POLYTECHNIQUE MONTRÉAL

affiliée à l'Université de Montréal

Explainable IDS for DoS attacks

JULIEN GILBERT

Département de génie informatique et génie logiciel

Mémoire présenté en vue de l'obtention du diplôme de *Maîtrise ès sciences appliquées*

Génie informatique

Décembre 2020

POLYTECHNIQUE MONTRÉAL
affiliée à l'Université de Montréal

Ce mémoire intitulé :

Explainable IDS for DoS attacks

présenté par **Julien GILBERT**
en vue de l'obtention du diplôme de *Maîtrise ès sciences appliquées*
a été dûment accepté par le jury d'examen constitué de :

José FERNANDEZ, président
Gabriela NICOLESCU, membre et directrice de recherche
Alejandro QUINTERO, membre

DEDICATION

I dedicate my work to my family and friends,
thank you for supporting me

ACKNOWLEDGEMENTS

I warmly thank my research director Prof. Gabriela Nicolescu for her support throughout my research. This research would not have been possible without the help of the company Humanitas Solutions. I would like to thank the whole team for their support and especially Luca Gianoli who helped me with regular follow-up. The advice provided by Thales also helped me to improve my research. I would also like to warmly thank my laboratory colleagues for their friendship and advice. Finally, I would like to express my sincere thanks to my family and friends who gave me their moral support in the accomplishment of my research project.

RÉSUMÉ

L'internet des objets (Internet of Things, IoT) est un secteur d'activité en plein développement. Cette technologie va permettre de faire communiquer entre eux différents appareils qui pourront alors échanger un nombre important de données. Sécuriser les informations transmises est un requis important de l'IoT. Des mécanismes de sécurité utilisés dans les réseaux actuels peuvent être repris (chiffrement, authentification, etc). Néanmoins, l'augmentation de la surface d'attaque nécessite de développer de nouveaux outils afin d'améliorer la sécurité de ce type de réseau.
Le mécanisme étudié dans cette étude est le système de détection d'intrusions (Intrusion Detection System, IDS). Les systèmes de détection d'intrusions analysent un ensemble de données afin de détecter de potentielles intrusions. Le développement de l'apprentissage automatique a permis d'augmenter les performances de ces algorithmes. Néanmoins, les algorithmes d'apprentissage automatique sont souvent très difficilement interprétables par un humain. Des méthodes, nommées Explainable Artificial Intelligence (XAI), ont été développées pour permettre une meilleure interprétation des résultats. La revue de littérature a montré que plusieurs méthodes pouvaient être utilisées afin de réaliser un système de détection. Les contraintes des objets connectés nous ont orientés vers une approche de détection d'anomalie à l'aide de l'analyse de paquets réseau. L'étude de la littérature a mis en avant l'algorithme Suport Vector Machine dans la détection des intrusions et la méthode Partial Dependence Plot (PDP) pour l'interprétation des résultats. Nous proposons une approche combinant ces deux algorithmes dans l'objectif d'obtenir un système de détection d'intrusions performant et ayant une meilleure interprétabilité.
Le mécanisme obtenu a fait l'objet de 3 expériences: une analyse des erreurs de l'algorithme de détection à l'aide de la méthode PDP, une confrontation à un algorithme attaquant l'IDS et une implémentation dans un simulateur ad hoc. Pour la première étape, notre système a été testé sur le dataset NSL-KDD, qui est un dataset classiquement utilisé pour la détection d'anomalies dans le domaine de la cybersécurité. À la suite de cette expérience, nous avons découvert qu'une majorité des faux positifs étaient dus à une seule feature. Le système permet également de donner un niveau de confiance à l'utilisateur, car l'algorithme explicatif permet de savoir si une anomalie détectée est probablement un vrai positif, ou probablement un faux positif. Le même type d'information peut être montré à l'utilisateur dans le cas des vrais négatifs et des faux négatifs. Nous avons également souhaité confronter notre mécanisme de sécurité à une attaque adversarial. Le principe de ce type d'attaque est de modifier légèrement des attaques pour qu'elles soient détectées comme des données normales

par l'algorithme de détection. Les attaques ont été créées en ajoutant du bruit dans les données normales. Une expérience a également été effectuée sur le simulateur de réseau de notre partenaire industriel. Cet outil permet de simuler un réseau ad hoc en prenant en compte les capacités de calculs et de mémoires de chaque noeud du réseau. Il offre également un déploiement rapide de différentes configurations du réseau. Grâce à cet outil, nous avons testé notre solution dans des configurations de réseaux différents. Cela a permis d'étudier le comportement de l'IDS avec un nombre différent de noeuds et d'attaquants. Dans ce contexte, des attaques de type Denial-of-service (DoS) ont été créées afin de mesurer les performances de détection.
Les résultats ont montré que l'approche proposée fournit une capacité supplémentaire à l'utilisateur en ayant une meilleure compréhension des décisions de l'algorithme de détection. De plus, nous avons montré que ce mécanisme peut être utilisé dans différents cas d'étude.

ABSTRACT

The Internet of Things (IoT) is a rapidly developing sector of activity. This technology will enable different devices to communicate with each other and exchange a large amount of data. Securing the information transmitted is an important requirement of the IoT. Security mechanisms used in current networks can be used (encryption, authentication, etc.). Nevertheless, the increase of the attack surface requires the development of new tools to improve the security of this type of network.
The mechanism studied in this study is the Intrusion Detection System (IDS). Intrusion detection systems analyse a set of information in order to detect potential intrusions. The development of automatic learning has made it possible to increase the performance of these algorithms. Nevertheless, machine learning algorithms are often very difficult for a human to interpret. Methods, called Explainable Artificial Intelligence (XAI), have been developed to allow a better interpretation of the results. The literature review showed that several methods could be used to build a detection system. The constraints of the connected objects led us to an anomaly detection approach using network packet analysis. The literature review highlighted the Support Vector Machine algorithm in intrusion detection and the Partial Dependence Plot (PDP) method for the interpretation of the results. We propose an approach combining these two algorithms with the objective of obtaining a high-performance intrusion detection system with better interpretability.
The resulting mechanism has been the subject of 3 experiments: an analysis of the errors in the detection algorithm using the PDP method, a comparison with an algorithm attacking the IDS and an implementation in a network simulator. For the first step, our system was tested on the NSL-KDD dataset, which is a dataset classically used for the detection of anomalies in the field of cybersecurity. The study performed on the NSL-KDD with an SVM algorithm resulted in an accuracy of 97.76%. As a result of this experiment, we discovered that a majority of false positives were due to a single feature. The system also provides a level of confidence to the user, as the algorithm allows to know if a detected anomaly is probably a true positive, or probably a false positive. The same type of information can be shown to the user in the case of true negatives and false negatives. We also wanted to confront our security mechanism with an adversary attack. The principle of this type of attack is to slightly modify attacks so that they are detected as normal data by the detection algorithm. The attacks were created by adding noise to the normal data. An experiment was also carried out on our industrial partner's network simulator. This tool allows to simulate an ad hoc network by taking into account the computation and memory capacities of each node of the

network. It also offers rapid deployment of different network configurations. Thanks to this tool, we have tested our solution in different network configurations. This allowed us to study the behaviour of the IDS with a different number of nodes and attackers. In this context, Denial-of-service (DoS) attacks were created in order to measure detection performance.
The results showed that the proposed approach provides an additional capability to the user by having a better understanding of the decisions of the detection algorithm. Moreover, we have shown that this mechanism can be used in different case studies.

TABLE OF CONTENTS

LIST OF TABLES

LIST OF FIGURES

LIST OF SYMBOLS AND ACRONYMS

IDS	Intrusion Detection System
HIDS	Host Intrusion Detection System
NIDS	Network Intrusion Detection System
DoS	Denial-of-service attack
IoT	Internet of Things
IoD	Internet of Drones
UAV	Unmanned aerial vehicle
MANET	Mobile ad hoc networks
SVM	Support Vector Machine
OCSVM	One Class Support Vector Machine
LSTM	Long short-term memory
HTM	Hierarchical Temporal Memory

CHAPTER 1 INTRODUCTION

In this chapter, we introduce the global context of the research project and the security solutions for intrusion detection in ad hoc networks. Then, we describe our objectives and the proposed contributions.

1.1 Context

The recent development of unmanned aerial vehicles (UAV) has provided access to new functionalities in many areas. At the same time, the appearance of connected objects in recent years has also made it possible to develop communication technologies that complement the functionalities offered by UAV. The studies increasing the performance of these objects as well as the new means of communication bring significant improvements in several sectors. This technology offers interesting opportunities in the field of emergency response to natural disasters. The communication between a fleet of UAV and ground rescuers enables various tasks such as locating victims or mapping rugged terrain. In the first moments of a humanitarian intervention, the rescuers need to communicate between them in order to obtain the best possible organization. A solution to achieve this purpose is to deploy a communication network via a UAV fleet. Thanks to the drone capabilities, the network coverage area is increased and the quantity of exchange information can be improved. Moreover the altitude reached allows to map the ground, lead the first aiders and detect the victims. Although it is a great help during the mission, the sensitivity of the information requires protecting the communication against potential attackers. An intrusion detection system (IDS) is a classification algorithm of the network packets into normal traffic or irregular behaviour. This mechanism is the second layer of security after the classic methods like authentification or encryption. The IDS are common in other fields (for instance, the banking actors use them to detect transaction frauds).
The rule-based algorithms detect an anomaly when one of the rules present in its database is not respected. Machine learning based anomaly detection algorithms detect an anomaly when a data deviates from the model they have learned during their training phase. Nowadays, intrusion detection algorithms are based on the signature of the traffic. The most popular are Snort [1] and Bro [2]. The principle is to detect a pattern in the data which belongs to a database of fraudulent patterns. These solutions have proven their utility during the years, but they suffer against zero-day attacks. Indeed, as long as an attack has not been analyzed and recorded in the database, the signature-based algorithm cannot classify this

attack as an anomaly. To counteract this weakness, studies show the interest in a machine learning approach.
During the recent years the field of machine learning greatly improved and it has found itself in a lot of areas. This paradigm is also applied in the context of cyber security. In general, when machine learning is applied to cybersecurity, a model is trained with a set of data composed by normal and anomalous data and the algorithm acquires the ability to separate the new data into several categories. Nevertheless they are not infallible. For instance, some data, named false positives, are classified as anomalies while it is in reality normal data. Therefore, in case of a detected attack, the user must have the capability to rapidly know the causes of this decision in order to get the possibility to react as fast as possible. In order to make the right decision at the right time, it is necessary for the detection system to provide as much information as possible on why the algorithm made the decision. To provide these additional information, research has to be conducted to apply the field of explainable Artificial Intelligence (AI) to an anomaly detection system [3].

1.2 Research Challenges

Several challenges have to be faced to obtain an explainable machine learning algorithm implemented in a real system. The first challenge is given by the interpretability of the outputs. The main issue is to adapt the existing methods of interpretation in the context of cyber security. The difficulty lies in the complexity of the features and their relation between them. That makes the validation process complex because the appreciation of the explanation is more qualitative rather than quantitative. Eventually the inherent constraints of embedded systems have to be taken account to implement the IDS into the whole system. The classic algorithms of explainable AI are not designed to be used in an embedded system so an optimization is necessary.

This master thesis aims to provide answers to the following research question :

How an embedded intrusion detection system based on a machine learning algorithm can explain accurately its decisions for DoS attacks ?

1.3 Objectives

The main objective of the thesis is to implement an embedded intrusion detection system based on an explainable machine learning algorithm.
The proposed solution has to respect the embedded constraints and provide the result as well as the reasons behind this prediction. The drone characteristics bring many challenges for the

implementation of an algorithm to respect the constraints in terms of power consumption. Another performance specification is the execution time of the detection system. Indeed, the algorithm needs to check the validity of the packet before the arrival of the next one. The satisfaction of this property must be validated for the maximum packet size. Some previous works developed IDS, based on a machine learning algorithm to detect the novel attacks. The research produced algorithms with good accuracy, but the literature does not study the problem of explaining ability of the machine learning outputs. The papers which developed anomaly detection system are focused on the improvement in terms of temporal performance, power consumption or accuracy. Nevertheless the obtained model is considered as black box systems and the reason behind the decision of the algorithm are not examined.

In order to define and develop a new IDS based on machine learning and explaining the classification output, we define the following specific objectives :

- Analyse the possible attacks on an ad hoc network
- Design an algorithm which explains the decision of the IDS
- Validate the software in a network simulator

1.4 Contribution

The previous research studies in the field have shown very interesting results, but to the best of our knowledge, there is no contribution implementing an explainable IDS in ad hoc networks. This is a key requirement in order to secure the network and increase the confidence into the machine learning algorithms that detect the anomalies. The research issue focuses on the trust increase of an IDS implemented in an embedded system. The implemented algorithm must obtain good detection scores while explaining the reasons for these choices. Our contributions are:

- application of an explainable AI method on an intrusion detection algorithm in order to provide a method to analyze the detection results of the machine learning algorithm. An analysis of false positives and false negatives is also performed to understand these errors.
- analysis of the system behaviour against an attacking algorithm: a confrontation between a GAN attacking algorithm and an anomaly detection algorithm is achieved in order to prepare for attacks of this kind against intrusion detection systems.

- An implementation of our defense system in an ad hoc network simulator is carried out in order to analyse the behaviour of our IDS in a simulated environment.

CHAPTER 2 LITERATURE REVIEW

2.1 Security in the internet of things

The Internet of Things (IoT) sector offers important developments in different sectors: domotic, automotive, Internet of Drone for example. Chiang [4] shows the technological opportunity provided by Fog architectures, method in which data management and operations are carried out as close as possible to the sensors, for the field of connected objects. However, the strong growth in this area is taking place at the expense of aspects that are essential for the proper development of the society. Louchez and Thomas [5] show the ecological impact of these new components and Waisel [6] points out the significant increase in connectivity between all digital components greatly increases the attack surface. The strong pressure on companies to commercialize their systems as quickly as possible in order not to miss the market window is causing a dramatic underinvestment in their security. Therefore, the damage caused can be catastrophic because of its use and computing capacity. The Mirai attack in September 2016 [7] shows the early stages of the potential of a set of connected objects accessed by malicious individuals. Kolas [8] is conducting a case study on the Mirai botnet, and similar products, with the objective of characterizing the attack and pointing out problems related to the attacked components. With the same objective of understanding the risks associated with this new technology, several studies have been conducted to increase the understanding of security vulnerabilities. Sikder [9] focuses on the field of sensors. These components allow the extraction of a lot of information on a system, such as acceleration, temperature or position, thus allowing the user to increase his control over the object under study. However, the researcher highlights the attacks that these components can undergo and the impact that they can have. This subject of security is not only for researchers and must also be discussed at the legislative level. The new European regulations on the protection of personal data ensured a new privacy rights for the citizen [10].

2.1.1 Security in Internet of Drones (IoD)

The technology of the embedded systems allows the deployment of a new type of network: the Internet of Drones (IoD). It refers to the communications systems that enable the exchange of information and ensure the operation of a UAV fleet. This will be possible thanks to the improvement in terms of energy consumption and time performance. Nevertheless the increase of the number of communication on this small device, compared to a classic computer, increases the security risk. The IoD requires security mechanisms on all the layers and not only at the application level like it is the case nowadays [11]. Indeed the potential attacks can also affect the routing layer or the physical layer. In their article, the authors present the global architecture of the network and the challenges that the community needs to face before an implementation in the real life.

