqu'à l'obtention d'un modèle performant et optimal.

Cette méthode permet la collaboration d'une manière efficace à la détection des APT tout en assurant la confidentialité et la résilience du système face aux cyber menaces en général et aux APT spécialement.

Les données traitées incluent :

- Adresses IP suspectes et connexions répétées.
- Anomalies dans les flux réseau (fréquence et volume inhabituels).
- Exfiltrations de données via des canaux non autorisés.

2.6.3 Avantages et Limites :

Avantages :

- Détection améliorée grâce au partage des modèles entre organisations.
- Identifie les attaques réseau avancées et furtives.

Limites :

- Nécessite une infrastructure plus complexe.
- Peut rencontrer des problèmes de latence et de synchronisation des modèles.

Classification des travaux basés sur la détection des APT basés sur le Trafic Réseau et l'Apprentissage Fédéré

Auteurs	**Année**	**Titre de l'étude**	**Modèles ML utilisés**	**Résultats (ACC, FPR, TPR, etc.)**
Huynh et al.[29]	2023	*XFedHunter : An Explainable Federated Learning Framework for APT Detection in SDN*	GNN + CNN + GRU + SHAP	ACC = 92.8%, FPR = 3.4%, TPR = 91.2%, F1 = 93.0%
Ongun et al.[30]	2022	*CELEST : Federated Learning for Globally Coordinated Threat Detection*	Active Learning + FL	ACC = 91.6%, FPR = 4.1%, Precision = 90.3%, Recall = 89.9%
Nazari et al[31].	2024	*P3GNN : Privacy-Preserving Provenance Graph Model for APT Detection*	GCN + FL + chiffrement homomorphe	ACC = 93.4%, FPR = 6.2%, TPR = 94.1%, Precision = 92.8%
Rey et al.[32]	2021	*Federated Learning for Malware Detection in IoT Devices*	MLP, Autoencoder	ACC = 90.1%, FPR = 4.6%, F1 = 89.5%, Recall = 88.9%

TABLE 2.3 – Tableau récapitulatif des traveaux basés sur le trafic réseau etl'apprentissage fédéré

Discussion des travaux existants :

Dans cette étude, j'ai choisi de me concentrer sur deux approches récentes et complémentaires pour détecter les attaques APT en utilisant le ML : RAPID et XFedHunter.

RAPID se base sur l'analyse des journaux système pour repérer les comportements suspects. Elle est particulièrement efficace dans des systèmes individuels bien surveillés, car elle permet une détection rapide à partir des événements enregistrés localement. Cependant, son principal point faible, c'est qu'elle fonctionne mieux dans des environnements isolés. Dès qu'on est face à des attaques distribuées ou à des systèmes complexes, RAPID montre certaines limites : elle dépend fortement de la centralisation des données, est conçue principalement pour les systèmes Linux, et montre des faiblesses face à des

attaques distribuées ou des architectures complexes.

XFedHunter, de son côté, propose une autre vision. Elle mise sur une collaboration entre plusieurs entités grâce à l'apprentissage fédéré. Concrètement, chaque organisation entraîne un modèle en local à partir de son propre trafic réseau, sans avoir besoin de partager ses données personnelles. C'est un avantage énorme en matière de confidentialité et protège de la vie privée.

De plus, les algorithmes utilisés (comme Random Forest, Isolation Forest et K-Means) permettent de bien classer les connexions, repérer les comportements rares et mettre en évidence les flux anormaux. Mais cette approche a aussi ses contraintes : elle demande une infrastructure plus lourde, une bonne coordination entre les différents sites et une gestion complexe des mises à jour des modèles partagés.

Au final, XFedHunter convient mieux aux environnements distribués et sensibles à la confidentialité, tandis que RAPID semble mieux adapté aux systèmes isolés qui veulent une détection locale rapide.

Par conséquent, il est justifié que cette dernière approche basée sur l'analyse des logs systèmes vaut la peine d'être explorée en détail et de poursuivre des recherches dans ce domaine dans le but de l'améliorer et explorer pleinement son potentiel.

2.7 Conclusion

Dans ce chapitre, j'ai examiné plusieurs travaux avancés dans le domaine de la détection des menaces persistantes avancées. J'ai constaté que l'approche Rapid offre des perspectives intéressantes pour retracer le déroulement complet d'une attaque APT, tandis que XfedHunter semble prometteuse et a donné lieu à des résultats satisfaisants en terme de distrbuvité et confidentialité. Cependant, Rapid apparait comme particulièrement pertinente à approfondir. Malgré des performances encourageantes, cette approche souffre de limites liées à : la centralisation des données, conçue principalement pour les systèmes Linux, et montre des faiblesses face à des attaques distribuées ou des architectures complexes. Ma contribution se concentrera sur la conception d'un système de détection des APT, en se focalisant sur une combinaison d'une détection des attaques APT en analysants les logs systèmes (inspiré de Rapid) avec une méthode de détection en analysants le trafic réseau tout en améliorant davantage les performances du modèle.

Chapitre 3

Contribution,Résultats et Discussion

3.1 Introduction

Dans ce chapitre, nous présentons en détail notre contribution visant à relever les défis de la détection des menaces persistantes avancées en développant l'approche RAPID qui s'appuie sur l'analyse des Logs système.

Comme identifié dans l'état de l'art, cette approche offre un potentiel prometteur, mais souffre encore de limites, notamment liées à l'adoption aux environnements distribués.

notre méthodologie se concentre sur le développement d'un modèle d'apprentissage machine hybride qui reprend les fondements de RAPID,et les combine avec une analyse complémentaire du trafic réseau. En fusionnant ces deux méthodes via une régression logistique, le modèle devient plus robuste et plus performant et donne des résultats plus précis.

Ce chapitre détaillera l'ensemble du processus de conception, d'implémentation et d'évaluation de ma solution. Je présenterai l'architecture générale proposée, une description des jeux de données exploités ainsi que les outils et environnements utilisés.

nous expliquerons ensuite en profondeur les différentes étapes clés telles que le prétraitement de données, la construction de modèle et son entraînement.

Enfin, nous analyserons les performances obtenues en les comparant aux approches existantes.

3.2 Motivation et Objectifs

La principale motivation réside dans la volonté de relever les défis majeurs de la détection des menaces persistantes avancées dans les environnements critiques tels que les administrations, les institutions militaires ou les grandes entreprises, notamment la capacité d'échapper aux mécanismes classiques de détection, exploitation des failles sur de longues périodes tout en minimisant les signes apparents d'intrusion.

Face à cette problématique, la méthode RAPID a proposé une approche innovante en s'appuyant sur l'analyse des journaux système pour détecter des comportements anormaux liés à des APT. Cependant, se focaliser uniquement sur les logs système limite la portée de la détection, car certaines phases d'une APT (comme l'exfiltration de données ou le mouvement latéral) peuvent laisser des traces plus visibles dans le trafic réseau.

Notre objectif est de proposer une approche hybride, qui non seulement reprend les fondements de RAPID, mais les enrichit par une analyse complémentaire du trafic réseau et un fusionnement de ces deux sources d'information via une régression logistique,pour atteindre une détection efficace et robuste.

3.3 Approche Proposée

Notre proposition consiste à exploiter deux sources de données complémentaires (logs système et trafic réseau) afin de développer un modèle d'apprentissage automatique pour la détection des attaques APT. Pour cela, nous avons suivais une méthodologie rigoureuse en plusieurs étapes.

Étape 1 : Prétraitement des logs système (Détecteur A avec RRCF)

Tout d'abord, nous avons procédé à un prétraitement approfondi des journaux système collectés localement à partir de notre ordinateur, rassemblés dans un fichier CSV nommé system_logs.csv

Ce prétraitement comprend la gestion des valeurs manquantes et infinies, ainsi que l'encodage des colonnes catégorielles par one-hot encoding.

