République Algérienne Démocratique et Populaire
Ministère de l'Enseignement Supérieur et de la Recherche Scientifique
Université Chadli Bendjedid
Faculté des Sciences et de la Technologie
Département d'Informatique

الجمهورية الجزائرية الديمقراطية الشعبية
وزارة التعليم العالي والبحث العلمي
جامعة الشاذلي بن جديد
كلية العلوم و التكنولوجيا
قسم الاعلام الالي

MEMOIRE

Présenté par

BOUAZZA AYMENE

Pour l'obtention de diplôme de

MASTER

Filière : Informatique

Spécialité : Systèmes Informatiques Intelligents

Thème

Un mécanisme de sécurité basé sur le Fog Computing contre les botnet attaques

Soutenu le : 25/06/2023 Devant le Jury composé de :

Qualité	Nom et Prénom	Grade	Université
Président	Mr. Bentrad Sassi	MCB	Chadli Bendjedid El-Tarf
Rapporteur	**Mr. Benmachich Abd El Madjid**	**MCA**	**Chadli Bendjedid El-Tarf**
	Mlle. Labiod Yasmine	**MCA**	**Chadli Bendjedid El-Tarf**
Examinateur	Mr. Boutouil Abd El Latif	MCA	Chadli Bendjedid El-Tarf

Année Universitaire : 2022/2023

Remerciements

Je tiens à exprimer mes sincères remerciements à Dieu pour m'avoir accordé la force et la persévérance nécessaires pour mener à bien ce travail de mémoire.

Je tiens à remercier mes parents ainsi que tous mes frères et mes sœurs pour leurs soutiens et encouragements d'accomplir mes études que ce soit par leurs encouragements, leurs conseils ou leur aide pratique

Je tiens également à remercier mon encadrant, **M. Benmachiche**, pour sa guidance, son expertise et ses conseils avisés tout au long de la réalisation de ce travail.

Un grand merci également à mon encadrante, **Mlle Labiod**, pour son aide précieuse, ses suggestions pertinentes et son soutien constant. Sa contribution a été d'une grande valeur et a enrichir mon travail.

Table des matières

Liste des figures

Liste des tableaux

Liste des acronymes

IoT	Internet of Things
IDS	Intrusion Detection System,
ML	Machine Learning
MLP	Multi Layer Perceptron
IPS	Intrusion Prevention System
NIDS	Network Intrusion Detection System
NIPS	Network Intrusion Prevention System
HIDS	Host-based Intrusion Detection System
M2M	Machine-to-Machine
DDos	Distributed denial-of-service
DoS	Denial of Service,
PCAP	Packet CAPture
RFID	Radio Frequency Identification
SaaS	Software as a Service
OWASP	Open Web Application Security Project
C&C	Command and Control
MIPS	Millions of Instructions Per Second
AIDS	Acquired Immune Deficiency Syndrome
CPU	Central Processing Unit
TCP	Transmission Control Protocol
IA	Intelligence artificiel
NB	Naive Bayes
DoS	Denial of Service,
PCAP	Packet CAPture

Introduction

L'Internet des objets (IoT) s'est considérablement développé ces dernières années, permettant à de nombreux objets du quotidien de se connecter à Internet. Cependant, cette connectivité généralisée crée également des problèmes de sécurité importants, notamment en ce qui concerne la détection des intrusions. Dans cette thèse, nous abordons le problème de la détection d'intrusion dans l'IoT et proposons une approche innovante pour résoudre ce problème.

Contexte du projet et problématique

Dans un monde de plus en plus connecté où les objets du quotidien sont de plus en plus connectés à internet, la sécurité IoT est devenue une préoccupation majeure. Un attaquant peut exploiter les vulnérabilités des objets de connexion pour accéder au réseau, voler des données sensibles ou interrompre le service. Les méthodes actuelles de détection d'intrusion IoT ne sont pas totalement satisfaisantes car elles n'offrent pas une protection adéquate contre les attaques émergentes et sophistiquées.

Motivations

La sécurité des systèmes IoT est essentielle pour garantir la sécurité, l'intégrité et la disponibilité des données et des services. Cependant, les méthodes traditionnelles de détection d'intrusion sont souvent insuffisantes pour répondre aux particularités de l'IoT. Nous avons été poussés par la nécessité de développer un système de détection d'intrusion spécifiquement conçu pour l'Internet des Objets, capable de détecter les attaques émergentes et d'assurer la sécurité d’objets connectés.

Objectifs

L'objectif principal de cette thèse est de proposer une architecture d’un système de détection d'intrusion dans l'Internet des Objets robuste et fiable capable de détecter les activités malveillantes dans le réseau des objets connectés. À cette fin, nous explorerons l'utilisation de techniques avancées telles que l'apprentissage automatique et l'analyse comportementale pour détecter de manière proactive et précise les intrusions.

Contenu du mémoire

Ce mémoire est structuré en plusieurs chapitres, abordant différents aspects de la détection d'intrusion dans l'Internet des objets.

- **Chapitre 1 :** Introduction à l'Internet des Objets et au Fog Computing

Ce chapitre représente le contexte général de l'Internet des Objets, et Fog Computing et les définitions et les architectures et les différents domaines d'application de l'IoT et Fog Computing et les défis

- **Chapitre 2 :** La sécurité de l'Internet des objets

Ce chapitre introduit les concepts de base de la sécurité de l'IoT, en mettant en évidence les défis et enjeux associés à la protection des objets connectés contre les attaques.

- **Chapitre 3 :** Apprentissage automatique pour la détection d'intrusion dans l'IoT

Ce chapitre examine l'utilisation de techniques d'apprentissage automatique, les algorithmes de classification, pour la détection d'intrusion dans l'IoT.

- **Chapitre 4 :** Approche proposée

Dans ce chapitre, nous présenterons notre approche novatrice pour la détection d'intrusion dans l'Internet des objets. Nous détaillerons les approches et les techniques que nous proposons d'utiliser.

1 : Introduction à l'internet des objets et au Fog Computing

1.1. Introduction

L'infrastructure de Fog Computing est un élément essentiel Professionnel de l'internet. Du fait de leur capacité de stockage et de calcul, et de leur emplacement Proches des utilisateurs finaux, ils permettent de traiter les données à proximité Ils sont générés. Dans ce chapitre, nous présenterons des définitions et des explications Quelques concepts utilisés dans cet article. Bien que dans le domaine de l'Internet des objets, ou l'Internet des objets (IoT), a été largement adopté et connu, nous allons commencer Ce chapitre commence par expliquer l'IoT suivi d'une brève discussion d'une application de cette technologie. Nous détaillerons ensuite bon nombre des modifications nécessaires Pour réaliser une infrastructure de fog computing, parce que nous Savoir aujourd'hui.

1.2. Internet des objets

L'Internet des Objets, ou l'Internet des Objets (IoT) en anglais, se définit comme un ensemble d'objets physiques et virtuels interconnectés qui communiquent en réseau, généralement sans fil, vers Internet. Ces objets permettent de capturer et de transmettre des données en utilisant une puissance limitée.

Ces capacités leur permettent d'analyser et de traiter des données, de générer des actions sans aucune intervention humaine autre que leur configuration, de surveiller et de connecter leurs réseaux. L'intelligence intégrée garantit que les objets peuvent effectuer des actions plus rapidement, de manière plus cohérente et plus fiable que les humains. Ils peuvent intervenir dans des situations où les opérations peuvent présenter un danger pour les humains. Cette technologie existe déjà dans des domaines tels que la santé, les transports, le secteur privé…

La technologie diffère de l'informatique traditionnelle. En effet, les données peuvent être petites la taille et les transferts sont plus fréquents. De plus, de plus en plus de modules ou des nœuds seront connectés au réseau, ce qui conduira à une gestion plus complexe La sécurité de ces infrastructures [1].

1.2.1. Définitions et architectures de l'Internet des Objets

L'Internet des Objets (IdO) est un réseau de dispositifs physiques interconnectés via Internet qui collectent et échangent des données. Les objets connectés peuvent être de toutes formes et tailles, allant de simples capteurs aux voitures intelligentes et aux maisons connectées.

Les architectures de l'IdO peuvent varier en fonction des besoins spécifiques de l'application. Voici quelques exemples :

Architecture en étoile :

Cette architecture est souvent utilisée dans les maisons intelligentes et les bâtiments connectés, où un hub central relie tous les appareils à un réseau unique.

Architecture en maillage :

Cette architecture est utilisée pour créer des réseaux de capteurs qui se connectent les uns aux autres pour collecter et transmettre des données. Les réseaux de capteurs sont souvent utilisés dans les applications de surveillance environnementale et de sécurité.

Architecture en arbre :

Cette architecture est utilisée dans les applications de surveillance à grande échelle, où des capteurs sont répartis sur une zone géographique étendue et connectés à un nœud central qui agrège et transmet les données.

Architecture de l'Ido basée sur le cloud :

Dans cette architecture, les données collectées par les dispositifs IdO sont stockées et traitées dans le cloud, permettant ainsi aux utilisateurs d'accéder facilement à ces données à partir de n'importe quel endroit.

Ces architectures peuvent également être combinées pour créer des réseaux plus complexes et adaptés à des applications spécifiques.

1.2.2. Domaines d'application de l'internet des objets

Les applications IoT sont déployées dans divers secteurs, tels que la santé, la gestion des stocks et de la production, les chaînes d'approvisionnements alimentaires, le transport, la sécurité et la surveillance [2, 3, 4, 5].

À titre d'exemple, les objets intelligents peuvent être utilisés par des patients qui ont besoin de collecter des données concernant leur état de santé, comme leurs battements de cœur, leur

pression artérielle et leur taux de glucose. Dans ce cas, il possible de suivre l'état de santé des patients grâce à ces objets intelligents. Par ailleurs, l'IoT est aussi employé pour améliorer les maisons intelligentes lorsque des capteurs détectent des variations de température, les systèmes de climatisation peuvent être contrôlés pour rétablir la température idéale. Les caméras de sécurité domestique sont capables de capturer n'importe quel intrus et de transmettre des alertes aux propriétaires de la maison via des applications mobiles. De plus, l'IoT peut surveiller les systèmes de transport pour créer des villes intelligentes. Une fois collectées et analysées, les données permettent de comprendre les mises à jour des réseaux de trafic et des systèmes de transport. Enfin, une chaîne logistique fonctionnant avec l'IoT donne la possibilité d'enregistrer et suivre en temps réel toutes les livraisons. Dès que les marchandises ont été expédiées vers leur destination, les enregistrements de livraison peuvent être mis à jour. Il est possible de cartographier ces cas d'utilisation vers un modèle générique, lequel doit permettre une intégration plus aisée des différents services, allant des capteurs à l'infrastructure de réseau, des interfaces de programmation d'application ou Application Programming Interface (API) au traitement de données volumineuses, et enfin de l'analyse à la modélisation prédicative

1.2.3. Défis de l'internet des objets

L'Internet des Objets (IoT) est un réseau de dispositifs physiques tels que des capteurs, des caméras, des véhicules, des appareils électroménagers, des montres connectées, etc., qui sont connectés à Internet et qui peuvent communiquer entre eux pour échanger des données et réaliser des tâches. Bien que l'IoT offre de nombreux avantages, il existe également plusieurs défis à relever, notamment :

Sécurité :

La sécurité des données est un aspect crucial de l'IoT, car les appareils connectés peuvent être vulnérables aux cyberattaques. Il est important de mettre en place des mesures de sécurité pour protéger les données sensibles et prévenir les cyberattaques.

Intégration :

L'IoT implique souvent de nombreuses technologies et plates-formes différentes, ce qui peut rendre l'intégration des appareils difficile. Les développeurs doivent travailler ensemble pour créer des normes et des protocoles pour faciliter l'intégration des appareils IoT.

Interopérabilité :

L'IoT implique souvent des appareils de différents fabricants, ce qui peut rendre difficile la communication entre les appareils. Les normes et les protocoles doivent être développés pour permettre l'interopérabilité des appareils IoT.

Évolutivité :

L'IoT génère une quantité massive de données qui doivent être gérées et traitées. Les plateformes IoT doivent être évolutives pour pouvoir gérer ces données.

Vie privée :

L'IoT collecte souvent des données sensibles sur les utilisateurs, ce qui peut soulever des préoccupations en matière de vie privée. Les organisations doivent respecter les règles de confidentialité et de protection des données pour protéger la vie privée des utilisateurs.

Consommation d'énergie :

Les appareils IoT fonctionnent souvent sur batterie, ce qui peut limiter leur durée de vie. Les fabricants doivent trouver des moyens d'améliorer l'efficacité énergétique des appareils IoT pour augmenter leur durée de vie.

Coût :

Les appareils IoT peuvent être coûteux, ce qui peut limiter leur adoption par les consommateurs. Les fabricants doivent trouver des moyens de réduire les coûts des appareils IoT pour les rendre plus accessibles aux consommateurs.

1.3. Fog Computing

Aujourd'hui, des milliards d'objets intelligents sont connectés à Internet. Cisco a estimé qu'en 2020, ils représenteraient plus de 50 milliards [6]. Bien que nous sachions aujourd'hui que cette estimation est surévaluée, le nombre d'objets connectés n'a pas cessé de croître. Ces objets ont tous des caractéristiques très diverses. Par exemple, les applications de contrôle de vols de drones et les applications de jeux et de réalité virtuelle peuvent nécessiter un traitement rapide de leurs données. Étant donné que presque toutes les données sensorielles sont générées par des dispositifs finaux, ces dernières doivent être transférées (et analysées) vers des appareils possédant d'avantages de ressources. Pour pouvoir répondre à une telle exigence, deux paradigmes d'infrastructure – le cloud computing et fog computing – ont été positionnés comme des catalyseurs clés des applications IoT.

1.3.1. Définition de Fog Computing

Introduit par Cisco en 2012 [7], le Fog computing a ensuite été développé et défini par Open Fog Consortium (OFC). Le Fog computing relève d'une définition plus large du Edge computing, qui

consiste à pousser des applications et des services, totalement ou partiellement, vers l'extrémité du réseau. En conséquence, certaines fonctions (traitement, stockage temporaire, agrégation de données) sont transférées à proximité des utilisateurs finaux et des objets connectés, ce qui améliore le temps de réponse des applications et des services [8]. Comme mentionné précédemment, bien que les paradigmes Fog computing et cloud computing détiennent un fondement commun, à savoir la virtualisation, leurs caractéristiques sont différentes. Récemment, des propositions similaires au Fog peuvent être trouvées sous différents noms, comme Edge computing [9], Extrême Edge computing [10] ou Mobile Edge computing [11]. Cependant, le terme Fog computing ne désigne pas une seule architecture, mais une approche plus générique consistant à rapprocher l'intelligence plus près des utilisateurs finaux. Les infrastructures Fog peuvent être hiérarchiques, et comporter une architecture de cloud en haut de cette hiérarchie [12]. L'idée est que plus l'emplacement des équipements du Fog se trouve haut dans l'architecture (plus près du cloud), plus les ressources que le Fog met à disposition des utilisateurs sont importantes. En contrepartie, la latence de propagation est plus élevée. Le cloud fournit une capacité de stockage et de calcul quasi illimitée, au prix d'une latence élevée. À l'inverse, le Fog, plus proche de l'utilisateur, fournit une très faible latence, une faible capacité de calcul et de stockage. De la même manière, Perrera et al. [13] ont essayé de clarifier et d'élargir le concept du Fog computing.

« Le Fog computing est un scénario dans lequel un nombre considérable de dispositifs omniprésents et décentralisés hétérogènes (sans fil et parfois autonomes) communiquent et coopèrent potentiellement entre eux et avec le réseau pour effectuer des tâches de stockage et de traitement sans l'intervention de tiers. Ces tâches peuvent être destinées à prendre en charge des fonctions réseau de base ou de nouveaux services et applications s'exécutant dans un environnement en bac à sable. Les utilisateurs qui louent une partie de leurs appareils pour héberger services sont incités à le faire. »

1.3.2. Architectures de Fog Computing :

La figure 1 illustre les trois couches d'une architecture de Fog computing de manière schématique.