The review of Motlagh [12] points out the challenges that companies will have to face. The requirements inherent in the use of collaborative unmanned aerial vehicles (UAV) increase the need to communication. Consequently the attack area is larger for a malicious intruder. Moreover, the scientist highlights the need to reduce the consumption to increase the autonomy. This makes security implementation more difficult.

2.1.2 Potential threats on connected objects

In [13] the authors examine the security of the intelligent road. The application of the DREAD model (Damage potential, Reproductibility, Exploitability, Affected users, Discoverabilty) makes understandable the actual risks in this intelligent road infrastructure. First a comprehensive list of different attacks is made, then each attack is studied under the scope of the DREAD model. On that basis, the scientists could show that more than half of these attacks can be qualified as highly risky. This analysis is limited to the well-known attacks and says nothing about any new attacks we may have in the future.

Koslov [14] also mentioned the role of legislation on the management of personal data. It shows that a complementary approach must be taken between scientific research and legislative debate to increase the security of Internet of Things (IoT), the network of connected devices and their applications.

In [15], Conti et al. points out the major challenge to solve in order to develop a trustworthy IoT network. Their analysis focuses on forensic, a method to discover the intruder and the exploited vulnerabilities, in an IoT network. This technique is very important to improve continuously the security of a system.

The new development of the IoT allowed the opening of many opportunities. A large number of companies design their own system to a precise application. In this field, Gubbi et al. [16] address the concern related to the lack of standard for protocols and communication of all these connected objets with each other. To reduce this gap, the authors propose a new method based on cloud technology to create a smart environment. They also underline the future directions for this technology.

In [17], Sciari et al. also propose their analysis of the future security challenge focusing on the different mechanisms to ensure the protection of the nodes and the data. They discuss about the need to implement the classic methods of security, like authentification or confidentiality, into an IoT network where bandwidth and energy constraints need to be considered.

In [18], Sfar et al. discuss the security challenge in IoT in order to have clarified the actual and future challenges the engineers must solve before the massive use of this technology in the real world. The notions of Privacy and Trust have to be considered deeply to protect the data of the users. The authors discuss a case study of the smart manufacturing. The central role of sensors in this architecture increase the number of threats on the installation.

2.1.3 Known attacks

In [19], a study of power analysis attacks is lead in the context of sensor networks in the Smart Cities. The communication network we are working on in this study is based on a fleet of UAV and a set of ground sensors. So it's useful to see what is being done in the Smart Cities field. The objective of these connected cities is to collect various data in order to optimize the city's resources. The use of this type of attack can provide useful information to an intruder in order to break the encryption of a message and consequently the whole data security of the networks. The vulnerability of the connected objects that are easily accessible against these attacks are very high and countermeasures are required. According to this article, some techniques like masking are very efficient.

In order to improve the IoT security, the first requirement is to understand the threat of this type of network. In [20], Babar et al. analyze the structure and the constraint of an IoT network and propose a classification of threat and mechanism to respond to these threats. Their aim is to underline the different challenges to solve before a safe implementation of the IoT technology. For this purpose, they discuss about a security model proposed by their team to have a good understanding of the security in this context.

2.2 Intrusion Detection System (IDS)

2.2.1 Existing IDS

In order to improve the security of traditional networks, Martin Roesch [1] developed the SNORT software in 1998. In his article, the author presents the characteristics of the IDS and the basic principles, which has become over time one of the most widely used in the field. It describes in particular its advantages such as its flexibility and lightness for a classic environment.

In [21], the authors develop SVELTE, a real-time IDS that overcomes the constraint inherent to the IoT components. Their implementation in Contiki OS allows the system to be efficient against the attack on the IoT protocol 6LoWPAN. Their evaluation is based on simulated scenarios and they obtain very good performances in terms of detection and overhead. In the paper, the researchers claim that their IDS was tested on routing attacks but it can also detect other types of attacks.

In [22] the authors develop ORUNADA (Online and Real-time Unsupervised Network Anomaly Detection Algorithm). The purpose of their software is to meet the requirement of the application which manages a large number of data and needs a very fast detection. The result obtained seems to outperform the literature in this context. The reactivity of IDS is often an essential problem in order to react as quickly as possible. It provides a sliding window mechanism to improve the system's time performance. The use of an incremental algorithm such as Density Grid-based Clustering allows good classification results to be obtained, while being adapted to a real-time context.

2.2.2 Embedded IDS

In [23], the authors analyse the current validation strategy realised before the release of the IoT system. They also discuss about the different ways to implement security and the weakness of the technology nowadays.

In [24], the authors highlight the potential of machine learning for the edge security in IoT. Their main contribution is to provide a survey for this area. They discuss challenges like the lack of an IoT benchmark. Despite these, the use of machine learning seems to offer the better result in the context of IoT because of the dynamic properties of the network.

The fast development in the recent years of ad hoc networks brings some new security issues. The research community tries to improve these systems with mechanisms like the IDS specialised in ad hoc networks. In [25], the authors present the main IDS architectures. Indeed, an ad hoc IDS can revert to several forms: stand-alone, distributed, hierarchical or with the use of mobile agents.

According to [26], a rapid response capability is the key to save as many lives as possible during a natural disaster crisis. In this context, the autonomy as well as the speed of deployment and human detection are key points for an effective intervention. The main objective of their project is to provide the maximum information to the chain of command in terms of geographical data, position and behaviour of victims. These requirements implies some risks in terms of safety. The researchers point out two major risks. First a malicious person can take control of a part of the drone and lead to a loss of control, potentially even a crash. The second risk is related to the data on the air. These data can be altered or stolen for the attacker's benefit. To overcome these security challenges, an architecture to manage security is implemented and plausibility checks are performed on the sensor to ensure its nominal operation. The authors emphasis also the necessity to consider the safety since the beginning of the project.

In [27], the authors underline the importance of implementing a second line of security into the network. The layer has to be implemented after the classic mechanism like encryption or authentification. His goal is to analyze the network behaviour in order to detect the potential intrusion in the system.

Avionics is a highly critical sector where the computer systems have taken up more and more space. Nowadays a challenge is to integrate all the systems of an airplane into an Integrated Modular Avionics (IMA) to reduce the weight of the plane and improve the performance of the whole system. In the same time, the number of communication between the different devices of the ecosystem is increasing in order to reduce the cost. Unfortunately, it is also increasing the risk of security because an intruder, or a defective component, can cause serious damage. In [28], a Host Intrusion Detection System (HIDS), that monitor the state of the host node, is proposed to add a new layer of security in the system. The constraints mentioned by the authors are the need to non interfere with the existing system in terms of performance and to respect the classical constraint on embedded systems: memory, temporal performance and consumption. Moreover the IDS need to pass through various sets of tests to be implemented in a real aircraft.

Time performance and power consumption characteristics are essential in the aviation sector, but they are also essential in many fields such as the automotive industry, drones and connected objects.

2.2.3 Edge computing

The benefits

Thanks to the development of the technology in terms of computation power, computation is brought closer to the environment. This is called edge computing. The advantages of this technique are the reduction of packet in the network and the gain in terms of privacy because the sensitive information stay near the environment. In [29], Raponi et al. review the existing solutions in this context and present the future challenges.

Embedded systems

In [30], Shams and Rizaner explain the mechanism of an IDS in the context of ad hoc networks based on an improving SVM method. Their work is focused on the discovery of DoS attack and the reduction of delay to fight against the constraint of embedded systems in the ad hoc network.

A research challenge is to detect the anomaly in real time to act as quickly as possible. In [31], Ahmad et al. propose a mechanism to achieve this goal with streaming data. The use of the technique named Hierarchical Temporal Memory (HTM) allows the system to compute streaming data with an online algorithm. To test their proposal the researchers provide an open source benchmark, Numenta Anomaly Benchmark (NAB), which allows to validate a new real-time IDS wih labeled data in a streaming context. Their two contributions facilitate the development of this kind of technology. They claim that the increase of the streaming data in the IoT context will force the improvement of the machine learning algorithms. Consequently a benchmark is needed to validate the behaviour of the anomaly detection system.

An attack on an embedded system can modify the behaviour of the processor. Therefore, the modelisation during normal operation of the processor can be used to detect anomalies during an attack. In [32], Lu, Seo and Lysecky model the system based on the runtime system. The results show a low rate of false positive.

In [33], the authors present a hardware accelerator to overcome the issue of low power and low latency in the context of domotics. Their NIDS is based on a simple one layer neural network and was tested on an FPGA to validate the temporal performance of the system.

The embedded systems are subject to constraints like performance and power consumption. In [34] the notion of memory constraints is discussed. The proposal is to focus on the optimization of the memory to use less than 4MB of memory while providing a good detection rate for an intrusion detection system application.

2.2.4 Machine learning for IDS

In [35] the authors classify the anomaly detection techniques in four categories: classification, statistical, clustering and information theory. On that basis, they analyze their advantages and drawbacks. Their study is focused on the classic method but they point out that collaborative IDS is a very interesting area in the future of research.

Despite its strong capacity in the detection of probing and Denial-of-Service (DoS) attacks, the major drawbacks of machine learning in the context of anomaly detection is given by the high false positive rate. On the contrary, the specification-based algorithm suffers from a low detection rate but outperform the machine learning regarding the false positives. So in [36] the idea of the authors is to combine these two perspectives. Their result shows a net decrease in the rate of false positives compared to the approaches based only on machine learning algorithms.

In [37], Tsai et al. provide a review of machine learning adopted in the context of intrusion detection, between 2000 and 2007. Their main conclusion is the need to develop multiple classifiers to improve the accuracy of the actual methods.

In [38], Chaabouni et al. propose a comprehensive review of the use of machine learning in the context of IoT to detect network intrusion. They consider in particular the different challenges of the IoT security which are the high connectivity between the devices and threats inherent to this like DoS attack, information leakage, etc. Their conclusion approves the use of machine learning for a Network Intrusion Detection System (NIDS), an IDS that monitors network data, in an IoT network. However it underlines some research challenges like the improving in terms of computation performance to be implemented in an edge and FOG environment. Another big challenge of this field is the need to design methodology to validate the security mechanisms and to increase the trust into the IoT network.

In [39], the authors explain their design of an incremental decision tree as an online learning method. Decision tree is a rapid and accurate machine learning algorithm for anomaly detection but it suffers from overfitting. The incremental aspect of this technique allows to build a tree that improves with new data.

In [40], a mechanism to detect anomalies in a network is presented. The neural network proposed represents the groundwork to more sophisticated and efficient neural networks.

During the recent years, many machine learning and rule-based algorithms were developed to detect intrusions in a network. A large part of them were tested on the KDD99 dataset and show very different result. In [41], Sabhnami et al. analyze a set of these algorithms and compare their effectiveness on the KDD99. The tested algorithms are Multilayer Perceptron, Gaussian classifier, K-means clustering, Nearest cluster, Incremental radial basis function, Leader algorithm, Hypersphere algorithm, Fuzzy ARTMAP and a Decision Tree. Their conclusion shows the best solution for probing attack is the Multilayer Perceptron, while K-means is the most appropriate for *DoS* and *U2R*, and Gaussian classifier is the most efficient for *R2L* attacks. As a consequence, the authors combine these best algorithms to pool together the advantage of each. The limitation of this study is the lack of the Support Vector Machine (SVM) algorithm, which seems to be very efficient for the intrusion detection problem.

Anomaly detection has been implemented in the aerospace field [42], which is very constrained in terms of performance, by the use of a Gradient Tree Boosting. This choice was made after tests on different machine learning algorithms: logistic regression, naive Bayes, support vector machines and gradient tree boosting. The algorithm was tested in a simulated environment and confronted with anomalies automatically generated by Fault Tree Analysis programs as well as a Failure Mode and Effect Analysis.

Boser et al. [43] propose an algorithm to separate a set of data with a hyperplane. This method is based on a learning approach where the algorithm is trained to optimize the separation between the data. The purpose of the algorithm is to maximise the distance between the data and the separator. This method will later be called: Support Vector Machine (SVM). In order to test their solution, they used a set of handwritten numbers and they obtained very encouraging result.

In [44], Manevitz and Yousef apply the one-class SVM algorithm to a problem of classification in the context of text retrieval in documents.

In [45], Li et al. propose a method to improve the one-class SVM in the context of security. The method is an IDS based on anomaly detection technique. The test was realised on the KDD99 dataset.

In [46], the authors study the threats inherent to the use of sensor networks in an industrial network and propose a mechanism to detect an intrusion. Their proof of concept is based on a k Nearest Neighbors (kNN) algorithm to classify the packets. They experiment their system thanks to a simulation of the sensors. Their solution suffers of major issues concerning the trust that the user can put into the security but they show this system can obtain good results.

In the classic methodology, the developer has to set the hyperparameter of the One Class Support Vector Machine (OCSVM) by himself, or use methods which cannot always be relevant because of the lack of label data. To overcome this difficulty, Ghafoori et al. [47] developed a mechanism to estimate these parameters to facilitate the work of the developer.

In a large majority of anomaly detection problems, the number of normal data is much greater than the number of anomalies. This characteristic is named unbalanced dataset and it is the problem of many classification algorithms. To overcome this issue, Xi et al. [48] propose a new algorithm based on Least squares SVM in the context of fault detection in an aircraft engine. This algorithm is also real time to respond as quickly as possible to the anomaly detected.

In [49], Hodo et al. demonstrate the use of multi-layer perceptron for anomaly detection.

Their work focus on the DoS attack and their algorithm obtain an accuracy of 99,4% on this type of attack.

Many open source libraries are available for machine learning applications. Concerning the SVM algorithms, libSVM [50] is a C++ implementation of a large number of methods. It allows to solve classification and regression problems.This library is released under the BSD license. Some Python librairies like Scikit learn use libSVM to improve their performance in the classification problem.

The monitoring of a system is also a possibility offered by the anomaly detection. Indeed, the huge amounts of data provided by some systems cannot be interpreted only by a human engineer. In [51], Hundman applies the Long short-term memory (LSTM) algorithm to detect anomalies in the data send by a spacecraft. They realised their tests with the data of a satellite and a rover. Their algorithm could handle the data, using techniques to reduce the false positives, like pruning anomalies or learning from history, the authors obtained good result.

The machine learning algorithms have shown their effectiveness in the context of intrusion detection but they are highly data dependent. Unfortunately, the real-world data are not always available during the development. That's why some researcher developed dataset in order to help the community. In [52], the authors present UNSW-NB15, a dataset oriented for intrusion detection. They discuss the interest of this set and compare it to KDD99. Their dataset is more difficult for the detection and provide more attacks than the KDD99 and NSL-KDD dataset. Indeed the detection rates achieve by the actual intrusion detection systems on the UNSW-NB15 are lower than on the KDD99 dataset.

In [53], Jindal et al. apply the concept of anomaly detection in a smart grid system. The role of their mechanism is to detect and locate the frauds realised by the users in terms of energy consumption. They combine two machine learning algorithms: a Decision Tree and a SVM. The Decision Tree computes an expected electricity consumption based on 5 features (the number of appliances, the number of persons, the temperature, the season and the time slot). The SVM classifies a sample thanks to these 5 features, the value at the output of the decision tree and the actual electricity consumption. The authors realized two tests with the SVM. First they trained the SVM alone with the 5 basic features and test on a new dataset. Then they add the decision tree into the mechanism. The first experiment gives an accuracy of 87,5% whereas the second give an accuracy of 92,5%. The authors state that their combination can be used in a real-world infrastructure because the accuracy is suitable for a smart grid application.