Certaines colonnes textuelles ont également été analysées afin d'extraire des caractéristiques pertinentes, notamment la détection de mots-clés liés aux erreurs.

Ensuite, nous avons sélectionné un ensemble réduit de caractéristiques discriminantes

telles que : ErrorKeyword, IsError, IsWarning et IsInformation. Ces variables ont été normalisées pour garantir une échelle homogène.

Le cœur de cette étape repose sur l'algorithme RRCF (Robust Random Cut Forest), particulièrement adapté à la détection d'anomalies dans des séries temporelles ou des données non supervisées. Une forêt de 100 arbres a été construite, chaque arbre apprenant sur un sous-échantillon aléatoire. Pour chaque enregistrement, un score d'anomalie est calculé en utilisant la codispersion, puis normalisé entre 0 et 1.

Étape 2 : Prétraitement du trafic réseau (Détecteur B avec AdaBoost)

Parallèlement, nous avons exploité les données du jeu CIC-IDS2017, qui ont été collectées depuis plusieurs fichiers CSV et fusionnés en un seul DataFrame. Les valeurs manquantes ont été traitées et les colonnes catégorielles (hors Label) ont été encodées.

Ensuite nous avons sélectionné trois caractéristiques clés du trafic réseau : Flow Duration, Total Fwd Packets, et Total Backward Packets. Le label binaire a été défini comme 0 pour le trafic bénin (BENIGN) et 1 pour toute autre attaque. Les données ont été normalisées, puis divisées en deux ensembles : entraînement et test.

L'algorithme choisi pour cette tâche est AdaBoost, une méthode d'apprentissage supervisé, efficace dans les problèmes de classification binaire. Le modèle a été entraîné sur l'ensemble d'apprentissage, puis a généré un score pour chaque exemple de test, représentant la probabilité qu'il s'agisse d'une attaque.

Étape 3 : Fusion dynamique des scores via Régression Logistique

Afin de renforcer la robustesse de détection,nous avons adopté à une approche de fusion dynamique des deux détecteurs qui consiste à combiner les scores normalisés produits par les deux modèles en une matrice de fusion, chaque ligne représentant une paire de scores (log système, trafic réseau).

Cette matrice a été utilisée comme entrée pour une régression logistique, technique choisie pour sa capacité à effectuer une combinaison pondérée de variables continues. Elle a été entraînée sur les sorties des deux détecteurs, en se basant sur les vrais labels d'attaque issus des données réseau.

Les scores finaux ainsi produits représentent une probabilité plus affinée d'APT, tenant compte de deux sources hétérogènes d'information. Un seuil de décision empirique (0.246728) fixé après plusieurs itération a été appliqué pour distinguer les cas normaux des attaques APT.

Évaluation et perspectives

Pour évaluer les performances de notre système, nous avons utilisé des métriques classiques telles que la courbe ROC & AUC ainsi que la Matrice de Confusion. Ces indicateurs permettent de quantifier la capacité du modèle à détecter correctement les attaques.

Cette approche hybride offre une vision complémentaire et enrichie de l'environnement système et réseau, rendant la détection d'APT plus précise que l'utilisation d'une seule source de données.

3.4 Description du jeu de données

La qualité, la diversité et la représentativité des données influencent directement la performance et la fiabilité de la détection des attaques APT d'où Le choix d'un jeu de données pertinent joue un rôle crucial dans la conception du modèle de machine learning. Dans ce cadre,nous avons adopté à la combinaison de deux sources de données complémentaires : les journaux système de Windows collectés localement, et le jeu de données CICIDS2017, largement connu dans la communauté de la cybersécurité.

Journaux système Windows

Les Event Logs de Windows contiennent des informations riches sur l'activité du système d'exploitation, des utilisateurs et des applications.

Dans notre approche, nous avons extrait ces événements directement depuis la machine personnelle via PowerShell, en ciblant principalement le journal sécurité, fait pour enregistrer les événements critiques liés à la sécurité (authentifications, élévation de privilèges, exécution de services, etc...).

La commande utilisée pour exporter ces événements au format CSV depuis l'ordinateur est :

```
$env:USERPROFILE\Documents\Security_logs.csv -NoTypeInformation
```

Le fichier obtenu comporte un ensemble de colonnes clés notamment :

TimeCreated (horodatage de l'événement),

Id (identifiant de l'événement),

Level (niveau de sévérité),

ProviderName (source génératrice),

UserId (utilisateur concerné),

Message (description textuelle de l'événement),

des métadonnées comme Keywords, ActivityId ou Bookmark.

Ces journaux permettent de détecter des activités malveillantes associés aux attaques APT, comme des changements inhabituels dans les services, des erreurs récurrentes ou des comportements anormaux du système.

Ils ont été exploités dans notre détecteur par le détecteur A, basé sur l'algorithme RRCF (Robust Random Cut Forest), afin d'identifier des anomalies sans supervision.

CICIDS2017

Pour l'analyse du trafic réseau nous avons utilisé le jeu de données CICIDS2017 (Canadian Institute for Cybersecurity Intrusion Detection System 2017), reconnu pour sa richesse en scénarios d'attaques récents et sa structure réaliste, adaptée aux environnements réseau modernes. Ce dataset a été conçu par le Canadian Institute for Cybersecurity (CIC) pour objectif de simuler le comportement des utilisateurs légitimes et des attaquants dans un réseau d'entreprise typique.

La collecte des données s'est déroulée en cinq jours (du 3 au 7 juillet 2017) où chaque jour a était dédié à un type d'activité ou d'attaque spécifique. Le trafic généré comprenait à la fois des comportements normaux (navigation web, transfert de fichiers, courriels, etc.) et une variété d'attaques sophistiquées telles que :

les attaques par force brute (FTP, SSH).

les attaques par déni de service (Hulk, GoldenEye, Slowloris, Slowhttptest).

les attaques par botnet.

les infiltrations dans le réseau,

le port scanning.

les attaques Web (XSS, SQL Injection).

l'exploitation de vulnérabilités comme Heartbleed.

Chaque flux est associé à une étiquette (label) qui indique s'il s'agit d'un trafic BENIGN ou d'un type d'attaque spécifique. Cela permet son utilisation dans les algorithmes d'apprentissage supervisé.

Dans cette approche, nous avons exploité ce jeu de données à deux niveaux :

pour l'entraînement et l'évaluation du détecteur B, chargé de classifier le trafic réseau à l'aide d'AdaBoost,

pour superviser la fusion des résultats via régression logistique, en utilisant les éti-

quettes connues pour apprendre à combiner les scores issus du détecteur A (logs système) et du détecteur B (réseau).

Grâce à la diversité des attaques présentes dans CICIDS2017, nous avons constaté qu'il est particulièrement adapté pour modéliser des situations réalistes et complexes, telles que celles rencontrées lors d'attaques APT.