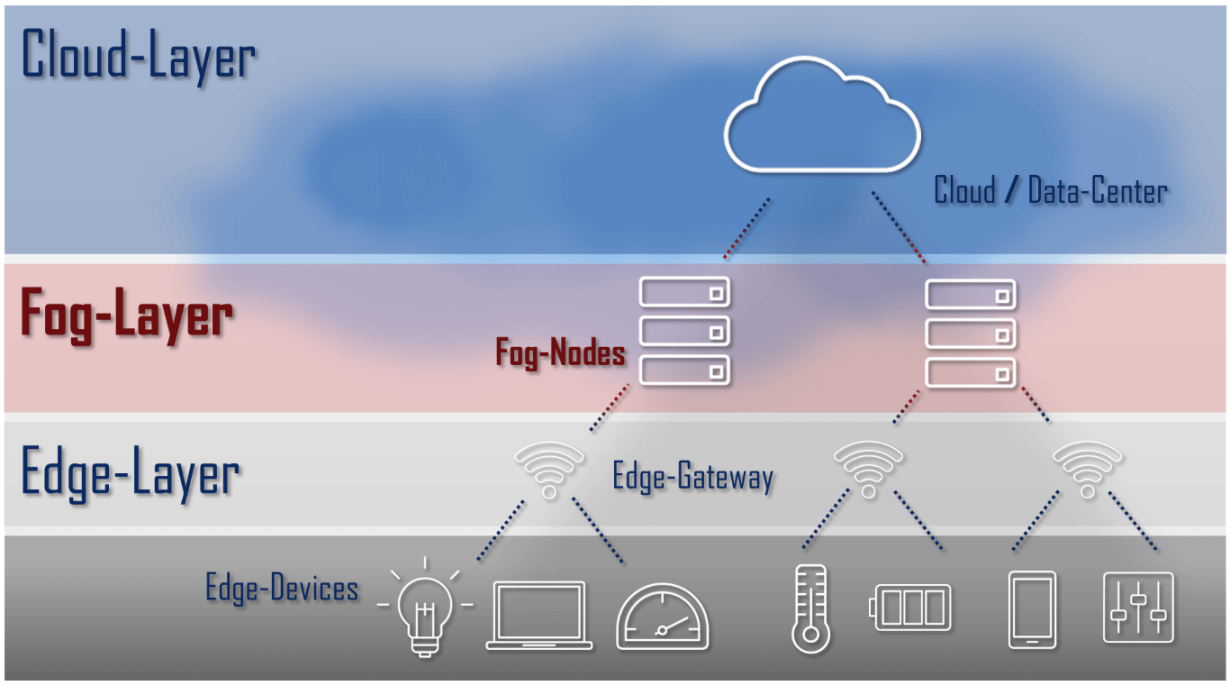

Figure 1 : Architecture du Fog Computing [50]

1/ Couche périphérique :

La couche périphérique englobe l'ensemble des dispositifs "intelligents" (dispositifs périphériques) d'une architecture IdO. Les données produites au sein de cette couche sont soit traitées localement sur le périphérique lui-même, soit transmises à un serveur (nœud fog) situé dans la couche Fog.

2/ Couche Fog :

La couche fog est constituée d'un ensemble de serveurs puissants chargés de recevoir les données provenant de la couche périphérique, de les prétraiter et de les transférer vers le cloud si nécessaire.

3/ Couche cloud :

La couche cloud représente le point central de traitement des données dans une architecture de cloud computing.

1.3.3. Paradigmes et technologies connexes

Plusieurs architectures sont proposées pour pallier le manque de ressources dans les installations connectées tout en permettant une mise à l'échelle et un traitement des données à faible latence. Ci-dessous nous définirons quelques paradigmes proches du Fog computing.

	Cloud	Fog
Taille	Des centres de données étendus avec un grand espace de stockage et des ressources de calculs.	Espace de stockage et des ressources de calculs limités. Un certain nombre de petits nœuds Fog peuvent être utilisés par un système Fog
Déploiement centralisé	Nécessite un déploiement soigneux et une maintenance maximale.	Déployé en centralisé ou distribué par des entreprises locales avec une maintenance minimale.
Emplacement	Informations globales recueillies dans le monde entier. Prend généralement en charge toutes les applications qui ne nécessitent pas un traitement en temps réel ou à faible latence (tolérantes au délai).	Informations liées à des emplacements de déploiement spécifiques. Peut prendre en charge des applications plus sensibles au délai.
Forte mobilité	Afin de prendre en charge la mobilité, le chemin d'un service doit changer à chaque fois, car le service est fourni à partir d'un cloud principal.	Un service peut être implémenté sur plusieurs nœuds Fog en suivant les dispositifs mobiles cibles.

Tableau 1 : Comparaison entre le Fog computing et le Cloud computing

Edge computing :

Comme le fog computing, c'est une des solutions aux problèmes ci-dessus. L'idée générale est d'effectuer des calculs sur des serveurs situés à la périphérie du réseau, plus près des utilisateurs et des objets connectés. Il existe également serveurs proches des utilisateurs à la périphérie du réseau [14, 15]. Dans le cas d'un réseau cellulaire, les serveurs sont situés à proximité de chaque station de base. Les utilisateurs utilisent donc ressources mises à disposition par les serveurs

situés dans la cellule à laquelle ils sont connectés. Ces serveurs effectuent des calculs à faible réponse et téléchargent les plus gourmands en ressources dans le cloud. Cette architecture est également utile pour le stockage des données : l'utilisateur envoie une requête à un serveur en périphérie du réseau. Si ce dernier ne dispose pas de la ressource demandée, la requête est envoyée à l'infrastructure cloud. Lorsque les données sont renvoyées depuis le cloud, elles sont mises en cache afin de pouvoir répondre directement aux demandes futures. Bien que les serveurs à la périphérie du réseau aient moins de ressources que les serveurs dans le cloud, ils disposent toujours de suffisamment de ressources pour effectuer des calculs à faible latence.

Extrême Edge computing :

Son objectif est le même que l'edge computing, qui est de traiter requêtes plus près des utilisateurs pour réduire le besoin d'une infrastructure cloud. Cependant, la mise en œuvre est différente : au lieu de nouveaux serveurs à proximité de utilisateurs, les utilisateurs se connectent aux utilisateurs et partagent leurs ressources utilisant des protocoles P2P ou ad hoc [16]. Le calcul et le stockage s'effectuent alors sur les appareils eux-mêmes (objets connectés). Les différences entre ces modèles d'infrastructure sont résumées dans [17].

Mobile Cloud computing :

Ces infrastructures permettent aux abonnés mobiles et aux personnes à faible revenu de tirer parti des ressources de l'infrastructure cloud. Deux architectures principales offrent cette possibilité [17]. La première consiste à déployer des serveurs (appelés cloudlets) à proximité des utilisateurs à la périphérie du réseau. Cette approche correspond au Edge computing ou mobile Edge computing [18]. Quant à la deuxième architecture, elle vise à combiner les ressources des différents périphériques qui se trouvent à proximité du afin de disposer d'une puissance de traitement suffisante pour l'application. Cette dernière approche est également connue sous le nom d'Extrême Edge Computing. Des modèles hybrides combinant ces deux approches ont également été proposés [15].

Mobile Edge computing :

Le cloud computing mobile (MCC) introduit une latence de traitement importante pendant le déchargement du calcul en transférant des données liées au calcul (c'est-à-dire du code vers le hub de cloud). Pour résoudre le problème de latence introduit par MCC, un autre paradigme a été proposé, Mobile Edge Computing (MEC). L'idée de MEC est de déplacer les ressources de calcul et de stockage à la périphérie du réseau cellulaire tout en maintenant une latence cohérente et cohérente avec de courtes distances de transmission de données. La clé de MEC est de rapprocher les ordinateurs avec plus de capacité de traitement et de stockage du mobile.

Utilisateurs à apporter [18]. Le contrôle du système d'exploitation vous permet de mettre en œuvre une gestion dynamique et flexible des petites cellules et des informations sur les utilisateurs mobiles (par exemple, l'emplacement), des informations sur le cloud périphérique (par exemple, les ressources de calcul et de stockage), des informations sur les utilisateurs et les petites cellules BS (par exemple, état du réseau et interférences de signal).

1.3.4. Exemples d'application du Fog Computing

De nombreux cas d'utilisation ont été proposés qui nécessitent l'utilisation d'une architecture de Fog computing.

Des feux de signalisation intelligents dans la ville intelligente :

Plusieurs travaux [15] ont considéré l'utilisation de l'infrastructure de Fog computing pour le contrôle du trafic. L'idée est que le feu de circulation relaie son statut aux voitures à proximité pour ralentir. L'état des feux et la position des véhicules sont régulièrement transmis à la plateforme de nébulisation qui établit une politique globale pour que les usagers attendent le moins possible et connaissent le temps de trajet le plus court. De manière générale, le Fog a été proposé comme aide à la décision pour le réseau de véhicules (VANET) [19]. L'utilisation du Fog a été présentée en lien avec les villes intelligentes [20]. L'idée générale est d'utiliser Nebula Nodes comme plate-forme de traitement des données collectées par les capteurs autour de la ville.

Traitement vidéo :

Pour Satya Narayanan et al. [21] Le Fog pourrait être utilisé pour archiver des séquences vidéo capturées par des caméras situées à la périphérie du réseau. Les flux vidéo sont en effet cohérents et le cloud ne serait pas en mesure de gérer un grand nombre de caméras. Cependant, votre suggestion ne se limite pas à cette idée ; Une hiérarchie nébuleuse permettrait le traitement des flux vidéo et en particulier la création d'une base de données cloud centralisés pour mettre en cache les vidéos stockées dans la nébuleuse. Un autre cas d'utilisation de Fog computing est la reconnaissance faciale [22]. Les passants sont filmés par une caméra et l'image est envoyée en temps réel au nœud de Fog pour une détection humaine.

1.3.5. Défis du Fog Computing

Le Fog Computing est une architecture informative présente de nombreux avantages, notamment une latence réduite, une bande passante plus faible, une meilleure sécurité et une meilleure efficacité énergétique. Cependant, elle comporte également plusieurs défis :

Gestion des données :

Le Fog Computing nécessite la collecte, le stockage et la gestion des données en temps réel, ce qui peut être difficile à mettre en œuvre et à maintenir.

Sécurité :

Avec de nombreux périphériques connectés au réseau, la sécurité est une préoccupation majeure pour les fournisseurs de services Fog Computing. Les données sensibles doivent être protégées contre les attaques malveillantes, les vols de données et les pertes de données.

Hétérogénéité des périphériques :

Les périphériques connectés au réseau dans une architecture Fog Computing peuvent être très différents les uns des autres en termes de matériel, de système d'exploitation, de protocoles de communication, etc. Il est donc difficile de garantir une compatibilité parfaite entre eux.

Gestion des ressources :

Le Fog Computing nécessite une gestion efficace des ressources, telles que la puissance de traitement, la mémoire et la bande passante. Il est important de s'assurer que chaque périphérique dispose des ressources nécessaires pour fournir des services de manière fiable et efficace.

Coût :

Le déploiement d'une infrastructure Fog Computing peut être coûteux, notamment en termes de matériel, de logiciels et de maintenance. Les coûts doivent être soigneusement pris en compte pour garantir la viabilité financière de la solution.

Complexité du réseau :

La mise en place d'un réseau de Fog Computing peut être complexe et coûteuse. Les appareils doivent être correctement configurés pour communiquer efficacement entre eux, et une infrastructure de réseau solide est nécessaire pour garantir que les données sont traitées de manière rapide et fiable.

Maintenance :

La maintenance des appareils de Fog Computing peut être difficile car ils sont souvent situés dans des environnements difficiles ou à distance. Des plans de maintenance réguliers doivent être mis en place pour garantir que les appareils fonctionnent correctement et sont mis à jour en permanence.

Intégration des systèmes existants :

L'intégration des systèmes existants dans un réseau de Fog Computing peut être difficile, car les appareils doivent être compatibles et interopérables. Des outils de développement efficaces et des normes de communication claires sont nécessaires pour faciliter l'intégration des systèmes existants.

1.4. Conclusion

Le Fog Computing a été introduit pour répondre au besoin de calcul à faible latence que les infrastructures de cloud computing ne peuvent pas fournir, mais aussi pour évoluer. En fait, le cloud n'a pas fait face à l'explosion du nombre d'utilisateurs. Il convient également de noter que les approches de Fog computing ou des bords sont très similaires, ne différant que de manière mineure. En d'autres termes, Fog examine les ressources organisées en permanence entre le cloud et les utilisateurs, tandis que l'Edge computing ne considère que le niveau intermédiaire. En pratique, la plupart des solutions logicielles implémentées pour l'Edge computing fonctionnent sur une infrastructure de Fog et vice versa. Des infrastructures de Fog sont actuellement introduites dans le réseau 5G pour fournir de la puissance de calcul aux téléphones mobiles [23, 24].

Dans le chapitre suivant, nous discuterons sur la sécurité de l'Internet des objets

2 : La sécurité de l'Internet des Objets basé sur le Fog Computing

2.1. Introduction

La sécurité est un aspect très souvent évoqué et discuté dans le domaine des objets connectés et plus généralement dans l'Internet des Objets. Le manque de sécurité de nombre de ces objets est relevé par plusieurs articles scientifiques et journaux. A partir des réflexions sur l'environnement connecté et des caractéristiques révélées précédemment, nous estimons que l'intégration d'objets dans un environnement peut poser certains problèmes liés à la sécurité. Ce chapitre présente la sécurité dans un environnement IoT. Nous présentons les éléments de définition nécessaires pour comprendre ce qu'est la sécurité informatique De plus, nous étudions les principaux concepts de base concernant la sécurité dans l'IoT et les mécanismes de sécurité pour protéger les environnements IoT contre les différents types d'attaques.

Ce chapitre introduit les concepts de base de la sécurité de l'IoT, en mettant en évidence les défis et enjeux associés à la protection des objets connectés contre les attaques.

2.2. La sécurité de l'internet des objets

La sécurité dans l'Internet des objets est un défi majeur pour les objets en réseau. Les architectures IoT devraient gérer des milliards de ces appareils interagissant entre eux et avec d'autres entités comme les humains et différentes plates-formes [25, 26]. Ces interactions doivent être sécurisées pour assurer un échange d'informations fluide. [27].

Les IoT imposent des exigences strictes en matière de débit de données, de bande passante disponible, de puissance de calcul, de stockage, de hiérarchisation de la qualité de service et de gestion des flux de données [28].

De plus, l'hétérogénéité de ces appareils et leur nature imprévisible de manipulation de mouvement compliquent les choses. Les solutions Internet sécurisées existantes doivent être évaluées et adaptées à l'environnement IoT. En conséquence, leur adaptation dans l'environnement IoT peut conduire à des résultats indésirables [29].

Chaque protocole Internet a sa propre portée et ses propres spécifications. La modification des propriétés du protocole peut différer de son utilisation d'origine, car de nombreux protocoles Internet ont été conçus pour les plates-formes informatiques traditionnelles. Les appareils IoT

n'ont pas été pris en compte lors du développement de ces fonctionnalités. Même les solutions IoT sécurisées existantes nécessitent des modifications importantes de leurs applications

2.3. Principaux comptes de bases en sécurité dans l'internet des objets

La sécurité de l'Internet des objets (IoT) est une préoccupation majeure pour de nombreuses personnes et organisations. Voici les principaux comptes de base en sécurité dans l'IoT [30, 31,32, 33, 34, 35, 36] :

Confidentialité des données :

L'IoT implique souvent la collecte, le stockage et le traitement de grandes quantités de données, y compris des données personnelles. Il est donc important de garantir la confidentialité de ces données.

Intégrité des données :

Les données collectées par l'IoT doivent être exactes et fiables. Il est important de garantir l'intégrité des données afin d'éviter toute modification ou altération malveillante.

Disponibilité des données :

Les données de l'IoT doivent être disponibles lorsque cela est nécessaire. Cela peut être un défi en cas de pannes ou d'attaques par déni de service (DDoS).

Sécurité des dispositifs :

Les dispositifs IoT doivent être sécurisés contre les accès non autorisés et les attaques malveillantes. Cela peut inclure des mécanismes de protection tels que des mots de passe forts et des certificats de sécurité.

Sécurité des communications :

Les communications entre les dispositifs IoT doivent être sécurisées pour empêcher les interceptions et les attaques de type "man-in-the-middle". Cela peut inclure l'utilisation de protocoles de communication sécurisés tels que TLS (Transport Layer Security) et SSH (Secure Shell).

Gestion des identités :

Les dispositifs IoT doivent être capables d'authentifier les utilisateurs et les autres dispositifs avec lesquels ils interagissent. Cela peut inclure l'utilisation de mécanismes d'authentification tels que des certificats numériques et des jetons d'authentification.