In [54], the authors develop a method to detect anomalies thanks to the analysis of system calls.

2.2.5 IDS for drone

One sector that requires a high level of security with regard to their criticality is the UAV, Unmanned Aerial Vehicle. In [55], Lin describes the problems posed by the lack of awareness of security in IT systems within drones. Indeed, the problem of physical constraints, in terms of computing time or bandwidth, causes implementation challenges that need to be addressed.

2.2.6 Methods to bypass IDS

Several techniques to evade signature based IDS were proposed in the past. In [56], the authors list some techniques and bring a new type of attack: the shadow attacks which evade the IDS based on system call visualization.

In [57], Fred Cohen evokes some techniques to evade an IDS. The author discusses the weaknesses of IDS technology and shows that it is possible to find techniques to overcome this security mechanism.

The use of machine learning algorithms is not without risk: IDSs have to face new evasion techniques that have the role of bypassing this type of security. Lin [55] shows the possibility of designing an IDS attack mechanism based on adversarial attacks. The objective is to model the behaviour of the attacked IDS and then generate attacks detected as normal. This study shows that an IDS should not be the first layer of security but that it should be a solution in addition to the security mechanisms. In summary, this is a necessary but not sufficient solution.

A common attack against the machine learning algorithm is the adversarial example. Indeed as [58] shows it is possible to trick the algorithms embedded in a car thanks to stickers on traffic signs. In [55], the authors show this type of attack is also possible against IDS. To solve this important issue, Lecuyer et al. [59] used the concept of differential privacy to obtain a robust system against these attacks. This method was initially used to guarantee the confidentiality of an individual data in a database by introducing a probabilistic mechanism. The simulation on the ImageNet dataset shows the robustness of this mechanism.

In [60], Bruckner et al. proposed the use of the game theory to improve the effectiveness of the classic IDS based on machine learning. They develop two algorithms, Nash logistic regression and Nash support vector machine.

Attacks that aim to evade IDS by creating intrusions that looks like normal behavior are also called mimicry attacks. Kayacik et al. [61] studied this type of methods by analyzing buffer overflow attacks. They showed that by modifying the behavior of an attack it was possible to carry out an attack without being detected. To counter this type of attack, solutions have been studied. Tapiador et al. [62] proposed a statistical approach to detect mimicry attacks. Nevertheless, their study is limited to a few detection algorithms and it underlines the importance of extending their work to other systems, especially SVMs.

2.2.7 False positives

A major challenge from the anomaly based algorithms is the high rate of false positives (e.g., normal data classified as an anomaly). To overcome this issue, Goeschel [63] proposes a technique based on the combination of three machine learning algorithms: SVM, decision tree and Naive Bayes.

In the same way, Zohrevand and Glasser [64] focus their reflection on the existing mechanisms to reduce the number of alarm caused by false positives. They underline the main role of the context and the type of the false positives to choose the good method to reduce these alarms.

2.2.8 Collaborative IDS

The limits encountered by the individual IDS are the lack of communication between them. Recent attacks show the disability of these types of IDS to detect distributed intrusions. In order to solve this issue, collaborative IDSs were proposed. Meng [65] shows the use of the blockchain technology as a potential solution. In order to increase trust into the data sharing, this mechanism is based on cryptographic functions and different protocols like the proof of work, which makes it possible to limit denial-of-service attacks by asking the computing unit wishing to access the service to perform a complex and time-consuming operation.. The main application of research is the improving security in the sharing of information. Nevertheless, the author points out some challenges like the energy which is needed for the computation of the proof.

In [66], Kannadiga and Zulkernine propose a distributed IDS based on mobile agent. The role of these IDS is to enforce the safety in a multi-node network. These agents are entities which can move through the nodes and detect an intruder who makes an attack on several nodes in the same time. The main advantage, observed by the authors, of distributed IDS compared to central IDS is the reduction of the bandwidth usage.

2.2.9 Tracer

The main resource of the detection algorithms is the data. Indeed, they rely on this information to infer classification. In the field of machine learning, there are three major types of learning: supervised, unsupervised and semi-supervised. In a supervised learning, the training algorithm calculates the model thanks to a dataset containing both the parameters that make up the data and a label allowing its identification. In the case of unsupervised training, the data are not identified beforehand. Finally, in the case of semi-supervised training, the collected data are partially labeled. This type of learning can for example be used when the data collected comes from a controlled environment that does not include attacks. In this case, the normal data is collected and provided to the algorithm during the training phase, allowing it to build its model of the normal behavior of the system. Then in the prediction phase, the algorithm detects behavior that differs from this model to detect anomalies. An algorithm belonging to this type of training is the One Class SVM.
Data labeling is an important issue in the development of an anomaly detection algorithm. The method used by researchers at MIT Lincoln Labs [67] to obtain the KDD99 dataset was to design an environment that they mastered in order to retrieve normal data and inject attacks to obtain data corresponding to intrusions.

In the world of IoT data are everywhere, but they are difficult to obtain and manage because of the time-based performance and memory constraint. To limit the overhead, Gassais [68] uses the tool LTTng on a domotic network. This allows to trace the kernel of the IoT object with low latency and train an algorithm on these data.

The use of machine learning algorithms for a context of network intrusion detection leads to the need to collect a huge amount of data in order to test and develop new techniques. In [69], the authors present their method to create traces in a mobile ad hoc networks (MANET). The technique is based on emulation to capture the packet send between the nodes, 30 in their case, in the *libcap* format. Then this packet can be processed by *tcpdump*.

2.3 Explainable AI

2.3.1 The benefits of explanations

Artificial intelligence technologies have achieved high results in recent years, even surpassing humans in some tasks such as GO play [70] or Starcraft 2 [71] which had a reputation for being so complex that the machine would not be able to beat humans for several years. In addition, the increase in computing power of computers as well as the decrease in their size and power consumption allow manufacturers to consider their implementation in critical areas (automotive, aeronautics, computer security, etc.). However, the complexity of current algorithms makes it more difficult for humans to analyze the results obtained. For example, if a complex machine learning algorithm blocks a node in a network without a human being able to understand the computation sequence, that allows it to make this decision, or if no additional information is provided by the algorithm then solving the problem will be very difficult. On the other hand, this type of algorithm is subject to decision errors. Therefore it is necessary to obtain additional information in order to facilitate the development of the algorithm and to verify that they are making the decisions on the training data for the correct reasons, in the other case that could cause wrong decisions when deployed in the real environment.

The DARPA, an agency within the US Department of Defense, addresses the problem of explainable machine learning by supporting research in this area [72]. David Gunning's update document shows the need to develop explanatory mechanisms for certain critical areas such as medicine, finance or security. The challenges put forward concern the provision of additional information while maintaining good performance.

2.3.2 Explainable AI Definitions

The published research on the field of Explainable AI shows great interest. However, this notion suffers from a lack of a clear definition. For this purpose, [73] and [74] give a precise definition of the interpretability problem. The objective of these articles is to characterize the existing algorithms that exist in terms of understanding for a human. A difference is made in [74] between the various systems based on the comprehension of the user. An *opaque system* is defined as a black box where the user sees only the inputs and the outputs. An *interpretable system* is a mechanism where an output can mathematically be explained. Finally, a *comprehensible system* sends its output but also symbols in order to understand the relation between inputs and outputs. In [73], the authors distinguish three levels of *transparency*. A model where a human can redo the prediction rapidly is called *simulatability*. The notion of decomposability stands for the comprehension of each component, for example a node of a decision tree is easily understandable. Finally, the level of algorithmic *transparency* is the full understanding of the algorithm.

2.3.3 Explanation methods

In [75], the authors show the need to give an explanation of the output of AI in critical sectors and also the recent scientific intent to propose different methods to achieve this goal. They also underline the innovation needs in this field in the lack of trust of the actual algorithms.

The concept of explainable is recently defined in [76]. The authors explain the importance of thinking about this subject and underline the lack of a clear and consensual definition in this field. The authors take the example of AlphaGo Zero and point the benefit that the human can derive from information about the strategy of the algorithm. They also assert the necessity to give explanation on the decision in the field of transportation, healthcare, legal, finance and military. Indeed all the decisions taken in this sector can cause serious damage on humans and infrastructure. For this reason it is highly recommended to develop explainable algorithms before their implementation.

Previous research on machine learning focused on finding the best algorithm to classify a set of data. The main metric used for classification is the accuracy. In [77], the authors proposed a new methodology to improve the capability of a machine learning algorithm. In this article, they develop a mechanism to explain the decision. This method allows to reduce the "Black Box" perception that developers have on their system.

Marino [78] develops a mechanism to explain a machine learning algorithm dedicated to the classification of NSL-KDD dataset attacks. Its objective is to obtain an answer to the question: why has a data been misclassified by the algorithm ? An adversarial approach is used to find the minimum modifications required in order to correctly classify this data.

Ribeiro [79] proposes the LIME mechanism to explain the reasons for a classification. In addition, researchers used an SVM to classify data from a dataset containing the religious affiliation of certain individuals. This experience shows the weakness of the dataset used for classification purposes. According to the researchers, it is very difficult to see this issue with data and simple predictions. The provision of additional explanations then makes it possible to provide a more precise analysis.

Wang et al. [80] used the SHAPE method to study the NSL-KDD dataset. They have developed a framework to obtain a better interpretability of the results obtained by an anomaly detection algorithm on the NSL-KDD dataset. Their objectives are to analyze the important features that allowed the IDS to classify a dataset and to highlight the impact of a particular feature on the prediction. Their analysis shows the importance of the feature dst_host_serror_rate in the classification of a dataset as a DoS attack. Their work has shown the interest of applying explainable methods to machine learning algorithms detecting intrusions, because it allows to have additional information. Nevertheless, their study uses the SHAPE algorithm to perform the interpretation of predictions. It is therefore necessary to implement other types of classification algorithms in order to compare the different results obtained. Their study is also limited to highlighting the features that allowed the classification of the data set. An interesting approach would be to analyze the reasons behind the classification of a certain category of data, such as false positives or false negatives.

In [81], the authors propose a mechanism to provide post-hoc explanation of an anomaly detection algorithm for DoS attacks. The goal of this method is to know the trust that a user can have on a prediction. Their method is based on a Neural Netword and was tested on the NSL-KDD dataset with a 97% of accuracy.

In conclusion, the internet of things refers to a set of technologies allowing communication between several objects. Objects can then communicate directly with each other and exchange a large amount of information. This provides a lot of functionality, but it also offers greater vulnerabilities for potential attackers. It is therefore necessary to design security mechanisms that are adapted to this new type of communication. One of the mechanisms studied in the field of cybersecurity is the Intrusion Detection System (IDS). This tool is a second level of security, after mechanisms such as encryption, which allows to detect suspicious behaviour. There are two main types of IDS: signature-based and anomaly-based. The first one requires a regular update of a database, so the IDS works with the detection of anomalies are more adapted to the context of the connected objects. Nevertheless, this type of algorithm is not flawless and makes classification errors. When it classifies a data as an anomaly when it is a normal data, it is called a false positive. When an anomaly is classified as normal data, it is called a false negative. Reducing these errors is an important research issue. Indeed, if the number of false positives is too high, compared to the true positives, then the user will tend to ignore the alerts, making the system useless. If there are false negatives, it means that the system has not been able to detect an intrusion. Another negative aspect of machine learning algorithms is that they are difficult for a human being to interpret. In the case of IDS this corresponds to the lack of knowledge of the reasons why the system is not able to detect an intrusion which led to the detection of an anomaly.

CHAPTER 3 A NEW APPROACH IN THE DETECTION OF ANOMALIES

In this chapter we explain the solution we implemented in this thesis. We describe the dataset used, the detection and explanation algorithms implemented.

3.1 Dataset

The NSL KDD dataset is an improvement of the KDD dataset that was made by DARPA for a competition in 1999. This old dataset was designed by simulating a US Air Force network for 9 weeks and injecting different attacks. The NSL KDD dataset is composed of 125973 data in the training file and 22544 data in the test file. Each data contains 41 features and a label indicating its class of membership.

All the attacks contained in the files can be grouped into 4 categories :

- DoS: sending a very large number of packets over the network to make it unusable.
- Probe: an attacker tries to gather information about a network element by analyzing the network's responses to the requests he sends to recover confidential data or observe a vulnerability.
- R2L (Remote-to-Local): an external attacker tries to exploit different vulnerabilities in order to obtain local access rights on a machine in the network
- U2R (User-to-Root): an attacker has local access on a machine and tries to get root access.

The following research focuses on the study of DoS attacks.

3.2 Design choices

The development of the algorithm carried out in this research was done in several phases. First an analysis of the dataset and the different detection methods was performed on computer. The use of the Python language was particularly adapted in this context because the numerous machine learning libraries developed in this language allow to quickly and simply test the performance of the different methodologies used. In particular, the use of the

Scikit learn library allowed to test the different machine learning algorithms and to compare the results obtained with the literature. The following algorithms have been tested on the NSL-KDD in order to compare their results:

- Random Forest
- Decision Tree
- K Neighbor Classifier
- One Class SVM
- SVM
- Voting Classifier
- Perceptron

This phase allowed us to choose the SVM algorithm because of its good performance on the NSL KDD dataset and his adaptation to other environment, according to the literature review. The use of the panda library also facilitated the preprocessing of the data and thus to be able to test several preprocessing actions in order to obtain the best possible results.

The Python language is very useful in machine learning as many researchers have developed tools to facilitate development. However, the performances in terms of computing time do not allow a program to be deployed on an embedded system. It is true that most of the Python functions used call programs in C but the performances and characteristics of a system developed only in C allow a more adequate use in the field of embedded systems. Therefore, a second phase of development has been carried out in order to design C programs.

The implementation of the intrusion detection system within a node of the network is carried out on a computing unit in parallel with the processor which allows the normal operation of the network. The objective is to reduce the execution time of the detection program. It also allows a large number of tests to be carried out by isolating the detection system. The choice of this calculation unit is based on several criteria. Firstly, the computing power and consumption are two very important constraints in the development of an embedded system. It is indeed necessary to respect strong time constraints so that the processing time of a packet is less than the time interval between the reception of 2 packets. Concerning power consumption, it is necessary to take into account the fact that the system must be deployed in systems with high energy constraints such as UAV. Then one of the important points to take into account is the ease of development and modification of the program. Productivity is a very important concept in the field of IoT because competition is very important and requires a very high speed of development in order not to miss the time to market. It is also necessary to be able to modify the program, or at least part of it, quickly because this allows the program to be improved quickly after a test. For all these reasons, FPGA-type boards are relevant in the context of intrusion detection for IoT.

The role of the resulting system is to detect intrusions and determine that they were the features that most contributed to this classification. It was therefore necessary to choose a method that performs a post-hoc analysis of the execution of the machine learning algorithm and that allows to highlight important features. Another element is the type of data. Indeed the analysis cannot be performed in the same way if it is image, textual information or numerical data. For example, a classical method for explaining machine learning algorithms working on images is able to highlight a part of the image by having one or more images at the output of the explainable AI, which does not correspond to what we want to achieve here. In summary, the explainable mechanism had to be adapted to numerical data whose classification we want to study after the execution of the machine learning algorithm. Therefore, the chosen method is the Partial Dependence Plot which respects all of these conditions.