3.5 Outils de développement

3.5.1 Matériel

Les principales caractéristiques de la machine utilisée pour l'implémentation sont :

Nom de l'appareil :

DESKTOP-V0S9NNT

Caractéristiques :

Processeur :Architecture x64

Mémoire vive (RAM) : 4 Go (3,87 Go utilisable)

Type du système Système : d'exploitation 64 bits, processeur x64

Système d'exploitation :Windows 10 Professionnel, version 22H2

3.5.2 Langage, Logiciels et bibliothèques

Dans le cadre de notre mémoire nous avons utilisé plusieurs outils notamment :

Python :

un langage de programmation multi-paradigme, multiplateformes et orienté objet, il offre de nombreuses bibliothèques additionnelles pour divers domaines comme l'intelligence artificielle, data science et cybersécurité, des structures de données efficaces Il a été choisi pour sa richesse en bibliothèques scientifiques, sa simplicité de syntaxe et sa large communauté. [35]

FIGURE 3.1 – python

[36]

Visual Studio Code (VS Code) : Éditeur de texte open-source développé par Microsoft, offrant des fonctionnalités avancées telles que la coloration syntaxique, la complétion automatique, le débogage intégré et la gestion de projets, Son intégration fluide avec l'interpréteur Python et ses extensions facilitent grandement le développement. [37]

FIGURE 3.2 – vscode

[38]

Bibliothèques :

Bibliothèque	**Version**	**Fonction principale**
`pandas`	2.2.3	Manipulation et analyse de données tabulaires (logs, CSV)
`numpy`	2.1.3	Traitement numérique, opérations matricielles
`matplotlib`	3.10.0	Visualisation graphique de données
`seaborn`	0.13.2	Graphiques statistiques avancés, liés à pandas
`scikit-learn`	1.5.2	Classification, AdaBoost et régression logistique
`rrcf`	0.4.4	Détection d'anomalies par forêts aléatoires (Random Cut Forest)
`scipy`	1.14.1	Fonctions scientifiques et statistiques avancées
`joblib`	1.4.2	Sérialisation et gestion de pipelines de modèles
`pillow`	11.0.0	Traitement d'image (affichage ou export de graphiques)
`python-dateutil`, `pytz`, `tzdata`	2024.2	Gestion du temps et des fuseaux horaires

TABLE 3.1 – Bibliothèques Python utilisées

Autre Outils :

Invite de commandes (CMD) : Outil de ligne de commande intégré à Windows, il a été utilisé pour collecter les logs systèmes.

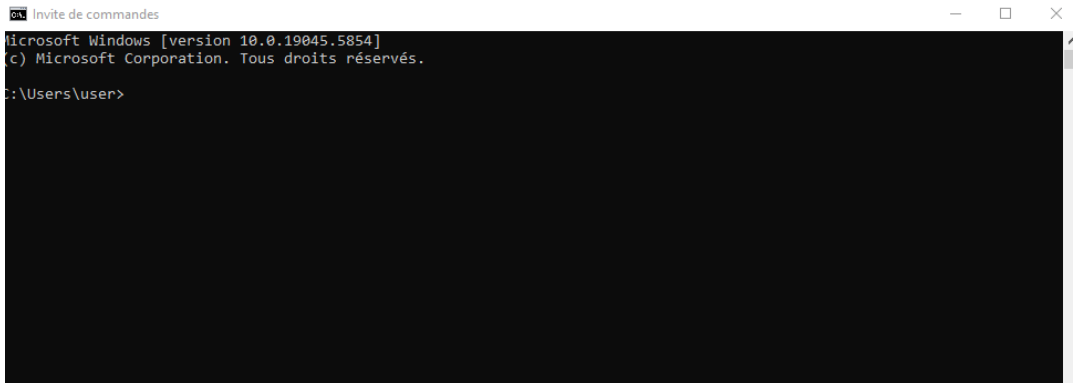

FIGURE 3.3 – invite de commande

Microsoft Excel : Logiciel tableur utilisé pour visualiser,analyser, nettoyer et structurer les jeux de données (Fichiers csv).

FIGURE 3.4 – Excel [39]

3.6 Implémentation

3.6.1 Détecteur A (Log système)

Préparation de données

Le fichier de logs système exploité est structuré en plusieurs colonnes, regroupées en deux catégories principales : les colonnes génériques et les colonnes supplémentaires, qui incluent des textes libre, des identifiants numériques, des niveaux de sévérité, des timestamps, etc.

Le tableau ci-dessous présente une description détaillée des principaux attributs du fichier de log utilisé.

N°	**Nom de l'attribut**	**Type**	**Description**
1	Message	Texte libre	Contenu descriptif de l'événement (ex. "Le service X a été arrêté").
2	Id	Numérique	Code identifiant l'événement (ex. 7040, 16, etc.).
3	Level	Numérique	Niveau de gravité (2 = Erreur, 3 = Avertissement, 4 = Information, etc.).
4	ProviderName	Nominal	Source de l'événement (ex. Service Control Manager, WindowsUpdateClient).
5	TimeCreated	Date/Heure	Timestamp de génération de l'événement.
6	UserId	Nominal	Identifiant de l'utilisateur ou du compte système associé à l'événement.
7	LogName	Nominal	Nom du journal source de l'événement (généralement *System*).

TABLE 3.2 – Description des attributs du fichier journal système

En plus de ces colonnes principales, le fichier contient également des colonnes supplémentaires, telles que :

Keywords, ActivityId, Bookmark, etc., qui fournissent des métadonnées supplémentaires.

Ces colonnes additionnels, Souvent moins exploités, peuvent fournir des informations pertinentes dans le cadre d'analyses approfondies ou pour la corrélation d'événements au sein d'une session ou entre différents services.

Chargement de données :

Durant cette étape, nous avons importé les événements système à partir d'un fichier CSV nommé `system_logs.csv`, puis nous avons utilisé la bibliothèque Pandas pour lire ce fichier et le convertir en un DataFrame, facilitant ainsi son exploration et sa manipulation.

```python
df_system = pd.read_csv(r"system_logs.csv", encoding="utf-8", sep=",")
print("Aperçu initial des logs système:")
print(df_system.head())  # Afficher les 5 premières lignes du DataFrame
print("Colonnes initiales:", df_system.columns.tolist())  # Affiche les noms des colonnes
```

FIGURE 3.5 – chargement des Logs

Nettoyage et encodage de données :

Durant cette étape, nous avons commencé par remplacer les valeurs infinies présentes dans les logs par des valeurs manquantes (NaN), afin d'éviter d'éventuelles erreurs lors de l'analyse, ensuite ces valeurs manquantes ont été imputées en utilisant la technique du forward fill (ffill), qui consiste à les remplir avec la dernière valeur valide précédente.

Par la suite, nous avons identifié les colonnes de type catégoriel, à l'exception de "Message" qui contient du texte libre peu exploitable tel quel. Ces colonnes ont été numérisé par l'encodage "one-hot".Le paramètre "drop_first=True" a été utilisé pour éviter la redondance des colonnes issues de l'encodage.

```python
df_system.replace([np.inf, -np.inf], np.nan, inplace=True)
df_system = df_system.ffill()  # Remplit les valeurs NaN avec la dernière valeur valide connue
# Identification des colonnes catégorielles à encoder (sauf la colonne 'Message' qui est du texte libre)
cat_cols_system = df_system.select_dtypes(include=['object']).columns.tolist()
cat_cols_system = [col for col in cat_cols_system if col not in ['Message']]
print("Colonnes catégorielles à encoder:", cat_cols_system)

# Encodage des colonnes catégorielles en variables numériques via one-hot encoding
df_system_encoded = pd.get_dummies(df_system, columns=cat_cols_system, drop_first=True)
```

FIGURE 3.6 – nettoyage et encodage des Logs

Extraction de caractéristiques

Dans cette étape, nous avons enrichi les données brutes par la création de nouvelles variables explicites à partir des logs système :

ErrorKeyword : indique si le message système contient les mots “échec” ou “erreur”.

IsError : vaut 1 si le niveau de l'événement est une erreur (niveau 2).

IsWarning : vaut 1 pour un avertissement (niveau 3).

IsInformation : vaut 1 pour un événement informatif (niveau 4).

Ces variables permettent de résumer l'importance et la nature des événements système sous forme numérique, facilitant ainsi leur traitement par des algorithmes de détection.

```python
df_system.replace([np.inf, -np.inf], np.nan, inplace=True)
df_system = df_system.ffill()  # Remplit les valeurs NaN avec la dernière valeur valide connue
# Identification des colonnes catégorielles à encoder (sauf la colonne 'Message' qui est du texte libre)
cat_cols_system = df_system.select_dtypes(include=['object']).columns.tolist()
cat_cols_system = [col for col in cat_cols_system if col not in ['Message']]
print("Colonnes catégorielles à encoder:", cat_cols_system)

# Encodage des colonnes catégorielles en variables numériques via one-hot encoding
df_system_encoded = pd.get_dummies(df_system, columns=cat_cols_system, drop_first=True)
```

FIGURE 3.7 – extraction de de caractèristiques

Normalisation de caractéristiques

Afin d'assurer que toutes les caractéristiques aient une échelle comparable, une normalisation standard a été appliquée à l'aide de l'outil "StandardScaler" de la bibliothèque "scikit-learn".