Gestion des mises à jour :

Les dispositifs IoT doivent être mis à jour régulièrement pour corriger les vulnérabilités de sécurité. Il est donc important de disposer de mécanismes efficaces de gestion des mises à jour logicielles et matérielles.

Protection de la vie privée :

L'IoT peut potentiellement collecter de grandes quantités de données personnelles, il est donc important de protéger la vie privée des utilisateurs. Cela peut inclure la mise en œuvre de mesures telles que la minimisation des données collectées, la limitation de la durée de conservation des données et la prise en compte des exigences en matière de protection de la vie privée lors de la conception des dispositifs IoT.

2.4. Principales attaques dans l'internet des objets

Les attaques les plus critiques sont décrites ci-dessous. Ils interviennent lors de la transmission des données, plus spécifiquement au niveau de la couche réseau [37,38] :

Déni de service (DoS) : cette menace est essentiellement une forme d'attaque par déni de service (DoS). Au lieu d'écouter les communications, l'intrus peut essayer de les interrompre en envoyant des données importantes ou en essayant de bloquer le canal de transmission.

Hello Flood : Congestion du canal de transmission avec un nombre infini de messages inutiles, entraînant un trafic important. Il s'agit d'une attaque par déni de service.

Spoofing (usurpation d'identité ou d'adresse IP) : mauvaise transmission ou modification du trafic pour voler des données, distribuer des logiciels malveillants ou contourner les contrôles d'accès. Un intrus peut compromettre une partie du signal sans fil avec un module externe exécutant un code malveillant.

Selective forwarding : Le nœud infecté envoie les données à plusieurs autres nœuds du réseau (choisis par l'attaquant) au lieu de les envoyer à tout le monde comme prévu. Par conséquent, certains modules ne transmettent pas les paquets de données ou le font de manière incorrecte.

Sybil : L'intrus réplique un nœud de réseau unique et le configure avec plusieurs identités pour d'autres nœuds de réseau. Cela se fait en introduisant un autre nœud ou fragment de programme qui est transmis sur le réseau.

Wormhole (trou noir) : Cette attaque amène le réseau à changer de position dans les paquets de données, provoquant un retard de transmission.

Acknowledgement flooding : Des accusés de réception synchronisés sont requis dans les réseaux de capteurs lors de l'utilisation d'algorithmes de routage. Dans ce type d'attaque, un nœud malveillant envoie par erreur de faux acquittements et informations à des nœuds voisins ou définis comme destinataires.

Man-in-the-middle (homme du milieu) : c'est une forme d'interception et de corruption de données. Un intrus opère entre deux modules réseau, interceptant les communications pour récupérer et modifier les informations transmises. La troisième partie sert de relais de transmission. Bien entendu, les deux sujets autorisés et intéressés ne sont pas informés de la présence de l'intercepteur. Pour éviter cette menace, les transferts actif-passif sont peer-to-peer. Cela signifie qu'en mode continu, un module d'exploitation doit être celui qui envoie des informations et l'autre qui les reçoit.

Attaque par relais : Dans ce cas, l'attaquant trompe le dispositif de réception local et redirige la communication vers un dispositif distant sans contact. En termes simples, il s'agit d'une attaque de l'homme du milieu sur les appareils NFC. Avec l'avènement des cartes à puce sans contact, ce type d'attaque devient un problème de plus en plus sérieux.

Protocol Stack Fuzzing (collecte dans la pile de protocoles) : Une autre catégorie d'attaques liées à NFC est basée sur des techniques de pêche à la pile de protocoles ou de fuzzing. Un intrus peut intercepter et analyser le logiciel de transfert pendant le transfert. Il peut alors forcer l'appareil mobile à collecter et analyser des images, des contacts, des documents ou d'autres contenus à l'insu de l'utilisateur. Utilise la technologie de récupération d'information ou de collecte de données (antenne, matériel). Par conséquent, il peut utiliser les données lors d'activités habituelles (par exemple, payer un ticket de transport en commun) et plus tard, par ex. B. lors d'appels ou d'envoi de messages, et même lors d'opérations illégales.

Rogue Access Points (points d'accès illicites) : Ce sont des périphériques réseau qui sont illégalement connectés au réseau. Ils peuvent être utilisés comme relais pour la transmission de données.

2.5. Les techniques de sécurité pour l'internet des objets basé sur le Fog Computing

La sécurité est un enjeu majeur du monde informatique et vise à maintenir la confiance des utilisateurs et la cohérence de l'ensemble du système d'information. De nombreux chercheurs internationaux et européens, ainsi que des industriels dans le domaine des technologies de l'information, ont proposé de nouvelles solutions et mécanismes pour résoudre les problèmes de sécurité dans l'environnement IoT.

Le Software-Defined Networking (SDN):

SDN, une nouvelle façon de construire, concevoir, sécuriser et exploiter des réseaux, est un concept émergent pour les gérer. Il fonctionne en séparant le plan de contrôle, ne gérant que le routage des paquets rendant le réseau programmable et flexible, et se charge d'associer la décision de routage aux paquets du plan de données qui le représentent dans l'infrastructure virtuelle ou physique et ce, est basé sur une centralisation gestion des flux réseaux. Avec le SDN, un dispositif externe appelé "contrôleur" gère les fonctions du plan de contrôle, la gestion et l'externalisation de l'intelligence du réseau. Le contrôleur, cette entité intelligente, peut résider sur une ou plusieurs machines physiques ou virtuelles distribuées. Configurer le réseau signifie programmer le contrôleur, via une interface de programmation d'application (API) ouverte, connue sous le nom d'API Northbound.La communication entre le périphérique réseau et le contrôleur se fait via un canal sécurisé via l'API Southsound (par exemple OpenFlow). Les grandes perspectives données au concept SDN par l'engouement des acteurs du numérique, comme Microsoft et Google, en mettant en place le SDN commencent à se concrétiser au niveau de leur data center. En d'autres termes, l'objectif du SDN est de simplifier la gestion des réseaux en leur permettant d'être flexibles, agiles et programmables. Ce concept SDN offre de nombreux avantages et est mondialement reconnu comme une architecture qui permet une mise en réseau ouverte pour les applications [39].

La Blockchain :

La blockchain, qui permet de se passer d'un tiers de confiance pour interagir en toute sécurité sur un réseau non sécurisé, est un nouveau paradigme que la communauté scientifique propose pour assurer le contrôle d'accès aux données. Nakamoto [40], qui a inventé la crypto-monnaie Bitcoin, a écrit un article dans lequel on retrouve la première description de la blockchain. Disponible sur un réseau peer-to-peer, où chaque nœud effectuant une tâche spécifique pour laquelle il est récompensé peut agir comme une sorte de garant des informations qui y sont stockées, la blockchain est un enregistrement indestructible et infalsifiable. Ces nœuds sont appelés mineurs.

Chaque compte est identifié sur la blockchain Bitcoin par une paire de clés publique/privée et est identifié par un surnom dérivé de sa clé publique. Prenons un exemple pour comprendre comment fonctionne la blockchain : Alice veut transférer x bitcoins d'un de ses comptes vers le compte de Bob. Alice génère une transaction, qu'elle signe avec la clé privée de son compte, contenant comme informations : le montant du virement, les adresses des comptes source et destination, et une référence à une ou plusieurs transactions antérieures ayant servi à créditer le compte (sans droit de découvert). Cette transaction, partagée dans un réseau peer-to-peer, y est téléchargée par des nœuds qui jouent le rôle de mineurs. Plusieurs transactions sont connectées par des nœuds pour créer un bloc qui est vérifié mathématiquement puis enchaîné à la blockchain via un processus de consensus spécifique entre les mineurs. Dans le cas de Bitcoin, ce processus est appelé preuve de travail. Une fois qu'une transaction est enregistrée sur la blockchain, elle devient irrévocable, publique, indestructible et infalsifiable. Ceci n'est pas garanti par une autorité centrale digne de confiance, mais par la communauté en ligne. Christidis et Devetsikiotis fournissent une définition plus approfondie des principes de la blockchain [41].

Système de Détection/Prévention d'Intrusion (IDS/IPS) :

Les systèmes de détection ou de prévention des intrusions, ou IDS/IPS en anglais, sont les composants les plus récents utilisés dans le paysage de la cybersécurité. Leur but est d'analyser les échanges au sein ou entre les systèmes pour prévenir ou détecter les intrusions. Pour cela, ces systèmes doivent pouvoir capter ces échanges, identifier une attaque, stopper la tentative d'intrusion et/ou émettre une alerte. Deux principaux types de IDS/IPS sont identifiables :

IDS/IPS réseaux :

Appelés aussi NIDS (ou NIPS), ils fonctionnent en utilisant une sonde spécifique positionnée sur le réseau et surveillant les échanges qui y sont effectués.

IDS/IPS hôtes :

Également appelés HIDS (ou HIPS), ils fonctionnent avec une ou plusieurs sondes implantées dans le serveur et surveillent la sécurité au niveau du serveur.

Dans notre cas, nous nous intéressons au type d'IDS/IPS qui permettra de surveiller les communications entre objets dans un environnement, en l'occurrence NIDS. Pour vérifier que les actions effectuées dans le système n'affectent pas les propriétés de sécurité, HIDS est couramment utilisé dans ledit système et ce, en vérifiant l'accès à certaines ressources suite à des paramètres système, des appels système exécutés, ou encore des fichiers journaux, par exemple. Dès lors, un composant spécifique similaire au NIDS serait une solution intéressante aux problèmes de l'IoT, du moment qu'il soit suffisamment générique pour pouvoir surveiller tous les

objets et limiter le besoin de configuration. Deux stratégies différentes sont utilisées pour identifier les attaques :

IDS/IPS à signatures :

si les éléments de capture (métadonnées, contenu des paquets) correspondent avec la signature d'une attaque connue ou connue, une attaque est identifiée comme telle. Ce mode de fonctionnement est équivalent au mode de fonctionnement d'un logiciel anti-virus, pour pouvoir identifier les menaces potentielles, à l'aide d'une base de données de signatures d'attaques.

IDS/IPS comportemental :

Si les éléments de capture correspondent à un comportement inhabituel, alors une attaque est définie comme telle. Un comportement normal, dans le cas de ces IDS/IPS, est préétabli et correspond à toutes les communications considérées comme légitimes, c'est-à-dire non malveillantes. Ainsi, la capture d'une communication est comparée à ce modèle, et s'il diffère significativement du modèle légitime, la communication est considérée comme illégale [42, 43].

2.6. Les systèmes de détection d'intrusions dans les objets connectés basé sur le Fog Computing

Le Fog Computing, également connu sous le nom de calcul en périphérie, est un modèle informatique qui vise à rapprocher le traitement des données du bord du réseau, plutôt que de les envoyer vers le cloud distant. L'utilisation du Fog Computing dans les systèmes de détection d'intrusions permet de traiter les données localement, réduisant ainsi la latence et améliorant la réactivité du système de détection.

Quelques exemples de systèmes de détection d'intrusions basés sur le Fog Computing :

Fog-based Intrusion Detection System (IDS):

Ce système de détection d'intrusions utilise des capteurs placés au niveau des périphéries du réseau pour collecter les données de trafic. Les données sont ensuite analysées et traitées localement dans le Fog Computing, où les règles de détection d'intrusions sont appliquées pour identifier les comportements malveillants.

Distributed Fog-based IDS :

Ce système utilise un réseau distribué de périphéries de Fog Computing pour la détection d'intrusions. Les capteurs de trafic sont répartis sur plusieurs périphéries, et les données sont analysées collectivement pour détecter les intrusions. Cette approche distribuée permet une détection plus rapide et une meilleure résilience face aux pannes d'un nœud spécifique.

Edge-based IDS :

Dans ce type de système, les périphéries du réseau (les appareils intelligents, les routeurs, les commutateurs, etc.) sont utilisées pour détecter les intrusions au niveau le plus proche des sources de données. Les données de trafic sont analysées et les anomalies sont détectées localement, réduisant ainsi la latence associée à l'envoi des données vers le cloud.

Hybrid Fog-Cloud IDS :

Ce système combine les avantages du Fog Computing et du Cloud Computing pour la détection d'intrusions. Les capteurs de trafic collectent les données au niveau des périphéries et les transmettent au Fog Computing pour une analyse initiale. Ensuite, les résultats sont transmis au cloud pour une analyse plus approfondie et une corrélation des événements sur une plus grande échelle.

2.7. Analyse comparative des systèmes de détection d'intrusions basées sur le Fog et les IDS existants pour l'IoT

Les systèmes de détection d'intrusions (IDS) jouent un rôle essentiel dans la sécurité des systèmes IoT (Internet des objets), car ils permettent de détecter et de prévenir les attaques potentielles. L'évolution récente des technologies a introduit de nouvelles approches, notamment les systèmes de détection d'intrusions basés sur le Fog, qui offrent des fonctionnalités spécifiques à l'environnement IoT. Dans cette analyse comparative, nous examinerons les caractéristiques et les avantages des systèmes de détection d'intrusions basées sur le Fog par rapport aux systèmes de détection d'intrusions existants pour l'IoT.

	IDS basés sur le Fog	**IDS existants pour l'IoT**
Architecture et déploiement	Ces systèmes sont conçus pour être déployés au niveau de l'infrastructure Fog, qui se situe entre les dispositifs IoT et le Cloud. Ils utilisent les ressources informatiques du Fog pour effectuer des analyses de données en temps réel et détecter les intrusions.	Ces systèmes sont généralement déployés soit directement sur les dispositifs IoT, soit dans le Cloud. Ils peuvent analyser les flux de données provenant des dispositifs IoT, mais peuvent rencontrer des limitations en termes de latence et de bande passante.
Latence et temps de réponse	En étant plus proches des dispositifs IoT, ces systèmes réduisent la latence en traitant les données localement, ce qui permet une détection d'intrusion plus rapide et une réponse plus immédiate.	Étant donné que ces systèmes doivent envoyer les données vers le Cloud pour les analyser, la latence est généralement plus élevée, ce qui peut entraîner des retards dans la détection et la réponse aux intrusions.
Bande passante	Ces systèmes sont capables de réduire la quantité de données transmises vers le Cloud en effectuant une analyse préliminaire localement. Cela permet d'économiser de la bande passante et d'alléger la charge sur les réseaux.	Les systèmes qui envoient toutes les données vers le Cloud peuvent entraîner une utilisation intensive de la bande passante, en particulier dans les déploiements IoT massifs avec un grand nombre de dispositifs.
Précision de détection	Grâce à la proximité des dispositifs IoT, ces systèmes peuvent bénéficier d'une visibilité plus approfondie sur le trafic réseau local, ce	Les systèmes de détection d'intrusions basés sur le Cloud peuvent être limités par les données disponibles à partir du flux réseau externe,

	qui peut conduire à une détection plus précise des intrusions et des anomalies.	ce qui peut affecter la précision de détection.
Évolutivité et extensibilité	En étant déployés au niveau du Fog, ces systèmes peuvent s'adapter facilement aux besoins évolutifs des déploiements IoT et offrir une extensibilité horizontale en ajoutant des ressources Fog supplémentaires.	Les systèmes basés sur le Cloud peuvent nécessiter des ajustements d'infrastructure pour faire face à la croissance des déploiements IoT, ce qui peut entraîner des coûts et des efforts supplémentaires.

Tableau 2 :Comparaison des systèmes de détection d'intrusions

En résumé, les systèmes de détection d'intrusions basés sur le Fog offrent des avantages significatifs en termes de latence réduite, de temps de réponse plus rapide, d'économie de bande passante et de précision de détection accrue par rapport aux systèmes de détection d'intrusions existants pour l'IoT. Cependant, le choix entre ces deux approches dépendra des exigences spécifiques du déploiement IoT et de la disponibilité des ressources Fog appropriées.

2.8. Conclusion

Le Fog Computing peut aider à améliorer la sécurité de l'IoT de plusieurs façons, notamment en réduisant la latence et la bande passante nécessaires pour transmettre des données entre les appareils IoT et le cloud, en offrant une capacité de traitement distribuée pour les tâches de sécurité et en permettant une isolation et une segmentation des réseaux pour empêcher les attaques de se propager.