3.3 Support Vector Machine

The machine learning algorithm applied for the detection of attacks in this research belongs to the Support Vector Machines (SVM) family. This method can be used in several types of applications such as classification, regression or anomaly detection. The principle of regression is to predict the value of a variable, for example the price of a house, according to different parameters, such as the size of the house or its location. This is out of the scope of this research, therefore the rest of the paper will be devoted solely to classification and anomaly detection. The objective of SVMs is to classify input data into several categories. In the context of our research, we mainly seek to distinguish normal data from anomalous data. This type of classification containing only 2 classes is called *binary classification*. A SVM type algorithm looks for the equation of a hyperplane to separate the data into two categories. In the case of a three dimensional space, the separation is performed using a plan. In the general case of n-dimensional data, the separation is performed with a hyperplane of dimension (n-1).

In order to obtain the equation of this hyperplane, the algorithm must first be driven by a set of data. Classically, the execution of a machine algorithm takes place in two phases:

- The training phase. At this stage, part of the dataset is used to train the algorithm and obtain the model. The main advantage of machine learning is that the algorithm is general to several application domains and that only the training data modify the behaviour of the algorithm by obtaining a different model for different training data.

- The testing phase. Once this model is obtained the algorithm can be used to classify new data. The other part of the dataset is then used to test the performance of the obtained model. The data are divided into several categories after classification (false positive, false negative, true positive, true negative). Metrics are then calculated to determine the classification performance of the model.

The figure 3.1 shows a type of separation when the data are not distributed in such a way that a separation could be obtained using a straight line.

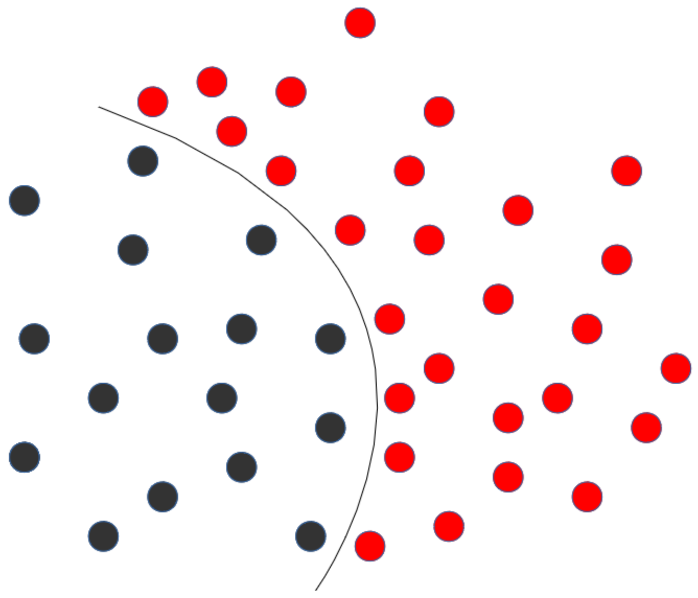


Figure 3.1 Classification with a curve

This example shows a data separation that cannot be achieved by a linear hyperplane. In this case the classification is called *non-linear classification*. Therefore it is necessary to modify our approach so that the algorithm can separate these data into 2 categories. To do this we add an algorithm that modifies the data to adapt them to the linear classification algorithm. This mechanism is called Kernel Trick [82]. Its objective is to modify the data in such a way that we can apply the classification algorithm afterwards. The principle is to convert the input data into data of a higher dimension so that a linear separation can be achieved in this higher dimension. The resulting data can then be sent to the linear classification algorithm. In summary, when classifying a new data, the n-dimensional data to be classified are sent to the algorithm performing the Kernel Trick. The obtained data is then of dimension p>n. The latter is sent to the linear classifier which calculates the class to which the element belongs.

Different formulas are classically used to apply the Kernel Tricks method. The most common are the following, with x the data represented as a vector and x' the *support vector* calculated during the training phase:

. Radial basis function (RBF) :

$$K(x, x') = e^{-\gamma . ||x - x'||^2} \tag{3.1}$$

. Polynomial :

$$K(x, x') = (\gamma . < x, x' > + r)^d \tag{3.2}$$

. Linear :

$$K(x, x') = < x, x' > \tag{3.3}$$

. Sigmoid :

$$K(x, x') = tanh(\gamma . < x, x' > + r) \tag{3.4}$$

The machine learning algorithm used is the Support Vector Classification, the most commonly used SVM, with a linear kernel trick. It belongs to the category of classification algorithms since it is capable of classifying input data into several categories. However, in our research we are looking to classify these data into only 2 categories: normal or anomaly. The functioning of this algorithm is identical to what has been explained above. First a training phase allows the algorithm to learn the model and then an inference phase is performed on new data.

3.4 Partial Dependence Plot

The main contribution of this research work is the application of a method to generate explanations of the results obtained by the machine learning algorithm. In this research we focused on a particular method in this field: the Partial Dependence Plot method [83]. Its objective is to show the impact of one or more features in the output of the machine learning algorithm. The result obtained can be displayed in several ways.

The objective is to highlight the relationship between a particular feature and the output of the machine learning algorithm. To do this, predictions are computed by setting a feature and varying the other features on their domain of validity. This operation is then performed for each value of the feature studied.

The formula used to arrive at this result is as follows:

$$\widehat{f}_{x_s}(x_s) = \int SVM(x_c, x_s)dx_c \tag{3.5}$$

x_s: the studied feature
x_c: the other features
$\widehat{f}_{x_s}$: the partial function
SVM: the prediction function

The use of the exact formula is not realistic given the calculation time required. Indeed, for each step of the integral, a call to the prediction function must be applied on the whole dataset. Therefore it is necessary to approach this calculation by approximation methods.

The Monte Carlo method is then used to strongly reduce the calculation time. By this method we can know the average output result of the algorithm for a given value of the feature. The principle is to use only the values which are present in the dataset. For that, we fix the feature of all the data of the dataset to a precise value then we make a prediction for each data of the dataset. An average is then realized on all the predictions obtained. The formula is as follows:

$$\widehat{f}_{x_s}(x_s) = \frac{1}{n}.\sum_{i=1}^{n} SVM(x_s, x_c^{(i)}) \tag{3.6}$$

n: the number of data in the dataset
$x_c^{(i)}$: actual features of the dataset

3.5 Proposed approach

In our research, we use this mechanism to study the impact of each feature on the classification of a particular dataset. Therefore, we will not try to display all the features but rather to determine which features had the greatest impact on the classification of a particular data item. This allows us to highlight only the features that had the most impact on the detection of a particular anomaly.

The process of detection and explanation is divided into 3 phases:

- SVM training: data from the training dataset is sent to the SVM training algorithm. The model, composed of the support vectors, is then obtained.

- Computation of partial dependence values: for each feature in the dataset, the algorithm computes its partial dependence values. To do so, it queries the trained SVM for each data obtained by the Monte Carlo method. The SVM algorithm detects whether it is an anomaly or a normal data. The partial dependence algorithm then stores its results in a file.

- Test phase: during the actual execution of our method, the analysed data is sent to the SVM which determines whether it is an anomaly or a normal data and to the partial dependence algorithm which provides an explanation to the user.

The figures 3.2, 3.3 and 3.4 show these different phases:

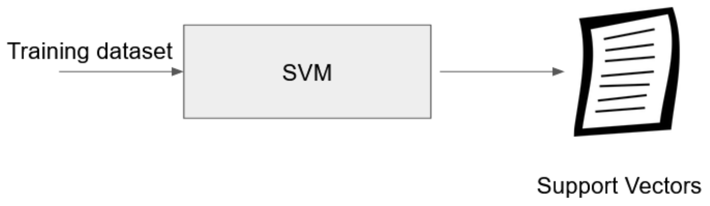


Figure 3.2 SVM training

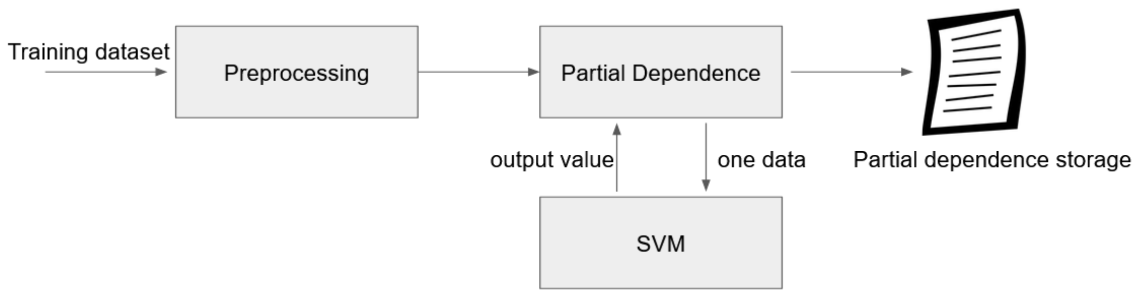


Figure 3.3 Computation of partial dependence values

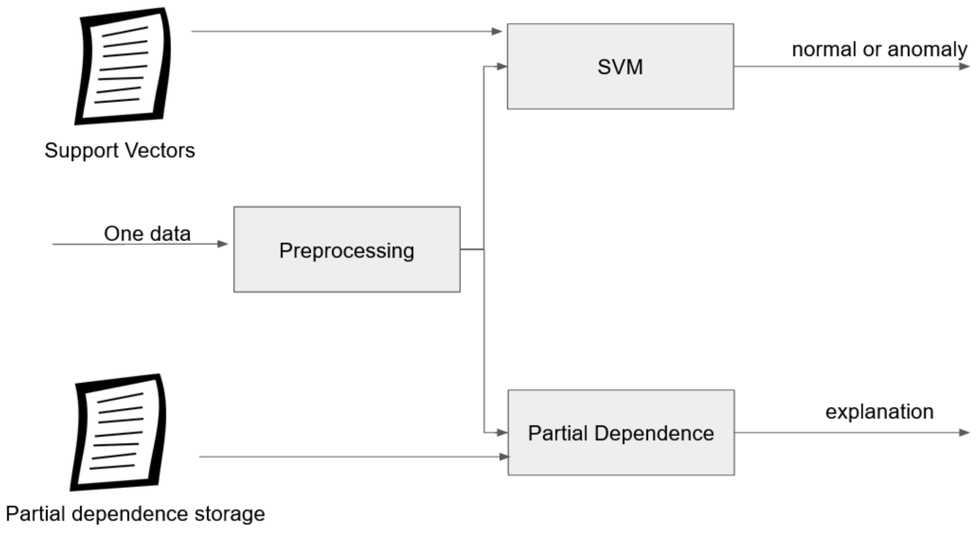


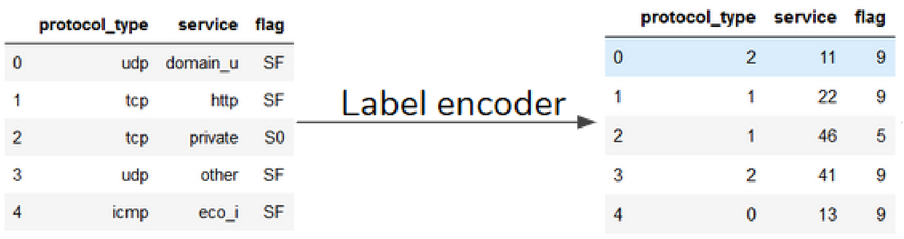
Figure 3.4 Test phase

The second objective of our research is to obtain a mechanism that can be implemented in a network of connected objects, therefore system constraints have to be taken into account. In particular the detection of an anomaly must be as fast as possible so that the system can react quickly to an attack. The method used to satisfy this constraint is to perform a large part of the calculations offline, i.e. before the implementation on the real system. The second operation consists in calculating offline the partial dependence for each feature and then storing this information in the memory of the embedded system. Then, during the operation of the system, when an anomaly is detected, the algorithm retrieves the corresponding value for each feature of the anomaly in order to highlight the features that have the most impact in the classification of this particular data.

The proposed approach combines the two previous methods. The SVM machine learning algorithm detects anomalies and the Partial Dependence algorithm highlights an explanation of the IDS results.

3.5.1 Preprocessing operations

The first phase of development was to implement the SVM algorithm to obtain good results on the NSL-KDD dataset. For this purpose, several steps are necessary. The first phase is a classical machine learning operation: preprocessing. Indeed, the data present in the dataset are of several types (text, number) whereas the SVM must have only numerical values in input to work. It is therefore necessary to convert text features into numbers. In this dataset the text values are categorical values, i.e. the feature has a fixed number of possible values. For example, the feature protocol_type has only 3 possible values: tcp, udp or icmp. The first operation on these categorical values is called Label Encoding. In this phase a numeric value is assigned to each category. The figure 3.5 shows the result of this step on some data of the dataset.

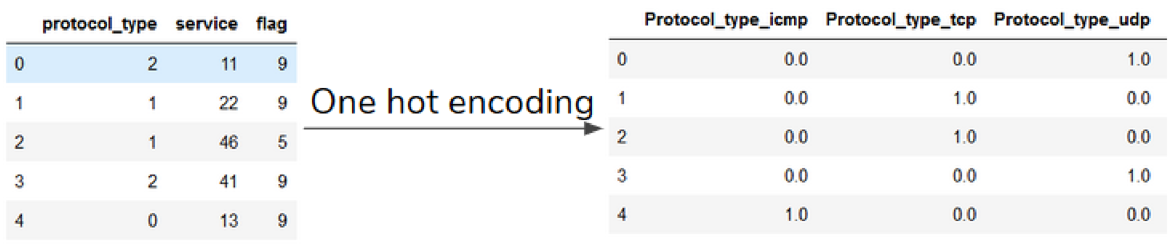

Label encoder

	protocol_type	service	flag
0	udp	domain_u	SF
1	tcp	http	SF
2	tcp	private	S0
3	udp	other	SF
4	icmp	eco_i	SF

	protocol_type	service	flag
0	2	11	9
1	1	22	9
2	1	46	5
3	2	41	9
4	0	13	9

Figure 3.5 Label encoding operation

The second operation is called One Hot Encoding. The previous calculation transformed a text value into a numeric value. However, it is not possible to send this data directly to the machine learning algorithm. Indeed, an important heuristic in this area is that the performance will be better if the values of the input data are similar. For example, there should not be a feature with values between -10 and -5, and another with values between 20 and 30. In our case, the number of categories is different depending on the feature. The feature protocol_type has 3 while the feature service can have 70 possible values. To solve this problem, the One Hot Encoding operation transforms the feature with n possible values into n features. Therefore each new feature corresponds to a possible category. The value of this feature is 1 if the feature of the original dataset corresponded to this category, and 0 otherwise. The figure 3.6 shows the result of this step on some data of the dataset.