Ce processus est illustré dans la figure ci-dessous.

```python
scaler_system = StandardScaler()
X_system_scaled = scaler_system.fit_transform(X_system)
```

FIGURE 3.8 – normalisation de caractéristiques

Construction du détecteur

Pour la détection d'anomalies dans les événements système, nous avons opté pour l'algorithme Robust Random Cut Forest (RRCF) qui est non supervisé, particulièrement efficace pour les données non étiquetées.

Dans un premier temps, une forêt constituée de plusieurs arbres RRCF est construite. Chaque arbre est généré à partir d'un sous-échantillon aléatoire des données prétraitées et normalisées. Ensuite, pour chaque point de données, un score de codispersion (codisp) est calculé.

Ce score mesure la manière dont un point perturbe la structure de l'arbre : plus un point est isolé, plus son score est élevé, ce qui suggère un comportement potentiellement anormal.

Le score d'anomalie final d'un échantillon est obtenu en faisant la moyenne de ses scores de codispersion sur tous les arbres de la forêt. Cette méthode permet une évaluation robuste en exploitant la redondance et la diversité des arbres.

Le choix de RRCF est motivé par sa capacité à gérer efficacement les données en

grande dimension, à détecter des anomalies sans nécessiter de labels, et à offrir une bonne interprétabilité grâce à sa structure en arbres.

```python
# Détection d'anomalies avec l'algorithme RRCF
n_system = X_system_scaled.shape[0]  # Nombre total d'échantillons
num_trees = 100    # Nombre d'arbres dans la forêt aléatoire
tree_size = 256    # Nombre maximal de points par arbre

# Construction de la forêt d'arbres RRCF
forest = []
for i in range(num_trees):
    sample_size = min(tree_size, n_system)  # Taille de l'échantillon à utiliser
    sample_idx = np.random.choice(n_system, size=sample_size, replace=False)  # Sélection aléatoire d'échantillons
    tree = rrcf.RCTree(X_system_scaled[sample_idx])  # Construction d'un arbre
    forest.append(tree)  # Ajout de l'arbre à la forêt

# Calcul des scores d'anomalie pour chaque échantillon
anomaly_scores = np.zeros(n_system)
for tree in forest:
    for leaf in tree.leaves:
        anomaly_scores[leaf] += tree.codisp(leaf)  # Score basé sur la codispersion
anomaly_scores /= num_trees  # Moyenne des scores pour normalisation
```

FIGURE 3.9 – Détection par RRCF

Normalisation des résultats

Les scores d'anomalie produits par la forêt RRCF ont été normalisés dans l'intervalle [0, 1] afin de permettre une comparaison plus simple et de faciliter l'interprétation. Une valeur proche de 1 indique une forte probabilité d'anomalie, tandis qu'une valeur proche de 0 indique un comportement considéré comme normal.

```python
# Normalisation des scores d'anomalie entre 0 et 1
system_scores = (anomaly_scores - anomaly_scores.min()) / (anomaly_scores.max() - anomaly_scores.min())
print("scores d'anomalie (Logs Système):")
print(system_scores[:10])  # Affichage des 10 premiers scores
```

FIGURE 3.10 – Normalisation des résultats

3.6.2 Détecteur B trafic réseau (CICIDS2017)

Préparation des données

Le jeu de données CICIDS 2017 contient des enregistrements de flux réseau décrits à l'aide de plusieurs attributs. Ces attributs peuvent être de type numérique, binaire, ou nominal, et décrivent diverses caractéristiques des connexions, telles que leur durée, leur direction, leur taille, ainsi que le comportement du trafic.

N°	**Nom de l'attribut**	**Type**	**Description**
1	Flow Duration	Numérique	Durée totale du flux en microsecondes.
2	Protocol	Nominal	Protocole utilisé dans la connexion (TCP, UDP, ICMP).
3	Total Fwd Packets	Numérique	Nombre total de paquets envoyés depuis la source.
4	Total Backward Packets	Numérique	Nombre total de paquets envoyés vers la source.
5	Total Length of Fwd Packets	Numérique	Somme des tailles des paquets envoyés en direction avant.
6	Total Length of Bwd Packets	Numérique	Somme des tailles des paquets envoyés en direction arrière.
7	Fwd Packet Length Mean	Numérique	Moyenne des tailles des paquets envoyés depuis la source.
8	Bwd Packet Length Mean	Numérique	Moyenne des tailles des paquets envoyés vers la source.
9	Flow Bytes/s	Numérique	Débit du flux en octets par seconde.
10	Flow Packets/s	Numérique	Nombre de paquets transmis par seconde.
11	PSH Flag Count	Binaire	Nombre de paquets avec le drapeau PSH activé.

12	ACK Flag Count	Binaire	Nombre de paquets avec le drapeau ACK activé.
13	URG Flag Count	Binaire	Nombre de paquets avec le drapeau URG activé.
14	Active Mean	Numérique	Temps moyen d'activité (entre deux transmissions de paquets).
15	Idle Mean	Numérique	Temps moyen d'inactivité entre deux paquets.
16	Init_Win_bytes_forward et backward	Numérique	Taille initiale de la fenêtre TCP dans le sens aller et retour.
17	Label	Nominal	Classe associée au flux (Benign, DoS, PortScan, etc.).

TABLE 3.3 – Détails des attributs sélectionnés du jeu de données CICIDS 2017

Chargement et fusionnement des données

Dans cette étape, nous avons récupéré les traces réseau issues du jeu de données **CIC-IDS 2017**, réparties sous forme de plusieurs fichiers CSV. ensuite ils ont été fusionné en une seule base exploitable.

Chaque fichier a été lu à l'aide de la bibliothèque pandas Afin de réduire la consommation mémoire, les colonnes numériques ont été converties au type float32.

Une colonne supplémentaire Source a été ajoutée pour conserver l'origine de chaque enregistrement.

Tous les fichiers ont ensuite été fusionnés en un seul DataFrame nommé df_network grâce à pd.concat().

Cette fusion permet de regrouper l'ensemble des données de trafic réseau dans une structure unifiée pour les étapes ultérieures de prétraitement et d'analyse.

En ce qui suit une figure qui illustre cette partie

```python
# Charger les fichiers CSV du trafic réseau depuis un dossier
dossier_reseau = r"CICIds data2017"
liste_fichiers = glob.glob(os.path.join(dossier_reseau, "*.csv"))
print("Fichiers CIC trouvés :", liste_fichiers)

# Lecture et fusion des fichiers CSV
liste_df = []
for fichier in liste_fichiers:
    df_temp = pd.read_csv(fichier)
    for col in df_temp.select_dtypes(include=['float64', 'int64']).columns:
        df_temp[col] = df_temp[col].astype('float32')  # Optimisation mémoire
    df_temp["Source"] = os.path.basename(fichier)  # Ajout d'une colonne indiquant le fichier source
    liste_df.append(df_temp)

df_network = pd.concat(liste_df, ignore_index=True) if liste_df else exit("Aucun fichier trouvé")
print("Aperçu du DataFrame CIC combiné:")
print(df_network.head())
```

FIGURE 3.11 – Chargement et fusionnement des Données CICIDS2017

Nettoyage des données

Une fois la fusion des fichiers terminée, nous avons procédé au nettoyage du DataFrame réseau `df_network` afin d'assurer la qualité des données avant leur traitement.