Cependant, pour que le Fog Computing puisse contribuer efficacement à la sécurité de l'IoT, il est essentiel de prendre des mesures de sécurité appropriées, telles que la mise en place d'un accès sécurisé et la surveillance constante des appareils IoT et du réseau Fog. De plus, la collaboration entre les fabricants d'appareils IoT, les fournisseurs de services de Fog Computing et les autorités de réglementation est cruciale pour garantir la sécurité de l'IoT dans son ensemble.

En conclusion, le Fog Computing peut être un outil précieux pour renforcer la sécurité de l'IoT, mais il ne doit pas être considéré comme une solution à lui seul. Une approche globale et coordonnée de la sécurité de l'IoT est nécessaire pour protéger les appareils IoT et les données qu'ils collectent et transmettent.

Le chapitre suivant sera consacré à l’apprentissage automatique pour la détection d'intrusion dans l'IoT.

3 : Apprentissage automatique

3.1. Introduction

Il est nécessaire d'étudier les technologies récentes et les travaux existants dans la littérature, afin de pouvoir proposer une solution de détection adaptée aux problèmes de sécurité des objets et environnements connectés. Lors de la mise en place d'une politique de sécurité (règles de pare-feu, etc.) ou de la définition d'un modèle de sécurité.

Il existe deux approches pour établir le modèle de sécurité de type comportemental particulier requis pour l'IDS comportemental : la première s'appuie sur des experts ayant une connaissance adéquate des enjeux, des actes juridiques ou illégaux détectés et des interactions, et incluant ce modèle créé par eux ; et la deuxième repose sur l'apprentissage automatique. Dans le cas d'un domaine comme l'IoT, la définition d'un tel modèle présente de nombreuses limites. En effet, dans le cas de l'environnement connecté, les compétences spécialisées et la connaissance des différents facteurs évoqués ci-dessus sont difficiles à trouver par rapport à la jeunesse relative de l'IoT et plus chères en termes d'argent et de temps. De plus, faillibles, les gens ne rendent pas nécessairement leurs modèles robustes aux changements de l'environnement dans lequel ils se déploient, car ils ne sont pas toujours capables d'ajuster leur conscience à cet effet. Cette caractéristique est l'une des caractéristiques clés d'un environnement connecté. Dans cette situation, au contraire, une deuxième solution basée sur l'apprentissage automatique est souvent préférée. Elle implique l'utilisation d'algorithmes pour pouvoir paramétrer un modèle général à partir de données apprises, afin d'adapter le modèle à un problème donné. Cependant, cette solution a pour principal inconvénient de nécessiter de collecter des données représentatives du problème et suffisantes pour obtenir un modèle précis, donc généraliser correctement. De plus, le paramétrage automatique du modèle, donc l'apprentissage, basé sur des algorithmes est chronophage du fait de sa grande complexité. Cependant, contrairement à la première solution, la mise en place d'un apprentissage automatique pour identifier des modèles à partir de données, permet de s'améliorer grâce à de nouvelles données, plus adaptables, entièrement automatisées et moins coûteuses. Aujourd'hui, l'utilisation généralisée de l'apprentissage automatique dans les solutions de sécurité s'explique par ces raisons [44].

Dans ce présent chapitre nous explorons l'apprentissage automatique avec ses trois catégories supervisées, non supervisé et semi-supervisé. Nous présentons également la détection d'anomalie

Nous commençons par expliquer les principes fondamentaux de l'apprentissage automatique, en détaillant ses trois principales catégories : supervisée, non supervisée et semi-supervisée. Ensuite, nous abordons la détection d'anomalies, en décrivant les différents types d'anomalies ainsi que le concept essentiel de la détection. Enfin, nous présentons l'algorithmes d'apprentissage automatique qui a été mis en œuvre dans le cadre de notre recherche.

3.2. Définition de l'apprentissage automatique

Concepts de l'apprentissage automatique :

Plusieurs processus, difficiles à décrire avec précision, constituent l'apprentissage humain. Un avantage décisif pour le développement humain lui a été donné grâce à sa capacité d'apprentissage. Les « facultés d'apprendre » impliquent un ensemble de compétences telles que :

- Acquérir, par l'observation des autres, la capacité de parler.
- Acquérir la capacité d'écrire, de lire et d'effectuer des opérations arithmétiques et logiques avec l'aide d'un tuteur.
- Acquérir des habiletés athlétiques et motrices par l'exercice.

L'une des caractéristiques essentielles des formes de vie supérieures est la capacité de s'adapter et d'apprendre des expériences passées. Cette capacité est inhérente à la première étape de la vie d'une personne, afin qu'elle puisse apprendre des choses de base comme marcher, parler, comprendre ce qui se dit ainsi que reconnaître des visages et des voix familiers.

L'apprentissage automatique peut être défini comme une tentative de reproduire cette capacité d'apprentissage dans des systèmes artificiels, une fois qu'elle a été comprise. Schématiquement, le problème est de concevoir des algorithmes capables d'assimiler son essence à partir d'un grand nombre d'exemples (données correspondant à l'expérience passée), afin que ce qui peut être appliqué soit appris dans des cas futurs.

Définition de l'apprentissage automatique :

A partir d'une expérience E, comparée à un ensemble de tâches T et selon la mesure de performance P, un programme informatique peut savoir si sa performance sur l'exécution d'une tâche T mesurée par P s'est améliorée ou non selon l'expérience E. Un ensemble de tâches spécifiques T liées à l'apprentissage automatique. Une mesure de performance P est utilisée pour déterminer les performances de la machine. Peut-être que l'est la machine avec l'expérience E définie en premier, ou elle l'enrichit plus tard. Transformer les données en connaissances est l'objectif du machine Learning,

sous la forme d'un modèle. Le problème est donc de décrire un système à l'aide de concepts linguistiques et mathématiques. Différentes "familles" de modèles, appelées classes de modèles, peuvent être utilisées à cette fin. L'entrée de l'algorithme d'apprentissage automatique est constituée de données et d'une classe de modèle, et la sortie de cet algorithme est un modèle qui est l'instanciation de cette classe avec des paramètres, y compris ceux appris à partir des données. Les paramètres qui composent un modèle peuvent être divisés en trois catégories :

- Paramètres de classe : ils dépendent de la classe de modèle utilisée ;
- Paramètres fournis par le développeur : ils sont également appelés "hyperparamètres ».
- Les paramètres sont appris automatiquement pendant la phase d'apprentissage : ils sont appelés "paramètres".

De nombreuses classes de modèles, adaptées à différents problèmes, existent. Par exemple, les comportements probabilistes peuvent être modélisés par certaines classes, tandis que les évolutions temporelles peuvent être modélisées par d'autres classes. L'apprentissage automatique vise à apprendre un modèle capable de généraliser et donc de prédire de nouvelles données ou d'agir sur la base de données. Différentes familles d'apprentissage peuvent être distinguées en fonction du format des données et de la tâche apprise.

3.2.1. Apprentissage supervisé

L'apprentissage supervisé est une technique d'apprentissage automatique (Machine Learning) dans laquelle un modèle est entraîné à partir de données d'entrée et de sorties correspondantes. L'objectif est de faire apprendre au modèle une fonction qui peut prédire la sortie souhaitée pour de nouvelles données d'entrée.

Dans le cadre de l'apprentissage supervisé, les données utilisées pour l'entraînement sont appelées ensemble de données d'apprentissage. Chaque exemple dans cet ensemble est composé d'une entrée (également appelée vecteur de caractéristiques ou variables indépendantes) et d'une sortie attendue (également appelée variable dépendante ou étiquette).

L'algorithme d'apprentissage supervisé utilise cet ensemble de données d'apprentissage pour ajuster les paramètres internes du modèle, de manière à minimiser l'erreur entre les sorties prédites et les sorties réelles. Une fois que le modèle a été entraîné, il peut être utilisé pour prédire les sorties correspondantes pour de nouvelles données d'entrée qui n'ont pas été vues pendant l'entraînement.

Il existe plusieurs types d'algorithmes d'apprentissage supervisé, notamment :

1. **Les algorithmes de régression** :
   Ils sont utilisés lorsque la variable de sortie est continue. L'objectif est de prédire une valeur numérique, telle que le prix d'une maison en fonction de ses caractéristiques.
2. **Les algorithmes de classification** :
   Ils sont utilisés lorsque la variable de sortie est discrète ou catégorique. L'objectif est de prédire une classe ou une étiquette, telle que la détection de spam dans les e-mails.

Parmi les algorithmes d'apprentissage supervisé couramment utilisés, on trouve les arbres de décision, les réseaux de neurones artificiels, les machines à vecteurs de support (SVM) et les méthodes d'ensemble, telles que le Random Forest et le Gradient Boosting.

Il convient de noter que pour l'apprentissage supervisé, il est crucial d'avoir des données d'entraînement étiquetées, c'est-à-dire des exemples pour lesquels la sortie attendue est connue. Cela nécessite souvent un travail de préparation des données, y compris l'étiquetage manuel par des experts humains.

3.2.2. Apprentissage non supervisé

L'apprentissage non supervisé est une branche de l'apprentissage automatique (ou Machine Learning) dans laquelle un modèle est entraîné à trouver des motifs ou des structures dans les données sans avoir d'étiquettes ou de réponses prédéterminées.

Contrairement à l'apprentissage supervisé, où les données d'entraînement sont étiquetées avec des réponses connues, l'apprentissage non supervisé explore les données sans aucune connaissance préalable. Il s'agit donc d'une méthode d'apprentissage automatique non guidée.

L'objectif principal de l'apprentissage non supervisé est de découvrir des motifs cachés, des regroupements ou des associations dans les données, afin de fournir des informations et des connaissances utiles pour la prise de décision ou pour une meilleure compréhension des données.

Il existe plusieurs approches couramment utilisées en apprentissage non supervisé :

- **Clustering (regroupement)** :
  Cette approche consiste à regrouper les données en fonction de leurs similarités. Les algorithmes de clustering tels que K-Means, DBSCAN ou Hierarchical Clustering sont utilisés pour identifier les groupes naturels ou les structures dans les données.
- **Réduction de dimension** :
  L'objectif de cette approche est de réduire la dimensionnalité des données en conservant les informations essentielles. Les techniques de réduction de dimension comme l'analyse

en composantes principales (PCA) ou le t-SNE permettent de visualiser et de représenter les données dans un espace de dimension réduit.

- **Règles d'association** :

  Cette approche consiste à découvrir des relations fréquentes entre les différents éléments d'un ensemble de données. L'algorithme d'association le plus connu est l'apriori, qui est souvent utilisé dans les domaines du marketing et du panier d'achat

- **Détection d'anomalies** :

  Cette approche vise à détecter les observations anormales ou les comportements inattendus dans les données. Les techniques d'apprentissage non supervisé permettent de repérer les valeurs aberrantes ou les points de données qui ne correspondent pas aux schémas normaux.

L'apprentissage non supervisé est largement utilisé dans de nombreux domaines, tels que l'analyse des données, la vision par ordinateur, la bio-informatique, la recommandation de contenu, la détection de fraudes, etc. Il permet de découvrir des connaissances précieuses à partir de données brutes sans avoir besoin d'une supervision explicite.

3.2.3. Apprentissage semi-supervisé

L'apprentissage semi-supervisé est une approche de l'apprentissage automatique qui combine des données étiquetées et non étiquetées pour entraîner un modèle. Contrairement à l'apprentissage supervisé traditionnel qui nécessite des données étiquetées pour chaque exemple d'entraînement, l'apprentissage semi-supervisé permet d'exploiter des données non étiquetées supplémentaires pour améliorer les performances du modèle.

Dans un contexte d'apprentissage semi-supervisé, un sous-ensemble des données est étiqueté, c'est-à-dire qu'il contient des exemples associés à des étiquettes ou des catégories connues. Cependant, la majorité des données restent non étiquetées. L'idée est d'utiliser les exemples étiquetés pour apprendre les modèles et les relations entre les caractéristiques et les étiquettes, puis de généraliser ces connaissances aux données non étiquetées.

Il existe différentes approches pour l'apprentissage semi-supervisé. L'une des méthodes courantes est la propagation de l'étiquette, où les étiquettes sont propagées à partir des exemples étiquetés vers les exemples non étiquetés voisins dans l'espace des caractéristiques. Une autre méthode est l'apprentissage par Co-formation, où plusieurs modèles sont entraînés sur différentes parties des données et se corrigent mutuellement.

L'apprentissage semi-supervisé présente plusieurs avantages. Il peut réduire la dépendance aux données étiquetées coûteuses en permettant l'utilisation de grandes quantités de données non étiquetées plus facilement accessibles. Cela peut être particulièrement utile lorsque l'étiquetage

manuel des données est coûteux, long ou difficile. De plus, l'utilisation de données non étiquetées supplémentaires peut aider à améliorer la généralisation du modèle, en réduisant le risque de surajustement aux données étiquetées limitées.

Cependant, il est important de noter que l'apprentissage semi-supervisé présente également des défis. La qualité des données non étiquetées peut varier et peut introduire du bruit dans le processus d'apprentissage. De plus, l'utilisation de données non étiquetées peut rendre le processus d'apprentissage plus complexe et nécessiter des techniques spécifiques pour tirer pleinement parti de ces données.

En résumé, l'apprentissage semi-supervisé est une approche d'apprentissage automatique qui combine des données étiquetées et non étiquetées pour améliorer les performances du modèle. Cela permet d'utiliser plus efficacement les ressources de données et peut être particulièrement bénéfique dans des scénarios où l'étiquetage des données est coûteux ou difficile.

3.3. Détection d'anomalies

La détection d'anomalies, également connue sous le nom de détection d'exceptions ou de détection de valeurs aberrantes, est un processus d'identification de modèles ou de points de données qui diffèrent significativement du comportement attendu dans un ensemble de données donné. L'objectif est de repérer les observations qui se situent en dehors de la norme et qui peuvent indiquer des comportements ou des événements inhabituels, potentiellement intéressants ou problématiques.

La détection d'anomalies est utilisée dans de nombreux domaines, tels que la sécurité des réseaux, la détection de fraude, la surveillance des systèmes, la détection de défauts dans la fabrication, la surveillance de la santé, l'analyse financière, etc. Les méthodes de détection d'anomalies peuvent varier en fonction du domaine d'application et des caractéristiques des données, mais il existe quelques approches couramment utilisées :

- **Méthodes statistiques** :
  Ces méthodes se basent sur des modèles statistiques pour détecter les observations qui s'éloignent significativement des valeurs attendues. Elles peuvent utiliser des techniques telles que la déviation standard, les tests d'hypothèse, les méthodes de clustering ou les modèles de régression.
- **Méthodes basées sur l'apprentissage automatique** :
  Ces méthodes utilisent des algorithmes d'apprentissage automatique pour apprendre à partir de données historiques et détecter les anomalies. Les approches populaires incluent

les méthodes de clustering, les arbres de décision, les machines à vecteurs de support (SVM), les réseaux de neurones, et les méthodes d'apprentissage par renforcement

- **Méthodes basées sur les règles** :

  Ces méthodes utilisent des règles prédéfinies ou des seuils pour détecter les anomalies. Par exemple, si la température d'un système dépasse un certain seuil, cela peut être considéré comme une anomalie

- **Méthodes basées sur l'apprentissage non supervisé** :

  Ces méthodes sont utilisées lorsque les anomalies ne sont pas clairement étiquetées dans les données d'apprentissage. Les algorithmes d'apprentissage non supervisé tentent de trouver des schémas ou des structures inhabituelles dans les données

Il est important de noter que la détection d'anomalies peut être un problème complexe et dépendant du contexte. Il n'existe pas de solution universelle qui fonctionne dans tous les cas. Le choix de la méthode de détection d'anomalies dépendra des caractéristiques des données, du contexte d'application et des objectifs spécifiques de l'analyse.