One hot encoding

	protocol_type	service	flag
0	2	11	9
1	1	22	9
2	1	46	5
3	2	41	9
4	0	13	9

	Protocol_type_icmp	Protocol_type_tcp	Protocol_type_udp
0	0.0	0.0	1.0
1	0.0	1.0	0.0
2	0.0	1.0	0.0
3	0.0	0.0	1.0
4	1.0	0.0	0.0

Figure 3.6 One hot encoding operation

These operations allow to change the textual values of the dataset. Nevertheless the new dataset is not sent to the anomaly detection algorithm because the performances would not be optimal. Indeed, as mentioned above, the features must be similar. This is not the case in our dataset because the possible value ranges of the numerical features are not similar. A feature scaling operation must therefore be performed in order to maximize SVM performance. There are several methods that can be used to perform this step. The formula used in this search is mean normalization. The preprocessing operations are computed to obtain only numerical values, so we can use this common formula in classification problems:

$$x_i := \frac{x_i - mean_i}{max_i - min_i} \tag{3.7}$$

3.5.2 Metrics

Following these pre-processing steps, the data can now be analysed by the machine learning algorithm. The performance of this algorithm is then studied by the cross-validation method. The principle is to divide the dataset into k blocks, (k-1) blocks are used to train the machine learning algorithm and the last block is used for validation, this iteration is repeated on each block. We can then obtain the mean, and the standard deviation, of the following metrics:

- *accuracy*: the rate of correct classification
- *precision*: measures the confidence we can have in the outcome of the IDS when it classifies a data as anomaly
- *recall*: measurement of the rate of anomalies detected among all the anomalies present in the dataset
- *F_measure*: harmonic mean of precision and recall

The value of its metrics is calculated using these formulas:

$$accuracy = \frac{True\_Positives + True\_Negatives}{Number\ of\ Samples} \tag{3.8}$$

$$precision = \frac{True\_Positives}{True\_Positives + False\_Positives} \tag{3.9}$$

$$recall = \frac{True\_Positives}{True\_Positives + False\_Negatives} \tag{3.10}$$

$$F\_measure = 2 * \frac{1}{\frac{1}{precision} + \frac{1}{recall}} \tag{3.11}$$

In conclusion, several operations need to be carried out to obtain a mechanism for detecting anomalies with explanations. First of all, the input data are subjected to a preprocessing operation to obtain numerical data suitable for classification by the SVM algorithm. The first phase is the training of the machine learning algorithm. This step creates the model for the classification. Next, the dependence values are calculated. This is based on the Monte Carlo method and the trained machine learning algorithm. Finally, during the validation phase, the input data are preprocessed and then sent separately to the detection algorithm and the explanation algorithm. The first classifies the data and the second provides the list of features, with their partial dependence, in order of importance for classification. The user can thus know the category of the data, as well as the features that have had the most impact in the detection.
In the next chapter, we present a software that allows to recover data from the simulation of an ad hoc network.

CHAPTER 4 APPLICATION OF THE IDS ON A SIMULATOR

A key issue in the field of machine learning is access to data. As the model is created from a set of data during the training phase, the data must be as close as possible to the environment in which the system will be deployed. The data from the NSL-KDD are useful for analysing the behaviour of the mechanism and obtaining results on the method of interpretability, but before it is put into service it must be tested in a similar context. For this purpose we used a simulator developed by our industrial partner.
This chapter describes the functioning of the simulated network as well as the architecture of the simulator.

4.1 Network description

The detection system developed during this research is integrated within an ad hoc network. This network consists of many different devices, such as drones or smartphones. The particularity of this type of network is its decentralization. Each node can transmit, and receive, information and the control of the routing is carried out thanks to a routing protocol. This type of network is not fixed in time and has the capacity to reconfigure itself automatically when an element of the network is modified, for example when a device is removed from, or added to, the network. The automatic reconfiguration of the network also allows different devices to move in space. Because different systems have limited communication ranges, two nodes may not be close enough to communicate directly with each other, the network routing mechanism solves this problem. The overall operation of the network is based on the transmission of network packets from one node to another. Figure 4.1 shows an example of how a packet is transmitted from node A to node B.

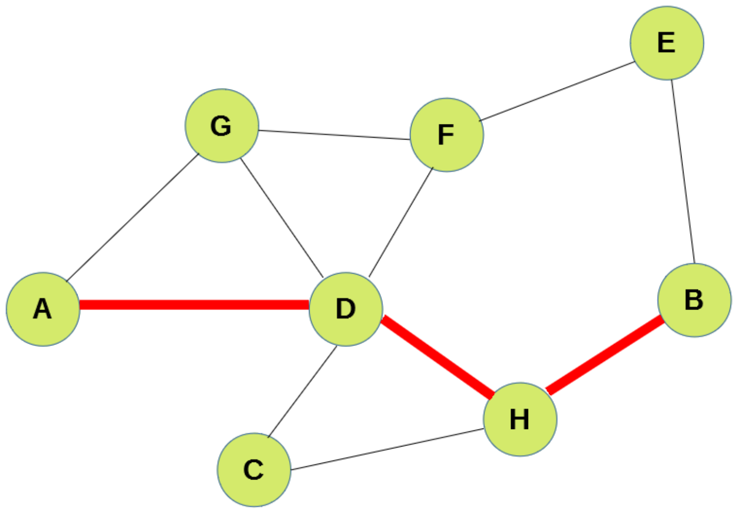


Figure 4.1 Example of packet transmission in an ad hoc network

The advantage of this network is its dynamism and great flexibility, however it also offers the attacker new types of possible attacks. In order to reduce potential attacks, our security system is placed on each node of the network. In order to avoid checking all the packets that would pass through a node, the system developed in this paper only checks the packets received by the node.

4.2 Simulator description

As the network I studied during my master's degree was not deployed in real conditions, the use of a network simulator was necessary to test the system on the ad hoc network. Its principle is to simulate the different network nodes and packet transmissions within the network. This has the advantage of making development and testing easier, as it makes it easier to change network configurations compared to actual deployment. It is therefore possible to obtain a large number of tests with very diverse configurations. Nevertheless, a test phase in real conditions will be necessary in order to validate the behaviour of the system in a real environment. The simulator allows the integration of different objects into the simulation which can be hardware or software.

4.3 Performance constraints

The detection system is intended to be implemented within a network of embedded systems with significant performance constraints. For example, one of the important characteristics of a node is its autonomy. Two constraints are to be taken into account in order to increase its autonomy: the energy stored on board and the energy consumed by all the equipment. In the case of a device that steals the energy that can be stored, it quickly comes up against the weight constraint. It is therefore essential to focus on reducing the energy consumed to increase autonomy. Therefore the intrusion detection system must also be the least energy consuming.

A third extremely important constraint to take into consideration in the implementation of our system is the program execution time. This is due to the methodology of this security mechanism which works as follows: first the network packet is captured, then it is analyzed and finally a second packet is captured. In this operating scheme it is therefore essential that the execution time of the detection system is less than the time between two network packets captures. It is possible to set up a pipelining system in order to facilitate the satisfaction of this constraint but this method is not addressed in this research. Another important constraint in the development of a security system that aims to be implemented in connected objects is the development time required to implement the system. Indeed, the strong competition in this field makes it necessary for companies to develop their technology quickly and cheaply. It is therefore important to take this imperative into account in order to propose a security mechanism that can be implemented in real systems. Finally, the system must also be adapted to changes in the network. Indeed, if the network developers modify the packets transmitted by the nodes or part of the protocols that allow the network to function properly, it is necessary not to have to start developing the security mechanism from the beginning. On this point the methodology used by machine learning algorithms is very efficient. Indeed an algorithm can adapt to the input data and re-train infinitely, as long as the input data have the same characteristics, it allows a great adaptation of the mechanism.

An on-board system must also be reliable during operation. It is therefore essential to design a system that can be tested in a simulation environment and then in real conditions. This aspect is a current weakness of the learning machine because it makes errors and the challenge of research in this field is to minimize these errors. This is an interesting area of research but we will not discuss it here. In this research we bring an interesting result that can be exploited by other research in order to reduce the error rate of the system, nevertheless we do not claim to make the system infallible. Indeed it is important to accept detection errors because they cannot be avoided. This is one of the main reasons why a detection system based on a machine-learning approach can never be used alone and must be seen as an additional layer of security.

4.4 Implementation of the IDS on the Humanitas Simulator

The NSL-KDD dataset is useful for developing the algorithm for detecting anomalies and analysing the results of the explainable AI method. Nevertheless one of the main objectives of the research is to develop the intrusion detection system for an ad hoc network. It is therefore important to carry out a test phase in an environment approaching this type of network architecture. The objective is to use the simulator designed by the company Humanitas Solutions, based on Docker, which simulates the interactions between several nodes using the network they have developed.

4.4.1 Docker

Docker is an open source software that offers the possibility to deploy several applications in separate containers. These applications can communicate with each other through a network service provided by the docker engine. This architecture offers many advantages:

- Isolation between the containers increases security
- The use of operating system services provides better performance than virtualization systems
- The application deployment is facilitated by the use of the containers

In our research, isolation is a great advantage because it improves the safety of the system. However, this architecture does not protect against all risks. One category of possible attacks is the use of root privilege on the host operating system. Indeed the containers use services from the same kernel, therefore if this one is compromised then the containers can be compromised too. Docker also allows to easily retrieve online application images that can be easily executed in a container. This feature is very useful, unfortunately many of these images are malicious. Therefore, care must be taken with the images used. Finally, another category of attack concerns the use of the network. Indeed if an application needs to communicate with another application a virtual network must be set up. This implies a possibility of attack by classical methods.

It is therefore essential to update the Docker installation to correct security flaws and to bear in mind that the isolation offered by Docker does not guarantee complete application security.

4.4.2 Description of the simulation environment

The simulator on which the IDS tests are carried out is based on Docker. This allows us to deploy applications easily, such as the simulation of a malicious node or the implementation of the IDS. The role of the simulator is to manage the hardware resources as well as to enable the launch of applications executed by each node in the network. The general principle is to decrypt the simulation in a scenario file and the simulator carries out the discovery of the available hardware resources and the deployment of each node on a container. The scenario lists the different agents that are used during the simulation. Its agents are then described by configuration files. The configuration files of the agents are used to detail the computation performances and the memory space allocated to each container. An orchestrator program then has the role of allocating the desired capacities and deploying the agents. The simulations that are carried out for the different experiments are based on a test scenario developed by the company: "network_tester". The role of this scenario is to carry out simple tests and to provide a set of commands to carry out actions on the network. For example, a "ping" command is available to allow packets to be sent by all the nodes.

The figure 4.2 summarises the different containers operating during a simulation:

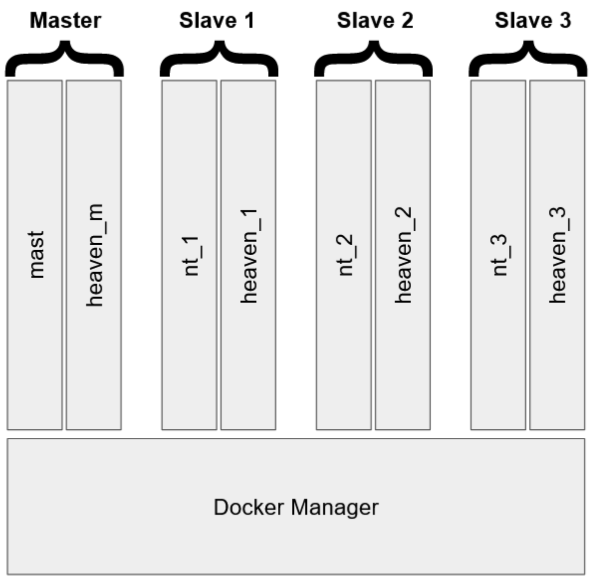


Figure 4.2 Example of running containers

4.4.3 Scenarios

The use of dockers in the network implementation allows for easy deployment of network nodes. In order to carry out several experiments, three configurations are carried out:

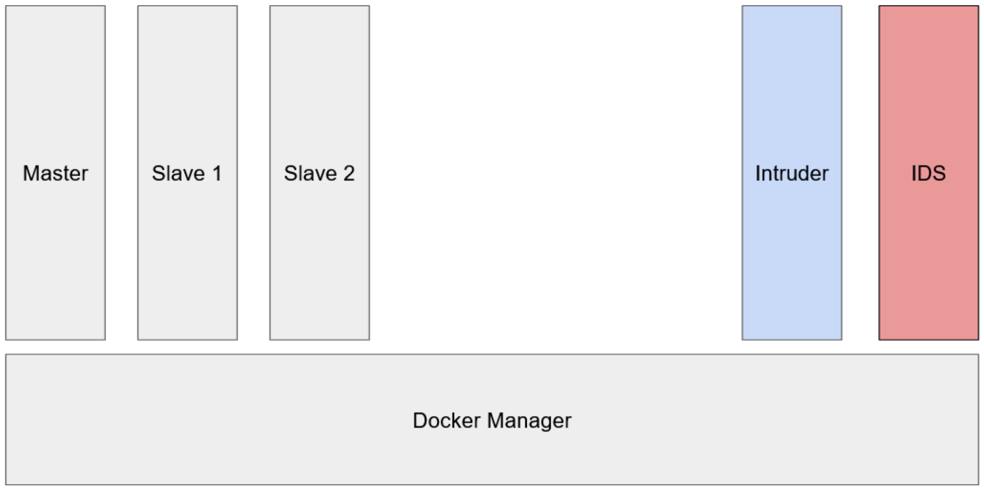


Figure 4.3 One attacker and two slave nodes

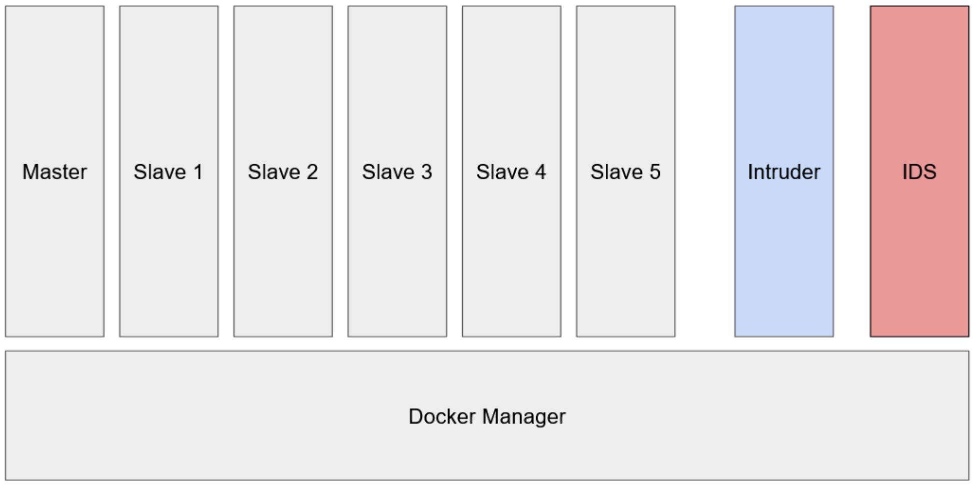


Figure 4.4 One attacker and five slave nodes

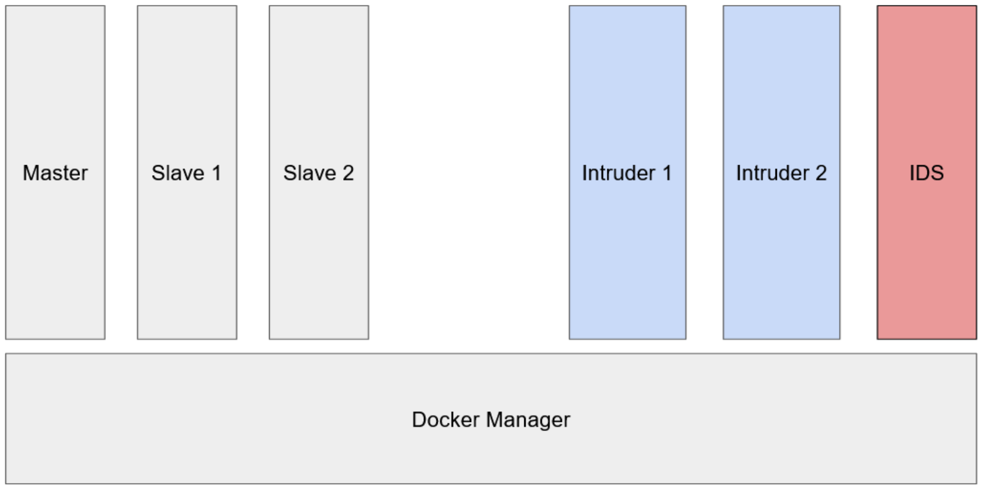


Figure 4.5 Two attackers carrying out a DoS attack simultaneously

These different configurations enable to analyse the behaviour of the IDS in several different situations. Firstly, the simplest case is to deploy an attacker in a network containing a master and two slaves. This case will allow to observe the result of the IDS in comparison with the simple behaviour of a single attacker. In the second configuration, three nodes are added to the network in order to observe the behaviour of the IDS in a larger network. This increases the number of normal packets with the same number of malicious packets. Next, an attacker is added to observe the behaviour of the IDS in the presence of several attackers. In this context, two simulations are carried out. The first operates simply by duplicating the attacker node. The number of packets then arriving on the network is doubled. These relatively simple case studies will allow us to observe the behaviour of the IDS in the network. These tests should not be taken as tests validating the implementation of the IDS in the simulator but as a tool for reflection for a future implementation in a real scenario.