Les valeurs infinies ont été remplacées par des NaN, puis comblées en utilisant la méthode du forward fill, qui remplace chaque valeur manquante par la dernière valeur valide précédente. Des espaces indésirables présents dans certains noms de colonnes ont été supprimés via la fonction str.strip(), ce qui permet d'éviter les erreurs lors du traitement ou de l'accès aux colonnes.

```python
df_network.replace([np.inf, -np.inf], np.nan, inplace=True)
df_network = df_network.ffill()
df_network.columns = df_network.columns.str.strip()  # Suppression des espaces
```

FIGURE 3.12 – Nettoyage de données

Encodage de données

Les colonnes de type object, à l'exception de la colonne Label, ont été identifiées comme variables catégorielles. Ces colonnes ne peuvent pas être directement utilisées par les algorithmes d'apprentissage automatique, car ces derniers nécessitent des données numériques.

Pour résoudre ce problème, nous avons appliqué la méthode One-Hot Encoding,et avons activé l'option drop_first=True afin d'éviter la redondance des variables générées.

Cette étape permet ainsi de rendre l'ensemble des données numériques, facilitant leur traitement par le modèle de classification.

```
cat_cols_net = df_network.select_dtypes(include=['object']).columns.tolist()
cat_cols_net = [col for col in cat_cols_net if col not in ['Label']]
df_network_encoded = pd.get_dummies(df_network, columns=cat_cols_net, drop_first=True)
```

FIGURE 3.13 – Encodage de données

Sélection des caractéristiques

Dans cette étape, nous avons sélectionné trois caractéristiques quantitatives importantes issues du trafic réseau : Flow Duration (la durée d'un flux), Total Fwd Packets (le nombre total de paquets transmis dans le sens aller) et Total Backward Packets (le nombre total de paquets transmis dans le sens retour).

Ces dernières ont été extraites dans un tableau X_network.

Parallèlement, la variable cible y_network est construite à partir de la colonne Label et un encodage binaire a été appliqué : les étiquettes contenant le mot-clé `BENIGN` sont converties en `0` (trafic normal), tandis que toutes les autres sont considérées comme des attaques et codées en `1`.

```
features_network = ["Flow Duration", "Total Fwd Packets", "Total Backward Packets"]
X_network = df_network_encoded[features_network].values
y_network = df_network["Label"].apply(lambda x: 0 if x.strip().upper() == "BENIGN" else 1).values
```

FIGURE 3.14 – Séléction des caractéristiques et de label

Normalisation des données

Afin d'assurer une homogénéité des échelles entre les différentes caractéristiques, une normalisation standard a été appliquée via l'outil StandardScaler. dans le but de centrer et réduire les données, améliorant ainsi la stabilité numérique et l'efficacité de l'algorithme appliqué.

```
scaler_network = StandardScaler()
X_network_scaled = scaler_network.fit_transform(X_network)
```

FIGURE 3.15 – Normalisation

Construction et entraînement du détecteur

Après la normalisation des données réseau, l'ensemble a été divisé en deux sous-ensembles : un pour l'entraînement 70% et un autre pour les tests 30 %. Pour la détection d'anomalies dans le trafic réseau, l'algorithme AdaBoost a été utilisé, Ce dernier a été entraîné sur les données d'entraînement, et les scores de probabilité de détection ont été obtenus sur l'ensemble de test. Ces scores reflètent la probabilité pour chaque échantillon d'appartenir à la classe 1, représentant une attaque.

```
Xn_train, Xn_test, yn_train, yn_test = train_test_split(X_network_scaled, y_network, test_size=0.3, random_state=42)

# Entraînement du modèle AdaBoost
model_network = AdaBoostClassifier(n_estimators=50, algorithm="SAMME", random_state=42)
model_network.fit(Xn_train, yn_train)
network_scores = model_network.predict_proba(Xn_test)[:, 1]

# Afficher les scores de probabilité de détection pour le trafic réseau
network_scores = model_network.predict_proba(Xn_test)[:, 1]  # Probabilité d'être une attaque (classe 1)
print("Scores de probabilité (Trafic Réseau) :")
print(network_scores[:10])
```

FIGURE 3.16 – entrainement de AdaBoost

3.6.3 Fusion Dynamique via Régression Logistique

Après l'obtention des scores de détection à partir de détecteur A basé sur les journaux système et détecteur B basé sur le trafic réseau, une stratégie de fusion qui s'agit de la régression logistique a été mise en place afin de combiner les résultats et de produire un score final de détection pour améliorer ainsi la précision globale du modèle.

Le modèle de régression logistique a ensuite été entraîné à partir de vecteur de fusion obtenu, et des étiquettes réelles extraites de l'ensemble de test du détecteur réseau.

Un seuil de décision (0.246728) déterminé de manière empirique après plusieurs expérimentations a ensuite été appliqué sur les scores fusionnés afin de produire les prédictions finales du système global (attaque ou activité bénigne).

```python
taille_min = min(len(system_scores), len(network_scores))
X_fusion = np.column_stack([system_scores[:taille_min], network_scores[:taille_min]])
y_fusion = yn_test[:taille_min]

# Régression logistique pour la fusion
fusion_model = LogisticRegression(random_state=42)
fusion_model.fit(X_fusion, y_fusion)
final_scores = fusion_model.predict_proba(X_fusion)[:, 1]

# Affichage des scores finaux de fusion
print("Scores finaux de fusion :")
print(final_scores[:10])

# Application du seuil choisi (0.246728)
seuil_final = 0.246728
y_pred = (final_scores >= seuil_final).astype(int)

# Affichage des prédictions finales après fusion
print("Prédictions finales après fusion :")
print(y_pred[:10])
```

FIGURE 3.17 – Fusionnement via régression logistique

3.6.4 Evaluation du modèle

Pour évaluer les performances de notre classifieur , j'ai utilisé les métriques de performance suivantes :

Matrice de confusion

La matrice de confusion est un outil fondamental pour évaluer performances d'un modèle de classification sur un jeu de données test avec des valeurs réelles connues. Cette matrice est organisée sous forme d'un tableau, avec quatre cellules clés.

TP (True Positives) : Nombre d'attaques correctement détectées comme attaques.

TN (True Negatives) : Nombre de comportements normaux correctement identifiés comme non attaques.

FP (False Positives) : Nombre de faux positifs, c'est-à-dire des instances normales incorrectement classées comme attaques (fausses alertes).

FN (False Negatives) : Nombre de fausses négatives, c'est-à-dire des attaques non détectées par le système.

Dans le cadre de l'analyse, nous avons employé une matrice de confusion afin d'examiner de près les performances de modèle en ce qui concerne la détection des diverses attaques APT.

En effet, la figure ci-dessous illustre cette matrice de confusion générée par le modèle, ce qui m'a permis d'effectuer une analyse approfondie et de déduire les conclusions suivantes :