3.3.1. Types d'anomalies

Les anomalies dans les systèmes IoT (Internet of Things) peuvent prendre différentes formes en fonction de la nature des appareils connectés et des infrastructures déployées. Voici quelques types courants d'anomalies dans les systèmes IoT :

- **Anomalies de connectivité** :

  Les appareils IoT peuvent parfois perdre leur connexion réseau, ce qui peut entraîner des interruptions de communication et des dysfonctionnements du système

- **Anomalies de trafic** :

  Les anomalies de trafic peuvent se produire lorsque le volume de données échangées entre les appareils IoT dépasse les seuils normaux, indiquant une activité anormale ou potentiellement malveillante

- **Anomalies de comportement** :

  Les anomalies de comportement se manifestent lorsque les appareils IoT fonctionnent de manière inattendue ou exécutent des actions qui ne correspondent pas à leur mode de fonctionnement normal

- **Anomalies de consommation d'énergie** :

  Les appareils IoT peuvent consommer anormalement plus d'énergie que prévu, ce qui peut indiquer des problèmes matériels ou logiciels

- **Anomalies de sécurité** :

  Les anomalies de sécurité dans les systèmes IoT peuvent inclure des tentatives d'intrusion, des attaques par déni de service, des comportements suspects ou des violations de la confidentialité des données

- **Anomalies environnementales** :

  Certains appareils IoT sont conçus pour surveiller et réagir à des conditions environnementales spécifiques. Les anomalies environnementales se produisent lorsque ces conditions dévient des plages normales, ce qui peut indiquer des problèmes ou des situations critiques

- **Anomalies de localisation** :

  Les appareils IoT dotés de capacités de localisation peuvent présenter des anomalies lorsqu'ils fournissent des informations de localisation inexactes ou incohérentes

Ces différents types d'anomalies nécessitent une surveillance continue et une analyse intelligente pour détecter les comportements anormaux et prendre des mesures appropriées pour résoudre les problèmes potentiels dans les systèmes IoT.

3.3.2. L'idée fondamentale de la détection d'anomalies :

L'approche utilisée par presque tous les algorithmes de détection d'anomalies consiste à créer un modèle qui modélise les données normales, puis à calculer un score pour chaque point de données mesurant l'écart par rapport à la normale, si le score est supérieur au seuil précédemment défini, classer le point comme une anomalie. Il est essentiel de concevoir un modèle et un score adaptés [45].

3.4. Algorithmes et plateformes d'apprentissage automatique (ML) utilisés

- **Le perceptron multicouche (MLP)**

Le perceptron multicouche (MLP) est l'une des méthodes de classification les plus puissantes et applicables dans l'IDS [46]. Le perceptron est une représentation mathématique d'un neurone, l'unité de calcul de base dans le cerveau. Dans le système nerveux humain, un neurone reçoit une

entrée de ses dendrites et envoie des signaux le long de son axone, qui se ramifiera et se connectera à ses dendrites d'autres neurones par le biais de synapses. ANN est un modèle de traitement de l'information basé sur cette structure biologique. Il modélise des relations complexes entre les entrées et les sorties. Comme le montre la figure 2, la structure du réseau MLP se compose de couches d'entrée, de sortie et cachées, ainsi que des poids et biais associés. Chaque couche cachée composée de plusieurs perceptrons est appelée couche cachée ou unité cachée [47].

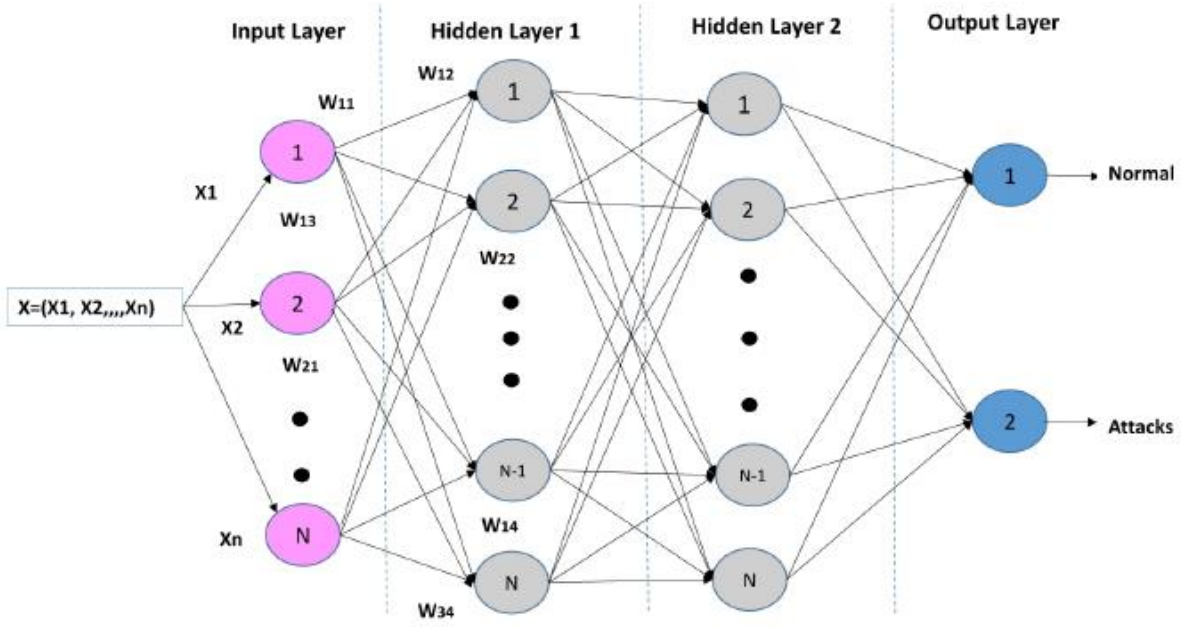

Figure 2 : Représentation de l'architecture MLP [51]

Les connexions entre les classes doivent être distinguées par des poids dans l'intervalle [-1;1]. Tous les neurones MLP remplissent deux fonctions : résumé et activation. Les résultats des entrées, des poids et des écarts sont additionnés à l'aide de la fonction (Équation 2)

$$Si = \sum_{i=1}^{N} WiXi + Bj$$

Équation 1

Où n représente le nombre d'entrées, xi représente la variable d'entrée I, w est la matrice de pondération entre les couches d'entrée et cachée (d*dl) et b est le vecteur de biais de la deuxième couche. Plusieurs formules de fonction d'activation peuvent être utilisées dans MLP. La fonction la plus populaire des travaux précédents est la sigmoïde, qui convertit le résultat sur une échelle de 0 à 1. La fonction sigmoïde est présentée dans (Équation 3)

$$Fi(x) = \frac{1}{1+e^{-H1j}}$$

Équation 2

Ce processus est répété pour la deuxième couche cachée. Enfin, la somme pondérée de la sortie de la deuxième couche cachée est calculée, en spécifiant les probabilités malveillantes et bénignes. Par conséquent, la sortie finale du neurone i peut être obtenue par (Equation 4):

$$Yi = fi\ (\sum_{i=1}^{N} w1xi + Bj)$$

Équation 3

Une fois l'architecture MLP conçue, une étape d'apprentissage est mise en œuvre pour ajuster et mettre à jour les poids du réseau. Ces poids sont rationalisés pour évaluer les résultats et minimiser l'erreur des sorties. Le processus d'apprentissage de MLP est une tâche difficile qui peut démontrer la capacité de MLP à résoudre différents types de problèmes d'optimisation [48].

- **Plateformes :**
- **Python**

Python est un langage de programmation interprété, orienté objet et de haut niveau avec une sémantique dynamique. Il est populaire pour le développement rapide d'applications grâce à ses structures de données de haut niveau intégrées, sa saisie dynamique et sa liaison dynamique.

- **Google Colab**

Colab est un produit Jupyter Notebook de Google Research. Les développeurs de programmes Python peuvent utiliser ce bloc-notes pour écrire et exécuter du code de programme Python aléatoire simplement à l'aide d'un navigateur Web.

En un mot, Colab est la version hébergée dans le cloud de Jupyter Notebook. Pour utiliser Colab, vous n'avez pas besoin d'installer et d'exécuter ou de mettre à niveau le matériel de votre ordinateur pour répondre aux exigences de charge de travail élevée en CPU/GPU de Python. De plus, Colab vous donne un accès gratuit à l'infrastructure informatique telle que le stockage, la mémoire, les capacités de traitement, les unités de traitement graphique (GPU) et unités de traitement de tenseur (TPU).

Google a spécifiquement programmé ce moteur de codage Python basé sur le cloud en gardant à l'esprit les besoins des programmeurs d'apprentissage automatique, des analystes de données volumineuses, des scientifiques des données, des chercheurs en IA et des apprenants Python.

- **Scikit-learn (Sklearn)**

Scikit-learn, également connu sous le nom de Sklearn, est la bibliothèque la plus puissante et la plus puissante pour l'apprentissage automatique en Python. Il fournit une sélection d'outils

puissants pour l'apprentissage automatique et la modélisation statistique, y compris la classification, la régression et le clustering via une interface cohérente en Python. Cette bibliothèque, principalement écrite en Python, est basée sur NumPy, SciPy et Matplotlib

Scikit-learn a été initialement développé par David Cournapeau dans le cadre du projet de codage d'été de Google en 2007. Matthieu Brucher a ensuite rejoint le projet et a commencé à l'utiliser pour sa thèse. En 2010, l'INRIA adhère et la première version publique (v0.1 beta) est publiée fin janvier 2010

Le projet compte actuellement plus de 30 contributeurs actifs et a reçu le soutien financier de l'INRIA, de Google, de Tinyclues et de la Python Software Foundation.

Pourquoi utiliser Scikit-learn ?

Algorithmes d'apprentissage automatique :

Scikit-learn couvre la plupart des algorithmes d'apprentissage automatique grâce à un excellent support communautaire : la possibilité d'effectuer des tâches d'apprentissage automatique en Python est l'une des principales raisons pour lesquelles Scikit-learn est si variable car Python est facile à apprendre et utiliser (Apprendre Python ici) et a déjà une grande communauté d'utilisateurs qui peuvent maintenant faire de l'apprentissage automatique sur une plate-forme avec laquelle ils sont confortablement familiarisés.

Organigramme des algorithmes :

Contrairement à d'autres langages de programmation où les utilisateurs sont souvent confrontés au problème de devoir choisir entre plusieurs implémentations concurrentes du même algorithme, Scikit-learn dispose d'une table de triche ou d'un diagramme d'algorithme pour aider les utilisateurs.

3.5. Conclusion

Dans ce chapitre nous avons d'abord présenté l'apprentissage automatique avec ses trois catégories supervisées, non supervisé et semi-supervisé. Ensuite, nous avons abordé la détection d'anomalies et les types d'anomalies ainsi que l'idée fondamentale de la détection d'anomalies.

Le prochain chapitre sera entièrement consacré à notre travail, qui consiste en la proposition d'une approche spécifique pour la détection des intrusions dans IoT.

4 : Approche proposé

Introduction

L'objectif de cette première proposition est de développer une architecture de détection d'intrusion basée sur le Fog computing pour protéger les réseaux IoT. Le système de détection d'intrusion proposé utilise l'algorithme du perceptron multicouche (MLP) pour résoudre le problème de réapprentissage dans les environnements IoT. Ce système présente une période d'apprentissage très courte, détecte les attaques à faible fréquence avec un taux élevé de détection, minimise les fausses alarmes et réduit significativement la latence de l'IDS. Il effectue une détection d'anomalie dans une première phase et fournit une classification précise des attaques pour aider à prendre des mesures de prévention dans une seconde phase. Les résultats ont été évalué par rapport à l'ensemble de tests à l'aide de différentes métriques d'évaluation telle que : l'accuracy, la précision, le rappel, le taux de vrais positif et le taux de faux positif

Description générale de l'approche

Dans cette section, nous décrivons les différents composants de l'architecture et leur utilité. Comme illustré dans la figure 3, notre système de détection est basé sur le Fog computing et comprend trois niveaux (Couche de nœuds IoT, Couche de nœuds de brouillard, Couche cloud)

La structure de modèle proposé

L'architecture de détection d'intrusion proposée exploite les capacités de stockage, de calcul et de puissance des nœuds de Fog ainsi que des couches de serveurs cloud. De plus, elle est conçue pour fonctionner à trois niveaux de calcul dans le Fog : les nœuds IoT, les nœuds de Fog et les couches de cloud.

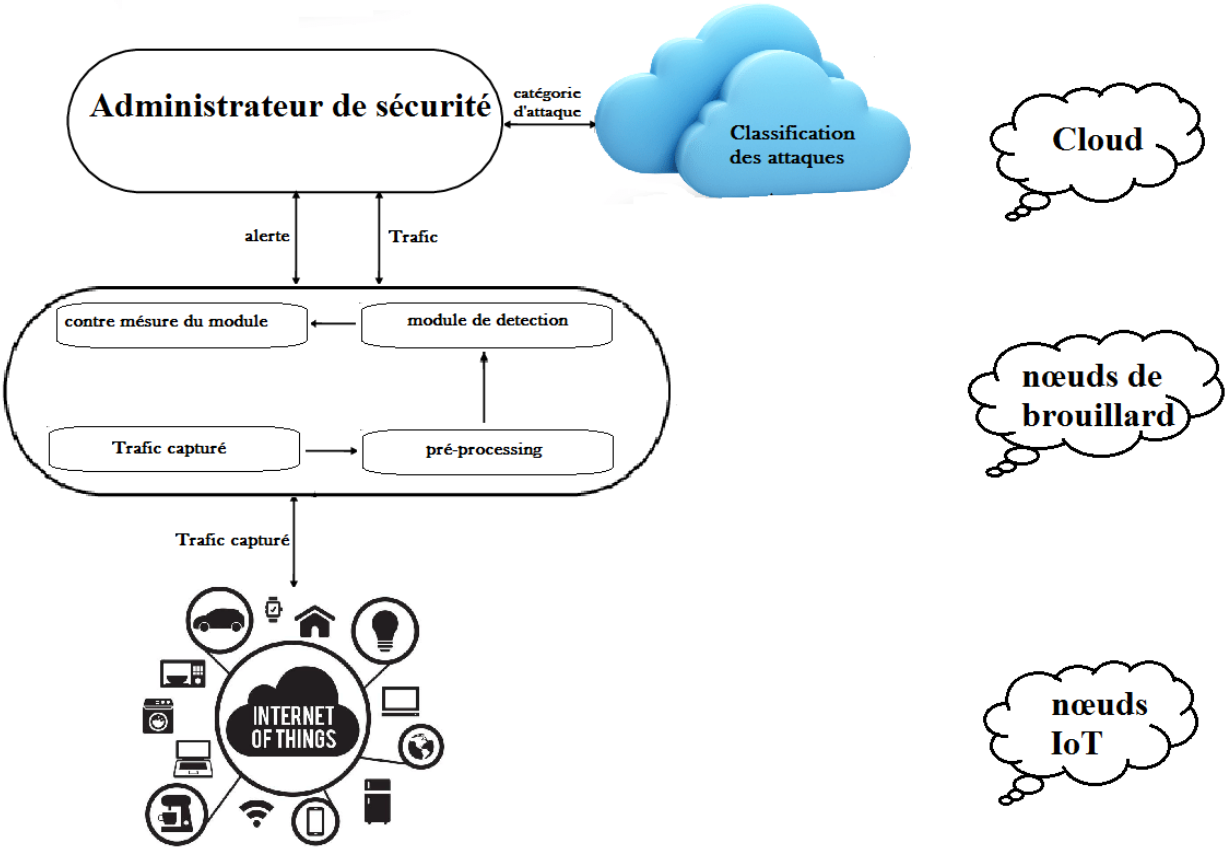

Figure 3 : Architecture proposé

Couche de nœuds IoT :

Dans cette couche, nous avons utilisé un ensemble de données à savoir IOT network intrusion. Chaque dispositif de Fog est équipé d'un module de détection situé au niveau du nœud de Fog. Ce module est chargé d'analyser et de classifier le trafic réseau en comportement normal ou en attaques. Il fonctionne indépendamment de la couche cloud, ce qui permet d'éviter toute latence. Chaque dispositif de Fog est responsable de la surveillance et de la sécurisation de son réseau IoT associé. Le nœud de Fog capture tout le trafic de son réseau IoT respectif en mode promiscuité.

Couche de nœuds de brouillard :

La couche de brouillard assume la responsabilité de l'entraînement du modèle et de l'hébergement de différents modules, tels que le module de prétraitement des caractéristiques et le module de détection du système. Un nœud de brouillard de coordination est nécessaire pour le partage collaboratif des paramètres. Cette approche vise à fournir une autonomie de détection locale grâce à l'apprentissage, au traitement et à l'ajustement des paramètres, tout en accélérant l'apprentissage des données à proximité de la source et en évitant les problèmes de latence.

Lorsqu'un nœud de brouillard reçoit du trafic réseau, il le transmet à différents modules, tels que le module de prétraitement, le module de détection et le module de contre-mesure. Le module de

prétraitement des données utilise un algorithme pour convertir les chaînes en une base de données spécifique, puis normalise ces données pour garantir la qualité des données d'entrée et améliorer l'efficacité de la détection.