4.4.4 Implementation of IDS

The security mechanism developed in this research is an IDS based on network packet analysis. It is therefore necessary that it is implemented in such a way that it can recover the packets that transit between containers. A simple way to achieve this is to use the advantages of Docker and deploy the defense system in a container during the test phase with the simulator. In this container, several functions are carried out:

- Packet retrieval using tcpdump
- Anomaly detection with the SVM algorithm

- Explanation of the results of the SVM algorithm using the PDP method

The network packets obtained with tcpdump are processed to create the different features used by the detection algorithm. The objective of this experiment is to detect DoS attacks. For this purpose, the following features are sent to the detection algorithm:

- Number of packets received in a given time slot
- Number of packets received from each other node in a given time slot
- Routing Method
- Hop Counter

The behaviour of the detection and explanation algorithms is identical to the experiments carried out with the NSL-KDD dataset. First a learning phase is carried out with a training dataset and then a second one with a test dataset. However, the method of obtaining data from the dataset is not the same. Indeed, the NSL-KDD is obtained by processing the KDD99 dataset. The latter was obtained by setting up a local-area network on a simulation of US Air Force computer hardware.

CHAPTER 5 RESULTS AND DISCUSSION

This chapter contains the major results obtained during this master's degree: the application of our method on the NSL-KDD dataset, allowing an analysis of false positives and false negatives, an experiment testing our IDS against Msika et al. [84] false negative generator and the operation of a detection algorithm in an ad hoc simulator.

5.1 Anomaly detection performance

In this paper the experiments were performed in Python using the scikit-learn library [85] for the classification of DoS attacks of the NSL-KDD dataset. As the execution time increases with the number of features used, the study is based on the analysis of the feature selection of [86]:

Features used
Src_bytes
Service
Dst_bytes
Flag
Diff_srv_rate
Same_srv_rate
Dst_host_srv_count
Dst_host_same_srv_rate
Dst_host_diff_srv_rate
Dst_host_serror_rate
Logged_in
Dst_host_srv_serror_rate
Serror_rate
Count
Srv_serror_rate

Table 5.1 List of features of the NSL-KDD used in this thesis

In order to measure the results obtained, we use the cross validation method with the following 4 metrics: accuracy, precision, recall and F_measure

These metrics were calculated using the cross_val_score function of scikit-learn. The following table shows the test results.

accuracy	97.76%
precision	98.64%
recall	96.19%
F_measure	97.40%

Table 5.2 Average value of the metrics

5.2 Partial Dependence Results

The interest of the explainable AI is to obtain a greater understanding of the model by a human, therefore testing to validate a technology in this area requires confronting the system with a large number of users to get their opinion. Nevertheless it is possible to design metrics to characterize its performance. As a reminder, the objective of the system studied here is to highlight the features that have had the greatest impact in classifying a particular data item as an anomaly by the IDS. We display in table the main 5 features for a true anomaly taken at random in the test dataset. Among these 5 features with the highest partial dependence value, there are three service features (service_ecr_i, service_private, service_domain), the feature dst_host_srv_count and a flag feature (flag_S0).

feature name	preprocessed value	partial dependence value
service_ecr_i	4.63	93.40%
dst_host_srv_count	-1.10	82.65%
service_private	-0.03	46.43%
service_domain_u	-0.23	42.68%
flag_S0	0.72	42.65%

Table 5.3 Example of the outcome of the explainable algorithm

In this example, we see that, on average, a data whose feature service_ecr_i is 4.63 has a 93.40% chance of being classified as an anomaly. In order to see if it can be said that the first feature has a significant impact on the decision, we modify the tested data by replacing the value of the important feature by the value that on average gives the detection of a less probable anomaly. The detection algorithm is then applied to determine whether the output remains the same or whether the data under study is now classified as an anomaly. By doing this on all data that have been classified as anomalies a modification success rate defined as follows can be achieved:

$$success\_rate = \frac{nb\_modification\_success}{nb\_anomalies} \tag{5.1}$$

This formula is applied by modifying only the first feature returned by the PDP method, then the first two and this up to the first 5 features. The results of this experiment are shown in figure 5.1, then we perform another experiment by modifying only the 6th feature, then the 6th and 7th and this up to the 15th feature. The results of this experiment are shown in figure 5.2.

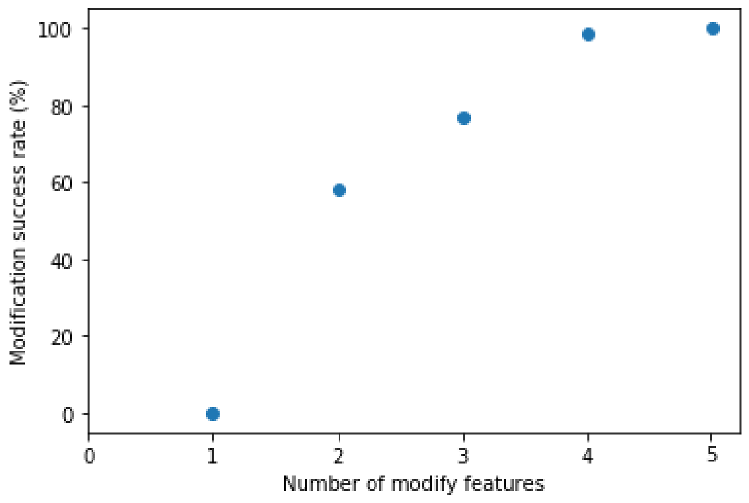


Figure 5.1 Evolution of the success rate for the first five features

The results obtained show that from the 5th modified feature the algorithm no longer classifies any data as anomalies.

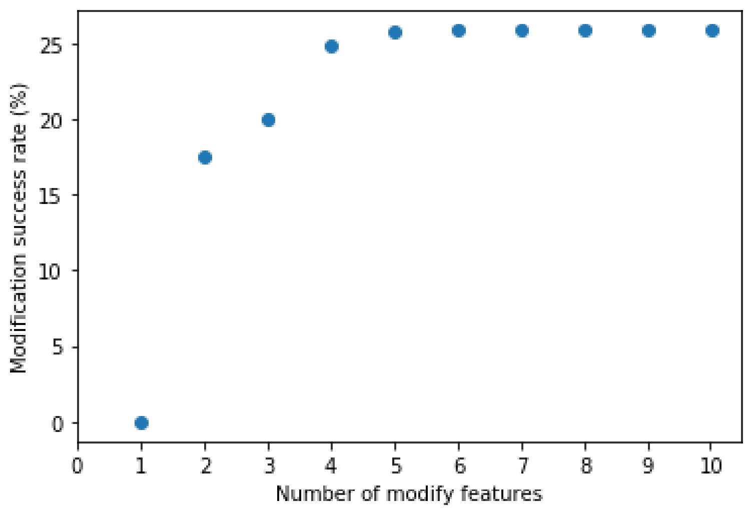


Figure 5.2 Evolution of the success rate for the features six to ten

Figure 5.2 shows the evolution of the success rate as a function of the number of features removed after the 6th feature returned by the PDP algorithm. A limit is then quickly observed, which corresponds to the fact that from the 9th feature onwards, the modifications of the data have little effect on the SVM decision.

The results obtained show that the first 4 features explain in more than 98% of the cases the classification as an anomaly of the processed data. It can thus be stated that the PDP method correctly calculates the most important features for a particular data item.

5.3 Discussion on explaination outputs

The application of the PDP method has therefore made it possible to highlight important features in the detection of an anomaly. It is thus possible to know which features contributed to the classification of a data item as an anomaly. And therefore it is also possible to know which features have misclassified a data. Indeed, an important problem in machine learning algorithms applied to anomaly detection is the number of false positives. In this research we have therefore wondered if it is possible to obtain additional information about these false positives by applying the partial dependence method. In this part, an analysis of the results is therefore carried out. The objective is to study separately the results of the PDP method for false positives and true positives to observe if different results appear. The following results were obtained by applying the PDP method on the whole dataset. The figure 5.3 shows the values of the feature that has the greatest partial dependence value for true positives.

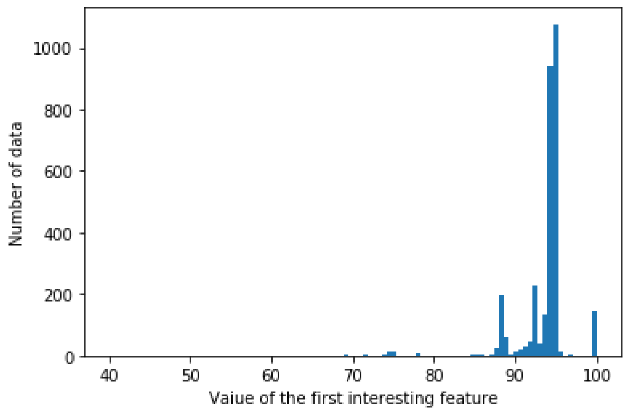


Figure 5.3 First interesting feature of True Positives

On this graph, we can see that the values are centered around 94%. This was expected because the greater the value of the partial dependence, the greater the chance of being classified as an anomaly on average. The figure 5.4 shows the values of the feature that has the largest partial dependence value for false positives.

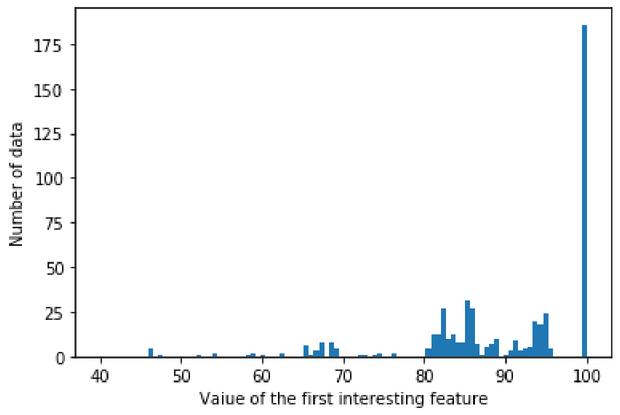


Figure 5.4 First interesting feature of False Positives

The results observed here explain in part why the machine learning algorithm misclassified this data. Indeed, the high values of partial dependence for these features correspond to what can be expected from an anomaly. Therefore it is interesting to analyze the other important features to see if there is still a resemblance between a positive and a false positive. The figure 5.5 shows the values of the feature that has the second largest partial dependence value for true positives.

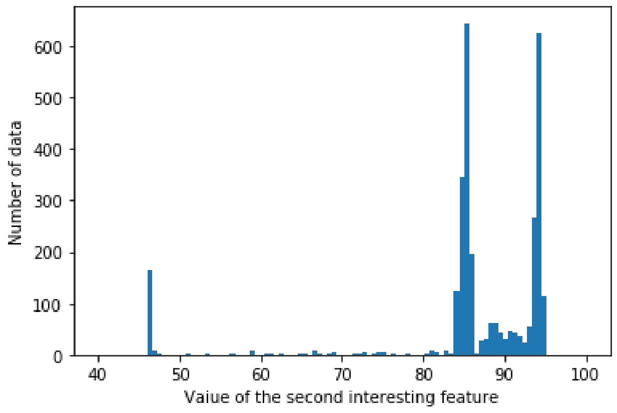


Figure 5.5 Second interesting feature of True Positives

On this graph we can see that there are a few values that are around 46% but the vast majority have high values. The fact that the majority of these values are high shows that there is not only one feature that influences the algorithm in the case of true anomalies but many.

The figure 5.6 shows the feature values that have the second highest partial dependence value for false positives:

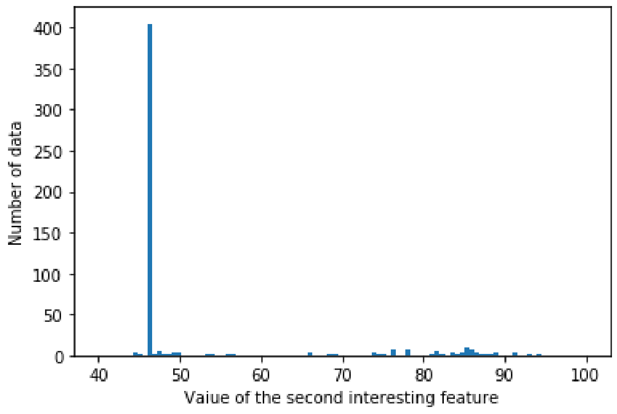


Figure 5.6 Second interesting feature of False Positives

The result obtained here is very interesting because, for the first time, we see a difference between true positives and false positives. Indeed the true positives have a majority of high values while it is the opposite for the false positives. This result seems to show that the classification errors are mainly due to a single feature.

The previous results focused on the analysis of false positives and true positives. But it is also interesting to analyze the results of our approach for true negatives and false negatives. Indeed it is also important to understand why an anomaly is classified as normal data. This can indeed have serious consequences in the field of security because such an attack would not be detected. The low percentage of non-detection of attack may seem to be enough but it is this security flaw that attack algorithms based on Generative Adversarial Network (GAN) approaches try to exploit to carry out their attacks.

The table 5.4 shows the results for the 5 most important features. In general, the first feature seems to be the cause of the classification error, because for example in the case of false positives, the values of partial dependence are high for the first feature and then low for the second. For true positives, these values are on average high for the first two features and then decrease progressively. Consequently, the observation of the second feature allows to say if it is probably a true positive or a false positive.