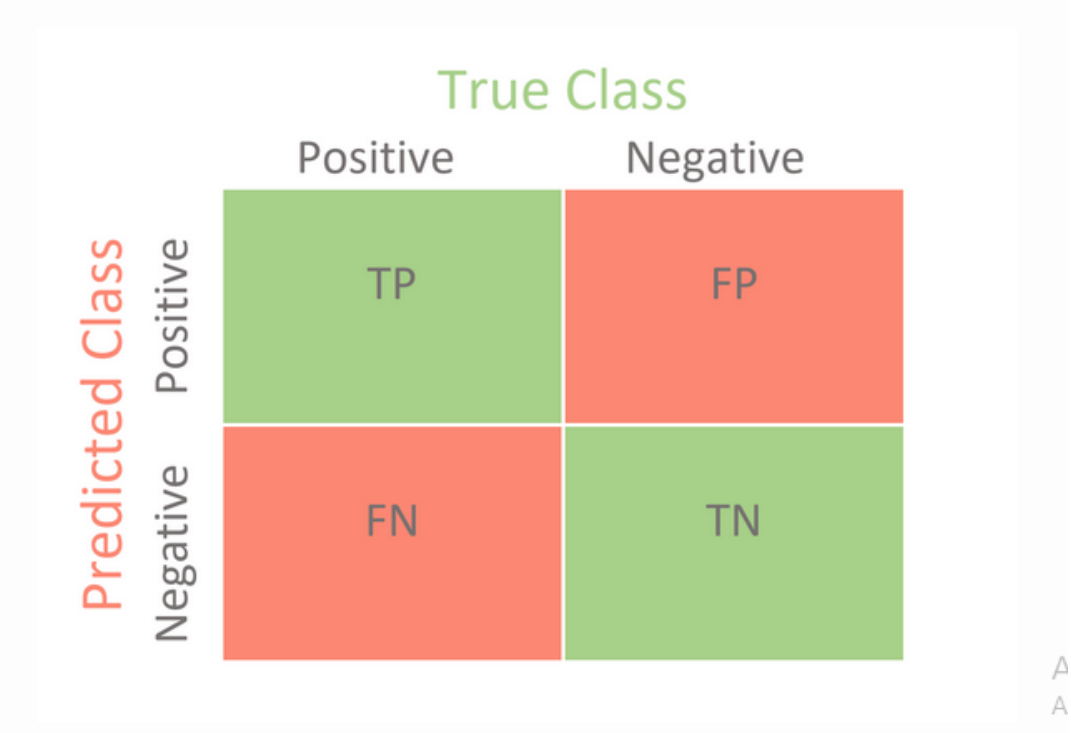

FIGURE 3.18 – Matrice de confusion

[41]

	Prédit Négatif	Prédit Positif
Réel Négatif	13 778 (TN)	3 307 (FP)
Réel Positif	700 (FN)	3 395 (TP)

TABLE 3.4 – Matrice de confusion du modèle

True Negative (TN) : 13 778 instances correctement identifiées comme non malveillantes.

False Positive (FP) : 3 307 instances non malveillantes incorrectement classées comme malveillantes.

False Negative (FN) : 700 attaques réelles non détectées par le modèle.

True Positive (TP) : 3 395 attaques correctement détectées.

Ces résultats indiquent une bonne capacité du modèle à détecter les attaques (TP) tout en limitant les fausses alertes (FP). Toutefois, la présence de 700 faux négatifs suggère que certaines attaques n'ont pas été détectées, Bien que ce nombre soit relativement bas par rapport au total,une marge d'amélioration doit être envisagée pour la détection des attaques APT .

Courbe Roc et AUC

La courbe ROC est un graphique qui montre la performance du modèle en affichant le Taux de Faux Positifs (FPR) en abscisse et le Taux de Vrais Positifs (TPR) en ordonnée, pour visualiser le compromis entre sensibilité (rappel) et spécificité du modèle, plus la courbe est proche du coin supérieur gauche, meilleur est le modèle.

L'AUC correspond à l'aire sous la courbe ROC, elle mesure globale la performance du modèle indépendamment d'un seuil particulier.

L'AUC varie entre 0 et 1 :

0.5 signifie un modèle sans pouvoir discriminant (aléatoire),

1 signifie un modèle parfait.

la figure ci-dessous illustre ces courbes générées par le modèle, ce qui m'a permis d'effectuer une analyse approfondie et de déduire les conclusions suivantes :

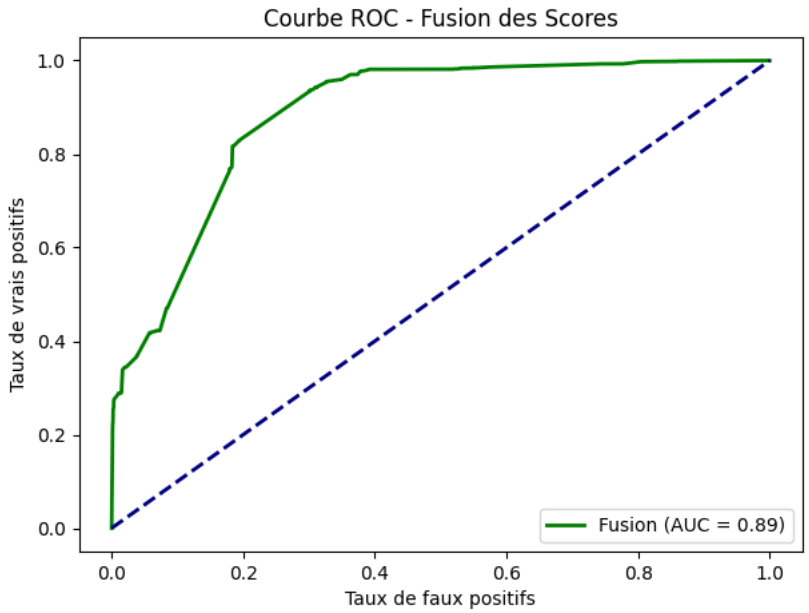

FIGURE 3.19 – Courbe ROC & AUC

la courbe ROC, avec une aire sous la courbe (AUC) significative, témoigne d'une différenciation efficace entre trafic normal et malveillant.

3.6.5 Étude Comparative

Calcul des métriques à partir de la matrice de confusion

Métrique	**Formule**	**Valeur obtenue**
Accuracy	$\frac{TP+TN}{TP+TN+FP+FN}$	$\frac{3395+13778}{21180} = 80.99\%$
Précision	$\frac{TP}{TP+FP}$	$\frac{3395}{3395+3307} = 50.66\%$
Recall (TPR)	$\frac{TP}{TP+FN}$	$\frac{3395}{3395+700} = 82.88\%$
FPR	$\frac{FP}{FP+TN}$	$\frac{3307}{3307+13778} = 19.35\%$
F1-score	$2 \cdot \frac{\text{Précision} \cdot \text{Recall}}{\text{Précision}+\text{Recall}}$	$\approx 62.74\%$

TABLE 3.5 – Métriques de performance obtenues à partir de la matrice de confusion

Le détecteur fusionné basé sur la régression logistique, appliqué à la fois aux logs système et au trafic réseau, atteint une accuracy de 80,79%, un rappel élevé de 82,88%, mais une précision plus modérée de 50,66%, avec un taux de faux positifs de 19,35%. Ce dernier indique que près d'un cinquième des alertes signalées sont des faux positifs.

En comparaison, les travaux de Du et al avec DeepLog basé sur LSTM sur des logs système rapportent une précision de 96,6% et un rappel de 92,6%, avec un F1-score de 94,6%. De même, Guo et al. (2021) exploitent le modèle BERT (basé sur les Transformers) sur les journaux HDFS et obtiennent des scores remarquables : précision de 98,9%, rappel de 98,3% et F1-score de 98,6%.

Plus récemment, Gunho No et al,proposent la méthode RAPID, rapportent une amélioration supérieure à +8% du taux de détection sur des journaux système réels provenant d'environnements de production et de test.

Comparativement à ces travaux, le modèle que nous proposons traite simultanément des données issues de sources différentes les journaux système et le trafic réseau afin de mieux refléter la réalité d'une attaque APT, qui exploite plusieurs vecteurs d'intrusion. Cette approche engendre naturellement une complexité accrue et explique un compromis entre un bon taux de détection (rappel élevé) et une augmentation du nombre de fausses alertes (précision plus basse et FPR plus élevé).

Ce compromis est acceptable dans le contexte des APT, où la priorité est de ne pas manquer de véritables menaces, même au prix d'une charge supplémentaire liée aux faux positifs.

3.6.6 Discussion des résultats

Les résultats obtenus démontrent l'efficacité de notre approche basée sur la fusion des détecteurs pour la détection des attaques APT.