Le module de détection analyse les attributs du trafic capturé et les classe comme comportement légitime ou trafic anormal (attaques). Pour assurer une réponse rapide et minimiser les dommages potentiels causés par un trafic anormal, ce module nécessite une latence très faible. Étant donné qu'il s'agit d'une classification binaire, un modèle moins complexe qu'une classification multi-classes est préférable.

Couche cloud :

Cette couche a pour responsabilité de classifier les attaques en différents types tels que le déni de service DoS, l'interception d'informations (MitM) et les attaques par analyse. La classification des attaques requiert un modèle plus complexe et demande davantage de ressources. De plus, cette tâche est moins sensible à la latence que la première couche.

Dans cette couche, nous avons développé un modèle de classification d'attaques en utilisant également l'algorithme MLP. Ce modèle est déployé dans le cloud en tant que classificateur de second niveau. Lorsqu'une catégorie d'attaque est prédite, les informations sont transmises à l'administrateur de sécurité afin qu'il puisse appliquer des mécanismes de prévention complémentaires et plus précis

Le mode de fonctionnement de notre modèle

Notre IDS fonctionne selon un mode de fonctionnement en deux étapes : le traitement des caractéristiques.

La méthode proposée, telle qu'illustrée à la figure 4, se compose de deux étapes principales :

1. La première étape est le prétraitement des données, qui se compose de quatre étapes. La première étape consiste à extraire toutes les caractéristiques du flux à partir d'un instantané brut du réseau à l'aide du logiciel CICFlowMeter. Cet outil traite, analyse et extraire le trafic réseau IoT en convertant des informations sur les flux de données du pcap au csv. La deuxième est le codage numérique 1-N et la troisième consistant à normaliser les données. La dernière étape est la sélection des attributs, nous utilisons la méthode : le coefficient de corrélation de Pearson (PCC), pour sélectionner un sous-ensemble représentatif des attributs pour les attributs donnés. Cette étape permet une classification précise du trafic provenant des attaques de botnet.

2. La deuxième étape concerne les résultats de modèle de classification construits à l'aide d'un ensemble de caractéristiques. Dans ce travail, nous nous concentrons sur l'algorithme de classification Multi-Layer perceptron (MLP) : MLPClassifier

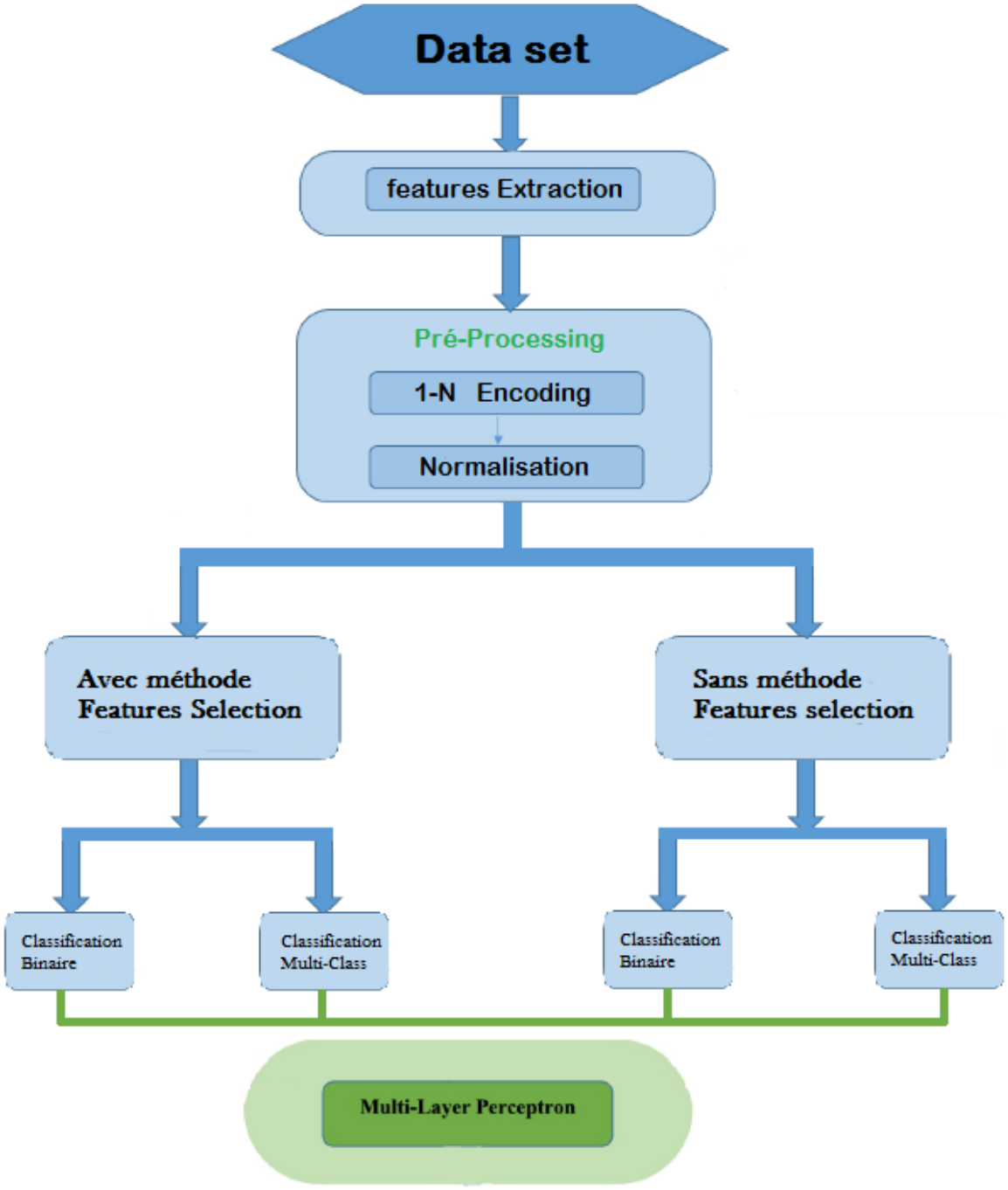

Figure 4 : Les étapes de l'approche proposé

Prétraitement des données

Le prétraitement des données est une méthode utilisée pour transformer des données brutes en un ensemble de données propre. En d'autres termes, lorsque des données sont collectées à partir de diverses sources, elles sont d'abord collectées sous une forme brute qui ne se prête pas à l'analyse. Cette méthode est effectuée avant l'entraînement des algorithmes d'apprentissage automatique, car ces algorithmes apprennent à partir des données et les résultats obtenus sont fortement dépendants de la qualité des données. Par conséquent, plusieurs étapes doivent être suivies pour convertir les données en un ensemble de données nettoyées, un processus connu sous le nom de prétraitement des données. Selon notre approche, l'étape de prétraitement des données se compose de quatre étapes : extraction de caractéristiques à l'aide de CICFlowMeter, le codage numérique 1-N, normalisation des données, et enfin sélection d'attributs à l'aide d'un ensemble de filtres. En fonction de l'importance de l'attribut

Extraction des caractéristiques

Différentes méthodes sont utilisées pour extraire les flux du trafic réseau, et l'une des approches courantes est l'utilisation d'un extracteur de flux. Dans notre cas, nous avons utilisé l'outil open source appelé CICFlowMeter.

CICFlowMeter prend en entrée des fichiers pcap et génère des flux doubles en extrayant les fonctionnalités de ces flux. Il permet la création de flux bidirectionnels, où le premier paquet définit le sens direct (source vers destination) et le sens inverse (destination vers source). Cette approche permet de calculer séparément les caractéristiques statistiques liées au temps dans les deux sens. CICFlowMeter offre également des fonctionnalités supplémentaires telles que la sélection de fonctionnalités à partir d'une liste existante, conversion des données su pcap au CSV, l'ajout de nouvelles fonctionnalités et le contrôle du délai d'expiration des flux.

Le Codage numérique 1-N

Le module de détection utilisé dans notre approche ne peut pas traiter directement l'ensemble de données d'intrusion du réseau IoT dans son format d'origine. Pour remédier à cela, nous utilisons un système de codage 1-N qui convertit les entités non numériques en caractéristiques numériques avant d'appliquer l'algorithme de classification MLP.

L'ensemble de données d'intrusion du réseau IoT comprend une caractéristique non numérique et 84 caractéristiques numériques. Ainsi, nous utilisons un système de codage pour traiter les caractéristiques non numériques, telles que le "Timestamp". Par exemple, nous convertissons les attributs distincts de la fonction "Timestamp", tels que AM et PM, en valeurs numériques en utilisant un système de codage approprié.

Normalisation

La normalisation est nécessaire car plusieurs caractéristiques des ensembles de données d'intrusion du réseau IoT ont des plages très étendues entre leurs valeurs maximales et minimales. Cela rend les valeurs de ces caractéristiques incomparables et inappropriées pour le traitement. Ainsi, nous normalisons ces caractéristiques en utilisant un processus de normalisation basé sur le calcul de la différence absolue moyenne, afin de mapper toutes les valeurs de caractéristiques à l'intervalle [0,1]. Cela est réalisé en utilisant l'équation (Equation 5) :

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

Équation 4

Où $\boldsymbol{X}$ désigne chaque point de données, $\boldsymbol{X_{min}}$ désigne la valeur minimale de tous les points de données et $\boldsymbol{X_{max}}$ désigne la valeur maximale de tous les points de données pour chaque entité.

Sélection d'attributs

Nous avons utilisé la méthode de sélection : le coefficient de corrélation de Pearson (PCC)

La méthode de sélection des caractéristiques utilisée dans notre étude est le coefficient de corrélation de Pearson (PCC). Cette méthode est une mesure de dépendance entre deux variables aléatoires X et Y, le coefficient de corrélation de Pearson ρ est donné par l'équation (Equation 6):

$$\rho = \frac{cov(\mathrm{X},\mathrm{Y})}{\sqrt{\sigma^2(\mathrm{X})\sigma^2(\mathrm{Y})}}$$

Équation 5

Où cov est la covariance et σ la variance, la valeur de ρ est comprise entre -1 et 1, p est proche des valeurs extrêmes -1 et 1 si X et Y sont fortement corrélés, et ρ=0 si X et Y sont totalement non corrélé. Ainsi, une caractéristique qui est fortement corrélée à d'autres caractéristiques est redondante

De cette méthode, nous avons obtenu 34 attributs.

{'ACK Flag Cnt', 'Bwd Header Len', 'Bwd IAT Max', 'Bwd IAT Min', 'Bwd Pkt Len Mean', 'Bwd Pkt Len Std', 'Bwd Pkts/b Avg', 'Bwd Pkts/s', 'Bwd Seg Size Avg', 'Flow Byts/s', 'Flow IAT Max', 'Flow IAT Min', 'Fwd Act Data Pkts', 'Fwd Header Len', 'Fwd IAT Max', 'Fwd IAT Min', 'Fwd IAT Tot', 'Fwd Pkt Len Mean', 'Fwd Pkt Len Std', 'Fwd Pkts/s', 'Fwd Seg Size Avg', 'Idle Max', 'Idle Min', 'PSH Flag Cnt', 'Pkt Len Max', 'Pkt Len Mean', 'Pkt Len Min', 'Pkt Len Std', 'Pkt Len Var', 'Pkt Size Avg', 'Subflow Bwd Byts', 'Subflow Fwd Byts', 'Tot Bwd Pkts', 'TotLen Fwd Pkts'}

La figure 5. Présente la matrice de corrélation, Les attributs avec une corrélation élevée sont plus linéairement dépendants et ont donc presque le même effet sur la variable dépendante.

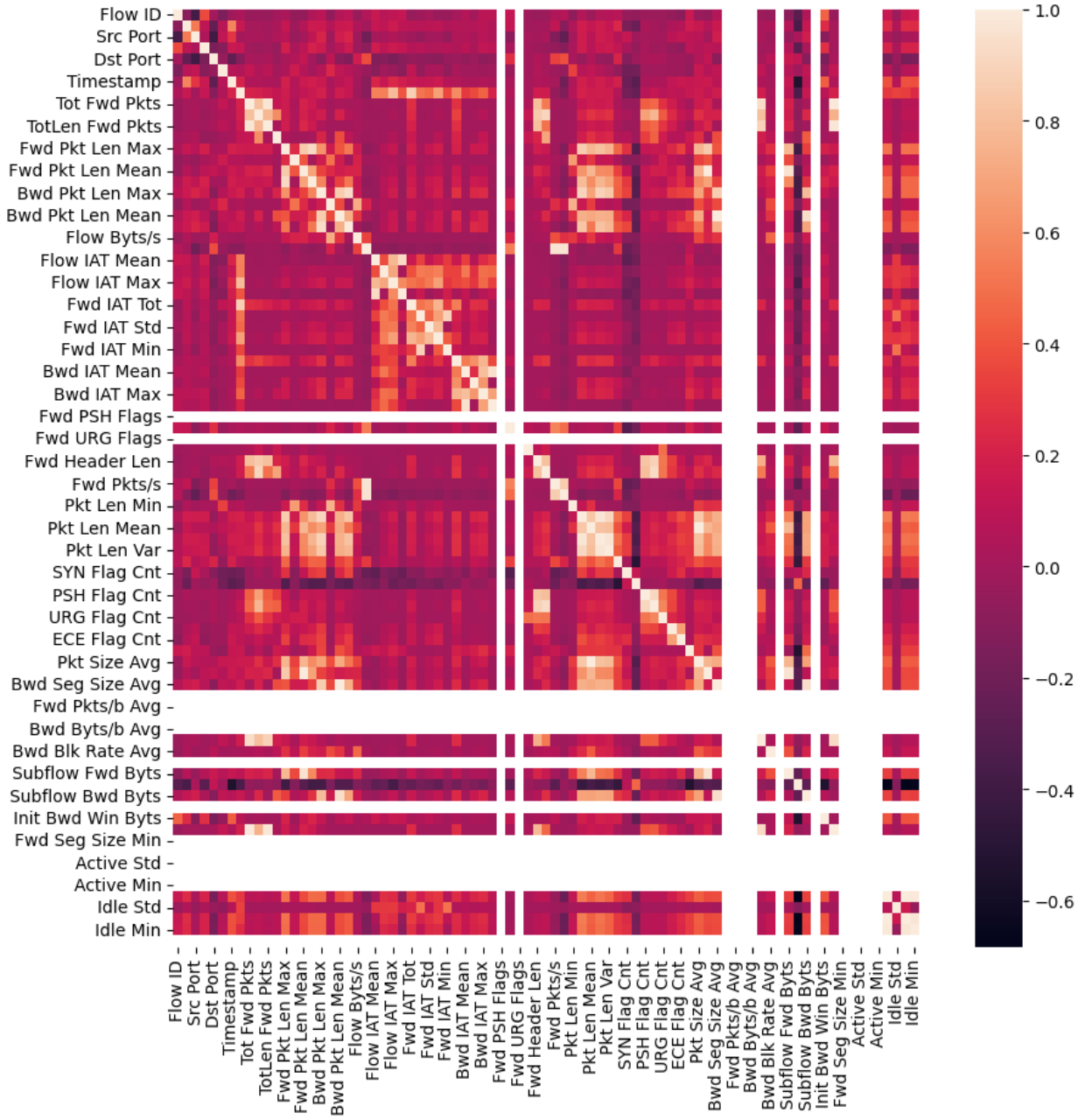

Figure 5 : Matrice de corrélation des attributs sélectionnés

L’entraînement du modèle

L'architecture du perceptron multicouche (MLP) se compose d'un réseau de neurones profond avec plusieurs couches. La fonction d'activation utilisée pour toutes les couches cachées dans le MLP est la fonction Relu. Pour la couche de sortie du MLP, la fonction d'activation utilisée est la fonction sigmoïde, qui génère une valeur d'évaluation comprise entre 0 et 1 pour chaque neurone.

La fonction softmax est utilisée pour transformer les sorties de chaque classe en valeurs comprises entre 0 et 1, en les divisant par la somme des sorties. Cela permet d'obtenir la probabilité que l'entrée appartienne à une classe particulière.

Le MLP est capable d'extraire automatiquement des caractéristiques de haut niveau, de sorte que les paramètres entraînés des couches cachées peuvent être utilisés pour initialiser les paramètres entraînés du MLP. Ensuite, un ensemble de données d'apprentissage est utilisé pour affiner le classifieur MLP.