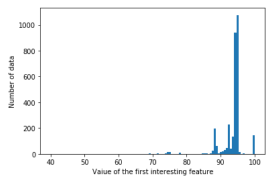
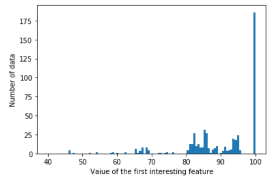
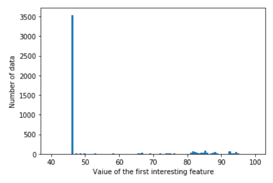
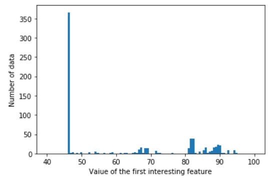
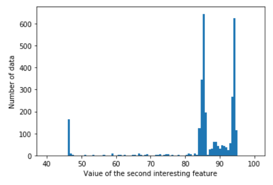
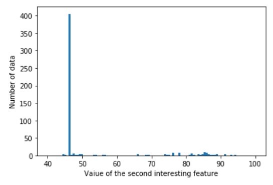
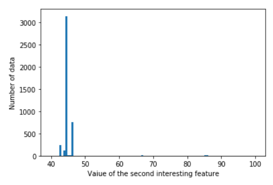
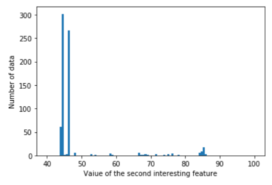
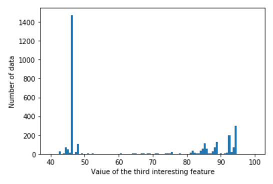
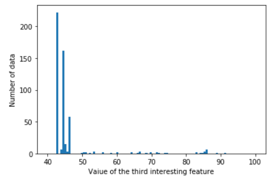
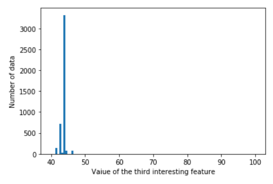
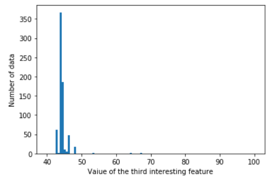
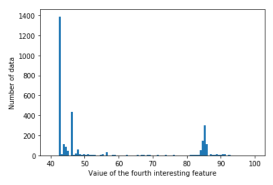
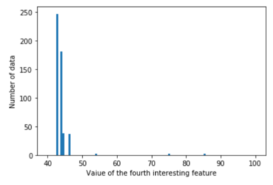
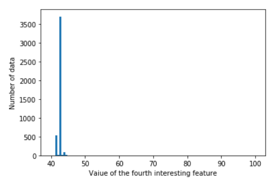
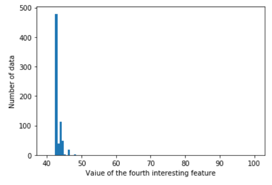
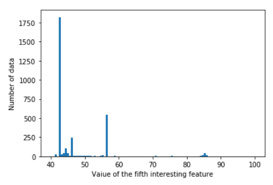
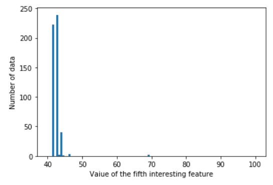
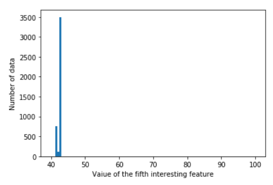
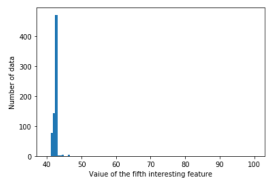
	True Positives	False Positives	True Negatives	False Negatives
First feature				
Second feature				
Third feature				
Fourth feature				
Fifth feature				

Table 5.4 Results for the 5 most important features

5.4 Threshold on anomalies

In order to detect a false positive, an interesting method would be to place a threshold on the partial dependence below which a data item is considered to be a false positive. The table 5.5 shows the results obtained by this method with different thresholds on the second and third important feature. What are called "false positives" in the table are data that are labeled as normal values in the dataset but have a partial dependence above the threshold.

These results show that the number of false positives that are still considered as anomalies after applying our method decreases sharply. Nevertheless, the number of anomalies detected is also decreasing.

	Number of True Positifs	Number of False Positifs
Without threshold	3 054	509
Threshold of 48% on second feature	2 878	93
Threshold of 52% on second feature	2 877	85
Threshold of 75% on second feature	2 814	66
Threshold of 85% on second feature	2 313	32
Threshold of 46% on third feature	2 892	101
Threshold of 60% on third feature	1 273	31

Table 5.5 Application of a threshold for data classified as positive

5.5 Threshold on normal data

So far, we are interested in differentiating between false positives and true positives. But it is also important to analyze false negatives. Indeed, this category contains data that are classified as normal while they are anomalies. These data are very dangerous in our context because they are attacks that the algorithm did not detect. The table 5.6 shows the results obtained by this method with different thresholds on the second and third important feature for the data classified by the detection algorithm as normal data. As in the previous one, the so-called "false negatives" in the table are data that are labelled as normal values in the dataset but have a partial dependence below the threshold.

	Number of True Negatifs	Number of False Negatifs
Without threshold	4 362	701
Threshold of 60% on first feature	3 626	392
Threshold of 50% on first feature	3 587	375
Threshold of 48% on second feature	4 288	632
Threshold of 52% on second feature	4 299	637
Threshold of 60% on second feature	4 316	645

Table 5.6 Application of a threshold for data classified as negative

In the case of the analysis of false positives or false negatives, we notice that the implementation of a threshold reduces the number of false positives (or false negatives) but in return it also reduces the number of true positives (or true negatives), these data then become false negatives (or false positives).

5.6 Experiment against a GAN algorithm

Machine learning algorithms also have weaknesses that an attacker can exploit. A powerful method to achieve this is to use a GAN type approach. Figure 5.7 shows the operating diagram of this type of attack. The principle is to generate anomalies but which will be detected as normal by the algorithm, i.e. being false negatives.

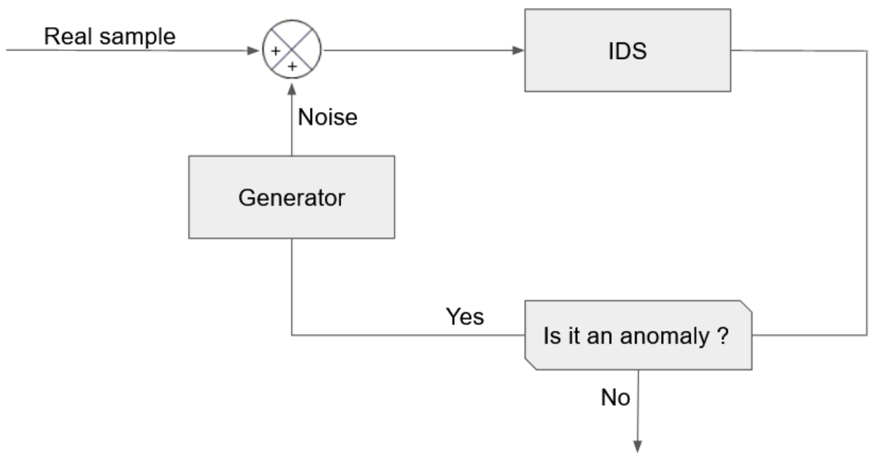


Figure 5.7 Creation of GAN type attacks

We wanted to know whether our approach could be useful against this type of attack. In order to do so, the attack generators developed by Msika et al. in [84] have been used. This generation system works by adding noise to an anomaly, which allows the IDS to be fooled. We then recover the false negatives that have been generated. These data are then sent to our algorithm for classification with a threshold on the values of the second important feature. of partial dependence of 60.
When the SVM algorithm is tested with the partial dependence computation mechanism, a accuracy of 93.8% is obtained. The result obtained is lower than the result during the normal operation of a detection algorithm that is not subject to an attack because true negatives had values of partial dependencies above the threshold, and have therefore been classified as anomaly. Similarly, true positives had a partial dependence score below the threshold. and

were therefore not recognised as positive after the threshold transaction.
The remark that can be made about this simple experiment is that our IDS has allowed us to better resist these types of attacks because the noise modifies the value of the features, and therefore its detection, but it modifies the partial dependence very slightly. Therefore our thresholding approach sees very little difference between a GAN-modified anomaly and the original anomaly.

5.7 Humanitas Solutions Simulator

The simulator provided by Humanitas Solutions makes it possible to carry out some experiments. As a reminder, the different configurations are as follows:

- Configuration 1: One attacker and two slave nodes
- Configuration 2: One attacker and five slave nodes
- Configuration 3: Two attackers carrying out a DoS attack simultaneously

5.7.1 Configuration 1

The objective of this first configuration is to have a simple case in order to have a point of comparison for the other scenarios. In this case, the accuracy obtained is 98.31%.

5.7.2 Configuration 2

In this case, we increase the number of nodes in order to see the impact of the *normal* nodes number on the detection capacity of the algorithm. The result obtained, an accuracy of 88.15%, shows that the detection algorithm has more difficulty in finding anomalies. Indeed, the normal nodes send more packets on the network, therefore the number of packets coming from the attacker are proportionally lower than in the case of the first configuration and therefore it is more difficult for the SVM to detect an anomaly.

5.7.3 Configuration 3

This scenario is realised by duplicating the behaviour of the attacking node. The packets coming from an attacker are then more numerous and the result of the detection is therefore better than in the previous configuration: 90,35%.

5.7.4 Partial dependence

At the same time, the calculation of partial dependencies showed that the two features that are most important in classifying an anomaly are the number of packets received in a given time slot and the number of packets received from each other network node in a given time slot. This result could have been expected because the attack carried out is a very simple DoS attack that sends a lot of packets over the network without making the effort to hide.

It is important to keep in mind that this experiment allows to set up a test architecture on the simulator but does not aim at validating the anomaly detector so that it can be deployed in a real use case. Indeed, it is limited to a DoS attack with only four features. In the real-world environment, it will be necessary to capture other features in order to be able to detect other attacks.

CHAPTER 6 CONCLUSION

6.1 Summary of Works

The rapid development of the Internet of Things offers new functionalities, but increases security risks. Indeed it increases the attack surface and increases the amount of data that is collected and transmitted. For these reasons, it is necessary to design security mechanisms to meet the constraints imposed by connected objects. This research proposes to design an intrusion detection system that is adapted to the context of the connected objects using a machine learning algorithm that allows the detection of new attacks. The application of a machine learning algorithm in the case of a security mechanism is interesting because it allows to add an additional layer of security, but it raises some problems that need to be studied. Machine learning algorithms do not currently have a 100% success rate and their certification is still very complicated. The large number of false positives can be a hindrance to their development, as a large number of false alarms can cause the user to disregard IDS decisions. A second issue in the integration of machine learning algorithms in this domain is their behaviour when faced with an attacker. Indeed, even if the algorithm classifies an anomaly as normal data with an extremely low rate, this failure can be exploited by an attacker. This research has therefore highlighted the possibility of using an explainable AI method in the study of false positives. The application of the Partial Dependence Plot method for anomaly detection with a Support Vector Machine algorithm on the NSL-KDD dataset showed a significant difference between true positives and false positives. Indeed, in the case of false positives the first important feature, according to the PDP method, has a major role in the error that is realized by the classification algorithm. This is not the case for the majority of true positive, which keeps an important value for the following features.

6.2 Limitations

This work revealed a differentiation between true and false positives on the NSL-KDD dataset. Nevertheless for an implementation of this method in the domain of connected objects, several limitations are present. First, this algorithm has not been tested on a real environment. The results obtained here are therefore valid in the particular case of the NSL-KDD dataset but this does not mean that they will necessarily be valid on a network of connected objects. The second limitation to this method is that it allows to say that a detected anomaly is probably a false positive, but the algorithm cannot classify the dataset as necessarily being in this case, because there are also true positives that have low second feature values.

6.3 Future Research

This research has shown the possibility and interest of applying an explainable AI method to an intrusion detection algorithm. Future research can therefore use this work to improve the performance of intrusion detection systems. The analysis of false positives and false negatives by this method can allow future research to design systems that give an appreciation of the possibility of a false positive or false negative. The implementation of this mechanism within a functional network would allow to validate its good functioning. This research would also make it possible to study the temporal and energy consumption performance of such a system. Once the simulator is operational, an analysis of different detection algorithms can be carried out. In particular, a study on the One Class SVM could be interesting. Indeed, this type of algorithm only needs normal data to build its model during the training phase, thus allowing to not limit itself to a reduced number of attacks. An interesting approach would also be to design a dataset adapted to a problem of connected objects in order to be able to test the algorithms without the need to have access to a real environment. Research would also be to study in detail other methods of the explainable AI. Indeed in this research, we focused on the PDP method but there are other algorithms that could obtain interesting results. Goeschel [63] shows that it was possible to reduce false positives by combining several detection algorithms with each other. The contribution of an explainable AI method makes it possible to understand the reasons for these errors. A future research would be to think about the best way to combine the classification algorithms thanks to the information provided by the interpretation method.

REFERENCES

[1] M. Roesch *et al.*, "Snort: Lightweight intrusion detection for networks." in *Lisa*, vol. 99, no. 1, 1999, pp. 229–238.

[2] V. Paxson, "Bro: a system for detecting network intruders in real-time," *Computer networks*, vol. 31, no. 23-24, pp. 2435–2463, 1999.

[3] K. Gade *et al.*, "Explainable ai in industry," in *Proceedings of the 25th ACM SIGKDD International Conference on Knowledge Discovery & Data Mining*, 2019, pp. 3203–3204.

[4] M. Chiang and T. Zhang, "Fog and IoT: An Overview of Research Opportunities," *IEEE Internet of Things Journal*, vol. 3, no. 6, pp. 854–864, 2016.

[5] A. Louchez and V. Thomas, "E-waste and the internet of things," *ITU News*, 2014.

[6] A. Waisel, 2018. [Online]. Available: https://www.brighttalk.com/webcast/288/318941/how-iot-expands-hackers-attack-surface

[7] M. Antonakakis *et al.*, "Understanding the mirai botnet," 2017, pp. 1093–1110. [Online]. Available: https://www.usenix.org/conference/usenixsecurity17/technical-sessions/presentation/antonakakis

[8] C. Kolias *et al.*, "DDoS in the IoT: Mirai and other botnets," *Computer*, vol. 50, no. 7, pp. 80–84, 2017.

[9] A. K. Sikder *et al.*, "A Survey on Sensor-based Threats to Internet-of-Things (IoT) Devices and Applications," 2018. [Online]. Available: http://arxiv.org/abs/1802.02041

[10] B. Goodman and S. Flaxman, "European union regulations on algorithmic decision-making and a "right to explanation"," *AI Magazine*, vol. 38, no. 3, pp. 50–57, 2017.

[11] M. Gharibi, R. Boutaba, and S. L. Waslander, "Internet of Drones," *IEEE Access*, vol. 4, pp. 1148–1162, 2016.

[12] N. Hossein Motlagh, T. Taleb, and O. Arouk, "Low-Altitude Unmanned Aerial Vehicles-Based Internet of Things Services: Comprehensive Survey and Future Perspectives," *IEEE Internet of Things Journal*, vol. 3, no. 6, pp. 899–922, 2016.

[13] N. Huq, R. Vosseler, and M. Swimmer, "Cyberattacks against intelligent transportation systems," TrendLabs, Tech. Rep., 2017, Tech. Rep., 2017.

[14] D. Kozlov, J. Veijalainen, and Y. Ali, “Security and privacy threats in iot architectures,” in *Proceedings of the 7th International Conference on Body Area Networks*, ser. BodyNets ’12. ICST, Brussels, Belgium, Belgium: ICST (Institute for Computer Sciences, Social-Informatics and Telecommunications Engineering), 2012, pp. 256–262. [Online]. Available: http://dl.acm.org/citation.cfm?id=2442691.2442750

[15] M. Conti *et al.*, “Internet of Things security and forensics: Challenges and opportunities,” *Future Generation Computer Systems*, vol. 78, pp. 544–546, 2018. [Online]. Available: http://dx.doi.org/10.1016/j.future.2017.07.060

[16] J. Gubbi *et al.*, “Internet of Things (IoT): A vision, architectural elements, and future directions,” *Future Generation Computer Systems*, vol. 29, no. 7, pp. 1645–1660, 2013. [Online]. Available: http://dx.doi.org/10.1016/j.future.2013.01.010

[17] S. Sicari *et al.*, “Security, privacy and trust in Internet of things: The road ahead,” *Computer Networks*, vol. 76, pp. 146–164, 2015. [Online]. Available: http://dx.doi.org/10.1016/j.comnet.2014.11.008

[18] A. Riahi Sfar *et al.*, “A roadmap for security challenges in the Internet of Things,” *Digital Communications and Networks*, vol. 4, no. 2, pp. 118–137, 2018. [Online]. Available: https://doi.org/10.1016/j.dcan.2017.04.003

[19] S. R. Shanmugham and S. Paramasivam, “Survey on power analysis attacks and its impact on intelligent sensor networks,” *IET Wireless Sensor Systems*, vol. 8, no. 6, pp. 295–304, 2018.