En combinant les scores issus de l'analyse des journaux système avec l'algorithme RRCF et ceux issus de l'examen du trafic réseau avec AdaBoost, la méthode améliore la précision de la détection des menaces. La matrice de confusion met en évidence une bonne capacité à détecter les attaques, bien que le modèle génère encore un certain nombre de faux positifs, ce qui limite la précision globale. De plus, la courbe ROC, avec une aire sous la courbe (AUC) significative, témoigne d'une différenciation efficace entre trafic normal et malveillant. Un des avantages clés de cette approche est la complémentarité des détecteurs, permettant d'augmenter la robustesse face aux attaques sophistiquées. En effet, les logs système permettent d'identifier les anomalies locales sur une machine spécifique, tandis que l'analyse du trafic réseau détecte les comportements anormaux à plus grande échelle. Cette fusion des scores via une régression logistique optimise la prise de décision et renforce la capacité de détection des activités malveillantes.

Cependant, certaines limitations doivent être prises en compte. L'évaluation du modèle s'est faite sur des jeux de données simulés, ce qui peut différer des scénarios réels où les attaquants emploient des techniques plus avancées d'évasion. De plus, bien que notre approche détecte efficacement les premières phases des APT, elle ne couvre pas l'ensemble du cycle de vie des attaques. Plus précisément,notre modèle permet d' identifier :

La reconnaissance.

La compromission initiale.

Le commandement et contrôle (C2).

Toutefois, il ne prend pas en charge les phases ultérieures, telles que :

Le mouvement latéral.

L'exfiltration des données.

La persistance.

3.6.7 Conclusion

Ce chapitre a présenté la contribution proposée pour relever les défis de la détection des menaces persistantes avancées, à travers une fusion des analyses des journaux système

et du trafic réseau via la régression logistique.

Un modèle d'apprentissage automatique a été conçu et mis en œuvre, capable de détecter efficacement les anomalies caractéristiques des attaques APT. Les résultats expérimentaux obtenus se sont révélés très prometteurs et ouvrent la voie à des améliorations futures visant à renforcer la couverture des différentes phases du cycle de vie des attaques APT.

Conclusion générale

Dans le cadre de ce mémoire, j'ai étudié l'application de l'intelligence artificielle, et plus particulièrement des techniques d'apprentissage automatique, pour améliorer la détection des attaques persistantes avancées (APT). Après avoir présenté les concepts fondamentaux liés à la cybercriminalité,la sécurité informatique, aux menaces APT et aux méthodes de Machine Learning, j'ai mis l'accent sur son utilisation pour la détection des attaques APT.

L'état de l'art a été réalisé pour examiner les travaux connexes dans le domaine de l'application des méthodes de ML pour améliorer la détection des APT. J'ai identifié deux approches récentes existantes notamment : la méthode XFedHunter et la méthode RAPID. Force est de constater que cette dernière possède une limitation majeure qui réside dans l'adoption aux environnements distribués.

Face à ce constat, j'ai proposé une approche novatrice combinant deux détecteurs complémentaires :

Le Détecteur A,imitant la méthode RAPID, en se basant sur l'algorithme RRCF (Robust Random Cut Forest) et appliquant la détection d'anomalies aux logs système, afin d'identifier les comportements suspects au niveau des hôtes.

Le Détecteur B, quant à lui, utilise l'algorithme AdaBoost pour classer le trafic réseau à partir du jeu de données CIC-IDS 2017.

Les résultats issus des deux détecteurs sont ensuite fusionnés via une régression logistique, dans le but d'améliorer la prise de décision et d'augmenter la robustesse globale du système.

Les expérimentations menées ont permis d'évaluer la performance de chaque composant du système ainsi que du modèle fusionné. Les résultats obtenus montrent une bonne capacité de détection (Recall de 82,88%), malgré un taux de faux positifs relativement élevé (FPR de 19,35%), ce qui reste acceptable dans le cadre d'une détection proactive. L'approche adoptée démontre également une capacité à détecter des attaques dans les

premières phases du cycle APT (reconnaissance, compromission initiale, commande et contrôle).

Les perspectives futures de ce travail se déclinent en plusieurs axes. D'une part, il serait pertinent d'étendre la couverture de notre système à l'ensemble du cycle de vie des APT, notamment les phases de mouvement latéral, d'exfiltration de données et de persistance. D'autre part, l'intégration de techniques plus avancées comme les réseaux neuronaux profonds ou les modèles de fédération d'apprentissage pourrait renforcer encore davantage les performances, en particulier en matière de réduction des faux positifs. Enfin, pour évaluer la généralisation et la robustesse de notre approche, il sera nécessaire de la tester sur des jeux de données plus récents, tels que DAPT 2020 ou OpenAPT, représentant des scénarios plus proches des menaces réelles et actuelles.

Bibliographie

[1] L. Ablon, M. C. Libicki, and A. A. Golay, *Markets for Cybercrime Tools and Stolen Data : Hackers' Bazaar*. RAND Corporation, 2014.
https://www.rand.org/pubs/research_reports/RR610.html

[2] C. Tankard, "Advanced Persistent Threats and how to monitor and deter them," *Network Security*, vol. 2011, no. 8, pp. 16–19, 2011.
https://doi.org/10.1016/S1353-4858(11)70086-1

[3] A. L. Buczak and E. Guven, "A survey of data mining and machine learning methods for cyber security intrusion detection," *IEEE Communications Surveys & Tutorials*, vol. 18, no. 2, pp. 1153–1176, 2016.
https://doi.org/10.1109/COMST.2015.2494502

[4] Y. Zhu, M. Kumar, S. Rehman, R. A. Baig et W. Liu, *Modern Threats and Advanced Defense Strategies : A Cybersecurity Perspective*, Computer Science and IT Research Journal, vol. 12, no. 2, pp. 45–61, 2024. Disponible sur : https://www.fepbl.com/index.php/csitrj/article/view/758

[5] CHOUARFIA, A., *Introduction à la sécurité des réseaux et des systèmes d'information*, Support de cours, 2010. Disponible à : https://www.exoco-lmd.com/securite/introduction-a-la-securite-des-reseaux-et-des-systemes-dinformation/?action=dlattach;attach=7788.

[6] DJEBARI, N., *Sécurité des systèmes d'information*, Support de cours, 2023. Disponible à : https://elearning.univ-bejaia.dz/course/view.php?id=15997.

[7] Sharma, H., *The Evolution of Cybersecurity Challenges and Mitigation Strategies in Cloud Computing Systems*, International Journal of Computer Engineering and Technology, vol. 15, no. 4, pp. 118–127, Jul–Aug 2024.

https://iaeme.com/MasterAdmin/Journal_uploads/IJCET/VOLUME_15_ISSUE_4/IJCET_15_04_010.pdf

[8] Liu, F. T., Ting, K. M., & Zhou, Z. H. (2008). Isolation forest. *2008 Eighth IEEE International Conference on Data Mining*. https://doi.org/10.1109/ICDM.2008.17

[9] Cortes, C., & Vapnik, V. (1995). Support-vector networks. *Machine Learning*, 20(3), 273–297. https://doi.org/10.1007/BF00994018

[10] MacQueen, J. (1967). Some methods for classification and analysis of multivariate observations. *Proc. of the Fifth Berkeley Symposium on Mathematical Statistics and Probability*, Vol. 1, pp. 281–297.

[11] Chandola, V., Banerjee, A., & Kumar, V. (2009). Anomaly detection : A survey. *ACM Computing Surveys*, 41(3), 1–58. https://doi.org/10.1145/1541880.1541882

[12] Ahmed, M., Mahmood, A. N., & Hu, J. (2016). A survey of network anomaly detection techniques. *Journal of Network and Computer Applications*, 60, 19–31. https://doi.org/10.1016/j.jnca.2015.11.016

[13] Real-Time APT Detection Technologies : A Literature Review. [Consulté en 2025].