Enfin, les fonctionnalités de test sont introduites dans le classifieur MLP entraîné afin de détecter les attaques

Evaluation des performances

Dans cette section, nous présentons la mise en œuvre de notre méthode de test. Pour commencer, nous avons utilisé Python pour implémenter notre méthode dans le cadre du service cloud gratuit Google Colab (Collaboration). Google Colab est basé sur Jupyter Notebook et est spécialement conçu pour la formation et la recherche en apprentissage automatique. Il offre un accès facile à diverses bibliothèques qui permettent l'utilisation des services fournis par Google.

L'avantage de Colab est que tous les scripts Jupyter sont enregistrés sur Google Drive, ce qui facilite leur partage efficace où que vous soyez. Afin de ne pas perdre de données lors de la déconnexion du service, nous avons décidé de télécharger une base de données contenant des fichiers CSV sur Google Drive. Nous les lisons à partir de là chaque fois que nous en avons besoin.

Pour l'implémentation de classificateur **MLPClassifier**, nous avons utilisé la bibliothèque Python Scikit-learn [49].

Pour évaluer notre méthode proposée, nous avons utilisé quatre indicateurs de performance. Ces chiffres sont calculés à l'aide de quatre mesures différentes : vrai positif (TP), vrai négatif (TN), faux positif (FP) et faux négatif (FN).

- **TP** : si une instance anormale est classée comme anormale, elle est acceptée comme TP.

- **FP** : si une instance normale est classée comme anormale, elle est acceptée comme FP.

- **TN** : si une instance normale est classée comme normale, elle est acceptée comme TN.

- **FN** : si une instance anormale est classée comme normale, elle est acceptée comme FN.

Précision (Accuracy) :

La précision (Accuracy) mesure la proportion de cas anormaux et normaux correctement classés par rapport au nombre total de cas. Il évalue la capacité générale d'un modèle à être bien formé et à bien performer. Cependant, la précision ne fournit pas de détails spécifiques sur l'application du modèle à des problèmes spécifiques. Il permet d'évaluer la précision globale du modèle, mais ne fournit pas d'informations détaillées sur ses performances pour des instances ou des classes spécifiques à l'aide du fonction (Equation 7).

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

Équation 6

Précision (Précision) :

La précision est le rapport entre le nombre d'anomalies correctement classées et le nombre total de cas classés comme anormaux par le modèle. Il mesure la fréquence à laquelle le modèle est correct lorsqu'il fait une prédiction positive. Il est important de noter que la précision ici est différente de la précision décrite précédemment (Accuracy), qui mesure la précision globale du modèle dans toutes les classes, indépendamment de l'anomalie ou de la normale. La précision est calculée à l'aide de la formule (Equation 8).

$$\boldsymbol{Précision} = \frac{TP}{(TP + FP)}$$

Équation 7

Rappel (Recall) :

Le rappel (Recall) est le rapport entre le nombre d'instances anormales correctement classées et le nombre total d'instances anormales réelles. Il évalue la capacité d'un modèle de classification à détecter toutes les instances pertinentes. Le rappel est particulièrement utile lorsque le taux de faux négatifs est élevé, car il mesure l'efficacité de l'approche dans l'identification des réseaux de botnet à l'aide du fonction (Equation9).

$$\boldsymbol{Recall} = TP\frac{TP}{(TP + FN)}$$

Équation 8

F1-score :

C'est une mesure de la précision d'un test. Elle considère à la fois la précision et le rappel du test pour calculer le score, c'est la moyenne harmonique de la précision et du rappel, elle atteint sa meilleure valeur à 1, le F1-score est calculé à l'aide de la formule (Equation 10).

F1 Score = 2*(Recall * Precision) / (Recall + Precision)

Équation 9

Résultat et discussion

Dans cette section, nous présenterons et analyserons les résultats obtenus après la mise en œuvre de l'algorithme décrit au chapitre 3.

Pour évaluer les performances de la classification, nous utiliserons plusieurs indicateurs tels que la précision, l'exactitude, la précision, le rappel, le f1-score.

La couche normale de notre ensemble de données comprendra des données d'entraînement 70% et des données de test 30%. Nous prévoyons donc de faire deux expériences distinctes.

Nous avons utilisé deux ensembles de bases de données distincts. La première base de données sera dédiée à la classification binaire, tandis que la seconde sera utilisée pour la classification multi-classes et les résultats sont illustrées dans les tableaux 3 et 4.

- **La première base de données :**

Nous avons regroupé le trafic de IOT pour construire la classe normale. Et pour la classe anomalie nous avons regroupé les types de trafic malveillant, considérons la classification binaire.

Des nombreuses pistes sont prises en considération pour améliorer les résultats de détection en termes de faux négatifs et de faux positifs telles que la sélection des attributs pour le classifieur MLP Classifier.

Les résultats sont résumés dans le tableau 3 :

	Type	Accuracy	Precision	Recall	F1-Score
Avec méthode de sélection	**Normale**	99%	93%	87%	90%
	Attaque		100%	100%	100%
Sans méthode de sélection	**Normale**	99%	82%	80%	81%
	Attaque		99%	99%	99%

Tableau 3 : Résultat du classification Binaire

Le tableau ci-dessus présente les résultats des métrique obtenus pour l'évaluation de classifieur proposé à l'aide d'apprentissage de l'ensemble de donnée pour la comparaison des valeurs obtenues nous avons pris la métrique précision (Accuracy) qui indique le nombre d'instances anormales et normales correctement classées

Avant la sélection des attributs L'algorithme MLP a obtenu des résultats come ci-dessous : Pour le type de trafic normal, nous obtenons une Accuracy de 99%, ce qui est similaire à la précédente méthode de sélection. Cependant, la précision chute à 82%, ce qui signifie que parmi les exemples classés comme normaux, seulement 82% étaient réellement normaux. Le rappel est de 80%, ce qui indique une certaine difficulté à identifier tous les exemples normaux. Le F1-Score est de 81%, ce qui est légèrement inférieur à celui obtenu avec la méthode de sélection.

Pour le type de trafic attaque, nous observons une Accuracy de 99%, identique à la méthode de sélection. La précision est de 99%, ce qui est élevé, mais légèrement inférieur à la méthode de sélection. Le rappel est également de 99%, montrant une capacité élevée à identifier les exemples d'attaques. Le F1-Score est de 99%, ce qui est proche des performances obtenues avec la méthode de sélection.

Après la sélection des attributs nous avons obtenu une petite amélioration comme suivant :

Pour le type de trafic normal, nous observons une Accuracy de 99%, ce qui signifie que le modèle a correctement classé 99% des exemples. La précision est de 93%, ce qui indique que parmi les exemples classés comme normaux, 93% étaient effectivement normaux. Le rappel (Recall) est de 87%, ce qui représente la capacité du modèle à identifier correctement les exemples normaux parmi tous les exemples normaux présents dans les données. Le F1-Score, qui combine la précision et le rappel en une seule mesure, est de 90%.

Pour le type de trafic attaque, nous constatons une Accuracy de 99%, ce qui signifie que le modèle a correctement classé 99% des exemples d'attaques. La précision est de 100%, ce qui indique que tous les exemples classés comme attaques étaient effectivement des attaques. Le rappel est également de 100%, ce qui montre que le modèle a identifié tous les exemples d'attaques parmi tous les exemples d'attaques présents dans les données. Le F1-Score est de 100%, ce qui suggère une excellente performance pour la détection des attaques.

- **La deuxième base de données :**

Nous avons construit notre modèle avec le trafic légitime des objets réelles séparément, le modèle a été testé par les différents trafics malveillant généré par les botnets (*dos-synflooding, mirai-ackflooding, mirai-hostbruteforce, mirai-httpflooding, mirai-udpflooding, mitm-arpspoofing, scan-hostport, scan-portos*) Nous voulons savoir par cette expérimentation à quel point notre modèle peut différencier entre le trafic légitime de chaque objet et le trafic malveillant, ainsi l'efficacité de modèle à détecter les différents types de trafic malveillant considèrent la classification Multi-Classe ainsi qu'on pris en considèrent la sélection des attributs pour améliorer la détection

	Type de malveillant	Accuracy	Precision	Recall	F1-Score
Avec méthode de sélection	**Benign**	90%	100%	82%	90%
	dos-synflooding		99%	99%	99%
	mirai-ackflooding		48%	83%	61%
	mirai-hostbruteforce		97%	93%	95%
	mirai-httpflooding		86%	62%	72%
	mirai-udpflooding		38%	22%	28%
	mitm-arpspoofing		95%	97%	96%
	scan-hostport		98%	99%	99%
	scan-portos		99%	100%	99%
Sans méthode	**Benign**	89%	95%	86%	90%
	dos-synflooding		100%	100%	100%
	mirai-ackflooding		49%	77%	60%
	mirai-hostbruteforce		88%	95%	92%
	mirai-httpflooding		70%	73%	72%

de sélection	**mirai-udpflooding**		00%	00%	00%
	mitm-arpspoofing		89%	92%	90%
	scan-hostport		99%	96%	97%
	scan-portos		99%	99%	99%

Tableau 4 : Résultat du classification Multi-Class

Le tableau ci-dessus présente les résultats des métrique obtenus pour l'évaluation de classifieur proposé à l'aide d'apprentissage de l'ensemble de donnée pour la comparaison des valeurs obtenues nous avons pris la métrique précision (Accuracy) qui indique le nombre d'instances anormales et normales correctement classées

Avant la sélection des attributs L'algorithme MLP a obtenu un résultat dans cet ensemble de données comme ci-dessous :

Pour le type d'attaque dos-synflooding les résultats montrent une amélioration de l'accuracy à 90% et des mesures de précision, rappel et F1-score de 100%. Et pour le type d'attaque mirai-ackflooding bien que l'accuracy soit de 90%, les mesures de précision, rappel et F1-score sont encore relativement faibles, indiquant des difficultés persistantes pour classifier correctement ce type de trafic. Et pour le type d'attaque mirai-hostbruteforce les résultats montrent une précision et un rappel améliorés à 88% et 95% respectivement, avec un F1-score de 92%. Et pour le type d'attaque mirai-httpflooding les résultats montrent une légère amélioration de la précision à 70%, mais le rappel reste relativement faible à 73%. Cela indique toujours une certaine difficulté à détecter correctement ce type de trafic. Et pour le type d'attaque mirai-udpflooding les performances sont très faibles avec une précision, un rappel et un F1-score de 0%. Cela suggère que le modèle a du mal à classifier ce type de trafic sans la méthode de sélection. Et pour le type d'attaque mitm-arpspoofing les résultats montrent une précision et un rappel légèrement, avec un F1-score de 90%. Cependant, la performance globale reste relativement similaire. Pour le type d'attaque scan-hostport les résultats montrent une précision à 99% et un rappel à 96%. Cependant, la performance globale reste élevée pour ce type de trafic. Pour le type d'attaque scan-portos les résultats restent similaires à ceux de la méthode de sélection, avec une précision de 99%, un rappel de 99% et un F1-score de 99%

Après la sélection des attributs nous avons obtenu une petite amélioration comme si de suit :

Pour le type d'attaque dos-synflooding les résultats montrent une accuracy élevée de 89%, avec une précision, un rappel et un F1-score de 99%. Cela indique une bonne capacité à identifier ce type de trafic, avec un faible taux de faux positifs et de faux négatifs. Et pour le type d'attaque mirai-ackflooding bien que l'accuracy soit de 89%, les autres mesures (précision, rappel, F1-score) sont relativement plus faibles. Cela suggère que le modèle peut avoir du mal à distinguer correctement ce type de trafic, avec un taux de faux positifs élevé. Et pour le type d'attaque

mirai-hostbruteforce les résultats montrent des performances solides avec une accuracy de 89% et des mesures de précision, rappel et F1-score autour de 97%. Cela indique une bonne capacité à détecter et classifier ce type de trafic. Pour le type d'attaque mirai-httpflooding les résultats montrent une précision décente de 86%, mais un rappel plus faible de 62%. Cela suggère que le modèle peut manquer certains exemples positifs de ce type de trafic, conduisant à un taux de faux négatifs plus élevé. Pour le type d'attaque mirai-udpflooding les performances sont relativement faibles avec une précision de 38%, un rappel de 22% et un F1-score de 28%. Cela indique que le modèle peut avoir des difficultés à détecter correctement ce type de trafic, avec un risque élevé de faux positifs et de faux négatifs. Pour le type d'attaque mitm-arpspoofing les résultats montrent de bonnes performances avec une accuracy de 89% et des mesures de précision, rappel et F1-score autour de 95% à 97%. Cela indique une bonne capacité à détecter et classifier ce type de trafic. Pour le type d'attaque scan-hostport les résultats sont solides avec une accuracy de 89% et des mesures de précision, rappel et F1-score autour de 98% à 99%. Cela indique une capacité élevée à détecter et classifier ce type de trafic. Et pour le type d'attaque scan-portos les résultats sont également bons avec une accuracy de 89%, une précision de 99% et un rappel de 100%. Cela suggère une capacité élevée à identifier ce type de trafic sans trop de faux positifs ou de faux négatifs

En conclusion, la méthode de sélection semble améliorer les performances de classification pour certains types de trafic, tandis que pour d'autres types, elle peut entraîner une baisse de la précision et du rappel. Il est important d'analyser attentivement les résultats et d'ajuster la méthode de sélection en conséquence pour améliorer la performance globale du modèle de classification.

D'après les résultats obtenus dans les deux ensembles de données nous pouvons voir que les performances d'un classifieur dépendent fortement de la qualité de représentation des données à traiter, ce qui implique généralement l'obligation de représenter ces dernières au moyen d'un nombre élevé des attributs représentatifs. Il est alors fréquent qu'une partie de celles-ci ne contienne que des informations non pertinentes, redondantes ou inutiles à la tâche de classification, rendant des résultats moins satisfaisants. Il est donc important de limiter le nombre d'attributs pris en compte, de manière à optimiser ses performances. C'est exactement ce que fait l'étape de sélection d'attributs.

Comparaison de notre approche avec d'autre approche

On a été choisie pour la comparaison avec notre approche, la mise en œuvre de l'algorithme **SGD Classifier** sur le même ensemble de données Il faut bien noter Le **SGD Classifier** n'est pas toujours pratique pour la détection d'anomalies

Nous avons appliqué l'algorithme sur le premier ensemble de données (la classification binaire) Avec et sans la sélection des attributs ainsi que sur le deuxième ensemble de données (la classification multi-class) avec et sans la méthode de sélection des attributs

Pour évaluer la performance et la comparaison de notre approche (MLP Classifier) et (SGD Classifier) nous avons utilisé la métrique d'évaluation la précision (Accuracy)

Les résultats sont illustrés dans les tableaux 5 et 6

- **Le premier ensemble de données :**

Les résultats de classification binaire du deux algorithmes

Classifieur	**Accuracy**
MLP Classifier	99%
SGD Classifier	96%

Tableau 5 : Comparaison des résultats MLP et SGD

Une précision de 99% pour le MLP Classifier est assez élevée et indique que le modèle est très performant dans la tâche de classification. Cela signifie que le modèle a réussi à prédire correctement 99% des échantillons de test.

D'autre part, une précision de 96% pour le SGD Classifier est également respectable, bien qu'un peu inférieure à celle du MLP Classifier. Cela signifie que le modèle a correctement prédit 96% des échantillons de test.

En fin de compte, pour d'obtenir la plus haute précision possible, le MLP Classifier semble être le meilleur choix avec sa précision de 99%.

- **Le deuxième ensemble de données :**

Les résultats de classification multi-class du deux algorithmes

Classifieur	Accuracy
MLP Classifier	90%
SGD Classifier	78%

Tableau 6 : Comparaison des résultats MLP et SGD

Le MLP Classifier a obtenu une précision plus élevée de 90%. Cela peut indiquer qu'il est capable de mieux généraliser les modèles et de faire des prédictions plus précises sur de nouvelles données. La précision de 90% peut être considérée comme relativement bonne, indiquant que le modèle a réussi à classer correctement une grande majorité des échantillons.

D'autre part, le SGD Classifier a obtenu une précision de 78%. Bien que cette précision soit inférieure à celle du MLP Classifier, elle peut encore être considérée comme raisonnablement bonne. Cependant, il est important de noter que le SGD Classifier a une précision inférieure, ce qui peut indiquer qu'il peut être moins performant dans la classification de certaines classes ou qu'il a plus de difficultés à généraliser les données.