[20] S. Babar *et al.*, “Proposed security model and threat taxonomy for the internet of things (iot),” in *International Conference on Network Security and Applications*. Springer, 2010, pp. 420–429.

[21] S. Raza, L. Wallgren, and T. Voigt, “SVELTE: Real-time intrusion detection in the Internet of Things,” *Ad Hoc Networks*, vol. 11, no. 8, pp. 2661–2674, 2013. [Online]. Available: http://dx.doi.org/10.1016/j.adhoc.2013.04.014

[22] J. Dromard, G. Roudière, and P. Owezarski, “Online and scalable unsupervised network anomaly detection method,” *IEEE Transactions on Network and Service Management*, vol. 14, no. 1, pp. 34–47, 2016.

[23] B. B. Zarpelão *et al.*, “A survey of intrusion detection in Internet of Things,” *Journal of Network and Computer Applications*, vol. 84, pp. 25–37, 2017. [Online]. Available: http://dx.doi.org/10.1016/j.jnca.2017.02.009

[24] F. Hussain *et al.*, "Machine Learning in IoT Security: Current Solutions and Future Challenges," pp. 1–23, 2019. [Online]. Available: http://arxiv.org/abs/1904.05735

[25] T. Anantvalee and J. Wu, "A Survey on Intrusion Detection in Mobile Ad Hoc Networks," *Wireless Network Security*, pp. 159–180, 2007.

[26] L. Apvrille, Y. Roudier, and T. J. Tanzi, "Autonomous drones for disasters management: Safety and security verifications," in *2015 1st URSI Atlantic Radio Science Conference (URSI AT-RASC)*, May 2015, pp. 1–2.

[27] E. Benkhelifa, T. Welsh, and W. Hamouda, "A critical review of practices and challenges in intrusion detection systems for iot: Toward universal and resilient systems," *IEEE Communications Surveys Tutorials*, vol. 20, no. 4, pp. 3496–3509, Fourthquarter 2018.

[28] A. Damien *et al.*, "Anomaly Based Intrusion Detection for an Avionic Embedded System," *SAE Technical Paper Series*, vol. 1, pp. 1–12, 2018.

[29] S. Raponi, M. Caprolu, and R. D. Pietro, *Edge Computing EDGE 2019.* Springer International Publishing, 2019, vol. 11520. [Online]. Available: http://link.springer.com/10.1007/978-3-030-23374-7

[30] E. A. Shams and A. Rizaner, "A novel support vector machine based intrusion detection system for mobile ad hoc networks," *Wireless Networks*, vol. 24, no. 5, pp. 1821–1829, 2018.

[31] S. Ahmad *et al.*, "Unsupervised real-time anomaly detection for streaming data," *Neurocomputing*, vol. 262, pp. 134–147, 2017.

[32] S. Lu, M. Seo, and R. Lysecky, "Timing-based anomaly detection in embedded systems," *20th Asia and South Pacific Design Automation Conference, ASP-DAC 2015*, no. October, pp. 809–814, 2015.

[33] H. Huang *et al.*, "A fast online sequential learning accelerator for iot network intrusion detection: Work-in-progress," in *Proceedings of the Twelfth IEEE/ACM/IFIP International Conference on Hardware/Software Codesign and System Synthesis Companion*, ser. CODES '17. New York, NY, USA: ACM, 2017, pp. 18:1–18:2. [Online]. Available: http://doi.acm.org/10.1145/3125502.3125532

[34] F. M. Tabrizi and K. Pattabiraman, "Flexible Intrusion Detection Systems for Memory-Constrained Embedded Systems," *Proceedings - 2015 11th European Dependable Computing Conference, EDCC 2015*, pp. 1–12, 2016.

[35] M. Ahmed, A. N. Mahmood, and J. Hu, "A survey of network anomaly detection techniques," *Journal of Network and Computer Applications*, vol. 60, pp. 19 – 31, 2016. [Online]. Available: http://www.sciencedirect.com/science/article/pii/S1084804515002891

[36] R. Sekar *et al.*, "Specification-based anomaly detection: a new approach for detecting network intrusions," in *Proceedings of the 9th ACM conference on Computer and communications security*. ACM, 2002, pp. 265–274.

[37] C. F. Tsai *et al.*, "Intrusion detection by machine learning: A review," *Expert Systems with Applications*, vol. 36, no. 10, pp. 11 994–12 000, 2009. [Online]. Available: http://dx.doi.org/10.1016/j.eswa.2009.05.029

[38] N. Chaabouni *et al.*, "Network Intrusion Detection for IoT Security based on Learning Techniques," *IEEE Communications Surveys & Tutorials*, vol. PP, no. 0, pp. 1–1, 2019.

[39] K. Gokcesu *et al.*, "Sequential outlier detection based on incremental decision trees," *IEEE Transactions on Signal Processing*, vol. 67, no. 4, pp. 993–1005, Feb 2019.

[40] J. D. Cannady, "Artificial neural networks for misuse detection," *Proceedings of the 21st National information systems security conference*, pp. 368–381, 1998.

[41] M. Sabhnani and G. Serpen, "Application of machine learning algorithms to kdd intrusion detection dataset within misuse detection context." in *MLMTA*, 2003, pp. 209–215.

[42] M. Mazzoleni, Y. Maccarana, and F. Previdi, "A comparison of data-driven fault detection methods with application to aerospace electro-mechanical actuators," *IFAC-PapersOnLine*, vol. 50, no. 1, pp. 12 797–12 802, 2017. [Online]. Available: https://doi.org/10.1016/j.ifacol.2017.08.1837

[43] E. Boser *et al.*, "Training Algorithm Margin for Optimal Classifiers," *Annual Workshop on Computational Learning Theory*, pp. 144 – 152, 1992.

[44] L. M. Manevitz and M. Yousef, "One-class svms for document classification," *Journal of machine Learning research*, vol. 2, no. Dec, pp. 139–154, 2001.

[45] K.-L. Li *et al.*, "Improving one-class svm for anomaly detection," in *Proceedings of the 2003 International Conference on Machine Learning and Cybernetics (IEEE Cat. No. 03EX693)*, vol. 5. IEEE, 2003, pp. 3077–3081.

[46] S. Henningsen, S. Dietzel, and B. Scheuermann, "Misbehavior detection in industrial wireless networks: challenges and directions," *Mobile Networks and Applications*, vol. 23, no. 5, pp. 1330–1336, 2018.

[47] Z. Ghafoori *et al.*, "Efficient Unsupervised Parameter Estimation for One-Class Support Vector Machines," *IEEE Transactions on Neural Networks and Learning Systems*, vol. 29, no. 10, pp. 5057–5070, 2018.

[48] P. P. Xi *et al.*, "Least squares support vector machine for class imbalance learning and their applications to fault detection of aircraft engine," *Aerospace Science and Technology*, vol. 84, pp. 56–74, 2019. [Online]. Available: https://doi.org/10.1016/j.ast.2018.08.042

[49] E. Hodo *et al.*, "Threat analysis of IoT networks using artificial neural network intrusion detection system," *2016 International Symposium on Networks, Computers and Communications, ISNCC 2016*, pp. 1–6, 2016.

[50] C.-C. Chang and C.-J. Lin, "Libsvm: A library for support vector machines," *ACM transactions on intelligent systems and technology (TIST)*, vol. 2, no. 3, pp. 1–27, 2011.

[51] K. Hundman *et al.*, "Detecting spacecraft anomalies using lstms and nonparametric dynamic thresholding," in *Proceedings of the 24th ACM SIGKDD International Conference on Knowledge Discovery & Data Mining.* ACM, 2018, pp. 387–395.

[52] N. Moustafa and J. Slay, "The evaluation of Network Anomaly Detection Systems: Statistical analysis of the UNSW-NB15 data set and the comparison with the KDD99 data set," *Information Security Journal*, vol. 25, no. 1-3, pp. 18–31, 2016.

[53] A. Jindal *et al.*, "Decision Tree and SVM-Based Data Analytics for Theft Detection in Smart Grid," *IEEE Transactions on Industrial Informatics*, vol. 12, no. 3, pp. 1005–1016, 2016.

[54] S. A. Hofmeyr, S. Forrest, and A. Somayaji, "Intrusion detection using sequences of system calls," *Journal of computer security*, vol. 6, no. 3, pp. 151–180, 1998.

[55] Z. Lin, Y. Shi, and Z. Xue, "IDSGAN: Generative Adversarial Networks for Attack Generation against Intrusion Detection," 2018. [Online]. Available: http://arxiv.org/abs/1809.02077

[56] W. Ma *et al.*, "Shadow attacks: Automatically evading system-call-behavior based malware detection," *Journal in Computer Virology*, vol. 8, no. 1-2, pp. 1–13, 2012.

[57] F. Cohen, "50 ways to defeat IDS," 1997. [Online]. Available: http://all.net/Analyst/netsec/1997-12.html

[58] K. Eykholt *et al.*, "Robust physical-world attacks on deep learning models," *arXiv preprint arXiv:1707.08945*, 2017.

[59] M. Lecuyer *et al.*, "Certified Robustness to Adversarial Examples with Differential Privacy," 2018. [Online]. Available: http://arxiv.org/abs/1802.03471

[60] M. Brückner, T. Scheffer, and C. Kanzow, "Static prediction games for adversarial learning problems," *Journal of Machine Learning Research*, vol. 13, pp. 2617–2654, 2012.

[61] H. G. Kayacik and A. N. Zincir-Heywood, "Mimicry attacks demystified: What can attackers do to evade detection?" in *2008 Sixth Annual Conference on Privacy, Security and Trust*. IEEE, 2008, pp. 213–223.

[62] J. E. Tapiador and J. A. Clark, "Masquerade mimicry attack detection: A randomised approach," *computers & security*, vol. 30, no. 5, pp. 297–310, 2011.

[63] K. Goeschel, "Reducing false positives in intrusion detection systems using data-mining techniques utilizing support vector machines, decision trees, and naive Bayes for off-line analysis," *Conference Proceedings - IEEE SOUTHEASTCON*, vol. 2016-July, 2016.

[64] Z. Zohrevand and U. Glässer, "Should I Raise The Red Flag? A comprehensive survey of anomaly scoring methods toward mitigating false alarms," 2019. [Online]. Available: http://arxiv.org/abs/1904.06646

[65] W. Meng *et al.*, "When intrusion detection meets blockchain technology: A review," *IEEE Access*, vol. 6, pp. 10 179–10 188, 2018.

[66] P. Kannadiga and M. Zulkernine, "DIDMA: A distributed intrusion detection system using mobile agents," *Proceedings - Sixth Int. Conf. on Softw. Eng., Artificial Intelligence, Netw. and Parallel/Distributed Computing and First ACIS Int. Workshop on Self-Assembling Wireless Netw., SNPD/SAWN 2005*, vol. 2005, pp. 238–245, 2005.

[67] S. Stolfo *et al.*, "Cost-based modeling and evaluation for data mining with application to fraud and intrusion detection: Results from the jam project," 09 1999.

[68] R. Gassais, "Détection d'intrusion sur les objets connectés par analyse comportementale," 2018. [Online]. Available: https://publications.polymtl.ca/3209/

[69] A. Karygiannis and K. Robotis, "Creating Offline MANET IDS," 2007.

[70] F.-Y. Wang *et al.*, “Where does alphago go: From church-turing thesis to alphago thesis and beyond,” *IEEE/CAA Journal of Automatica Sinica*, vol. 3, no. 2, pp. 113–120, 2016.

[71] O. Vinyals *et al.*, “Alphastar: Mastering the real-time strategy game starcraft ii,” *DeepMind Blog*, 2019.

[72] D. Gunning, “Explainable Artificial Intelligence (XAI),” Tech. Rep., 2017. [Online]. Available: https://www.cc.gatech.edu/~alanwags/DLAI2016/(Gunning)%20IJCAI-16%20DLAI%20WS.pdf

[73] Z. C. Lipton, “The mythos of model interpretability,” *Communications of the ACM*, vol. 61, no. 10, pp. 36–43, sep 2018. [Online]. Available: http://dl.acm.org/citation.cfm?doid=3281635.3233231

[74] D. Doran, S. Schulz, and T. R. Besold, “What does explainable AI really mean? A new conceptualization of perspectives,” in *CEUR Workshop Proceedings*, vol. 2071. CEUR-WS, 2018.

[75] F. K. Dosilovic, M. Brcic, and N. Hlupic, “Explainable artificial intelligence: A survey,” in *2018 41st International Convention on Information and Communication Technology, Electronics and Microelectronics, MIPRO 2018 - Proceedings.* Institute of Electrical and Electronics Engineers Inc., jun 2018, pp. 210–215.

[76] A. Adadi and M. Berrada, “Peeking inside the black-box: A survey on explainable artificial intelligence (xai),” *IEEE Access*, vol. 6, pp. 52 138–52 160, 2018.

[77] D. Baehrens, “How to Explain Individual Classification Decisions,” 2010.

[78] D. L. Marino, C. S. Wickramasinghe, and M. Manic, “An adversarial approach for explainable AI in intrusion detection systems,” *Proceedings: IECON 2018 - 44th Annual Conference of the IEEE Industrial Electronics Society*, vol. 1, pp. 3237–3243, 2018.

[79] M. T. Ribeiro, S. Singh, and C. Guestrin, “"Why Should I Trust You?",” in *Proceedings of the 22nd ACM SIGKDD International Conference on Knowledge Discovery and Data Mining - KDD ’16.* New York, New York, USA: ACM Press, 2016, pp. 1135–1144. [Online]. Available: http://dl.acm.org/citation.cfm?doid=2939672.2939778

[80] M. Wang *et al.*, “An explainable machine learning framework for intrusion detection systems,” *IEEE Access*, vol. 8, pp. 73 127–73 141, 2020.

[81] K. Amarasinghe, K. Kenney, and M. Manic, "Toward explainable deep neural network based anomaly detection," in *2018 11th International Conference on Human System Interaction (HSI).* IEEE, 2018, pp. 311–317.

[82] B. E. Boser, I. M. Guyon, and V. N. Vapnik, "A training algorithm for optimal margin classifiers," in *Proceedings of the fifth annual workshop on Computational learning theory*, 1992, pp. 144–152.

[83] C. Molnar, *Interpretable Machine Learning*, 2019, https://christophm.github.io/interpretable-ml-book/.

[84] S. Msika, A. Quintero, and F. Khomh, "Sigma: Strengthening ids with gan and metaheuristics attacks," *arXiv preprint arXiv:1912.09303*, 2019.

[85] F. Pedregosa *et al.*, "Scikit-learn: Machine learning in python," *the Journal of machine Learning research*, vol. 12, pp. 2825–2830, 2011.

[86] R. A. Calix and R. Sankaran, "Feature ranking and support vector machines classification analysis of the nsl-kdd intrusion detection corpus," in *The Twenty-Sixth International FLAIRS Conference*, 2013.