[14] Rapid APT Detection in Resource-Constrained IoT Devices Using Global Vision Federated Learning (GV-FL). [Consulté en 2025].

[15] Un cadre d'apprentissage fédéré explicable pour la détection des APT dans les réseaux définis par logiciel (SDN). [Consulté en 2025].

[16] Machine Learning for APT Detection : Performance Analysis and Proposed Model Evaluation. [Consulté en 2025].

[17] Pernet, C., *Sécurité et espionnage informatique : Connaissance de la menace APT et du cyberespionnage*, Éditions Eyrolles, Paris, 2020 (ou l'année indiquée dans ton exemplaire), ISBN : 978-2-212-??????

[18] Pierre Albert, *Le Machine Learning avec Python : De la Théorie à la Pratique*, Tech-Press, Paris, France, 2022.

[19] Huynh Thai Thi, Ngo Duc Hoang Son, Phan The Duy, Nghi Hoang Khoa, Khoa Ngo-Khanh, Van-Hau Pham.
XFedHunter : An Explainable Federated Learning Framework for Advanced Persistent

Threat Detection in SDN.
arXiv preprint arXiv :2309.05757, 2023.
Disponible sur : `https://arxiv.org/abs/2309.05757`

[20] Vincent Granet. *Machine Learning avec Scikit-Learn – Mise en œuvre et cas concrets (2e édition)*. Dunod / O'Reilly, 2022. Fichier disponible sur l'ordinateur de l'auteur.

[21] Aurélien Géron. *Machine Learning avec Scikit-Learn et TensorFlow*. Traduit de l'anglais par Anne Bohy. Dunod / O'Reilly Media, Paris, 2017. Fichier disponible sur l'ordinateur de l'auteur.

[22] Collectif d'auteurs. *L'intelligence artificielle expliquée*. Éditions ENI, 2021. Fichier disponible sur l'ordinateur de l'auteur.

[23] DataScientest.*Les autoencoders, modèles d'apprentissage non supervisé*.
Disponible sur :
`https:datascientest.com/lesautoencoders-modeles-dapprentissage-non-supervise`
Consulté en avril 2025.

[24] DataCamp. *Federated Learning*. Disponible sur : `https://www.datacamp.com/fr/blog/federated-learning` Consulté en juin 2025.

[25] Zhang, J., Li, W., & Chen, H. *Detecting Advanced Persistent Threats in Host Logs Using XGBoost*. IEEE Access, vol.9, pp.138562–138572,2021.
Disponible sur : `https://ieeexplore.ieee.org/document/9617137/`

[26] Du, M., Li, F., Zheng, G., & Srikumar, V. *DeepLog : Anomaly Detection and Diagnosis from System Logs through Deep Learning*. In Proceedings of the 2017 ACM SIGSAC Conference on Computer and Communications Security (CCS'17), 2017.
Disponible sur : `https://github.com/logpai/DeepLog`

[27] Guo, H., Li, Y., Zheng, R., Xu, D., & Jin, H. *LogBERT : Log Anomaly Detection via BERT*. In 2021 IEEE International Conference on Big Data (Big Data). Disponible sur : `https://arxiv.org/abs/2106.04441`

[28] author = Yonatan Amaru and Dima Shmilovici and Rami Puzis, title = RAPID : Robust APT Detection and Investigation using Context-Aware Deep Learning, year = 2024, howpublished = `https://arxiv.org/abs/2402.10700`, note = Disponible sur arXiv : 2402.10700,

[29] Huynh, T. T., Ngo, D. H. S., Phan, T. D., Nghi, H. K., Ngo-Khanh, K., & Pham, V.-H. *XFedHunter : An Explainable Federated Learning Framework for Advanced Persistent Threat Detection in SDN*. arXiv preprint arXiv :2309.08485, 2023. Disponible sur : `https://arxiv.org/abs/2309.08485`

[30] Ongun, M. N., Gündüz, M. Z., & Gül, F. *CELEST : Federated Learning-Based Intrusion Detection in SDNs*. arXiv preprint arXiv :2205.11459, 2022. Disponible sur : `https://arxiv.org/abs/2205.11459`

[31] Nazari, M., Alipour, A., & Rahmani, A. M. *P3GNN : Privacy-Preserving Graph Neural Network for APT Detection*. arXiv preprint arXiv :2406.12003, 2024. Disponible sur : `https://arxiv.org/abs/2406.12003`

[32] Rey, C., Raza, S., & Björkman, M. *Federated Learning for Malware Detection in IoT Systems*. arXiv preprint arXiv :2104.09994, 2021. Disponible sur : `https://arxiv.org/abs/2104.09994`

[33] Huynh, V. T., Nguyen, H. X., Nguyen, T. M., & Niyato, D. *XFedHunter : An Explainable Federated Learning Framework for APT Detection in SDN Environments*. IEEE Transactions on Information Forensics and Security, 2023.

[34] Sharafaldin, I., Lashkari, A. H., & Ghorbani, A. A. (2018). Toward Generating a New Intrusion Detection Dataset and Intrusion Traffic Characterization. In *Proceedings of the 4th International Conference on Information Systems Security and Privacy (ICISSP)* (pp. 108–116). DOI : `https://doi.org/10.5220/0006639801080116`

[35] Python Software Foundation. (2023). *Python Language Reference Manual*. Disponible sur : `https://www.python.org`

[36] Python Software Foundation. Disponible sur : `https://www.python.org/community/logos`

[37] "Visual Studio Code," *Wikipedia, The Free Encyclopedia*, mise à jour début juin 2025. Description : éditeur open-source multiplateforme ; coloration syntaxique, complétion automatique (IntelliSense), gestion de projets, débogage intégré, extensible via extensions et intégration fluide avec Python.
`https://en.wikipedia.org/wiki/Visual_Studio_Code`

[38] Microsoft. Disponible sur : `https://code.visualstudio.com/brand`

[39] Wikimedia Commons. Disponible sur : `https://commons.wikimedia.org`

[40] Sokolova, M., & Lapalme, G. (2009). A systematic analysis of performance measures for classification tasks. Information Processing & Management, 45(4), 427–437. DOI : https://doi.org/10.1016/j.ipm.2009.03.002

[41] Picsellia. Matrice de confusion – Computer Vision. Disponible sur : https://www.picsellia.fr/post/matrice-confusion-computer-vision

Résumé

Ce mémoire explore l'application des techniques de Machine Learning pour détecter les menaces persistantes avancées dans le domaine de la sécurité informatique. Les APT, en tant que cyberattaques sophistiquées, présentent des défis majeurs pour les méthodes de détection classiques.

L'objectif principal est de concevoir un système hybride combinant deux détecteurs intelligents pour repérer les comportements suspects à partir des journaux système et du trafic réseau. Le premier détecteur, basé sur l'algorithme RRCF, s'applique aux logs système, tandis que le second utilise AdaBoost pour analyser le trafic réseau. Une régression logistique fusionne les résultats de ces deux modules pour une décision finale robuste.

Après un prétraitement rigoureux des données), le système atteint un vrai négatif de 13778 , un faux positif de 3307,un faux négatif de 700 et vrai positif de 3395 ; et celà signifie une bonne performance du modèle.

Abstract

This thesis explores the application of Machine Learning techniques for detecting Advanced Persistent Threats (APTs) in cybersecurity. APTs are sophisticated attacks that traditional methods often fail to detect.

The main goal is to develop a hybrid system with two intelligent detectors that identify suspicious behaviors from system logs and network traffic. The first detector uses the Robust Random Cut Forest (RRCF) algorithm for system logs, while the second applies AdaBoost to network data. A logistic regression model merges both outputs for final decision-making.

After thorough data preprocessing , the system achieves a true negatif (TN) = 13778 , a false positif (FP) = 3307,a false negatif (FN) = 700 and true positif (TP) = 3395 and This means good performance of the model.