En fin de compte, pour d'obtenir la plus haute précision possible, le MLP Classifier semble être le meilleur choix avec sa précision de 90%

Conclusion

Dans ce chapitre, nous avons présenté notre architecture pour la détection d'intrusion dans les réseaux IoT et sa mise en œuvre. L'architecture proposée est basée sur un classificateur MLP et utilise une méthode de filtrage pour sélectionner les attributs (PCC) après avoir l'étape du prétraitement du données (1-N Encoding et Normalisation), En premier temps, nous avons décrivé l'architecture et nous avons exploré le mode de fonctionnement de cette approche. Nous avons discuté ensuite sur la méthode de sélection d'attributs pour ne conserver que les attributs pertinents dans le flux. Ensuite nous avons vu l'entrainement d'algorithme de classification utilisé dans notre approche Multi Layer Perceptron. Enfin, nous avons présenté et discuté de l'évaluation des performances de cet algorithme et nous avons donné une comparaison entre l'algorithme **MLP Classifier** et **SGD Classifier**

Conclusion Générale

Aujourd'hui, l'Internet des objets (IoT) connaît une croissance exponentielle et les appareils IoT sont devenus omniprésents dans notre vie quotidienne. Cependant, leur connectivité croissante et leur sécurité insuffisante en font des cibles attrayantes pour les cyber-attaquants, qui exploitent ces appareils pour lancer des attaques et former des cybercriminels dans un botnet. Dans le cadre de nos recherches, notre objectif est de proposer une architecture d'un système pour détecter les intrusions dans l'internet des objets basé sur le Fog computing afin de les protéger sans perturber le fonctionnement normal de ceux-ci. Nous avons développé un système robuste et fiable capable de détecter les activités malveillantes dans le réseau des objets connectés.

Notre approche comprend l'analyse du trafic réseau pour détecter les attaques de botnet. Pour mettre en œuvre cette approche, nous avons utilisé des outils puissants d'analyse et de traitement de données, tels que le puissant outil CICFlowMeter, ainsi que le langage Python avec ses bibliothèques. De plus, nous avons exploité la méthode de sélection des attributs de flux, PCC, pour améliorer les performances de notre méthode. Pour évaluer notre approche, nous avons testé avec le classificateur : MLP.

Références Bibliographiques

[1] Internet of Things Security : Principles and Practice Éditeur : Qinghao Tang, Springer Verlag, Singapore ; 1st ed. 2021 édition (28 janvier 2021) ISBN-10 : 9811599416 ISBN-13 : 978- 9811599415

[2] Jayavardhana Gubbi, Rajkumar Buyya, Slaven Marusic, and Marimuthu Palaniswami. Internet of things (iot): A vision, architectural elements, and future directions. Future generation computer systems, 29(7) :1645–1660, 2013.

[3] Daniele Miorandi, Sabrina Sicari, Francesco De Pellegrini, and Imrich Chlamtac. Internet of things : Vision, applications and research challenges. Ad hoc networks, 10(7) :1497–1516, 2012.

[4] Harald Sundmaeker, Patrick Guillemin, Peter Friess, and Sylvie Woelfflé. Vision and challenges for realising the internet of things. Cluster of European Research Projects on the Internet of Things, European Commision, 3(3):34–36, 2010.

[5] Loic Letondeur, François-Gaël Ottogalli, and Thierry Coupaye. A demo of application lifecycle management for iot collaborative neighborhood in the fog. 2017.

[6] Dave Evans. The internet of things: How the next evolution of the internet is changing everything. CISCO white paper, 1(2011) :1–11, 2011.

[7] Flavio Bonomi, Rodolfo Milito, Jiang Zhu, and Sateesh Addepalli. Fog computing and its role in the internet of things. pages 13–16, 2012.

[8] Luis M Vaquero and Luis Rodero-Merino. Finding your way in the fog : Towards a comprehensive definition of fog computing. ACM SIGCOMM Computer Communication Review, 44(5) :27–32, 2014.

[9] Rajesh Krishna Balan, Darren Gergle, Mahadev Satyanarayanan, and James Herbsleb. Simplifying cyber foraging for mobile devices. 2007.

[10] Tim Verbelen, Pieter Simoens, Filip De Turck, and Bart Dhoedt. Cloudlets : Bringing the cloud to the mobile user. pages 29–36, 2012.

[11] Michael Till Beck, Martin Werner, Sebastian Feld, and S Schimper. Mobile edge computing : A taxonomy. pages 48–55, 2014.

[12] Charles C Byers. Architectural imperatives for fog computing : Use cases, requirements, and architectural techniques for fog-enabled iot networks. IEEE Communications Magazine, 55(8) :14–20, 2017.

[13] Charith Perera, Yongrui Qin, Julio C Estrella, Stephan Reiff-Marganiec, and Athanasios V Vasilakos. Fog computing for sustainable smart cities : A survey. ACM Computing Surveys (CSUR), 50(3) :32, 2017. (CSUR), 50(3) :32, 2017.

[14] Niroshinie Fernando, Seng W Loke, and Wenny Rahayu. Mobile cloud computing : A survey. Future generation computer systems, 29(1) :84–106, 2013.

[15] Yaser Jararweh, Ahmad Doulat, Omar AlQudah, Ejaz Ahmed, Mahmoud Al-Ayyoub, and Elhadj Benkhelifa. The future of mobile cloud computing : integrating cloudlets and mobile edge computing. pages 1–5, 2016.

[16] Alexander Klemm, Christoph Lindemann, and Oliver P Waldhorst. A special-purpose peer-to-peer file sharing system for mobile ad hoc networks. 4 :2758–2763, 2003.

[17] Shanhe Yi, Zijiang Hao, Zhengrui Qin, and Qun Li. Fog computing : Platform and applications. pages 73–78, 2015.

[18] Yuyi Mao, Changsheng You, Jun Zhang, Kaibin Huang, and Khaled B Letaief. A survey on mobile edge computing : The communication perspective. IEEE Communications Surveys and Tutorials, 19(4) :2322–2358, 2017.

[19] Athanasios V Vasilakos, Zhe Li, Gwendal Simon, and Wei You. Information centric network : Research challenges and opportunities. Journal of network and computer applications, 52 :1–10, 2015.

[20] Ben Zhang, Nitesh Mor, John Kolb, Douglas S Chan, Ken Lutz, Eric Allman, John Wawrzynek, Edward Lee, and John Kubiatowicz. The cloud is not enough : Saving iot from the cloud. 2015.

[21] Mahadev Satyanarayanan. Cloudlets : At the leading edge of cloud-mobile convergence. pages 1–2, 2013.

[22] Pengfei Hu, Huansheng Ning, Tie Qiu, Yanfei Zhang, and Xiong Luo. Fog computing based face identification and resolution scheme in internet of things. IEEE transactions on industrial informatics, 13(4) :1910–1920, 2017.

[23] Yifan Wang, Tetsutaro Uehara, and Ryoichi Sasaki. Fog computing : Issues and challenges in security and forensics. 3 :53–59, 2015.

[24] Mung Chiang and Tao Zhang. Fog and iot : An overview of research opportunities. IEEE Internet of Things Journal, 3(6) :854–864, 2016.

[25] Díaz, M., Martín, C., Rubio, B., 2016. State-of-the-art, challenges, and open issues in the integration of internet of things and cloud computing. J. Netw. Comput. Appl. 67, 99–117

[26] Li, S., Wang, G., Yang, J., 2019a. Survey on cloud model based similarity measure of uncertain concepts. CAAI Trans. Intell. Technol. 4, 223–230.

[27] Roman, R., Zhou, J., Lopez, J., 2013. On the features and challenges of security and privacy in distributed internet of things. Comput. Network. 57, 2266–2279.

[28] Al-Turjman, F., Radwan, A., 2017. Data delivery in wireless multimedia sensor networks : challenging and defying in the iot era. IEEE Wireless Commun. 24, 126–131.

[29] Zhou, L., Chao, H.-C., 2011. Multimedia traffic security architecture for the internet of things. IEEE Network 25, 35–40.

[30] Atzori, L., Iera, A., & Morabito, G. (2010). The Internet of Things: A survey. Computer Networks, 54(15), 2787-2805.

[31] Roman, R., Najera, P., & Lopez, J. (2011). Securing the Internet of Things. IEEE Computer, 44(9), 51-58.

[32] Gubbi, J., Buyya, R., Marusic, S., & Palaniswami, M. (2013). Internet of Things (IoT): A vision, architectural elements, and future directions. Future Generation Computer Systems, 29(7), 1645-1660.

[33] Bandyopadhyay, D., & Sen, J. (2011). Internet of Things: Applications and challenges in technology and standardization. Wireless Personal Communications, 58(1), 49-69.

[34] Alaba, F. A., Aderounmu, G. A., & Siyanbola, T. O. (2017). Internet of Things (IoT) security: A review. Journal of Information Security, 8(4), 238-251.

[35] Cao, Y., & Zhang, Y. (2018). A survey on the security of the Internet of Things. Security and Communication Networks, 2018, Article ID 2568935.

[36] Liu, Y., Zhang, X., Chen, Y., & Li, H. (2019). Security challenges and solutions for the Internet of Things. IEEE Communications Magazine, 57(5), 21-27.

[37] A technical review of wireless security for the internet of things : Software defined radio perspective, José de Jesús Rugeles Uribe, Edward Paul Guillen, Leonardo S. Cardoso https://doi.org/10.1016/j.jksuci.2021.04.003.

[38] Perry Lea, Internet of Things for Architects : Architecting IoT solutions by implementing sensors, communication infrastructure, edge computing, analytics, and security

[39] Mahamat Charfadine Salim. Gestion dynamique et evolutive de regles de se´curitepour l'Internet des Objet. PhD thesis, Université de Reims Champagne-Ardenne, 2019.

[40] Kai Zhao and Lina Ge. A survey on the internet of things security. In 2013 Ninth international conference on computational intelligence and security, pages 663–667. IEEE, 2013. doi : 10.1109/cis.2013.145..

[41] aylor, Tooska Dargahi, Ali Dehghantanha, Reza M Parizi, and Kim-Kwang Raymond Choo. A systematic literature review of blockchain cyber security. Digital Communications and Networks, 6(2) :147–156, 2020. doi : 10.1016/j.dcan.2019.01.005.

[42] Leonel Santos, Carlos Rabadao, and Ramiro Gonçalves. Intrusion detection systems in internet of things : A literature review. In 2018 13th Iberian Conference on Information Systems and Technologies (CISTI), pages 1–7. IEEE, 2018. doi : 10.23919/cisti.2018. 8399291.

[43] Idriss Idrissi, Mostafa Azizi, and Omar Moussaoui. Iot security with deep learningbased intrusion detection systems : A systematic literature review. In 2020 Fourth International Conference On Intelligent Computing in Data Sciences (ICDS), pages 1–10. IEEE, 2020. doi : 10.1109/icds50568.2020.9268713.

[44] Serpil Ustebay, Zeynep Turgut, and Muhammed Ali Aydin. Intrusion detection system with recursive feature elimination by using random forest and deep learning classifier. In 2018 international congress on big data, deep learning and fighting cyber terrorism (IBIGDELFT), pages 71–76. IEEE, 2018. doi : 10.1109/ibigdelft.2018.8625318.

[45] intel, "Anomaly detection."

[46] Yu Yao, Yang Wei, Fu-Xiang Gao, and Yu Ge. Anomaly intrusion detection approach using hybrid mlp/cnn neural network. In Sixth international conference on intelligent systems design and applications, volume 2, pages 1095–1102. IEEE, 2006. doi : 10.1109/ isda.2006.253765.

[47] Felipe de Almeida Florencio, Edward David Moreno, Hendrik Teixeira Macedo, Ricardo JP de Britto Salgueiro, Filipe Barreto do Nascimento, and Flavio Arthur Oliveira Santos. Intrusion detection via mlp neural network using an arduino

embedded system. In 2018 VIII Brazilian Symposium on Computing Systems Engineering (SBESC), pages 190 195. IEEE, 2018. doi : 10.1109/sbesc.2018.00036.

[48] A Arul Anitha and L Arockiam. Annids : artificial neural network based intrusion detection system for internet of things. Int. J. Innov. Technol. Explor. Eng. Regul,(2019) :8, 2019.

[49] F. Pedregosa, G. Varoquaux, A. Gramfort, V. Michel, B. Thirion, O. Grisel, M. Blondel, P. Prettenhofer, R. Weiss, V. Dubourg, J. Vanderplas, A. Passos, and D. Cournapeau,"Scikit-learn: Machine Learning in Python," MA- CHINE LEARNING IN PYTHON

[50] IONOS, L. éditoriale. (2019, June 3). Fog computing : Approche Décentralisée des clouds ido. IONOS Digital Guide. https://www.ionos.fr/digitalguide/serveur/know-how/fog-computing/

[51] Felipe de Almeida Florencio, Edward David Moreno, Hendrik Teixeira Macedo, Ricardo JP de Britto Salgueiro, Filipe Barreto do Nascimento, and Flavio Arthur Oliveira Santos. Intrusion detection via mlp neural network using an arduino embedded system. In 2018 VIII Brazilian Symposium on Computing Systems Engineering (SBESC), pages 190–195. IEEE, 2018. doi : 10.1109/sbesc.2018.00036.

Résumé

L'Internet des objets (IoT) révolutionne notre monde. Son concept principal est de connecter des objets physiques à Internet. Cependant, avec l'augmentation du nombre d'appareils connectés et la croissance de l'IoT, de nouvelles menaces à la cybersécurité émergent en raison des vulnérabilités présentes dans ces appareils. Les botnets malveillants sont une menace courante, utilisant des appareils IoT vulnérables pour mener des attaques informatiques. Face à ces menaces, il est nécessaire de développer de nouvelles méthodes pour détecter les botnets IoT.
Dans cette étude, nous proposons de mettre en place un système de détection d'intrusion spécifiquement conçu pour l'Internet des Objets, en utilisant l'apprentissage automatique pour identifier le trafic réseau malveillant des botnets. Notre modèle est basé sur la classification. Pour analyser le comportement des appareils sur le réseau, nous avons utilisé l'ensemble de données d'intrusion sur le réseau Iot.

Mots clés : Apprentissage automatique, Détection d'anomalies, Internet des objets, système de détection d'intrusion, IDS.

Abstract

The Internet of Things (IoT) is revolutionizing our world. Its main concept is to connect physical objects to the Internet. However, with the increase in the number of connected devices and the growth of IoT, new cybersecurity threats are emerging due to the vulnerabilities present in these devices. Malicious botnets are a common threat, using vulnerable IoT devices to carry out computer attacks. Faced with these threats, it is necessary to develop new methods to detect IoT botnets.

In this study, we propose to implement an intrusion detection system specifically designed for the Internet of Things, using machine learning to identify malicious network traffic from botnets. Our model is based on classification. To analyze the behavior of devices on the network, we used the iot network intrusion dataset.

Keywords: Machine Learning, Anomaly Detection, Internet of Things, Intrusion Detection System, IDS.

ملخّص

أحدثت إنترنت الأشياء (IoT) ثورة في عالمنا. مفهومها الرئيسي هو توصيل الأشياء المادية بالإنترنت. ومع ذلك ، مع الزيادة في عدد الأجهزة المتصلة ونمو إنترنت الأشياء ، تظهر تهديدات جديدة للأمن السيبراني بسبب نقاط الضعف الموجودة في هذه الأجهزة. تمثل شبكات الروبوت الخبيثة تهديدًا شائعًا ، حيث تستخدم أجهزة إنترنت الأشياء الضعيفة لتنفيذ هجمات على الكمبيوتر. في مواجهة هذه التهديدات ، من الضروري تطوير طرق جديدة لاكتشاف شبكات إنترنت الأشياء.

في هذه الدراسة ، نقترح تنفيذ نظام كشف التسلل مصمم خصيصًا لإنترنت الأشياء ، باستخدام التعلم الآلي لتحديد حركة مرور الشبكة الضارة من شبكات الروبوتات. يعتمد نموذجنا على التصنيف. لتحليل سلوك الأجهزة على الشبكة ، استخدمنا مجموعة بيانات اختراق شبكة Iot.

الكلمات المفتاحية: تعلم الآلة ، كشف الشذوذ ، إنترنت الأشياء ، نظام كشف التسلل ، IDS